Version 16

JMP Documentation Library

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.2

All JMP® 16 books are included in this document.

The correct bibliographic citation for this document is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Documentation Library*. Cary, NC: SAS Institute Inc.

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

March 2021

August 2021

December 2021

Version 16

Discovering JMP

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *Discovering JMP® 16*. Cary, NC: SAS Institute Inc.

Discovering JMP® 16

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Discovering JMP

About This Book 9

Gallery of JMP Graphs 11

1 Learn about JMP 33

Documentation and Additional Resources

Formatting Conventions in JMP Documentation 35
JMP Help 36
JMP Documentation Library 36
Additional Resources for Learning JMP 42
JMP Tutorials 42
Sample Data Tables 42
Learn about Statistical and JSL Terms 43
Learn JMP Tips and Tricks 43
JMP Tooltips 43
JMP User Community 44
Free Online Statistical Thinking Course 44
JMP New User Welcome Kit 44
Statistics Knowledge Portal 44
JMP Training 44
JMP Books by Users 45
The JMP Starter Window 45
JMP Technical Support 45

2 Introducing JMP 47

Basic Concepts

JMP Concepts That You Should Know 49
How Do I Get Started with JMP? 49
Starting JMP 50
Using Sample Data 52
Understand Data Tables 53
Understand the JMP Workflow 54
Step 1: Launch a JMP Platform and View Results 55
Step 2: Remove the Box Plot from a JMP Report 57

Step 3: Request Additional JMP Output ... 57
Step 4: Interact with JMP Platform Results ... 58
How is JMP Different from Excel? ... 59
Structure of a Data Table ... 59
Formulas in JMP ... 60
JMP Analysis and Graphing ... 61

3 Work with Your Data ... 63
Prepare Your Data for Graphing and Analyzing
Get Your Data into JMP ... 65
Copy and Paste Data into a Data Table ... 65
Import Data into a Data Table ... 65
Enter Data in a Data Table ... 68
Transfer Data from Excel to JMP ... 70
Work with Data Tables ... 72
Edit Data in a Data Table ... 72
Select, Deselect, and Find Values in a Data Table ... 74
View or Change Column Information in a Data Table ... 78
Example of Calculating Values with Formulas ... 80
Example of Filtering Data in a Report ... 82
Examples of Reshaping Data ... 83
Examples of Viewing Summary Statistics ... 84
Examples of Creating Subsets ... 88
Example of Joining Data Tables ... 90
Example of Sorting Data ... 92

4 Visualize Your Data ... 95
Common Graphs
Analyze Single Variables in Univariate Graphs ... 97
Use Histograms for Continuous Variables ... 97
Use Bar Charts for Categorical Variables ... 100
Compare Multiple Variables ... 102
Compare Multiple Variables Using Scatterplots ... 103
Compare Multiple Variables Using a Scatterplot Matrix ... 107
Compare Multiple Variables Using Side-by-Side Box Plots ... 110
Compare Multiple Variables Using Graph Builder ... 113
Compare Multiple Variables Using Bubble Plots ... 119
Compare Multiple Variables Using Overlay Plots ... 124
Compare Multiple Variables Using a Variability Chart ... 129

5 Analyze Your Data 133
Distributions, Relationships, and Models

About This Chapter 135
The Importance of Graphing Your Data 135
Understand Modeling Types 138
Example of Viewing Modeling Type Results 139
Change the Modeling Type 141
Analyze Distributions 143
Distributions of Continuous Variables 143
Distributions of Categorical Variables 146
Analyze Relationships 149
Use Regression with One Predictor 149
Compare Averages for One Variable 154
Compare Proportions 157
Compare Averages for Multiple Variables 160
Use Regression with Multiple Predictors 165

6 The Big Picture 171
Exploring Data in Multiple Platforms

Fun Fact: Linked Analyses 173
Explore Data in Multiple Platforms 173
Analyze Distributions in the Distribution Platform 173
Analyze Patterns and Relationships in the Multivariate Platform 177
Analyze Similar Values in the Clustering Platform 181

7 Save and Share Your Work 187
Save and Re-create Results

Work with Projects 189
Create a New Project 190
Open Files in a Project 190
Rearrange Files in Projects 191
Save a Project 192
Project Workspace 193
Project Bookmarks 193
Project Contents 194
Recent Files in Projects 195
Project Log 196
Move Files into Projects 196
Write a Project On Open Script 197
Example of Creating a New Project 197
Save Platform Results in Journals 200

Example of Creating a Journal ... 200
Add Analyses to a Journal ... 201
Save and Run Scripts ... 202
Example of Saving and Running a Script ... 202
About Scripts and JSL ... 203
Save Reports as Interactive HTML ... 204
Interactive HTML Contains Data ... 205
Example of Creating Interactive HTML ... 205
Share Reports as Interactive HTML ... 206
Save a Report as a PowerPoint Presentation ... 210
Create Dashboards ... 211
Example of Combining Windows ... 211
Example of Creating a Dashboard with Two Reports ... 213

8 Special Features ... 215
Automatic Analysis Updates and SAS Integration
Automatically Update Analyses and Graphs ... 217
Example of Using Automatic Recalc ... 217
Change Preferences ... 221
Example of Changing Preferences ... 222
Integrate JMP and SAS ... 224
Example of Creating SAS Code ... 224
Example of Submitting SAS Code ... 225

A Technology Notices ... 227

About This Book

Discovering JMP provides a general introduction to the JMP software. This guide assumes that you have no knowledge of JMP. Whether you are an analyst, researcher, student, professor, or statistician, this guide gives you a general overview of JMP's user interface and features.

This guide introduces you to the following information:

- Starting JMP
- The structure of a JMP window
- Preparing and manipulating data
- Using interactive graphs to learn from your data
- Performing simple analyses to augment graphs
- Customizing JMP and special features
- Sharing your results

This guide contains six chapters. Each chapter contains examples that reinforce the concepts presented in the chapter. All of the statistical concepts are at an introductory level. The sample data used in *Discovering JMP* are included with the software. Here is a description of each chapter:

- "Introducing JMP" provides an overview of the JMP application. You learn how content is organized and how to navigate the software.
- "Work with Your Data" describes how to import data from a variety of sources, and prepare it for analysis. There is also an overview of data manipulation tools.
- "Visualize Your Data" describes graphs and charts that you can use to visualize and understand your data. The examples range from simple analyses involving a single variable, to multiple-variable graphs that enable you to see relationships between many variables.
- "Analyze Your Data" describes many commonly used analysis techniques. These techniques range from simple techniques that do not require the use of statistical methods, to advanced techniques, where knowledge of statistics is useful.
- "The Big Picture" shows you how to analyze distributions, patterns, and similar values in several platforms.

- "Save and Share Your Work" describes sharing your work with non JMP users in PowerPoint presentations and interactive HTML. Saving analyses as scripts and saving work in journals and projects for JMP users are also covered.
- "Special Features" describes how to automatically update graphs and analyses as data changes, how to use preferences to customize your reports, and how JMP interacts with SAS.

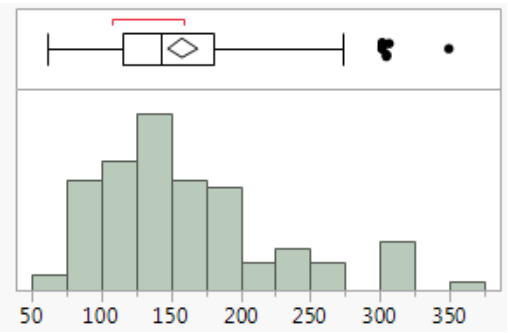
After reviewing this guide, you will be comfortable navigating and working with your data in JMP.

JMP is available for both Windows and macOS operating systems. However, the material in this guide is based on a Windows operating system.

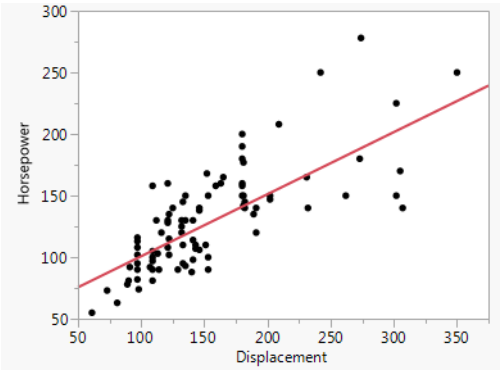
Gallery of JMP Graphs

Various Graphs and Their Platforms

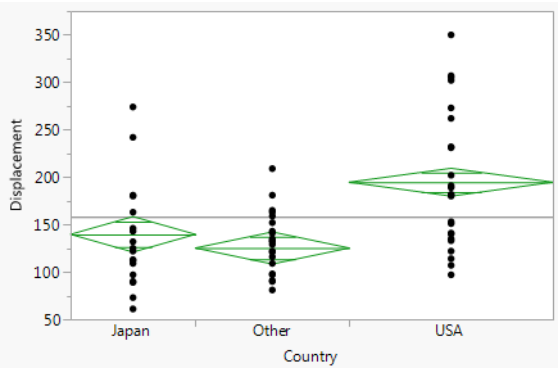
Here are pictures of many of the graphs that you can create with JMP. Each picture is labeled with the platform used to create it. For more information about the platforms and these and other graphs, see the documentation on the **Help > Books** menu.



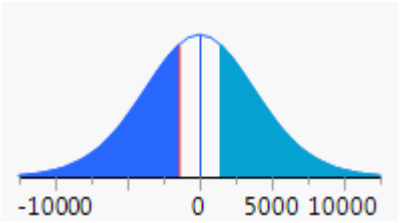
Histogram
Analyze > Distribution



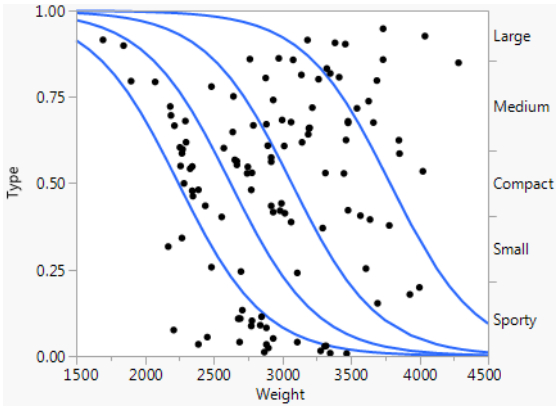
Bivariate
Analyze > Fit Y by X



Oneway
Analyze > Fit Y by X

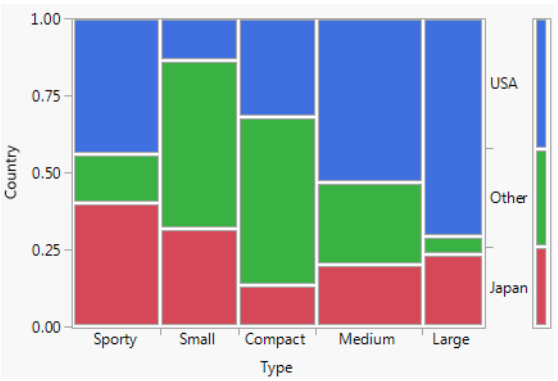


Oneway t Test
Analyze > Fit Y by X



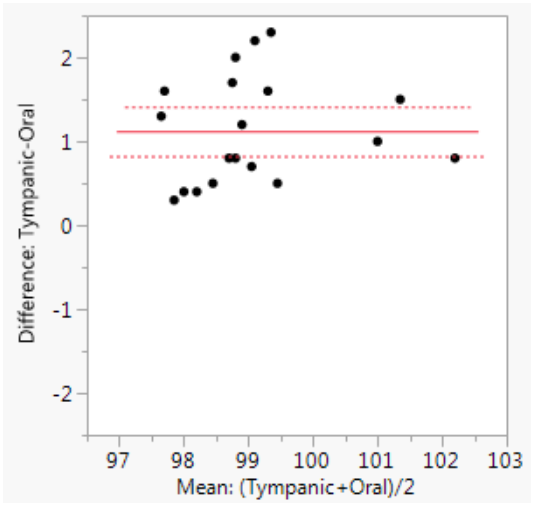
Logistic

Analyze > Fit Y by X



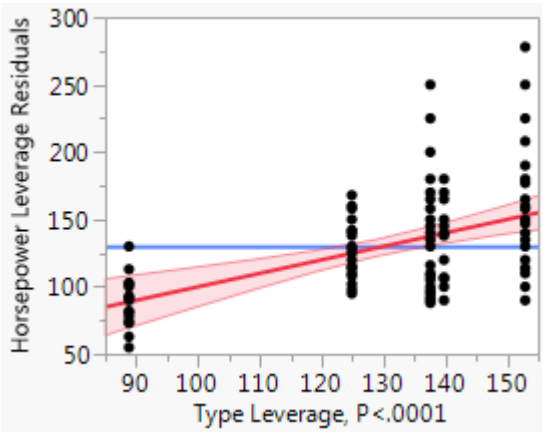
Mosaic Plot

Analyze > Fit Y by X



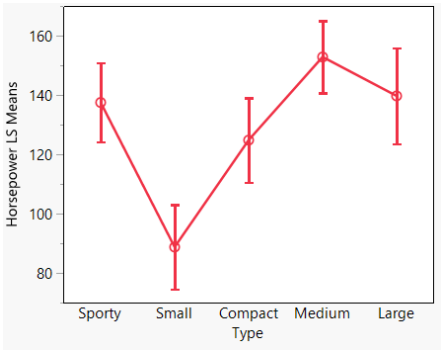
Matched Pairs

Analyze > Specialized Modeling > Matched Pairs

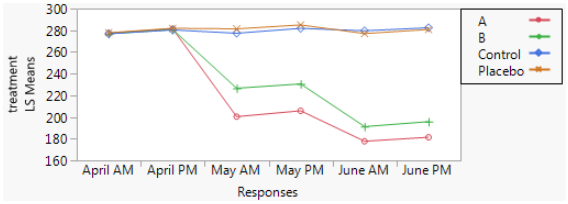


Leverage Plot

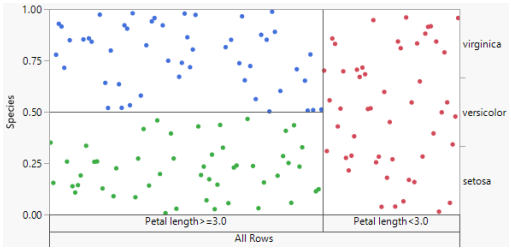
Analyze > Fit Model



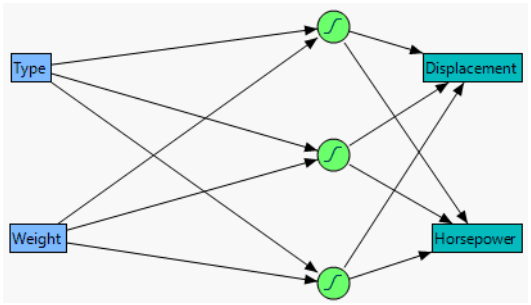
LS Means Plot
Analyze > Fit Model



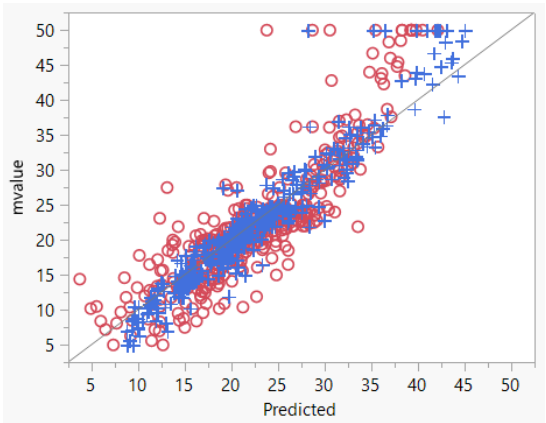
MANOVA
Analyze > Fit Model



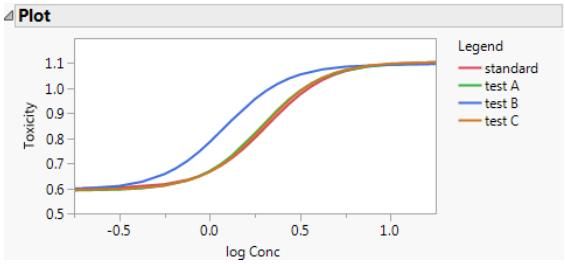
Partition
Analyze > Predictive Modeling > Partition



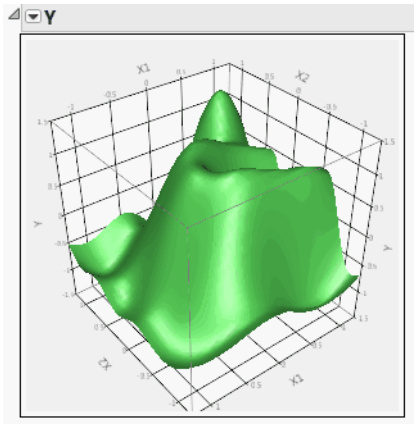
Neural Diagram
Analyze > Predictive Modeling > Neural



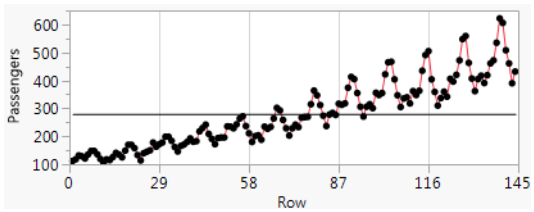
Actual by Predicted
Analyze > Predictive Modeling > Model Comparison



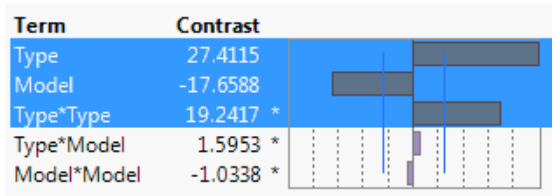
Nonlinear Fit
Analyze > Specialized Modeling > Nonlinear



Surface Profiler
Analyze > Specialized Modeling > Gaussian Process

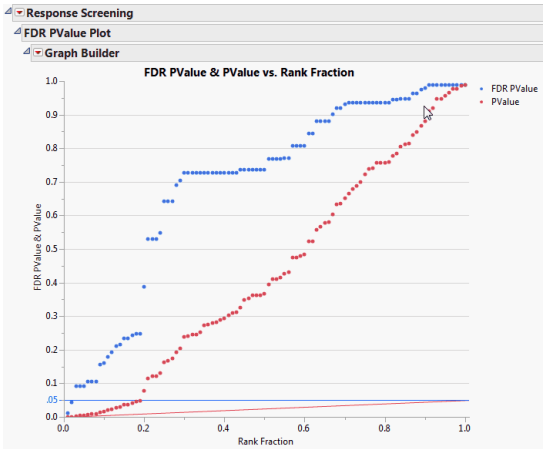


Time Series
Analyze > Specialized Modeling > Time Series



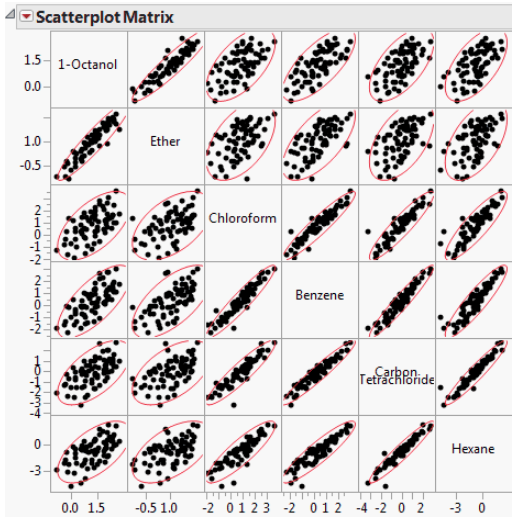
Screening

Analyze > Specialized Modeling > Specialized DOE Models > Fit Two Level Screening



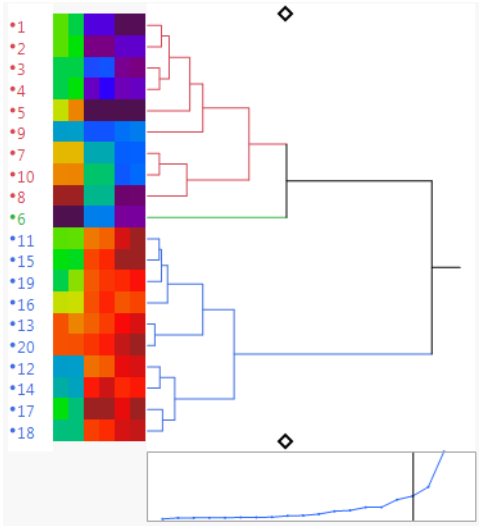
FDR pValue Plot

Analyze > Screening > Response Screening



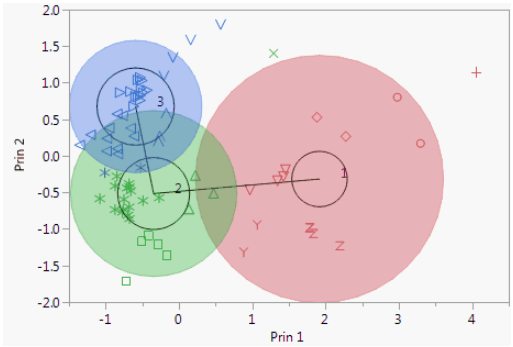
Scatterplot Matrix

Analyze > Multivariate Methods > Multivariate

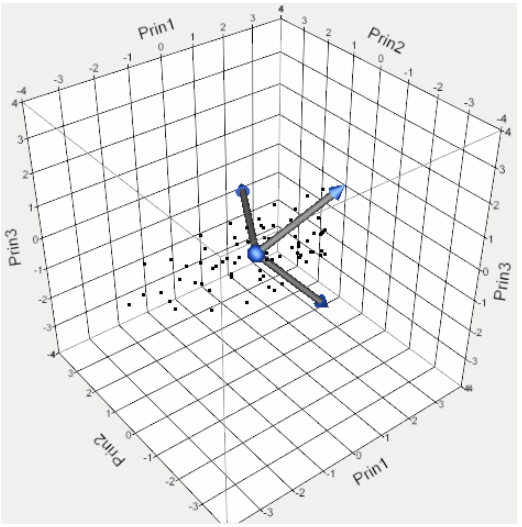


Dendrogram

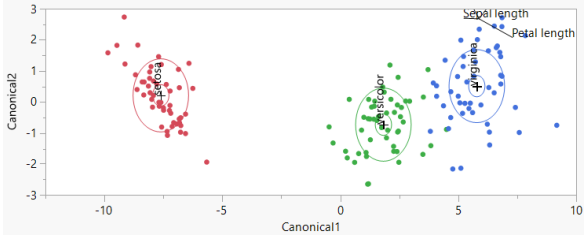
Analyze > Clustering > Hierarchical Cluster



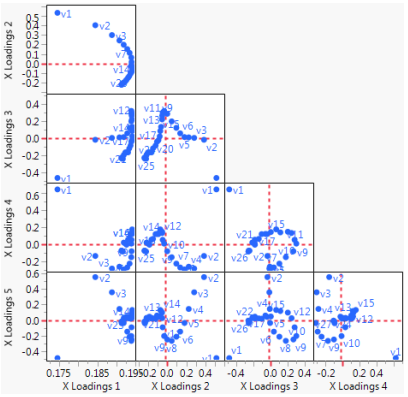
Self Organizing Map
Analyze > Clustering > K Means Cluster



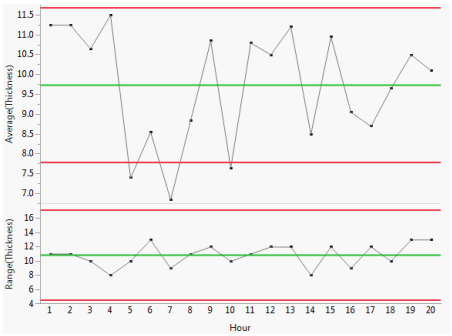
Principal Components
Analyze > Multivariate Methods > Principal Components



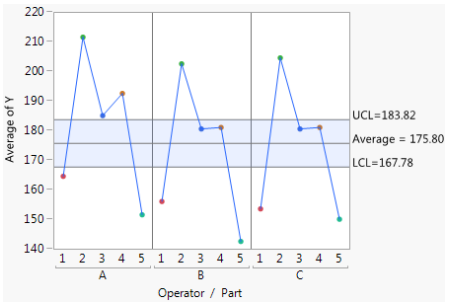
Canonical Plot
Analyze > Multivariate Methods > Discriminant



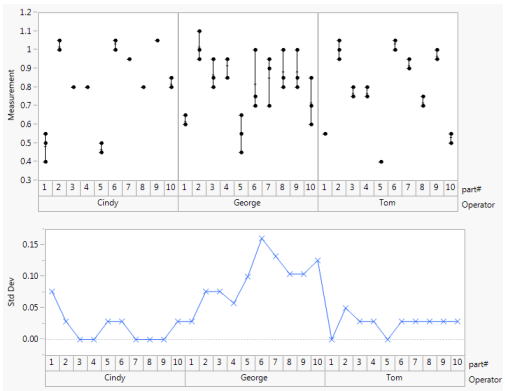
Loadings Plot
Analyze > Multivariate Methods > Partial Least Squares



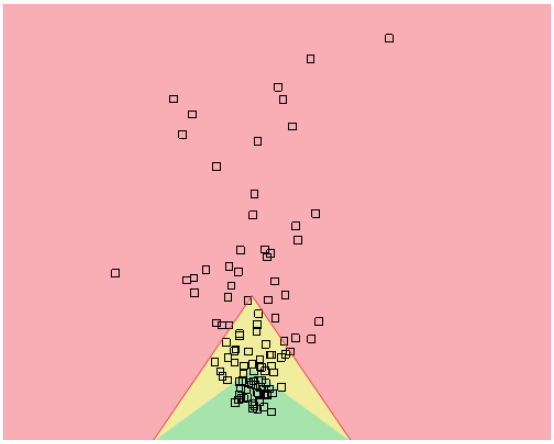
XBar and R Charts
Analyze > Quality and Process > Control Chart Builder



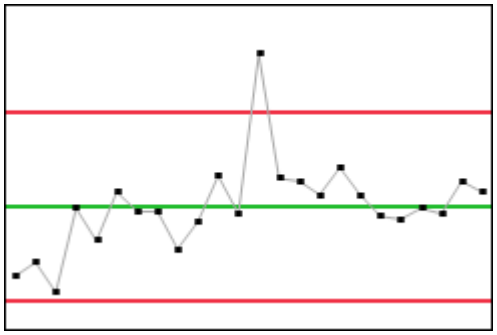
Average Chart
Analyze > Quality and Process > Measurement Systems Analysis



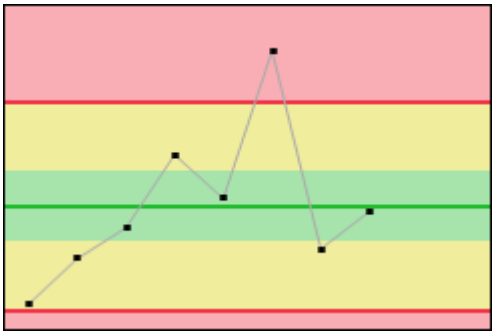
Variability Chart
Analyze > Quality and Process > Variability/Attribute Chart



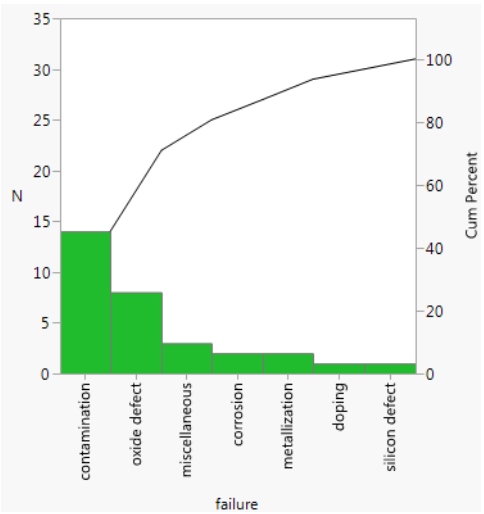
Goal Plot
Analyze > Quality and Process > Process Capability



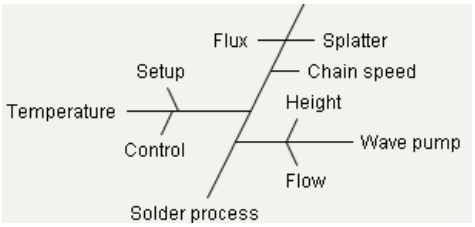
Individual Measurement Chart
Moving Range Chart
Analyze > Quality and Process > Control Chart > IR



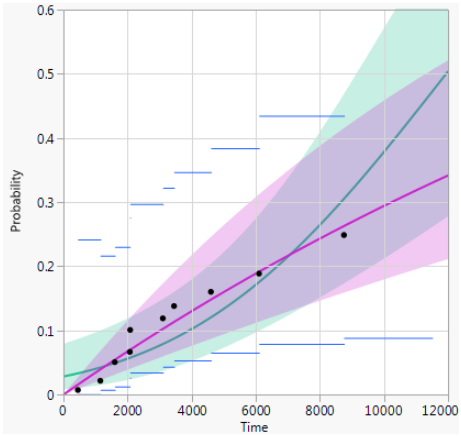
XBar Chart
Analyze > Quality and Process > Control Chart > XBar



Pareto Plot
Analyze > Quality and Process > Pareto Plot

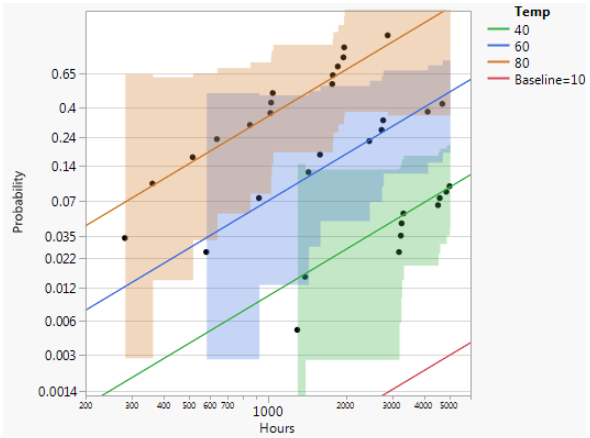


Ishikawa Chart
Fishbone Chart
Analyze > Quality and Process> Diagram



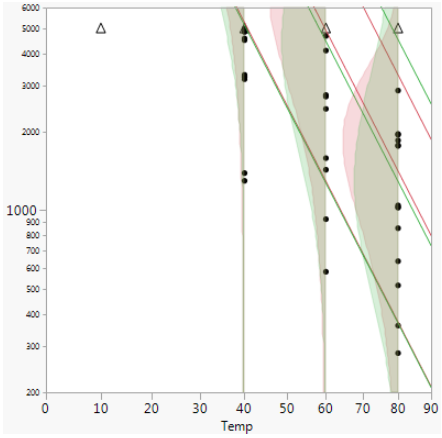
Compare Distributions

Analyze > Reliability and Survival > Life Distribution



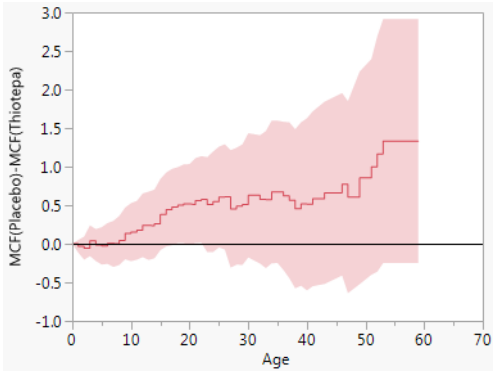
Nonparametric Overlay

Analyze > Reliability and Survival > Fit Life by X



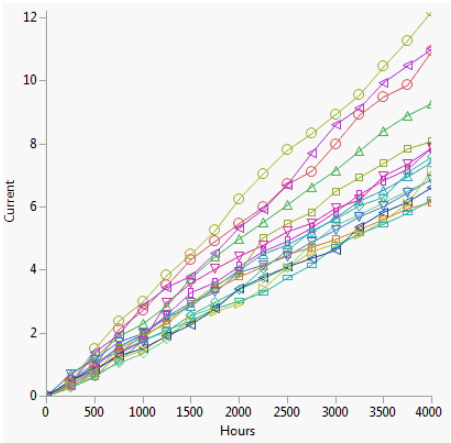
Scatterplot

Analyze > Reliability and Survival > Fit Life by X



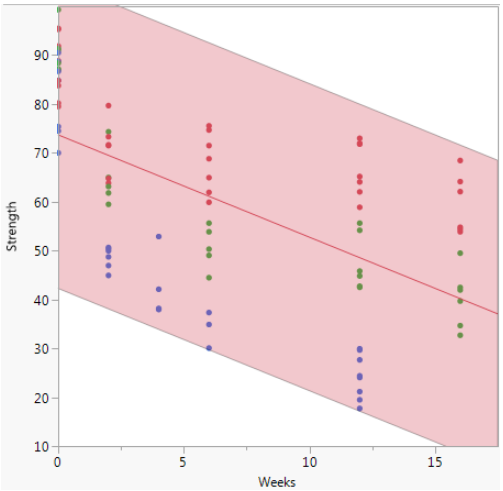
MCF Plot

Analyze > Reliability and Survival > Recurrence Analysis



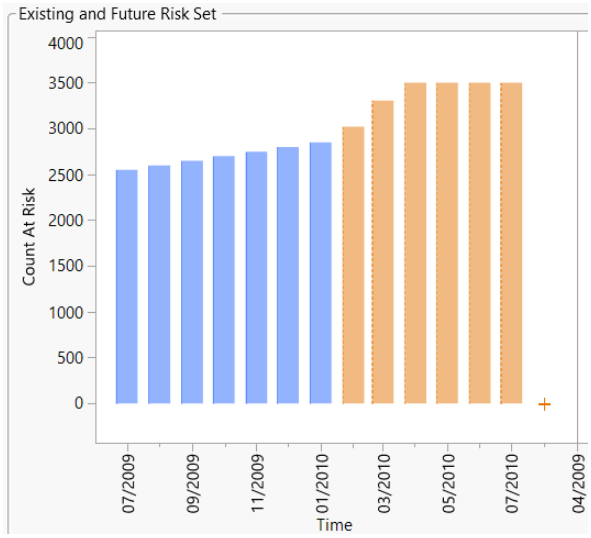
Overlay

Analyze > Reliability and Survival > Degradation



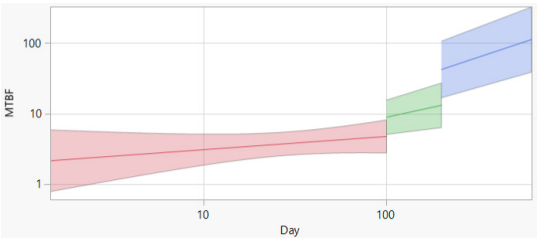
Prediction Interval

Analyze > Reliability and Survival > Destructive Degradation



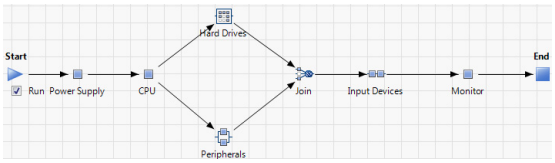
Forecast

Analyze > Reliability and Survival > Reliability Forecast



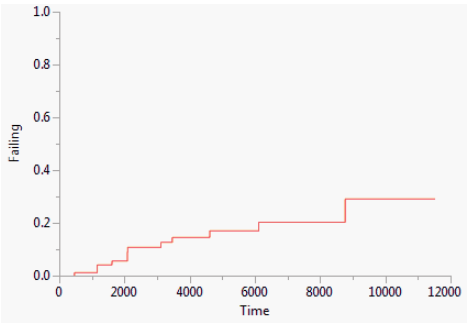
Piecewise Weibull NHPP

Analyze > Reliability and Survival > Reliability Growth



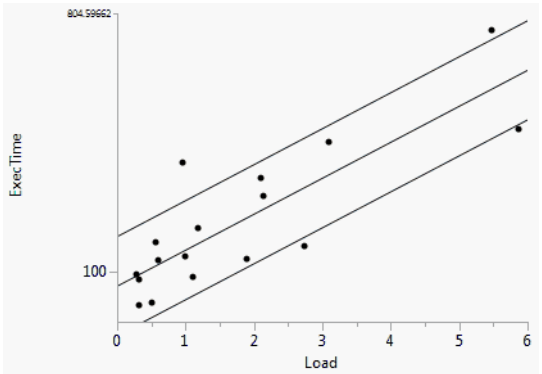
Reliability Block Diagram

Analyze > Reliability and Survival > Reliability Block Diagram



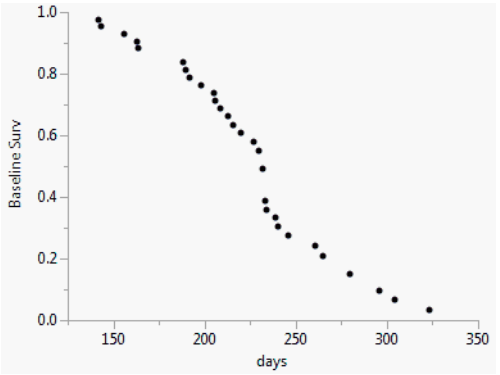
Failure Plot

Analyze > Reliability and Survival > Survival



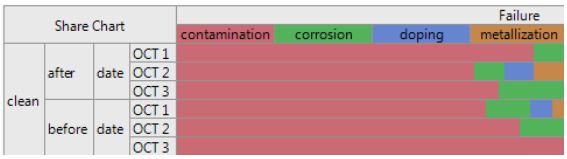
Survival Quantiles

Analyze > Reliability and Survival > Fit Parametric Survival



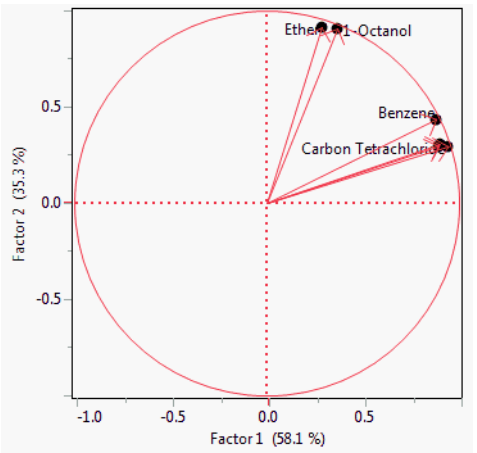
Baseline Survival

Analyze > Reliability and Survival > Fit Proportional Hazards



Mixture Profiler

Analyze > Consumer Research > Categorical

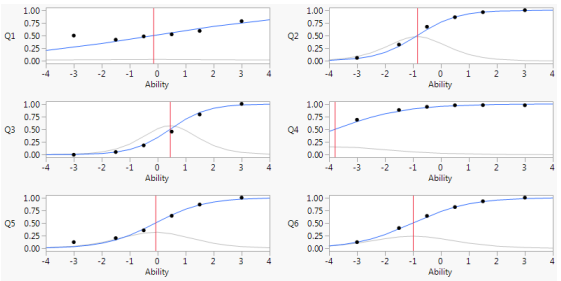


Factor Loading Plot

Analyze > Multivariate Methods > Factor Analysis

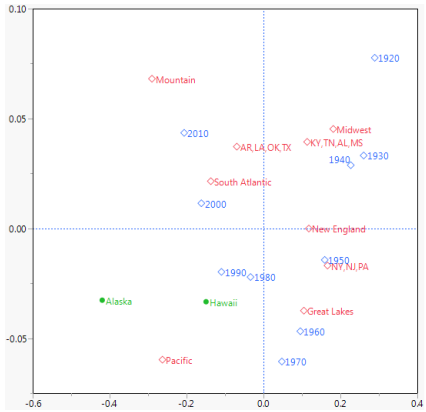
Prediction Profile

Analyze > Consumer Research > Choice



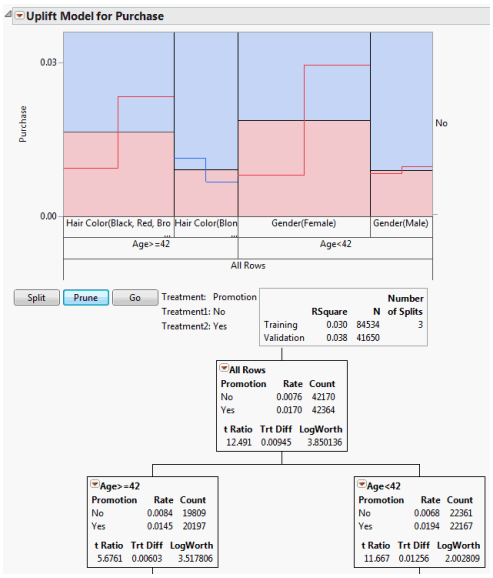
Characteristic Curves

Analyze > Multivariate Methods > Item Analysis



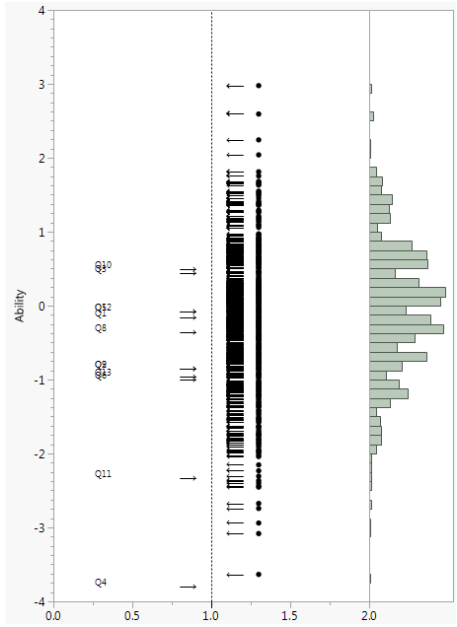
Multiple Correspondence Analysis

Analyze > Multivariate Methods > Multiple Correspondence Analysis



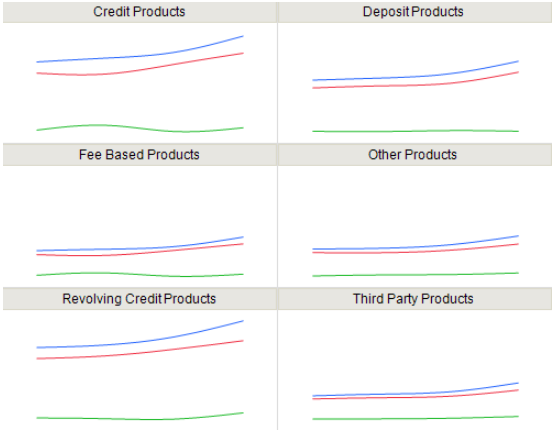
Uplift Model

Analyze > Consumer Research > Uplift



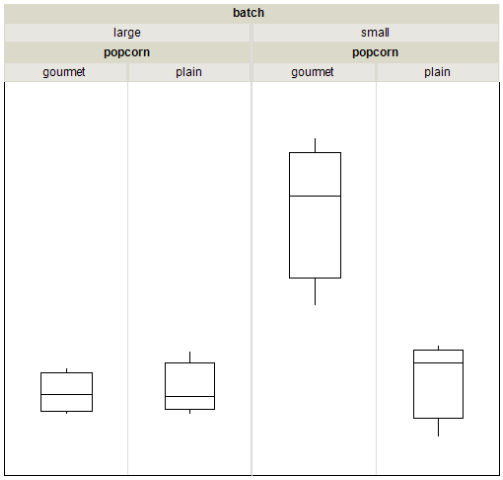
Dual Plot

Analyze > Multivariate Methods > Item Analysis

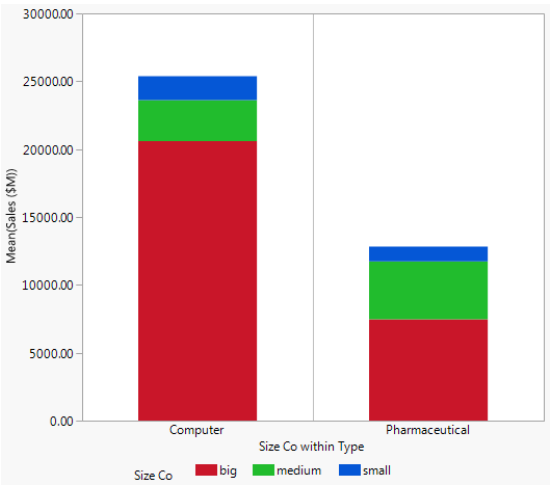


Line Graphs

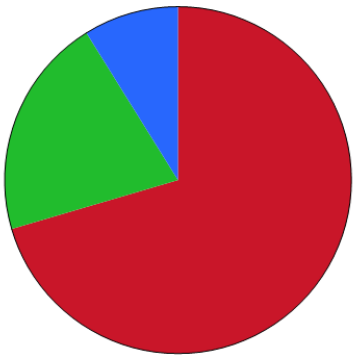
Graph > Graph Builder



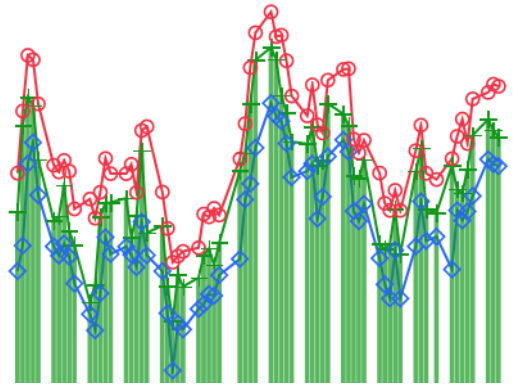
Box Plots
Graph > Graph Builder



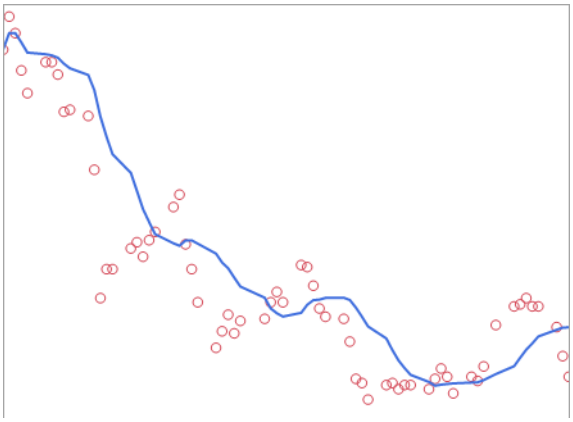
Stacked Bar Chart
Graph > Graph Builder



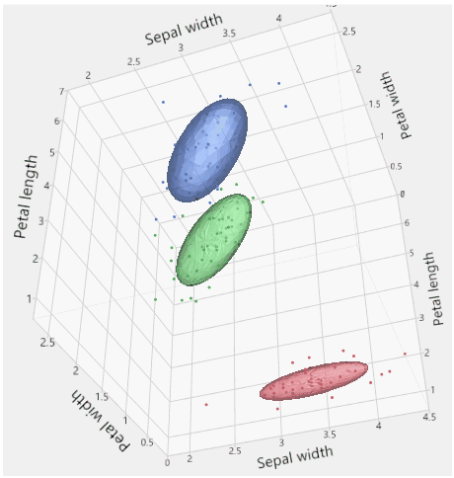
Pie Chart
Graph > Graph Builder



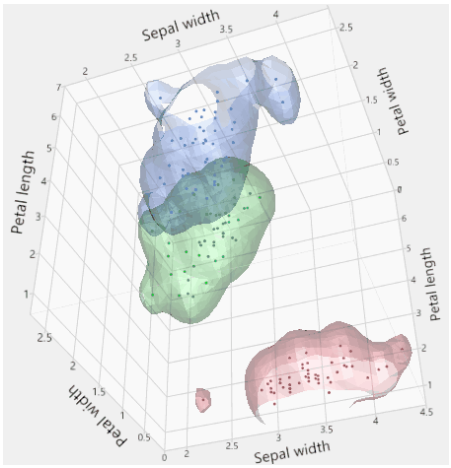
Needle and Line Chart
Graph > Graph Builder



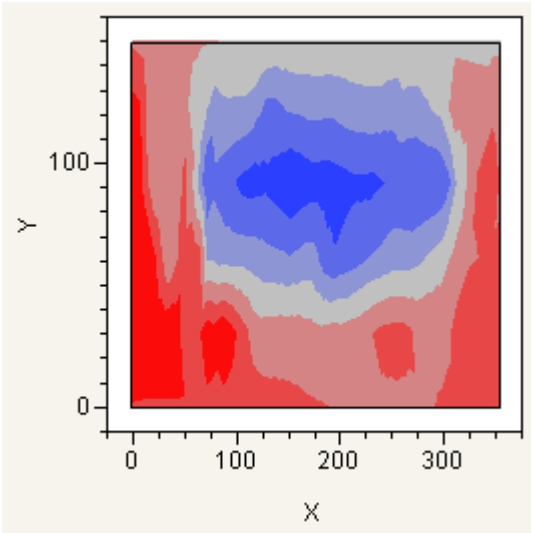
Smoother
Graph > Graph Builder



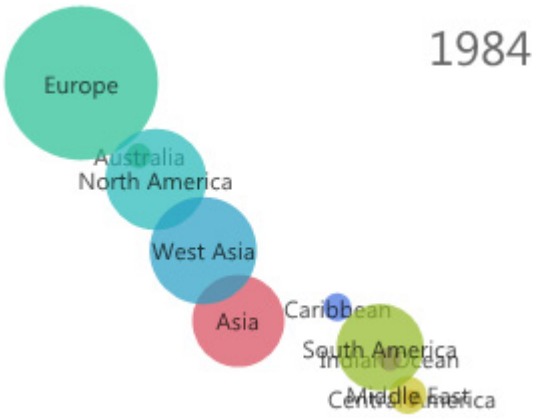
Three Dimensional Scatterplot
Graph > Scatterplot 3D



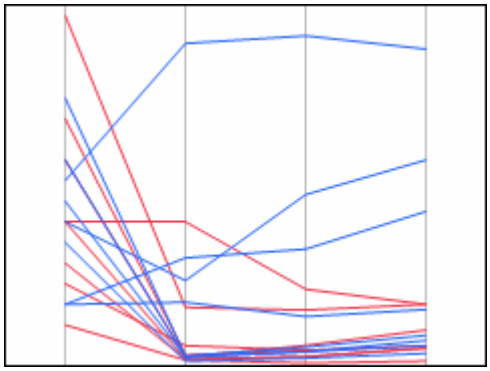
Three Dimensional Scatterplot
Graph > Scatterplot 3D



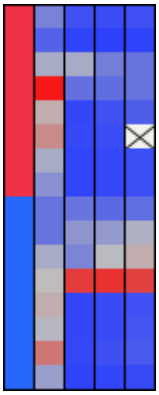
Contour Plot
Graph > Graph Builder



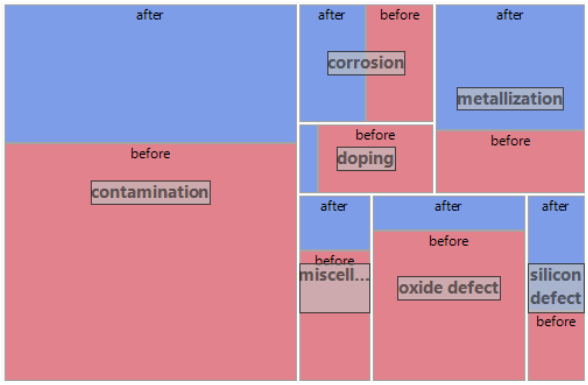
Bubble Plot
Graph > Bubble Plot



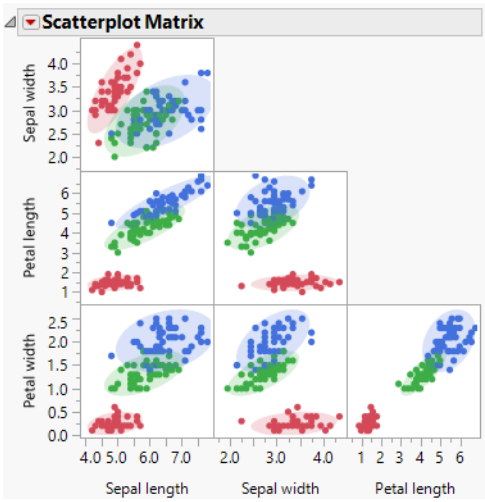
Parallel Plot
Graph > Graph Builder



Cell Plot
Graph > Cell Plot

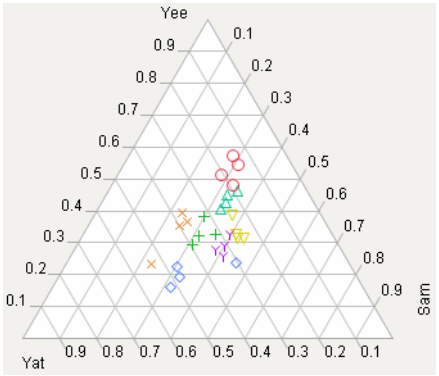


Treemap
Graph > Graph Builder



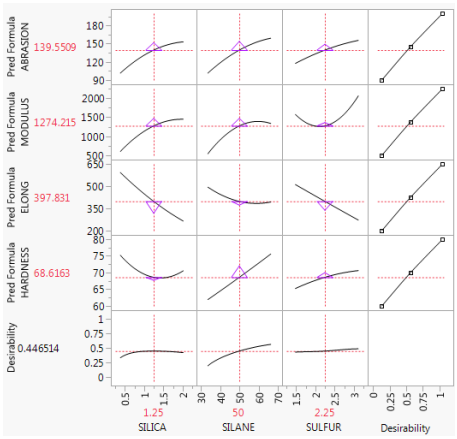
Scatterplot Matrix

Graph > Scatterplot Matrix



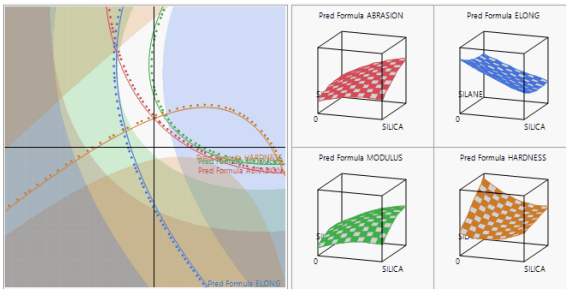
Ternary Plot

Graph > Ternary Plot



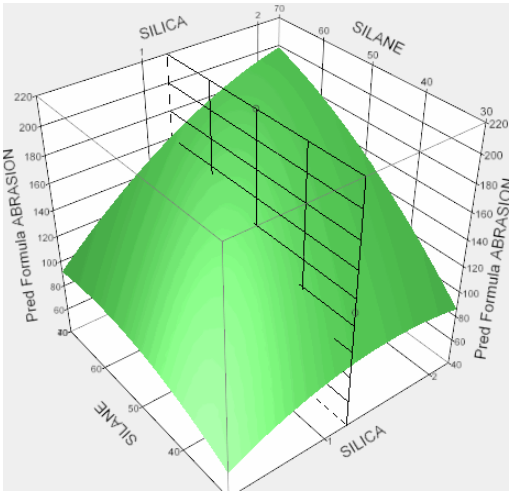
Prediction Profiler

Graph > Profiler



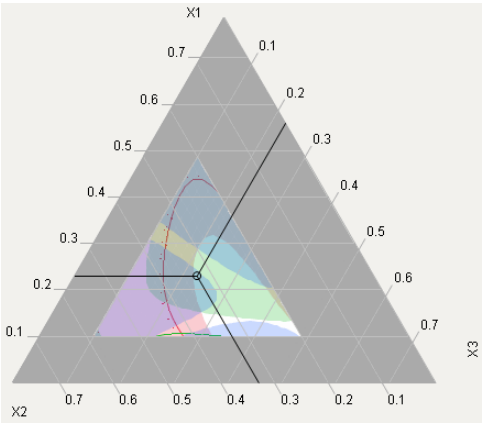
Contour Profiler

Graph > Contour Profiler



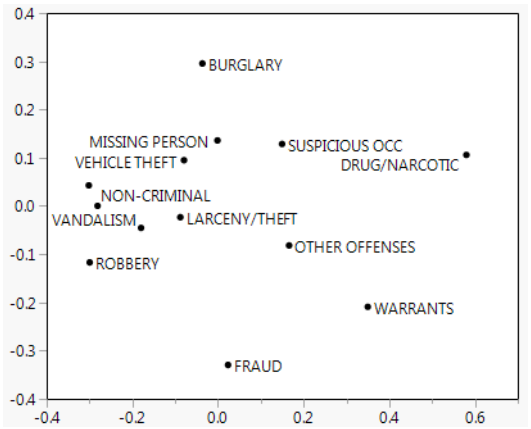
Surface Plot

Graph > Surface Plot



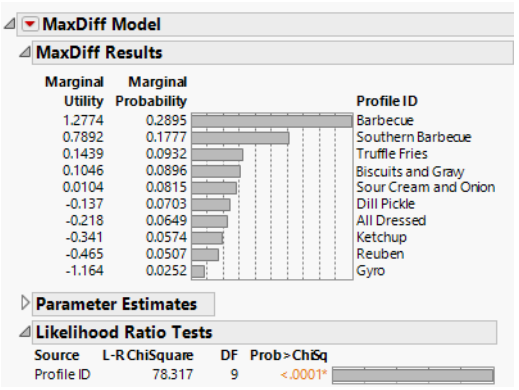
Mixture Profiler

Graph > Mixture Profiler



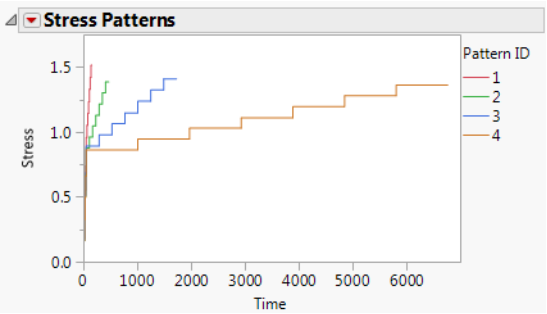
Multidimensional Scaling

Analyze > Multivariate Methods > Multidimensional Scaling



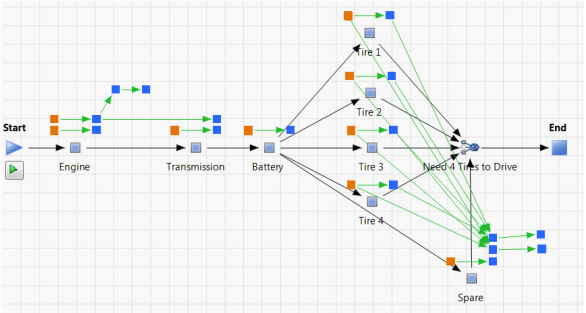
MaxDiff

Analyze > Consumer Research > MaxDiff



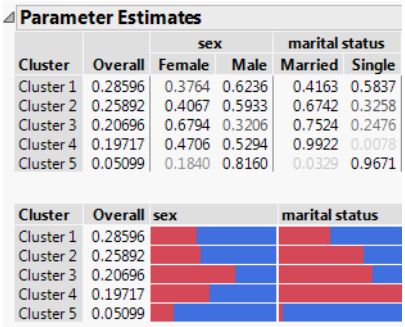
Stress Patterns Plot

Analyze > Reliability and Survival > Cumulative Damage



Repairable Systems Simulation

Analyze > Reliability and Survival > Repairable Systems Simulation



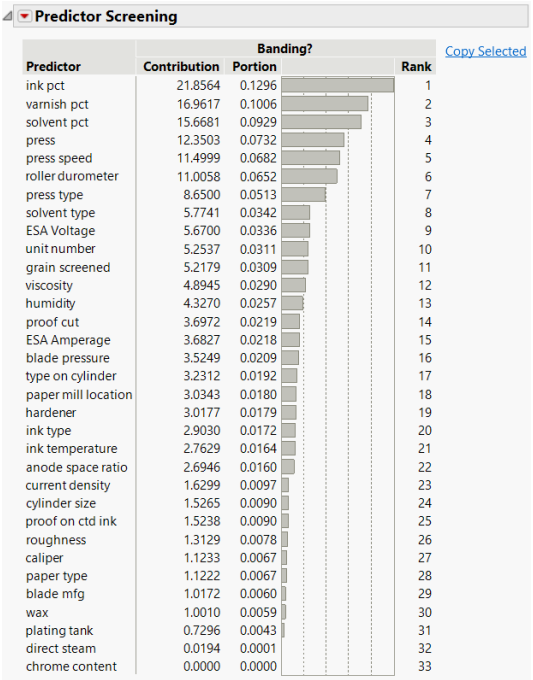
Parameter Estimates

Cluster	Overall	sex		marital status	
		Female	Male	Married	Single
Cluster 1	0.28596	0.3764	0.6236	0.4163	0.5837
Cluster 2	0.25892	0.4067	0.5933	0.6742	0.3258
Cluster 3	0.20696	0.6794	0.3206	0.7524	0.2476
Cluster 4	0.19717	0.4706	0.5294	0.9922	0.0078
Cluster 5	0.05099	0.1840	0.8160	0.0329	0.9671

Cluster	Overall	sex	marital status
Cluster 1	0.28596		
Cluster 2	0.25892		
Cluster 3	0.20696		
Cluster 4	0.19717		
Cluster 5	0.05099		

Latent Class Analysis

Analyze > Clustering > Latent Class Analysis



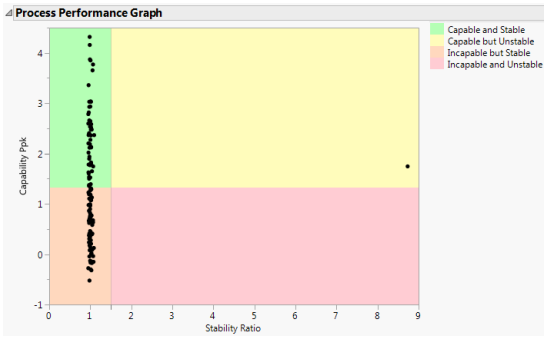
Predictor Screening

Copy Selected

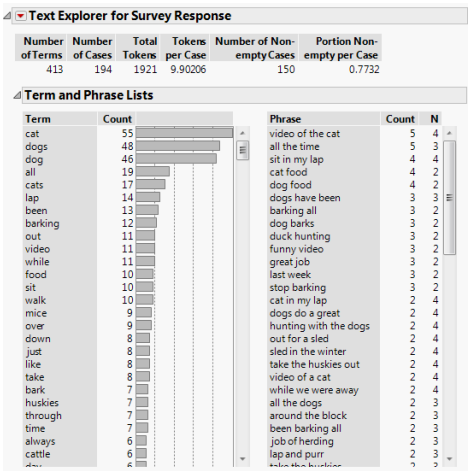
Predictor	Contribution	Banding? Portion	Rank
ink pct	21.8564	0.1296	1
varnish pct	16.9617	0.1006	2
solvent pct	15.6681	0.0929	3
press	12.3503	0.0732	4
press speed	11.4999	0.0682	5
roller durometer	11.0058	0.0652	6
press type	8.6500	0.0513	7
solvent type	5.7741	0.0342	8
ESA Voltage	5.6700	0.0336	9
unit number	5.2537	0.0311	10
grain screened	5.2179	0.0309	11
viscosity	4.8945	0.0290	12
humidity	4.3270	0.0257	13
proof cut	3.6972	0.0219	14
ESA Amperage	3.6827	0.0218	15
blade pressure	3.5249	0.0209	16
type on cylinder	3.2312	0.0192	17
paper mill location	3.0343	0.0180	18
hardener	3.0177	0.0179	19
ink type	2.9030	0.0172	20
ink temperature	2.7629	0.0164	21
anode space ratio	2.6946	0.0160	22
current density	1.6299	0.0097	23
cylinder size	1.5265	0.0090	24
proof on ctd ink	1.5238	0.0090	25
roughness	1.3129	0.0078	26
caliper	1.1233	0.0067	27
paper type	1.1222	0.0067	28
blade mfg	1.0172	0.0060	29
wax	1.0010	0.0059	30
plating tank	0.7296	0.0043	31
direct steam	0.0194	0.0001	32
chrome content	0.0000	0.0000	33

Predictor Screening

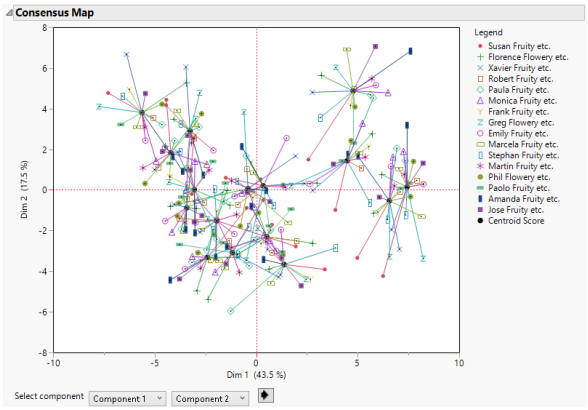
Analyze > Screening > Predictor Screening



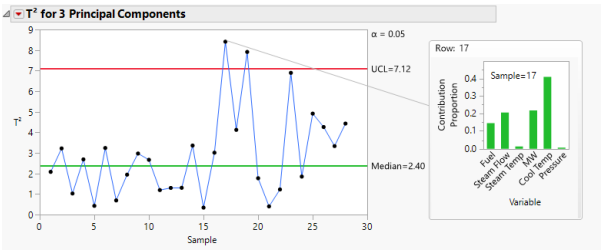
Process Screening
Analyze > Screening > Process Screening



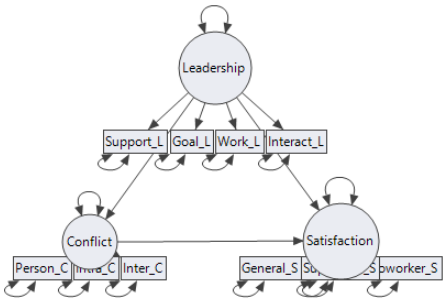
Text Explorer
Analyze > Text Explorer



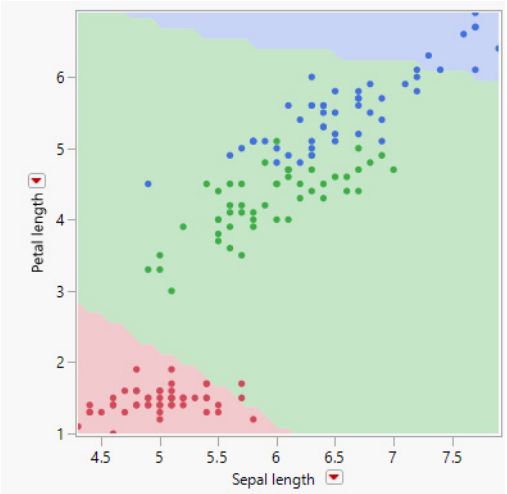
Multiple Factor Analysis
Analyze > Consumer Research > Multiple Factor Analysis



Model Driven Multivariate Control Chart
Analyze > Quality and Process > Model Driven Multivariate Control Chart



Structural Equation Models
Analyze > Multivariate Methods > Structural Equation Models



Support Vector Machines
Analyze > Predictive Modeling > Support Vector Machines

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation 35
JMP Help 36
JMP Documentation Library 36
Additional Resources for Learning JMP 42
JMP Tutorials 42
Sample Data Tables 42
Learn about Statistical and JSL Terms 43
Learn JMP Tips and Tricks 43
JMP Tooltips 43
JMP User Community 44
Free Online Statistical Thinking Course 44
JMP New User Welcome Kit 44
Statistics Knowledge Portal 44
JMP Training 44
JMP Books by Users 45
The JMP Starter Window 45
JMP Technical Support 45

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introducing JMP

Basic Concepts

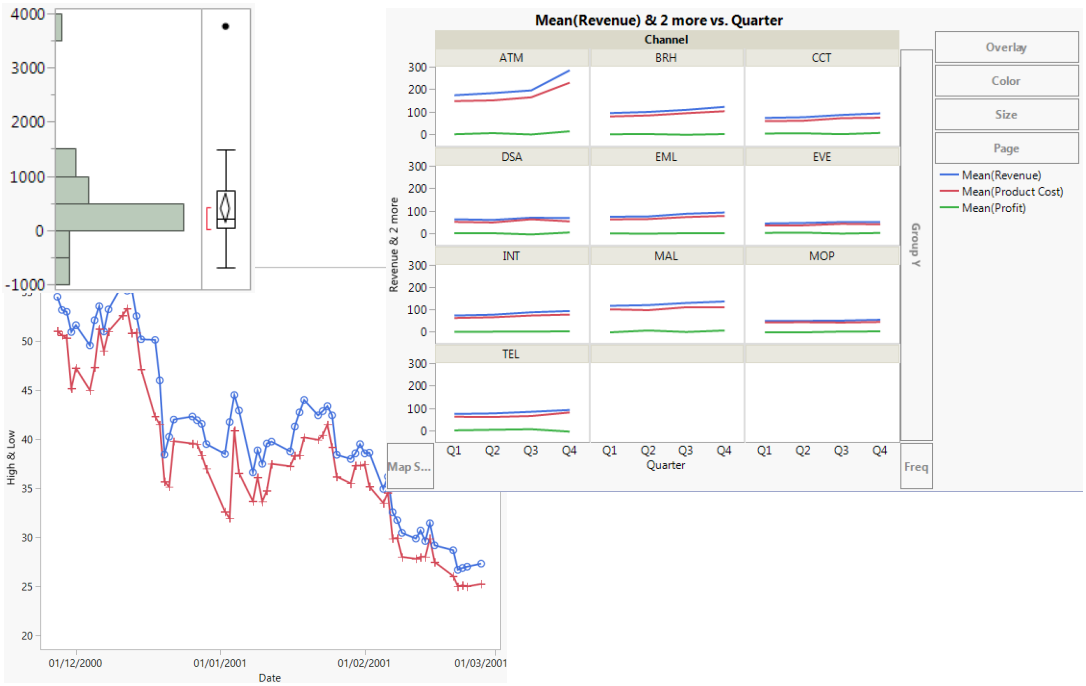
JMP (pronounced *jump*) is a powerful and interactive data visualization and statistical analysis tool. Use JMP to learn more about your data by performing analyses and interacting with the data using data tables, graphs, charts, and reports.

JMP enables researchers to perform a wide range of statistical analyses and modeling. JMP is equally useful to the business analyst who wants to quickly uncover trends and patterns in data. With JMP, you do not have to be an expert in statistics to get information from your data.

For example, you can use JMP to do the following:

- Create interactive graphs and charts to explore your data and discover relationships.
- Discover patterns of variation across many variables at once.
- Explore and summarize large amounts of data.
- Develop powerful statistical models to predict the future.

Figure 2.1 Examples of JMP Reports



Contents

JMP Concepts That You Should Know 49
How Do I Get Started with JMP? 49
Starting JMP 50
Using Sample Data 52
Understand Data Tables 53
Understand the JMP Workflow 54
Step 1: Launch a JMP Platform and View Results 55
Step 2: Remove the Box Plot from a JMP Report 57
Step 3: Request Additional JMP Output 57
Step 4: Interact with JMP Platform Results 58
How is JMP Different from Excel? 59
Structure of a Data Table 59
Formulas in JMP 60
JMP Analysis and Graphing 61

JMP Concepts That You Should Know

Before you begin using JMP, you should be familiar with these concepts:

- Enter, view, edit, and manipulate data using JMP *data tables*.
- Select a *platform* from the **Analyze**, **Graph**, or **DOE** menus. Platforms contain interactive windows that you use to analyze data and work with graphs.
- Platforms use these windows:
  - *Launch windows* where you set up and run your analysis.
  - *Report windows* showing the output of your analysis.
- Report windows normally contain the following items:
  - A graph of some type (such as a scatterplot or a chart).
  - Specific *reports* that you can show or hide using the *disclosure icon* .
  - Platform *options* that are located within *red triangle menus* .

How Do I Get Started with JMP?

The general workflow in JMP is simple:

1. Get your data into JMP.
2. Select a platform and complete its launch window.
3. Explore your results and discover where your data takes you.

This workflow is described in more detail in "Understand the JMP Workflow".

Typically, you start your work in JMP by using graphs to visualize individual variables and relationships among your variables. Graphs make it easy to see this information, and to see the deeper questions to ask. Then you use analysis platforms to dig deeper into your problems and find solutions.

- "Work with Your Data" shows you how to get data into JMP.
- "Visualize Your Data" shows you how to use some of the useful graphs JMP provides to look more closely at your data.
- "Analyze Your Data" shows you how to use some of the analysis platforms.
- "The Big Picture" shows you how to analyze distributions, patterns, and similar values in several platforms.

Each chapter teaches through examples. The following sections in this chapter describe data tables and general concepts for working in JMP.

Starting JMP

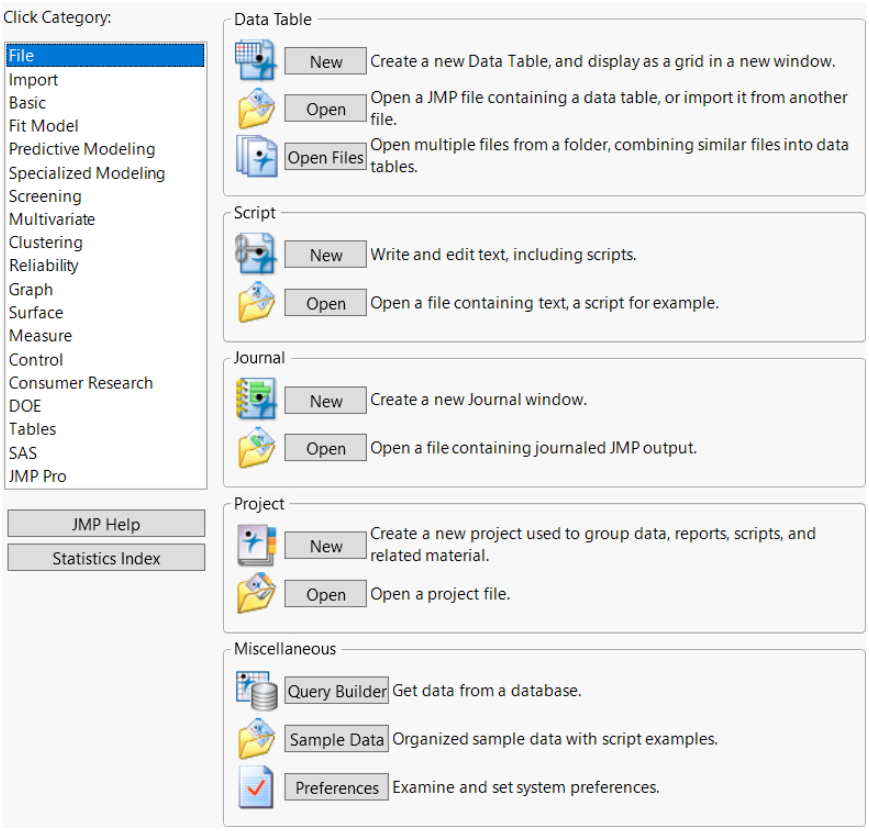
Start JMP in two ways:

- Double-click the JMP icon, normally found on your desktop. This starts JMP but does not open any existing JMP files.
- Double-click an existing JMP file. This starts JMP and opens the file.

The initial view of JMP includes the Tip of the Day window and the Home Window on Windows; on macOS, the Tip of the Day and JMP Starter, and Home Window initially appear.

The JMP Starter window classifies actions and platforms using categories.

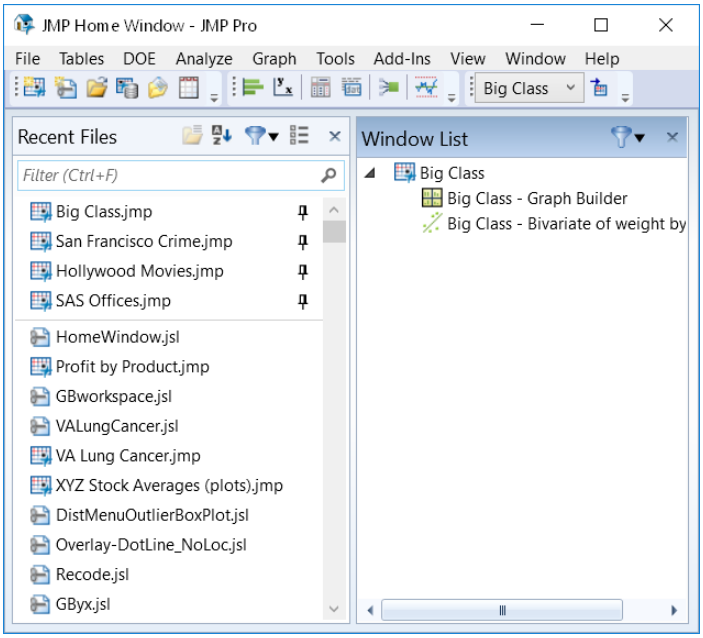
Figure 2.2 The JMP Starter



On the left is a list of categories. Click a category to see the features and the commands related to that category. The JMP Starter also lists JMP Pro features and platforms.

The Home Window helps you organize and access files in JMP.

Figure 2.3 The Home Window on Windows



To open the Home Window on Windows, select **View > Home Window**. On macOS, select **Window > JMP Home**. The Home Window includes links to the following:

- the data tables and report windows that are currently open
- files that you have opened recently

For more information about the Home Window, see *Using JMP*.

Almost all JMP windows contain a menu bar and a toolbar. You can find most JMP features in three ways:

- using the menu bar
- using the toolbar buttons
- using the buttons on the JMP Starter window

About the Menu Bar and Toolbars

The menus and toolbars are hidden in many windows. To see them, hover over the blue bar under the window's title bar. The menus in the JMP Starter window, the Home Window, and all data tables are always visible.

Using Sample Data

The examples in *Discovering JMP* and other JMP documentation use sample data tables. The default location on Windows for the sample data is:

C:/Program Files/SAS/JMP/16/Samples/Data

C:/Program Files/SAS/JMPPro/16/Samples/Data

The Sample Data Index groups the data tables by category. Click a disclosure icon to see a list of data tables for that category, and then click a link to open a data table.

macOS sample data is installed in /Library/Application Support/JMP/16/Samples/Data.

Opening a JMP Sample Data Table

1. From the **Help** menu, select **Sample Data**.
2. Open the **Data Tables used in Discovering JMP** list by clicking on the disclosure icon next to it.
3. Click the name of the data table to use it in the examples in *Discovering JMP*.

Sample Import Data

Use files from other applications to learn how to import data into JMP.

The default location on Windows for the sample import data is:

C:/Program Files/SAS/JMP/16/Samples/Import Data

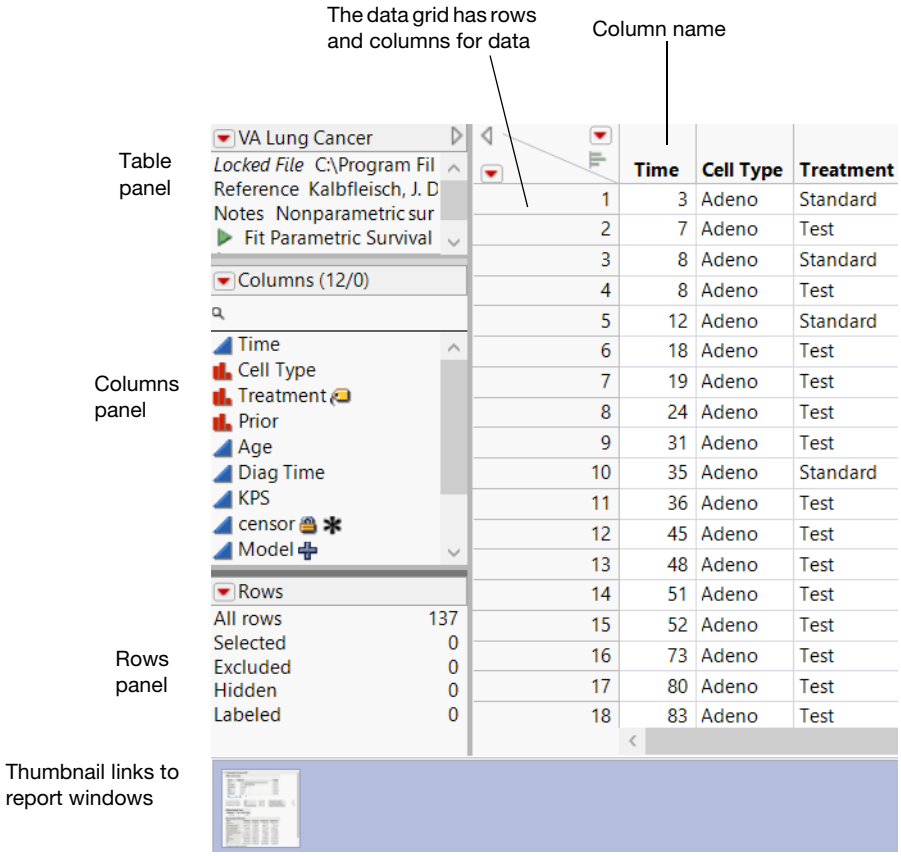
C:/Program Files/SAS/JMPPro/16/Samples/Import Data

Understand Data Tables

A JMP data table is a collection of data organized in rows and columns. A data table might also contain other information like notes, variables, and scripts. These supplementary items are discussed in later chapters.

Open the VA Lung Cancer data table to see the data table described here.

Figure 2.4 A Data Table



A data table contains the following parts:

Data grid The data grid contains the data arranged in rows and columns. Generally, each row in the data grid is an observation, and the columns (also called variables) give information about the observations. In Figure 2.4, each row corresponds to a test subject, and there are twelve columns of information. Although all twelve columns cannot be shown in the data grid, the Columns panel lists them all. The information given about

each test subject includes the time, cell type, treatment, and more. Each column has a header, or name. That name is not part of the table's total count of rows.

Table panel The table panel can contain table variables or table scripts. In Figure 2.4, there is one saved script called **Model** that can automatically re-create an analysis. This table also has a variable named Notes that contains information about the data. Table variables and table scripts are discussed in a later chapter.

Columns panel The columns panel shows the total number of columns, whether any columns are selected, and a list of all the columns by name. The numbers in parentheses (12/0) show that there are twelve columns, and that no columns are selected. An icon to the left of each column name shows that column's modeling type. Modeling types are described in "Understand Modeling Types". Icons to the right show any attributes assigned to the column. See "View or Change Column Information in a Data Table" for more information about these icons.

Rows panel The rows panel shows the number of rows in the data table, and how many rows are selected, excluded, hidden, or labeled. In Figure 2.4, there are 137 rows in the data table.

Thumbnail links to report windows This area shows thumbnails of all reports based on the data table. Hover over a thumbnail to see a larger preview of the report window. Double-click a thumbnail to bring the report window to the front.

Interacting with the data grid, which includes adding rows and columns, entering data, and editing data, is discussed in "Work with Your Data". If you open multiple data tables, each one appears in a separate window.

For more information about how a JMP data table differs from an Excel spreadsheet, see "How is JMP Different from Excel?".

Understand the JMP Workflow

Once your data is in a data table, you can create graphs or plots, and perform analyses. All features are located in platforms, which are found primarily on the **Analyze** or **Graph** menus. They are called platforms because they do not just produce simple static results. Platform results appear in report windows, are highly interactive, and are linked to the data table and to each other.

The platforms under the **Analyze** and **Graph** menus provide a variety of analytical features and data exploration tools.

Here are the general steps to produce a graph or analysis:

1. Open a data table.

2. Select a platform from the Graph or Analyze menu.
3. Complete the platform launch window to set up your analysis.
4. Click **OK** to create the report window that contains your graphs and statistical analyses.
5. Customize your report by using report options.
6. Save, export, and share your results with others.

Later chapters discuss these concepts in greater detail.

The following example shows you how to perform a simple analysis and customize it in four steps. This example uses the Companies.jmp file sample data table to show a basic analysis of the variable Profits ($M).

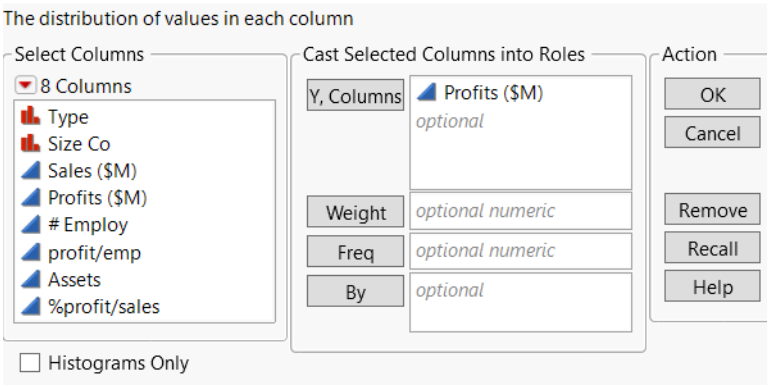
Step 1: Launch a JMP Platform and View Results

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze > Distribution** to open the Distribution launch window.
3. Select Profits ($M) in the Select Columns box and click the **Y, Columns** button.

   The variable Profits ($M) appears in the **Y, Columns** role.

   Another way to assign variables is to click and drag columns from the Select Columns box to any of the roles boxes.

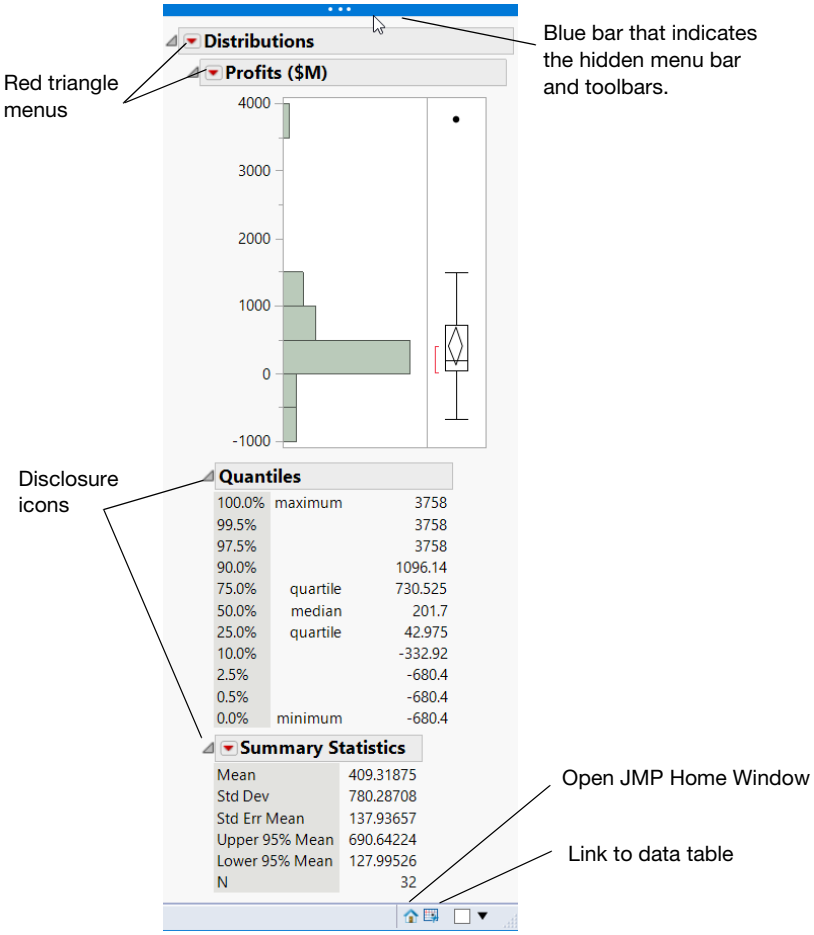
Figure 2.5 Assign Profits ($M)



4. Click **OK**.

   The Distribution report window appears.

Figure 2.6 Distribution Report Window on Windows



The report window contains basic plots or graphs and preliminary analysis reports. The results appear in an outline format, and you can show or hide any report by clicking on the disclosure icon.

Red triangle menus contain options and commands to request additional graphs and analyses at any time.

- On Windows, hover over the blue bar at the top of the window to see the menu bar and the toolbars.
- On Windows, click the data table button in the lower right corner to view the data table that was used to create this report. On macOS, click the **Show Data Table** button in the upper right corner of the report window.
- On Windows, click the **JMP Home Window** button in the lower right corner to view the Home Window. On macOS, select **Window > JMP Home**.

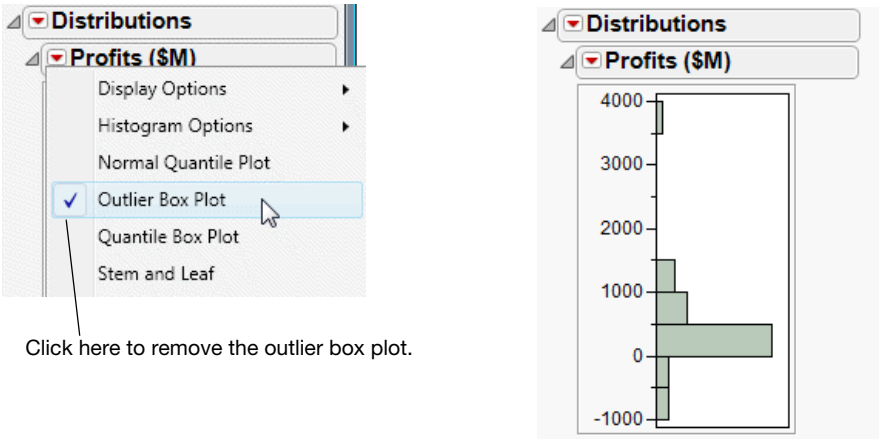
Step 2: Remove the Box Plot from a JMP Report

Continue using the Distribution report that you created earlier.

1. Click the red triangle next to Profits ($M) to see a menu of report options.
2. Deselect **Outlier Box Plot** to turn the option off.

   The outlier box plot is removed from the report window.

Figure 2.7 Removing the Outlier Box Plot



Step 3: Request Additional JMP Output

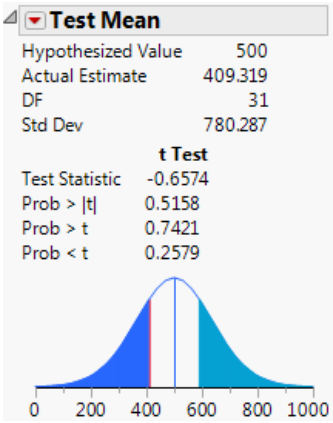
Continue to use the same report window that you created in "Step 2: Remove the Box Plot from a JMP Report".

1. Click the red triangle next to Profits ($M) and select **Test Mean**.

   The Test Mean window appears.
2. Enter 500 in the **Specify Hypothesized Mean** box.
3. Click **OK**.

   The test for the mean is added to the report window.

Figure 2.8 Test for the Mean



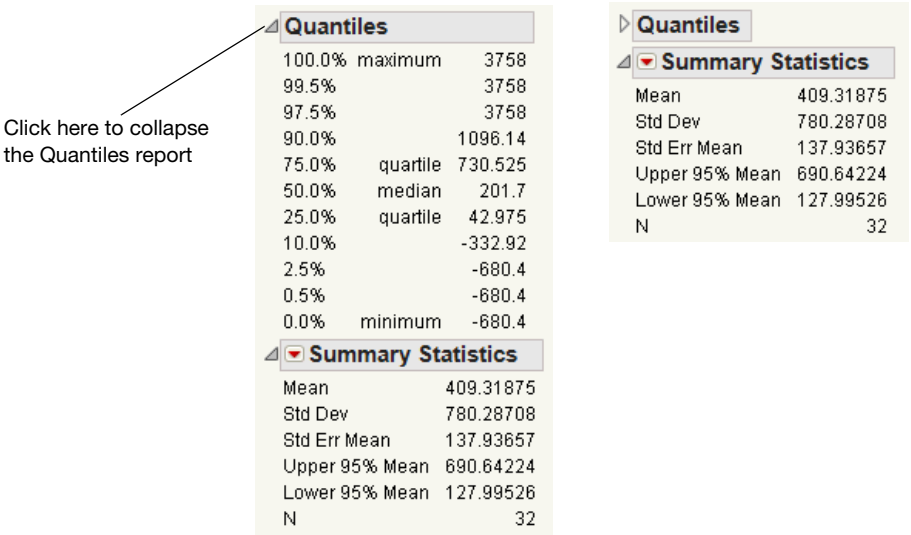
Step 4: Interact with JMP Platform Results

All platforms produce results that are interactive, for example, the following results:

- Reports can be shown or hidden.
- Additional graphs and statistical details can be added or removed to suit your purposes.
- Platform results are connected to the data table and to each other.

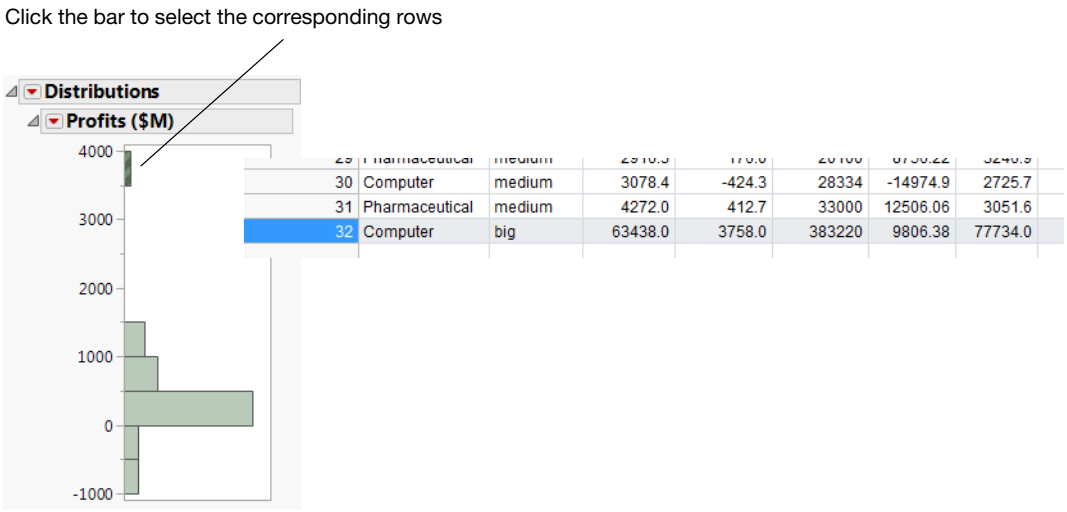
For example, to close the **Quantiles** report, click the disclosure icon next to **Quantiles**.

Figure 2.9 Close the Quantiles Report



Platform results are connected to the data table. The histogram in Figure 2.10 shows that a group of companies makes a much higher profit that the others. To quickly identify that group, click the histogram bar for them. The corresponding rows in the data table are selected.

Figure 2.10 Connection between Platform Results and Data Table



In this case, the group includes only one company, and that one row is selected.

How is JMP Different from Excel?

JMP is a statistical analysis program that uses data tables. Excel is a spreadsheet application. Data tables and spreadsheets have different structures.

- "Structure of a Data Table"
- "Formulas in JMP"
- "JMP Analysis and Graphing"

Structure of a Data Table

A data table has fixed rows and columns, while a spreadsheet is cell based. In a spreadsheet, data, headings, or formulas can be placed in any cell. In a data table, the structure organizes data for analysis. This structure is used by JMP analysis and graphing platforms.

Column Headings Column names are column headings.

Columns Columns contain data and are assigned one data type. Basic columns are either numeric or character. If a column contains both character and numeric data, the entire

column's data type is character, and the numbers are treated as character data. JMP also has specialized column types for capturing things such as images. JMP uses the column's data type to determine analysis options and results. For more information about data types, see "Understand Modeling Types".

Rows Rows contain observations. If there is no observation for a row, that cell is left empty. In JMP a dot signifies a missing numeric value, and a blank signifies a missing character value,

For more information about JMP data tables, see "Understand Data Tables". For more information about JMP column properties, see the *Using JMP* book.

JMP data tables cannot be arranged in a workbook such as in Excel. Each JMP data table is a separate file and appears in its own window. To combine multiple tables, see the *Using JMP* book. For organizing JMP tables and output see "Save and Run Scripts".

Tip: To use data from two or more tables in a single analysis, use Virtual Join. For more information, see the *Using JMP* book.

Formulas in JMP

In spreadsheets, formulas apply to a single cell and can utilize data from any cell in the spreadsheet, including cells on different tabs of the workbook. In data tables, formulas apply to an entire column. A formula can use data from any other column in the data table. Each row in the column will have the same calculation applied to it based on the data in the row.

For example, consider a data table with a simple sum as shown in Figure 2.11. The column height + weight has a formula. The formula adds height and weight by row for all rows in the data table.

Figure 2.11 Data Table with Formula Column

	name	age	sex	height	weight	height + weight
1	KATIE	12	F	59	95	154
2	LOUISE	12	F	61	123	184
3	JANE	12	F	55	74	129
4	JACLYN	12	F	66	145	211
5	LILLIE	12	F	52	64	116
6	TIM	12	M	60	84	144
7	JAMES	12	M	61	128	189

For more information about JMP formulas, see the *Using JMP* book.

Tip: For basic column summary statistics, use the Distribution platform. See the *Basic Analysis* book.

JMP Analysis and Graphing

JMP uses platforms to drive data analysis. To launch an analysis, go to the Analyze menu. Select the variables for your analysis in the platform launch window, and the analysis results appear in a report window that is separate from the data table. This differs from Excel, where an analysis is inserted on) the spreadsheet.

Graphing choices are found in the Graph menu. Graph Builder is a great place to start. Use Graph Builder to drag and drop your columns and quickly build a graph to explore your data. For more information about Graph Builder, see the *Essential Graphing* book.

Chapter 3

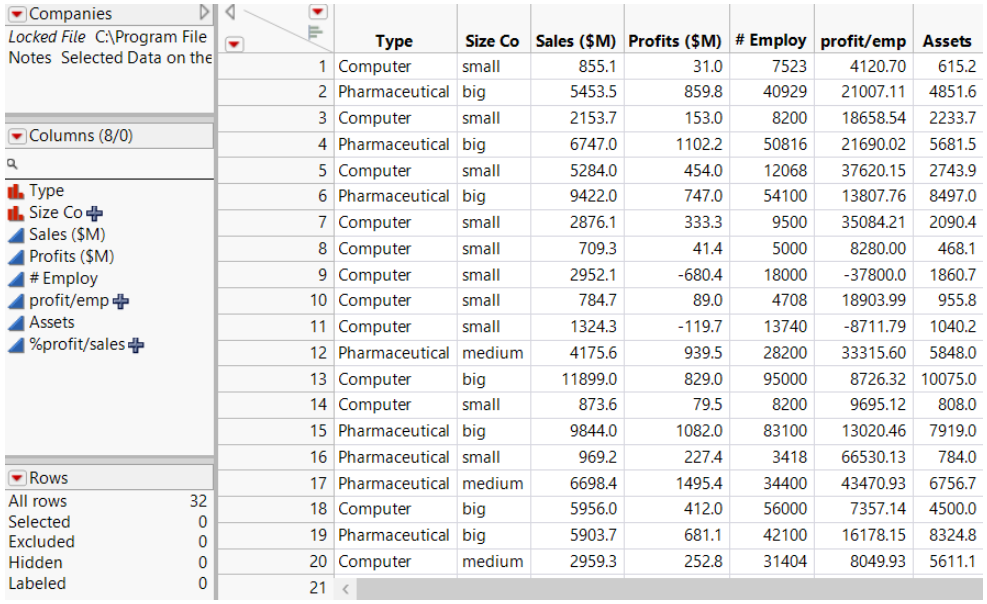
Work with Your Data

Prepare Your Data for Graphing and Analyzing

Before graphing or analyzing your data, the data has to be in a data table and in the proper format. This chapter shows some basic data management tasks, including the following:

- Creating new data tables
- Opening existing data tables
- Importing data from other applications into JMP
- Managing your data

Figure 3.1 Example of a Data Table



	Type	Size Co	Sales ($M)	Profits ($M)	# Employ	profit/emp	Assets
1	Computer	small	855.1	31.0	7523	4120.70	615.2
2	Pharmaceutical	big	5453.5	859.8	40929	21007.11	4851.6
3	Computer	small	2153.7	153.0	8200	18658.54	2233.7
4	Pharmaceutical	big	6747.0	1102.2	50816	21690.02	5681.5
5	Computer	small	5284.0	454.0	12068	37620.15	2743.9
6	Pharmaceutical	big	9422.0	747.0	54100	13807.76	8497.0
7	Computer	small	2876.1	333.3	9500	35084.21	2090.4
8	Computer	small	709.3	41.4	5000	8280.00	468.1
9	Computer	small	2952.1	-680.4	18000	-37800.0	1860.7
10	Computer	small	784.7	89.0	4708	18903.99	955.8
11	Computer	small	1324.3	-119.7	13740	-8711.79	1040.2
12	Pharmaceutical	medium	4175.6	939.5	28200	33315.60	5848.0
13	Computer	big	11899.0	829.0	95000	8726.32	10075.0
14	Computer	small	873.6	79.5	8200	9695.12	808.0
15	Pharmaceutical	big	9844.0	1082.0	83100	13020.46	7919.0
16	Pharmaceutical	small	969.2	227.4	3418	66530.13	784.0
17	Pharmaceutical	medium	6698.4	1495.4	34400	43470.93	6756.7
18	Computer	big	5956.0	412.0	56000	7357.14	4500.0
19	Pharmaceutical	big	5903.7	681.1	42100	16178.15	8324.8
20	Computer	medium	2959.3	252.8	31404	8049.93	5611.1
21							

Contents

Get Your Data into JMP 65
Copy and Paste Data into a Data Table 65
Import Data into a Data Table 65
Enter Data in a Data Table 68
Transfer Data from Excel to JMP 70
Work with Data Tables 72
Edit Data in a Data Table 72
Select, Deselect, and Find Values in a Data Table 74
View or Change Column Information in a Data Table 78
Example of Calculating Values with Formulas 80
Example of Filtering Data in a Report 82
Examples of Reshaping Data 83
Examples of Viewing Summary Statistics 84
Examples of Creating Subsets 88
Example of Joining Data Tables 90
Example of Sorting Data 92

Get Your Data into JMP

JMP provides many ways to get your data into JMP.

- To copy and paste data from another application, see "Copy and Paste Data into a Data Table".
- To import data from another application, see "Import Data into a Data Table".
- To enter data directly into a data table, see "Enter Data in a Data Table"

You can also import data into JMP from a database. See *Using JMP*.

This chapter uses sample data tables and sample import data that is installed with JMP. To find these files, see "Using Sample Data".

Copy and Paste Data into a Data Table

You can move data into JMP by copying and pasting from another application, such as Microsoft Excel or a text file.

1. Open the VA Lung Cancer.xlsx file in Microsoft Excel. This file is located in the Sample Import Data folder.
2. Select all of the rows and columns, including the column names. There are 12 columns and 138 rows.
3. Copy the selected data.
4. In JMP, select **File > New > Data Table** to create an empty table.
5. Select **Edit > Paste with Column Names** to paste the data and column headings.

   If the data that you are pasting into JMP does *not* have column names, then you can use **Edit > Paste**.

Import Data into a Data Table

You can move data into a JMP data table by importing data from another application, such as Microsoft Excel, SAS, or text files. Here are the basic steps to import data:

1. Select **File > Open**.
2. Navigate to your file's location.
3. If your file is not listed in the Open Data File window, select the correct file type from the **Files of type** menu.
4. Click **Open**.

Example of Importing a Microsoft Excel File

1. Select **File > Open**.
2. Navigate to the Samples/Import Data folder.
3. Select Team Results.xlsx.

   Note the rows and columns on which the data begin. The spreadsheet also contains two worksheets. In this example, you import the Ungrouped Team Results worksheet.
4. Click **Open**.

   The spreadsheet opens in the Excel Import Wizard, where a preview of the data appears along with import options.

   Text from the first row of the spreadsheet are column headings. However, you want text in row 3 of the spreadsheet to be converted to column headings.
5. Next to **Column headers start on row**, type 3, and press Enter. The column headings are updated in the data preview. The value for the first row of data is updated to 4.
6. Save the settings only for this worksheet:
   - Deselect **Use for all worksheets** in the lower left corner of the window.
   - Select **Ungrouped Team Results** in the upper right corner of the window.
7. Click **Import** to convert the spreadsheet as you specified.

When you import Excel files, JMP predicts whether columns headings exist, and if the column names are on row one. The copy and paste method is recommended for the following situations:

- If the column names are located in a row other than row one
- If the file does not include column names and the data does not start in row one
- If the file contains column names and the data does not start in row two

See "Copy and Paste Data into a Data Table" and *Using JMP* for more information about importing Excel files.

Example of Importing a Text File

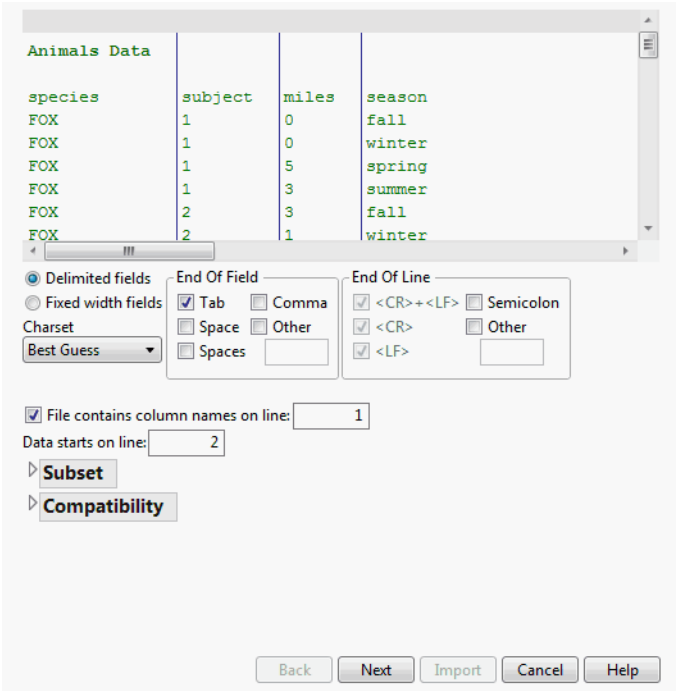
One way to import a text file is to let JMP assume the data's format and place the data in a data table. This method uses settings that you can specify in Preferences. See *Using JMP* for information about setting text import preferences.

Another way to import a text file is to use a Text Preview window to see what your data table will look like after importing, and make adjustments. The following example shows you how to use Text Import Preview window.

1. Select **File > Open**.
2. Navigate to the Samples/Import Data folder.

3. Select Animals_line3.txt.
4. At the bottom of the Open window, select **Data with Preview**.
5. Click **Open**.

Figure 3.2 Initial Preview Window



This text file has a title on the first line, column names on the third line, and the data starts on line four. If you opened this directly in JMP, the Animals Data line would be the first column name, and all the column names and data afterward would be out of sync. The Preview window lets you adjust the settings before you open the file, and see how your adjustments affect the final data table.

6. Enter 3 in the **File contains column names on line** field.
7. Enter 4 in the **Data starts on line** field.
8. Click **Next**.

   In the second window, you can exclude columns from the import and change the data modeling of the columns. For this example, use the default settings.
9. Click **Import**.

The new data table has columns named species, subject, miles, and season. The species and season columns are character data. The subject and miles columns are continuous numeric data.

Tip: You can import several text files at once to create a data table. See *Using JMP*.

Enter Data in a Data Table

You can enter data directly in a data table. The following example shows you how to enter data that was collected over several months into a data table.

Scenario

Table 3.1 shows the data from a study that investigated a new blood pressure medication. Each individual's blood pressure was measured over a six-month period. Two doses (300mg and 450mg) of the medication were used, along with a control and placebo group. The data shows the average blood pressure for each group.

Table 3.1 Blood Pressure Data

Month	Control	Placebo	300mg	450mg
March	165	163	166	168
April	162	159	165	163
May	164	158	161	153
June	162	161	158	151
July	166	158	160	148
August	163	158	157	150

Enter Data in a New Data Table

1. Select **File > New > Data Table** to create an empty data table.

   A new data table has one column and no rows.
2. Select the column name and change the name to Month.

   Note: To rename a column, you can also double-click the column name or select the column and press Enter.

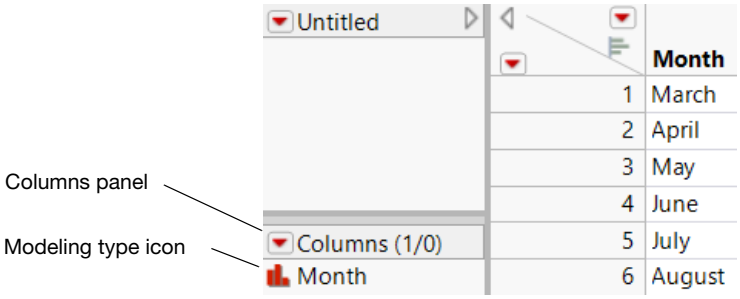
Figure 3.3 Entering a Column Name

Click once to select the column. Then type “Month”.

3. Select **Rows > Add Rows**.

   The Add Rows window appears.
4. Since you want to add six rows, type 6.
5. Click **OK**. Six empty rows are added to the data table.
6. Enter the Month information by clicking in a cell and typing.

Figure 3.4 Month Column Completed



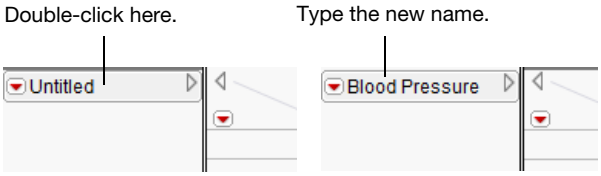
In the columns panel, look at the modeling type icon to the left of the column name. It has changed to reflect that Month is now nominal (previously it was continuous). Compare the modeling type shown for Column 1 in Figure 3.3 and for Month in Figure 3.4. This difference is important and is discussed in “View or Change Column Information in a Data Table”.

7. Double-click in the space on the right side of the Month column to add the Control column.
8. Change the name to Control.
9. Enter the Control data as shown in Table 3.1. Your data table now consists of six rows and two columns.
10. Continue adding columns and entering data as shown in Table 3.1 to create the final data table with six rows and five columns.

Change the Data Table Name

1. Double-click the data table name (Untitled) in the Table Panel.
2. Type the new name (Blood Pressure).

Figure 3.5 Changing the Data Table Name



Transfer Data from Excel to JMP

You can use the JMP Add In for Excel to transfer a spreadsheet from Excel to JMP:

- a data table
- Graph Builder
- Distribution platform
- Fit Y by X platform
- Fit Model platform
- Time Series platform
- Control Chart platform

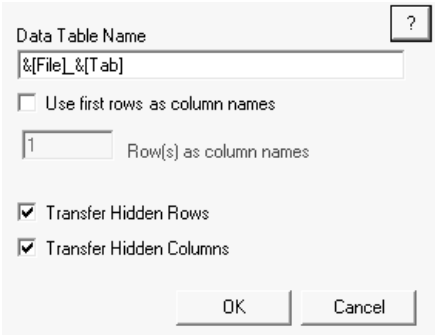
Set JMP Add In Preferences in Excel

To configure JMP Add In Preferences:

1. In Excel, select **JMP > Preferences**.

   The JMP Preferences window appears.

Figure 3.6 JMP Add In Preferences



2. Accept the default **Data Table Name** (File name_Worksheet name) or type a name.
3. Select to **Use the first rows as column names** if the first row in the worksheet contains column headers.
4. If you selected to use the first rows a column headers, type the number of rows used.
5. Select to **Transfer Hidden Rows** if the worksheet contains hidden rows to be included in the JMP data table.
6. Select to **Transfer Hidden Column** if the worksheet contains hidden columns to be included in the JMP data table.
7. Click **OK** to save your preferences.

Transfer to JMP

To transfer an Excel worksheet to JMP:

1. Open the Excel file.
2. Select the worksheet to transfer.
3. Select **JMP** and then select the JMP destination:
   - Data Table
   - Graph Builder
   - Distribution platform
   - Fit Y by X platform
   - Fit Model platform
   - Time Series platform
   - Control Chart platform

   The Excel worksheet is opened as a data table in JMP and the selected platform's launch window appears.

Work with Data Tables

This section describes the basic concepts for working with data tables.

- "Edit Data in a Data Table"
- "Select, Deselect, and Find Values in a Data Table"
- "View or Change Column Information in a Data Table"
- "Example of Calculating Values with Formulas"
- "Example of Filtering Data in a Report"

Tip: Consider setting the *Autosave timeout* value in the General preferences to automatically save open data tables at the specified number of minutes. This autosave value also applies to journals, scripts, projects, and reports.

Edit Data in a Data Table

You can enter or change data, either a few cells at a time or for an entire column. This section contains the following information:

- "Change Values in a Data Table Cell"
- "Recode Values"
- "Create Patterned Data"

Change Values in a Data Table Cell

To change a value, select a cell and type the change. You can also double-click a cell to edit it.

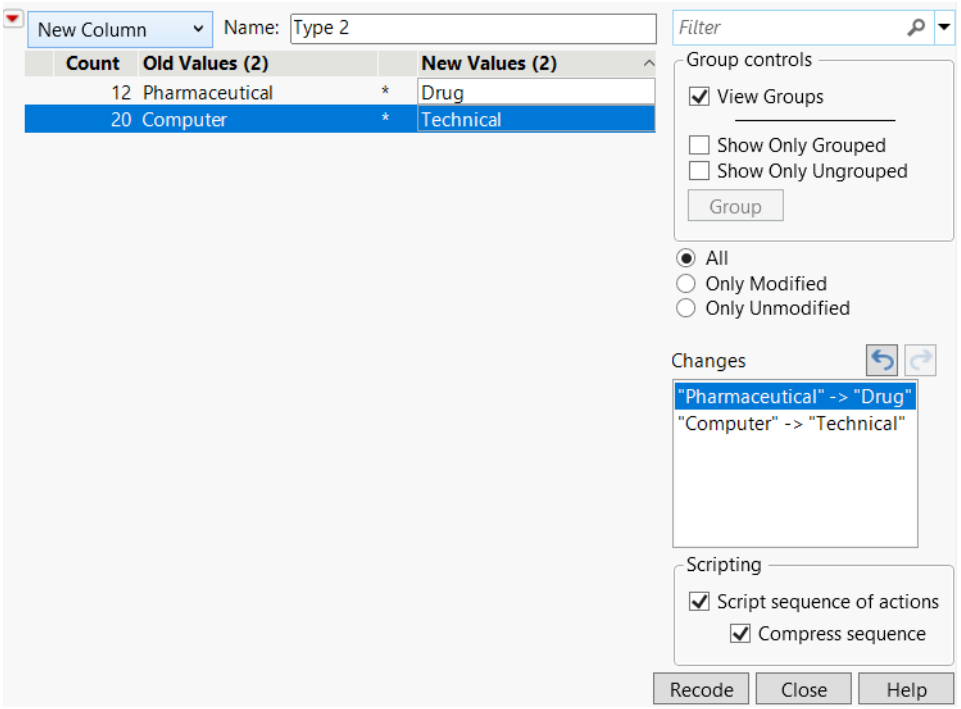
Note: Double-clicking in a cell is not the same as selecting a cell. A single click selects a cell. You can select more than one cell at the same time, and you can perform certain actions on selected cells. Double-clicking only lets you edit a cell. For more information about selecting rows, columns, and cells, see "Select, Deselect, and Find Values in a Data Table".

Recode Values

Use the recoding tool to change all of the values in a column at once. For example, suppose that you are interested in comparing the sales of computer and pharmaceutical companies. Your current company labels are Computer and Pharmaceutical. You want to change them to Technical and Drug. Going through all 32 rows of data and changing all the values would be tedious, inefficient, and error-prone, especially if you had many more rows of data. Recode is a better option.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select the Type column by clicking once on the column heading.
3. Select **Cols > Recode**.
4. In the New Value column of the Recode window, type Technical in the Computer row and Drug in the Pharmaceutical row.
5. Select **In Place** from the New Column list.
6. Click **Recode**.

Figure 3.7 Recode Window



All cells are updated automatically to the new values.

Create Patterned Data

Use the Fill options to populate a column with patterned data. The Fill options are especially useful if your data table is large, and typing in the values for each row would be cumbersome.

Example of Filling a Column with the Pattern

1. Add a new column.
2. Enter 1 in the first cell, 2 in the second cell, and 3 in the third cell.
3. Select the three cells, and right-click anywhere in the selected cells to see a menu.
4. Select **Fill > Repeat sequence to end of table**.

   The rest of the column is filled with the sequence (1, 2, 3, 1, 2, 3, ...).

To continue a pattern instead of repeating it (1, 2, 3, 4, 5, 6, ...), select **Continue sequence to end of table**. This command can also be used to generate patterns like (1, 1, 1, 2, 2, 2, 3, 3, 3, ...).

The Fill options can recognize simple arithmetic and geometric sequences. For character data, the Fill options only repeat the values.

Select, Deselect, and Find Values in a Data Table

You can select rows, columns, or cells within a data table. For example, to create a subset of an existing data table, you must first select the parts of the table that you want to subset. Also, selecting rows can make data points stand out on a graph. Select rows and columns manually by clicking, or select rows that meet certain search criteria. This section contains the following information:

- "Select and Deselect Rows"
- "Select and Deselect Columns"
- "Select and Deselect Cells"
- "Search for Values"

Select and Deselect Rows

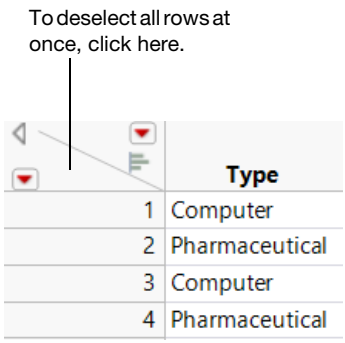
Table 3.2 Selecting and Deselecting Rows

Task	Action
Select rows one at a time	Click the row number.

Table 3.2 Selecting and Deselecting Rows *(Continued)*

Task	Action
Select multiple adjacent rows	Click and drag on the row numbers. or Select the beginning row, press Shift, and then click the last row number.
Select multiple non-adjacent rows	Select the first row, press Ctrl, and then click the other row numbers.
Deselect rows one at a time	Press Ctrl and click the row numbers.
Deselect all rows	Click in the lower-triangular space in the top left corner of the table (Figure 3.8).

Figure 3.8 Deselecting Rows



Select and Deselect Columns

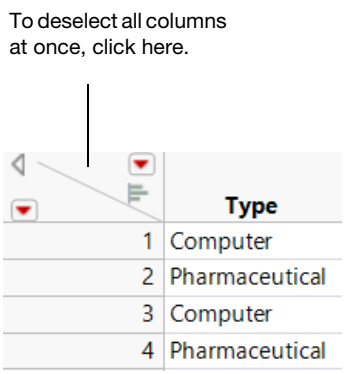
Table 3.3 Selecting and Deselecting Columns

Task	Action
Select columns one at a time	Click the column heading.

Table 3.3 Selecting and Deselecting Columns *(Continued)*

Task	Action
Select multiple adjacent columns	Click and drag across the column headings. or Select the beginning column, press Shift, and then click the last header.
Select multiple non-adjacent columns	Select the first column, press Ctrl, and then click the other column headings.
Deselect columns one at a time	Press Ctrl and click the column heading.
Deselect all columns	Click in the upper-triangular space in the top left corner of the table (Figure 3.9).

Figure 3.9 Deselecting Columns



Select and Deselect Cells

Table 3.4 Selecting and Deselecting Cells

Task	Action
Select cells one at a time	Click each cell individually.

Table 3.4 Selecting and Deselecting Cells *(Continued)*

Task	Action
Select multiple adjacent cells	Click and drag across the cells. or Select the beginning cell, press Shift, and then click the last cell.
Select multiple non-adjacent cells	Select the first cell, press Ctrl, and then click the other cells.
Deselect all cells	Click in the upper and lower triangular spaces in the top left corner of the table.

Search for Values

In a data table that has thousands or tens of thousands of rows, it can be difficult to locate a particular cell by scrolling through the table. If you are looking for specific information, use the Search feature to find it. If data match the search criteria, the cell is selected and the data grid scrolls to show it in the window. For example, the Companies.jmp data table contains information about a company that has total sales of $11,899. Use the Search feature to find that cell.

Example of Searching for a Value

1. Select **Edit > Search > Find** to launch the Search window.
2. In the **Find what** box, enter 11899.
3. Click **Find**. JMP finds the first cell that has 11,899 in it, and selects it.

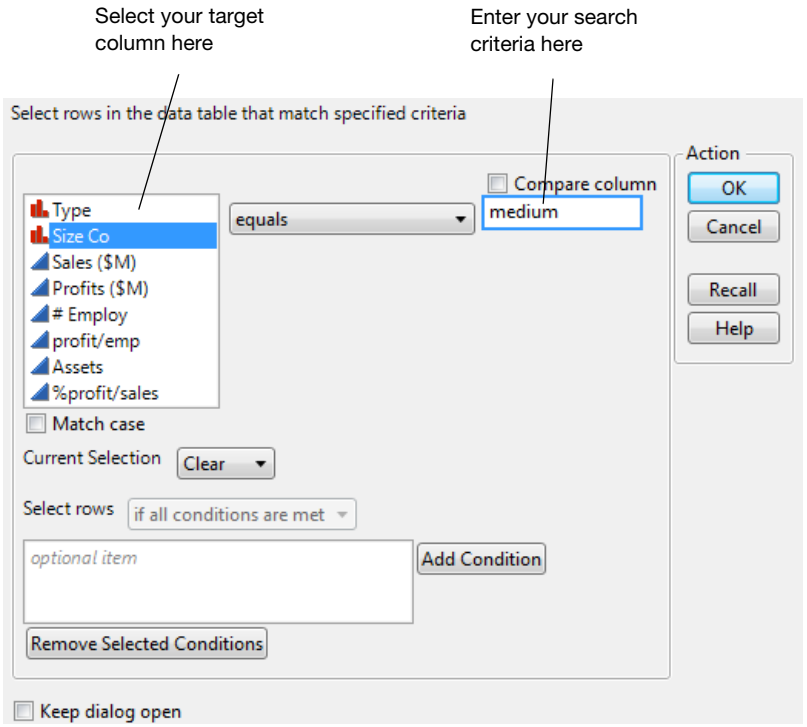
If multiple cells meet the search criteria, click **Find** again to find the next cell that matches the search term.

You can also search for multiple rows at once, with each row matching some criteria.

Example of Select All Rows That Correspond to Medium-Sized Companies

1. Select **Rows > Row Selection > Select Where** to open the **Select rows** window.
2. In the column list box on the left, select Size Co.
3. In the text box on the right, enter medium.
4. Click **OK**.

Figure 3.10 Select Rows Window



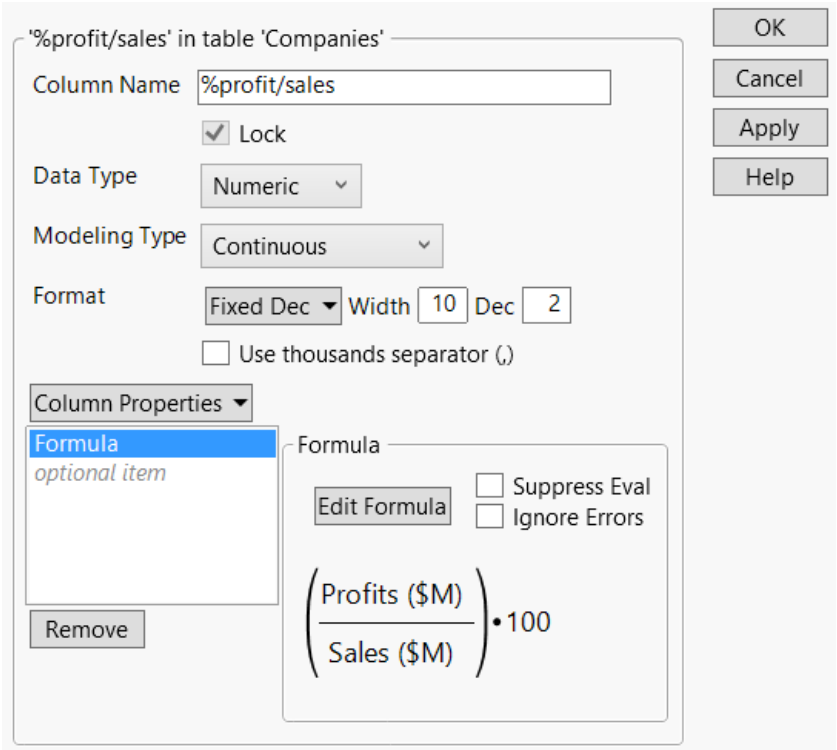
JMP selects all of the rows that have Size Co equal to medium. There are seven.

View or Change Column Information in a Data Table

Information about a data table column is not limited to the data in the column. Data type, modeling type, format, and formulas can also be set.

To view or change column characteristics, double-click the column heading. Or, right-click the column heading and select **Column Info**. The Column Info window appears.

Figure 3.11 Column Info Window



Column Name Enter or change the column name. No two columns can have the same column name.

Data Type Select one of the following data types:

Numeric Specifies the column values as numbers.

Character Specifies the column values as non-numeric, such as letters or symbols.

Row State Specifies the column values as row states. This is an advanced topic. See *Using JMP*.

Modeling Type Modeling types define how values are used in analyses. Select one of the following modeling types:

Continuous Numeric only

Ordinal Either numeric or character, and are ordered categories

Nominal Either numeric or character, but not ordered

Format Select a format for numeric values. This option is not available for character data. Here are a few of the most common formats:

Best Lets JMP choose the best display format.

Fixed Dec Specifies the number of decimal places that appear.

Date Specifies the syntax for date values.

Time Specifies the syntax for time values.

Currency Specifies the type of currency and decimal points that are used for currency values.

Column Properties Set special column properties such as formulas, notes, and value orders. See *Using JMP*.

Lock Lock a column, so that the values in the column cannot be changed.

Example of Calculating Values with Formulas

Use the Formula Editor to create columns that contain calculated values.

Scenario

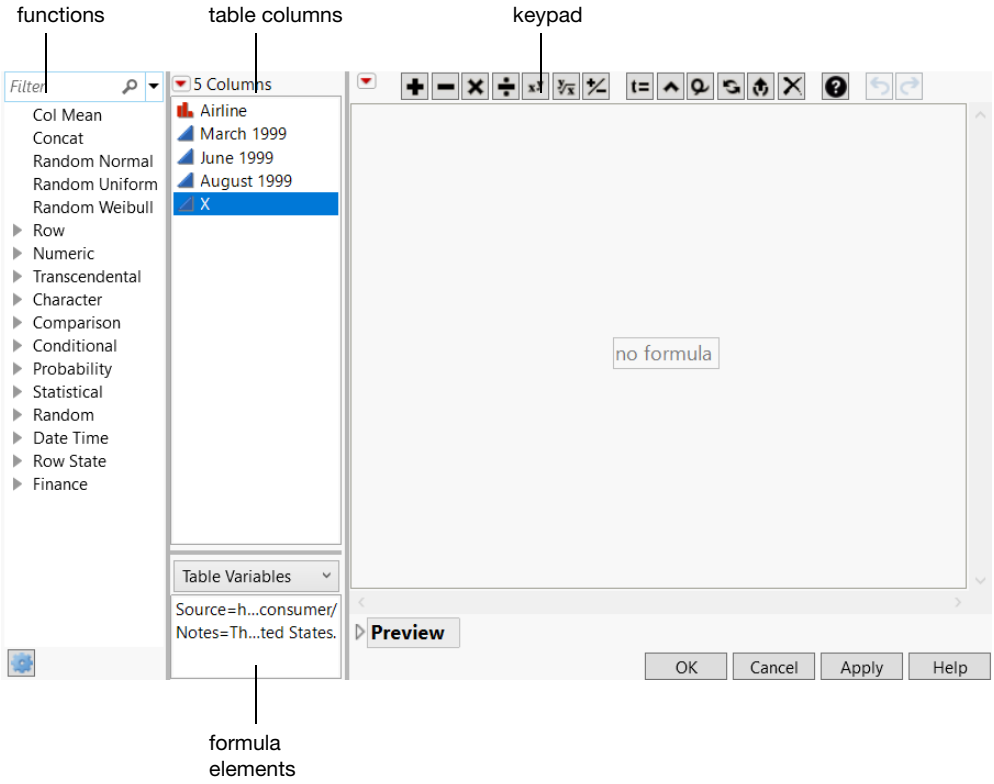
The sample data table On-Time Arrivals.jmp reflects the percent of on-time arrivals for several airlines. The data was collected for March, June, and August of 1999.

Create the Formula

Suppose that you want to create a new column containing the average on-time percentage for each airline.

1. Add a new column.
2. Right-click the column heading of the new column and select **Formula**. The Formula Editor window appears.

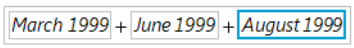
Figure 3.12 Formula Editor



Create the formula for the average on-time percentage of each airline:

3. From the Columns list, select March 1999.
4. Click the ✚ button on the keypad.
5. Select June 1999, followed by another ✚ sign.
6. Select August 1999.

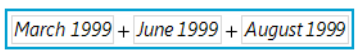
Figure 3.13 Sum of the Months



Notice that only August 1999 is selected (has the blue box around it).

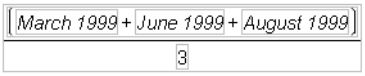
7. Click the box surrounding the entire formula.

Figure 3.14 Entire Formula Selected



8. Click the ÷ button.
9. Type a 3 in the denominator box, and then click outside of the formula in any of the white space.

Figure 3.15 Completed Formula



10. Click **OK**

    The new column contains the averages.

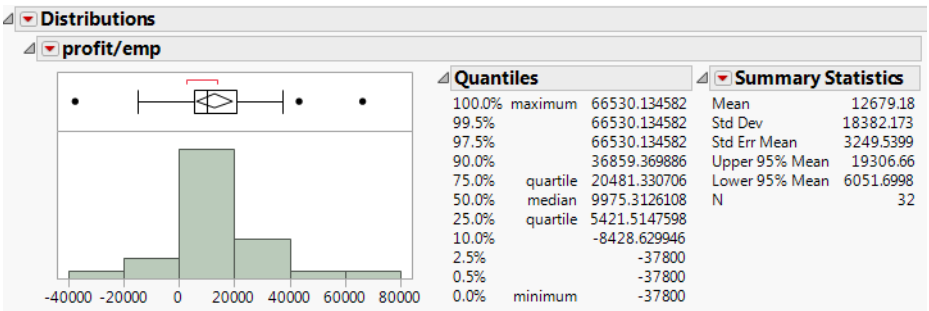
The Formula Editor has many built-in arithmetic and statistical functions. For example, another way to calculate the average on-time arrival percentage is to use the Mean function in the Statistical functions list. For more information about all of the Formula Editor functions, see *Using JMP*.

Example of Filtering Data in a Report

Use the **Data Filter** to interactively select complex subsets of data, hide these subsets in plots, or exclude them from analyses. For example, look at profit per employee for computer and pharmaceutical companies.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Distribution**.
3. Select profit/emp and click **Y, Columns**.
4. Click **OK**.
5. Click the red triangle next to profit/emp and select **Display Options > Horizontal Layout**.

Figure 3.16 Distribution of profit/emp



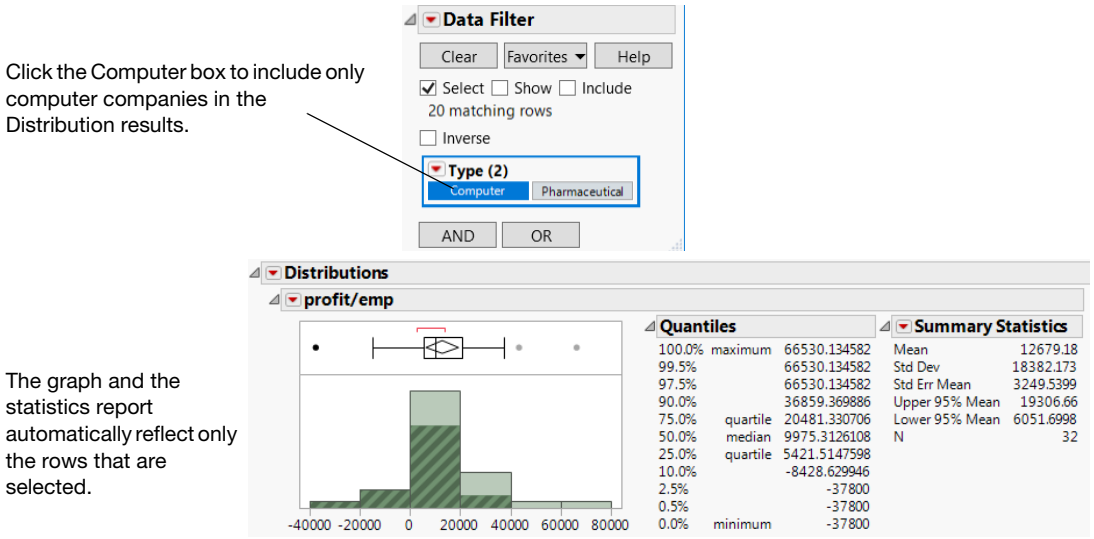
6. Turn on Automatic Recalc by selecting **Redo > Automatic Recalc** from the Distributions red triangle.

   When this option is on, every change that you make (for example, hiding or excluding points) causes your report window to automatically update itself.

7. In the data table, select **Rows** > **Data Filter**.
8. Select Type and click **Add**.
9. Make sure that Select is selected.
10. To filter out the Pharmaceutical companies from the Distribution results, and include only the Computer companies, click the **Computer** box in the Data Filter window.

    The distribution results update to only include Computer companies.

Figure 3.17 Filter for Computer Companies



Conversely, to change the Distribution results to include only the Pharmaceutical companies, click the **Pharmaceutical** box on the Data Filter window.

Examples of Reshaping Data

The commands on the **Tables** menu (and Tabulate on the **Analyze** menu) summarize and manipulate data tables into the format that you need for graphing and analyzing. This section describes five of these commands:

Summary Creates a table that contains summary statistics that describe your data.

Tabulate Provides a drag and drop workspace to create summary statistics.

Subset Creates a table that contains a subset of your data.

Join Joins the data from two data tables into one new data table.

Sort Sorts your data by one or more columns.

For more information about these and the other Tables menu commands, see *Using JMP*.

Examples of Viewing Summary Statistics

Summary statistics, such as sums and means, can instantly provide useful information about your data. For example, if you look at the annual profit of each company out of thirty-two companies, it's difficult to compare the profits of small, medium, and large companies. A summary shows that information immediately.

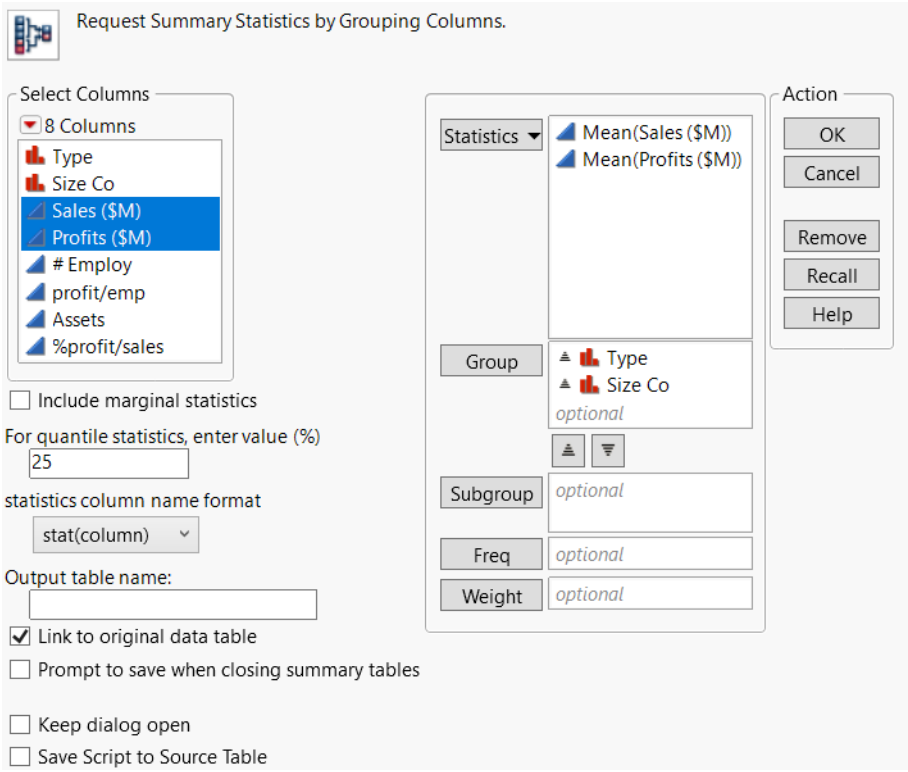
Create summary tables by using either the **Summary** or **Tabulate** commands. The **Summary** command creates a new data table. As with any data table, you can perform analyses and create graphs from the summary table. The **Tabulate** command creates a report window with a table of summary data. You can also create a table from the Tabulate report.

Summary Table Example

A summary table contains statistics for each level of a grouping variable. For example, look at the financial data for computer and pharmaceutical companies. Suppose that you want to calculate the mean of sales and the mean of profits, for each combination of company type and size.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Tables > Summary**.
3. Select Type and Size Co and click **Group**.
4. Select Sales ($M) and Profits ($M) and click **Statistics > Mean**.

Figure 3.18 Completed Summary Window



5. Click **OK**.

JMP calculates the mean of Sales ($M) and the mean of Profit ($M) for each combination of Type and Size Co.

Figure 3.19 Summary Table

	Type	Size Co	N Rows	Mean(Sales ($M))	Mean(Profits ($M))
1	Computer	big	4	20597.48	1089.93
2	Computer	medium	2	3018.85	-85.75
3	Computer	small	14	1758.06	44.94
4	Pharmaceutical	big	5	7474.04	894.42
5	Pharmaceutical	medium	5	4261.06	698.98
6	Pharmaceutical	small	2	1083.75	156.95

The summary table contains the following:

- There are columns for each grouping variable (in this example, Type, and Size Co).

- The N Rows column shows the number of rows from the original table that correspond to each combination of grouping variables. For example, the original data table contains 14 rows corresponding to small computer companies.
- There is a column for each summary statistic requested. In this example, there is a column for the mean of Sales ($M) and a column for the mean of Profits ($M).

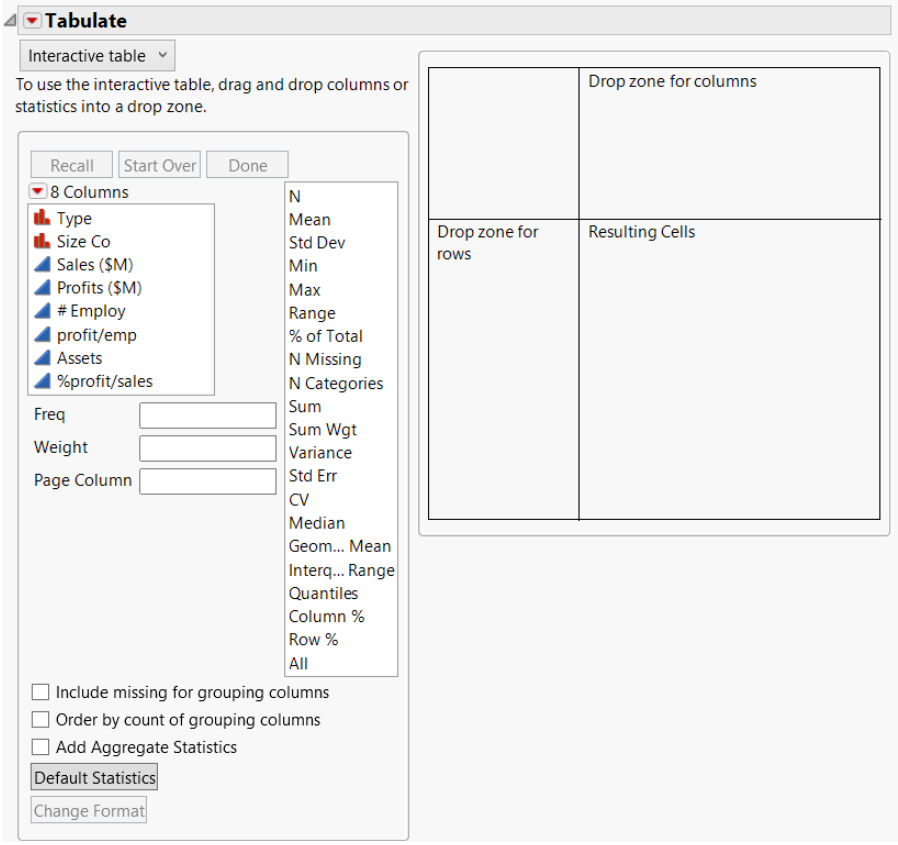
The summary table is linked to the source table. Selecting a row in the summary table also selects the corresponding rows in the source table.

Tabulate Example

Use the Tabulate command to drag columns into a workspace, creating summary statistics for each combination of grouping variables. This example shows you how to use Tabulate to create the same summary information that you just created using Summary.

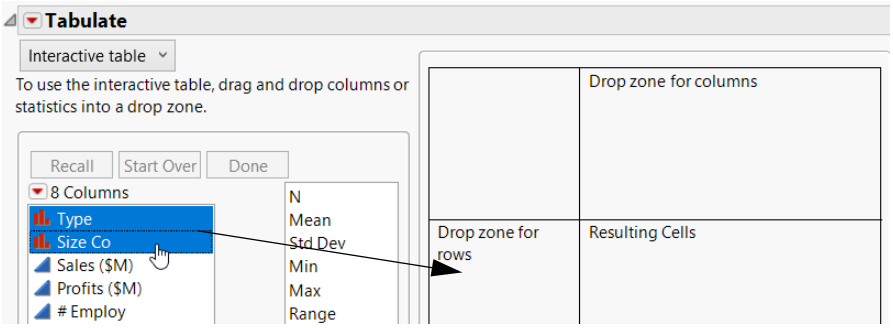
1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Tabulate**.

Figure 3.20 Tabulate Workspace



3. Select both Type and Size Co.
4. Drag and drop them into the **Drop zone for rows**.

Figure 3.21 Dragging Columns to the Row Zone



5. Right-click a heading and select **Nest Grouping Columns**.

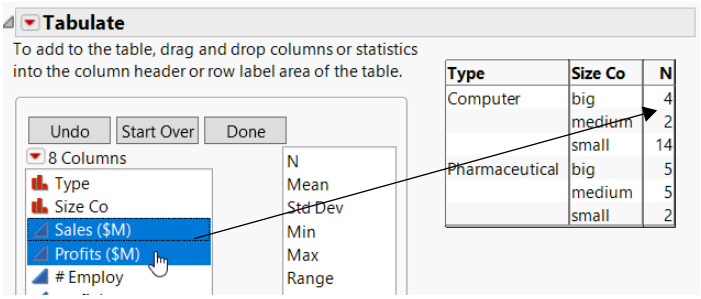
   The initial tabulation shows the number of rows per group.

Figure 3.22 Initial Tabulation

Type	Size Co	N
Computer	big	4
	medium	2
	small	14
Pharmaceutical	big	5
	medium	5
	small	2

6. Select both Sales ($M) and Profits ($M), and drag and drop them over the **N** in the table.

Figure 3.23 Adding Sales and Profit



The tabulation now shows the sum of Sales ($M) and the sum of Profits ($M) per group.

Figure 3.24 Tabulation of Sums

		Sales ($M)	Profits ($M)
Type	Size Co	Sum	Sum
Computer	big	82389.9	4359.7
	medium	6037.7	-171.5
	small	24612.8	629.1
Pharmaceutical	big	37370.2	4472.1
	medium	21305.3	3494.9
	small	2167.5	313.9

7. The final step is to change the sums to means. Right-click **Sum** (either of them) and select **Statistics** > **Mean**.

Figure 3.25 Final Tabulation

		Sales ($M)	Profits ($M)
Type	Size Co	Mean	Mean
Computer	big	20597.48	1089.9
	medium	3018.85	-85.75
	small	1758.06	44.94
Pharmaceutical	big	7474.04	894.42
	medium	4261.06	698.98
	small	1083.75	156.95

The means are the same as those obtained using the Summary command. Compare Figure 3.25 to Figure 3.19.

Examples of Creating Subsets

If you want to look closely at only part of your data table, you can create a subset. For example, suppose that you have already compared the sales and profits of big, medium, and small computer and pharmaceutical companies. Now you want to look at the sales and profits of only the medium-sized companies.

Creating a subset is a two-step process. First select the target data, and then extract the data into a new table.

Subset with the Subset Command

1. Select **Help > Sample Data Library** and open Companies.jmp.

Selecting the Rows and Columns That You Want to Subset

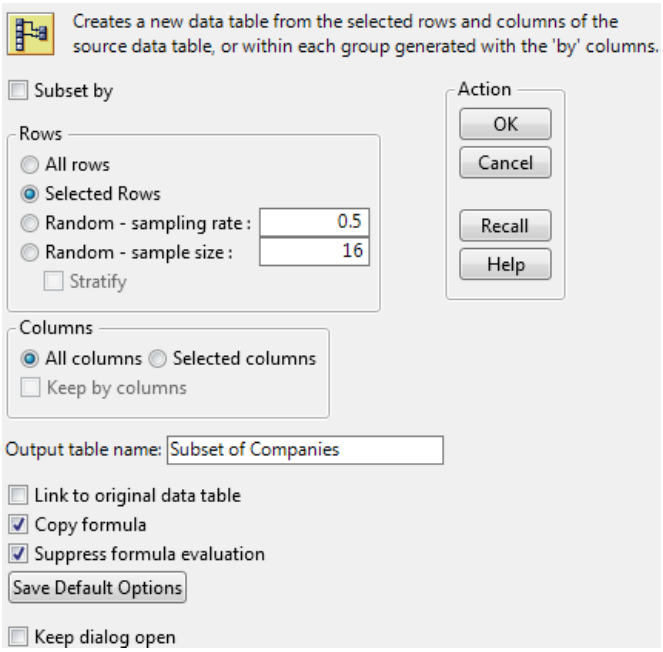
2. Select **Rows > Row Selection > Select Where**.
3. Select Size Co in the column list box on the left.
4. Enter medium in the text enter box.

5. Click **OK**.
6. Press Ctrl and select the Type, Sales ($M), and Profits ($M) columns.

Creating the Subset Table

7. Select **Tables > Subset** to launch the Subset window.

Figure 3.26 Subset Window



8. Select **Selected columns** to subset only the columns that you selected. You can also customize your subset table further by selecting additional options.
9. Click **OK**.

The resulting subset data table has seven rows and three columns. For more information about the Subset command, see *Using JMP*.

Subset with the Distribution Platform

Another way to create subsets uses the connection between platform results and data tables.

Example of Creating a Subset Using the Distribution Command

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze > Distribution**.

3. Select Type and click **Y, Columns**.
4. Click **OK**.
5. Double-click the histogram bar that represents Computer to create a subset table of the Computer companies.

Caution: This method creates a *linked* subset table. This means if you make any changes to the data in the subset table, the corresponding value changes in the source table.

Example of Joining Data Tables

Use the Join option to combine information from multiple data tables into a single data table. For example, suppose that you have a data table containing results from an experiment on popcorn yields. In another data table, you have the results of a second experiment on popcorn yields. To compare the two experiments or to analyze the trials using both sets of results, you need to have the data in the same table. Also, the experimental data was not entered into the data tables in the same order. One of the columns has a different name, and the second experiment is incomplete. This means that you cannot copy and paste from one table into another.

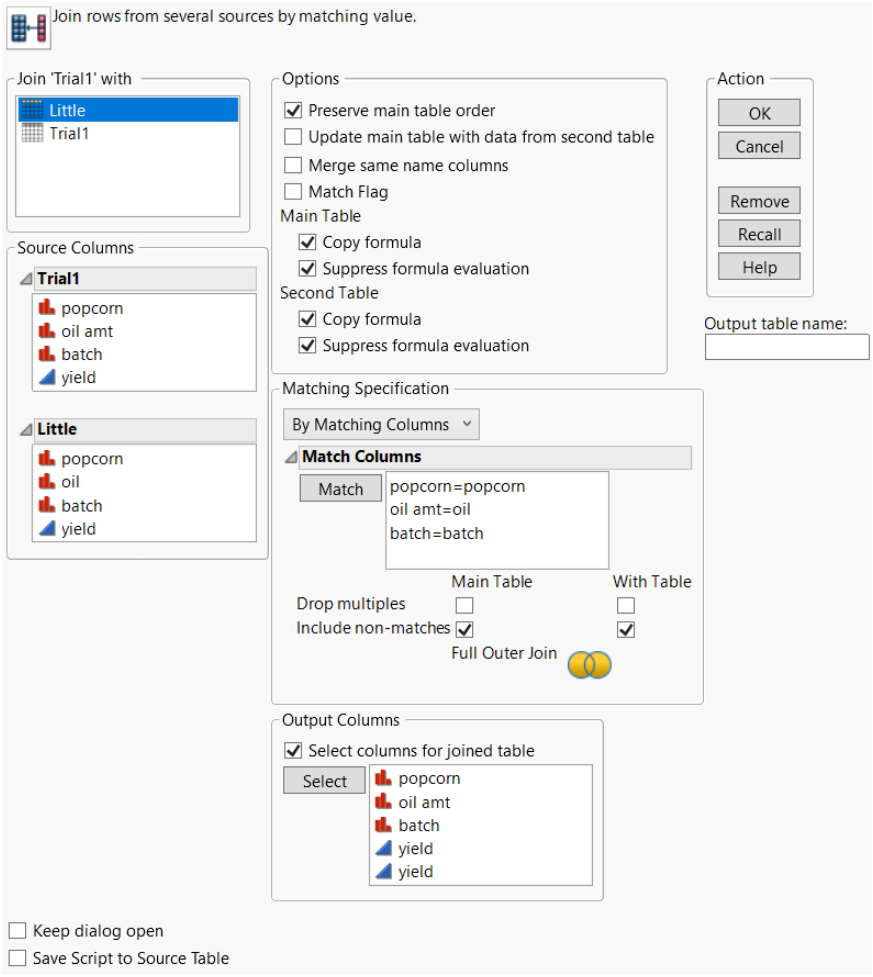
Example of Joining Two Data Tables

1. Select **Help > Sample Data Library** and open Trial1.jmp and Little.jmp.
2. Click Trial1.jmp to make it the active data table.
3. Select **Tables > Join**.
4. In the **Join 'Trial1' With** box, select Little.
5. From the **Matching Specification** menu, select **By Matching Columns** if it's not already selected.
6. In the **Source Columns** boxes, select popcorn in both boxes, and then click **Match**.
7. In the same way, match batch to batch and oil amt to oil in both boxes.

   Your matching columns do not have to have the same name.
8. Select **Include non-matches** for both tables.

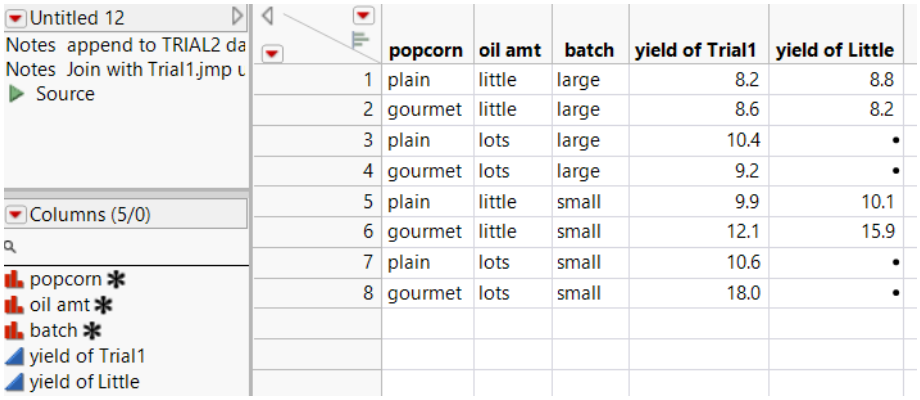
   Since one experiment is partial, you want to include all rows, including any with missing data.
9. To avoid duplicate columns, select the **Select columns for joined table** option.
10. From Trial1, select all four columns and click **Select**.
11. From Little, select only yield and click **Select**.

Figure 3.27 Completed Join Window



12. Click **OK**.

Figure 3.28 Joined Table



	popcorn	oil amt	batch	yield of Trial1	yield of Little
1	plain	little	large	8.2	8.8
2	gourmet	little	large	8.6	8.2
3	plain	lots	large	10.4	•
4	gourmet	lots	large	9.2	•
5	plain	little	small	9.9	10.1
6	gourmet	little	small	12.1	15.9
7	plain	lots	small	10.6	•
8	gourmet	lots	small	18.0	•

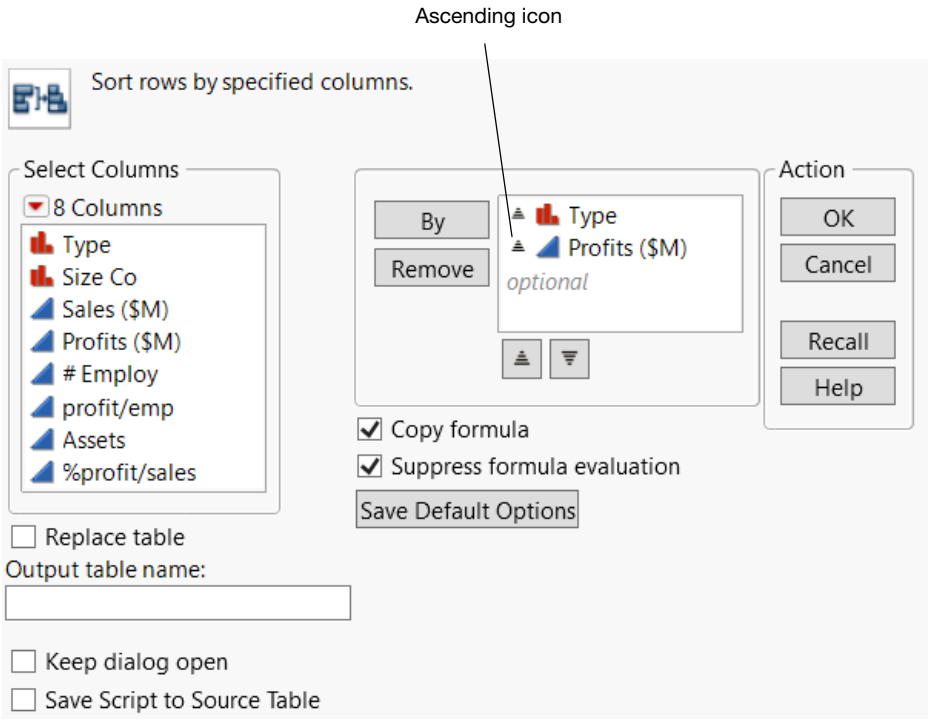
Example of Sorting Data

Use the Sort command to sort a data table by one or more columns in the data table. For example, look at financial data for computer and pharmaceutical companies. Suppose that you want to sort the data table by Type, then by Profits ($M). Also, you want Profits ($M) to be in descending order within each Type.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Tables** > **Sort**.
3. Select Type and click **By** to assign Type as a sorting variable.
4. Select Profits ($M) and click **By**.

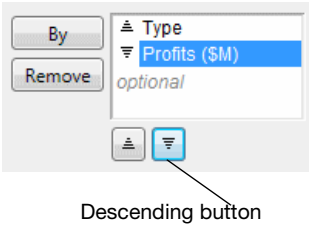
At this point, both variables are set to be sorted in ascending order. See the ascending icon next to the variables in Figure 3.29.

Figure 3.29 Sort Ascending Icon



5. To change Profits ($M) to sort in descending order, select Profits ($M) and click the descending button.

Figure 3.30 Change Profits to Descending



The icon next to Profits ($M) changes to descending.

6. Select the **Replace Table** check box.

   When selected, the **Replace Table** option tells JMP to sort the original data table instead of creating a new table with the sorted values. This option is not available if there are any open report windows created from the original data table. Sorting a data table with open report windows might change how some of the data is displayed in the report window, especially in graphs.

7. Click **OK**.

The data table is now sorted by type alphabetically, and by descending profit totals within type.

Chapter 4

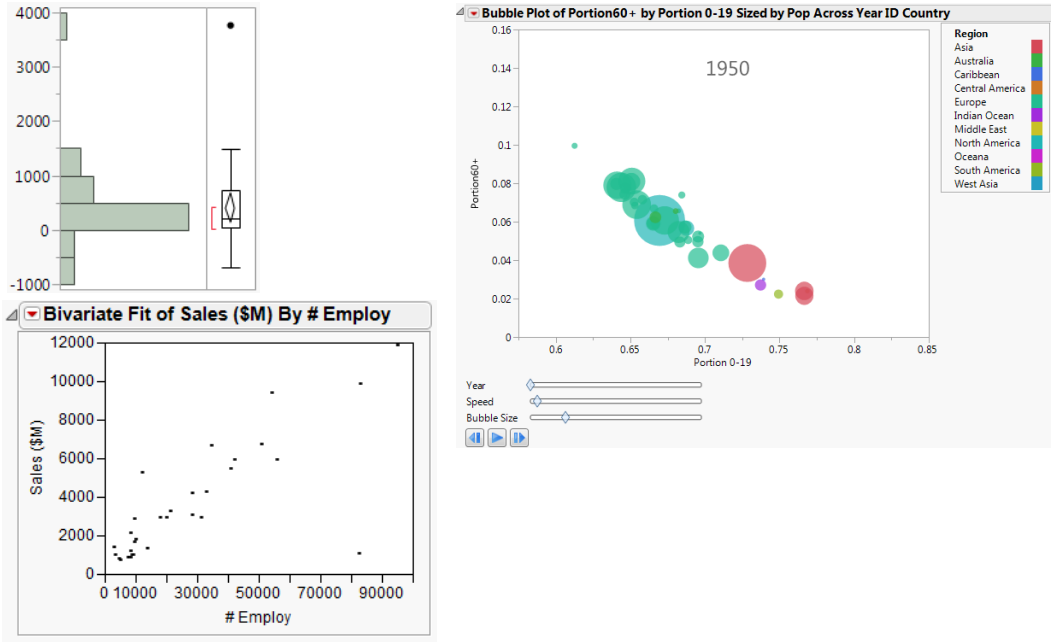
Visualize Your Data

Common Graphs

Visualizing your data is an important first step. The graphs described in this chapter help you discover important details about your data. For example, histograms show you the shape and range of your data, and help you find unusual data points.

This chapter presents several of the most common graphs and plots that enable you to visualize and explore data in JMP. This chapter is an introduction to some of JMP's graphical tools and platforms. Use JMP to visualize the distribution of single variables, or the relationships among multiple variables.

Figure 4.1 Visualizing Data with JMP



Contents

Analyze Single Variables in Univariate Graphs 97
Use Histograms for Continuous Variables 97
Use Bar Charts for Categorical Variables. 100
Compare Multiple Variables 102
Compare Multiple Variables Using Scatterplots 103
Compare Multiple Variables Using a Scatterplot Matrix 107
Compare Multiple Variables Using Side-by-Side Box Plots 110
Compare Multiple Variables Using Graph Builder 113
Compare Multiple Variables Using Bubble Plots 119
Compare Multiple Variables Using Overlay Plots 124
Compare Multiple Variables Using a Variability Chart. 129

Analyze Single Variables in Univariate Graphs

Single-variable graphs, or *univariate* graphs, let you look closely at one variable at a time. When you begin to look at your data, it's important to learn about each variable before looking at how the variables interact with each other. Univariate graphs let you visualize each variable individually.

This section covers two graphs that show the distribution of a single variable:

- "Use Histograms for Continuous Variables"
- "Use Bar Charts for Categorical Variables"

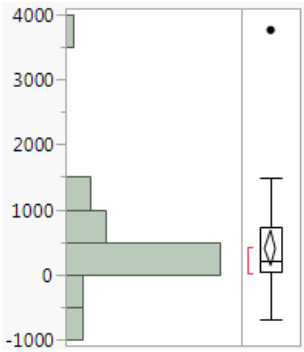
Use the Distribution platform to create both of these graphs. Distribution produces a graphical description and descriptive statistics for each variable.

Use Histograms for Continuous Variables

The histogram is one of the most useful graphical tools for understanding the distribution of a continuous variable. Use a histogram to find the following in your data:

- the average value and variation
- extreme values

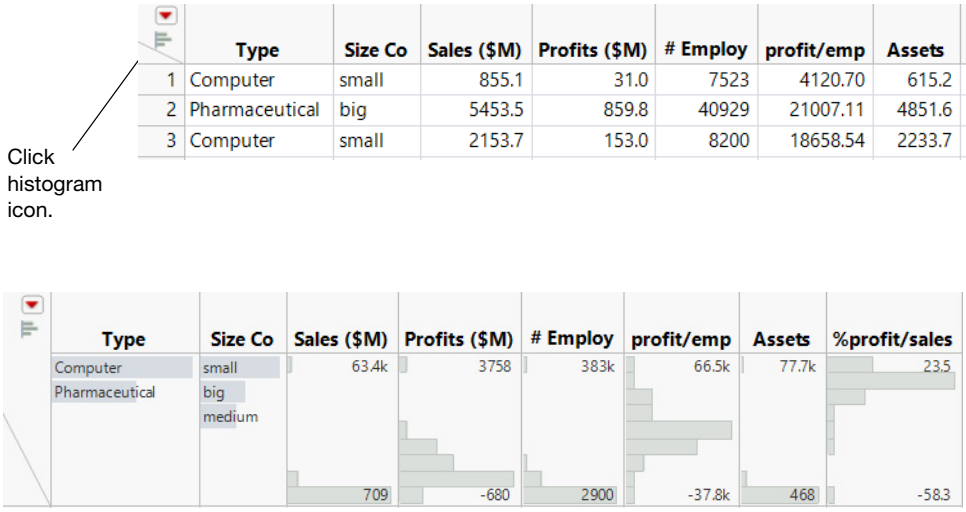
Figure 4.2 Example of a Histogram



Instant Histograms

You can view a histogram instantly by clicking the histogram icon in the column header. Histograms appear below the column header.

Figure 4.3 Instant Histograms



Scenario

This example uses the Companies.jmp data table, which contains data on profits for a group of companies.

A financial analyst wants to explore the following questions:

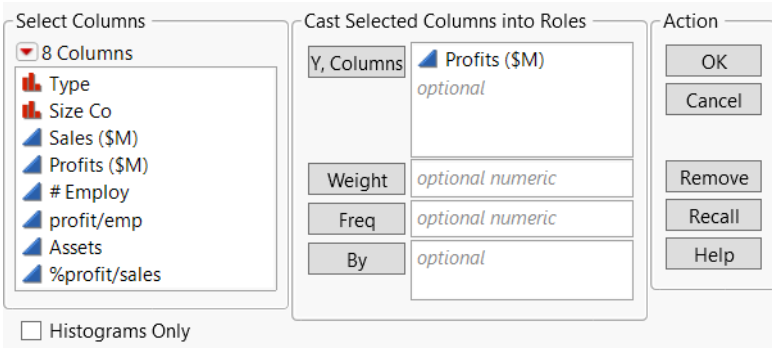
- Generally, how much profit does each company earn?
- What is the average profit?
- Are there any companies that earn either extremely high or extremely low profits compared to the other companies?

To answer these questions, use a histogram of Profits ($M).

Create the Histogram

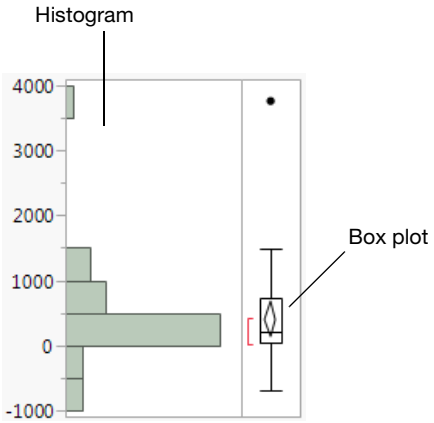
1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Distribution**.
3. Select Profits ($M) and click **Y, Columns**.

Figure 4.4 Distribution Window for Profits ($M)



4. Click **OK**.

Figure 4.5 Histogram of Profits ($M)



Interpret the Histogram

The histogram provides these answers:

- Most companies' profits are between $-1000 and $1500.

  All the bars except for one are located in this range. Also, more companies' profits range from $0 to $500 than any other range. The bar representing that range is much longer than the others.

- The average profit is a little less than $500.

  The middle of the diamond in the box plot indicates the mean value. In this case, the mean is slightly lower than the $500 mark.

- One company has significantly higher profits than the others, and might be an *outlier*. An outlier is a data point that is separated from the general pattern of the other data points.

This outlier is represented by a single, very short bar at the top of the histogram. The bar is small and represents a small group (in this case, a single company), and it is widely separated from the rest of the histogram bars.

In addition to the histogram, this report includes the following:

- The box plot, which is another graphical summary of the data. For detailed information about the box plot, see *Essential Graphing*.
- **Quantiles** and **Summary Statistics** reports. These reports are discussed in "Analyze Distributions".

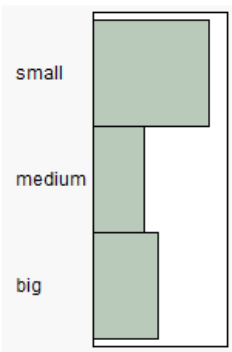
Interact with the Histogram

Data tables and reports are all connected in JMP. Click a histogram bar to select the corresponding rows in the data table.

Use Bar Charts for Categorical Variables

Use a bar chart to visualize the distribution of a categorical variable. A bar chart looks similar to a histogram, since they both have bars that correspond to the levels of a variable. A bar chart shows a bar for every level of the variable, whereas the histogram shows a range of values for the variable.

Figure 4.6 Example of a Bar Chart



Scenario

This example uses the Companies.jmp data table, which contains data on the size and type of a group of companies.

A financial analyst wants to explore the following questions:

- What is the most common type of company?
- What is the most common size for a company?

To answer these questions, use bar charts of Type and Size Co.

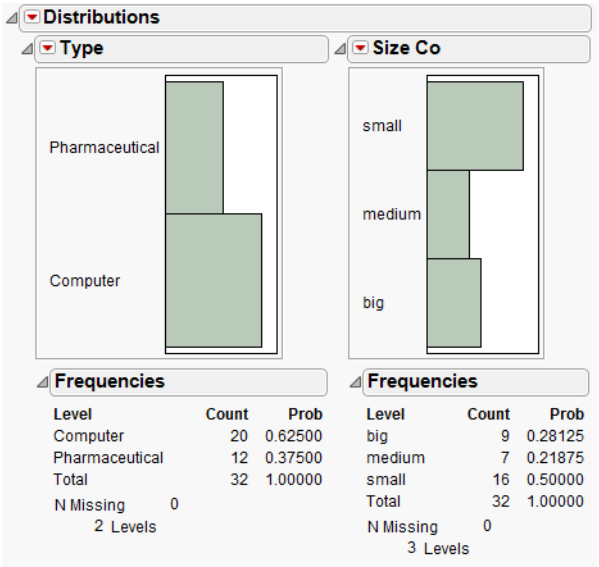
Create the Bar Chart

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Distribution**.
3. Select Type and Size Co and click **Y, Columns**.
4. Click **OK**.

Figure 4.7 Bar Charts of Type and Size Co

Bar Charts

Summary Information



Interpret the Bar Charts

The bar charts provide these answers:

- There are more computer companies than pharmaceutical companies.

  The bar that represents computer companies is larger than the bar that represents pharmaceutical companies.
- The most common company size is small.

  The bar that represents small companies is larger than the bars that represent medium and big companies.

The additional summary output gives detailed frequencies. This report is discussed in "Distributions of Categorical Variables".

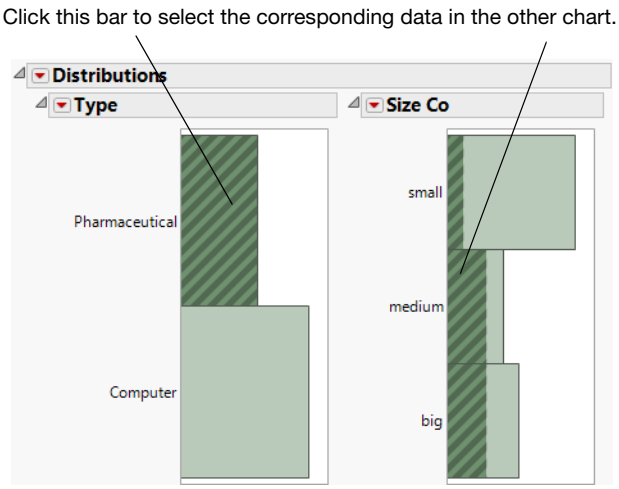
Interact with the Bar Charts

As is the case with histograms, click individual bars to highlight rows of the data table. If more than one graph is created, clicking on a bar in one bar chart highlights the corresponding bar or bars in the other bar chart.

For example, suppose that you want to see the distribution of company size for the pharmaceutical companies. Click the Pharmaceutical bar in the Type bar chart, and the pharmaceutical companies are highlighted on the Size Co bar chart. Figure 4.8 shows that although most companies in this data table are small, most of the pharmaceutical companies are medium or big.

Also, the corresponding rows in the data table are selected.

Figure 4.8 Clicking Bars



Compare Multiple Variables

Use multiple-variable graphs to visualize the relationships and patterns between two or more variables. This section covers the following graphs:

Table 4.1 Multiple-Variable Graphs

"Compare Multiple Variables Using Scatterplots"	Use scatterplots to compare two continuous variables.

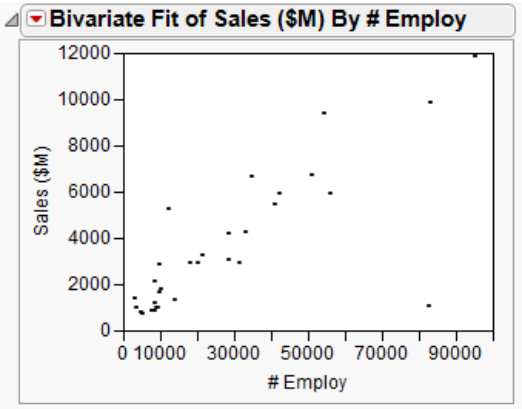
Table 4.1 Multiple-Variable Graphs *(Continued)*

"Compare Multiple Variables Using a Scatterplot Matrix"	Use scatterplot matrices to compare several pairs of continuous variables.
"Compare Multiple Variables Using Side-by-Side Box Plots"	Use side-by-side box plots to compare one continuous and one categorical variable.
"Compare Multiple Variables Using a Variability Chart"	Use variability charts to compare one continuous Y variable to one or more categorical X variables. Variability charts show differences in means and variability across several categorical X variables.
"Compare Multiple Variables Using Graph Builder"	Use Graph Builder to create and change graphs interactively.
"Compare Multiple Variables Using Overlay Plots"	Use overlay plots to compare one or more variables on the Y-axis to another variable on the X-axis. Overlay plots are especially useful if the X variable is a time variable, because you can compare how two or more variables change across time.
"Compare Multiple Variables Using Bubble Plots"	Bubble plots are specialized scatterplots that use color and bubble sizes to represent up to five variables at once. If one of your variables is a time variable, you can animate the plot to see your other variables change through time.

Compare Multiple Variables Using Scatterplots

The scatterplot is the simplest of all the multiple-variable graphs. Use scatterplots to determine the relationship between two continuous variables and to discover whether two continuous variables are *correlated*. Correlation indicates how closely two variables are related. When you have two variables that are highly correlated, one might influence the other. Or, both might be influenced by other variables in a similar way.

Figure 4.9 Example of a Scatterplot



Scenario

This example uses the Companies.jmp data table, which contains sales figures and the number of employees of a group of companies.

A financial analyst wants to explore the following questions:

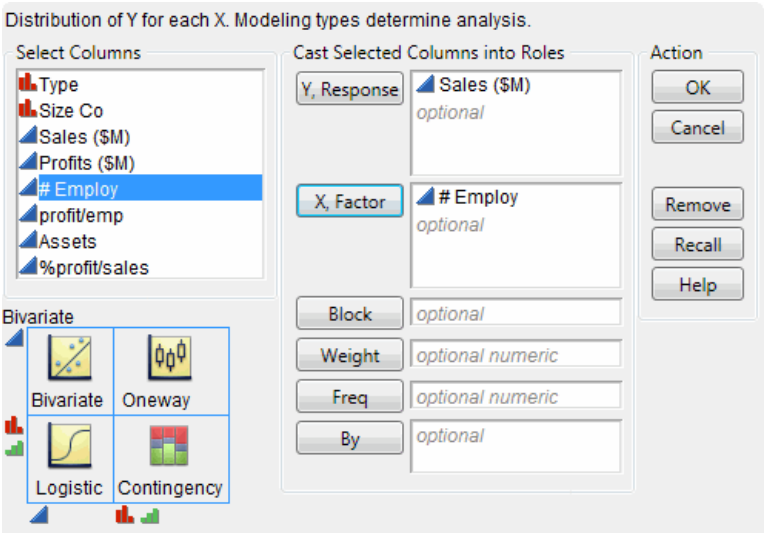
- What is the relationship between sales and the number of employees?
- Does the amount of sales increase with the number of employees?
- Can you predict average sales from the number of employees?

To answer these questions, use a scatterplot of Sales ($M) versus # Employ.

Create the Scatterplot

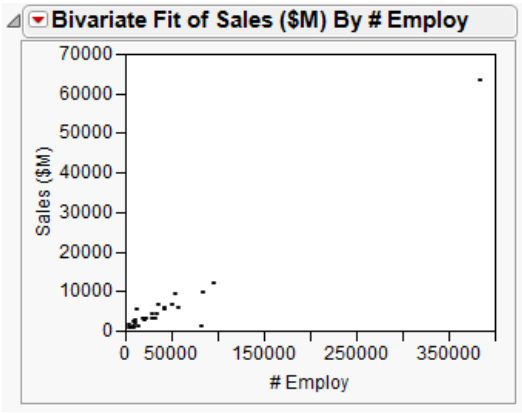
1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Sales ($M) and **Y, Response**.
4. Select # Employ and **X, Factor**.

Figure 4.10 Fit Y by X Window



5. Click **OK**.

Figure 4.11 Scatterplot of Sales ($M) versus # Employ



Interpret the Scatterplot

One company has a large number of employees and high sales, represented by the single point at the top right of the plot. The distance between this data point and all the rest makes it difficult to visualize the relationship between the rest of the companies. Remove the point from the plot and re-create the plot by following these steps:

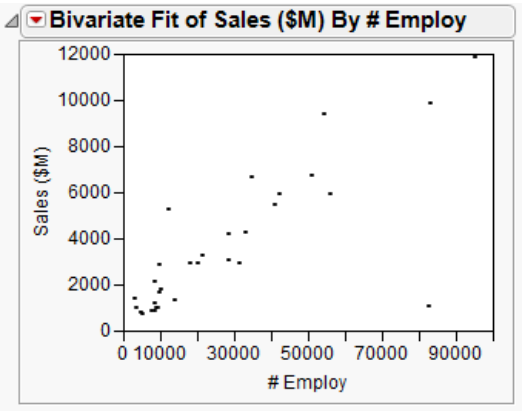
1. Click the point to select it.

2. Select **Rows > Hide and Exclude**. The data point is hidden and no longer included in calculations.

   Note: The difference between hiding and excluding is important. Hiding a point removes it from any graphs but statistical calculations continue to use the point. Excluding a point removes it from any statistical calculations but does not remove it from graphs. When you both hide and exclude a point, you remove it from all calculations and from all graphs.

3. To re-create the plot without the outlier, click the Bivariate red triangle and select **Redo > Redo Analysis**. You can close the original report window.

Figure 4.12 Scatterplot with the Outlier Removed



The updated scatterplot provides these answers:

- There is a relationship between the sales and the number of employees.

  The data points have a discernible pattern. They are not scattered randomly throughout the graph. You could draw a diagonal line that would be near most of the data points.

- Sales do increase with the number of employees, and the relationship is linear.

  If you drew that diagonal line, it would slope from bottom left to top right. This slope shows that as the number of employees increases (left to right on the bottom axis), sales also increases (bottom to top on the left axis). A straight line would be near most of the data points, indicating a linear relationship. If you would have to curve your line to be near the data points, there would still be a relationship (because of the pattern of the points). However, that relationship would not be linear.

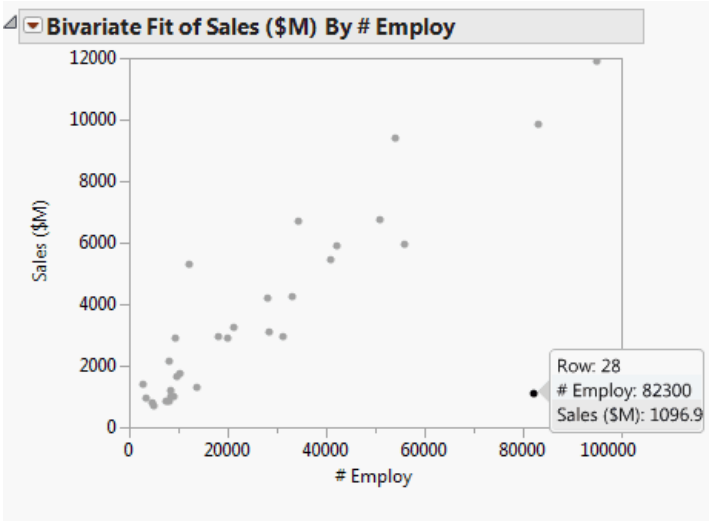
- You can predict average sales from the number of employees.

  The scatterplot shows that sales generally increase as the number of employees does. You could predict the sales for a company if you knew only the number of employees of that company. Your prediction would be on that imaginary line. It would not be exact, but it would approximate the real sales.

Interact with the Scatterplot

As with other JMP graphics, the scatterplot is interactive. Hover over the point in the bottom right corner with the mouse to reveal the row number and the x and y values.

Figure 4.13 Hover Over a Point



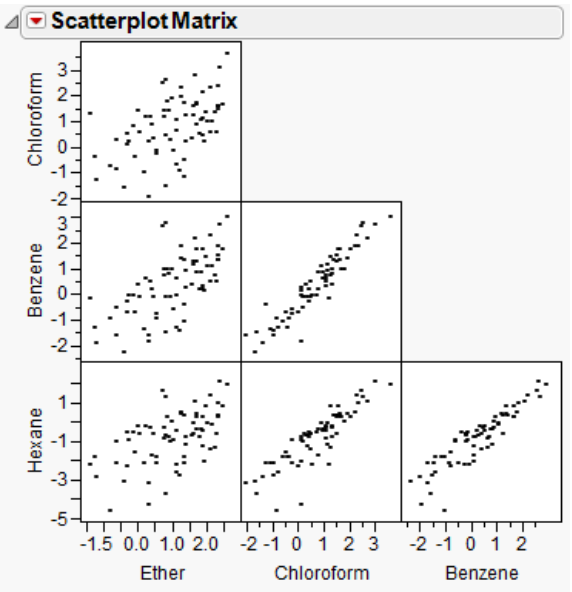
Click a point to highlight the corresponding row in the data table. Select multiple points by doing one of the following:

- Click and drag with the cursor around the points. This selects points in a rectangular area.
- Select the lasso tool, and then click and drag around multiple points. The lasso tool selects an irregularly shaped area.

Compare Multiple Variables Using a Scatterplot Matrix

A scatterplot matrix is a collection of scatterplots organized into a grid (or matrix). Each scatterplot shows the relationship between a pair of variables.

Figure 4.14 Example of a Scatterplot Matrix



Scenario

This example uses the Solubility.jmp data table, which contains data for solubility measurements for 72 different solutes.

A lab technician wants to explore the following questions:

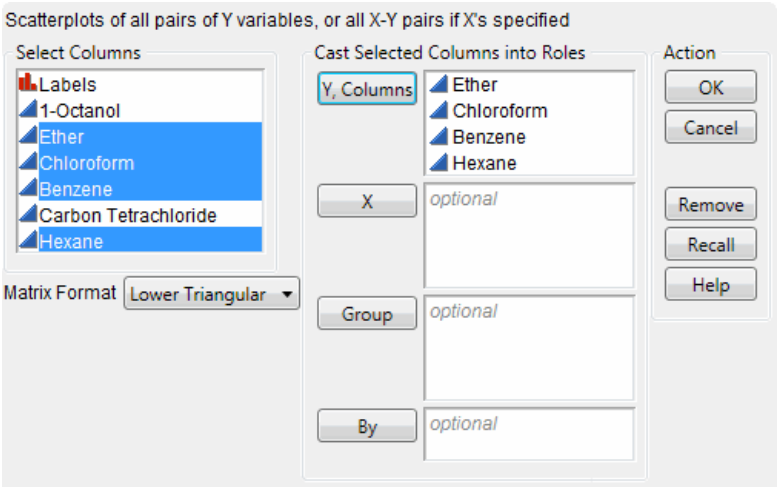
- Is there a relationship between any pair of chemicals? (There are six possible pairs.)
- Which pair has the strongest relationship?

To answer these questions, use a scatterplot matrix of the four solvents.

Create the Scatterplot Matrix

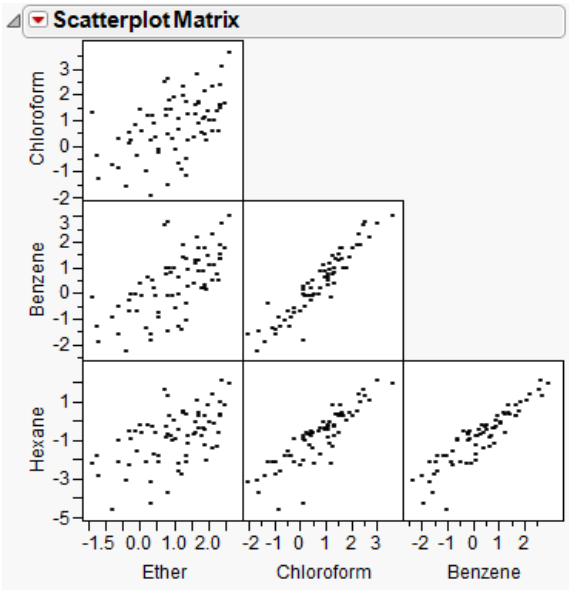
1. Select **Help > Sample Data Library** and open Solubility.jmp.
2. Select **Graph** > **Scatterplot Matrix**.
3. Select Ether, Chloroform, Benzene, and Hexane, and click **Y, Columns**.

Figure 4.15 Scatterplot Matrix Window



4. Click **OK**.

Figure 4.16 Scatterplot Matrix



Interpret the Scatterplot Matrix

The scatterplot matrix provides these answers:

- All six pairs of variables are positively correlated.

  As one variable increases, the other variable increases too.

- The strongest relationship appears to be between Benzene and Chloroform.

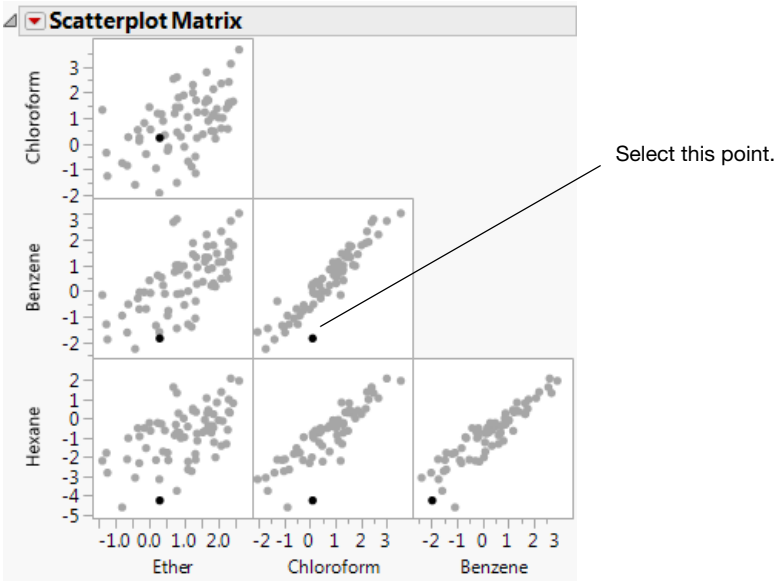
  The data points in the scatterplot for Benzene and Chloroform are the most tightly clustered along an imaginary line.

Interact with the Scatterplot Matrix

If you select a point in one scatterplot, it is selected in all the other scatterplots.

For example, if you select a point in the Benzene versus Chloroform scatterplot, the same point is selected in the other five plots.

Figure 4.17 Selected Points

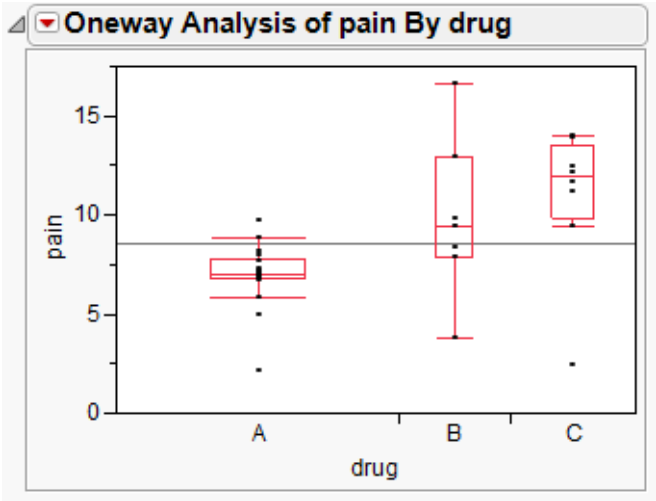


Compare Multiple Variables Using Side-by-Side Box Plots

Side-by-side box plots show the following:

- the relationship between one continuous variable and one categorical variable
- differences in the continuous variable across levels of the categorical variable

Figure 4.18 Example of Side-by-Side Box Plots



Scenario

This example uses the Analgesics.jmp data table, which contains data on pain measurements taken on patients using three different drugs.

A researcher wants to explore the following questions:

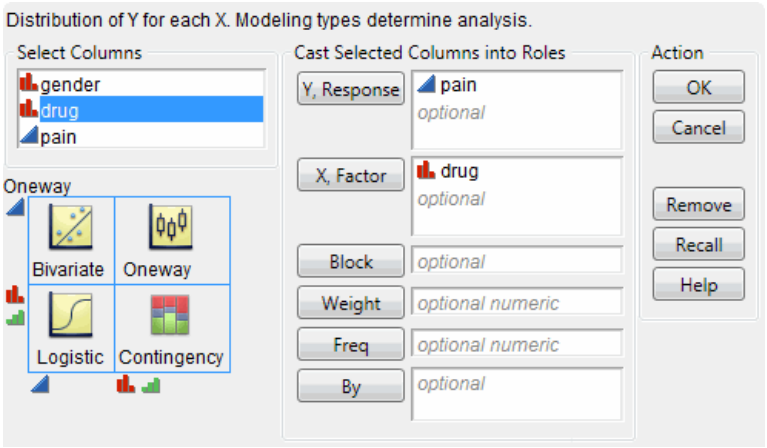
- Are there differences in the average amount of pain control among the drugs?
- Does the *variability* in the pain control given by each drug differ? A drug with high variability would not be as reliable as a drug with low variability.

To answer these questions, use a side-by-side box plot for the pain levels and the drug categories.

Create the Side-by-Side Box Plots

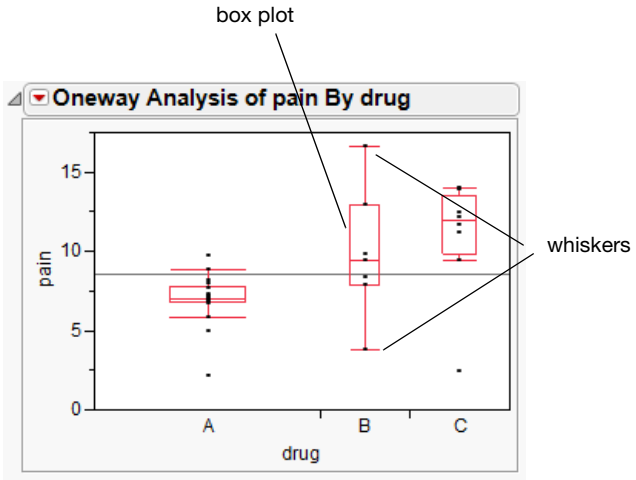
1. Select **Help > Sample Data Library** and open Analgesics.jmp.
2. Select **Analyze** > **Fit Y by X**.
3. Select pain and click **Y, Response**.
4. Select drug and click **X, Factor**.

Figure 4.19 Fit Y by X Window



5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of pain By drug and select **Display Options > Box Plots**.

Figure 4.20 Side-by-Side Box Plots



Interpret the Side-by-Side Box Plots

Box plots are designed according to the following principles:

- The line through the box represents the median.
- The middle half of the data is within the box.
- The majority of the data falls between the ends of the whiskers.

- A data point outside the whiskers might be an outlier.

The box plots in Figure 4.20 show these answers:

- There is evidence to believe that patients on drug A feel less pain, since the box plot for drug A is lower on the pain scale than the others.
- Drug B appears to have higher variability than Drugs A and C, since the box plot is taller.

There is one point for drug C that is a lot lower than the other points for drug C. Hover over the lower point to see that it is row 26 of the data table. That point looks like it is more similar to the data in drug group A or B. The information in row 26 deserves investigation. There might have been a typographical error when the data was recorded.

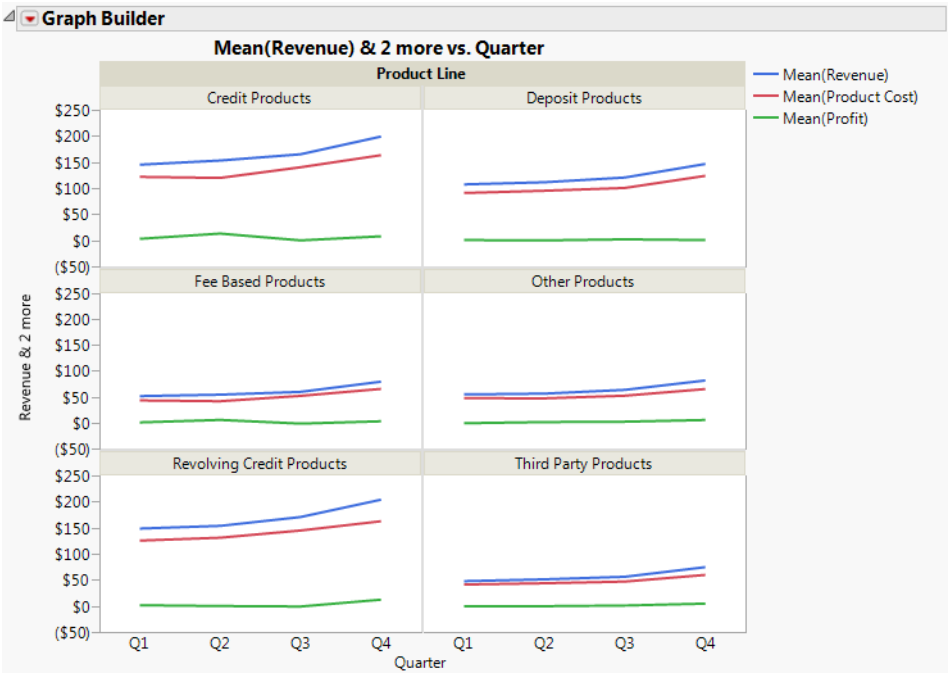
Compare Multiple Variables Using Graph Builder

Use Graph Builder to interactively create and modify graphs. Most graphs in JMP are created by launching a platform and specifying variables. If you want to create a different type of graph, you launch a specific platform from the Graph menu. However, with Graph Builder, you can change the variables and change the type of graph at any time.

Use Graph Builder to accomplish the following tasks:

- Change variables by dragging and dropping them in and out of the graph.
- Create a different type of graph with a few mouse clicks.
- Partition the graph horizontally or vertically.

Figure 4.21 Example of a Graph That Was Created with Graph Builder



Note: Only some of the Graph Builder features are covered here. See *Essential Graphing*.

Scenario

This example uses the Profit by Product.jmp data table, which contains profit data for multiple product lines.

A business analyst wants to explore the following question:

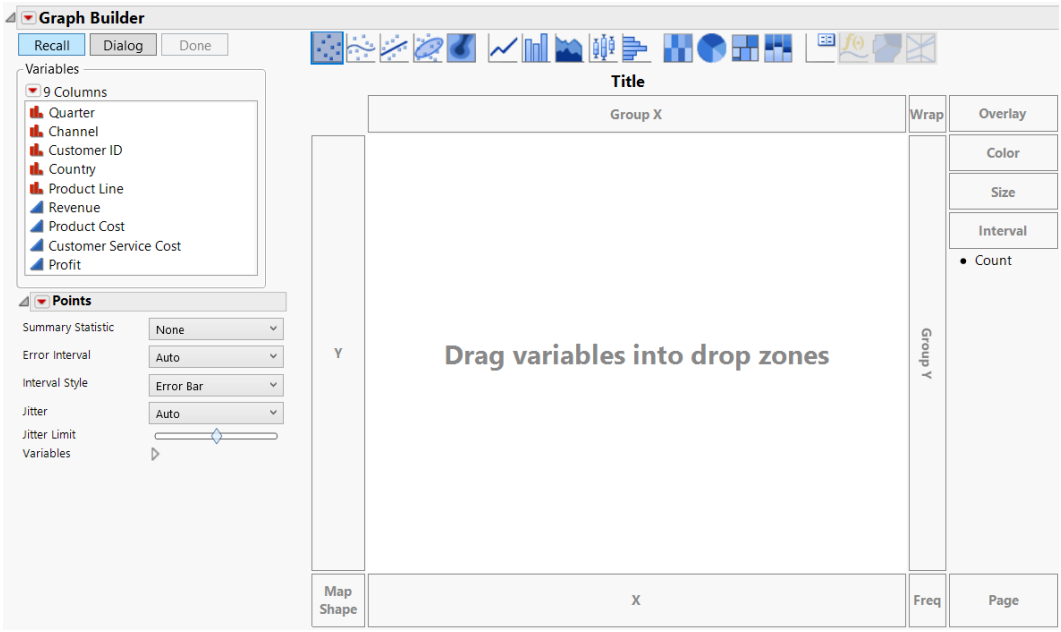
- How is the profitability different between product lines?

To answer this question, use a line plot that displays revenue, product cost, and profit data across different product lines.

Create the Graph

1. Select **Help > Sample Data Library** and open Profit by Product.jmp.
2. Select **Graph** > **Graph Builder**.

Figure 4.22 Graph Builder Workspace

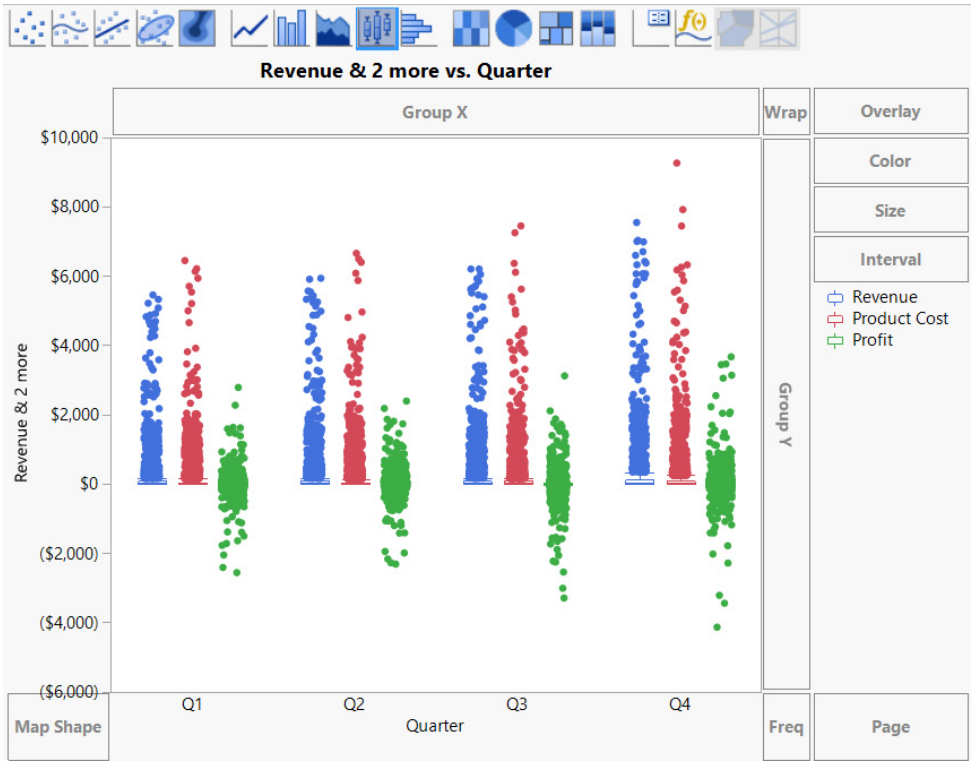


3. Click Quarter and then drag and drop it onto the X zone to assign Quarter as the X variable.
4. Click Revenue, Product Cost, and Profit, and drag and drop them onto the Y zone to assign all three variables as Y variables.

   The X and Y zones are now axes.

   Note: You can also click variables and then click a zone to assign them. However, after a zone becomes an axis, drag and drop additional variables onto the axis rather than clicking on the variables and axis.

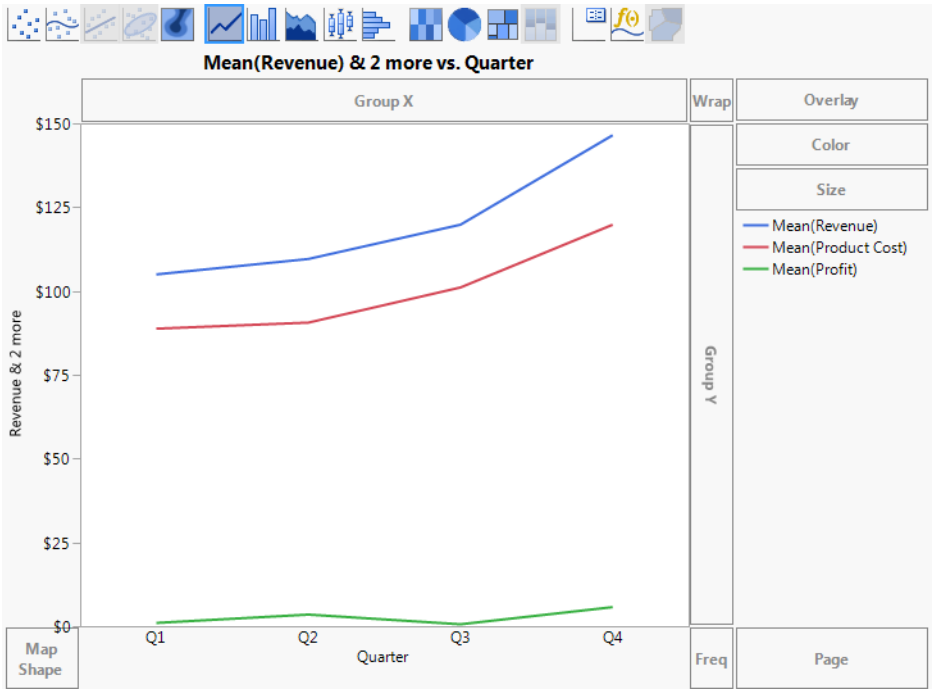
Figure 4.23 After Adding Y and X Variables



Based on the variables that you are using, Graph Builder shows side-by-side box plots.

5. To change the box plots to a line plot, click the Line icon .

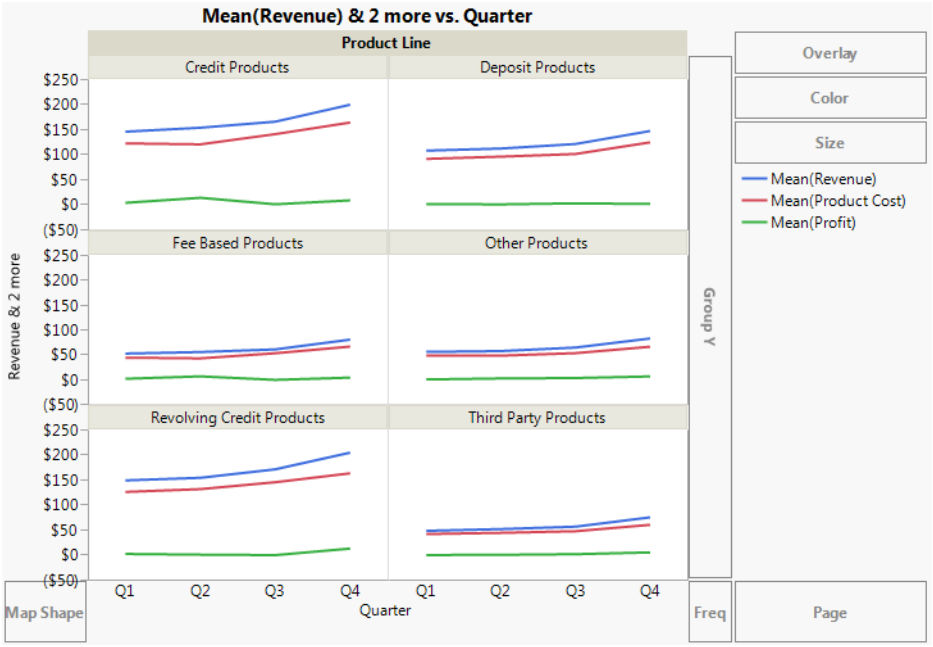
Figure 4.24 Line Plot



6. To create a separate chart for each product, click Product Line, and drag and drop it into the **Wrap** zone.

   A separate line plot is created for each product.

Figure 4.25 Final Line Plots



Interpret the Graph

Figure 4.25 shows revenue, cost, and profit broken down by product line. The business analyst was interested in seeing the difference in profitability between product lines. The line plots in Figure 4.25 can provide some answers:

- Credit products, deposit products, and revolving credit products produce more revenue than fee-based products, third-party products, and other products.
- However, the profits of all the product lines are similar.

The data table also includes data on sales channels. The business analyst wants to see how revenue, product cost, and profit differ between different sales channels.

1. To remove Product Line from the graph, click the title of the graph (Product Line) and drag and drop it into any empty area within Graph Builder.
2. To add Channel as the wrap variable, click Channel and drag and drop it into the **Wrap** zone.

Figure 4.26 Line Plots Showing Sales Channels

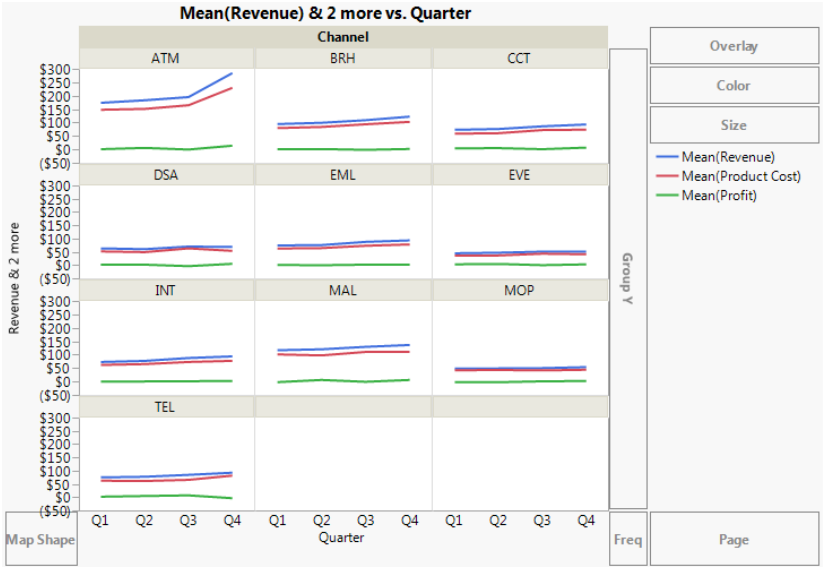
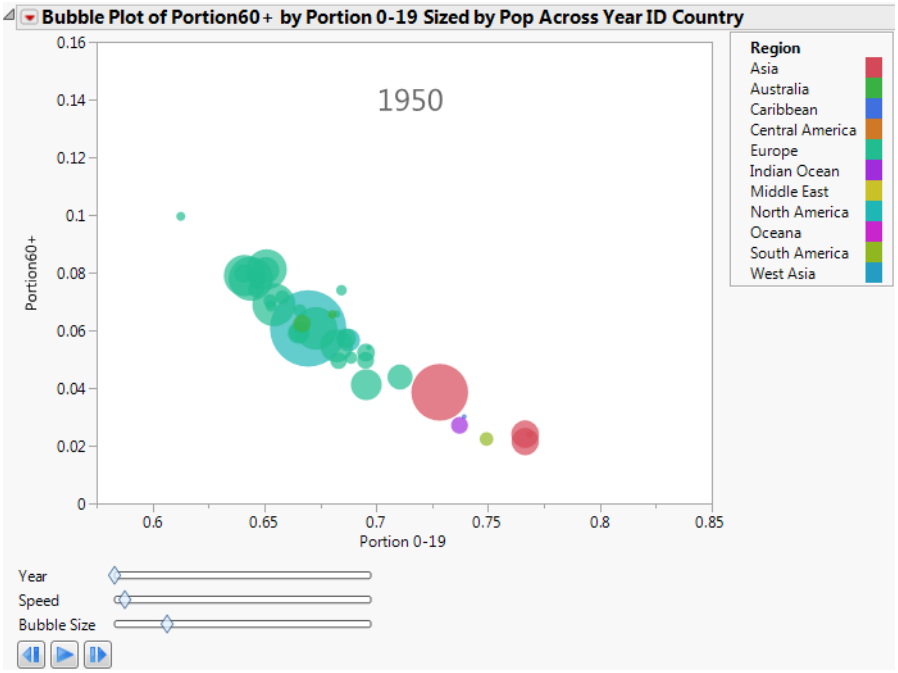


Figure 4.26 provides this answer: revenue and product cost for ATMs are the highest and are growing the most quickly.

Compare Multiple Variables Using Bubble Plots

A bubble plot is a scatterplot that represents its points as bubbles. You can change the size and color of the bubbles, and even animate them over time. With the ability to represent up to five dimensions (*x* position, *y* position, size, color, and time), a bubble plot can produce dramatic visualizations and make data exploration easy.

Figure 4.27 Example of a Bubble Plot



Scenario

This example uses the PopAgeGroup.jmp data table, which contains population statistics for 116 countries or territories between the years 1950 to 2004. Total population numbers are broken out by age group, and not every country has data for every year.

A sociologist wants to explore the following question:

- Is the age of the population of the world changing?

To answer this question, look at the relationship between the oldest (more than 59) and the youngest (younger than 20) portions of the population. Use a bubble plot to determine how this relationship changes over time.

Create the Bubble Plot

1. Select **Help > Sample Data Library** and open PopAgeGroup.jmp.
2. Select **Graph** > **Bubble Plot**.
3. Select Portion60+ and click **Y**.

   This corresponds to the *Y* variable on the bubble plot.
4. Select Portion 0-19 and click **X**.

This corresponds to the *X* variable on the bubble plot.

5. Select Country and click **ID**.

   Each unique level of the ID variable is represented by a bubble on the plot.

6. Select Year and click **Time**.

   This controls the time indexing when the bubble plot is animated.

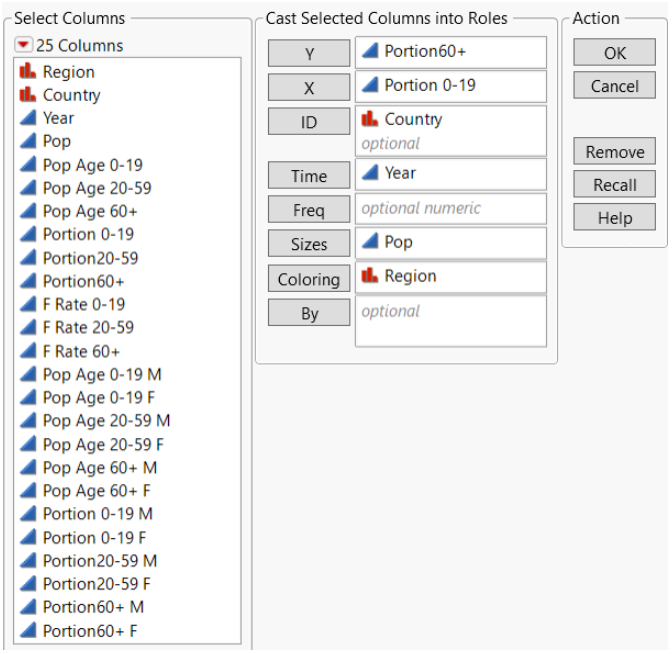
7. Select Pop and click **Sizes**.

   This controls the size of the bubbles.

8. Select Region and click **Coloring**.

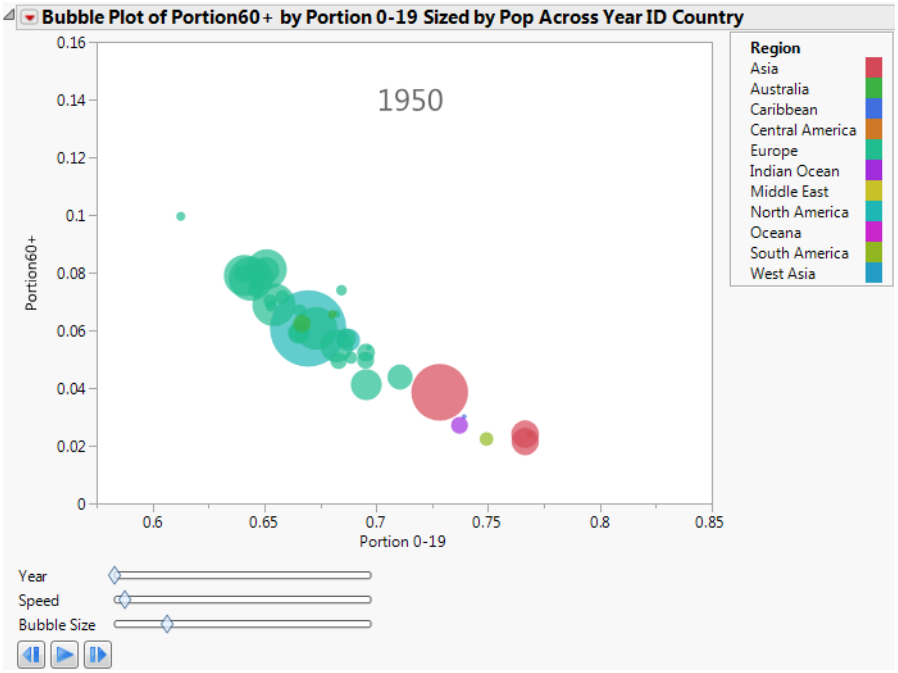
   Each level of the Coloring variable is assigned a unique color. So in this example, all the bubbles for countries located in the same region have the same color. The bubble colors that appear in Figure 4.29 are the JMP default colors.

Figure 4.28 Bubble Plot Launch Window



9. Click **OK**.

Figure 4.29 Initial Bubble Plot



Interpret the Bubble Plot

Because the time variable (in this case, year) starts in 1950, the initial bubble plot shows the data for 1950. Animate the bubble plot to cycle through all the years by clicking the play/pause button. Each successive bubble plot shows the data for that year. The data for each year determines the following:

- The *X* and *Y* coordinates
- The bubble's sizes
- The bubble's coloring
- Bubble aggregation

Note: For detailed information about how the bubble plot aggregates information across multiple rows, see *Essential Graphing*.

The bubble plot for 1950 shows that if a country's proportion of people younger than 20 is high, then the proportion of people more than 59 is low.

Click the play/pause button to animate the bubble plot through the range of years. As time progresses, the Portion 0-19 decreases and the Portion60+ increases.

plays the animation, turns to a pause button after you click it.

pauses the animation.

manually controls the animation back one unit of time.

manually controls the animation forward one unit of time.

Year Changes the time index manually.

Speed Controls the speed of the animation.

Bubble Size Controls the absolute sizes of the bubbles, while maintaining the relative sizes

The sociologist wanted to know how the age of the world's population is changing. The bubble plot indicates that the population of the world is getting older.

Interact with the Bubble Plot

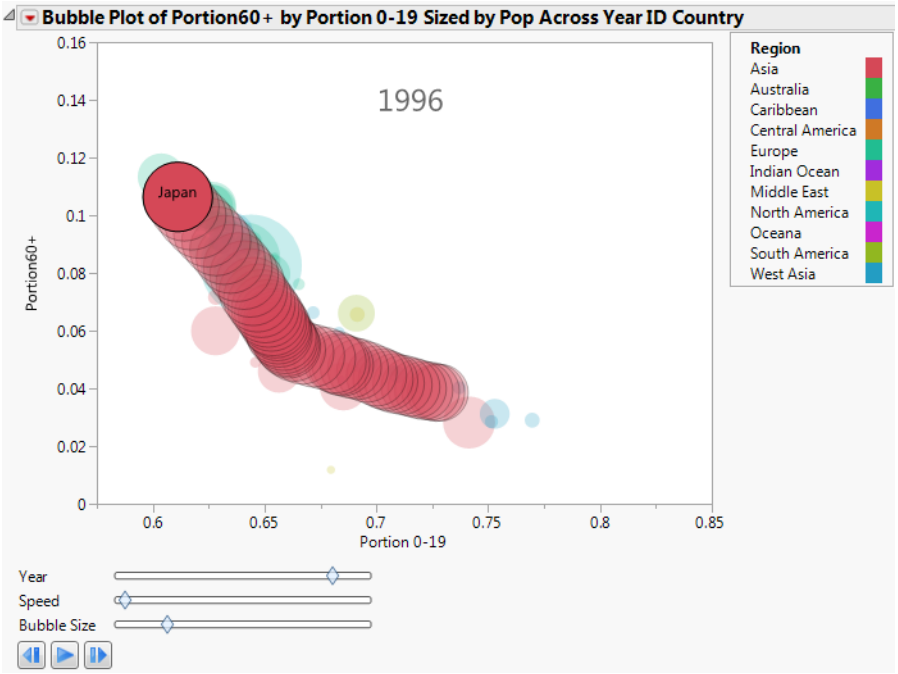
Click to select a bubble to see the trend for that bubble over time. For example, in the 1950 plot, the large bubble in the middle is Japan.

To See the Pattern of Population Changes in Japan through the Years

1. Click in the middle of the Japan bubble to select it.
2. Click the Bubble Plot red triangle and select **Trail Bubbles > Selected**.
3. Click the play button.

As the animation progresses through time, the Japan bubble leaves a trail of bubbles that illustrates its history.

Figure 4.30 Japan's History of Population Shifts



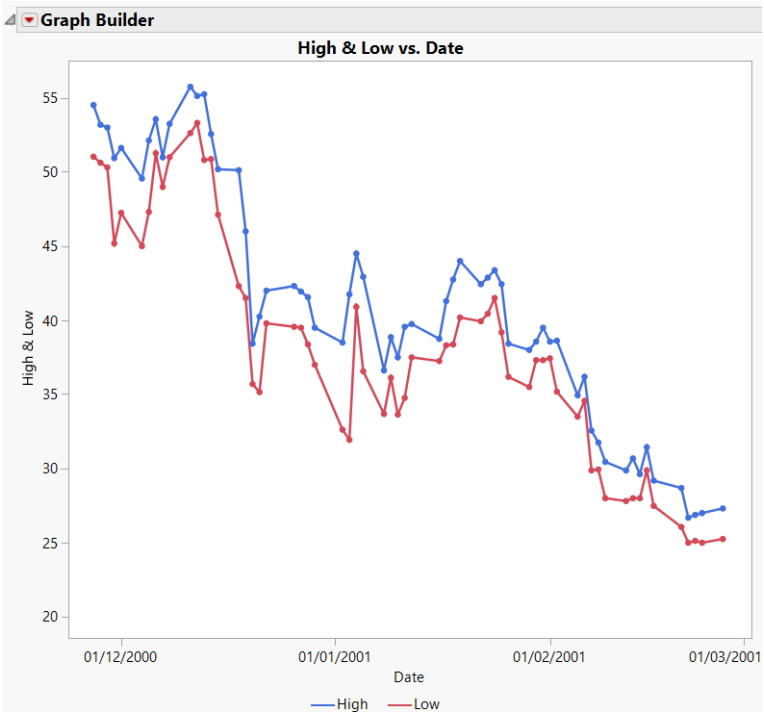
Focusing on the Japan bubble, you can see the following over time:

- The proportion of the population 19 years old or less decreased.
- The proportion of the population 60 years old or more increased.

Compare Multiple Variables Using Overlay Plots

Like scatterplots, overlay plots show the relationship between two or more variables. However, if one of the variables is a time variable, an overlay plot shows trends across time better than scatterplots do.

Figure 4.31 Example of an Overlay Plot



Note: To plot data over time, you can also use bubble plots, control charts, and variability charts. For more information about Graph Builder and bubble plots, see *Essential Graphing*. See and *Quality and Process Methods* for information about control charts and variability charts.

Scenario

This example uses the Stock Prices.jmp data table, which contains data on the price of a stock over a three-month period.

A potential investor wants to explore the following questions:

- Has the stock's closing price changed over the past three months?

  To answer this question, use an overlay plot of the stock's closing price over time.
- How do the stock's high and low prices relate to each other?

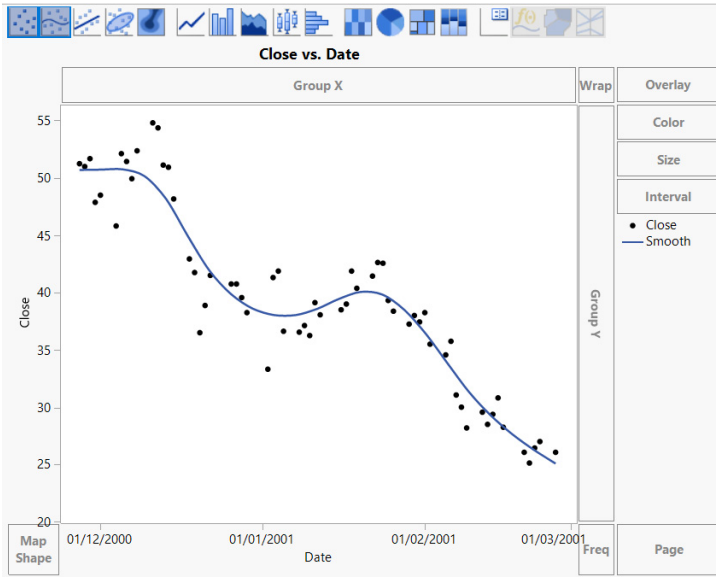
  To answer this question, use another overlay plot of the stock's high and low prices over time.

Create the first overlay plot to answer the first question, and then create a second overlay plot to answer the second question.

Create the Overlay Plot of the Stock's Price over Time

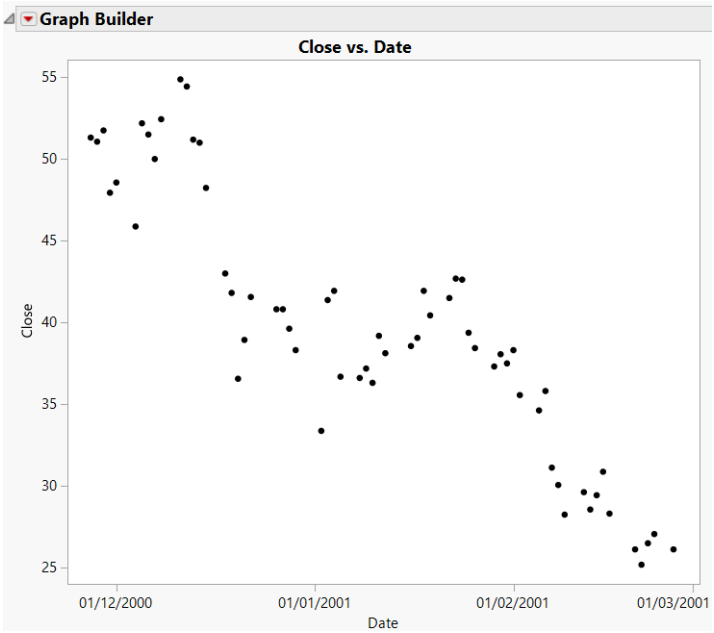
1. Select **Help > Sample Data Library** and open Stock Prices.jmp.
2. Select **Graph > Graph Builder**.
3. Select Close and click **Y**.
4. Select Date and click **X**.

Figure 4.32 Overlay Plot with Smoother



5. Press Ctrl and click the Smoother icon above the graph to remove the smoother line.

Figure 4.33 Overlay Plot of the Closing Price over Time

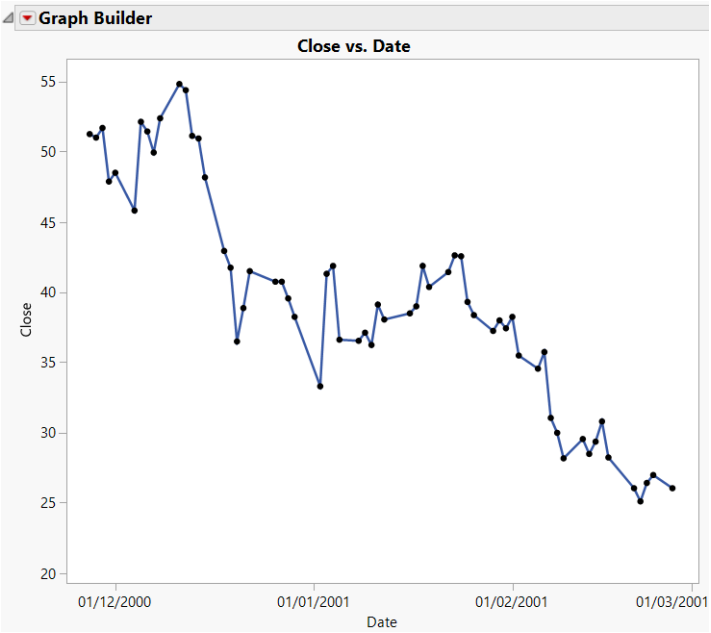


Interpret and Interact with the Overlay Plot

The overlay plot shows that the closing stock price has been decreasing over the last several months. To see the trend more clearly, connect the points.

1. Press Shift and click the Line icon above the graph.

Figure 4.34 Connected Points



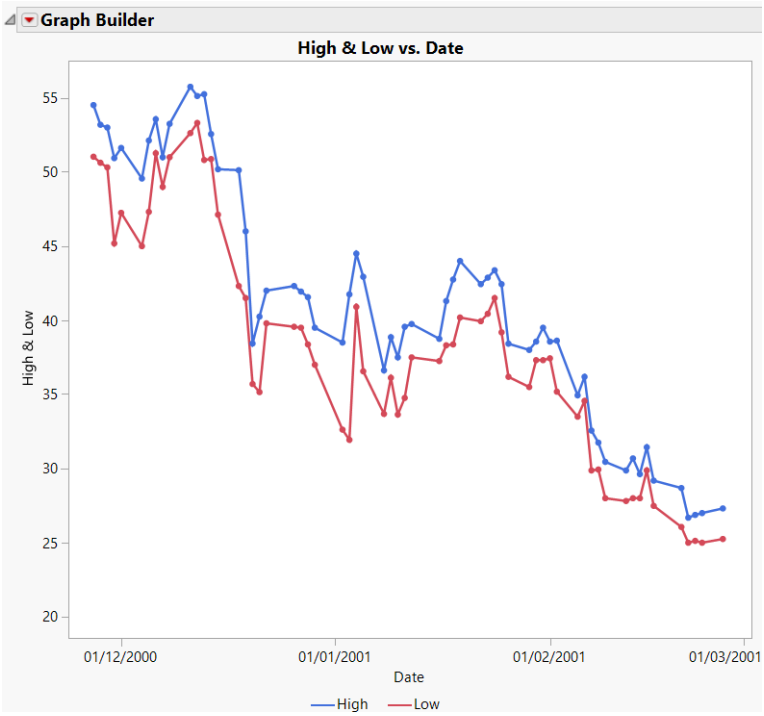
The potential investor can see that although the stock price has gone up and down over the past three months, the overall trend has been downward.

Create the Overlay Plot of the Stock's High and Low Prices

Use an overlay plot to plot more than one Y variable. For example, suppose that you want to see both the high and the low prices on the same plot.

1. Follow the steps in "Create the Overlay Plot of the Stock's Price over Time", this time assigning both High and Low to the **Y** role.
2. Connect the points and add grid lines as shown in "Interpret and Interact with the Overlay Plot".

Figure 4.35 Two Y Variables



The legend at the bottom of the plot shows the colors and markers used for the High and Low variables in the graph. The overlay plot shows that the High price and Low price track each other very closely.

Answer the Questions

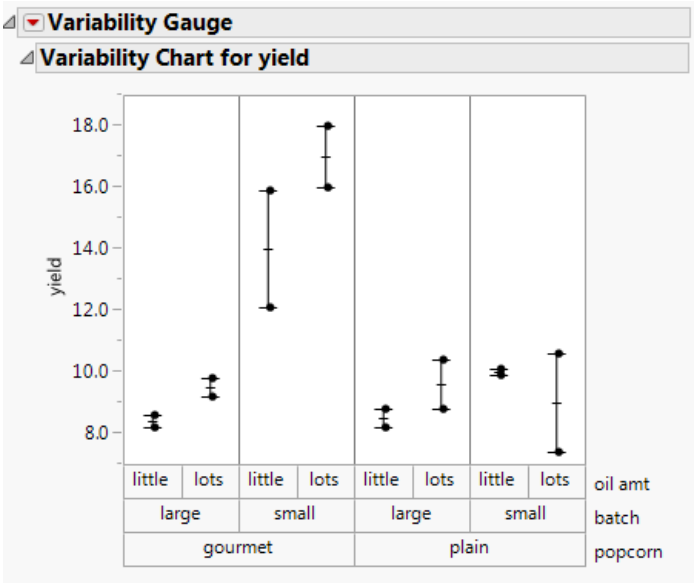
Both of the overlay plots answer the two questions asked at the beginning of this example.

- The first plot shows that the price of this stock has not remained the same, but has been decreasing.
- The second plot shows that the high and low prices of this stock are not very different from each other. The stock price does not vary wildly on any given day.

Compare Multiple Variables Using a Variability Chart

In the graphs described so far, you specified only a single X variable. Use a variability chart to specify multiple X variables and see differences in means and variability across all of your variables at once.

Figure 4.36 Example of a Variability Chart



Scenario

This example uses the Popcorn.jmp data table with data from a popcorn maker. The yield (the volume of popcorn for a given measure of kernels) was measured for each combination of popcorn style, batch size, and amount of oil used.

The popcorn maker wants to explore the following question:

- Which combination of factors results in the highest popcorn yield?

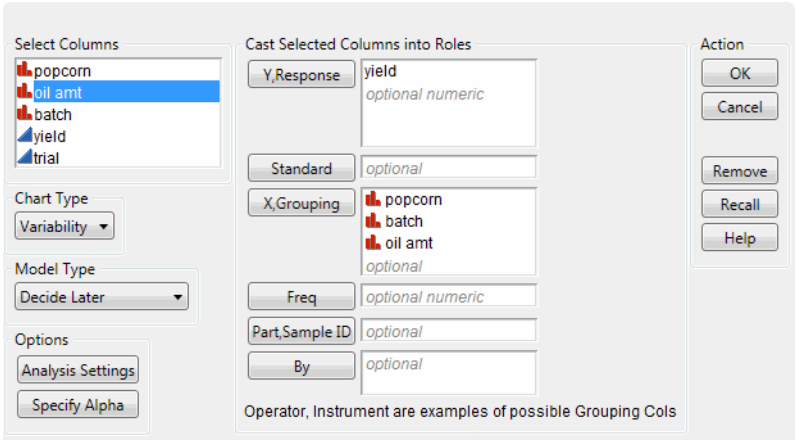
To answer this question, use a variability chart of the yield versus the style, batch size, and oil amount.

Create the Variability Chart

1. Select **Help > Sample Data Library** and open Popcorn.jmp.
2. Select **Analyze > Quality and Process > Variability/Attribute Gauge Chart**.
3. Select yield and click **Y, Response**.
4. Select popcorn and click **X, Grouping**.
5. Select batch and click **X, Grouping**.
6. Select oil amt and click **X, Grouping**.

Note: The order in which you assign the variables to the **X, Grouping** role is important, because the order in this window determines their nesting order in the variability chart.

Figure 4.37 Variability Chart Window

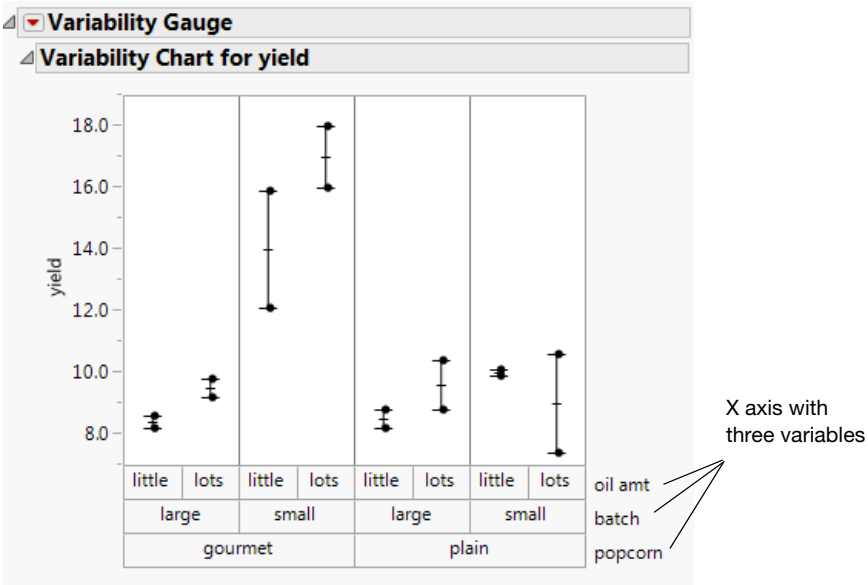


7. Click **OK**.

   The top chart is the variability chart, showing the yield broken down by each combination of the three variables. The bottom chart shows the standard deviation for each combination of the three variables. Since the bottom chart does not show the yield, hide it.

8. Click the Variability Gauge red triangle and deselect **Std Dev Chart**.

Figure 4.38 Results Window



Interpret the Variability Chart

The variability chart for yield indicates that small, gourmet batches produce the highest yield.

To be more specific, the popcorn maker might ask this additional question: Is the yield high because those batches are small, or because those batches are gourmet?

The variability chart shows the following:

- The yield from small, plain batches is low.
- The yield from large, gourmet batches is low.

Given this information, the popcorn maker can conclude that only the combination of small and gourmet at the same time results in batches with high yield. It would have been impossible to reach this conclusion with a chart that only allowed a single variable.

Chapter 5

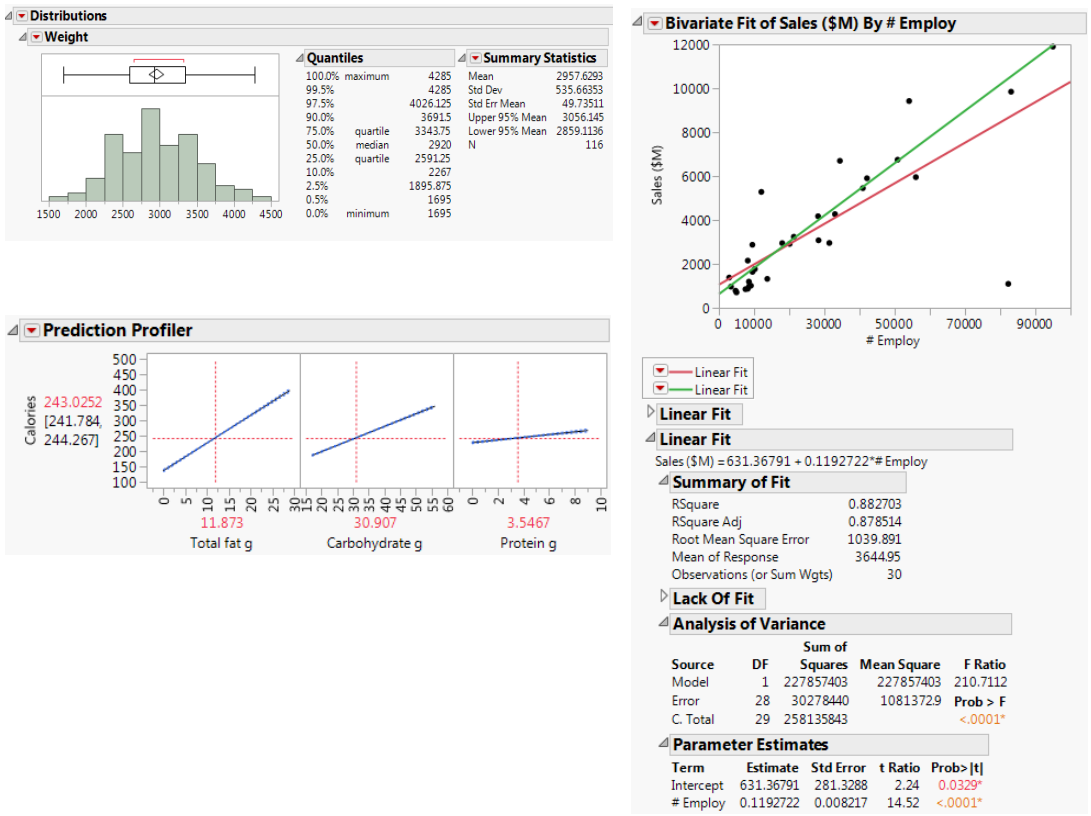
Analyze Your Data

Distributions, Relationships, and Models

Analyzing your data in JMP helps you make informed decisions. Data analysis often involves these actions:

- Examining distributions
- Discovering relationships
- Hypothesis testing
- Building models

Figure 5.1 Analysis Examples



Contents

About This Chapter 135
The Importance of Graphing Your Data 135
Understand Modeling Types 138
Example of Viewing Modeling Type Results 139
Change the Modeling Type 141
Analyze Distributions 143
Distributions of Continuous Variables 143
Distributions of Categorical Variables 146
Analyze Relationships 149
Use Regression with One Predictor 149
Compare Averages for One Variable 154
Compare Proportions 157
Compare Averages for Multiple Variables 160
Use Regression with Multiple Predictors 165

About This Chapter

Before you analyze your data, review the following information about basic concepts:

- "The Importance of Graphing Your Data"
- "Understand Modeling Types"

The rest of this chapter shows you how to use some basic analytical methods in JMP:

- "Analyze Distributions"
- "Analyze Relationships"

For a description of advanced modeling and analysis techniques, refer to the following JMP documentation:

- *Fitting Linear Models*
- *Multivariate Methods*
- *Predictive and Specialized Modeling*
- *Consumer Research*
- *Reliability and Survival Methods*
- *Quality and Process Methods*

The Importance of Graphing Your Data

Graphing, or visualizing, your data is important to any data analysis, and should always occur before the use of statistical tests or model building. To illustrate why data visualization should be an early step in your data analysis process, consider the following example:

1. Select **Help > Sample Data Library** and open Anscombe.jmp (F. J. Anscombe (1973), *American Statistician*, 27, 17-21).

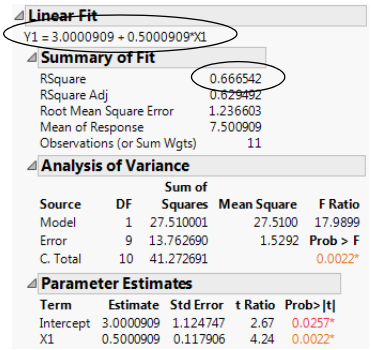
   This data consists of four pairs of *X* and *Y* variables.

2. In the Table panel, click the green triangle next to the **The Quartet** script.

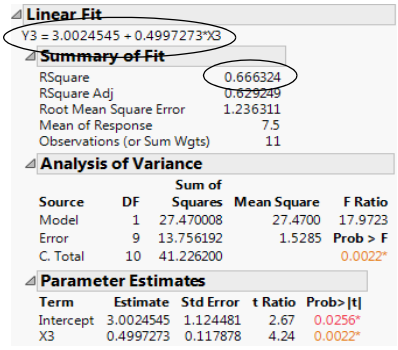
   The script creates a simple linear regression on each pair of variables using **Fit Y by X**. The **Show Points** option is turned off, so that none of the data can be seen on the scatterplots. Figure 5.2 shows the model fit and other summary information for each regression.

Figure 5.2 Four Models

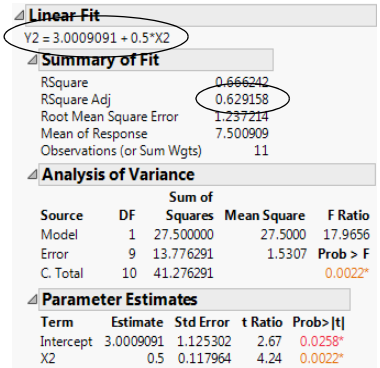
Model for pair 1



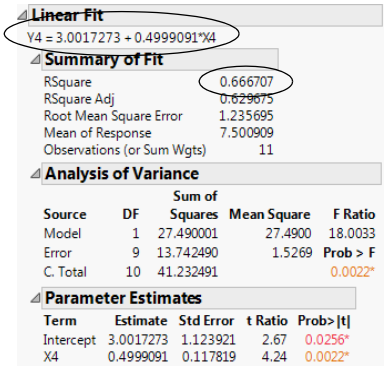
Model for pair 3



Model for pair 2



Model for pair 4



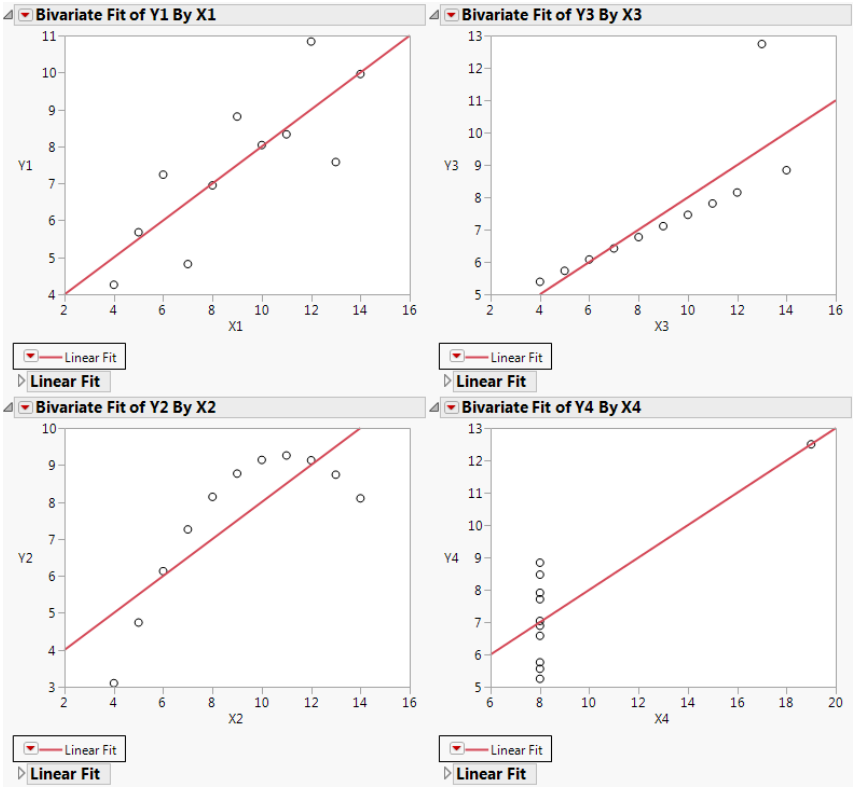
Notice that all four models and the RSquare values are nearly identical. The fitted model in each case is essentially $Y = 3 + 0.5X$, and the RSquare value in each case is essentially 0.66. If your data analysis took into account only the above summary information, you would likely conclude that the relationship between X and Y is the same in each case. However, at this point, you have not visualized your data. Your conclusion might be wrong.

To Visualize the Data, Add the Points to All Four Scatterplots

1. Press Ctrl.
2. Click the red triangle next to any one of the Bivariate Fits and select **Show Points**.

Figure 5.3 Scatterplots with Points Added

Plot for pair 1



Plot for pair 3

Plot for pair 2

Plot for pair 4

The scatterplots show that the relationship between *X* and *Y* is not the same for the four pairs, although the lines describing the relationships are the same:

- Plot 1 represents a linear relationship.
- Plot 2 represents a non-linear relationship.
- Plot 3 represents a linear relationship, except for one outlier.
- Plot 4 has all the data at x = 8, except for one point.

This example illustrates that conclusions that are based on statistics alone can be inadequate. A visual exploration of the data should be an early part of any data analysis.

Understand Modeling Types

In JMP, data can be of different types. JMP refers to this as the modeling type of the data. Table 5.1 describes the three modeling types in JMP.

Table 5.1 Modeling Types

Modeling Type and Description	Examples	Specific Example
Continuous Numeric data only. Used in operations like sums and means.	Height Temperature Time	The time to complete a test might be 2 hours, or 2.13 hours.
Ordinal Numeric or character data. Values belong to ordered categories.	Month (1,2,...,12) Letter grade (A, B,...F) Size (small, medium, large)	The month of the year can be 2 (February) or 3 (March), but not 2.13. February comes before March.
Nominal Numeric or character data. Values belong to categories, but the order is not important.	Gender (M or F) Color Test result (pass or fail)	The gender can be M or F, with no order. Gender categories can also be represented by a number (M=1 and F=2).
Multiple Response Character data only. Distinct entries in a single cell that are separated by commas.	When you brush your teeth College degrees Sports you play	There are several times a day that you could brush your teeth. First thing in the morning, after breakfast, after meals, before bed, or any combination of the above.
Unstructured Text Character data only. Usually all unique values that must be analyzed using Text Explorer.	Product reviews Song lyrics Free response field in survey	Most product reviews would be unique and Text Explorer is used to determine any underlying similarities.

Table 5.1 Modeling Types *(Continued)*

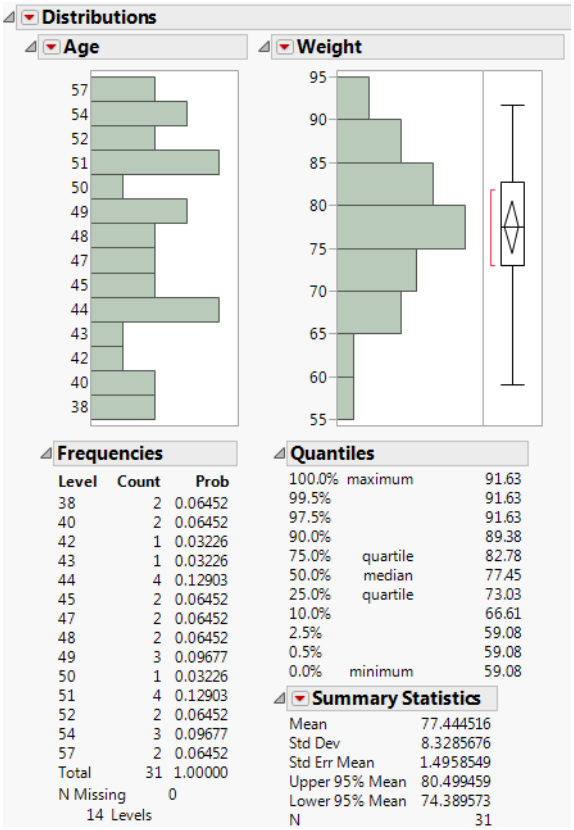
Modeling Type and Description	Examples	Specific Example
Vector Expression data only. Values in a cell are column or row vectors.	Prediction formulas	
None Any data type. Used in scenarios where a column is not well represented by the other modeling types.	Pictures ID Values	A picture column in a data table would not be used to modeling, but could be used as a marker on a graph for example.

Example of Viewing Modeling Type Results

Different modeling types produce different results in JMP. To see an example of the differences, follow these steps:

1. Select **Help > Sample Data Library** and open Linnerud.jmp.
2. Select **Analyze** > **Distribution**.
3. Select Age and Weight and click **Y, Columns**.
4. Click **OK**.

Figure 5.4 Distribution Results for Age and Weight



Although Age and Weight are both numeric variables, they are not treated the same. Table 5.2 compares the differences between the results for weight and age.

Table 5.2 Results for weight and age

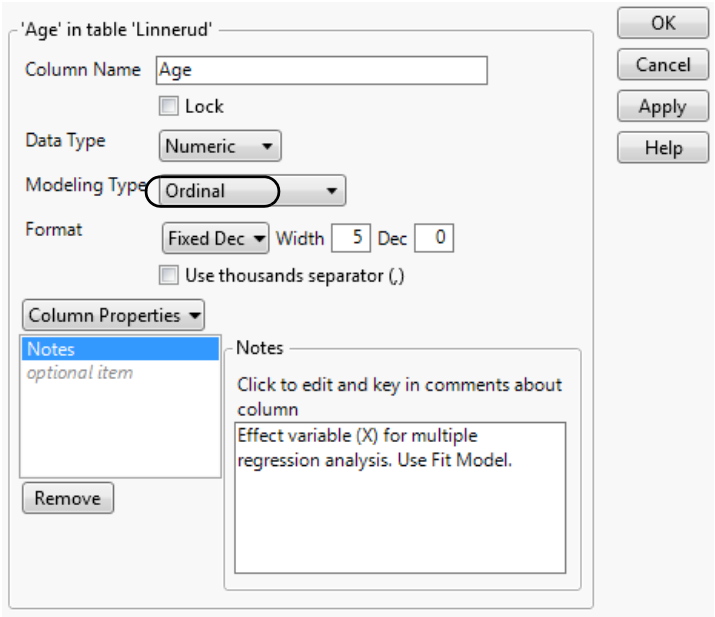
Variable	Modeling Type	Results
Weight	Continuous	Histogram, Quantiles, and Summary Statistics
Age	Ordinal	Bar chart and Frequencies

Change the Modeling Type

To treat a variable differently, change the modeling type. For example, in Figure 5.4, the modeling type for Age is ordinal. Remember that for an ordinal variable, JMP calculates frequency counts. Suppose that you wanted to find the average age instead of frequency counts. Change the modeling type to continuous, which shows the mean age.

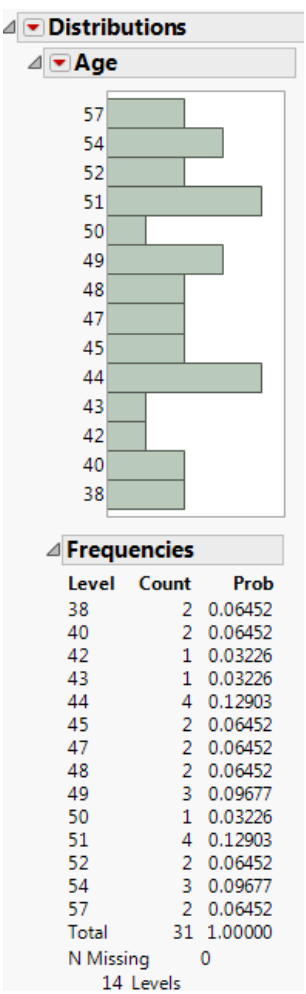
1. Double-click the Age column heading. The Column Info window appears.
2. Change the Modeling Type to **Continuous**.

Figure 5.5 Column Info Window

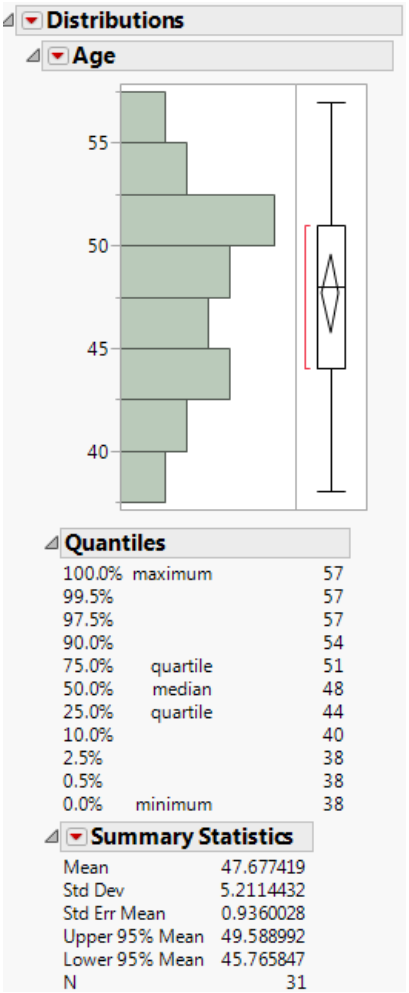


3. Click **OK**.
4. Repeat the steps in the example (see "Example of Viewing Modeling Type Results") to create the distribution. Figure 5.6 shows the distribution results when Age is ordinal and continuous.

Figure 5.6 Different Modeling Types for Age



Ordinal



Continuous

When age is ordinal, you can see the frequency counts for each age. For example, age 48 appears two times. When age is continuous, you can find the mean age, which is nearly 48 (47.677)

Analyze Distributions

To analyze a single variable, you can examine the distribution of the variable, using the Distribution platform. Report content for each variable varies, depending on whether the variable is categorical (nominal or ordinal) or continuous.

Note: For more information about the Distribution platform, see *Basic Analysis*.

Distributions of Continuous Variables

Analyzing a continuous variable might include questions such as the following:

- Does the shape of the data match any known distributions?
- Are there any outliers in the data?
- What is the average of the data?
- Is the average statistically different from a target or historical value?
- How spread out are the data? In other words, what is the standard deviation?
- What are the minimum and maximum values?

You can answer these and other questions with graphs, summary statistics, and simple statistical tests.

Scenario

This example uses the Car Physical Data.jmp data table, which contains information about 116 different car models.

A planning specialist has been asked by a railroad company to determine the possible issues involved in transporting cars by train. Using the data, the planning specialist wants to explore the following questions:

- What is the average car weight?
- How spread out are the cars' weights (standard deviation)?
- What are the minimum and maximum weights of cars?
- Are there any outliers in the data?

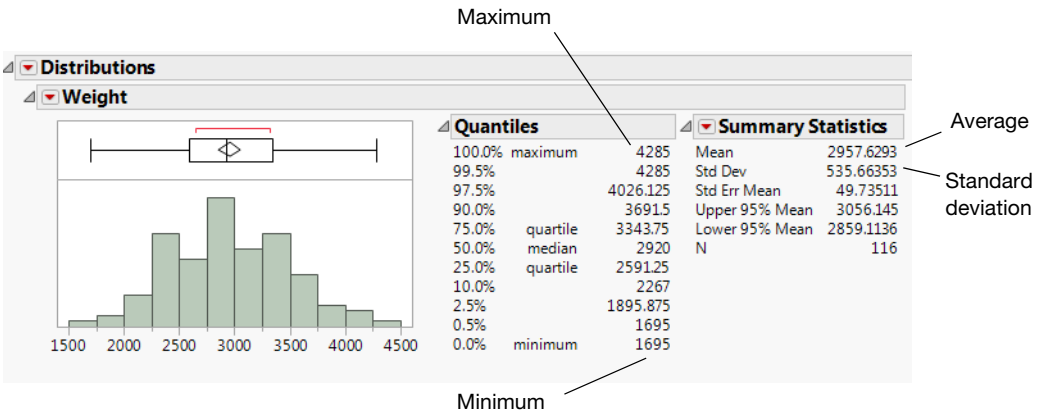
Use a histogram of weight to answer these questions.

Create the Histogram

1. Select **Help > Sample Data Library** and open Car Physical Data.jmp.

2. Select **Analyze** > **Distribution**.
3. Select Weight and click **Y, Columns**.
4. Click **OK**.
5. To rotate the report window, click the Weight red triangle and select **Display Options** > **Horizontal Layout**.

Figure 5.7 Distribution of Weight



The report window contains three sections:

- A histogram and a box plot to visualize the data.
- A Quantiles report that shows the percentiles of the distribution.
- A Summary Statistics report that shows the mean, standard deviation, and other statistics.

Interpret the Distribution Results

Using the results presented in Figure 5.7, the planning specialist can answer the questions.

What is the average car weight? The Histogram shows a weight of around 3,000 lbs.

How spread out are the weights (standard deviation)? The Summary Statistics show a weight of around 2,958 lbs. The Summary Statistics show a standard deviation of around 536 lbs.

What are the minimum and maximum weights? The Histogram shows a minimum of around 1,500 lbs. and a maximum of around 4,500 lbs. The Quantiles show a minimum of around 1,695 lbs. and a maximum of around 4,285 lbs.

Are there any outliers? No.

The default report window in Figure 5.7 provides a minimal set of graphs and statistics. Additional graphs and statistics are available on the red triangle menu.

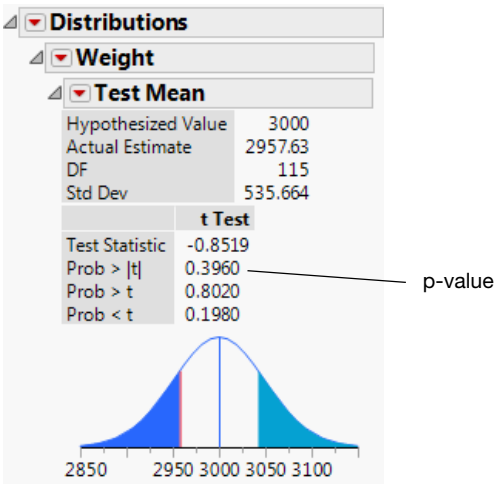
Draw Conclusions

Based on other research, the railroad company has determined that an average weight of 3000 pounds is the most efficient to transport. Now, the planning specialist needs to find out whether the average car weight in the general population of cars that they might transport is 3000 pounds. Use a *t* test to draw inferences about the broader population based on this sample of the population.

Test Conclusions

1. Click the Weight red triangle and select **Test Mean**.
2. In the window that appears, type 3000 in the Specify Hypothesized Mean box.
3. Click **OK**.

Figure 5.8 Test Mean Results



Interpret the t Test

The primary result of a *t* test is the *p*-value. In this example, the *p*-value is 0.396 and the analyst is using a significance level of 0.05. Since 0.396 is greater than 0.05, you cannot conclude that the average weight of car models in the broader population is significantly different from 3000 pounds. Had the *p*-value been lower than the significance level, the planning specialist would have concluded that the average car weight in the broader population *is* significantly different from 3000 pounds.

Distributions of Categorical Variables

Analyzing a categorical (ordinal or nominal) variable might include questions such as the following:

- How many levels does the variable have?
- How many data points does each level have?
- Is the data uniformly distributed?
- What proportions of the total do each level represent?

Scenario

See the scenario in "Distributions of Continuous Variables".

Now that the railroad company has determined that the average weight of the cars is not significantly different from the target weight, there are more questions to address.

The planning specialist wants to answer these questions for the railroad company:

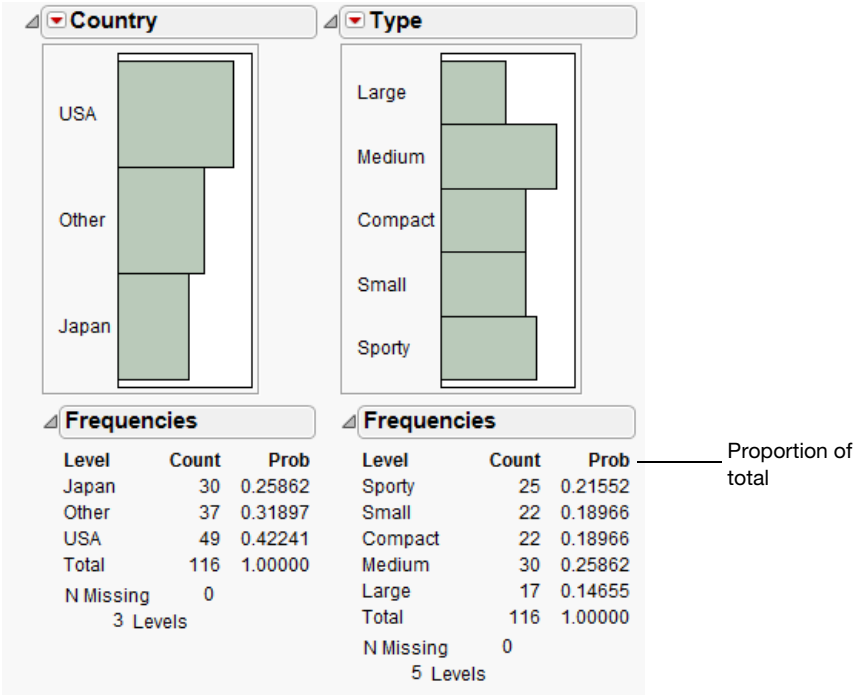
- What are the types of cars?
- What are the countries of origin?

To answer these questions, look at the distribution for Type and Country.

Create the Distribution

1. Select **Help > Sample Data Library** and open Car Physical Data.jmp.
2. Select **Analyze** > **Distribution**.
3. Select Country and Type and click **Y, Columns**.
4. Click **OK**.

Figure 5.9 Distribution for Country and Type



Interpret the Distribution Results

The report window includes a bar chart and a Frequencies report for Country and Type. The bar chart is a graphical representation of the frequency information provided in the Frequencies report. The Frequencies report contains the following:

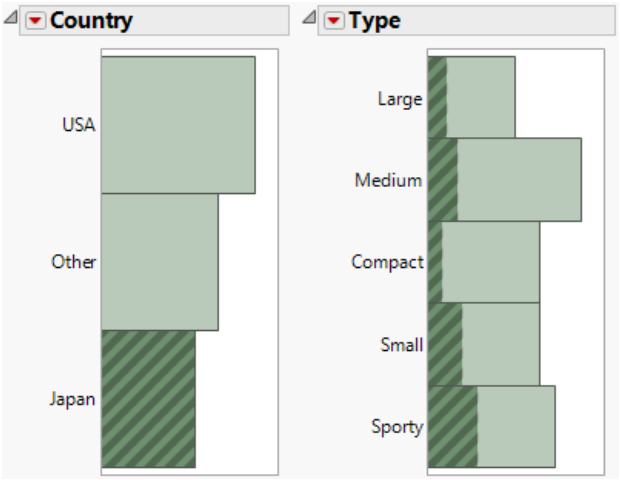
- Categories of data. For example, Japan is a category of Country, and Sporty is a category of Type.
- Total counts for each category.
- Proportion of the total each category represents.

For example, there are 22 compact cars, or about 19% of the 116 observations.

Interact with the Distribution Results

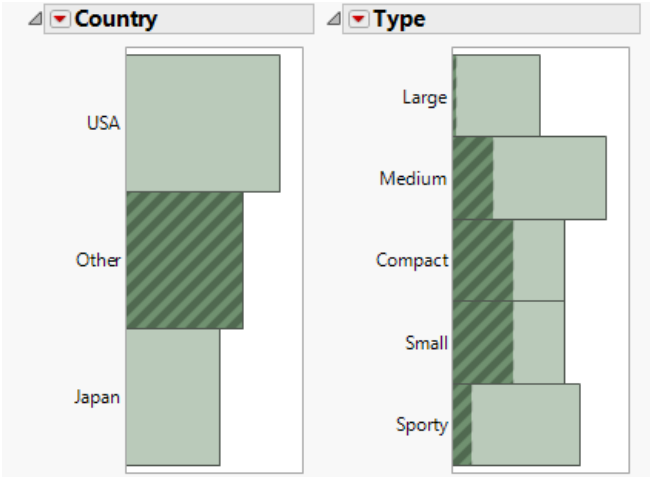
Selecting a bar in one chart also selects the corresponding data in the other chart. For example, select the Japan bar in the Country bar chart to see that a large number of Japanese cars are sporty.

Figure 5.10 Japanese Cars



Select the Other category to see that a majority of these cars are small or compact, and almost none are large.

Figure 5.11 Other Cars



Analyze Relationships

Scatterplots and other such graphs can help you visualize relationships between variables. Once you have visualized relationships, the next step is to analyze those relationships so that you can describe them numerically. That numerical description of the relationship between variables is called a *model*. Even more importantly, a model also predicts the average value of one variable (*Y*) from the value of another variable (*X*). The *X* variable is also called a predictor. Generally, this model is called a *regression* model.

With JMP, the **Fit Y by X** platform and the **Fit Model** platform creates regression models.

Note: Only the basic platforms and options are covered here. For explanations of all platform options, see *Basic Analysis*, *Essential Graphing*, and the documentation listed in "About This Chapter".

Table 5.3 shows the four primary types of relationships.

Table 5.3 Relationship Types

X	Y	Section
Continuous	Continuous	• "Use Regression with One Predictor" • "Use Regression with Multiple Predictors"
Categorical	Continuous	• "Compare Averages for One Variable" • "Compare Averages for Multiple Variables"
Categorical	Categorical	"Compare Proportions"
Continuous	Categorical	Logistic regression is an advanced topic. See *Basic Analysis*.

Use Regression with One Predictor

Scenario

This example uses the Companies.jmp data table, which contains financial data for 32 companies from the pharmaceutical and computer industries.

Intuitively, it makes sense that companies with more employees can generate more sales revenue than companies with fewer employees. A data analyst wants to predict the overall sales revenue for each company based on the number of employees.

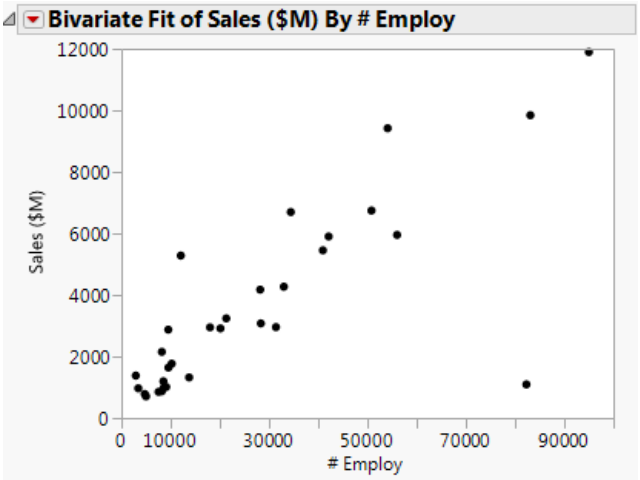
To accomplish this task, do the following:

- "Discover the Relationship"
- "Fit the Regression Model"
- "Predict Average Sales"

Discover the Relationship

First, create a scatterplot to see the relationship between the number of employees and the amount of sales revenue. This scatterplot was created in "Create the Scatterplot". After hiding and excluding one outlier (a company with significantly more employees and higher sales), the plot in Figure 5.12 shows the result.

Figure 5.12 Scatterplot of Sales ($M) versus # Employ

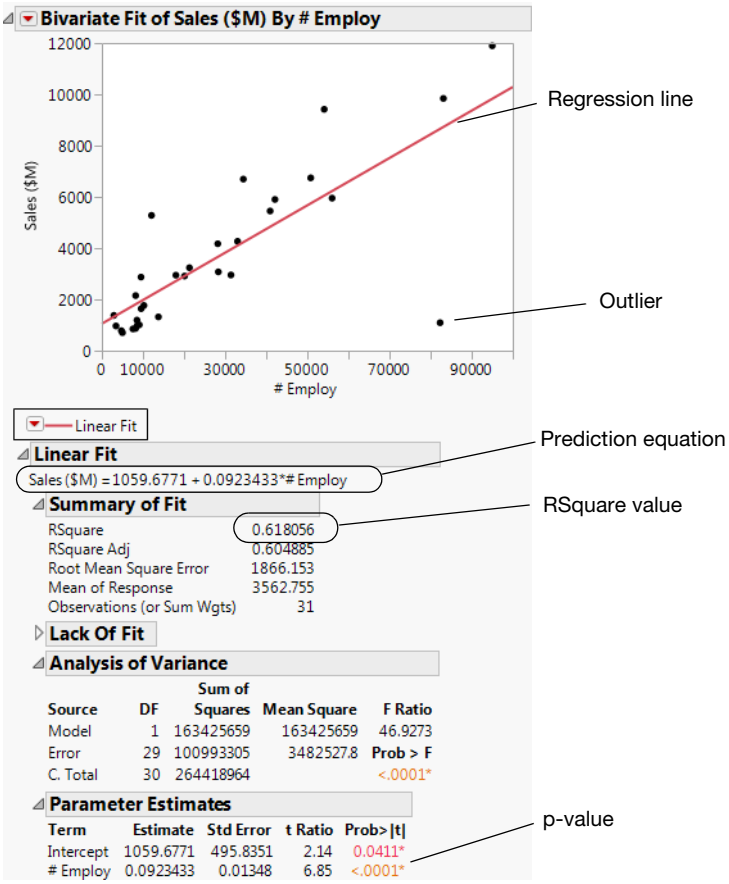


This scatterplot provides a clearer picture of the relationship between sales and the number of employees. As expected, the more employees a company has, the higher sales that it can generate. This visually confirms the data analyst's guess, but it does not predict sales for a given number of employees.

Fit the Regression Model

To predict the sales revenue from the number of employees, fit a regression model. Click the Bivariate Fit red triangle and select **Fit Line**. A regression line is added to the scatterplot and reports are added to the report window.

Figure 5.13 Regression Line



Within the reports, look at the following results:

- the *p*-value of <.0001
- the RSquare value of 0.618

From these results, the data analyst can conclude the following:

- The *p*-value for the #Employ model term is small. This supports that at the 0.05 significance level the coefficient for #Employ is not zero. Therefore, including the number of employees in the prediction model significantly improves the ability to predict average sales over a model without the number of employees.
- The RSquare value of 0.618 indicates that this model explains about 62% of the variability in sales. The RSquare value is the coefficient of determination and indicates the proportion of the variance in the dependent (response) variable that is explained by your model. RSquare can range from 0 to 1. A model with an RSquare of 0 has no explanatory power. A model with an RSquare of 1 predicts the response perfectly.

Predict Average Sales

Use the regression model to predict the average sales a company might expect if they have a certain number of employees. The prediction equation for the model is included in the report:

Average sales = 1059.68 + 0.092*employees

For example, in a company with 70,000 employees sales are predicted to be about $7,500:

$7,499.68 = 1059.68 + 0.092*70,000

In the lower right area of the current scatterplot, there is an outlier that does not follow the general pattern of the other companies. The data analyst wants to know whether the prediction model changes when this outlier is excluded.

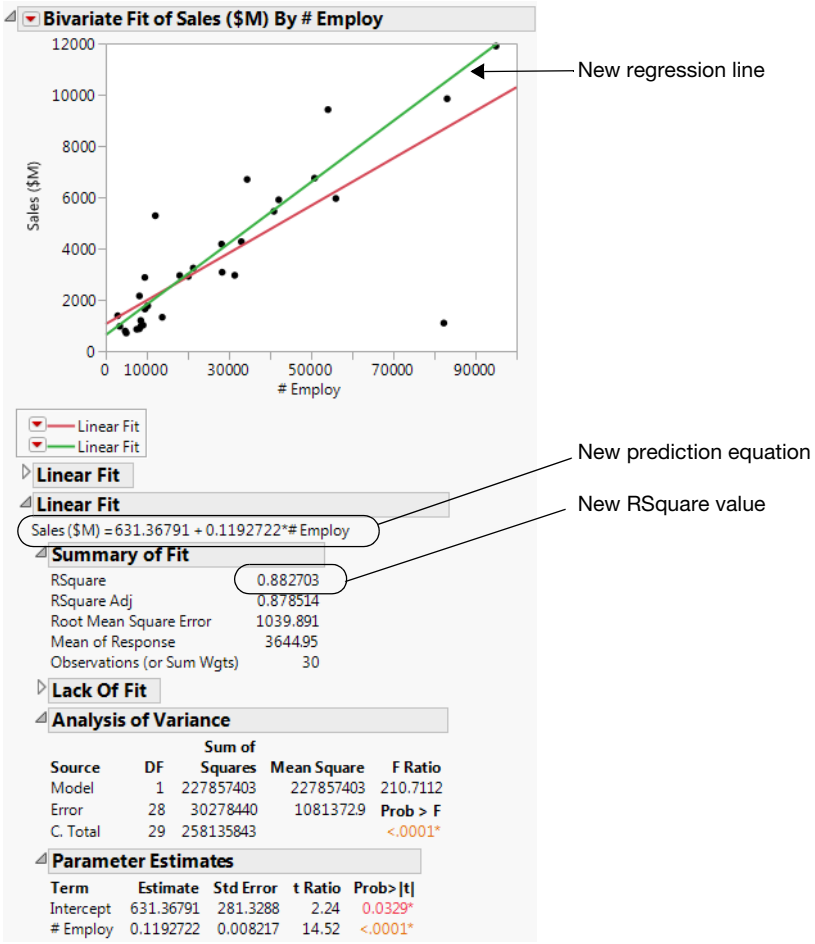
Exclude the Outlier

1. Click the outlier.
2. Select **Rows** > **Exclude/Unexclude**.
3. To fit this model, click red triangle next to Bivariate Fit of Sales (SM) By # Employ and select **Fit Line**.

The following are added to the report window (Figure 5.14):

- a new regression line
- a new Linear Fit report, which includes:
  - a new prediction equation
  - a new RSquare value

Figure 5.14 Comparing the Models



Interpret the Results

Using the results in Figure 5.14, the data analyst can make the following conclusions:

- The outlier was pulling down the regression line for the larger companies, and pulling the line up for the smaller companies.
- The new model for the data without the outlier is a stronger model than the first model. The new RSquare value of 0.88 is higher and closer to 1 than the initial analysis.

Draw Conclusions

Using the new prediction equation, the predicted average sales for a company with 70,000 employees can be calculated as follows:

$8961.37 = 631.37 + 0.119*70,000

The prediction from the first model was about $7500. The second model predicts a sales total of about $8960 or an increase of $1460 as compared to the first model.

The second model, after removing the outlier, describes and predicts sales totals based on the number of employees better than the first model. The data analyst now has a good model to use.

Compare Averages for One Variable

If you have a continuous Y variable, and a categorical X variable, you can compare averages across the levels of the X variable.

Scenario

This example uses the Companies.jmp data table, which contains financial data for 32 companies from the pharmaceutical and computer industries.

A financial analyst wants to explore the following question:

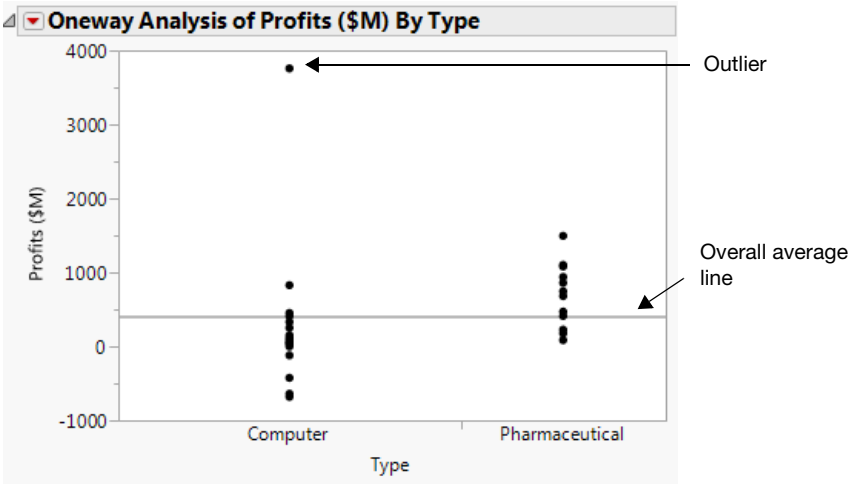
- How do the profits of computer companies compare to the profits of pharmaceutical companies?

To answer this question, fit Profits ($M) by Type.

Discover the Relationship

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. If you still have the Companies.jmp sample data table open, you might have rows that are excluded or hidden. To return the rows to the default state (all rows included and none hidden), select **Rows** > **Clear Row States**.
3. Select **Analyze** > **Fit Y by X**.
4. Select Profits ($M) and click **Y, Response**.
5. Select Type and click **X, Factor**.
6. Click **OK**.

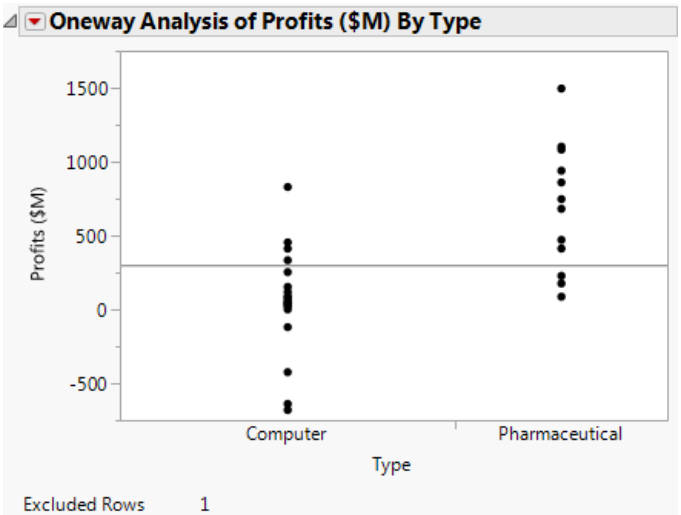
Figure 5.15 Profits by Company Type



There is an outlier in the Computer Type. The outlier is stretching the scale of the plot and making it difficult to compare the profits. Exclude and hide the outlier:

1. Click the outlier.
2. Select **Rows** > **Exclude/Unexclude**. The data point is no longer included in calculations.
3. Select **Rows** > **Hide/Unhide**. The data point is hidden from all graphs.
4. To re-create the plot without the outlier, click the Oneway Analysis of Profits ($M) By Type and select **Redo** > **Redo Analysis**. You can close the original Scatterplot window.

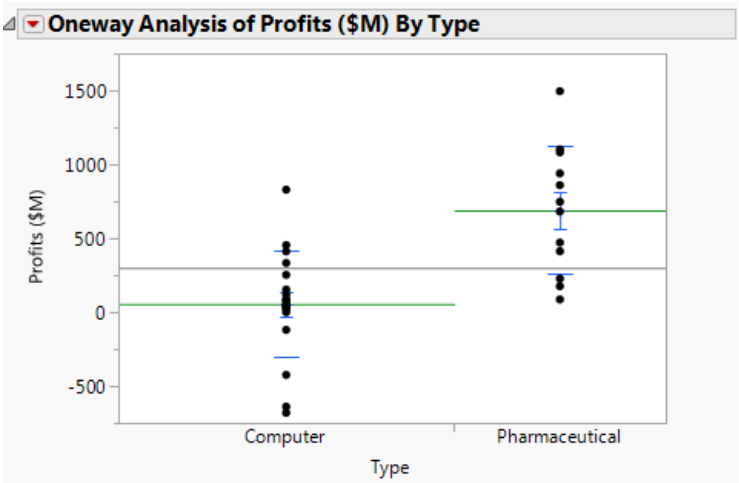
Figure 5.16 Updated Plot



Removing the outlier gives the financial analyst a clearer picture of the data.

5. To continue analyzing the relationship, select these options from the red triangle next to Oneway Analysis of Profits ($M) By Type:
   - **Display Options** > **Mean Lines**. This adds mean lines to the scatterplot.
   - **Means and Std Dev**. This displays a report that provides averages and standard deviations.

Figure 5.17 Mean Lines and Report



Interpret the Results

The financial analyst wanted to know how the profits of computer companies compared to the profits of pharmaceutical companies. The updated scatterplot shows that pharmaceutical companies have higher average profits than computer companies. In the report, if you subtract one mean value from the other, the difference in profit is about $635 million. The plot also shows that some of the computer companies have negative profits and all of the pharmaceutical companies have positive profits.

Perform the t Test

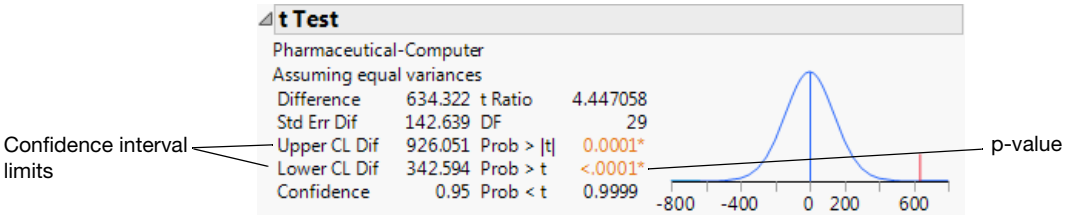
The financial analyst has looked at only a sample of companies (the companies in the data table). The financial analyst now wants to examine these questions:

- Does a difference exist in the broader population, or is the difference of $635 million due to chance?
- If there is a difference, what is it?

To answer these questions, perform a two-sample *t* test. A *t* test lets you use data from a sample to make inferences about the larger population.

To perform the *t* test, click the Oneway Analysis red triangle and select **Means/Anova/Pooled t**.

Figure 5.18 t Test Results



The *p*-value of 0.0001 is less than the significance level of 0.05, which indicates statistical significance. Therefore, the financial analyst can conclude that the observed difference in average profits for the sample data is statistically significant. This means that in the larger population, the average profits for pharmaceutical companies are different from the average profits for computer companies.

Draw Conclusions

Use the confidence interval limits to determine how much difference exists in the profits of both types of companies. Look at the **Upper CL Dif** and **Lower CL Dif** values in Figure 5.18. The financial analyst concludes that the average profit of pharmaceutical companies is between $343 million and $926 million higher than the average profit of computer companies.

Compare Proportions

If you have categorical *X* and *Y* variables, you can compare the proportions of the levels within the Y variable to the levels within the X variable.

Scenario

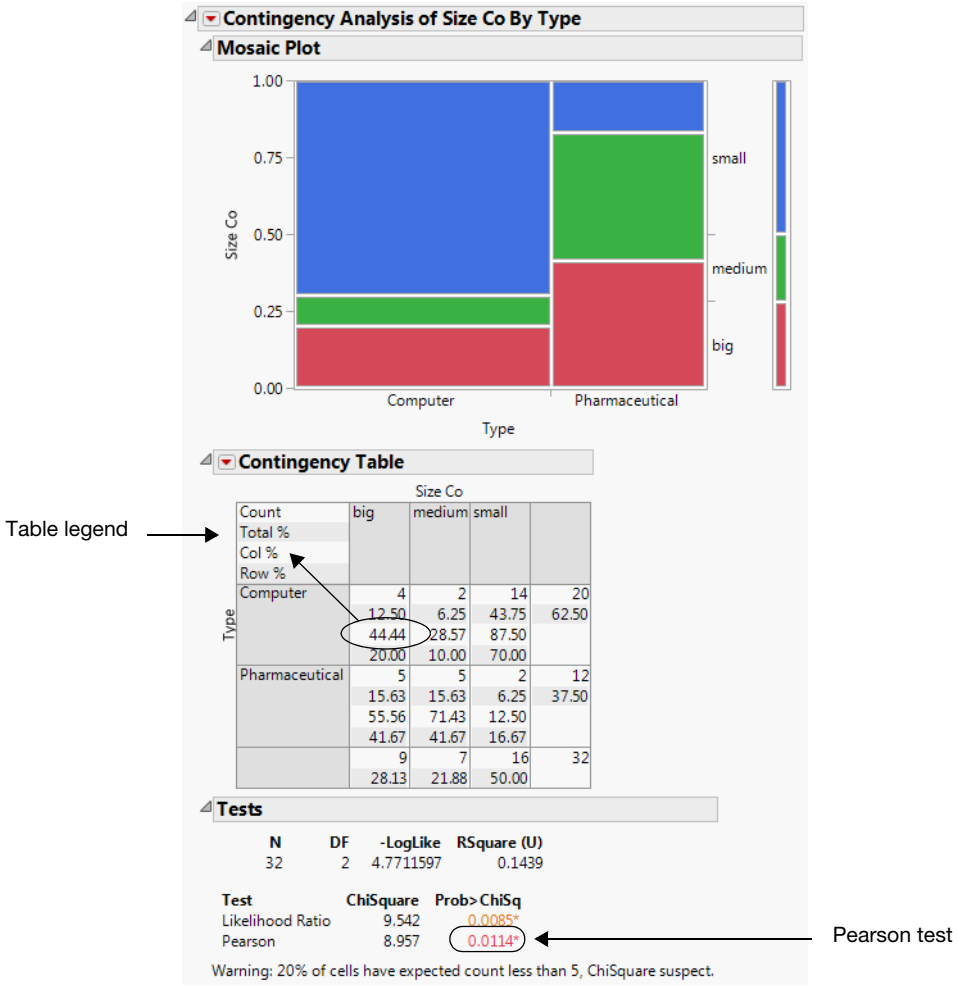
This example continues to use the Companies.jmp data table. In "Compare Averages for One Variable", a financial analyst determined that pharmaceutical companies have higher profits on average than do computer companies.

The financial analyst wants to know whether the size of a company affects profits more for one type of company than the other? However, before examining this question, the financial analyst needs to know whether the populations of computer and pharmaceutical companies consist of the same proportions of small, medium, and big companies.

Discover the Relationship

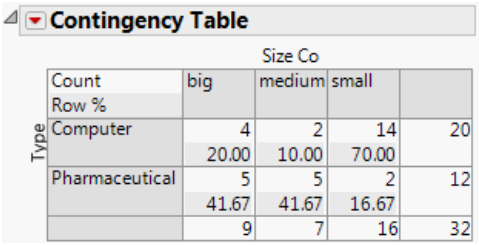
1. Select **Help > Sample Data Library** and open Companies.jmp.
2. If you still have the Companies.jmp data file open from the previous example, you might have rows that are excluded or hidden. To return the rows to the default state (all rows included and none hidden), select **Rows** > **Clear Row States**.
3. Select **Analyze** > **Fit Y by X**.
4. Select Size Co and click **Y, Response**.
5. Select Type and click **X, Factor**.
6. Click **OK**.

Figure 5.19 Company Size by Company Type



The Contingency Table contains information that is not applicable for this example. Click the Contingency Table red triangle and deselect **Total %** and **Col %** to remove that information. Figure 5.20 shows the updated table.

Figure 5.20 Updated Contingency Table



Interpret the Results

The statistics in the Contingency Table are graphically represented in the Mosaic Plot. Together, the Mosaic Plot and the Contingency Table compare the percentages of small, medium, and big companies between the two industries. For example, the Mosaic Plot shows that the computer industry has a higher percentage of small companies compared to the pharmaceutical industry. The Contingency Table shows the exact statistics: 70% of computer companies are small, and about 17% of pharmaceutical companies are small.

Interpret the Test

The financial analyst has looked at only a sample of companies (the companies in the data table). The financial analyst needs to know whether the percentages differ in the broader populations of all computer and pharmaceutical companies.

To answer this question, use the *p*-value from the Pearson test in the **Tests** report (Figure 5.19). Since the *p*-value of 0.011 is less than the significance level of 0.05, the financial analyst concludes the following:

- The differences in the sample data are statistically significant.
- The percentages differ in the broader population.

Now the financial analyst knows that the proportions of small, medium, and big companies are different, and can answer the question: Does the size of company affect profits more for one type of company than the other?

Compare Averages for Multiple Variables

The section "Compare Averages for One Variable", compared averages across the levels of a categorical variable. To compare averages across the levels of two or more variables at once, use the *Analysis of Variance* technique (or ANOVA).

Scenario

The financial analyst can answer the question that we started to work through in the Comparing Proportions section, which is: Does the size of the company have a larger effect on the company's profits, based on type (pharmaceutical or computer)?

To answer this question, compare the company profits by these two variables:

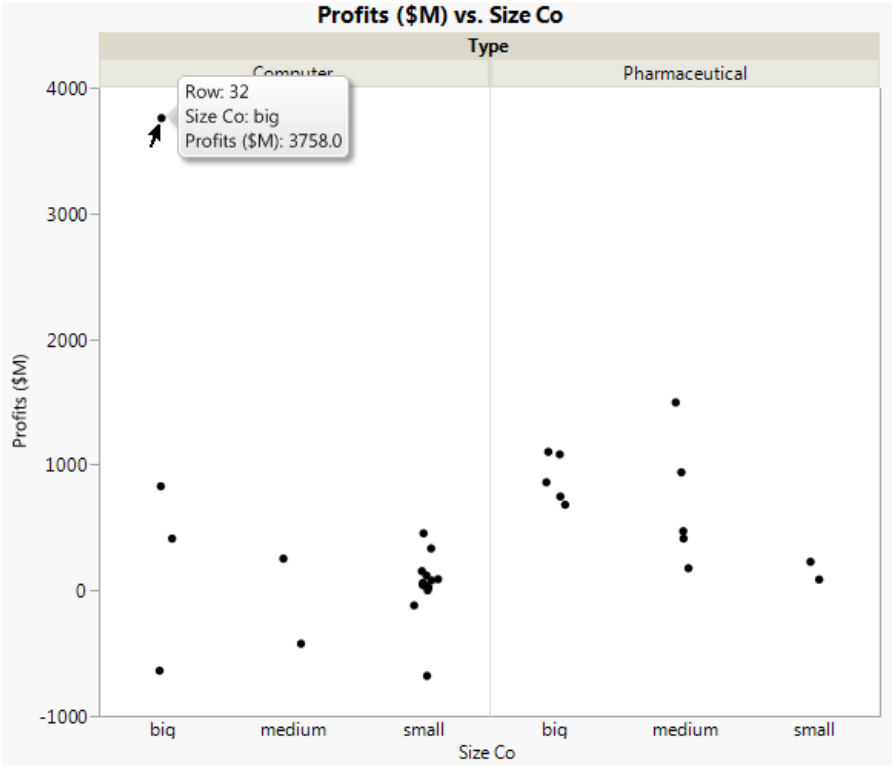
- Type (pharmaceutical or computer)
- Size (small, medium, big)

Discover the Relationship

To visualize the differences in profit for all of the combinations of type and size, use a graph:

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Graph** > **Graph Builder**. The Graph Builder window appears.
3. Click Profits ($M) and drag and drop it into the **Y** zone.
4. Click Size Co and drag and drop it into the **X** zone.
5. Click Type and drag and drop it into the **Group X** zone.

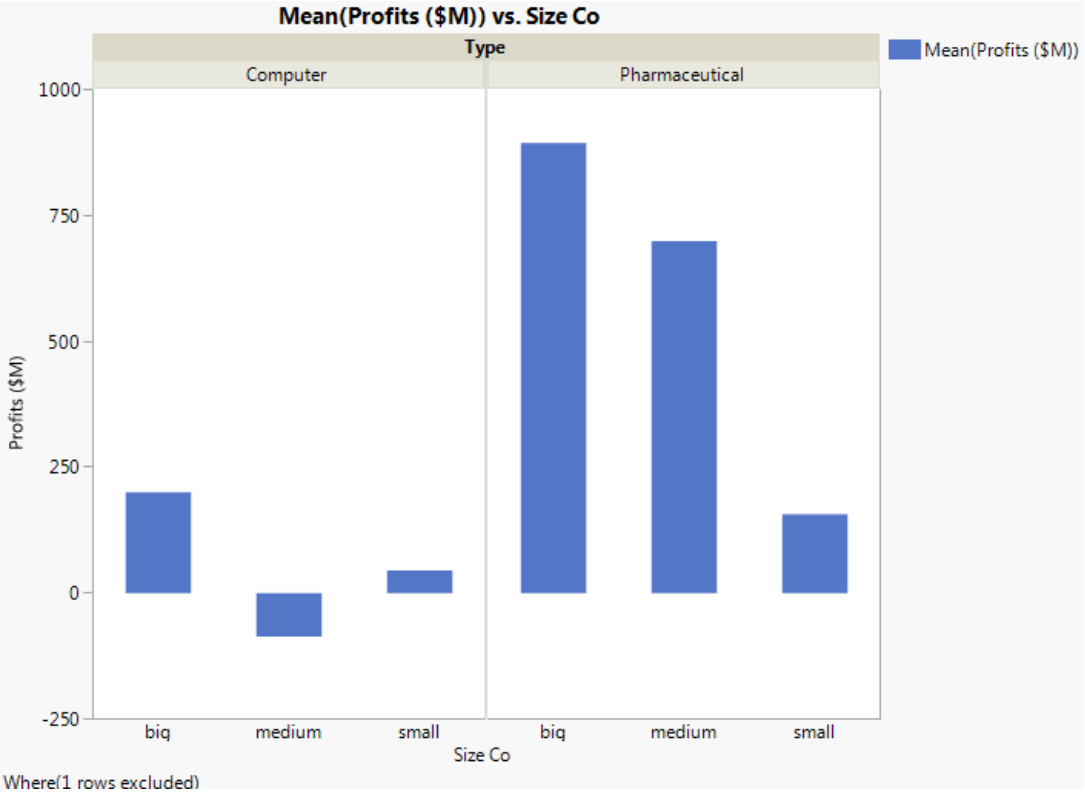
Figure 5.21 Graph of Company Profits



The graph shows that one big computer company has very large profits. That outlier is stretching the scale of the graph, making it difficult to compare the other data points.

6. Select the outlier, then right-click and select **Rows > Row Exclude**. The point is removed, and the scale of the graph automatically updates.
7. Click the Bar icon. Comparing mean profits is easier with bar charts than with points.

Figure 5.22 Graph with Outlier Removed



The updated graph shows that pharmaceutical companies have higher average profits. The graph also shows that profits differ between company sizes for only the pharmaceutical companies. When the effect of one variable (company size) changes for different levels of another variable (company type), this is called an *interaction*.

Quantify the Relationship

Because this data is only a sample, the financial analyst needs to determine the following:

- if the differences are limited to this sample and due to chance

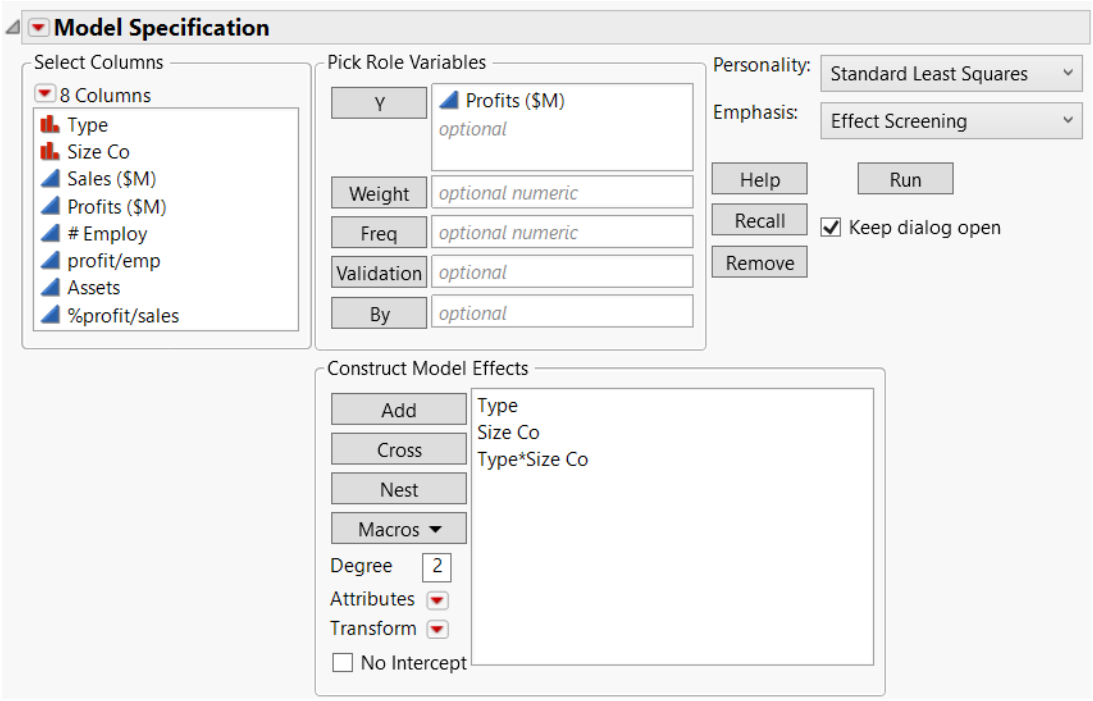
  or

- if the same patterns exist in the broader population

1. Return to the Companies.jmp sample data table that has the data point excluded. See "Discover the Relationship".
2. Select **Analyze** > **Fit Model**.
3. Select Profits ($M) and click **Y**.
4. Select both Type and Size Co.

5. Click the **Macros** button and select **Full Factorial**.
6. From the Emphasis menu, select **Effect Screening**.
7. Select the **Keep dialog open** option.

Figure 5.23 Completed Fit Model Window



8. Click **Run**. The report window shows the model results.

To decide whether the differences in profits are real, or due to chance, examine the **Effect Tests** report.

Note: For more information about all of the Fit Model results, see *Fitting Linear Models*.

View Effect Tests

The Effect Tests report (Figure 5.24) shows the results of the statistical tests. There is a test for each of the effects included in the model on the Fit Model window: Type, Size Co, and Type*Size Co.

Figure 5.24 Effect Tests Report

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F
Type	1	1	1401847.4	10.1368	0.0039*
Size Co	2	2	724616.2	2.6198	0.0927
Type*Size Co	2	2	448061.5	1.6200	0.2180

First, look at the test for the interaction in the model: the Type*Size Co effect. Figure 5.22 showed that the pharmaceutical companies appeared to have different profits between company sizes. However, the effect test indicates that there is no interaction between type and size as it relates to profit. The *p*-value of 0.218 is large (greater than the significance level of 0.05). Therefore, remove that effect from the model, and re-run the model.

1. Return to the Fit Model window.
2. In the Construct Model Effects box, select the **Type*Size Co** effect and click **Remove**.
3. Click **Run**.

Figure 5.25 Updated Effect Tests Report

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F
Type	1	1	1,356,298	9.3768	0.0049*
Size Co	2	2	434,161.3	1.5008	0.2410

The *p*-value for the Size Co effect is large, indicating that there are no differences based on size in the broader population. The *p*-value for the Type effect is small, indicating that the differences that you saw in the data between computer and pharmaceutical companies is not due to chance.

Draw Conclusions

The financial analyst wanted to know whether the size of the company has a larger effect on the company's profits, based on type (pharmaceutical or computer).The financial analyst can now answer this question:

- There is a real difference in profits between computer and pharmaceutical companies in the broader population.
- There is no correlation between the company's size and type and its profits.

Use Regression with Multiple Predictors

The section "Use Regression with One Predictor" showed you how to build simple regression models consisting of one predictor variable and one response variable. *Multiple regression* predicts the average response variable using two or more predictor variables.

Scenario

This example uses the Candy Bars.jmp data table, which contains nutrition information for candy bars.

A dietitian wants to predict calories using the following information:

- Total fat
- Carbohydrates
- Protein

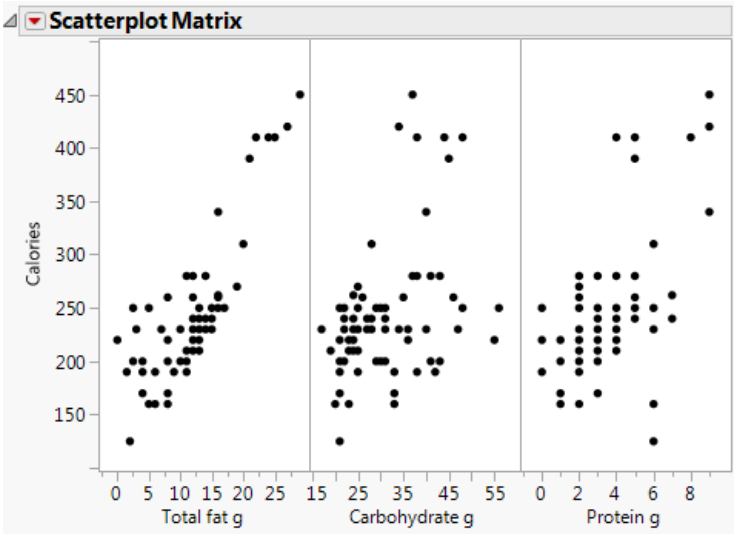
Use *multiple regression* to predict the average response variable using these three predictor variables.

Discover the Relationship

To visualize the relationship between calories and total fat, carbohydrates, and protein, create a scatterplot matrix:

1. Select **Help > Sample Data Library** and open Candy Bars.jmp.
2. Select **Graph** > **Scatterplot Matrix**.
3. Select Calories and click **Y, Columns**.
4. Select Total fat g, Carbohydrate g, and Protein g, and click **X**.
5. Click **OK**.

Figure 5.26 Scatterplot Matrix Results



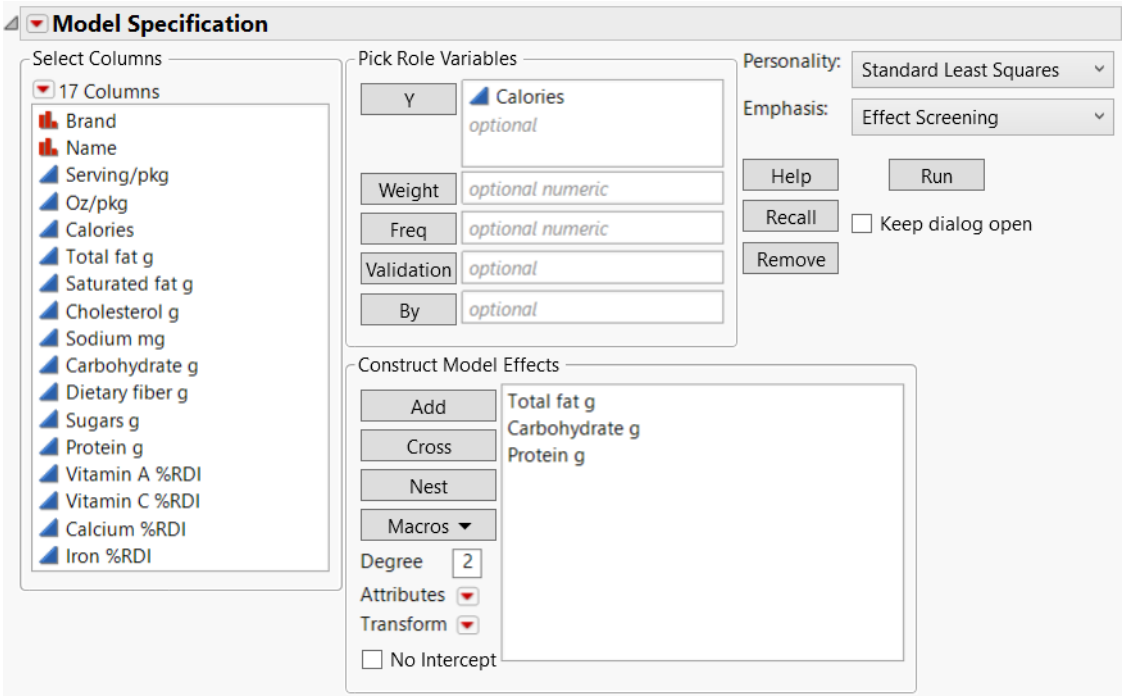
The scatterplot matrix shows that there is a positive correlation between calories and all three variables. The correlation between calories and total fat is the strongest. Now that the dietitian knows that there is a relationship, the dietitian can build a multiple regression model to predict average calories.

Build the Multiple Regression Model

Continue to use the Candy Bars.jmp sample data table.

1. Select **Analyze** > **Fit Model**.
2. Select Calories and click **Y**.
3. Select Total Fat g, Carbohydrate g, and Protein g and click **Add**.
4. Next to Emphasis, select **Effect Screening**.

Figure 5.27 Fit Model Window



5. Click **Run**.

The report window shows the model results. To interpret the model results, focus on these areas:

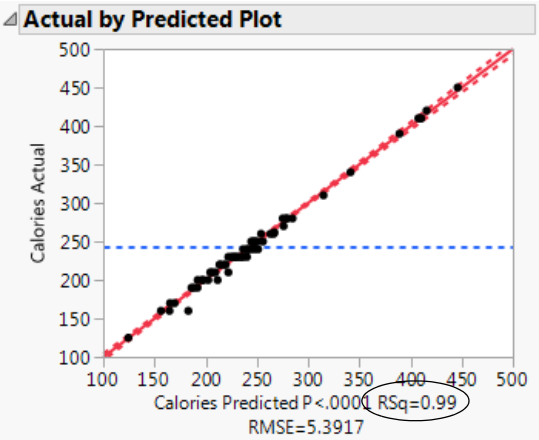
- "View the Actual by Predicted Plot"
- "Interpret the Parameter Estimates"
- "Use the Prediction Profiler"

Note: For more information about all of the model results, see *Fitting Linear Models*.

View the Actual by Predicted Plot

The Actual by Predicted Plot shows the actual calories versus the predicted calories. As the predicted values come closer to the actual values, the points on the scatterplot fall closer around the red line (Figure 5.28). Because the points are all very close to the line, you can see that the model predicts calories based on the chosen factors well.

Figure 5.28 Actual by Predicted Plot



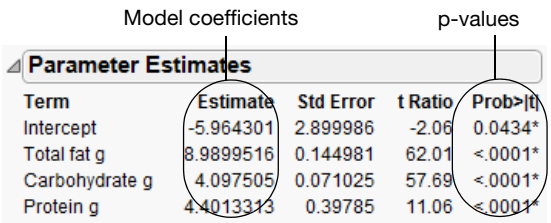
Another measure of model accuracy is the RSq value (which appears below the plot in Figure 5.28). The RSq value measures the percentage of variability in calories, as explained by the model. A value closer to 1 means a model is predicting well. In this example, the RSq value is 0.99.

Interpret the Parameter Estimates

The Parameter Estimates report shows the following information:

- The model coefficients
- *p*-values for each parameter

Figure 5.29 Parameter Estimates Report



In this example, the *p*-values are all very small (<.0001). This indicates that all three effects (fat, carbohydrate, and protein) contribute significantly when predicting calories.

You can use the model coefficients to predict the value of calories for particular values of fat, carbohydrate, and protein. For example, suppose that you want to predict the average calories for any candy bar that has these characteristics:

- Fat = 11 g
- Carbohydrate = 43 g
- Protein = 2 g

Using these values, you can calculate the predicted average calories as follows:

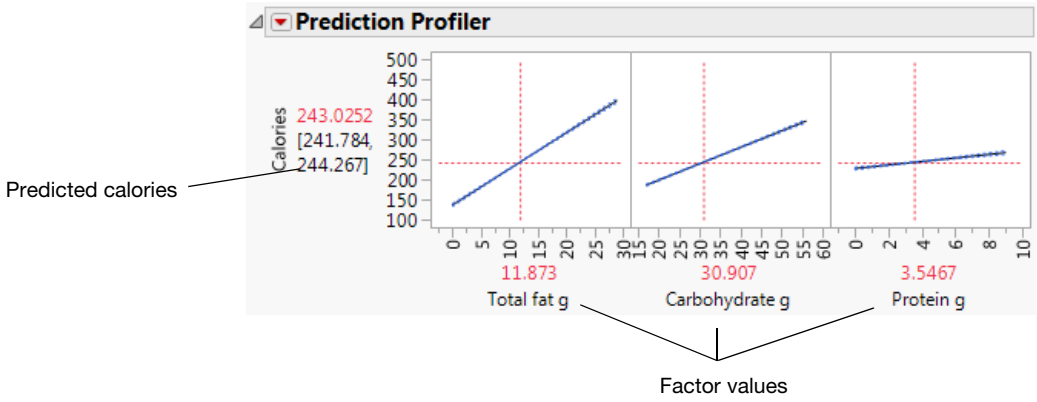
277.92 = -5.9643 + 8.99*11 + 4.0975*43 + 4.4013*2

The characteristics in this example are the same as the Milky Way candy bar (on row 59 of the data table). The actual calories for the Milky Way are 280, showing that the model predicts well.

Use the Prediction Profiler

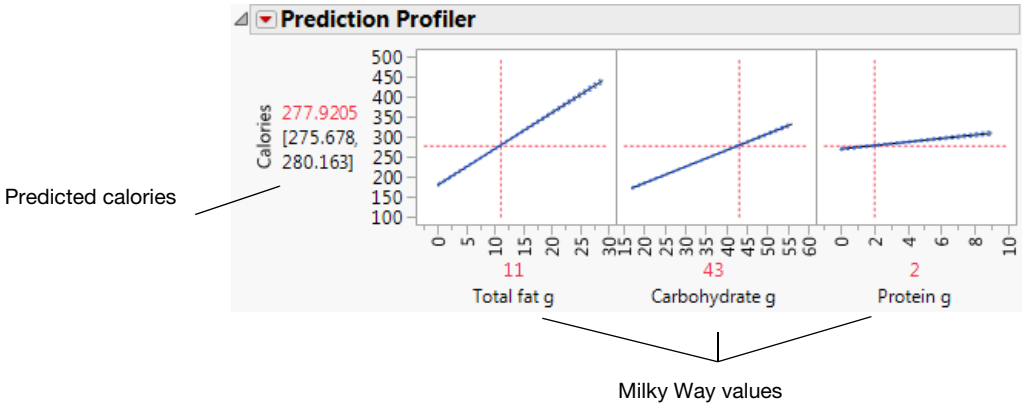
Use the Prediction Profiler to see how changes in the factors affect the predicted values. The profile lines show the magnitude of change in calories as the factor changes. The line for Total fat g is the steepest, meaning that changes in total fat have the largest effect on calories.

Figure 5.30 Prediction Profiler



Click and drag the vertical line for each factor to see how the predicted value changes. You can also click the current factor values and change them. For example, click the factor values and type the values for the Milky Way candy bar (row 59).

Figure 5.31 Factor Values for the Milky Way



Note: For more information about the Prediction Profiler, see *Profilers*.

Draw Conclusions

The dietitian now has a good model to predict calories of a candy bar based on its total fat, carbohydrates, and protein.

Chapter 6

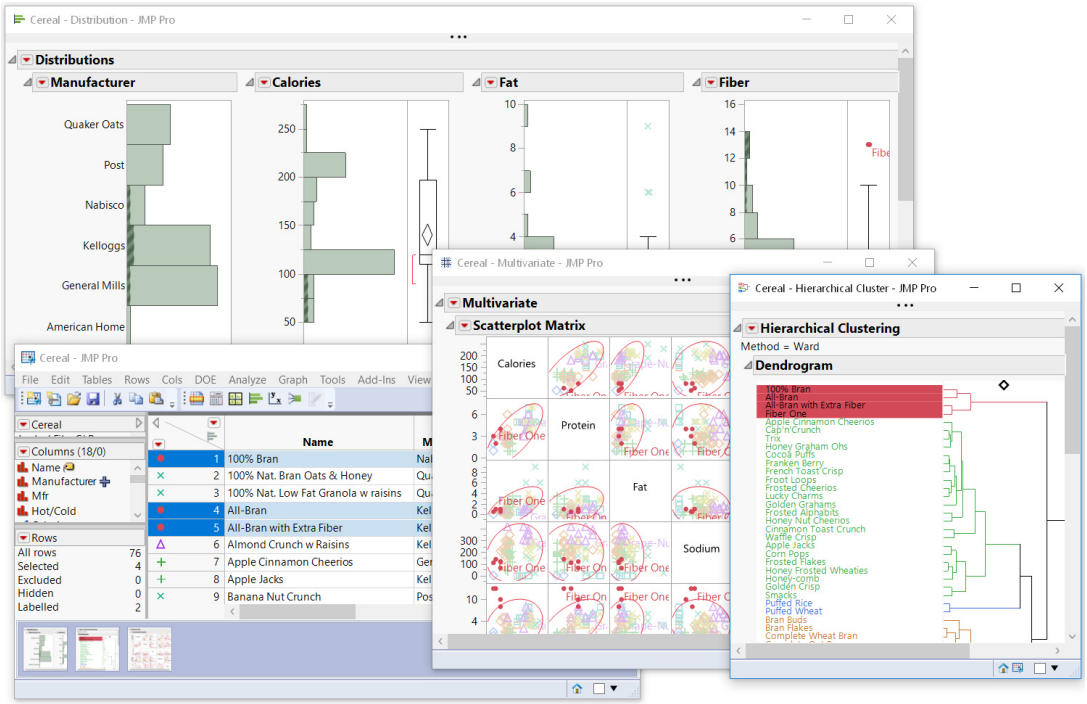
The Big Picture

Exploring Data in Multiple Platforms

JMP provides a host of statistical discovery platforms to help you explore different aspects of your data. You might start with a simple look at individual variables in histograms and then progress to multivariate and cluster analyses to get a deeper look. Each step of the way, you learn more about your data.

This chapter steps through an analysis of the Cereal.jmp sample data table that is installed with JMP. You learn how to explore the data in the Distribution, Multivariate, and Hierarchical Clustering platforms.

Figure 6.1 Linked Analyses in JMP



Contents

Fun Fact: Linked Analyses 173
Explore Data in Multiple Platforms 173
Analyze Distributions in the Distribution Platform 173
Analyze Patterns and Relationships in the Multivariate Platform 177
Analyze Similar Values in the Clustering Platform 181

Fun Fact: Linked Analyses

One of the powerful features in JMP is its linked analyses. The graphs and reports that you create are linked to each other through the data table. As shown in Figure 6.1, data that are selected in the data table are also selected in the three report windows. The linked analyses enable you to select data in one window and see where it occurs in the other windows. As you work through the examples in this chapter, keep the JMP windows open to see these interactions yourself.

Explore Data in Multiple Platforms

Which cereals are part of a healthy diet? The Cereal.jmp sample data (real data gathered from boxes of popular cereals) provides statistics on fiber content, calories, and other nutritional information. To identify the most healthful cereals, you step through interpreting histograms and descriptive statistics, correlations and outlier detection, scatterplots, and cluster analysis.

Analyze Distributions in the Distribution Platform

The Distribution platform illustrates the distribution of a single variable (*univariate* analysis) using histograms, additional graphs, and reports. The word *univariate* simply means involving one variable instead of two (bivariate) or many (multivariate). However, you can examine the distribution of several individual variables within a single report. The report content for each variable changes depending on whether the variable is categorical (nominal or ordinal) or continuous.

- For categorical variables, the initial graph is a histogram. The histogram shows a bar for each level of the ordinal or nominal variable. The reports show counts and proportions.
- For continuous variables, the initial graphs show a histogram and an outlier box plot. The histogram shows a bar for grouped values of the continuous variable. The reports show selected quantiles and summary statistics.

Once you know how your data are distributed, you can plan the appropriate type of analysis going forward.

Note: For more information about the Distribution platform, see *Basic Analysis*.

Scenario

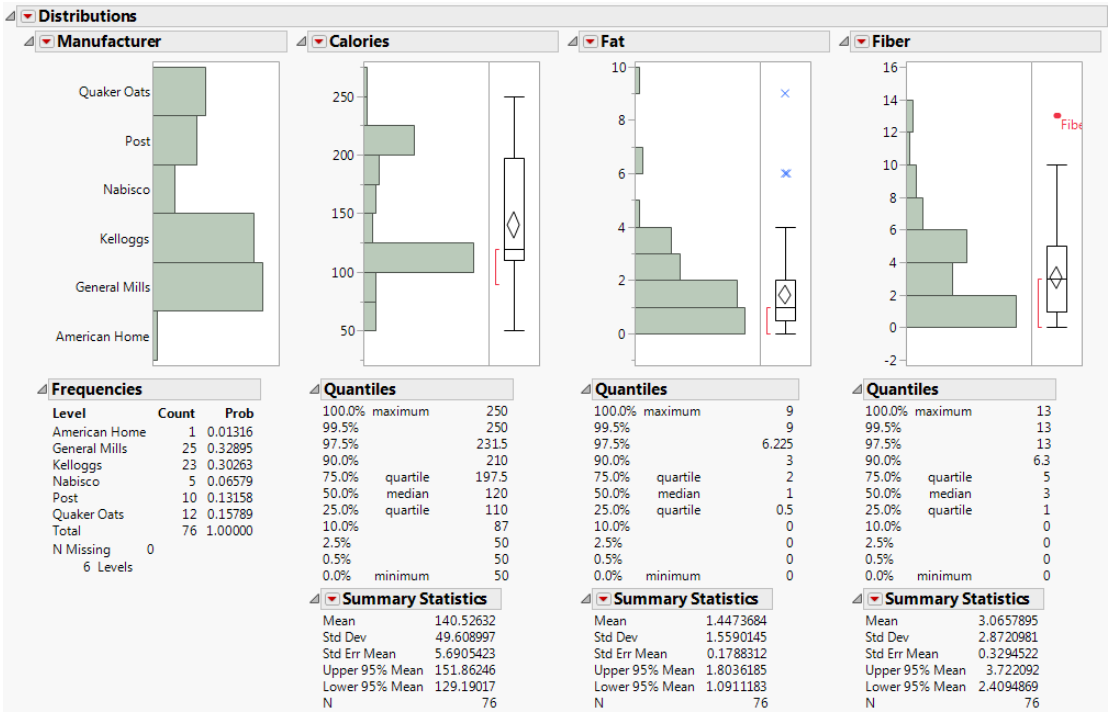
You want to view the nutritional values of cereals so that you can eat a more healthful diet. Analyzing distributions of cereal data reveals answers to the following questions:

- Which cereals are highest in fiber?
- What is the average, minimum, and maximum number of calories?
- What is the median amount of fat?
- Which cereal contains the most fat?
- Are there any outliers in the data?

Create the Distributions

1. Select **Help > Sample Data Library** and open Cereal.jmp.
2. Select **Analyze > Distribution**.
3. Press Ctrl and click Manufacturer, Calories, Fat, and Fiber.
4. Click **Y, Columns** and then click **OK**.

Figure 6.2 Distributions for Manufacturer, Calories, Fat, and Fiber



In the Fiber distributions, notice the following:

– Fiber One and All-Bran with Extra Fiber contain the most fiber as shown in the Fiber box plot. These cereals are outliers in terms of fiber content.

  The row that contains Fiber One in Cereal.jmp is labeled. This label shows the name of the cereal next to a data point in graphs. To see the entire label, drag the right-most vertical border to the right. Hover over the unlabeled data point to see "All Bran with Extra Fiber".

In the Fat distributions, notice the following:

– Hover over the top data point (the x marker) in the Fat box plot to see that 100% Nat. Bran Oats & Honey is the highest in fat.

– In the Fat Quantiles report, the median amount of fat is 1 gram.

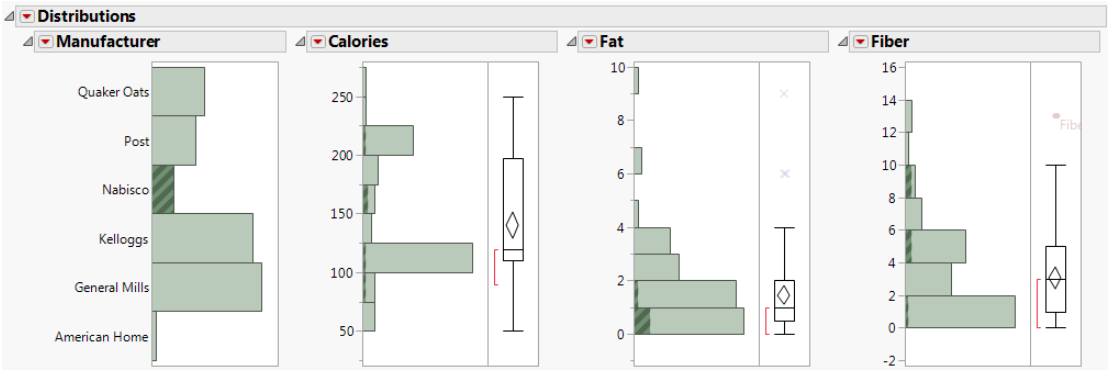
In the Calories Quantiles report, notice the following:

– The maximum number of calories is 250.

– The minimum number of calories is 50.

5. In the Manufacturer histogram, click the bar for Nabisco.

Figure 6.3 Distributions for Nabisco Cereals

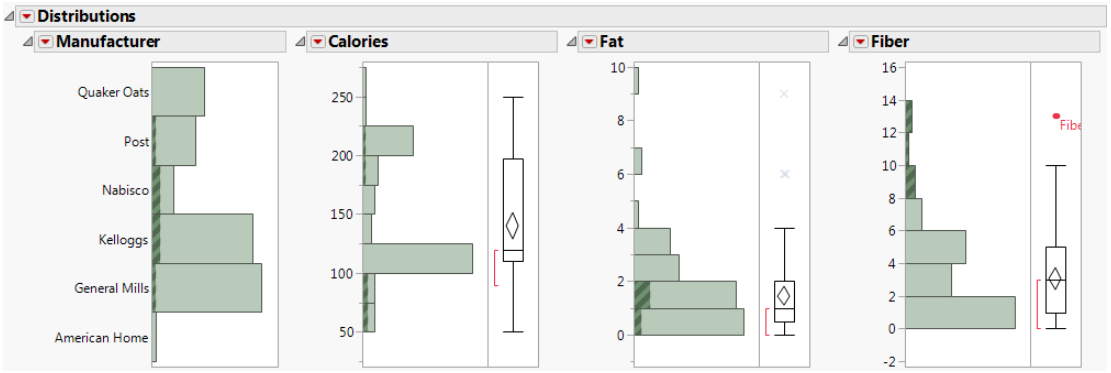


The Calories, Fat, and Fiber distributions for Nabisco cereals are highlighted in the other histograms. You can view the Calories, Fat, and Fiber distributions for the Nabisco cereals relative to the Calories, Fat, and Fiber distributions for the overall data. For example, the Fat distribution of Nabisco cereals seems to be lower than the Fat distribution for the overall data.

6. Click below the last Fiber bar to deselect all bars.

7. Press Shift and, in the Fiber histogram, click all histogram bars with a value above 8.

Figure 6.4 High-Fiber Cereals

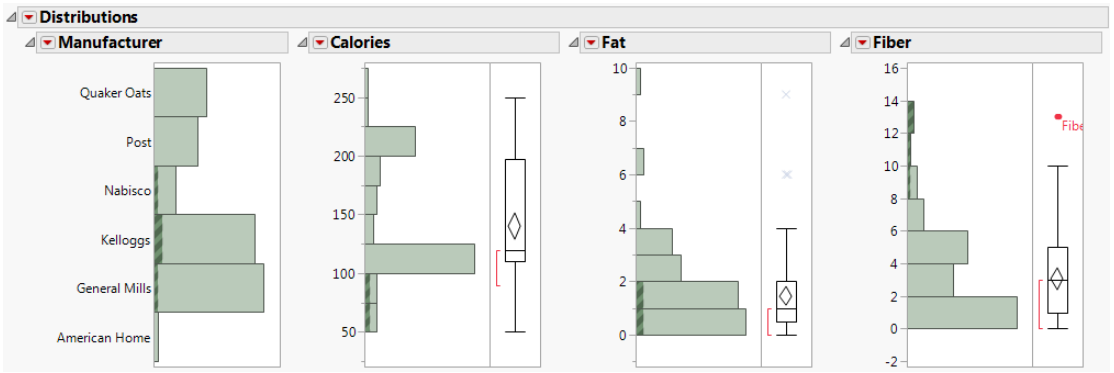


The highest-fiber cereals are highlighted in the Calories and Fat histograms. Because the histograms are linked, note that some of the high-fiber cereals are also low in fat.

8. Press Ctrl and Shift and deselect the two Calories histogram bars that are at or near 200.

   High calorie cereals are eliminated from the histograms.

Figure 6.5 High-Fiber and Low-Calorie Cereals



Tip: Leave the Distributions report open. You will use it later in a cluster analysis. See "Analyze Similar Values in the Clustering Platform".

Interpret the Results

Looking at the results, you can answer the following questions:

Which cereals are highest in fiber? The Fiber box plot shows that All-Bran with Extra Fiber and Fiber One have the highest amount of fiber. These two cereals are outliers.

What is the average, minimum, and maximum number of calories? The Calories histogram shows that the number of calories ranges from 50 to 275. The Calories Quantiles show that the number of calories ranges from 50 to 250, and the median number of calories is 120. The distribution is not uniform.

What is the median amount of fat? The Fat Quantiles report shows that the median amount of fat is 1 gram.

Which cereal contains the most fat? The Fat box plot shows that 100% Nat. Bran Oats & Honey is the highest in fat. This cereal is an outlier.

Draw Conclusions

To increase the amount of fiber in your diet, you decide to try All-Bran with Extra Fiber and Fiber One. These cereals are lower in calories and fat. Most cereals do not greatly increase the amount of fat in your diet, but you plan to avoid the high fat 100% Nat. Bran Oats & Honey. And although most cereals are relatively low in fat, they are not necessarily low in calories.

Analyze Patterns and Relationships in the Multivariate Platform

Now that you have identified which cereals to eat or avoid, you want to see how the cereal variables relate to each other. The Multivariate platform enables you to observe patterns and relationships between variables. From the Multivariate report, you can do the following:

- summarize the strength of the linear relationships between each pair of response variables using the Correlations table
- identify dependencies, outliers, and clusters using the Scatterplot Matrix
- use other techniques to examine multiple variables, such as partial, inverse, and pairwise correlations, covariance matrices, and principal components

Note: For more information about the Multivariate platform, see *Multivariate Methods*.

Scenario

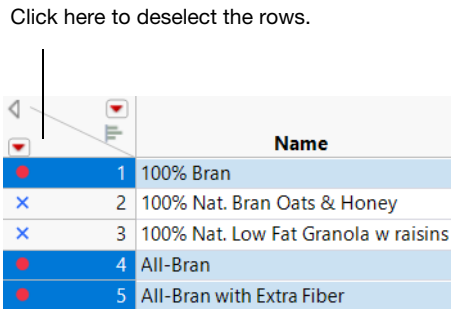
You want to see the relationships between variables such as fat and calories. Analyzing the cereal data in the Multivariate platform reveals answers to the following questions:

- Which pairs of variables are highly correlated?
- Which pairs of variables are not correlated?

Create the Multivariate Report

1. In the Cereal.jmp data table, click the bottom triangle at the top of the Columns panel to deselect the rows.

Figure 6.6 Deselecting Rows



2. Select **Analyze > Multivariate Methods > Multivariate**.
3. Select Calories through Potassium, click **Y, Columns**, and then click **OK**.

   The Multivariate report appears. The report contains the Correlations report and Scatterplot Matrix by default. The Correlations report is a matrix of correlation coefficients that summarizes the strength of the linear relationships between each pair of response (Y) variables. The dark numbers indicate a lower degree of correlation.

Figure 6.7 Correlations Report

Correlations

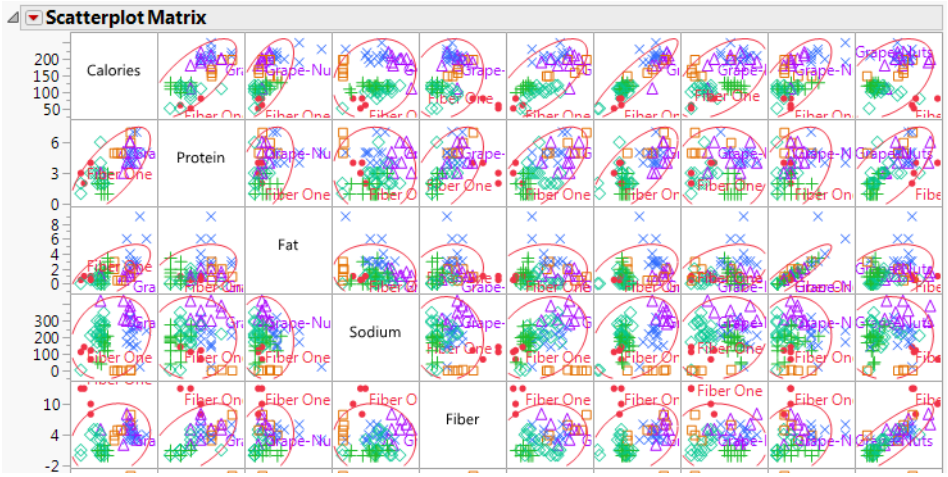
	Calories	Protein	Fat	Sodium	Fiber	Complex Carbos	Tot Carbo	Sugars	Calories fr Fat	Potassium
Calories	1.0000	0.7041	0.6460	0.1996	0.1953	0.6688	0.9076	0.5060	0.6709	0.4451
Protein	0.7041	1.0000	0.4080	0.0050	0.5470	0.6486	0.6937	-0.0010	0.4288	0.6782
Fat	0.6460	0.4080	1.0000	-0.0768	0.1824	0.1037	0.3860	0.4148	0.9013	0.3420
Sodium	0.1996	0.0050	-0.0768	1.0000	-0.0448	0.2619	0.3066	0.1767	0.0572	0.0459
Fiber	0.1953	0.5470	0.1824	-0.0448	1.0000	0.1769	0.3668	-0.1264	0.2553	0.8326
Complex Carbos	0.6688	0.6486	0.1037	0.2619	0.1769	1.0000	0.7773	-0.1601	0.1558	0.2693
Tot Carbo	0.9076	0.6937	0.3860	0.3066	0.3668	0.7773	1.0000	0.4263	0.4636	0.5375
Sugars	0.5060	-0.0010	0.4148	0.1767	-0.1264	-0.1601	0.4263	1.0000	0.4369	0.1166
Calories fr Fat	0.6709	0.4288	0.9013	0.0572	0.2553	0.1558	0.4636	0.4369	1.0000	0.3694
Potassium	0.4451	0.6782	0.3420	0.0459	0.8326	0.2693	0.5375	0.1166	0.3694	1.0000

Note the following:

- In the Calories column, the number of calories is highly correlated with all variables except for sodium and fiber.
- In the Fiber column, fiber and potassium appear to be highly correlated.
- In the Sodium column, sodium is not highly correlated with the other variables.

The density ellipses in the Scatterplot Matrix further illustrates relationships between variables.

Figure 6.8 Portion of the Scatterplot Matrix



By default, a 95% bivariate normal density ellipse is in each scatterplot. Assuming that each pair of variables has a bivariate normal distribution, this ellipse encloses approximately 95% of the points. If the ellipse is fairly round and is not diagonally oriented, the variables are uncorrelated. If the ellipse is narrow and diagonally oriented, the variables are correlated.

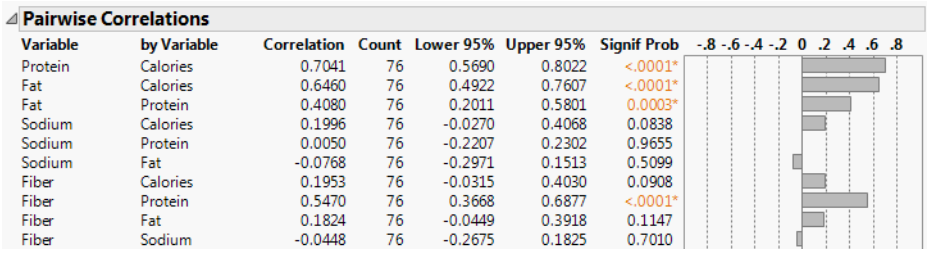
Note the following:

- The ellipses are fairly round in the Sodium row. This shape indicates that Sodium is uncorrelated with other variables.
- The blue x markers, which represent Nat. Bran Oats & Honey, Cracklin' Oat Bran, and Banana Nut Crunch, appear outside the ellipses in the Fat row. This placement indicates that the datum is an outlier (because of the amount of fat in the cereal).

You will further explore a scatterplot matrix later.

4. Click the Multivariate red triangle and select **Pairwise Correlations** to show the Pairwise Correlations report.

Figure 6.9 Portion of the Pairwise Correlations Report



Pairwise Correlations

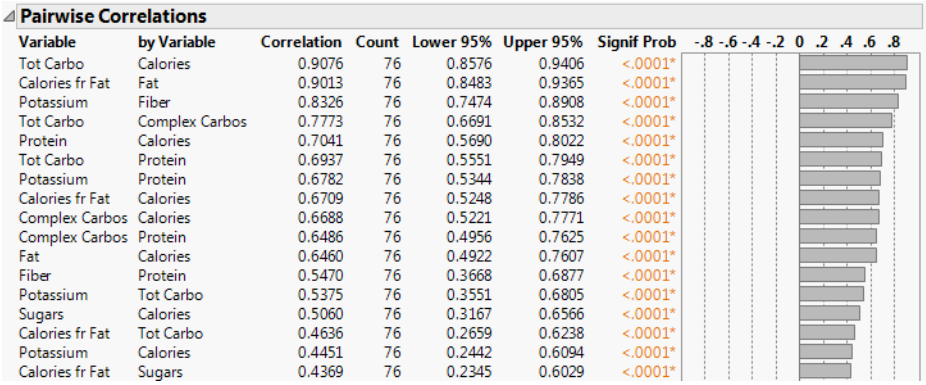
Variable	by Variable	Correlation	Count	Lower 95%	Upper 95%	Signif Prob
Protein	Calories	0.7041	76	0.5690	0.8022	<.0001*
Fat	Calories	0.6460	76	0.4922	0.7607	<.0001*
Fat	Protein	0.4080	76	0.2011	0.5801	0.0003*
Sodium	Calories	0.1996	76	-0.0270	0.4068	0.0838
Sodium	Protein	0.0050	76	-0.2207	0.2302	0.9655
Sodium	Fat	-0.0768	76	-0.2971	0.1513	0.5099
Fiber	Calories	0.1953	76	-0.0315	0.4030	0.0908
Fiber	Protein	0.5470	76	0.3668	0.6877	<.0001*
Fiber	Fat	0.1824	76	-0.0449	0.3918	0.1147
Fiber	Sodium	-0.0448	76	-0.2675	0.1825	0.7010

The Pairwise Correlations report lists the Pearson product-moment correlations for each pair of Y variables. The report also shows significance probabilities and compares the correlations in a bar chart.

5. To quickly see which pairs are highly correlated, right-click in the report and select the **Sort by Column**, **Signif Prob**, **Ascending** check box, and then click **OK**.

   The most highly correlated pairs appear at the top of the report. The small *p*-values for the pairs indicate evidence of correlation. The most significant correlation is between Tot Carbo (total carbohydrates) and Calories.

Figure 6.10 Small *p*-values for Pairs



Pairwise Correlations

Variable	by Variable	Correlation	Count	Lower 95%	Upper 95%	Signif Prob
Tot Carbo	Calories	0.9076	76	0.8576	0.9406	<.0001*
Calories fr Fat	Fat	0.9013	76	0.8483	0.9365	<.0001*
Potassium	Fiber	0.8326	76	0.7474	0.8908	<.0001*
Tot Carbo	Complex Carbos	0.7773	76	0.6691	0.8532	<.0001*
Protein	Calories	0.7041	76	0.5690	0.8022	<.0001*
Tot Carbo	Protein	0.6937	76	0.5551	0.7949	<.0001*
Potassium	Protein	0.6782	76	0.5344	0.7838	<.0001*
Calories fr Fat	Calories	0.6709	76	0.5248	0.7786	<.0001*
Complex Carbos	Calories	0.6688	76	0.5221	0.7771	<.0001*
Complex Carbos	Protein	0.6486	76	0.4956	0.7625	<.0001*
Fat	Calories	0.6460	76	0.4922	0.7607	<.0001*
Fiber	Protein	0.5470	76	0.3668	0.6877	<.0001*
Potassium	Tot Carbo	0.5375	76	0.3551	0.6805	<.0001*
Sugars	Calories	0.5060	76	0.3167	0.6566	<.0001*
Calories fr Fat	Tot Carbo	0.4636	76	0.2659	0.6238	<.0001*
Potassium	Calories	0.4451	76	0.2442	0.6094	<.0001*
Calories fr Fat	Sugars	0.4369	76	0.2345	0.6029	<.0001*

Interpret the Results

Looking at the results, you can answer the following questions:

Which pairs of variables are highly correlated? The Correlations report and Scatterplot Matrix show that the number of calories is highly correlated with all variables except for sodium and fiber. The Pairwise Correlations report shows that Tot Carbo (total carbohydrates) and Calories is the most correlated pair of variables.

Which pairs of variables are not correlated? The Correlations report and Scatterplot Matrix show that Sodium is not correlated with the other variables.

Draw Conclusions

You confirm the previous decision to avoid the high fat 100% Nat. Bran Oats & Honey. Trying All-Bran with Extra Fiber and Fiber One was also a smart decision. These two high-fiber cereals have the added benefit of contributing a lower number of calories, fat, and sugars and a higher amount of potassium. You also decide to avoid high-carbohydrate cereals because they likely contain a large number of calories.

Analyze Similar Values in the Clustering Platform

Clustering is a multivariate technique that groups observations together that share similar values across a number of variables. Hierarchical clustering combines rows in a hierarchical sequence that is portrayed as a tree. Cereals with certain characteristics, such as high fiber, are grouped in clusters so that you can view similarities among cereals.

Note: For more information about hierarchical clustering, see *Multivariate Methods*.

Scenario

You want to know which cereals are similar to each other and which ones are dissimilar. Analyzing clusters of cereal data reveals answers to the following questions:

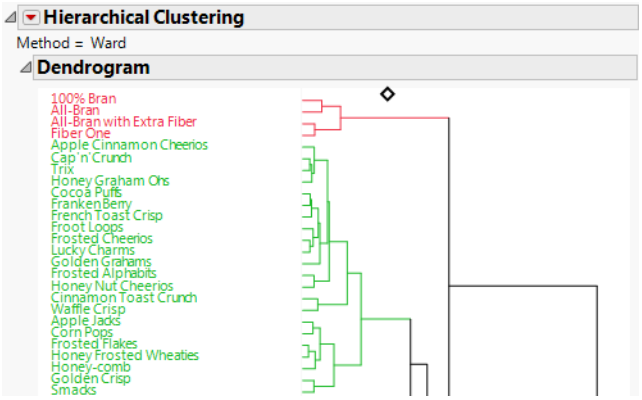
- Which cluster of cereals provides little nutritional value?
- Which cluster of cereals is high in vitamins and minerals and contains a low amount of sugar and fat?
- Which cluster of cereals is high in fiber and low in calories?

Create the Hierarchical Cluster Graph

1. With Cereal.jmp displayed, select **Analyze > Clustering > Hierarchical Cluster**.
2. Select Calories through Enriched, click **Y, Columns**, and then click **OK**.

   The Hierarchical Clustering report appears. The clusters are colored according to the data table row states.

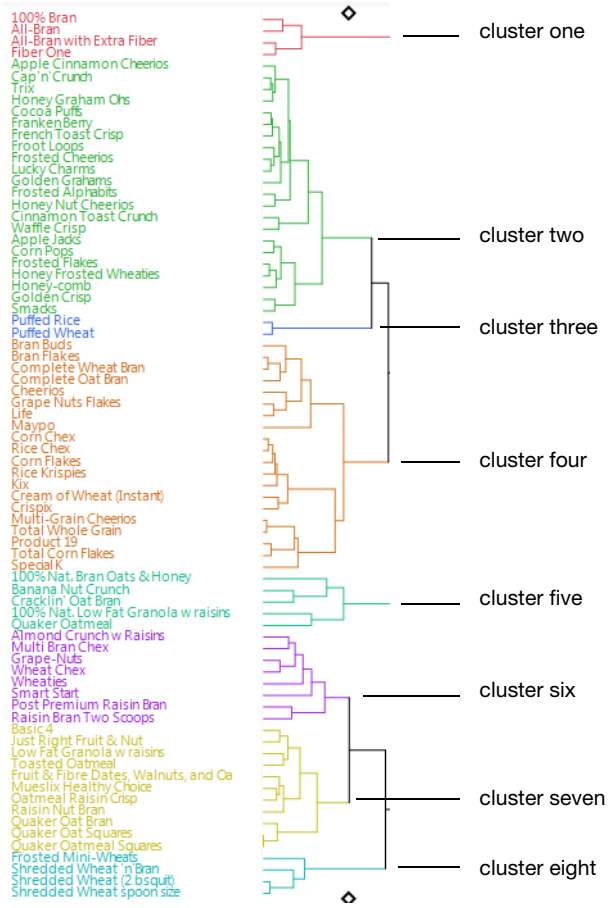
Figure 6.11 Portion of the Hierarchical Clustering Report



3. Click the Hierarchical Clustering red triangle and select **Color Clusters**.

   The clusters are colored according to their relationships in the dendrogram.

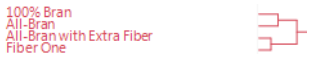
Figure 6.12 Colored Clusters



The cereals have similar characteristics within each cluster. For example, judging by the names of the cereals in cluster one, you guess that the cereals are high in fiber.

Notice how All-Bran with Extra Fiber and Fiber One are grouped in cluster one. These cereals are more similar to each other than the other two cereals in the cluster.

Figure 6.13 Similar Cereals in Cluster One



4. To select cluster one, click the red horizontal line on the right.

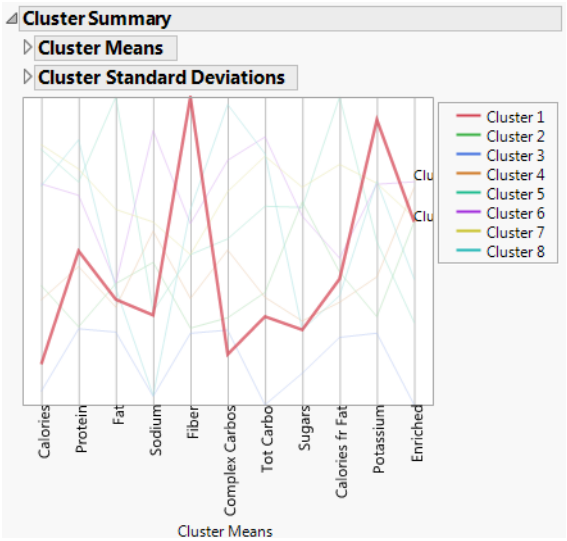
   The four cereals are highlighted in red.

Figure 6.14 Selecting a Cluster

5. To see the similar characteristics in the cluster, click the Hierarchical Clustering red triangle and select **Cluster Summary**.

   The Cluster Summary graph at the bottom of the report shows the mean value of each variable across each cluster. For example, the cereals in this cluster contain more fiber and potassium than cereals in other clusters.

Figure 6.15 Cluster Summary

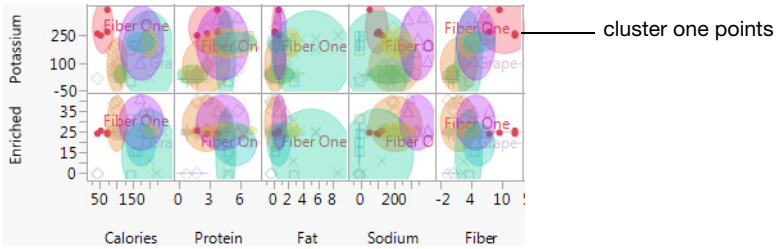


6. Click the Hierarchical Clustering red triangle and select **Scatterplot Matrix**.

   This option is an alternative to creating a scatterplot matrix in the Multivariate platform.

   Note the Fiber plot in the Potassium row. The selected cereals are located on the right side of the plot between 8 and 13 grams. This location indicates that the cereals in cluster one are high in fiber and potassium.

Figure 6.16 Cluster One Characteristics



Note: The points are also selected in the previous scatterplot matrix that you created if it is still open.

Interpret the Results

Clicking through the clusters and looking at the Cluster Summary report, you can see the following characteristics:

- Cluster one cereals, such as Fiber One and All-Bran, contain high fiber and potassium and low calories.
- Cluster two cereals, which contain many favorite children's cereals, are high in sugar and low in fiber, complex carbohydrates, and protein.
- Cluster three cereals (Puffed Rice and Puffed Wheat) are low in calories but provide little nutritional value.
- Cluster four cereals, such as Total Corn Flakes and Multi-Grain Cheerios, provide 100% of your daily requirement of vitamins and minerals. They are low in fat, fiber, and sugar.
- Cluster five cereals are high in protein and fat and low in sodium. The cluster consists of cereals such as Banana Nut Crunch and Quaker Oatmeal.
- Cluster six cereals are low in fat and high in sodium and carbohydrates. Traditional cereals such as Wheaties and Grape-Nuts are in this cluster.
- Cluster seven cereals are high in calories and low in fiber. Many cereals that include dried fruit are in this cluster (Mueslix Healthy Choice, Low Fat Granola w Raisins, Oatmeal Raisin Crisp, Raisin Nut Bran, and Just Right Fruit & Nut).
- Cluster eight cereals are low in sodium and sugar, and high in complex carbohydrates, protein, and potassium. Shredded Wheat and Mini-Wheat cereals are in this cluster.

By looking at the joins in the dendrogram, you can see which cereals in each cluster are most similar.

- In cluster one, Fiber One is similar in nutritional value to All-Bran with Extra Fiber. 100% Bran and All-Bran are also similar. Each pair of similar cereals are made by different companies, so the cereals are competing against each other.
- In cluster two, Frosted Flakes and Honey Frosted Wheaties are similar even though one is a corn flake and the other is a wheat flake. Lucky Charms and Frosted Cheerios are similar. Cap'n'Crunch and Trix are also similar.

Draw Conclusions

Based on your desire to eat more fiber and fewer calories, you decide to try the cereals in cluster one. You will avoid cereals in cluster three, which consists of puffed wheat and puffed rice and have little nutritional value. And you will try cereals in the highly nutritious cluster four.

Chapter 7

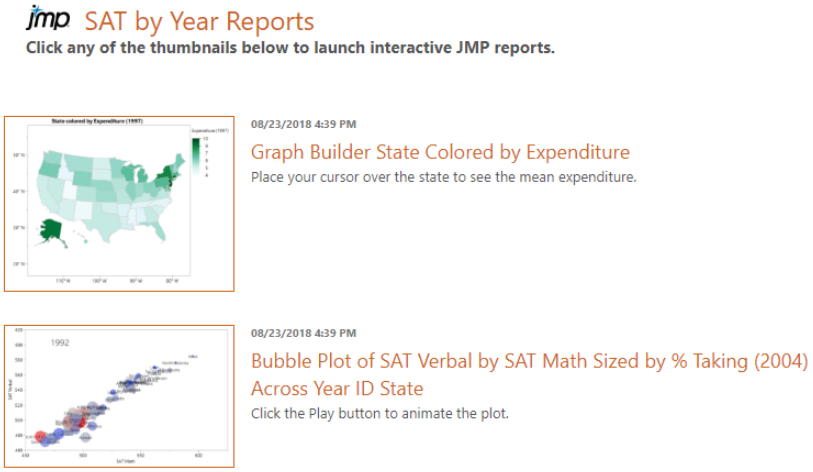
Save and Share Your Work

Save and Re-create Results

Once you have generated results from your data, JMP provides you with multiple ways to share your work with others. Here are some of the ways that you can share your work:

- Saving platform results as journals, projects, or web reports
- Saving results, data tables, and other files in projects
- Saving scripts to reproduce results in data tables
- Saving results as Interactive HTML (.htm, html)
- Saving results as a PowerPoint presentation (.pptx)
- Sharing results in a dashboard

Figure 7.1 Example of a Web Report



Contents

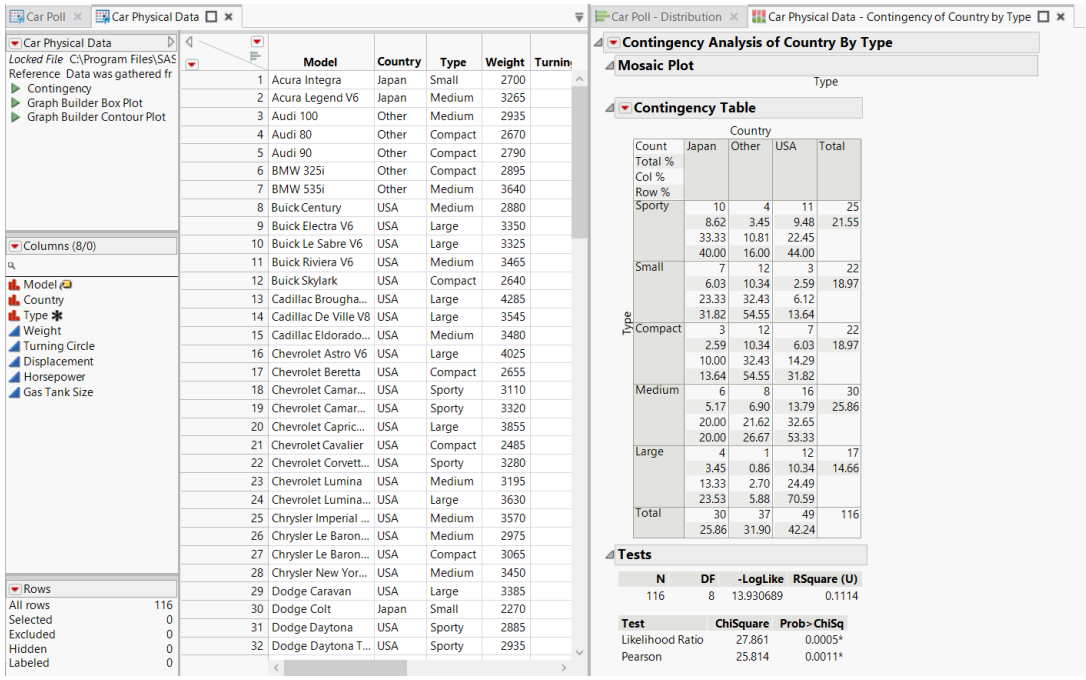
Work with Projects . . . 189
Create a New Project . . . 190
Open Files in a Project . . . 190
Rearrange Files in Projects . . . 191
Save a Project . . . 192
Project Workspace . . . 193
Project Bookmarks . . . 194
Project Contents . . . 194
Recent Files in Projects . . . 195
Project Log . . . 196
Move Files into Projects . . . 196
Write a Project On Open Script . . . 197
Example of Creating a New Project . . . 197
Save Platform Results in Journals . . . 200
Example of Creating a Journal . . . 200
Add Analyses to a Journal . . . 201
Save and Run Scripts . . . 202
Example of Saving and Running a Script . . . 202
About Scripts and JSL . . . 203
Save Reports as Interactive HTML . . . 204
Interactive HTML Contains Data . . . 205
Example of Creating Interactive HTML . . . 205
Share Reports as Interactive HTML . . . 206
Save a Report as a PowerPoint Presentation . . . 210
Create Dashboards . . . 211
Example of Combining Windows . . . 211
Example of Creating a Dashboard with Two Reports . . . 213

Work with Projects

With JMP projects, you can do the following:

- Explore your data more efficiently with a single, tabbed JMP window
- Quickly save and reopen a set of related files and reports
- Easily share your work by embedding your tables and scripts in a self-contained project file

Figure 7.2 Project File with Data Tables and Reports



This section contains the following information:

- "Create a New Project"
- "Open Files in a Project"
- "Rearrange Files in Projects"
- "Save a Project"
- "Project Workspace"
- "Project Bookmarks"
- "Project Contents"
- "Recent Files in Projects"
- "Project Log"
- "Move Files into Projects"
- "Write a Project On Open Script"
- "Example of Creating a New Project"

Create a New Project

To create a new, empty JMP project, select **File** > **New** > **Project** (Windows) or **File** > **New** > **New Project** (macOS).

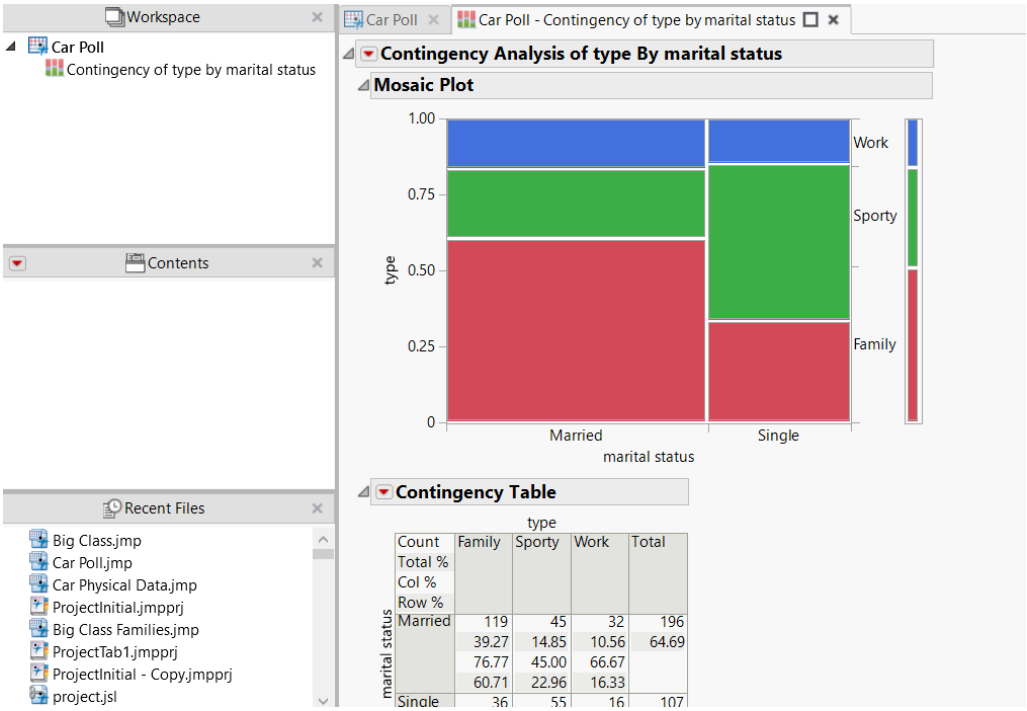
Open Files in a Project

In a JMP project, you can open data tables and then run analyses on the data. Each data table and analysis report opens in a new tab.

1. From a project window, select **File** > **Open** and navigate to the data tables that you want in the project.
2. From a data table, run an analysis.

If a data table is updated on your computer, any associated reports in your project update when you reopen the project.

Figure 7.3 Initial Project



Tip: You can use JMP keyboard shortcuts in projects, such as Ctrl+S to save a file, or Ctrl+W to close the active window pane. For a full list, select **Help** > **Quick Reference Card**.

Rearrange Files in Projects

In a JMP project, data tables and reports appear in individual tabs. You can rearrange tabs by dragging them into a dock zone.

Tip: To undo or redo a docked item, select **Project** > **Undo Layout** or **Project** > **Redo Layout**. To return all tables and reports to individual tabs, select **Project** > **Reset Layout**.

Example of Rearranging Project Files

1. Select **File** > **New** > **Project** (Windows) or **File** > **New** > **New Project** (macOS).
2. Select **Help** > **Sample Data Library** and open Car Physical Data.jmp and Car Poll.jmp.
3. In the Car Poll.jmp data table, run the Distribution script.
4. In the Car Physical Data.jmp data table, run the Contingency script.
5. Drag the **Car Poll - Distribution** report tab to the right and drop it into the *Dock right* zone.

The Distribution report pane appears at the right of the project window. The report is docked and stays visible when you switch between tabs.

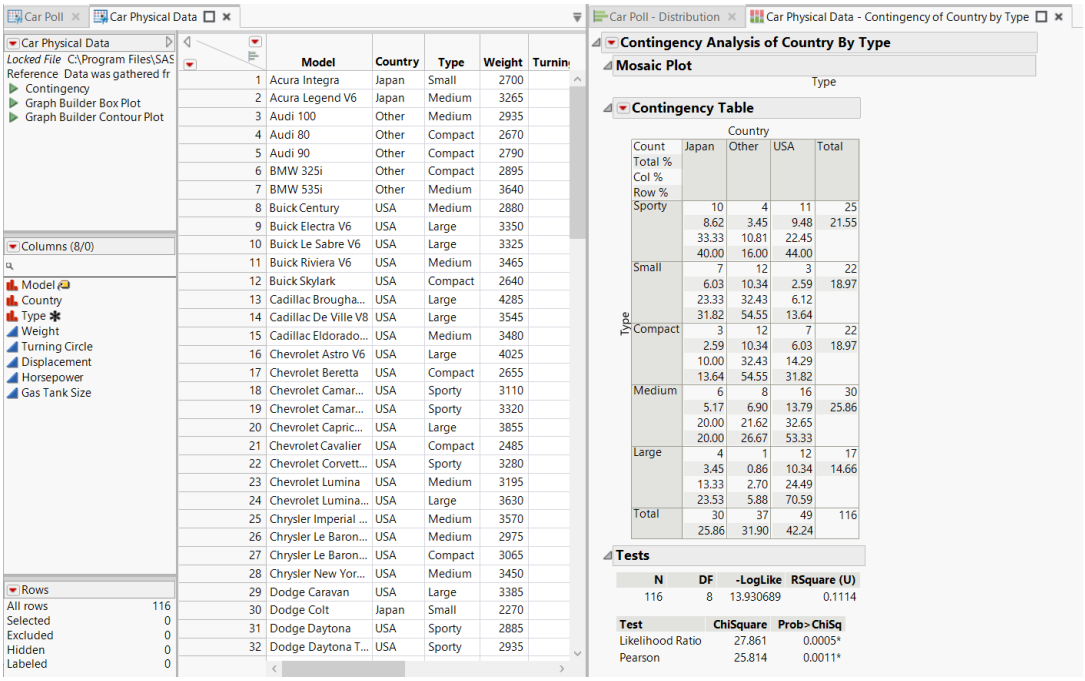
6. Drag the **Car Physical Data - Contingency** tab to the middle of the **Car Poll - Distribution** report and drop it into the *Dock tab* zone.

Tip: You can adjust the size of the data table window to fully show both reports.

7. Click the x to the right of Workspace, Contents, and Recent Files in the left column.

   You can show these panes later by selecting the options in the Project menu.

Figure 7.4 Reports and Data Tables Grouped into Tabs



The Distribution and Contingency reports are now tabbed together, as are the two data tables.

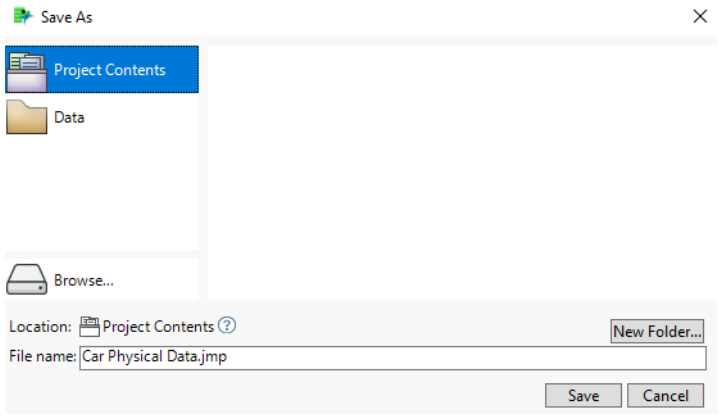
Save a Project

If you want to share, distribute, or archive your project, create a self-contained project by saving all your data tables and scripts to the project contents. This embeds all project files and folders into a single project file that you can share with other JMP users.

1. (Optional) To save the project as a self-contained project, save each data table and script in the project by selecting **File** > **Save As** and then select **Project Contents**. You can click **New Folder** to create a folder in the project contents.

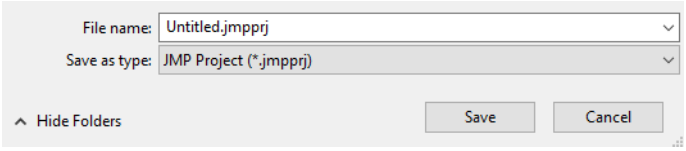
Note: You do not need to save reports as these are automatically saved to the project.

Figure 7.5 Save a Data Table to the Project Contents



2. Select **File** > **Save Project** to save the project file.

Figure 7.6 Save a Project File



Project Workspace

In a JMP project, you can see all currently open files in the Workspace tool pane. Reports appear under the corresponding data table. The active data table appears in bold.

Tip: To hide or show the Workspace tool pane, select **Project** > **Show Workspace**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

In the Workspace tool pane, you can do the following:

- Double-click a file to make it active.
- Right-click a file to close it, hide it (removes the tab and dims the file name), bookmark it, and more.

Project Bookmarks

In a JMP project, you can create a shortcut to a file or folder in the Bookmarks tool pane. You can also organize your bookmarked files and folders by creating a bookmark group.

Tip: To hide or show the Bookmarks tool pane, select **Project** > **Show Bookmarks**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

Bookmark an open file in the project

- Right-click a tab and select **Bookmark**.
- To bookmark all open files, right-click a tab and select **Bookmark All**.

Bookmark a file or folder on your computer

Drag a file or folder from your computer into the Bookmarks tool pane, or click the Bookmarks red triangle menu and select **Add Files** or **Add Folder**.

Note: If you add or remove files from a bookmarked folder on your computer, the folder contents automatically update in the project.

Open a bookmarked file

In the Bookmarks tool pane, double-click a file.

Remove a bookmark

In the Bookmarks tool pane, right-click a file and select **Remove Bookmark**.

Create a bookmark group

1. Click the Bookmarks red triangle menu and select **New Group**.
2. Name the group and click **OK**.
3. Drag new or existing bookmarks into the group, or in the Bookmarks tool pane, right-click the group and select **Add Files** or **Add Folder**.

Project Contents

In a self-contained JMP project, any files you save to the project contents appear in the Contents tool pane.

Tip: To hide or show the Contents tool pane, select **Project** > **Show Contents**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

Open a file from the project contents

In the Contents tool pane, double-click a file.

Create a folder in the project contents

1. Click the Contents red triangle menu and select **New Folder**.
2. Name the folder and click **OK**.

Move files into folders

In the Contents tool pane, drag a file into a folder.

Copy a file or folder from your computer into the project contents

1. Click the Contents red triangle menu and select **Copy Files into Project** or **Copy Folder into Project**.
2. Navigate to the file and click **Open**, or navigate to the folder and click **Select**.

Rename a file in the project contents

In the Contents tool pane, right-click a file and select **Rename**.

Delete a file in the project contents

In the Contents tool pane, right-click a file and select **Delete**.

Recent Files in Projects

In a JMP Project, you can open a file from the Recent Files tool pane.

Tip: To hide or show the Recent Files tool pane, select **Project** > **Show Recent Files**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

Open a file

In the Recent Files tool pane, double-click on a file or drag it into a project.

Note: Open files that are saved to the project contents do not appear in the Recent Files tool pane.

Project Log

In a JMP Project, you can see log messages in the Log tool pane.

Tips:

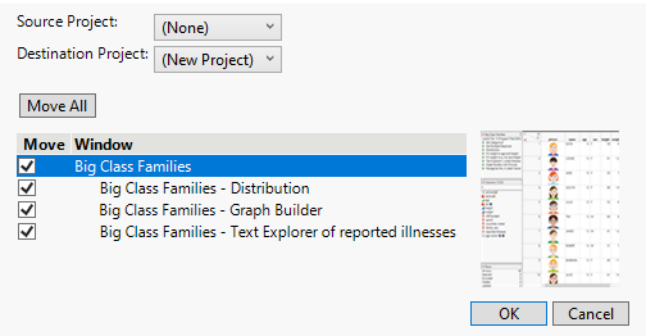
- To hide or show the Log tool pane, select **Project** > **Show Log**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.
- By default, log messages that occur in projects appear only in the project log, not in the main JMP log. To change this setting, select **File** > **Preferences** > **Projects** and update **Use Project Log Pane** to **If Open** or **Never**.

Move Files into Projects

In JMP, you can move files into a project, out of a project, or between projects.

1. Select **Help** > **Sample Data Library** and open Big Class Families.jmp.
2. Run the Distribution, Text Explorer, and Graph Builder scripts.
3. From any window, select **Window** > **Move to/from Project**.
4. Leave the Source Project as **(None)** to see files that are not open in any project.
5. Leave the Destination Project as **(New Project)** to move the selected files into a newly created project.
6. Select the check box next to Big Class Families.jmp. The graphs associated with the data table are also selected.

Figure 7.7 Move Windows To/From Project



7. Click **OK**. A new project is created with the selected table and reports.

Write a Project On Open Script

In the Project On Open Script box, you can add a JSL script that will run each time this specific project is opened. For example, you can create a script that creates or modifies reports, prompts the user for information, or writes usage information to the log window.

1. Select **Project** > **Project Settings**.
2. Paste the JSL script and click **OK**.

Tips:

- See the *Scripting Guide*, which explains how to create a start-up script when *any* project is opened (instead of a specific project).
- To use an existing project as a template for new projects, specify the existing project as a new project template under **File** > **Preferences** > **Projects**.

Example of Creating a New Project

In this example, you create a project, import data, generate an analysis and dock the report in the project window, create a subset table, and then save, close, and reopen the project.

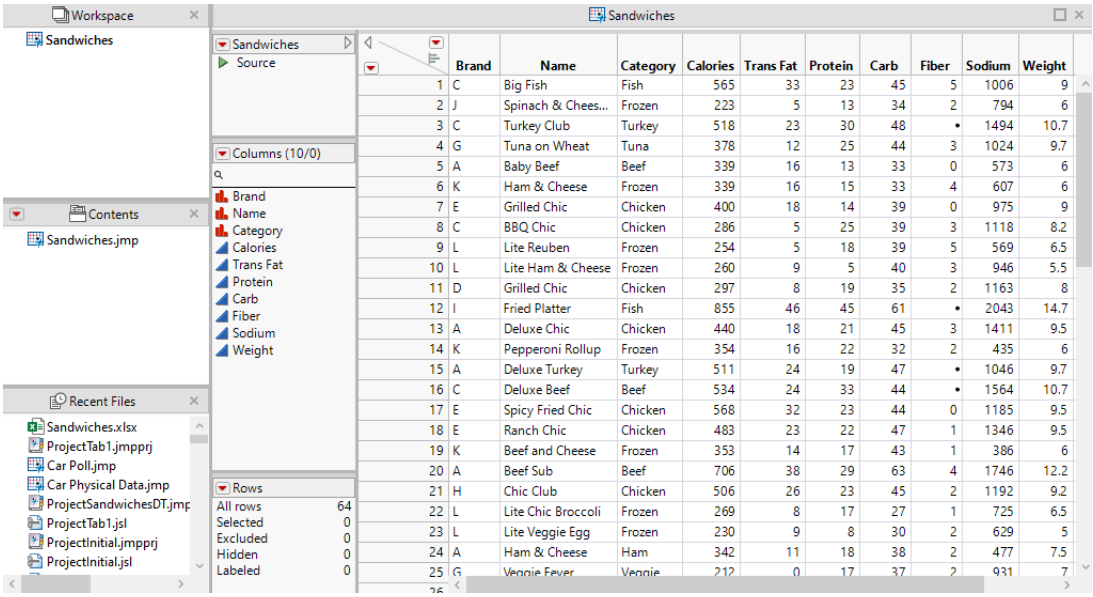
1. Select **File** > **New** > **Project** (Windows) or **File** > **New** > **New Project** (macOS).
2. From the project window, select **File** > **Open**.
3. Open the sandwiches.xlsx file, located here by default:

   C:/Program Files/SAS/JMP/16/Samples/Import Data

   Tip: At the bottom, you might need to change **All JMP Files** to **Excel Files**.

4. Click **Import**.
5. Select **File** > **Save**.
6. Make sure **Project Contents** is selected.
7. Change the file name to Sandwiches.jmp and click **Save**.

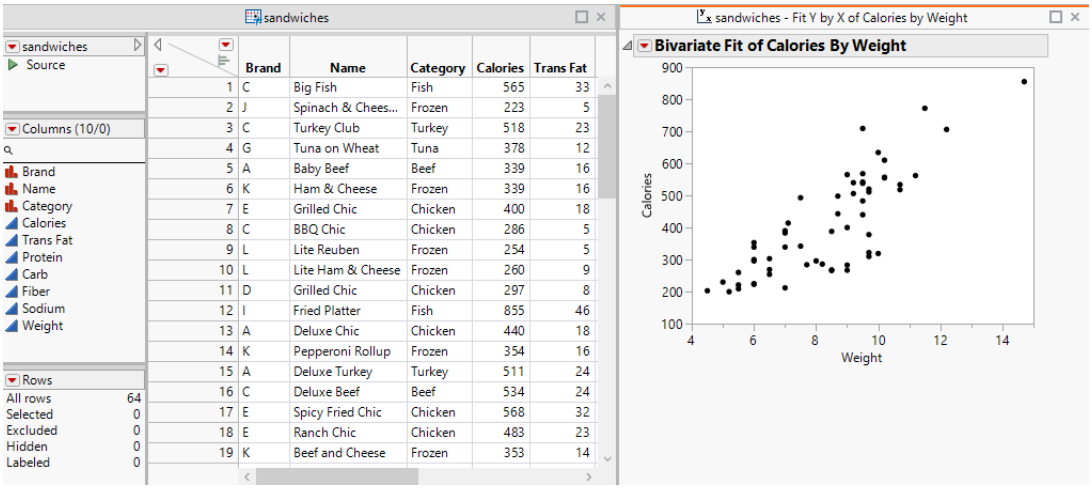
Figure 7.8 Project with Sandwiches Data Imported



8. Select **Analyze** > **Fit Y by X**.
9. Select Calories and click **Y, Response**.
10. Select Weight and click **X, Factor**.
11. Click **OK**.
12. Drag the **Sandwiches - Fit Y by X** tab to the right and drop it into the *Dock right* zone.

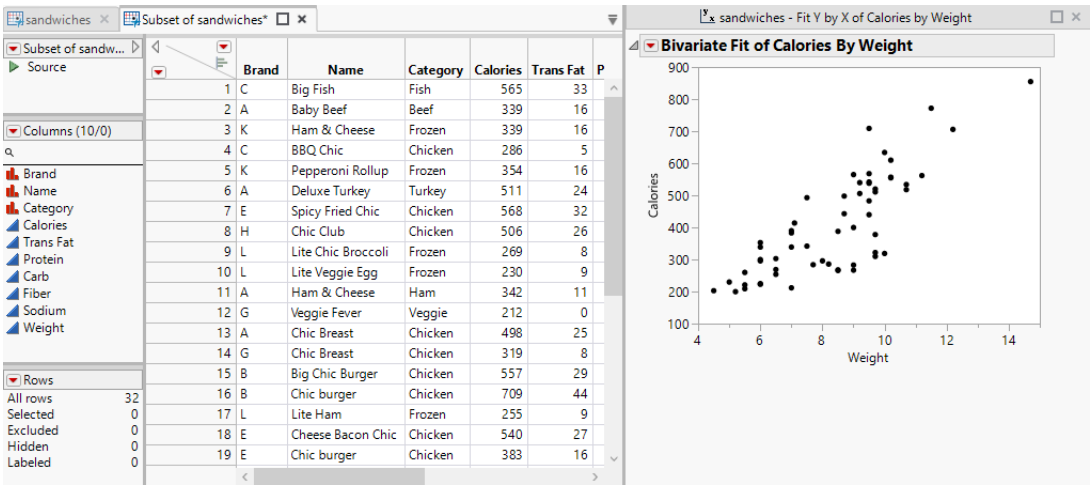
Tip: To show the entire report, you can drag the line between the data table and the report to the left.

Figure 7.9 Fit Y by X (Bivariate) Report Docked at Right



13. Select **Tables** > **Subset**.
14. Under Rows, select **Random: sampling rate 0.5**.
15. Click **OK**.

Figure 7.10 Project with Unsaved Subset Table



16. Select **File** > **Save Project**.
17. Navigate to the folder where you want to save your project, name the project file, and click **Save**.
18. Close the project.
19. Select **File** > **Open** and open your project file.

Notice that the subset table that you did not save has been saved automatically to the project file.

Save Platform Results in Journals

Save platform reports for future viewing by creating a journal of the report window. The journal is a copy of the report window. You can edit or append additional reports to an existing journal. The journal is not connected to the data table. A journal is an easy way to save the results from several report windows in a single report window that you can share with others.

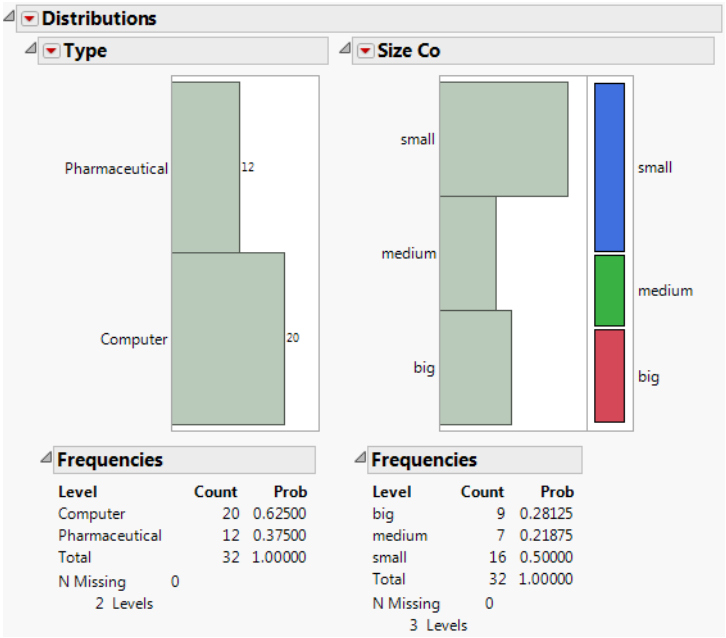
This section contains the following information:

- "Example of Creating a Journal"
- "Add Analyses to a Journal"

Example of Creating a Journal

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Distribution**.
3. Select both Type and Size Co and click **Y, Columns**.
4. Click **OK**.
5. Click the Type red triangle and select **Histogram Options** > **Show Counts**.
6. Click the Size Co red triangle and select **Mosaic Plot**.
7. Select **Edit** > **Journal** to journal these results. The results are duplicated in a journal window.

Figure 7.11 Journal of Distribution Results



The results in the journal are not connected to the data table. In the Type bar chart, if you click the Computer bar, no rows are selected in the data table.

Since the journal is a copy of your results, most of the red triangle menus do not exist. A journal does have a red triangle menu for each new report that you add to the journal. This menu has two options:

Rerun in new window If you have the original data table that was used to create the original report, this option runs the analysis again. The result is a new report window.

Edit Script This option opens a script window that contains a JSL script to re-create the analysis. JSL is a more advanced topic that is covered in the *Scripting Guide* and *JSL Syntax Reference*.

Add Analyses to a Journal

If you perform another analysis, you can add the results of the analysis to the existing journal.

1. With a journal open, select **Analyze** > **Distribution**.
2. Select profit/emp and click **Y, Columns**.
3. Click **OK**.
4. Select **Edit** > **Journal**. The results are appended to the bottom of the journal.

Save and Run Scripts

Most platform options in JMP are scriptable, meaning that most actions that you perform can be saved as a JMP Scripting Language (JSL) script. You can use a script to reproduce your actions or results at any time.

This section contains the following information:

- "Example of Saving and Running a Script"
- "About Scripts and JSL"

Example of Saving and Running a Script

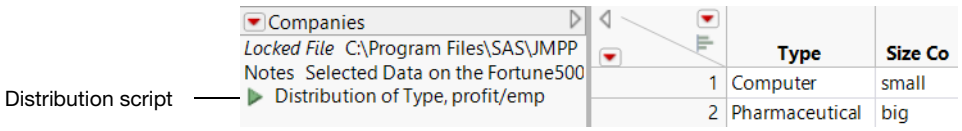
Create a Report

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Distribution**.
3. Select Type and profit/emp and click **Y, Columns**.
4. Click **OK**.
5. Click the Type red triangle and select these options:
   - **Histogram Options** > **Show Counts**
   - **Confidence Interval** > **0.95**
6. Click the profit/emp red triangle and select these options:
   - **Outlier Box Plot**, to remove the outlier box plot
   - **CDF Plot**
7. Click the Distributions red triangle and select **Stack**.

Save the Script to the Data Table and Run It

1. To save this analysis, click the Distributions red triangle and select **Save Script > To Data Table**. The new script appears in the Table panel.

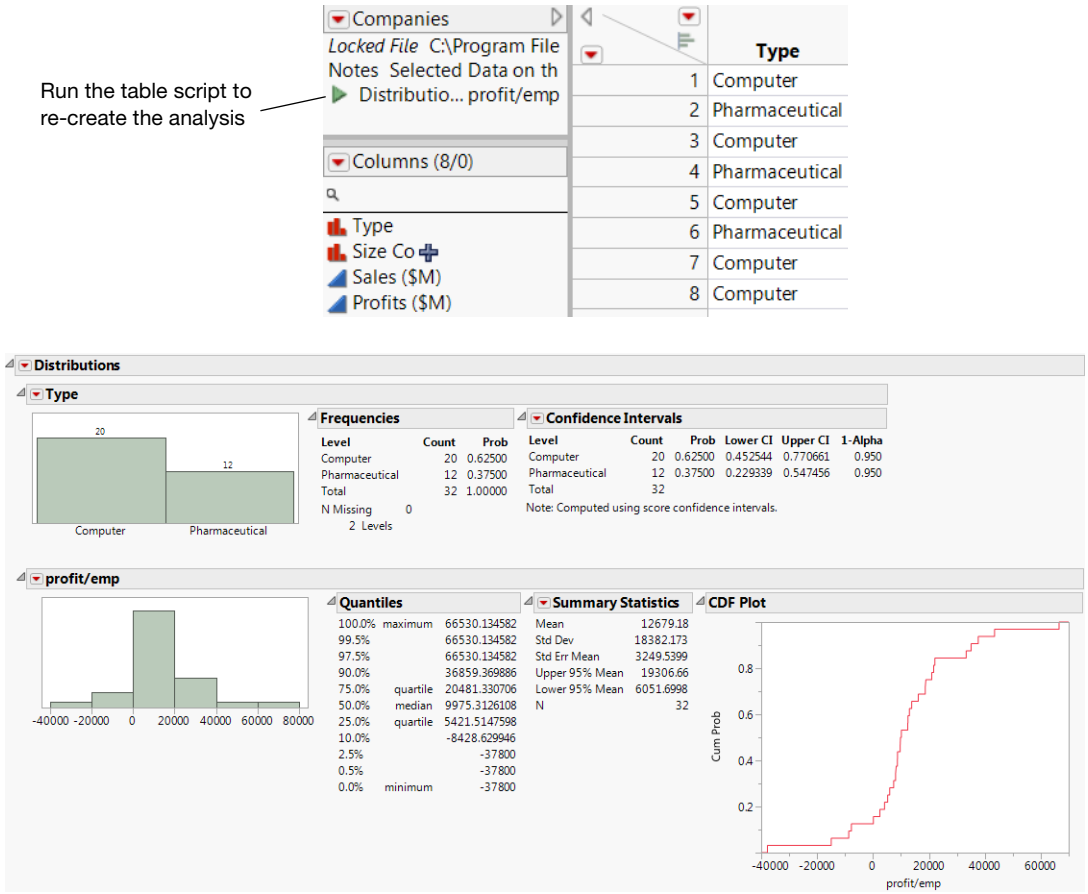
Figure 7.12 Distribution Script



2. Close the Distribution report window.

3. To re-create the analysis, click the green triangle next to the Distribution script.

Figure 7.13 Running the Distribution Script



Tip: Right-click the table script to view more options.

About Scripts and JSL

The script that you saved in this section contains JMP Scripting Language (JSL) commands. JSL is a more advanced topic that is covered in the *Scripting Guide* and *JSL Syntax Reference*.

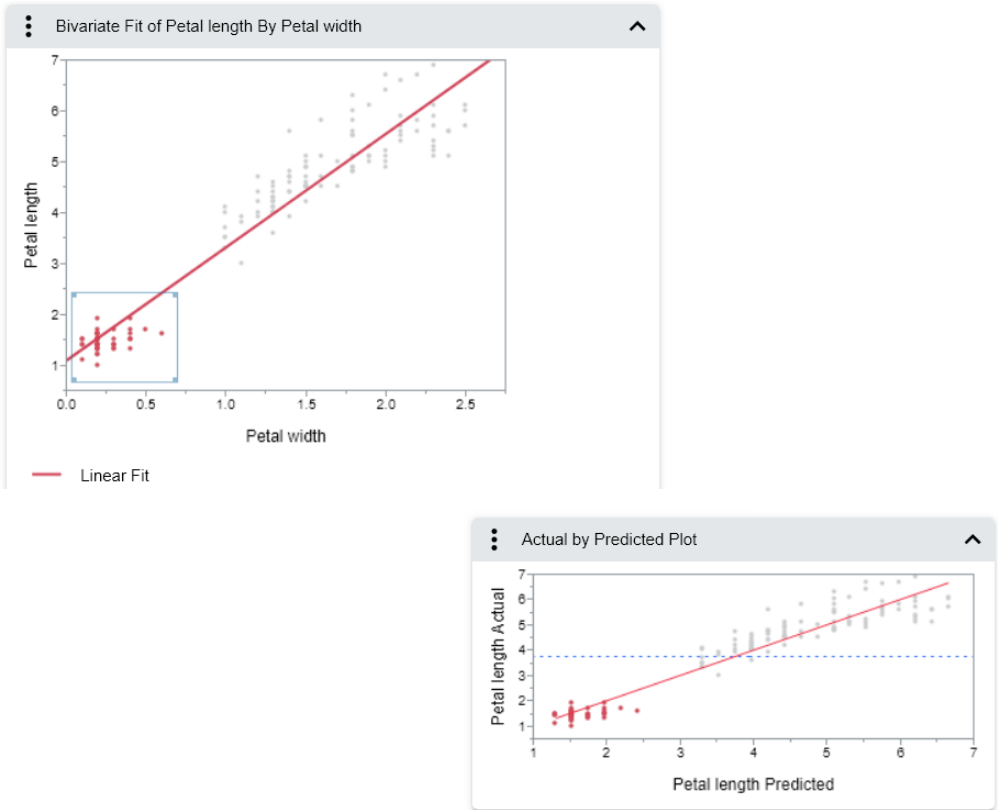
Save Reports as Interactive HTML

Interactive HTML enables JMP users to share reports that contain dynamic graphs so that even non JMP users can explore the data. The JMP report is saved as a web page in HTML 5 format, which you can email to users or publish on a website. Users then explore the data as they would in JMP.

Interactive HTML provides a subset of features from JMP:

- Explore interactive graph features, such as selecting histogram bars and viewing data values.
- View data by brushing.
- Show or hide report sections.
- Hover over the report for tooltips.
- Increase the marker size.

Figure 7.14 Brushing Data in Interactive HTML



Many changes that you make to the graphs, such as ordered variables, horizontal histograms, background colors, and colored data points, are saved in the web page. Graphs and tables that are closed when you save the content remain closed on the web page until the user opens them.

Interactive HTML Contains Data

When you save reports as interactive HTML in JMP, your data are embedded in the HTML. The content is unencrypted, because web browsers cannot read encrypted data. To avoid sharing sensitive data, save your results as a non-interactive web page. (Select **File > Export > Interactive HTML File with Data**.)

Example of Creating Interactive HTML

Create a Report

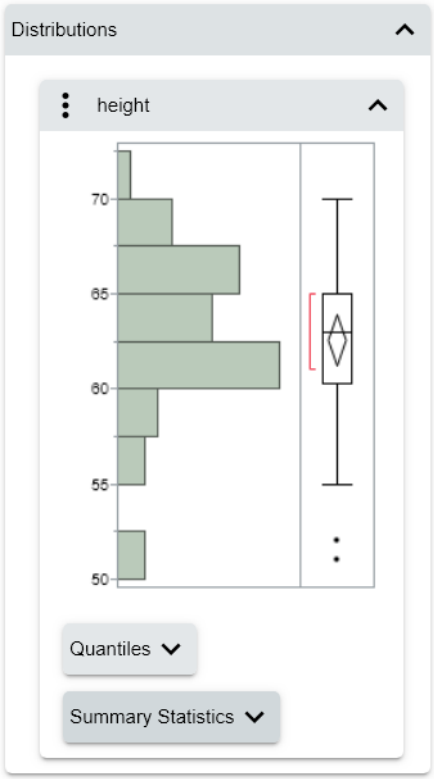
1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Distribution**.
3. Select height and click **Y, Columns**.
4. Click **OK**.

Save as Interactive HTML

1. (Windows) Select **File > Export**, select **Interactive HTML with Data**, and then click **OK**.
2. (macOS) Select **File > Export**, select **Interactive HTML with Data**, and then click **Next**.
3. On the Export window, select **Open the file after saving** if it's not already selected.
4. Name and save the file.

   The output appears in your default browser.

Figure 7.15 Interactive HTML Output



For information about exploring interactive HTML output, visit https://www.jmp.com/interactive.

Share Reports as Interactive HTML

Interactive HTML enables you to share JMP reports that contain dynamic graphs so that even non JMP users can explore the data. JMP reports are saved as an interactive web page that you can share with others (for example, on a shared network drive, by email, or on a website). Users then explore the data as they would in JMP.

Interactive HTML Contains Data

When you export or publish a report as interactive HTML, your data are embedded in the HTML. The content is unencrypted, because web browsers cannot read encrypted data. To avoid sharing sensitive data, export your results as a non-interactive web page instead by selecting **File > Export > HTML**.)

Which Features Are Supported Interactively

Interactive HTML provides a subset of features from JMP:

- If the features in your report are fully supported, the web page is created with no warnings.
- If your report contains unsupported features, you see a message in the export or publish window that interactive HTML is partially implemented. For details, see the JMP log (**View** > **Log**).
- Partially or unsupported features appear static in the web page. If you hover over an unsupported feature in the web page, a tooltip says that the feature is not yet interactive.

For information about exploring interactive HTML output, visit https://www.jmp.com/interactive.

Export a Single Report as Interactive HTML

To export a single JMP report as interactive HTML, use the Export as Interactive HTML with Data option to create a single web page.

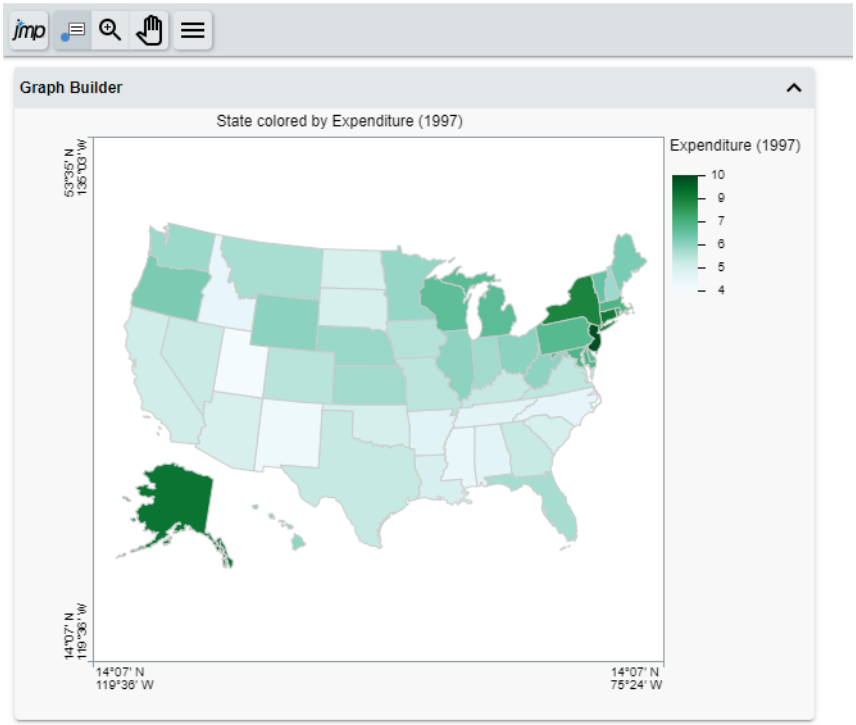
1. In JMP, create the report and make it the active window.

   Note: If a report contains a Local Data Filter, it is static in the web report and users cannot change the selections. To make the Local Data Filter interactive, deselect the **Include** mode. (In Graph Builder, to see the **Include** mode, select **Show Modes** from the Local Data Filter red triangle menu.)

2. Select **File > Export**, select **Interactive HTML with Data** and click **Next**.
3. Name the file.
4. (Optional) To open the HTML file in your default browser after exporting it, select **Open the file after saving**.
5. Click **Save**.

   The output is saved in the selected folder.

Figure 7.16 Web Page for Single Interactive HTML Report



Publish Multiple Reports as Interactive HTML

To publish multiple JMP reports as interactive HTML, use the Publish to File option, which creates an index page that contains the reports.

Tip: When you publish multiple JMP reports, you specify where to save the index page and report files. You can choose a shared network folder and provide the location to others, or choose a folder on your computer and zip the files before sharing. This is particularly helpful if you are sharing with non JMP users.

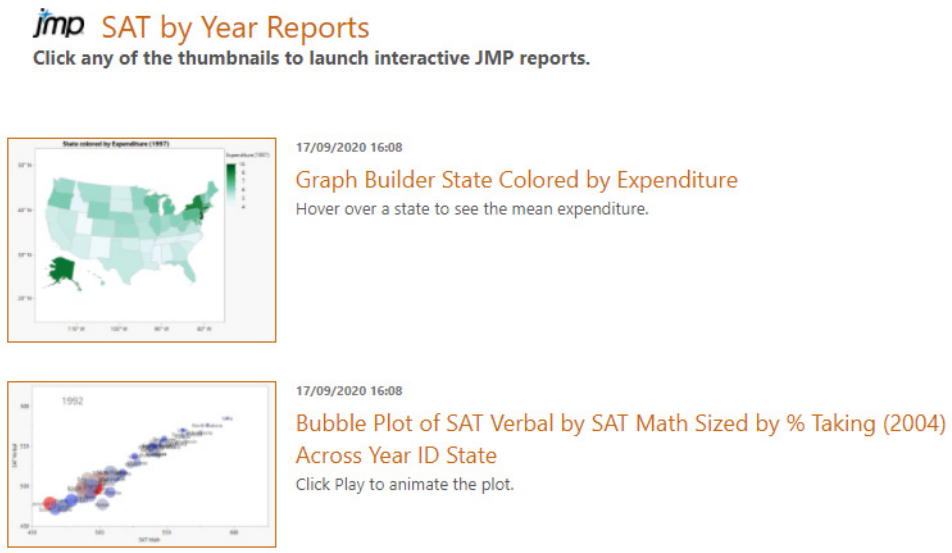
1. In JMP, create the reports.

   Note: If a report contains a Local Data Filter, it is static in the web report and users cannot change the selections. To make the Local Data Filter interactive, deselect the **Include** mode. (In Graph Builder, to see the **Include** mode, select **Show Modes** from the Local Data Filter red triangle menu.)

2. From a report window, select **File** > **Publish** > **Publish to File**.
3. Select the reports that you want to publish.

4. (Optional) Change where the parent folder resides or change the name of the subfolder that will contain the reports.
5. Click **Next**.
6. Enter a title for the index page. You can also update the report titles.
7. (Optional) Change additional options. See "Interactive HTML Report Options".
8. Click **Publish**.

Figure 7.17 Index Page for Multiple Interactive HTML Reports



9. Click a thumbnail to open a report.

For details about working with interactive reports, from an HTML report, click ☰ > **Help**. This opens the help at https://www.jmp.com/interactive.

Interactive HTML Report Options

Title Add a title for the index page (multiple reports only) or the reports.

Description (Optional) Add a description to the index page or the reports. The description will appear under the titles.

Customize (Appears for multiple reports only) Change the appearance of the web page. You can change the style, theme, font, logo, and whether the date or time appears. See "Customize Index Page Options".

Publish Data Select this for interactive reports. If you deselect this option, the reports are static.

Note: To avoid sharing sensitive data, save your results as a non-interactive web page. (Select **File > Export > HTML**.)

Add Image Adds an image to the bottom of the web page.

Open published web report Opens the web page in a browser once you click Publish.

Close reports after running Closes the reports in JMP once you publish the web report.

Delete icon (Appears for multiple reports only) Deletes a report.

Arrow icons (Appears for multiple reports only) Changes the order of reports.

Customize Index Page Options

Style Format Determines the layout of the reports.

Large List Shows the reports in a column with large thumbnail graphics.

Small List Shows the reports in a column with small thumbnail graphics.

Grid Shows the reports in rows.

Custom CSS Enables you to specify a CSS file to format the web page. The CSS file is copied into a subfolder called _css. The link to the CSS file is relative so that you can send the report and support files to another user and maintain the CSS formatting.

Color Theme Specifies the color of the web page, headings, and borders. The default web page has a white background, orange headings, and blue borders.

Change Font Change the font applied in the reports.

Change Logo Specifies an image to display along with the reports. Click the up or down arrow next to the image to move it above or below the reports.

Show date/time Shows or hides the date and time at which the web page was generated.

Save a Report as a PowerPoint Presentation

Create a presentation by saving JMP results as a PowerPoint presentation (.pptx). Rearrange JMP content and edit text in PowerPoint after saving as a .pptx file. Sections of a JMP report are exported into PowerPoint differently.

- Report headings are exported as editable text boxes.

- Graphs are exported as images. Certain graphical elements, such as legends, are exported as separate images. Images resize to fit the slide in PowerPoint.

Use the selection tool to select the sections that you want to save in your presentation. Delete unwanted content once after you open the file in PowerPoint.

Note: On Windows, PowerPoint 2007 is the minimum version required to open .pptx files created in JMP. On macOS, at least PowerPoint 2011 is required.

1. In JMP, create the report.
2. Select **File > Export,** select **Microsoft PowerPoint**, and then click **Next**.
3. Select a graphic file format from the list.

   On Windows, EMF is the default format. On macOS, PDF is the default format.
4. Name and save the file. (On macOS, name the file and click **Export**.

   The file opens in Microsoft PowerPoint because **Open the file after saving** is selected by default.

Note: The native EMF graphics produced on Windows are not supported on macOS. The native PDF graphics produced on macOS are not supported on Windows. For cross-platform compatibility, change the default graphics file format by selecting **File > Preferences > General**. Then, change the **Image Format for PowerPoint** to either PNG or JPEG.

Create Dashboards

A dashboard is a visual tool that lets you run and present reports on a regular basis. You can show reports, data filters, selection filters, data tables, and graphics on a dashboard. The content shown on the dashboard is updated when you open the dashboard.

This section contains the following information:

- "Example of Combining Windows"
- "Example of Creating a Dashboard with Two Reports"

Example of Combining Windows

You can quickly create dashboards by merging several open windows in JMP. Combining windows provides options to view a summary of statistics and include a selection filter.

1. Select **Help > Sample Data Library** and open Birth Death.jmp.
2. Run the Distribution and Bivariate table scripts.

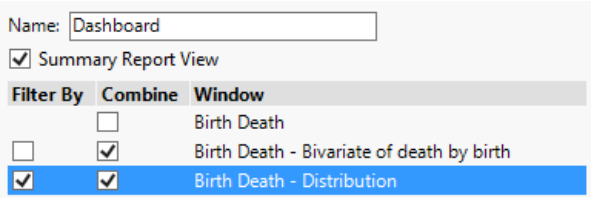
3. From one of the report windows, select **Window > Combine Windows**.

   The Combine Windows window appears.

   Tip: On Windows, you can also select Combine Windows from the Arrange Menu option in the lower right corner of JMP windows.

4. Select **Summary Report View** to display the graphs and omit the statistical reports
5. In the Combine column, select **Birth Death - Bivariate of death by birth** and **Birth Death - Distribution**.
6. In the Filter By column, select **Birth Death - Distribution**.

Figure 7.18 Combine Windows Options



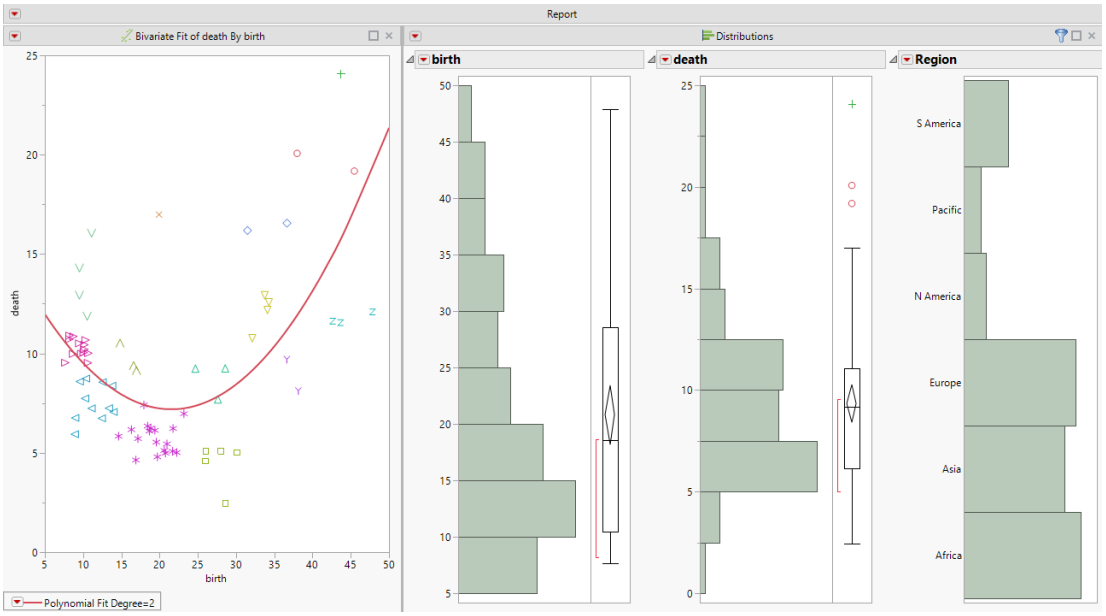
7. Click **OK**.

   The two reports are combined into one window. Notice the filter icon at the top of the Distribution report. When you select a bar in one of the histograms, the corresponding data in the Bivariate graph are selected.

Notes:

- To combine reports on Windows, you can also select Combine Windows from the Arrange Menu option in the lower right corner of JMP windows.
- In the Combine Windows window, select **Summary Report View** to see only the graphs in a report and omit the statistics.

Figure 7.19 Combined Windows



Example of Creating a Dashboard with Two Reports

Suppose that you created two reports and want to run the reports again the next day against an updated set of data. This example shows how to create a dashboard from the reports in Dashboard Builder.

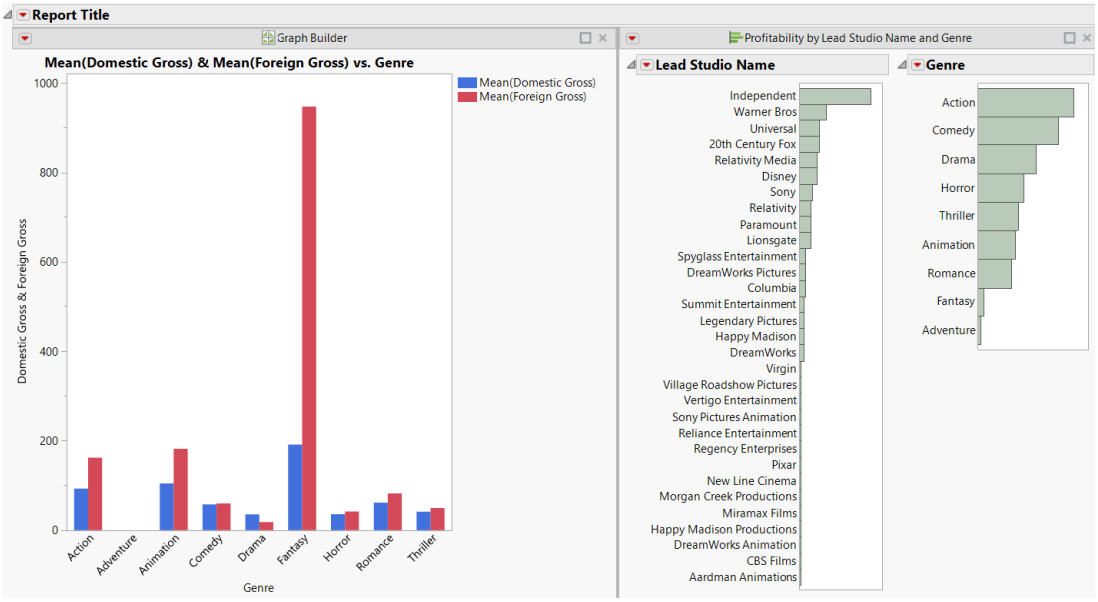
1. Select **Help > Sample Data Library** and open Hollywood Movies.jmp.
2. Run the table scripts named “Distribution: Profitability by Lead Studio and Genre” and “Graph Builder: World and Domestic Gross by Genre”.
3. From any window, select **File > New > Dashboard**.

   Templates for common layouts appear.
4. Select the **2x1 Dashboard** template.

   A box with room for two reports appears on the workspace.
5. In the Reports list, double-click the report thumbnails to put them on the dashboard.
6. Click the Dashboard Builder red triangle and select **Preview Mode**.

   A preview of the dashboard appears. Notice that the graphs are linked to each other and the data table. They also have the same red triangle options as the Distribution and Graph Builder platforms.
7. Click **Close Preview**.

Figure 7.20 Dashboard with Two Reports



For more information about creating dashboards, see *Using JMP*.

Chapter 8

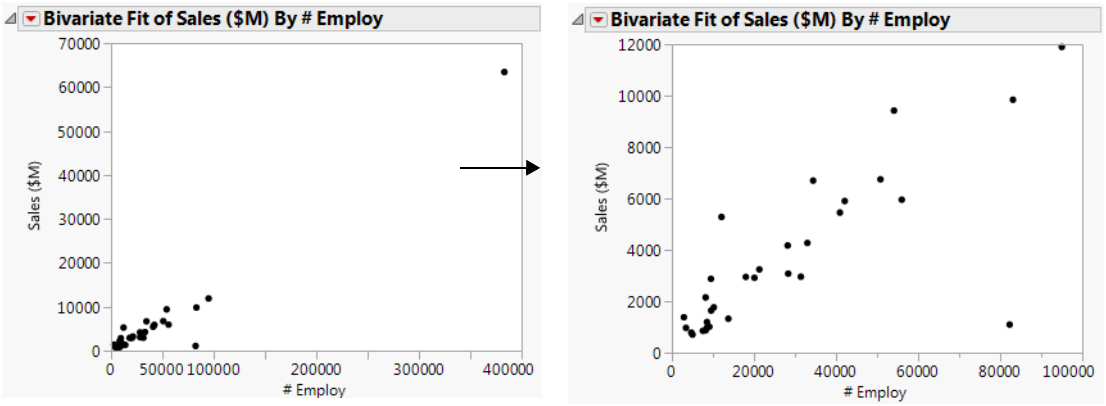
Special Features

Automatic Analysis Updates and SAS Integration

Using some of the special features in JMP, you can do the following:

- Update analyses or graphs automatically
- Customize platform results
- Integrate with SAS to use advanced analytical features

Figure 8.1 Examples of Special Features




```
DATA Candy_Bars; INPUT  Calories Total_fat_g Carbohydrate_g Protein_g; Lines;
310  20  28  6
230  12  27  4
220  12  24  3
170  8  21  3
200  2.5  43  1
260  16  26  5
190  1.5  42  2
190  11  21  2
230  12  28  3
;
RUN;

PROC GLM  DATA=Candy_Bars ALPHA=0.05;
MODEL  Calories =  Total_fat_g Carbohydrate_g Protein_g;
RUN;
```

Contents

Automatically Update Analyses and Graphs 217
Example of Using Automatic Recalc 217
Change Preferences 221
Example of Changing Preferences 222
Integrate JMP and SAS 224
Example of Creating SAS Code 224
Example of Submitting SAS Code 225

Automatically Update Analyses and Graphs

When you make a change to a data table, you can use the Automatic Recalc feature to automatically update analyses and graphs that are associated with the data table. For example, if you exclude, include, or delete values in the data table, that change is instantly reflected in the associated analyses or graphs. Note the following information:

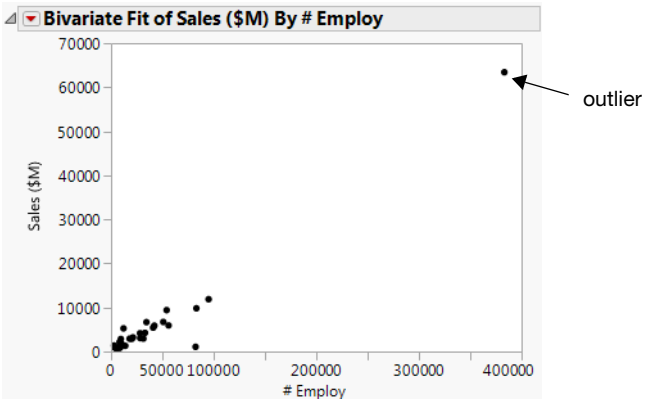
- Some platforms do not support Automatic Recalc. See *Using JMP*.
- For the supported platforms in the **Analyze** menu, Automatic Recalc is turned off by default. However, for the supported platforms in the **Quality and Process** menu, Automatic Recalc is turned on by default, except for the Variability/Attribute Gauge Chart, Capability, and Control Chart.
- For the supported platforms in the **Graph** menu, Automatic Recalc is turned on by default.

Example of Using Automatic Recalc

This example uses the Companies.jmp sample data table, which contains financial data for 32 companies from the pharmaceutical and computer industries.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Fit Y by X**.
3. Select Sales ($M) and click **Y, Response**.
4. Select # Employ and click **X, Factor**.
5. Click **OK**.

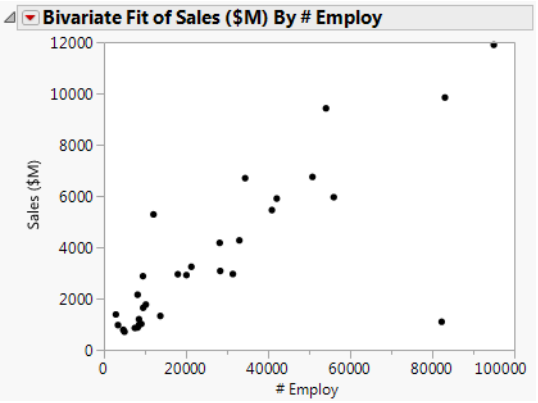
Figure 8.2 Initial Scatterplot



The initial scatterplot shows that one company has significantly more employees and sales than the other companies. You decide that this company is an outlier, and you want to exclude that point. Before you exclude the point, turn on Automatic Recalc so that your scatterplot is updated automatically when you make the change.

6. To turn on Automatic Recalc, click the red triangle next to Bivariate Fit of Sales ($M) By # Employ and select **Redo** > **Automatic Recalc**.
7. Click the outlier to select it.
8. Select **Rows** > **Exclude/Unexclude**. The point is excluded from the analysis and the scatterplot is automatically updated.

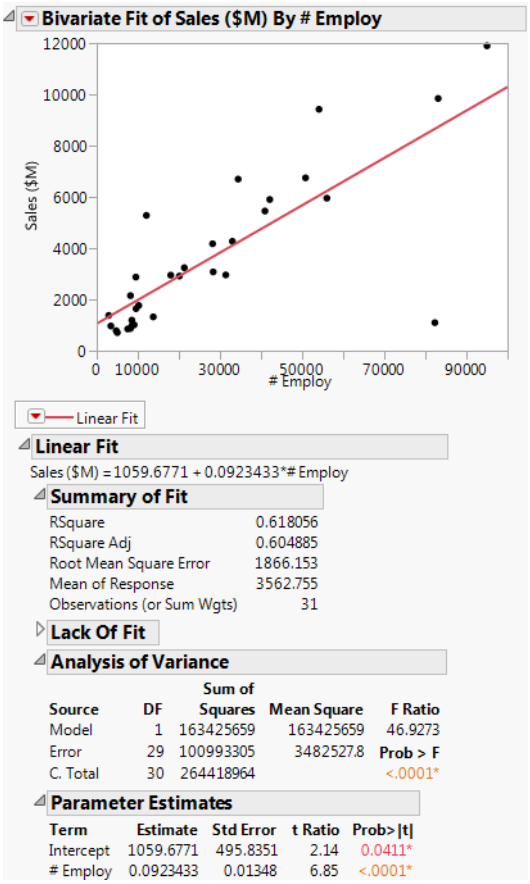
Figure 8.3 Updated Scatterplot



If you fit a regression line to the data, the point in the lower right corner is an outlier, and influences the slope of the line. If you then exclude the outlier with Automatic Recalc turned on, you can see the slope of the line change.

9. To fit a regression line, click the red triangle next to Bivariate Fit of Sales ($M) By # Employ and select **Fit Line**. Figure 8.4 shows the regression line and analysis results added to the report window.

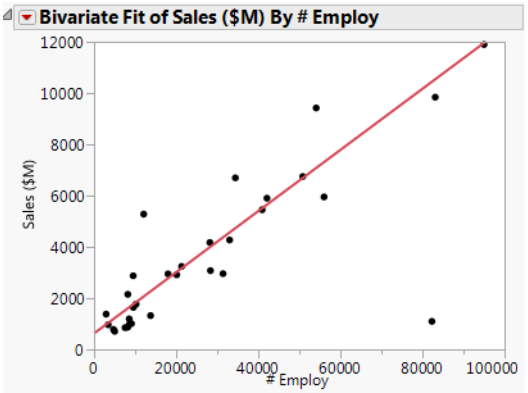
Figure 8.4 Regression Line and Analysis Results



10. Click the outlier to select it.
11. Select **Rows** > **Exclude/Unexclude**. The regression line and analysis results are automatically updated, reflecting the exclusion of the point.

Tip: When you exclude a point, the analyses are recalculated without the data point, but the data point is not hidden in the scatterplot. To also hide the point in the scatterplot, select the point, and then select **Rows** > **Hide and Exclude**.

Figure 8.5 Updated Regression Line and Analysis Results



Linear Fit

Linear Fit

Sales ($M) = 631.36791 + 0.1192722*# Employ

Summary of Fit

RSquare	0.882703
RSquare Adj	0.878514
Root Mean Square Error	1039.891
Mean of Response	3644.95
Observations (or Sum Wgts)	30

Lack Of Fit

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	1	227857403	227857403	210.7112
Error	28	30278440	1081372.9	Prob > F
C. Total	29	258135843		<.0001*

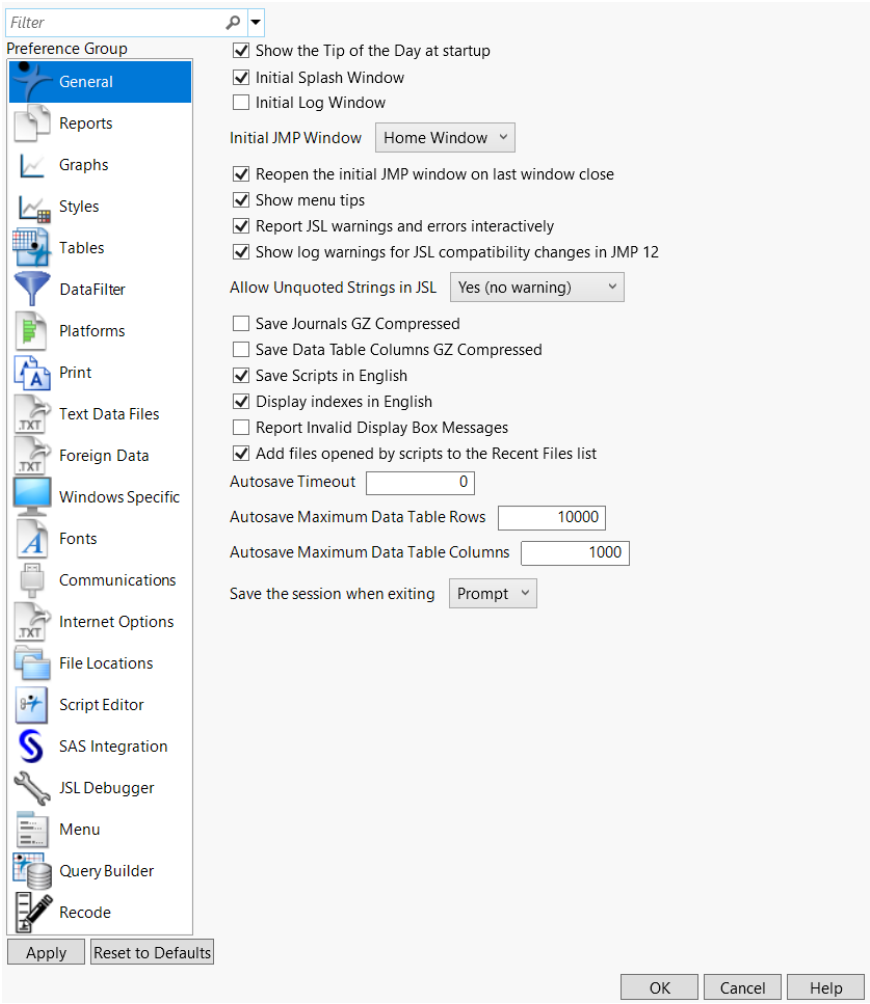
Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	631.36791	281.3288	2.24	0.0329*
# Employ	0.1192722	0.008217	14.52	<.0001*

Change Preferences

You can change preferences in JMP using the Preferences window. To open the Preferences window, select **File** > **Preferences** (Windows) or **JMP > Preferences** (macOS).

Figure 8.6 Preferences Window



On the left side of the Preferences window is a list of Preference groups. On the right side of the window are all of the preferences that you can change for the selected category.

Example of Changing Preferences

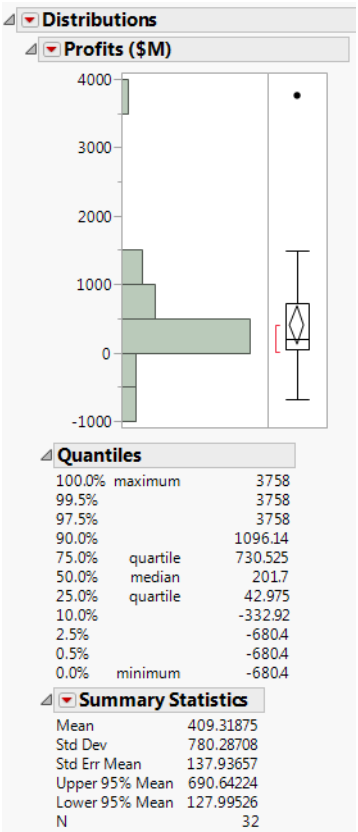
Every platform report window has options that you can turn on or off. However, your changes to these options are not remembered the next time you use the platform. If you want JMP to remember your changes every time you use the platform, change those options in the Preferences window.

This example shows how to set the Distribution platform so that an Outlier Box Plot is not added to the initial report.

Create a Distribution Using the Default Preference Setting

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Distribution**.
3. Select Profits ($M) and click **Y, Columns**.
4. Click **OK**.

Figure 8.7 Distribution Report Window

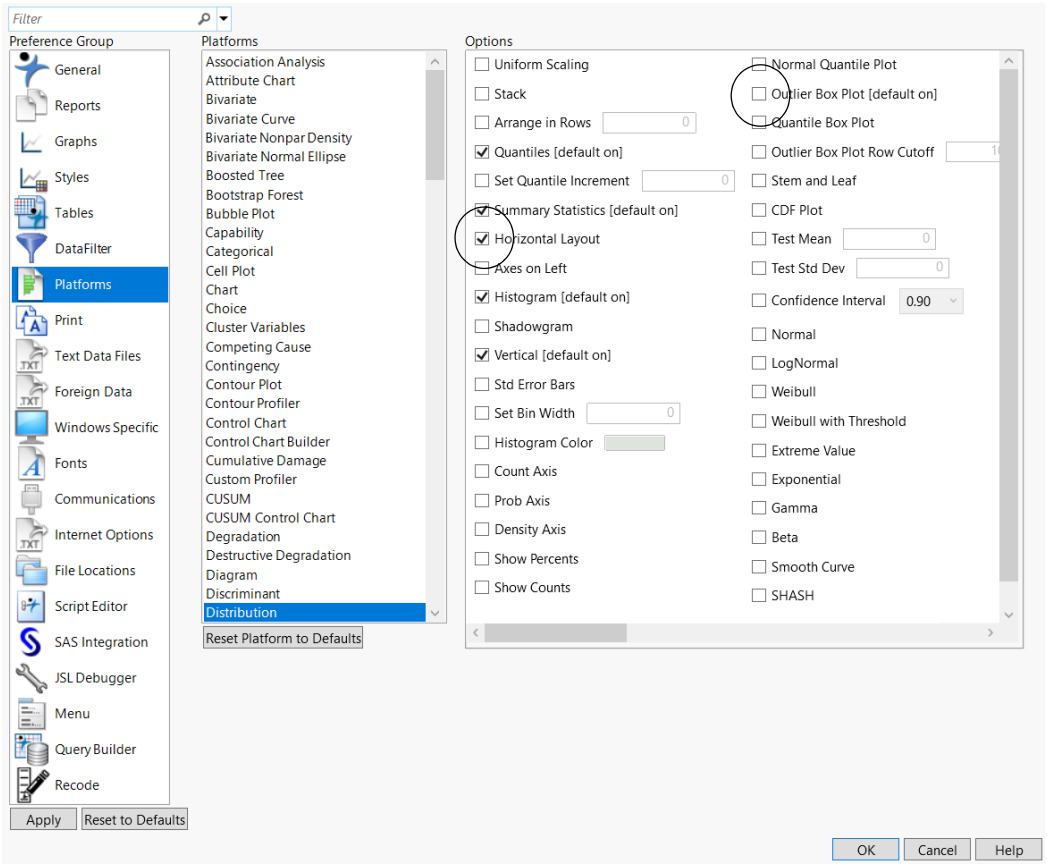


The histogram is vertical, and the graphs includes an outlier box plot. To change the histogram to horizontal and remove the outlier box, select the appropriate options from the red triangle menu for Profits ($M). However, if you want those preferences to be in effect every time you use the platform, then change them in the Preferences window.

Change the Preference for the Outlier Box Plot and Run Distribution Again

1. Select **File** > **Preferences** (Windows) **or JMP > Preferences** (macOS).
2. Select **Platforms** from the preference group.
3. Select **Distribution** from the Platforms list.
4. Select the **Horizontal Layout** option to turn it on.
5. Deselect the **Outlier Box Plot** option to turn it off.

Figure 8.8 Distribution Preferences



6. Click **OK**.

7. Repeat the Distribution analysis. See "Create a Distribution Using the Default Preference Setting".

The histogram is now horizontal and the outlier box plot does not appear. These preferences remain the same until you change them.

For more information about all of the preferences, see *Using JMP*.

Integrate JMP and SAS

Note: You must have access to SAS, either on your local machine or on a server, to use SAS through JMP.

Using JMP, you can interact with SAS in these ways:

- Write or create SAS code in JMP.
- Submit SAS code and view the results in JMP.
- Connect to a SAS Metadata Server or a SAS Server on a remote machine.
- Connect to SAS on your local machine.
- Open and browse SAS data sets.
- Retrieve and view data sets generated by SAS.

For more information about integrating JMP and SAS, see *Using JMP*.

Example of Creating SAS Code

This example uses the Candy Bars.jmp sample data table, which contains nutrition data for candy bars.

1. Select **Help > Sample Data Library** and open Candy Bars.jmp.
2. Select **Analyze** > **Fit Model**.
3. Select Calories and click **Y**.
4. Select Total fat g, Carbohydrate g, and Protein g, and click **Add**.
5. Click the Model Specification red triangle and select **Create SAS Job**.

Figure 8.9 shows the SAS code. (Not all of the data is shown.)

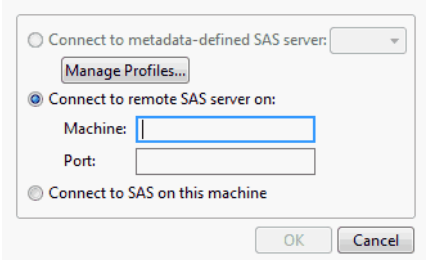
Figure 8.9 SAS Code

```
DATA Candy_Bars; INPUT  Calories Total_fat_g Carbohydrate_g Protein_g; Lines;
310  20  28  6
230  12  27  4
220  12  24  3
170  8  21  3
200  2.5  43  1
260  16  26  5
190  1.5  42  2
190  11  21  2
230  12  28  3
;
RUN;

PROC GLM  DATA=Candy_Bars ALPHA=0.05;
MODEL  Calories =  Total_fat_g Carbohydrate_g Protein_g;
RUN;
```

Example of Submitting SAS Code

1. Select **Help > Sample Data Library** and open Candy Bars.jmp.
2. Select **Analyze** > **Fit Model**.
3. Select Calories and click **Y**.
4. Select Total fat g, Carbohydrate g, and Protein g, and click **Add**.
5. Click the Model Specification red triangle and select **Submit to SAS**.
6. In the **Connect to SAS Server** window (Figure 8.10), choose a method to connect to SAS (if you are not already connected). For this example, select **Connect to SAS on this machine**.

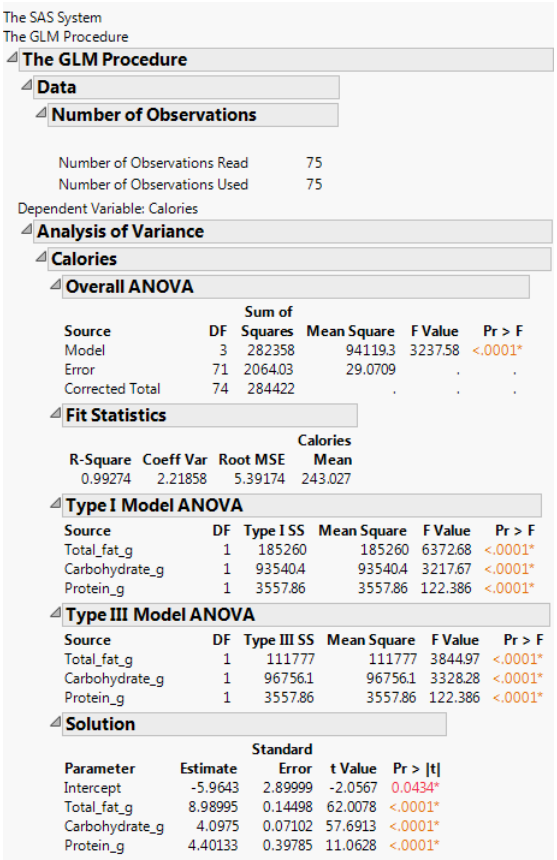
Figure 8.10 Connect to SAS Server



7. Click **OK**.

JMP connects to SAS. SAS runs the model and sends the results back to JMP. The results can appear as SAS output, HTML, RTF, PDF, or JMP report format (you can choose the format using JMP Preferences). Figure 8.11 shows the results formatted as a JMP report. See *Using JMP*.

Figure 8.11 SAS Results Formatted as a JMP Report



The SAS System
The GLM Procedure

The GLM Procedure

Data

Number of Observations

Number of Observations Read	75
Number of Observations Used	75

Dependent Variable: Calories

Analysis of Variance

Calories

Overall ANOVA

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	282358	94119.3	3237.58	<.0001*
Error	71	2064.03	29.0709	.	.
Corrected Total	74	284422	.	.	.

Fit Statistics

R-Square	Coeff Var	Root MSE	Calories Mean
0.99274	2.21858	5.39174	243.027

Type I Model ANOVA

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Total_fat_g	1	185260	185260	6372.68	<.0001*
Carbohydrate_g	1	93540.4	93540.4	3217.67	<.0001*
Protein_g	1	3557.86	3557.86	122.386	<.0001*

Type III Model ANOVA

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Total_fat_g	1	111777	111777	3844.97	<.0001*
Carbohydrate_g	1	96756.1	96756.1	3328.28	<.0001*
Protein_g	1	3557.86	3557.86	122.386	<.0001*

Solution

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	-5.9643	2.89999	-2.0567	0.0434*
Total_fat_g	8.98995	0.14498	62.0078	<.0001*
Carbohydrate_g	4.0975	0.07102	57.6913	<.0001*
Protein_g	4.40133	0.39785	11.0628	<.0001*

Appendix A

Technology Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
- You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS”, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Using JMP

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *Using JMP® 16*. Cary, NC: SAS Institute Inc.

Using JMP® 16

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Using JMP

1 Learn about JMP 19

Documentation and Additional Resources

Formatting Conventions in JMP Documentation 21
JMP Help 22
JMP Documentation Library 22
Additional Resources for Learning JMP 28
JMP Tutorials 28
Sample Data Tables 28
Learn about Statistical and JSL Terms 29
Learn JMP Tips and Tricks 29
JMP Tooltips 29
JMP User Community 30
Free Online Statistical Thinking Course 30
JMP New User Welcome Kit 30
Statistics Knowledge Portal 30
JMP Training 30
JMP Books by Users 31
The JMP Starter Window 31
JMP Technical Support 31

2 Get Started with JMP 33

Introduction to Basic Features

Anatomy of a JMP Session 35
Data Table Features 36
Data Table Panels 37
Data Grid in Data Tables 45
Options to Open Data File 50
Platforms and Reports 51
Launch Windows in Platforms 52
Analysis Results in Reports 55
Manage Files and Open Windows 55
JMP Home Window on Windows 55
JMP Home Window on macOS 59

Search for Recently Opened Files on Windows . . . 61
Close Multiple Files . . . 61
Display and Arrange Open Windows . . . 61
Preview JMP Files . . . 63
Using the Windows Reveal Feature . . . 64
JMP Starter Window . . . 64

3 Import Your Data . . . 67
Create Data Tables

About Importing Data . . . 70
Import Microsoft Excel Files . . . 71
Preview and Import Microsoft Excel Data . . . 71
Import a Microsoft Excel File Directly . . . 83
Import Text Files . . . 84
Find Missing Characters in Imported Data . . . 86
Options in the JMP Text Import Wizard . . . 87
Edit Text Before Importing into JMP . . . 91
Import Text from the Script Editor . . . 93
Import Multiple Files . . . 94
Import PDF Files . . . 99
Concatenate Tables into One Data Table . . . 100
Concatenate Tables by Column Headers . . . 102
Import Tables that JMP Does Not Auto-Detect . . . 102
PDF Preview Red Triangle Options . . . 105
Data Table Preview Options . . . 105
PDF Preview Right-Click Options . . . 106
Import XML and JSON Files . . . 106
Preview and Import XML and JSON Data . . . 106
Import XML Files Directly . . . 114
Import JSON Files Directly . . . 114
Build SQL Queries in Query Builder . . . 115
Connect to a SQL Database . . . 116
Select Tables from a SQL Database . . . 116
Build an SQL Query . . . 121
Save and Run a JMP Query . . . 135
Open the Selected Data . . . 136
Query Builder Red Triangle Options . . . 137
Write SQL Statements in Query Builder . . . 138
Import Data from SAS . . . 138
Java Runtime Environment (JRE) Requirements . . . 138

SAS Menu Options 139
Import SAS Data Sets 139
Create SAS Transport Files in SAS from JMP 142
Connect to SAS from JMP 142
Open SAS Data Sets with SAS Query Builder 148
Open SAS Data Sets through a SAS Server 149
Using SAS Extended Attributes to Import Metadata 157
Run Stored Processes 159
Submit SAS Code 159
Generate ODS Results 161
Import Generated SAS Data Sets 162
Import Data from a Database 164
Connect to a Database 165
Open Data from a Database 166
Write SQL Statements to Query a Database 168
Structured Query Language (SQL): A Reference 170
Import Data from Google Sheets 176
About Importing Google Sheets 177
Google Sheet Settings 177
Import SPSS Files 178
Import Triple-S Survey Data 180
Import HDF5 Data 180
Import Data from MATLAB 181
Import Data from R 181
Import Data Using the Excel Add-In 181
Transfer Excel Data to a Data Table 182
About the JMP Add-In for Excel 183
Uninstall the JMP Excel Add-In 184
Import Data from the Internet or a Remote Computer 184
Create New Data Tables 186

4 Enter and Edit Your Data 189
Perform Basic Data Table Tasks

Enter Data in Data Tables 192
Copy and Paste Data into Data Tables 192
Add Rows to Data Tables 193
Add Columns to Data Tables 193
Fill Data Table Columns with Sequential Data 195
Replace Missing Values with Previous Values in Data Table Columns 196
Enter Cell Formulas into Data Table Columns 197

Select Rows in Data Tables ... 198
Find Matching Cells in Rows ... 200
Select Rows That Contain Specific Values ... 201
Locate the Next Selected Row ... 202
Select Columns in Data Tables ... 203
How to Select Columns ... 203
Select Columns in the JMP Columns Viewer ... 204
Locate Next and Previously Selected Columns in Data Tables ... 210
Resize Rows and Columns ... 210
Organize Data in Data Tables ... 211
Delete Rows in Data Tables ... 211
Delete Columns in Data Tables ... 212
Rearrange Columns in Data Tables ... 212
Group Columns in Data Tables ... 213
Move Values in Data Tables ... 213
Color Cells in Data Tables ... 214
Edit or Delete Cells in Data Tables ... 214
Edit Column Names in Data Tables ... 214
Hide and Exclude Rows in Data Tables ... 215
Exclude Rows in Data Tables ... 216
Hide Rows in Data Tables ... 217
Exclude Columns in Data Tables ... 218
Hide Columns in Data Tables ... 218
View Patterns of Missing Data in Data Tables ... 219
Find and Replace Cell Values in Data Tables ... 221
Edit Cells in Data Tables ... 223
Right-Click Menus for Rows and Columns ... 225
Compare Data Tables ... 225
Assign Characteristics to Rows and Columns ... 232
How to Find Options for Rows and Columns ... 232
Label Rows and Columns ... 232
Assign Colors or Markers to Rows ... 233
Create Color Themes in Graphs ... 235
Delete Custom Color Themes in Graphs ... 241
Select Matching Row State Cells ... 241
Delete Row Characteristics ... 242
Lock Columns ... 242
Restructure Data ... 242
Split a Data Table Column into Multiple Columns ... 243
Create a New Column by Text Matching ... 243

Make Indicator Columns ... 245
Combine Columns in Data Tables ... 246
Compress Selected Columns in Data Tables ... 247
Make a Binning Formula in a Data Table Column ... 248
Convert Labels to Codes and Codes to Labels ... 253
Make a New Formula Column ... 254
Transform Columns in a JMP Platform ... 254
Recode Data in a Column ... 264
Edit Data Tables ... 281
Rename Data Tables ... 281
Lock Data Tables ... 281
Compress Data Tables ... 282
Copy Table Scripts ... 282
Use Data Table Variables ... 283
Create and Save Scripts in Data Tables ... 284

5 Set JMP Column Properties ... 289
The Column Info Window

About the Column Info Window ... 291
About Modeling Types ... 294
The Short-Integer Format ... 294
Numeric Formats ... 295
Store Information in Row State Columns ... 300
Initialize Data in a Column ... 302
Column Properties in JMP ... 304
Basic Column Properties ... 305
Properties That Validate Column Values ... 305
Properties That Attach Information to Column Values ... 307
Properties That Control the Display of Columns ... 312
Properties Used in Modeling and DOE ... 313
Properties Associated with Control Charts and Capability ... 314
Properties That Control How Columns Are Used in Platforms ... 316
SAS Properties ... 324
Link Data Tables with Virtual Join Properties ... 324
Additional Column Properties ... 326
Properties Assigned and Controlled by JMP ... 329
Standardize Attributes and Properties Across Columns ... 330
Recode Column Values ... 331
Standardize Column Attributes ... 331
Standardize Column Properties ... 332

Delete Column Properties ... 332
Example of Standardizing a Formula ... 332
Assign a Preselected Analysis Role to a Column ... 333

6 Reshape Your Data ... 335
Create Subsets, Sort Data, and More

Create a Subset Data Table ... 337
Stratified Subsets ... 339
Create a Subset Data Table from a Report ... 339
Sort Data Tables ... 340
Stack Columns ... 343
Example of Stacking Columns into a Single Column ... 345
Example of Stacking Columns into More Than One Column ... 346
Split Columns ... 348
Examples of Splitting Columns ... 350
Transpose Rows and Columns ... 352
Examples of Transposing Rows and Columns ... 354
Concatenate Data Tables ... 356
Example of Concatenating Data Tables ... 357
Example of Concatenating Data Tables and Table Variables ... 358
Join Data Tables ... 359
Examples of Joining Data Tables ... 362
Query and Join Data Tables with JMP Query Builder ... 371
About Links to Data Tables in Queries ... 374
Virtually Join Data Tables ... 374
Example of Virtually Joining Columns ... 376
Example of Virtually Joining Columns and Showing Row States ... 379
Example of Merging Referenced Data in Virtually Joined Data Tables ... 381
Example of a Link ID and a Link Reference on a Single Column ... 384
Update Data Tables ... 387
Example of Updating a Data Table ... 390
Anonymize Data ... 391

7 Create Formulas in JMP ... 393
Formula Editor

Overview of Formulas ... 395
Build Formulas ... 395
Refer to Data Table Values in Formulas ... 397
Use Local Variables in Formulas ... 400
Incorporate Parameters in Formulas ... 401
Insert Constants in Formulas ... 402

Add Operators to JMP Formulas . . . 404
Keypad Reference for the Formula Editor . . . 404
Use Functions in Formulas . . . 406
Find Frequently Used Functions in the Formula Editor . . . 408
Copy and Replicate Formulas . . . 410
Order Expressions in Formulas . . . 412
Build a Formula in Order of Precedence . . . 413
JMP Formula Editor Options . . . 414
Calculate Derivatives of JMP Functions . . . 414
Simplify Complex Formulas . . . 415
Evaluate Formulas . . . 416
Ignore Errors in Formulas . . . 416
View Formulas in JSL . . . 417
Edit Formulas . . . 417
Correct Mistakes in Formulas . . . 417
Select Expressions in Formulas . . . 418
Delete Functions Formulas . . . 418
Cut, Copy, and Paste Formulas . . . 419
Click and Drag Formulas . . . 419
Customize Formulas in the JMP Formula Editor . . . 420
Hide and Show Outlines Formulas . . . 420
Change the Font Size Formulas . . . 420
Specify the Matrix Size Formulas . . . 420
Change the Orientation of a JMP Formula . . . 421
Open and Close Arguments Formulas . . . 421
Examples of Building Formulas in the JMP Formula Editor . . . 421
Use Basic Features in the JMP Formula Editor . . . 422
Use Local Variables Formulas . . . 423
Use the Munger Function Formulas . . . 424
Use the Match Conditional Function Formulas . . . 425
Use the Delete Expression Button Formulas . . . 426
Keyboard Shortcuts for the JMP Formula Editor . . . 427
Glossary of Terms for the JMP Formula Editor . . . 428

8 Summarize Your Data . . . 431
The Table Summary Command

Summarize Columns . . . 433
Summary Launch Window . . . 433
Create a Summary Table . . . 433
Add a Statistics Column to an Existing Summary Table . . . 437

Explanation of Summary Statistics 438
Example of Creating a Summary Table with Groups 440

9 JMP Reports 443
Navigate and Customize Report Windows

Navigate JMP Reports 446
Use the Hand Tool 447
Display Options for JMP Reports 447
Show and Hide Parts of a JMP Report 450
Combine Several Reports 450
Rename Titles in Reports 450
Increase Font Sizes in Reports 450
Resize a Report Window 450
Red Triangle Options in JMP Reports 451
Local Data Filters in Reports 451
Redo Menus in Reports 454
Save Script Menus in Reports 455
The Data Filter 458
Types of Filter Columns 459
Filtering Modes 459
The Data Filter Control Panel 460
Red Triangle Options for the Data Filter 463
Animation in Data Filters 467
Red Triangle Options for Variables in Data Filters 470
Reformat Tables in Reports 473
Modify Display Box Properties in a JMP Report 474
Turn a Table in a JMP Report Into a Matrix 475
Use Conditional Formatting in Reports 475
Select Points in Graphs 481
Select Rows in Graphs 482
Select a Rectangular Area of Points in a JMP Graph 482
Select an Irregularly Shaped Area of Points in a JMP Graph 483
Use Markers in Graphs 484
Change the Shape of Markers 484
Change the Color of Markers 485
Change the Size of Markers 485
Work with Crowded Markers 485
Add Outlines Around Markers 486
Change the Selection Mode of Markers 487
Specify the Transparency of Markers 488

Exclude and Hide Markers ... 488
Use Images as Markers ... 489
Change the Shape or Color of Markers Based On Values ... 490
Change the Appearance of Graphs ... 492
Resize Graphs ... 492
Zoom In on Graphs ... 494
Change Line Widths in Graphs ... 494
Change the Background Color in Graphs ... 494
Change the Color of Histogram Bars ... 495
Display Coordinates and Temporary Reference Lines in Graphs ... 495
Scroll and Scale Axes in Graphs ... 495
Customize Axes and Axis Labels in Graphs ... 496
Change the Order of Values in Reports ... 507
Change the Pattern and Format of Selected Objects in Graphs ... 508
Add a Graph to a Data Table ... 508
Customize Markers, Lines, Text, and More in Graphs ... 509
Add Images to a JMP Graph or Report ... 515
Paste a Background Image into a JMP Graph ... 515
Paste an Image at the End of a JMP Report ... 516
Drag and Drop an Image into a JMP Graph ... 516
Add Geographical Images and Boundaries to JMP Graphs ... 518
Extract Data from an Image ... 519
Add Annotations and Shapes to JMP Reports and Graphs ... 520
Add Annotations to JMP Reports or Graphs ... 520
Add Shapes to JMP Reports ... 521
Customize Hover Labels in Graphs ... 524
Add Graphs or Images to Hover Labels ... 525
Add Rich Text to Hover Labels in Graphs ... 540
Update Names and Values in Hover Labels ... 543
Edit Hover Labels in Graphs ... 554
Add Images from Data Tables to Hover Labels ... 559
Pin Hover Labels in Graphs ... 560
Right-Click Options for Hover Labels ... 561

10 Save and Share Your Data ... 563
Get Your Data Out of JMP

Save and Share Data Tables ... 566
Save Data Tables as Microsoft Excel Files ... 567
Save JMP Data as a CSV File ... 568
Save JMP Data as a Text File ... 569

Save JMP Data as a JSON File ... 570
Save JMP Data as a SAS Transport File ... 570
Save JMP Data as a Google Sheet ... 572
Save JMP Data as a SAS Data Set ... 573
Save Data Tables to a Database ... 574
Replace a Database with a Data Table ... 575
Save and Share Reports ... 576
Share Reports as Interactive HTML ... 578
Publish Reports to JMP Live or JMP Public ... 582
Save Reports as PDF Files ... 591
Save Reports as PowerPoint Presentations ... 592
Setting the Graphic DPI for Exported Graphics ... 594
Save a JMP Analysis as a Script ... 594
Save Parts of a Report in a Graphic Format ... 594
Print Reports ... 596
Copy and Paste Reports ... 596
Work with Projects ... 598
Create a New Project ... 599
Open Files in a Project ... 599
Rearrange Files in Projects ... 600
Save a Project ... 601
Project Workspace ... 602
Project Bookmarks ... 603
Project Contents ... 603
Recent Files in Projects ... 604
Project Log ... 605
Move Files into Projects ... 605
Write a Project On Open Script ... 606
Example of Creating a New Project ... 606
Create Journals ... 609
Example of Making a Journal for a Presentation ... 610
Create New Journals ... 612
Manage Links in Shared Journals ... 613
Add and Edit Outlines to Journals ... 614
Right-Click Options in Journals ... 615
Control the Display of Outline Levels in Journals ... 616
Add a Graph or Graphic to Journals ... 616
Customize Items in Journals ... 616
Prevent Modifications to Journals ... 617
Append Reports to Journals ... 617

Save Journals in Other Formats ... 617
Delete Items from Journals ... 618
Save JMP Sessions ... 618
Save Sessions upon Exiting JMP ... 619
Save Sessions Manually ... 619
Save a Log Window ... 620

11 Combine and Present Reports Using Dashboards ... 621
Extend JMP

Combine JMP Reports by Creating a Dashboard ... 623
Example of Creating a Dashboard with Two JMP Reports ... 623
Using the Dashboard Builder ... 624
Red Triangle Options in the Dashboard Builder ... 625
Start with a Sample Dashboard ... 626
Edit a Dashboard ... 627
View a Running Dashboard ... 630
Example of Combining Windows to Create a Dashboard ... 633
Example of Adding Selection Filters to a Dashboard ... 635
Example of Creating a JMP Query Dashboard and Add-In ... 637
Example of Creating a Dashboard from Two Data Tables ... 639

12 Customize Menus and Toolbars ... 643
Personalize JMP

Customize JMP Toolbars and Menus on Windows ... 645
Customize Toolbars ... 646
Customize the Main Menu Bar ... 649
Create Menu Items and Toolbar Buttons ... 650
Set User Permissions for Customized Toolbars and Buttons ... 654
Rearrange Toolbars ... 655
Copy and Paste Menus, Menu Items, Toolbars, and Buttons ... 655
Rearrange Custom Menus, Menu Items, and Buttons ... 657
Delete Custom Items ... 658
Show or Hide Custom Items ... 659
Import Customizations ... 660
Remove Customizations from JMP ... 660
Customize Toolbars on macOS ... 661
Customize Menus on macOS ... 663
Customize Menus and Toolbars for Add-Ins ... 664

13 JMP Preferences 665
The Preferences Window
Overview of JMP Preferences 667
General JMP Preferences 667
Preferences for JMP Projects 671
Preferences for JMP Reports 672
Preferences for JMP Graphs 675
Style Preferences for JMP Graphs and Reports 679
Preferences for Data Tables 682
Preferences for JMP Data Filters 687
Preferences by JMP Platform 688
Print Preferences 690
Preferences for Importing and Exporting Text Files 691
Preferences for Third-Party Data 696
Windows Preferences 698
Mac OS Preferences 702
Fonts Preferences 704
Communications Preferences 707
Preferences for Internet Options 707
Preferences for File Locations 708
Preferences for the JSL Script Editor 710
Preferences for SAS Integration 713
Preferences for Updating JMP 716
Preferences for the JSL Debugger 718
Menu Preferences 719
Preferences for the Query Builder 720
Preferences for Recoding Columns 721
Preferences for the Column Switcher 723
Preferences for the Log 725

A Reference for JMP Functions in Formulas 727
Descriptions of Functions in the Formula Editor
Row Functions 729
Numeric Functions 731
Transcendental Functions 732
Trigonometric Functions 736
Character Functions 737
Comparison Functions 743
Conditional Functions 744
Probability Functions 749

Discrete Probability Functions 760
Statistical Functions 762
Random Functions 766
Date Time Functions 773
Row State Functions 777
Assignment Functions 783
Parametric Model Functions 783
Finance Functions 784

B JMP Technical Details 787

JMP Features That Support Multithreading 789
Conventions for Mapping JMP Attributes to SAS Extended Attributes 790

C References 801

D Technology License Notices 803

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation 21
JMP Help 22
JMP Documentation Library 22
Additional Resources for Learning JMP 28
JMP Tutorials 28
Sample Data Tables 28
Learn about Statistical and JSL Terms 29
Learn JMP Tips and Tricks 29
JMP Tooltips 29
JMP User Community 30
Free Online Statistical Thinking Course 30
JMP New User Welcome Kit 30
Statistics Knowledge Portal 30
JMP Training 30
JMP Books by Users 31
The JMP Starter Window 31
JMP Technical Support 31

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

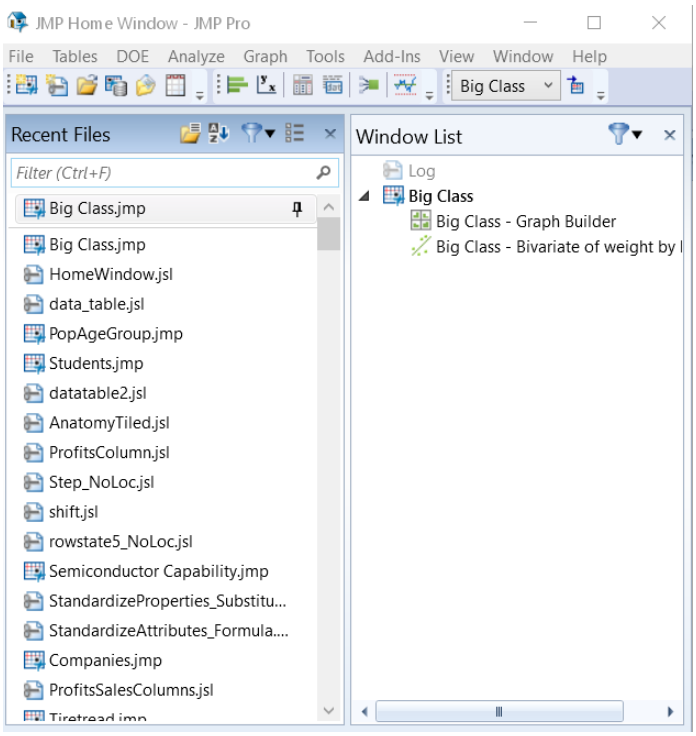
Get Started with JMP

Introduction to Basic Features

To get you started with JMP, this chapter covers the following topics:

- learn about the initial windows that appear when you start JMP
- understand data tables
- open data files
- manage open windows
- learn about the anatomy of a typical JMP user session

Figure 2.1 The JMP Home Window on Windows



Contents

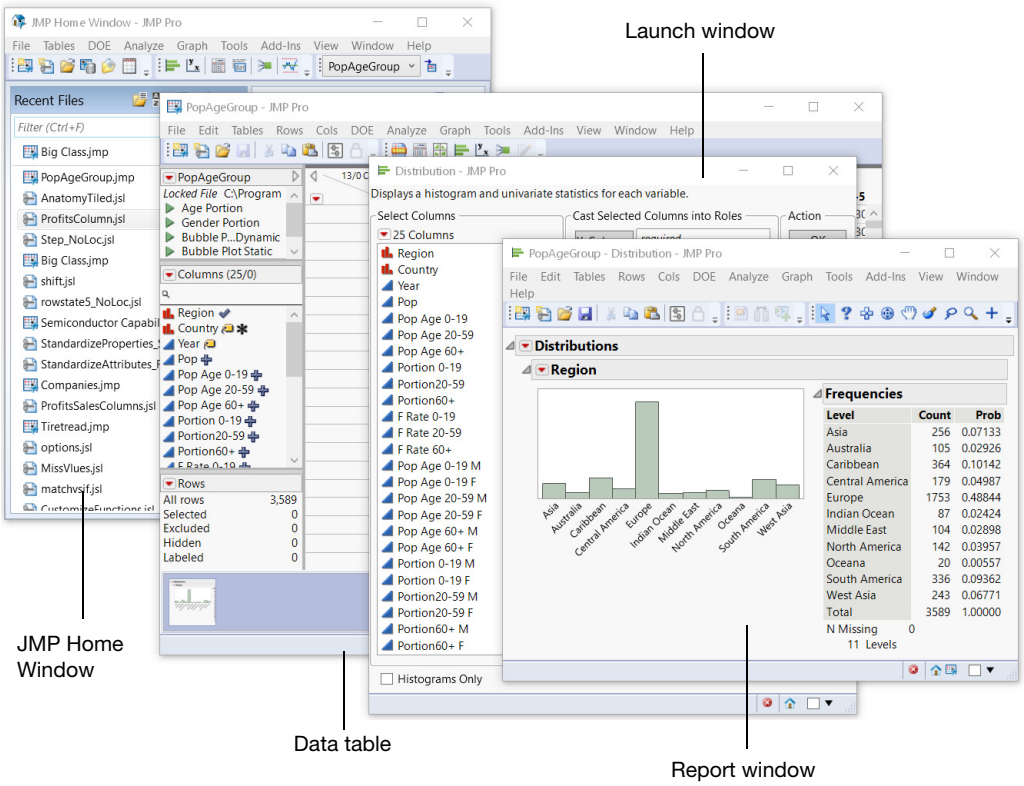
Anatomy of a JMP Session 35
Data Table Features 36
Data Table Panels 37
Data Grid in Data Tables 45
Options to Open Data File 50
Platforms and Reports 51
Launch Windows in Platforms 52
Analysis Results in Reports 55
Manage Files and Open Windows 55
JMP Home Window on Windows 55
JMP Home Window on macOS 59
Search for Recently Opened Files on Windows 61
Close Multiple Files 61
Display and Arrange Open Windows 61
Preview JMP Files 63
Using the Windows Reveal Feature 64
JMP Starter Window 64

Anatomy of a JMP Session

JMP consists of platforms that are organized by the type of statistical analysis. For example, the Distribution platform produces a univariate analysis using histograms, additional graphs, and reports. You might analyze data in the Distribution platform and then choose another platform to examine the data more thoroughly. As a result, several windows are open at once: at a minimum, the data table, platform launch windows, and the results of the analyses.

Figure 2.2 shows a typical JMP session. Note that the windows are not maximized so that you can quickly switch from one window to another.

Figure 2.2 Anatomy of a Typical JMP Session on Windows



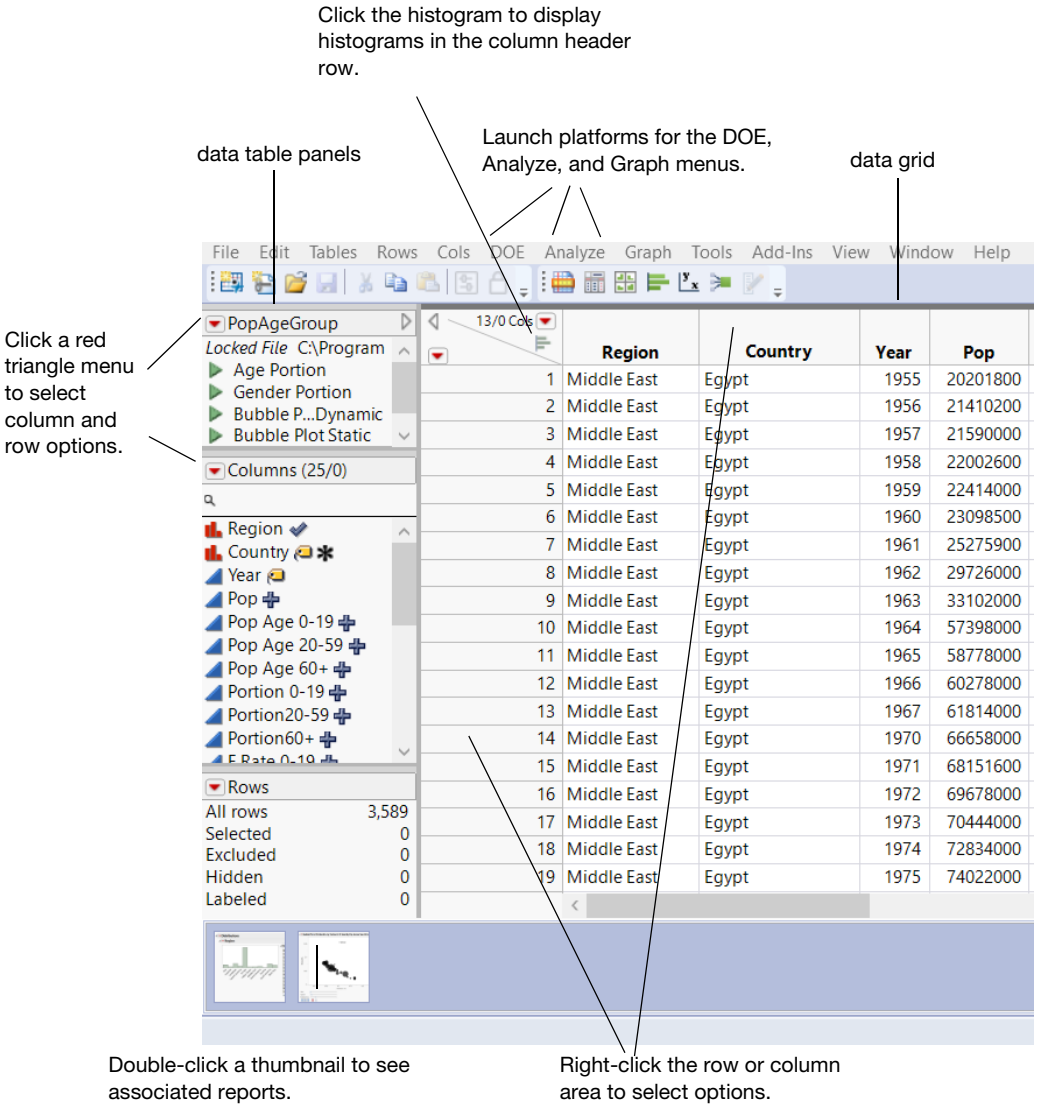
The following sections describe the windows that you typically work with.

Data Table Features

In JMP, data points are organized into rows and columns referred to as the *data table*. A data table has two parts: the *data table panels* on the left and the *data grid* on the right.

You can enter, view, edit, and manipulate data using data tables. In a data table, each variable is a column, and each observation is a row.

Figure 2.3 Data Table Features



The data table has the following characteristics:

- Column names can contain any keyboard character, including spaces, and can be up to 255 characters long.
- The maximum length of the data table's name depends on your computer's operating system.
- Consider setting the *Autosave timeout value* in the General preferences to automatically save data tables at the specified number of minutes. This autosave value also applies to journals, scripts, projects, and reports.
- Change the default size and font for names and values selecting **File > Preferences > Fonts**. (On macOS, select **JMP > Preferences > Fonts**.)
- Column names automatically wrap in the column name area to accommodate the column width that you specify.
- Move column boundaries and enlarge the column to view long values. Adjust widths of all selected columns at once by pressing Alt as you drag the double arrow cursor on any of the selected column boundaries.
- The number of rows and columns in a data table is limited only by your computer's memory.
- Resize rows by dragging one of the row borders. All rows are resized to the same height. Graphics that display inside each cell shrink based on the row height.

  To resize columns to the same width, select the columns and press Alt (Windows) or Option (macOS); drag one of the column borders. All columns are resized to the same width.

Tip: To close a report thumbnail in the data table on Windows, press the middle-click button on your mouse.

Data Table Panels

Data tables contain three panels:

- Table panel
- Columns panel
- Rows panel

These panels are located on the left of the data grid. They contain information about the table and its contents. Each panel has interactive areas.

Figure 2.4 Interacting with the Data Table Panels

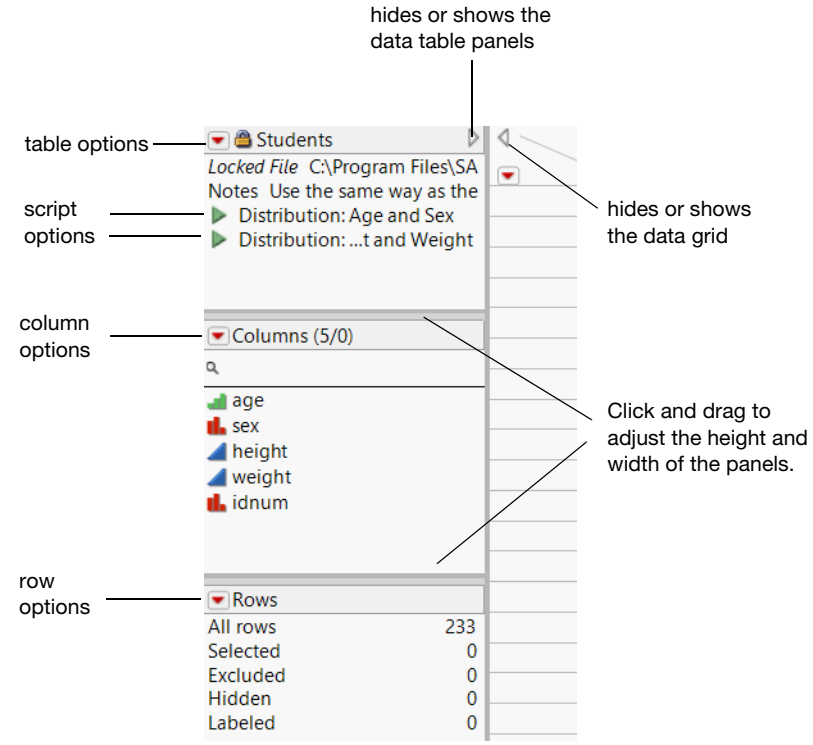


Table Panel in Data Tables

The Table panel contains the following elements:

- Name of the data table
- Icons indicating the table state
- Red triangle menus containing table options
- (Optional) Table variables
- (Optional) Green triangles for table scripts

Figure 2.5 Example of a Table Panel

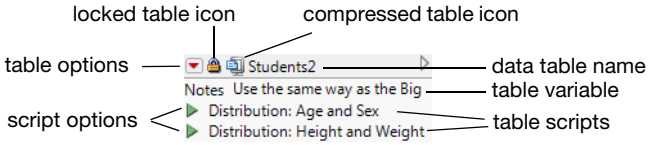


Table Options

Clicking on the red triangle menu next to the data table name shows these options:

Tables Contains the same options as the Tables menu. See "Reshape Your Data".

New Table Variable Creates a new table variable, which can be text or any other constant character value that you always want to be available in the data table. Table variables are normally used to document tables. See "Use Data Table Variables".

Note: To rename a table variable, double-click it and enter a new name in the Name field.

New Script Creates a JSL script to save with the data table. After selecting this command, name the script and type in the value (the JSL commands). After you click **OK**, the new script is listed in the Table panel and you can click its green triangle to run, edit, or delete it. See "Create and Save Scripts in Data Tables".

Group Scripts Moves the selected scripts into a group. Click the group name to rename it. Helpful when you want to organize scripts or minimize a long list of scripts.

Ungroup Scripts Ungroups scripts in the selected group to return them to the top level.

Suppress Formula Eval Turns off the feature that automatically evaluates formulas. You can turn off evaluation and build sections of a formula. To then test the formula, deselect Suppress Formula Eval.

Lock Data Table Locks the data table so that data and column properties cannot be edited or added. You can still run analyses and assign characteristics. See "Lock Data Tables".

Compress file when saved Compresses the data table when it is saved. After the data table is saved, a compressed file icon appears next to the data table name in the table panel. See "Compress Data Tables".

The **Compress file when saved** option only decreases the file size. This command does not affect the memory required to analyze the data. To reduce both the file size and memory required for analyzing, use **Cols > Utilities > Compress Selected Columns**. See "Compress Selected Columns in Data Tables".

Tip: You can also configure JMP to always use GZ compression when saving tables by selecting **Preferences > General > Save Data Table Columns GZ Compressed**.

Disable Undo Removes all actions from the undo history and does not record future actions. Undos are disabled only while the data table is open; the setting is not saved with the data table. This option saves memory, especially when you delete many rows or perform other tasks on the data table that require a large amount of memory to record the data.

Copy Table Script Copies the script that re-creates the table. To re-create the table, put the copied script in a new script and run it. Note that referenced columns in virtually joined tables are not included in the script.

Copy Table Script (No Data) Copies the script that re-creates the table but omits the data.

Rerun Formulas Re-evaluates all columns containing formulas within the data table.

Script Options

To run a script from the data table panel, click the green triangle next to the script name. Right-clicking the script name or green triangle shows these options:

Run Script Runs the script.

Tip: Run multiple table scripts at once by pressing Ctrl while selecting the table scripts you want to run. Then, right-click inside the empty area under the list of table scripts, and select **Run Script**.

Edit Opens most scripts in the script editor so that you can edit it. Opens a JMP application script in Application Builder.

Edit with Recode Opens a recode script (which you created by recoding a column and saving the script) in the script editor.

Delete Deletes the script.

Group Scripts Moves the selected scripts into a group. Click the group name to rename it. Helpful when you want to organize scripts or minimize a long list of scripts.

Ungroup Scripts Ungroups scripts in the selected group to return them to the top level.

Copy Copies the script. You can then paste it into the Table panel of another data table.

Debug Script Opens the script in the JSL Debugger. See the *Scripting Guide*.

Paste Pastes the script from another data table.

Additional Options

In the Table panel, you can also perform the following tasks:

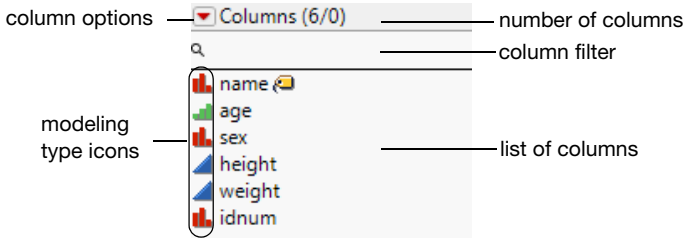
- Double-click a table variable or script name to edit the name and content.
- Drag a table variable or script to rearrange it.

Columns Panel in Data Tables

The Columns panel provides a way to view and move columns. The panel contains the following information:

- Column options (contains some of the same options as the Cols menu)
- Total number of columns and number of columns selected in the data table
- A filter box to help you quickly find columns (click the magnifying glass icon to set search filter options and parameters)
- A list of columns found in the data table
- Icons indicating each column's modeling type (see "About Modeling Types")
- Icons representing characteristics and properties assigned to the columns (not shown, see Figure 2.7)

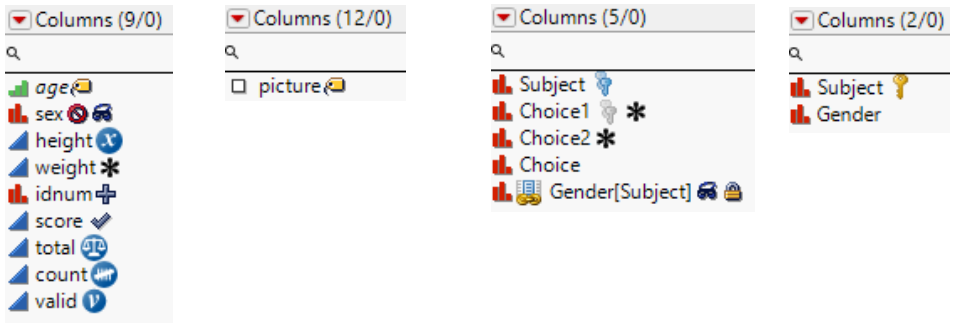
Figure 2.6 Example of a Columns Panel



Icons Representing Column Characteristics and Properties

Icons to the right of each column name indicate characteristics and properties the columns contain.

Figure 2.7 Icons Indicating Column Characteristics and Properties



Note: Italics indicate that the column is locked into place. When you scroll horizontally, the column remains visible.

Here are the icons that can appear in the Columns panel:

Indicates that points on plots corresponding to the column are labeled by the value instead of the row number. See "Label Rows and Columns".

Indicates that the column is excluded from the calculations. See "Exclude Rows in Data Tables".

Indicates that the column is not included in graphs. See "Hide Rows in Data Tables".

Can be X or Y. Indicates that the column has been assigned the preselected role of *x* or *y*. See "Assign a Preselected Analysis Role to a Column".

Indicates that the column has been assigned the preselected role of *validation*. See "Assign a Preselected Analysis Role to a Column".

Indicates that the column contains one or more properties. Click to reveal a list of properties the column contains.

Indicates that the values in the column result from a formula. When formula evaluation is suppressed, the icon appears gray. Double-click to view and edit the formula. See "JMP Formula Editor Options".

Indicates that the range check or the list check option is turned on. Click to view and edit the range or list. See "Range Check" and "List Check".

Indicates that the column has been assigned the preselected role of *weight*. See "Assign a Preselected Analysis Role to a Column".

Indicates that the column has been assigned the preselected role of *frequency*. See "Assign a Preselected Analysis Role to a Column".

Indicates that the column values cannot be edited. See "About the Column Info Window".

Indicates that the expression column contains images that can be used as markers. See "Use Images as Markers".

Virtual Join Icons

Indicates that the column has a Link Reference column property. This property links a column in the current data table to the ID column in the referenced data table. Blue indicates that the referenced data table is linked.

Indicates that the column has a Link Reference column property. Gray indicates that the referenced data table is not open or not linked properly.

Indicates that the column has a Link ID column property, which marks a column in the data table as the ID column. That is, the rows of the data table are uniquely identified by the values of the ID column.

Indicates that the column is linked from the referenced data table.

For more information about virtual join, see "Virtually Join Data Tables".

Search Filter Options

To show search filter options in the columns list, select Show Filter from the columns list's red triangle menu. This enables you to customize your search for columns, which is particularly helpful with long lists of columns.

Click the down arrow button next to the search box to refine your search.

Contains Terms Returns items that contain a part of the search criteria. A search for "ease oom" returns messages such as "Release Zoom".

Contains Phrase Returns items that contain the exact search criteria. A search for "text box" returns entries that contain "text" followed directly by "box" (for example, "Context Box" and "Text Box").

Starts With Phrase Returns items that start with the search criteria.

Ends With Phrase Returns items that end with the search criteria.

Whole Phrase Returns items that consist of the entire string. A search for "text box" returns entries that contain only "text box".

Regular Expression Enables you to use the wildcard (*) and period (.) in the search box. Searching for "get.*name" looks for items that contain "get" followed by one or more words. It returns "Get Color Theme Names", "Get Name Info", and "Get Effect Names", and so on.

Invert Result Returns items that do not match the search criteria.

Match All Terms Returns items that contain both strings. A search for "t test" returns elements that contain either or both of the search strings: "Pat Test", "Shortest Edit Script" and "Paired t test".

Ignore Case Ignores the case in the search criteria.

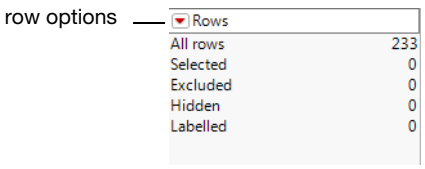
Match Whole Words Returns items that contain each word in the string based on the Match All Terms setting. If you search for "data filter", and Match All Terms is selected, entries that contain both "data" and "filter" are returned.

Rows Panel in Data Tables

The Rows panel contains the following information:

- Row options (same options as the **Rows** menu)
- Total number of rows
- Number of selected (highlighted), excluded, hidden, and labeled rows

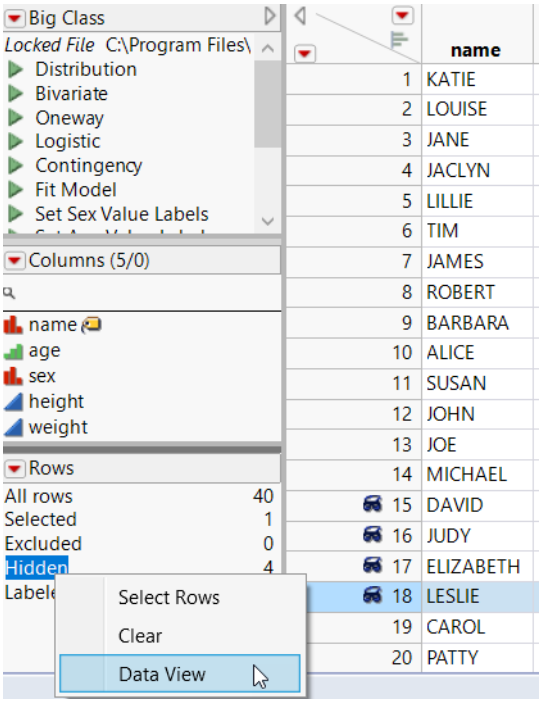
Figure 2.8 Example of a Rows Panel



Right-click the categories in the Rows panel to select rows, clear the selection, or to create a *data view*.

A data view creates a linked subset of the main data table. For example, if several rows are marked hidden, you might want to open a window that shows you only the hidden rows. Right-click **Hidden** in the Rows panel and select **Data View**.

Figure 2.9 Creating a Data View from the Rows Panel

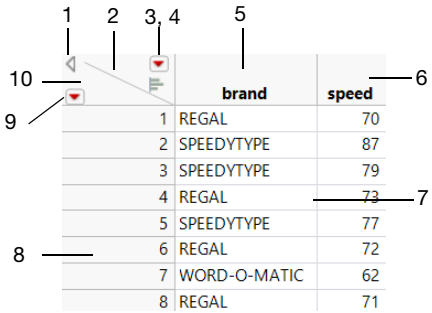


When using a data view, continue to do most of your editing in the main data table. When you make changes in either the main data table or in the data view, the changes are reflected in both. You can make minor changes (such as changing some data or adding a column) in the data view. However, if you want to make major changes (like adding a formula) you must make those changes in the main data table.

Data Grid in Data Tables

The data grid is the main part of the data table that contains your data. Figure 2.10 illustrates how to interact with the data grid. See also "Select Rows in Data Tables".

Figure 2.10 Interacting with the Data Grid



No.	Description
1	Hides or shows the data table panels.
2	Click to deselect any selected columns. To select all columns, press Shift and click. Double-click to add multiple columns.
3	The red triangle shows the columns menu.
4	The histogram shows histograms in the column header row.
5	Click to select the column. Double-click to view the Column Info window. See "About the Column Info Window". Right-click for column options. See "Right-Click Options for Columns".
6	Double-click the column name to edit it in the Column Info window, or select the column name and press Enter.
7	Click and drag to change a column width. To change several column widths, select the columns, press Alt, then click and drag.
8	Click to select the row. Double-click to open the Row Editor. See "Edit Cells in Data Tables". Right-click for row options. See "Right-Click Options for Rows".
9	Shows the rows menu.
10	Click to deselect all selected rows. To select all rows, press Shift and click. Double-click to add rows.

Right-Click Options for Columns

Right-clicking in a column heading shows these options:

Column Info Opens the Column Info window. See "About the Column Info Window".

Standardize Attributes Copies attributes (data types, modeling types, numeric formats, and so on) or properties (formulas, notes, list and range checks, and so on) from one column to other columns. Also deletes selected attributes or properties from columns. See "Standardize Attributes and Properties Across Columns".

Column Properties Contains a list of column properties. Select one to open the Column Info window and apply it to the column. This list is also available from the Column Info window. See "Column Properties in JMP".

Formula Opens the Formula Editor. See "Create Formulas in JMP".

Recode Enables you to change all of the values in a column at once. See "Recode Data in a Column".

New Formula Column Creates a formula column. See "Make a New Formula Column".

Insert Columns Inserts one or more columns before the selected column or columns.

Delete Columns Deletes all selected columns.

Label/Unlabel Labels or unlabels selected columns in all plots. See "Label Rows and Columns".

Link ID Used in virtually joined data tables to mark a column in the auxiliary data table as the ID column. That is, the rows of the data table are uniquely identified by the values of the ID column. The Link ID column property checkbox is selected if the column is the ID column for the data table. See "Link Data Tables with Virtual Join Properties".

Link Reference Used in virtually joined data tables to map a column in the main data table to the ID column in the auxiliary data table. The Link Reference column property specifies the path name of the auxiliary data table. See "Link Data Tables with Virtual Join Properties".

Sort Sorts all of the rows in the table by the values in the selected column. You can choose to sort the rows in ascending or descending order. See "Sort Data Tables".

Copy Column Properties Copies the column properties.

Paste Column Properties Pastes the column properties.

Copy Columns Copies the script to recreate the columns. It copies all of the attributes and properties but no data.

Paste Columns Creates new columns from the script generated by Copy Columns. The column will be placed after the selected column. If no column is selected, the new columns are placed at the end of the data table. The new columns populate (in the destination table) as many rows as there are data in the script.

Right-Click Options for Rows

Right-clicking in a row heading shows these options:

Hide and Exclude Hides the selected rows in all plots and graphs and excludes them from analyses. See "Hide and Exclude Rows in Data Tables".

Exclude/Unexclude Excludes or includes selected rows from analyses. See "Exclude Rows in Data Tables".

Hide/Unhide Hides or shows selected rows in all plots and graphs. See "Hide Rows in Data Tables".

Label/Unlabel Labels or unlabels selected rows in all plots. See "Label Rows and Columns".

Colors Provides a color palette. Select a color to apply it to the selected rows. The color is used in plots and graphs. See "Assign a Color to Rows".

Markers Provides a palette of markers or symbols. Select a marker to apply it to the selected rows. The marker is used in plots and graphs instead of points. See "Add Markers to Rows".

Color Rows by Row State Colors the row the same as the current row state color. See "Assign Colors or Markers to Rows Based on Column Values".

Select Matching Cells Selects rows in the active data table with values that match the selected row(s). See "Resize Rows and Columns"

Invert Selection Selects all previously deselected rows, and deselects all currently selected rows.

Clear Row States Clears all active row states in the data table. All rows become included, visible, unlabeled, and show in plots as black dots. It does not affect row states saved in row state columns. See "Delete Row Characteristics".

Add Rows Adds the specified number of rows to the data table. See "Add Rows to Data Tables".

Delete Rows Removes all selected rows from the data table. Use the **Undo** command on the **Edit** menu to undo an accidental deletion. See "Delete Rows in Data Tables".

Cursor Forms

The cursor takes different forms, depending on its location in the data grid.

Arrow Cursor The standard arrow cursor appears in the following locations:

- In the panels area to the left of the data table
- In the triangular rows and columns area, located in the upper left corner of the data grid

You can perform the following actions with the arrow cursor:

- To select a column using the arrow cursor, click its name in the Columns panel.
- Double-click a column name in the Columns panel to edit it.
- Click the triangular areas in the upper left corner of the data grid to deselect rows and columns.

Selection (Large Plus) Cursor When the cursor is at the top of a column heading, or in a row number area, it becomes a large plus, indicating that you can select rows or columns. When you click, that row or column is selected and highlighted. Click and drag to select multiple rows or columns, and press Ctrl and click to select discontiguous rows or columns.

- Double-click a column heading area to see the Column Info window for that column.
- Select a column to change the column name. The column highlights. Begin typing (if it is not in a locked column or locked data table).
- Double-click the row number area to edit the rows using the Row Editor.

I-beam Cursor When you select editable text, the cursor becomes a standard I-beam. To edit text, position the I-beam within highlighted text. Click to mark an insertion point, or drag to select text for replacement.

Double Arrow Cursor The cursor changes to a double arrow when it is on a column or a panel boundary. Drag this cursor left or right to change the width of a column or panel. Changing the width of a column does not affect the column field width specified in the Column Info window (accessed by double-clicking a column name).

Tip: You can adjust widths of all selected columns at once by pressing Alt as you drag the double arrow cursor on any of the selected column boundaries.

List Check Cursor The cursor changes form when you move the mouse over values in columns that have data validation in effect. It becomes a small, downward-pointing arrow on a column with list checking. When you click, the value is highlighted and the cursor becomes the I-beam. Enter or edit data as usual with any values defined as valid text or valid numbers. See "List Check".

Pointer Cursor The cursor changes to a pointer over these objects:

- Red triangle menus for options
- Triangular disclosure icons that open or close panels
- Modeling type icons for changing

Options to Open Data File

Note: For more information about opening files, see "Import Your Data".

To open a data set, select **File > Open** and select the file type. Some file types have additional features and options that appear in the Open Data File window (Table 2.1).

Tips:

- Windows only: To open the same file type every time, select the **Select this filter the next time this window is invoked** check box.
- Open a file by dragging it onto the JMP Home Window.
- To change which directory the **File > Open** command looks in, see "Preferences for File Locations".

Table 2.1 File > Open Options by File Type

File Type	Additional Features and Options
JMP Data Tables	Table notes, the number of columns (Cols), and the number of rows appear. Use the **Select Columns** option to select which columns are imported into the data table. On Windows, click the arrow next to **Open** and then use the Select Columns option to specify which columns are imported into the data table. On macOS, you can select which columns to import after you click **Open**.
Excel Files	• Import the file in the Excel Import Wizard by default to customize the layout and preview the data before you import it. • Click the arrow next to **Open** and then select one of the options. Convert the first spreadsheet row into column headings.

Table 2.1 File > Open Options by File Type *(Continued)*

File Type	Additional Features and Options
Text Files	• To automatically determine data arrangement, select one of the following options: – **Data (Using Preferences)** – **Data (Best Guess)** – **Text in the Script Editor** • To manually specify data arrangement, select the **Data (Using Preview)** option. See "Options in the JMP Text Import Wizard".
SAS Data Sets	• Use SAS variables for column names • (Windows only) Enter a password when you open a password-protected data set. • (SAS Transport files only) Select columns before opening
SPSS Data Files	Use SPSS variable or label names for column names.
XML Data Files	See "Text Files" for more information about the options.
JSON Data Files	See "Text Files" for more information about the options.
All Files	• To automatically determine data arrangement, select one of the following options: – Open as data using preferences – Open as data using a best guess – Open as plain text into a script window • To manually specify data arrangement, select the **Data (Using Preview)** option. • (Windows only) To open text files using your computer's default text editor, select **Use default program to open**.

Platforms and Reports

JMP consists of platforms that are organized by the type of statistical analysis. For example, the Distribution platform produces a univariate analysis (the distribution of a single variable) using histograms, additional graphs, and reports. You might analyze data in the Distribution platform and then choose another platform to examine the data more thoroughly.

Platforms and reports work together in these ways:

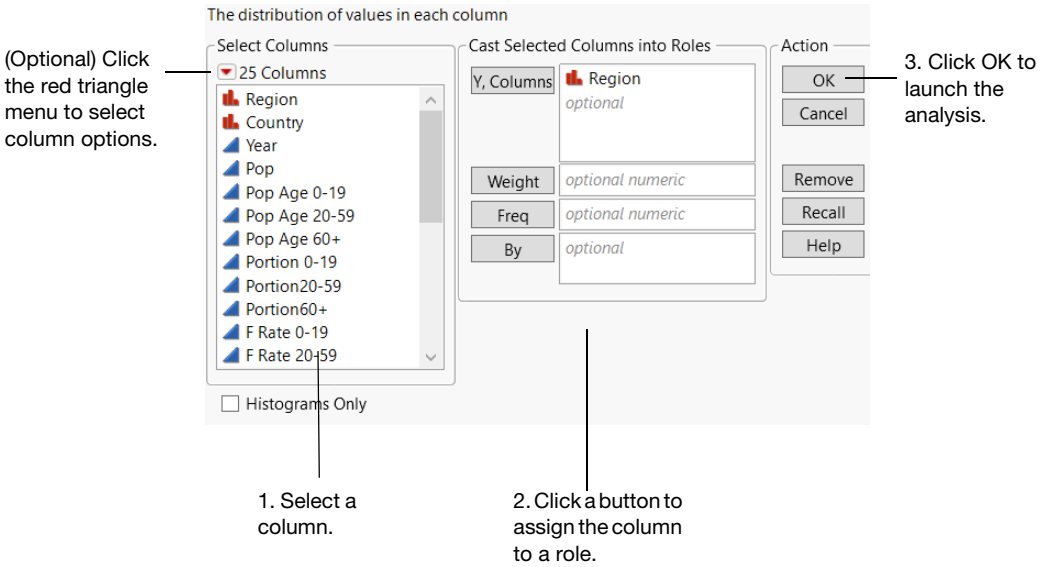
1. The data table is the input for a platform. See "Data Table Features".
2. The platform analysis starts in a launch window. See "Launch Windows in Platforms".
3. The platform results appear in a report window. See "Analysis Results in Reports".

"Anatomy of a Typical JMP Session on Windows" shows how a platform interacts with the data table and then displays the results in a report window. Information about using platforms is available in the JMP documentation library.

Launch Windows in Platforms

The launch window is your point of entry into a platform, where you specify the columns to analyze. Figure 2.11 shows the Distribution launch window.

Figure 2.11 Launch Window Features



All launch windows have in common provide the following options:

Select Columns Lists all of the variables in your current data table. Note the following:

- Right-click the modeling type icon next to a column name to change the modeling type.
- Right-click the column name to create a transform column. See "Transform Columns in a JMP Platform".
- Filter and sort the columns using the options in the red triangle menu. See "Column Filter Menu".

Cast Selected Columns into Roles Moves selected columns into roles (such as **Y**, **X**, and so on.) You cast a column into the role of a variable (like an actor is cast into a role). See "Cast Selected Columns into Roles Buttons".

In the Graph Builder window, click the **Dialog** button on the left or select **Redo > Relaunch Analysis** to show this panel.

Action The following options are available:

OK Performs the analysis.

Cancel Stops the analysis and quits the launch window.

Remove Deletes any selected variables from a role.

Recall Populates the launch window with the last analysis that you performed.

Help Takes you to the Help for the launch window.

Tip: To reopen the platform launch window that was last opened, select **View > Recall Last Platform** (Windows) or **Window > Recall Last Platform** (macOS).

Cast Selected Columns into Roles Buttons

The following buttons frequently appear throughout launch windows. Buttons that are specific to certain platforms are described in the chapter for the specific platform.

Y Identifies a column as a response or dependent variable whose distribution is to be studied.

X Identifies a column as an independent, classification, or explanatory variable that predicts the distribution of the Y variable.

Weight Identifies the data table column whose variables assign weight (such as importance or influence) to the data.

Freq Identifies the data table column whose values assign a frequency to each row. This option is useful when a frequency is assigned to each row in summarized data. If the value is 0 or a positive integer, then the value represents the frequencies or counts of observations for each row when there are multiple units recorded.

JMP PRO **Validation** Identifies the data table column whose values assign rows to training, validation, and test sets for cross validation in fitting models.

Notes:

– For some platforms, KFold validation is available if you specify more than three levels in the Validation column.

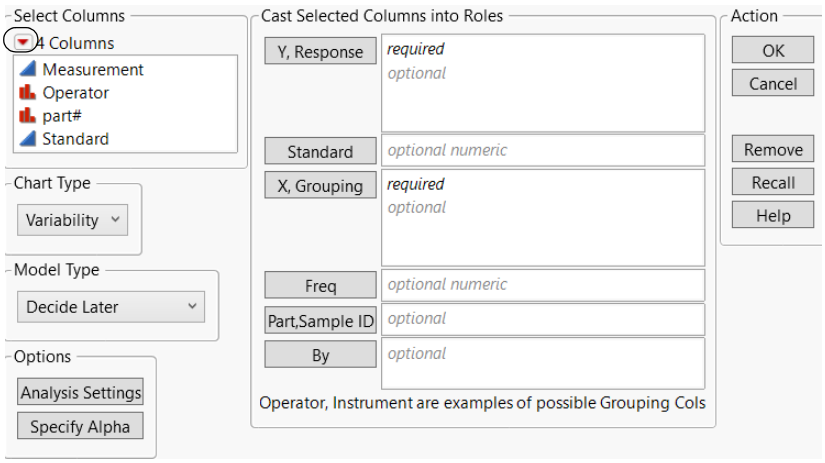
– If you click the Validation button with no columns selected in the Select Columns list, you have the option to create a Validation column. For more information about the Make Validation Column utility, see *Predictive and Specialized Modeling*.

By Identifies a column that creates a report consisting of separate analyses for each level of the variable.

Column Filter Menu

A Column Filter menu appears in most of the launch windows. The Column Filter menu is found by clicking the red triangle in the Select Columns panel. Use these options to sort columns, show or hide columns, or search columns.

Figure 2.12 The Column Filter Menu



Reset Resets the columns to its original list.

Show Filter Opens a box in which you type the column name that you want to find. Click the arrow next to the box and select options to customize the filter.

Sort by Name Sorts the columns in alphabetical order by name in ascending, descending, or original order.

Modeling Type Provides options for showing or hiding columns with specific modeling types.

Numeric Shows or hides columns whose data type is Numeric.

Character Shows or hides columns whose data type is Character.

Expression Shows or hides columns whose data type is Expression.

Exclude Formats Excludes columns with specific formats from the column selection list.

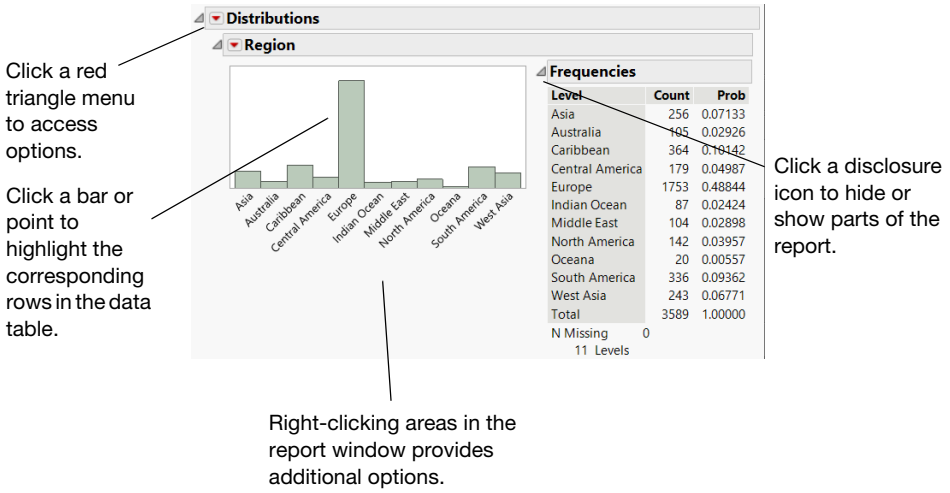
Column Groups Shows or hides groups of columns. See "Group Columns in Data Tables".

Ungrouped Columns Shows or hides columns that have not been grouped.

Analysis Results in Reports

After you launch your analysis, the JMP report window appears. The report window shows the output of your analysis using interactive graphs and text reports. See "JMP Reports".

Figure 2.13 Report Window Features



Manage Files and Open Windows

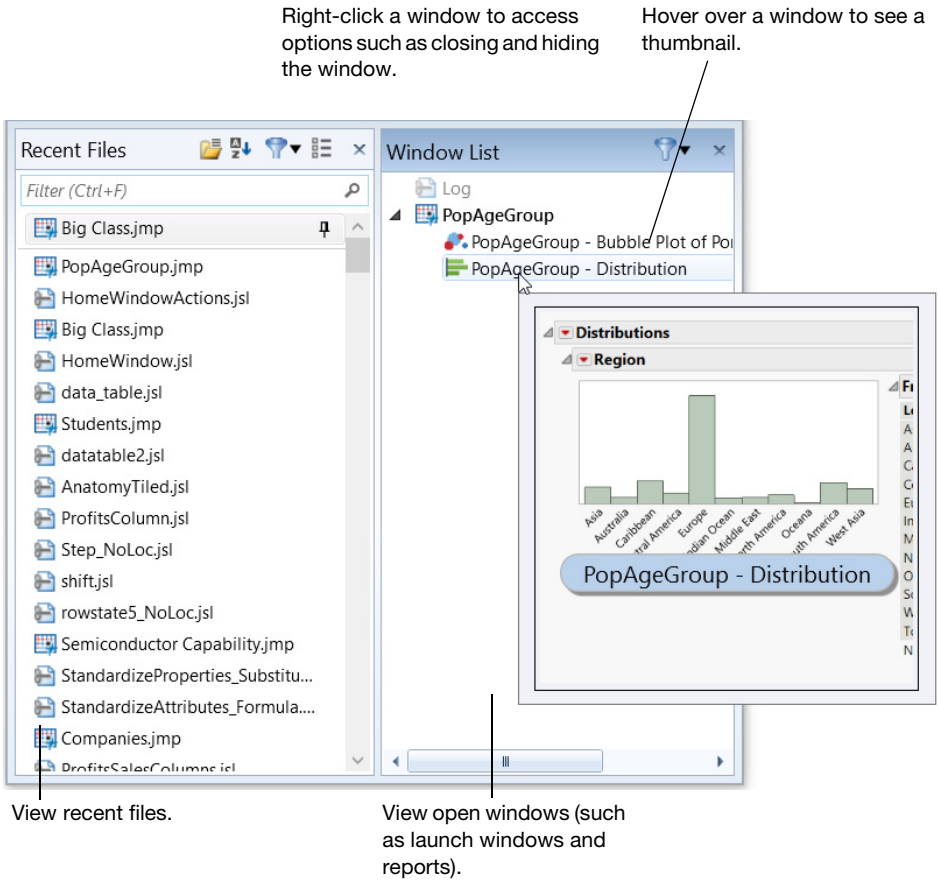
The JMP Home Window provides instant access to open window and files. Opening recently opened files, closing windows, and pinning frequently used files are a few of the options.

JMP Home Window on Windows

On Windows, the JMP Home Window appears when you open JMP.

- Open recent files in the Recent Files list or pin a frequently used file.
- Right-click files in the Recent Files list to perform common tasks.
- Open or close active JMP windows in the Window List.
- Set an open data table as the current data table by selecting it from the data table list at the top.

Figure 2.14 JMP Home Window Actions



JMP Home Window Buttons

On Windows, the JMP Home Window panels can contain the following buttons:

The Open Selected button opens the selected files in the Recent Files list.

The Sort By Name button sorts recent files alphabetically.

The Filter button filters the types of files that appear in the Recent Files and Window List panels.

Enlarges the font size and icon size of items in the Recent Files list.

The Close button closes the panel.

Recent Files

On Windows, the Recent Files list provides quick access to files that you recently opened.

Keep your favorite files at the top of the list for quick access. Hover over the filename and click the pin icon . To unpin a file, click the pin icon .

You can drag files from the Recent Files list into these locations:

- the JMP Window List to open the file
- a Windows folder or on to your desktop to create a copy of the file
- applications such as Microsoft Notepad or Microsoft Word to edit the file

Right-click files to access the following options:

- Open a file
- Copy the location path to a file
- Open a file within the folder that contains it
- Sort files alphabetically by name or by most recent
- Remove a selected file or files from the list
- Remove a file or files that are no longer in the same location (Alternatively, you can select the files and press Delete.)
- (JSL scripts) Edit, run, or debug a script
- (Text files) Change the import method or open as plain text
- (SQL queries) Run an SQL query that you saved in Query Builder. You can also create a new query using an existing query as a template.
- Run or edit a JMP application

  When you open a non JMP file from the Recent Files list, JMP applies your import preferences to arrange the data. You can right-click a text file to change the import method or to open the file in a text editing window. Your import preference is bolded in the pop-up menu.

Tip: To prevent files that were opened by a script from appearing in the Recent Files window, select **File** > **Preferences** > **General** and de-select **Add files opened by scripts to the Recent Files list**.

Search the Recent Files List for JMP Files

To find a recently opened file in a long list of files, enter part of the file name in the Recent Files Filter box.

The following options are provided:

- Enter part of the file name in the Filter box above the list of files.
- Hover over the Filter box to view search tips.

- As the files are filtered, the Up and Down arrow keys change the selected file. Press Enter to open the selected file, or select the file with your cursor.
- If a file is selected in the Recent Files list, press Ctrl+F to activate the Filter box.
- Press Esc or click the X button to exit the filtering mode after you typed a search term.

Window List

On Windows, the Window List shows open JMP windows, such as data tables, reports, and scripts. You can open, close, rearrange, and hide JMP windows.

- If you hover over a file in the Window List, a thumbnail appears.
- To open the Window List in its own window, select **View > Window List**.
- To display windows side-by-side, right-click the selected windows and select **Arrange**.
- To always automatically display the Window List inside maximized windows, select **File > Preferences > Windows Specific** and select **Dock the Window List in maximized windows**.

Right-click files to access the following options:

- View a window
- Close a window (Alternatively, you can select files and press Delete.)
- Close all windows except the currently selected window (if the selected window is a report window, the dependent data table also remains open)
- Move a window to the back
- Hide a window from the Windows taskbar or unhide a window
- Select all windows, or clear all selections

Note: The active file is formatted with bold text.

Rearrange Panels in the Home Window

To rearrange the panels in the Home Window on Windows, click and drag the title bar of the panel. Drop the title bar onto a top, bottom, left, or right arrow to position the panel. A blue box indicates where the panel will be placed. To turn the panels into tabs, drag and drop any panel into the middle of the Home Window.

Tips:

- To put the Home Window back into its original order, select **View > Home Window Panes > Revert to Factory Layout**.
- If you close a panel and want to reopen it, select **View > Home Window Panes** and select the panel that you want to open.

- Open the JMP Home Window by selecting **View > Home Window** or clicking the **JMP Home Window** button in the lower right corner of most JMP windows. If you cannot see the **JMP Home Window** button, select **View > Status Bars.**
- If you prefer to see the JMP Starter or the Window List upon start-up, you can specify that in the Preferences (Windows only). Select **File > Preferences > General** and select an option from the Initial JMP Window list.

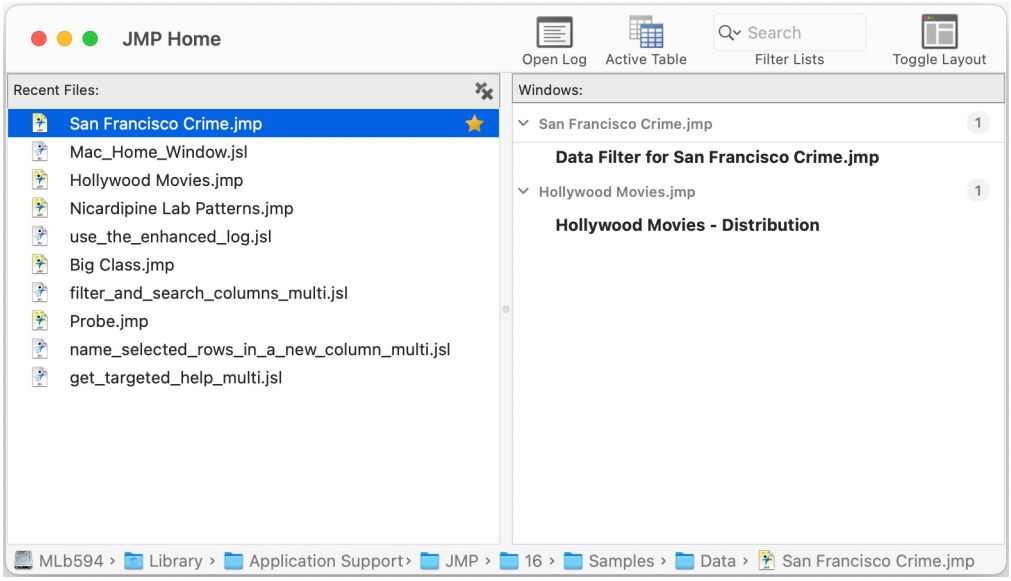
JMP Home Window on macOS

Use the JMP Home window on the macOS to quickly access JMP files and open windows.

- Open recent files in the Recent Files list.
- Press Control and select a file in the Recent Files list to perform common tasks, such as adding or removing a favorite.
- Open or close active JMP windows in the Windows list.
- Set an open data table as the current data table by selecting it from the Active Table list. Press Option while selecting the data table from the list to activate the data table without bringing it to the front.
- Open the log.
- Find a file in the Recent Files list or an open window in the Windows list by entering a search term in the Search box.
- Click **Toggle Layout** to display the Recent Files and Windows lists horizontally instead of vertically.

Open the JMP Home window on macOS by selecting **Window > JMP Home** (Figure 2.15). To view the window each time you open JMP, select **JMP > Preferences > General > Home Window**.

Figure 2.15 Example of the JMP Home Window (macOS)



Tips:

- The Status bar area shows the path for the currently selected file. You can click a folder to open the folder to view its contents.
- To be able to quickly view the JMP Home window from any JMP window, press Ctrl and click the window, select **Customize Toolbar**, and drag the JMP Home icon to the toolbar.
- To copy a file from Recent Files to another document or into the Finder, select the file and then select **Edit > Copy**.

Recent Files

On macOS, the Recent Files list provides quick access to files that you recently opened.

You can also click the star that appears when you hover over the file name to add it to your favorites.

Press Ctrl and select a file to access other options. For example, you can run or debug a JSL script by pressing Ctrl and selecting Run Script or Debug Script.

Tip: To prevent files that were opened by scripts from appearing in the Recent Files window, select **JMP** > **Preferences** > **General** and de-select **Add files opened by scripts to the Recent Files list**.

Windows List

On macOS, the Windows list shows open JMP windows, such as data tables, reports, and scripts.

To close a window, click the **x** button that appears when you hover over the window in the Windows list.

If the window that you are closing is a parent to one or more children, you are prompted to **Hide Data Table**, **Cancel**, or **Close All** windows. Select **Close All** to close the parent and all its children windows.

Search for Recently Opened Files on Windows

On Windows, recently opened files appear in the Home Window. You can also search for recently opened files from any JMP window.

Select **File > Quick Open** or press Alt+Shift+O to open the Filter box and enter your search term. This option also lets you view the path of the file. Press Esc or click the X button to close the window after you type a search term.

Close Multiple Files

On Windows, close several files at once from the JMP Home Window. Select the files in the Window List, right-click, and select **Close**. To leave a single file opened and close the rest, select the single file and then select **Close All But This**.

On macOS, select **File > Close Multiple** and then select the files that you want to close.

Display and Arrange Open Windows

In JMP, typically you have several windows open at once (for example, data tables, reports, and the JMP Home Window). JMP provides several ways to arrange and display these open windows.

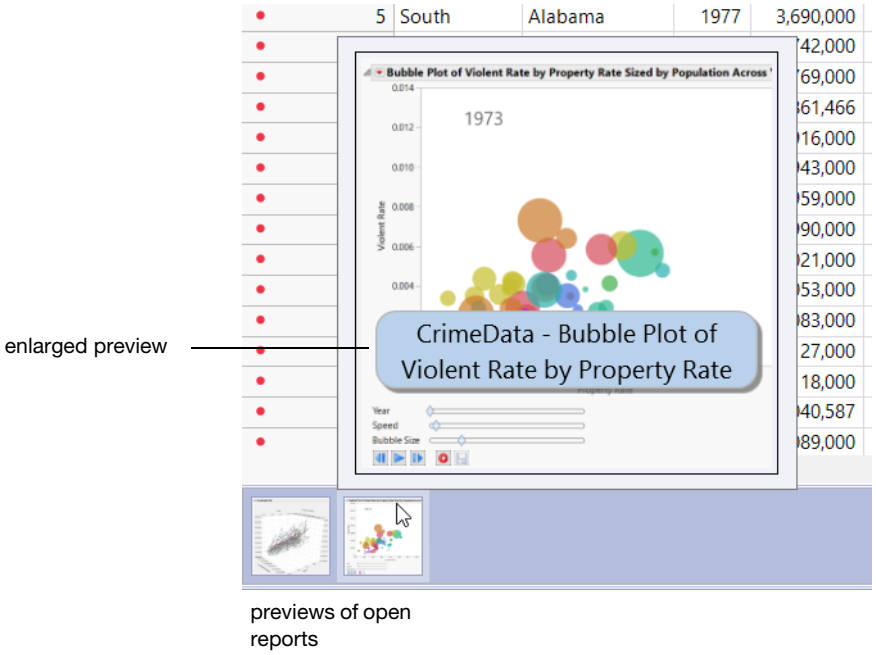
On Windows:

- Press Ctrl+Tab to switch between windows.
- Use the Reveal feature (F9 or Window > Reveal). See "Using the Windows Reveal Feature".
- Use the options in the Window menu. Note that Arrange options are also available using the Arrange Menu option in the bottom right corner of most windows. If you cannot see Arrange Menu, JMP Home Window, and View Associated Data buttons, select **View > Status Bars**.

Tip: From the Arrange Menu option on Windows, you can merge open windows by selecting Combine Windows. See "Example of Combining Windows to Create a Dashboard".

- To display the JMP Home Window, click the **JMP Home Window** button in the lower right corner of most windows, or press Ctrl+1.
- To show the data table for a report, click **View Associated Data** button in the bottom right corner of the report.
- To open an associated report when you are viewing a data table, double-click the thumbnail preview of the report in the bottom pane. To enlarge the preview, hover over the thumbnail (Figure 2.16).
- Middle-click the thumbnail preview to close the associated report.

Figure 2.16 Thumbnail Previews of Open Reports



On the macOS:

- Use the options in the Window menu.
- Use the Expose feature (F9 or F3).

Preview JMP Files

For some JMP file types, you can use Windows Explorer or macOS Finder to view a portion of the selected file. Windows supports a preview of JMP data tables, journals, and scripts. macOS supports a preview of data tables and scripts.

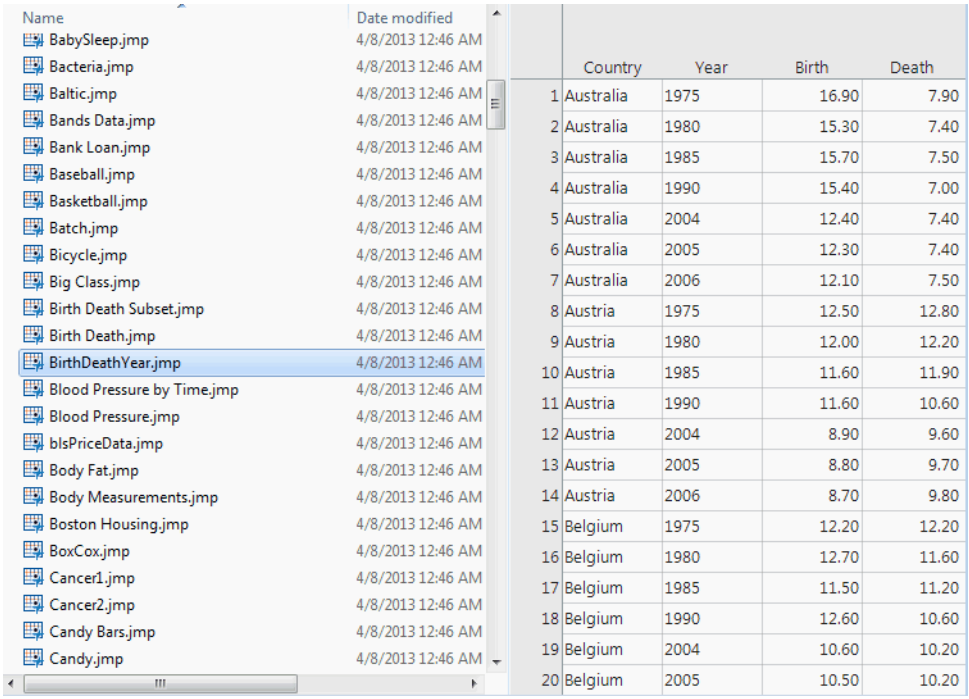
To preview a JMP file

1. In Windows Explorer or the macOS Finder, select the file that you want to preview.
2. In Windows Explorer, select the **Show the preview pane** button in the upper right corner. Support in e-mail programs varies.

   In the macOS Finder, select the file and press the spacebar, or select the **Quick Look** option on the pop-up menu or File menu. In Mail, select the **Quick Look Attachment** option on the pop-up menu or File menu.

   The preview pane shows a portion of the selected file.

Figure 2.17 Data Table Preview Pane (Windows)



	Country	Year	Birth	Death
1	Australia	1975	16.90	7.90
2	Australia	1980	15.30	7.40
3	Australia	1985	15.70	7.50
4	Australia	1990	15.40	7.00
5	Australia	2004	12.40	7.40
6	Australia	2005	12.30	7.40
7	Australia	2006	12.10	7.50
8	Austria	1975	12.50	12.80
9	Austria	1980	12.00	12.20
10	Austria	1985	11.60	11.90
11	Austria	1990	11.60	10.60
12	Austria	2004	8.90	9.60
13	Austria	2005	8.80	9.70
14	Austria	2006	8.70	9.80
15	Belgium	1975	12.20	12.20
16	Belgium	1980	12.70	11.60
17	Belgium	1985	11.50	11.20
18	Belgium	1990	12.60	10.60
19	Belgium	2004	10.60	10.20
20	Belgium	2005	10.50	10.20

Using the Windows Reveal Feature

The Reveal feature shows a top-level window with thumbnails of all open JMP windows. On Windows, press F9 to use the Reveal feature.

- Click a window to activate it.
- Press the spacebar to turn on Preview mode, where a full-sized view of the thumbnail under the cursor appears in the center of the screen.
- By default, the windows are ordered to match the original ordering on the screen in a top-down, left-to-right order. Press S to sort the thumbnails by name.

  To exit the Reveal feature, press F9, Esc, or Alt+Tab and then select another application.

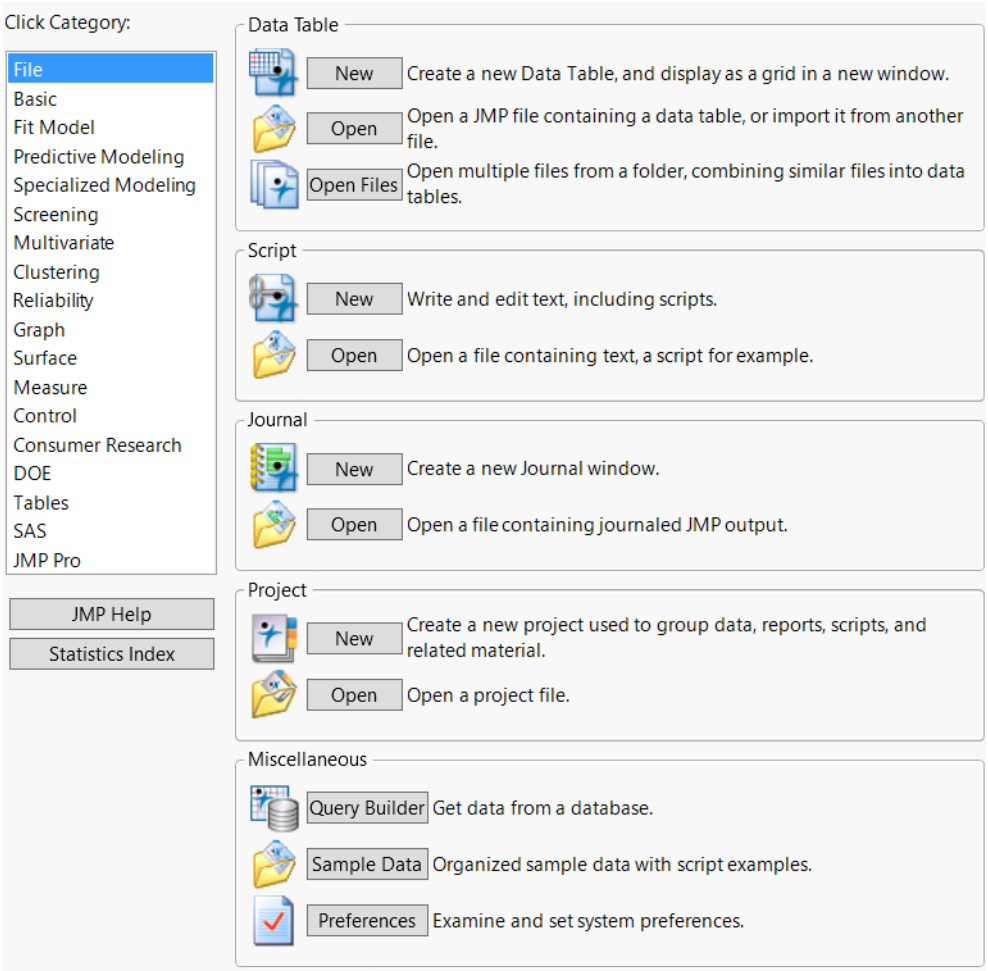
JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On the macOS, select this option on the **JMP > Preferences > General** page.

Figure 2.18 shows the JMP Starter window for Windows. The macOS JMP Starter window is identical.

Figure 2.18 The JMP Starter Window (Windows)



Chapter 3

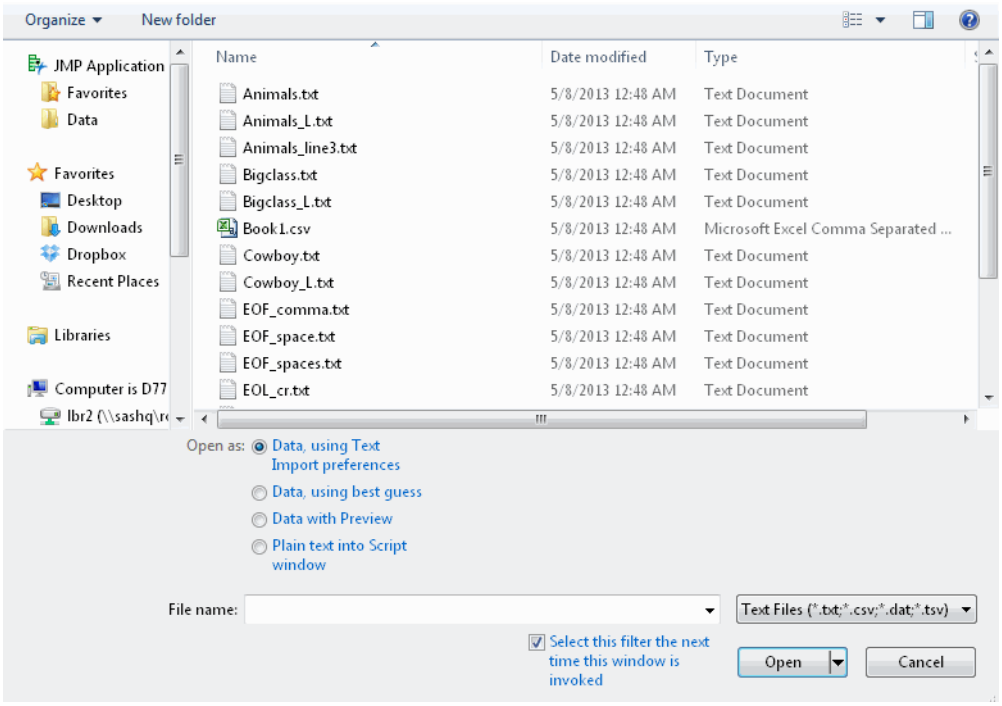
Import Your Data

Create Data Tables

This chapter covers the following topics:

- How to import data, such as text files, PDF files, SPSS files, and SAS data
- How to transfer Excel data
- How to read in real-time data
- How to create a new data table

Figure 3.1 Importing a Text File



Contents

About Importing Data 70
Import Microsoft Excel Files 71
Preview and Import Microsoft Excel Data 71
Import a Microsoft Excel File Directly 83
Import Text Files 84
Find Missing Characters in Imported Data 86
Options in the JMP Text Import Wizard 87
Edit Text Before Importing into JMP 91
Import Text from the Script Editor 93
Import Multiple Files 94
Import PDF Files 99
Concatenate Tables into One Data Table 100
Concatenate Tables by Column Headers 102
Import Tables that JMP Does Not Auto-Detect 102
PDF Preview Red Triangle Options 105
Data Table Preview Options 105
PDF Preview Right-Click Options 106
Import XML and JSON Files 106
Preview and Import XML and JSON Data 106
Import XML Files Directly 114
Import JSON Files Directly 114
Build SQL Queries in Query Builder 115
Connect to a SQL Database 116
Select Tables from a SQL Database 116
Build an SQL Query 121
Save and Run a JMP Query 135
Open the Selected Data 136
Query Builder Red Triangle Options 137
Write SQL Statements in Query Builder 138
Import Data from SAS 138
Java Runtime Environment (JRE) Requirements 138
SAS Menu Options 139
Import SAS Data Sets 139
Create SAS Transport Files in SAS from JMP 142
Connect to SAS from JMP 142
Open SAS Data Sets with SAS Query Builder 148
Open SAS Data Sets through a SAS Server 149
Using SAS Extended Attributes to Import Metadata 157
Run Stored Processes 159

Submit SAS Code . . . 159
Generate ODS Results . . . 161
Import Generated SAS Data Sets . . . 162
Import Data from a Database . . . 164
Connect to a Database . . . 165
Open Data from a Database . . . 166
Write SQL Statements to Query a Database . . . 168
Structured Query Language (SQL): A Reference . . . 170
Import Data from Google Sheets . . . 176
About Importing Google Sheets . . . 177
Google Sheet Settings . . . 177
Import SPSS Files . . . 178
Import Triple-S Survey Data . . . 180
Import HDF5 Data . . . 180
Import Data from MATLAB . . . 181
Import Data from R . . . 181
Import Data Using the Excel Add-In . . . 181
Transfer Excel Data to a Data Table . . . 182
About the JMP Add-In for Excel . . . 183
Uninstall the JMP Excel Add-In . . . 184
Import Data from the Internet or a Remote Computer . . . 184
Create New Data Tables . . . 186

About Importing Data

You can import many file formats into JMP and save them as data tables. JMP opens many files by default. The file formats which JMP does not support by default require specific Open Database Connectivity (ODBC) drivers.

The Following File Formats Are Supported by Default:

- Comma-separated (.csv)
- .dat files that consist of text
- ESRI shapefiles (.shp)
- Flow Cytometry versions 2.0 and 3.0 (.fcs)
- Hierarchical Data Format, Version 5 (.h5)
- HTML (.htm, .html)
- JSON (.json)
- MATLAB (.m, .M)
- Microsoft Excel 1997 through 2019 on macOS (.xls, .xlsx)
- Microsoft Excel 2010 through 2019 on Windows (*.xlsx, *.xlsm)
- Minitab Portable Worksheet (.mtp)
- Plain text (.txt)
- R (.r)
- SAS transport (.xpt, .stx)
- SAS versions 7 through 9 on macOS (.sas7bdat)
- SAS versions 7 through 9 on Windows (.sas7bdat, .sas7bxat)
- SPSS (.sav)
- SQLite 3.0 or higher (.sqlite, .db, .sqlite3, .db3)
- Tab-separated (.tsv)
- Teradata database (.trd)
- Triple-S (.sss, .xml)
- xBase data files (.dbf)
- XML data files (.xml)

Notes on SAS Support:

On both Windows and macOS, you can open SAS data sets directly through the **File > Open** command. See "Import SAS Data Sets".

Another option is connecting to a SAS server by selecting **File > SAS > Browse Data**. See "Open SAS Data Sets through a SAS Server".

The Following Files Require ODBC Drivers:

- Database (dBASE) (.ndx, .mdx) is supported with a V3+ compliant ODBC driver. .dbf files do not require an ODBC driver.
- Microsoft Access Database (.mdb) is supported with a V3+ compliant ODBC driver.

See "Import Data from a Database" for more information for working with databases.

Your computer's available memory affects data import. Very large files might load slowly or not at all. Consider splitting up large files before importing them. You can then join or concatenate the tables. See "Concatenate Data Tables" and "Join Data Tables".

Note: You can open R code (.R) and SAS program files (.sas) in JMP, but the text opens in a Script window, not in a data table.

Import Microsoft Excel Files

Microsoft Excel files open in the Excel Import Wizard by default. The wizard shows a preview of the data. You can then modify the settings before importing the data. For example, you might indicate which row the data begin on and whether the worksheet contains column headers or hidden rows or columns. Microsoft Excel .xls, .xlsm, and .xlsx file formats are supported.

For information about opening a Microsoft Excel file outside the wizard, see "Import a Microsoft Excel File Directly".

Notes:

- Between Windows and macOS, the number of digits after a decimal point might differ.
- Consider setting the *Autosave timeout* value in the General preferences to automatically save open data tables. This autosave value also applies to other JMP document types.

Preview and Import Microsoft Excel Data

Before you import a worksheet, open the spreadsheet in Excel and decide how you want the data to be structured in the final data table. For example, you need to know whether the worksheet includes hidden or merged cells. In the wizard, you can then exclude hidden columns or rows.

To import a Microsoft Excel file that contains several worksheets, follow these steps:

1. Open the worksheet in Microsoft Excel.

   For the figures in this example, we used the Team Results.xlsx file located in the JMP Samples/Import Data folder. The file has the following characteristics:

   – the data begin on row 4, column 2 and end on row 9, column 5
   – two worksheets
   – the second worksheet has two sets of merged cells
   – no hidden rows or columns

Figure 3.2 Team Results.xlsx Worksheet

	A	B	C	D	E	F
1		Here are the team results for this year's game.				
2						
3		**Team**	**Member Name**	**Age**	**Winnings**	
4		1	Joe	21	$50.22	
5		1	Mary	22	$24.34	
6		1	Cindy	23	$25.54	
7		2	Mark	22	$52.11	
8		2	Bill	23	$43.32	
9		2	Jennifer	24	$11.23	

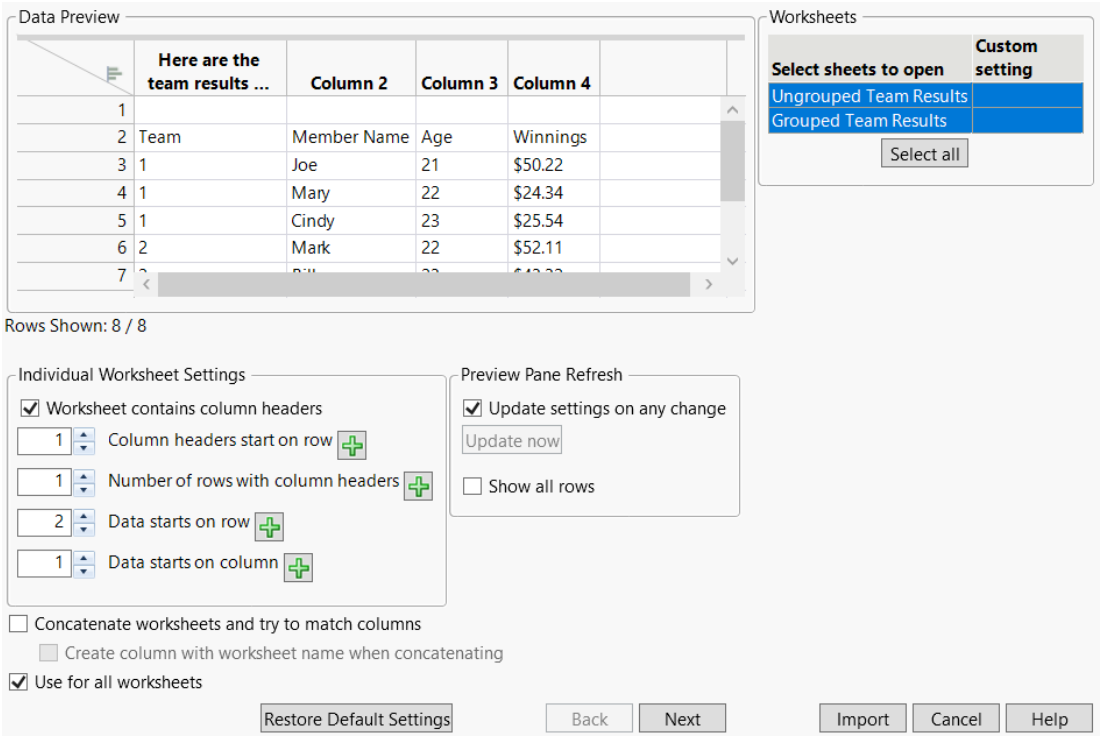
2. To open an Excel file, select **File > Open**.

   The Open Data File window appears.

3. Select **Excel Files** next to File name.
4. Select the Excel file and click **Open**.

   The worksheet opens in the Excel Import Wizard, where a preview of the data appears along with import options.

Figure 3.3 Example Initial Data Preview



Note the following characteristics in the Data Preview:

- Both worksheets are selected for import in the upper right corner.
- The first column has been automatically been removed.
- Text from the first row of the worksheet appears as the column headings. However, you want the text in row 3 of the worksheet to be used as the column headings.
- The first data row is empty.

Note: JMP remembers your previous changes each time you import a worksheet, even after closing and reopening JMP. This feature is helpful when you want to reimport the same worksheet several times and experiment with options. To clear those changes when you import a different worksheet, click **Restore Default Settings**.

5. For **Column headers start on row**, type 3.
6. For **Data starts on row**, type 4.
7. For **Number of rows with column headers**, type 1.
8. **Data starts on row** should be set to 4.
9. **Data start on columns** should be set to 1.

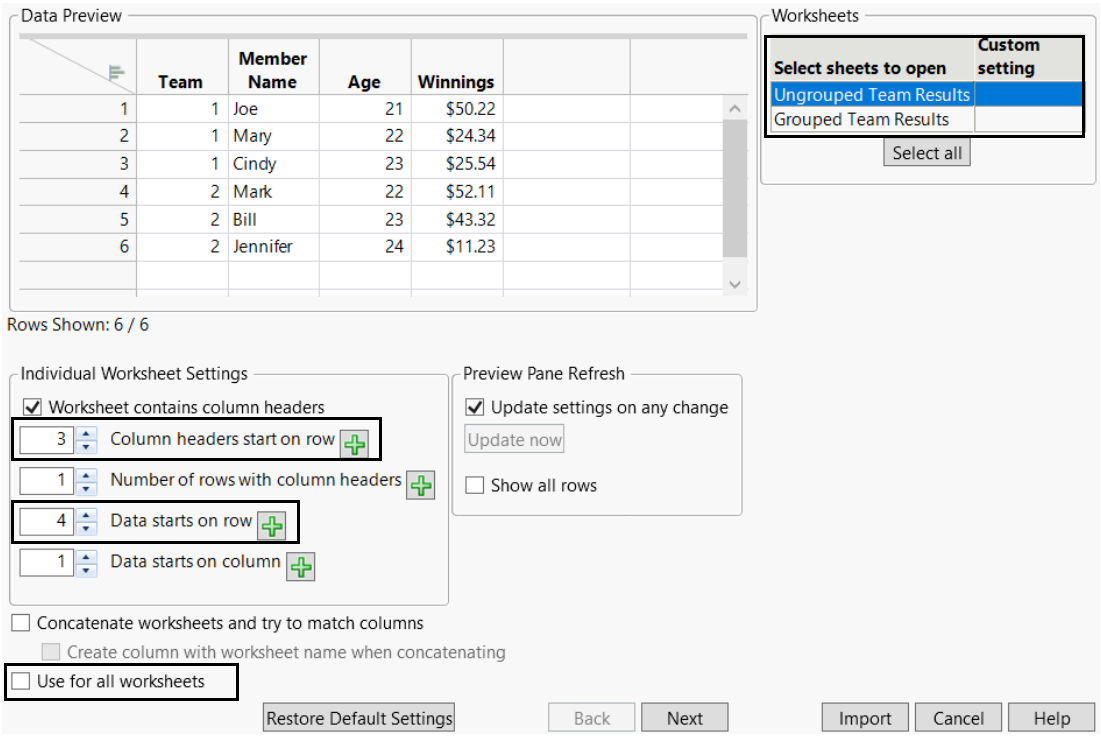
10. Select **Ungrouped Team Results** in the Worksheets pane.

    Only this worksheet will be imported.

11. Deselect **Use for all worksheets**.

    These settings apply only to Ungrouped Team Results.

Figure 3.4 Selecting the Column Header Row



Tip: Right-click a numeric column header in the Data Preview pane to change the format.

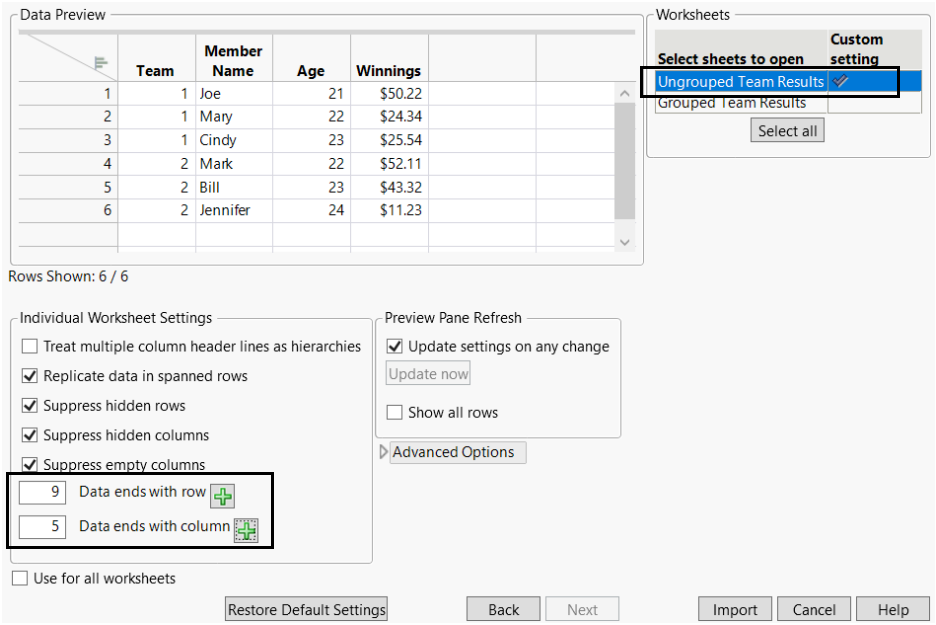
See "Individual Worksheet Settings" for more information about all options.

12. Click **Next** to configure other import settings.

    The window displays additional import settings.

13. For **Data ends with row**, type 9.
14. For **Data ends with column**, type 5.

Figure 3.5 Specifying the Last Column

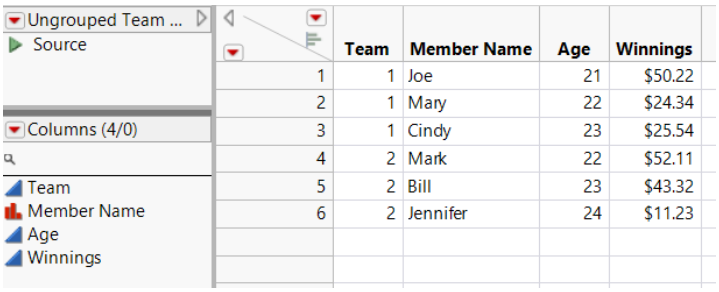


Custom setting is automatically selected when you modify the import options.

See "Additional Individual Worksheet Settings" for more information about all options.

15. Click **Import** to convert the worksheet as you specified.

Figure 3.6 Final Data Table



The following sections describe options in the Excel Import Wizard.

Individual Worksheet Settings

Worksheet contains column headers Select if the worksheet contains rows with column headers.

Column headers start on row Indicates which row the column headers begin on in the worksheet. Click the up arrow until the headers begin on the correct row, or enter the row number and press Enter.

Number of rows with column headers Indicates whether the worksheet has multiple rows as column headers. Click the up arrow until the header rows appear correctly, or enter the number of rows and press Enter.

Data starts on row Indicates which row the data start on in the worksheet.

Data starts on column Indicates which column the data start on in the worksheet.

Concatenate worksheets and try to match columns Merges all worksheets into one data table. JMP matches columns that have the same header.

Create column with worksheet name when concatenating Adds a new Source Table column that lists the worksheet name for each imported table. This option is available after you select the preceding concatenate option.

Use for all worksheets Applies the current import settings to all worksheets that are selected in the upper right corner.

Additional Individual Worksheet Settings

Treat multiple column header lines as hierarchies Indicates that the worksheet contains multiple rows as column headers and you want these headers to be hierarchies. This option is only for stackable data.

Replicate data in spanned rows Indicates cells are merged in the worksheet across rows. JMP unspan the cells and copy the cell contents into all of the resulting cells. The option is selected by default.

If you deselect **Replicate data in spanned rows**, JMP unspans the cells and copies the cell contents into the topmost cell. The remaining unspanned cells are left empty.

Suppress hidden rows Prevents hidden rows from appearing in the data table.

Suppress hidden columns Prevents hidden columns from appearing in the data table.

Suppress empty columns Indicates whether an empty column that has a column header is imported. Deselect the option to import the column.

Data ends with row Indicates the last row in the worksheet that contains data.

Data ends with column Indicates the last column in the worksheet that contains data.

Advanced Options

Column Name Separator String Indicates the separator between each word in a column heading if the headings were originally in different rows. Specify the number of rows with column headers on the first Excel Import Wizard window. Then enter a character or space

in the Column Name Separator String box. The default string, a hyphen, results in a column heading such as "First-Second-Third".

Multiple Series Stack Divides subcategories into separate columns in a worksheet with hierarchical headings. You must also select **Treat multiple column header lines as hierarchies**. The main category is imported as the Label column.

Replicate headers in spanned rows Repeats the header text in each cell for rows that are spanned in the worksheet. For example, the State column heading shown in Figure 3.7 was in a merged cell in the worksheet. On the left, the heading is replicated. Deselect the option to avoid repeating the heading as shown on the right.

Figure 3.7 Replicated Headers in a Microsoft Excel File

Replicated headers

	State-State
1	TX
2	TX
3	TX
4	TX

Unreplicated headers

	State
1	TX
2	TX
3	TX
4	TX

Import cell colors Applies the cell coloring from the worksheet to the data table. On macOS, only primary and secondary colors can be reliably imported.

Limit column type detection Scans a maximum of 100 rows to determine the column type. Select this option to speed up the import of large worksheets.

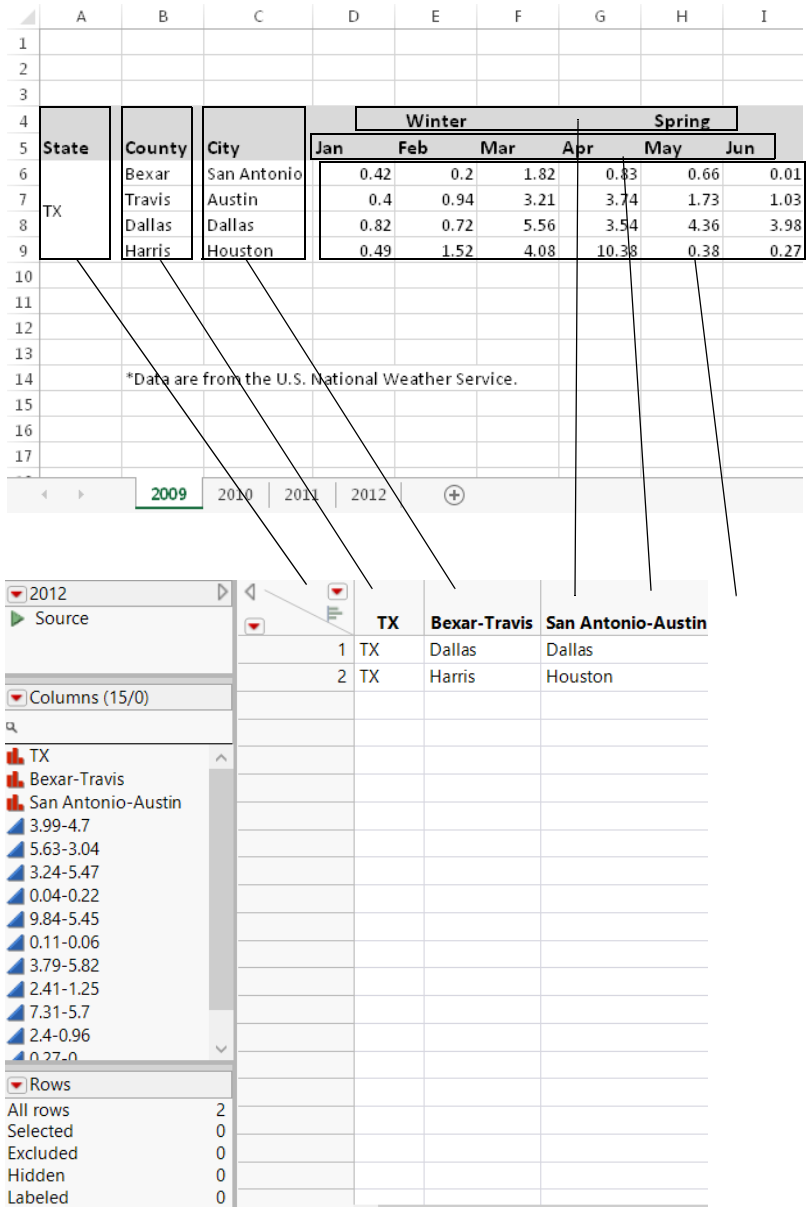
Tips:

- JMP remembers your previous changes each time you import a worksheet, even after closing and reopening JMP. This feature is very helpful when you want to reimport the same worksheet several times and experiment with options. To clear those changes when you import a different worksheet, click **Restore Default Settings**.
- Your import settings are saved in a data table script named Source. To reimport the worksheet using the same settings, run the script. The script includes the path to the worksheet, so make sure that other users have access to that location.
- To speed up the data preview in large worksheets, deselect **Update settings on any change** on the first wizard window. Modify the settings and then click **Update now** to refresh the data preview.
- To view all rows in the Data Preview pane, select **Show all rows**. The preview might be slightly delayed depending on the size of the spreadsheet.
- You can combine two worksheets from the same workbook into one data table. The column names are matched on import, so the order of the columns is irrelevant.

Import a Microsoft Excel File with Hierarchical Headings

In an Excel worksheet, multiple header rows can have an implied hierarchy; the second header row contains data that are categories of the first header row. Figure 3.8 shows an example. In the worksheet at the top, the seasons “Winter” and “Spring” are in spanned cells above the months within those seasons. In the JMP data table, you want the seasons in one column and their corresponding months in another column.

Figure 3.8 The Original Data in Excel and Final Data in JMP for 2012



To import the worksheets and maintain multiple column hierarchies, follow these steps.

1. In JMP, select **File > Open**.
2. In the Open Data File window, select Texas Precipitation.xlsx, located in the JMP Samples/Import Data folder, and then click **Open**.

The worksheet opens in the Excel Import Wizard, where a preview of the data appears along with import options.

Figure 3.9 Excel Import Wizard Preview

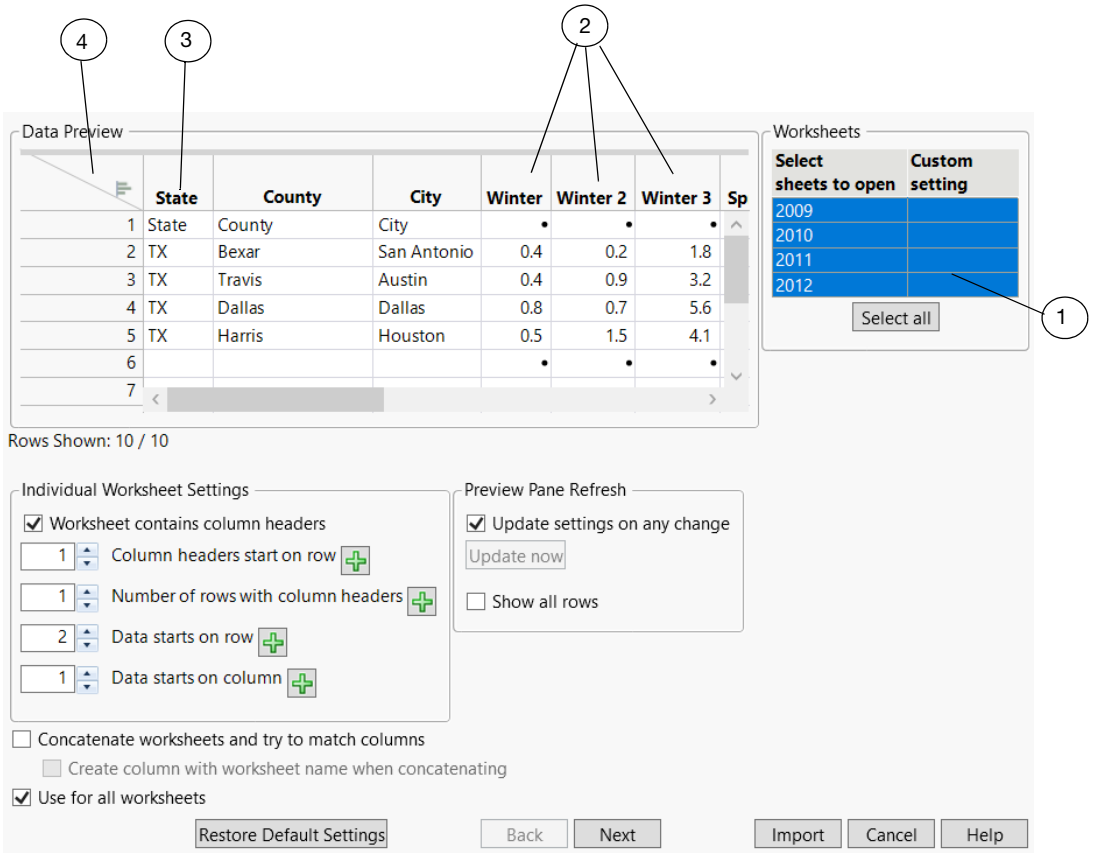


Figure 3.9 shows the default settings for this worksheet:

1	All worksheets are selected for import. Press Ctrl and click a worksheet to exclude it from the import.
2	Each season is split over several columns.
3	Data from the second heading row of the worksheet appear in the first row.
4	The empty rows at the top have been removed.

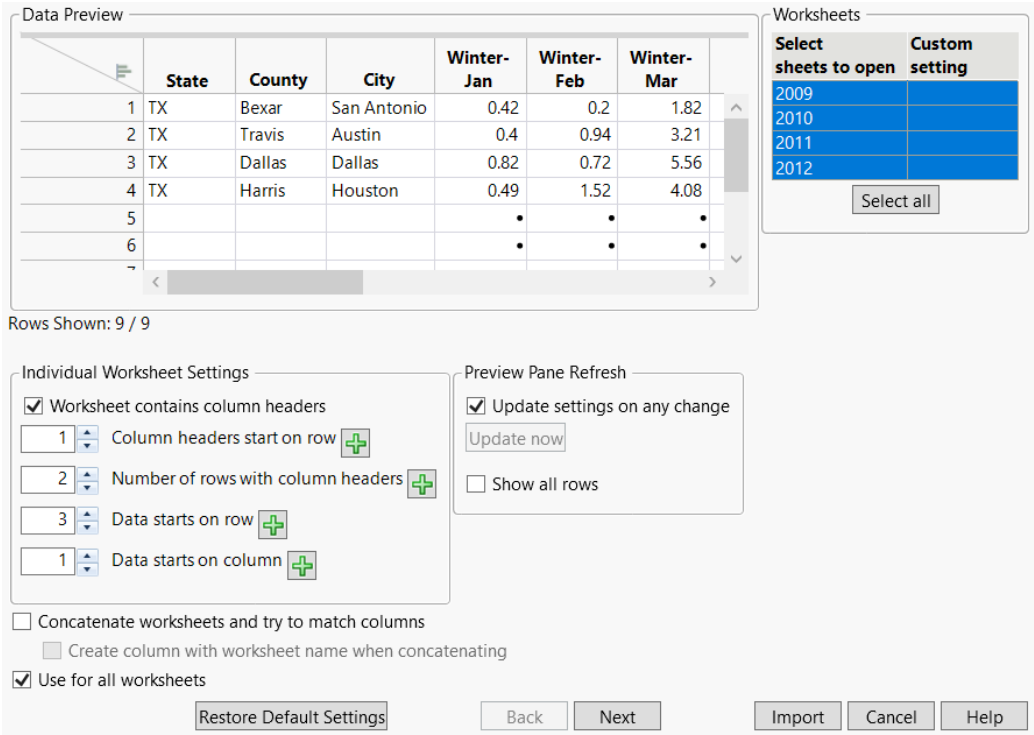
3. Under Preview Pane Refresh, make sure that **Update settings on any change** is selected.

This means that the Data Preview automatically refreshes when you make changes.

4. Next to **Number of rows with column headers**, click the up arrow once.

   Notice that **Data starts on row** automatically updates to 3.

Figure 3.10 Updated Settings on Page One

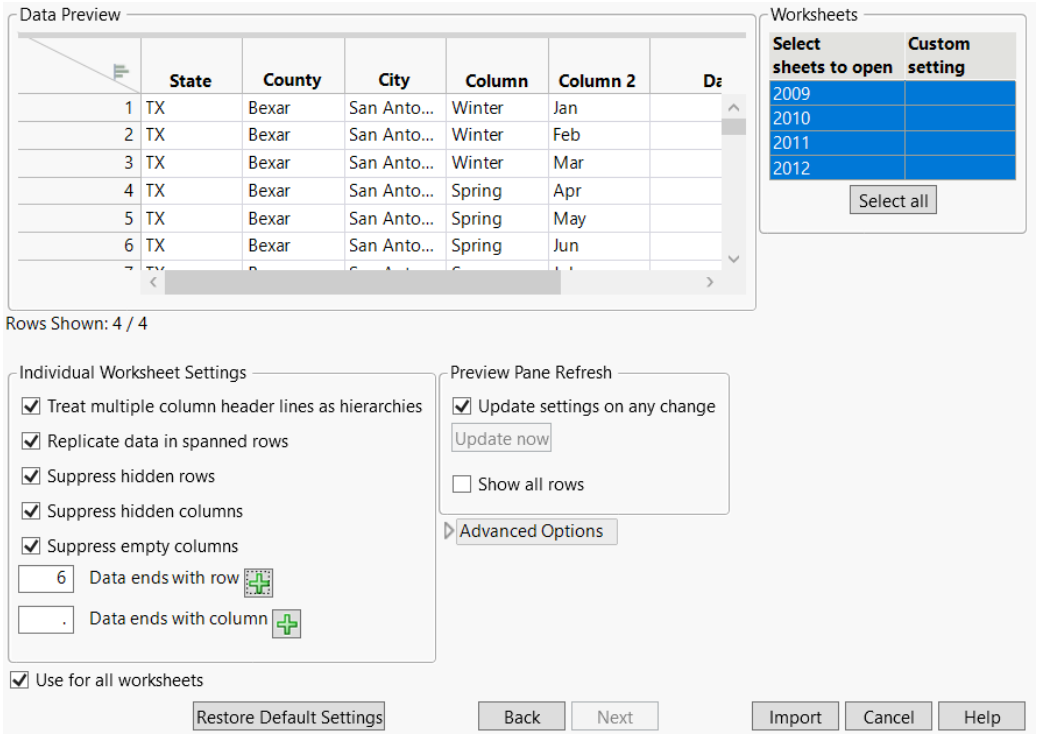


5. Click **Next**.
6. Next to **Data ends with row**, type 6 and press Enter.

   After the first three empty rows are removed from the worksheet, the data end with row 6.

Tip: Instead of you typing the end row number, JMP can calculate the row for you. In the Data Preview pane, select row 4. Click the plus sign next to **Data ends with row**.

Figure 3.11 Updated Settings on Page Two



Notice that **Replicate data in spanned rows** is selected by default. JMP unmerges data that were merged in the worksheet and copies the cell contents as separate categories. “TX” was originally in a merged cell. In JMP, it will be copied into separate cells.

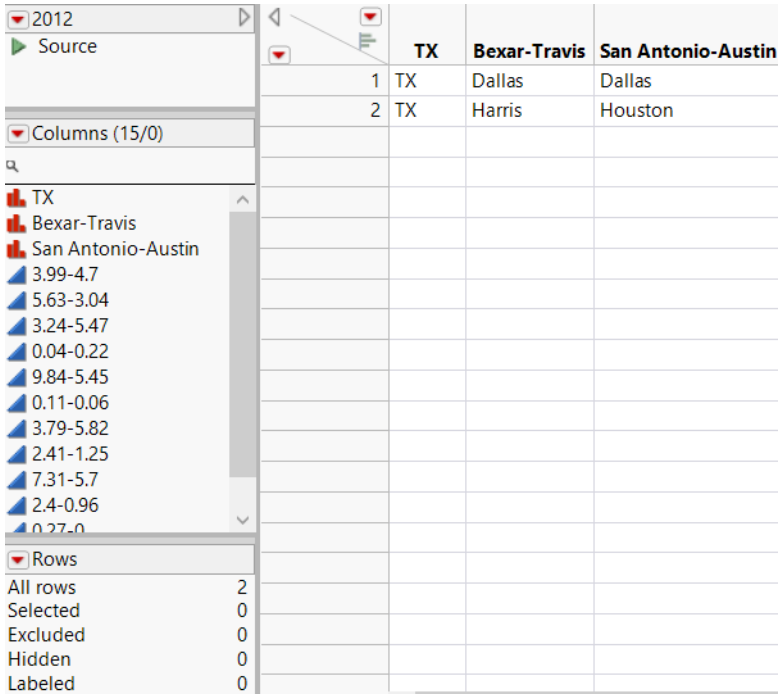
7. Select **Treat multiple column header lines as hierarchies**.

   Each season and month in the header rows of the worksheet will become categories in the data table.

8. Click **Import**.

   The four worksheets open as separate data tables. The data tables contain a Source script, which you can run to import the data into new data tables using the same import settings (Figure 3.12). Your import settings are also saved the next time you open the worksheet in the Excel Wizard in any JMP session.

Figure 3.12 The Final Data



Import a Microsoft Excel File Directly

Microsoft Excel files open in the Excel Import Wizard by default. This option is helpful when the structure of data in the worksheet is irregular. For example, you might want to exclude hidden columns or convert text in the third row to column headings.

Instead of opening spreadsheets in the Excel Import Wizard, you can select **File > Open** to open the file. By default, JMP detects whether the first row contains labels and converts them to column headings. You can change this setting in the General preferences. The Excel Open Method preferences are in File > Preferences > General (Windows) and JMP > Preferences > General (macOS). From the Use Excel Labels for Headings list, select **Always** or **Never**.

To set the Excel Open Method preference

To always open Microsoft Excel files outside the wizard, change the Excel Open Method preference. Choose to open all worksheets at once or select them from a list.

To open a Microsoft Excel file (Windows)

1. After you set the Excel Open Method as described above, select **File > Open**.

2. To convert text in the first row to column headings, select **Always** next to **Always enforce Excel Row 1 as labels**.
3. Select the **Excel Files** file type, select the file, or enter the URL.
4. To import all worksheets, click **Open**.

   or

   To select the worksheets that you want to open, click the **Open** button arrow, and then select **Open Selected Worksheets**. Select one or more worksheets and click **OK**.

To open a Microsoft Excel file (macOS)

1. Select **File > Open** and select the file.
2. (.xls only) To convert text in the first row to column headings, select **Use Excel Labels as Headings**.
3. (.xls only) To open specific worksheets, select **Select Individual Excel Worksheets**.
4. Click **Open**.

   If you chose to open specific worksheets, select those worksheets from the list, and then click **OK**. You can also click **Select All** if you change your mind and want to import all worksheets.

If you selected an .xlsx file, a preview of the data appears in the Excel Wizard. See "Preview and Import Microsoft Excel Data".

Import Text Files

You can open text files with the extensions .txt, .csv, and .tsv, and the text is converted to a data table. Files with the .dat extension that consist of text are also supported. Text files can be delimited using almost any character, or they can be fixed-width files.

Trailing whitespace is removed when you import text files into JMP.

When JMP finds an integer in the text file that is greater than *9,007,199,254,740,991*, the column is considered character data. You can set the column to numeric using the Text Import Wizard window or an import script.

Tip: You can import more than one text file at a time using the Multiple File Import option. See "Import Multiple Files".

To adjust import settings, choose from one of the following options

- Select **File > Preferences > Text Data Files** to change the import settings so that JMP determines the best way to structure and format the data table.

- Manually select the import settings as you open the file (described in this section).
- Open the file in the Script Editor, edit the content, and then import the content. This option is helpful when you need to add text delimiters or modify the text.

To import a text file

1. Select **File > Open**.
2. On Windows, you can set the file type to **Text Files**.
3. Select the text file that you want to open.

   For information about the options, see Table 3.1 and Table 3.2.

Table 3.1 Opening Text Files on Windows

Automatically Determining Data Arrangement	Manually Specifying Data Arrangement
1. Select **File > Open**. 2. From the list next to **File name**, select **Text Files**. 3. To use the import rules from the preferences, select **Data, (Using Preferences)**. (See "Preferences for Importing and Exporting Text Files".) To have text import use its best guess to arrange the data, select the **Data (Best Guess)** option. (Optional) Select the **Select this filter the next time this window is invoked** option to apply the filter that you chose by default. 4. Select the file that you want to open. 5. Click **Open**. **Tip:** The JMP Home window provides a shortcut to the above steps if you recently opened the file. Right-click the file in the Recent Files list and select **Import (Preferences)** or **Import (Best Guess).** (Your import preference is bolded in the right-click menu.)	1. Select **File > Open**. 2. From the list next to **File name**, select **Text Files**. 3. Select **Data (Using Preview)** next to **Open as** at the bottom of the window. 4. Select the file that you want to open. 5. Click **Open**. 6. Complete the Text Import window. See "Options in the JMP Text Import Wizard". 7. Click **Import**. **Tip:** The JMP Home window provides a shortcut to the above steps if you recently opened the file. Right-click the file in the Recent Files list and select **Import (Preview)**.

Table 3.2 Opening Text Files on macOS

Automatically Determining Data Arrangement	Manually Specifying Data Arrangement
1. Select **File > Open**. 2. Select the file that you want to open. 3. In the **Open As** field, select **Data (Best Guess)** or **Data (Using Preferences)**. 4. Click **Open**.	1. Select **File > Open**. 2. Select the file that you want to open. In the **Open As** field, select **Data (Using Preview)**. 3. Click **Open**. 4. Complete the Text Import Wizard window. See "Options in the JMP Text Import Wizard". 5. Click **Import**.

Notes:

- On Windows, JMP can open text files in your computer's default text editor. Select **File > Open**, and then select **All Files (*.*)** from the **File name** list. Select the text file, and then select **Use default program to open. Uncheck to open as text.**
- If you cancel the import while JMP is scanning the data, data is imported based on what JMP detected. The data might not appear as you expect. For example, you might see missing values in a column that was incorrectly detected as numeric. If you cancel the import *after* the data is scanned, only the data read up to that point is imported.

For more information about importing text from a Script window, see "Import Text from the Script Editor".

Find Missing Characters in Imported Data

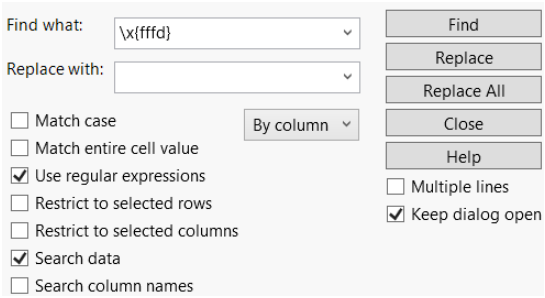
JMP attempts to detect the character encoding in your document. Sometimes the document might use an encoding that JMP does not recognize, or JMP might have chosen the wrong encoding.

Text that JMP does not recognize is replaced with a Unicode character . Find the text by searching for `\x{fffd}`.

1. In the data table, select **Edit > Search > Find**.
2. Type `\x{fffd}` next to **Find what**.
3. Make sure that **Use regular expressions** is selected.
4. Make sure that **Search data** is selected.
5. Click **Find** to find the first instance.

6. Replace all instances with another string.

Figure 3.13 Searching for the Unicode Character



Options in the JMP Text Import Wizard

When you open a text file that JMP supports, JMP can show a preview of the text before opening the file as a data table. This option lets you manually arrange and format the data. For example, you can specify the end-of-line character or strip quotation marks.

JMP detects the file's structure and shows options for importing text with either delimiters or fixed width fields. If JMP chooses the wrong file structure, click the **Delimited fields** or **Fixed width fields** radio button to import the data as the correct format. (For example, the fixed width window might appear when your file is actually delimited.)

The text import wizard options are shown in Figure 3.14 and Figure 3.15.

Figure 3.14 Text Import Wizard for Fixed Width Files

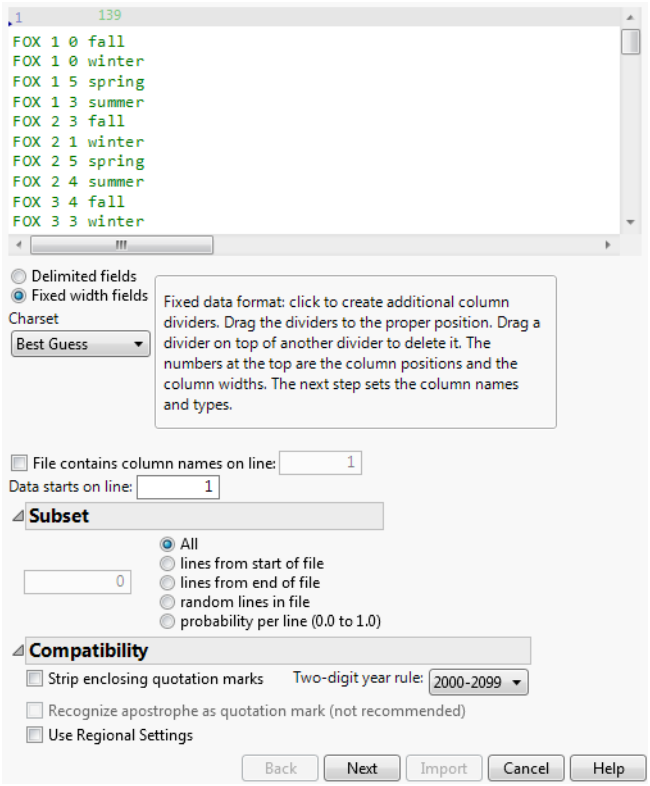
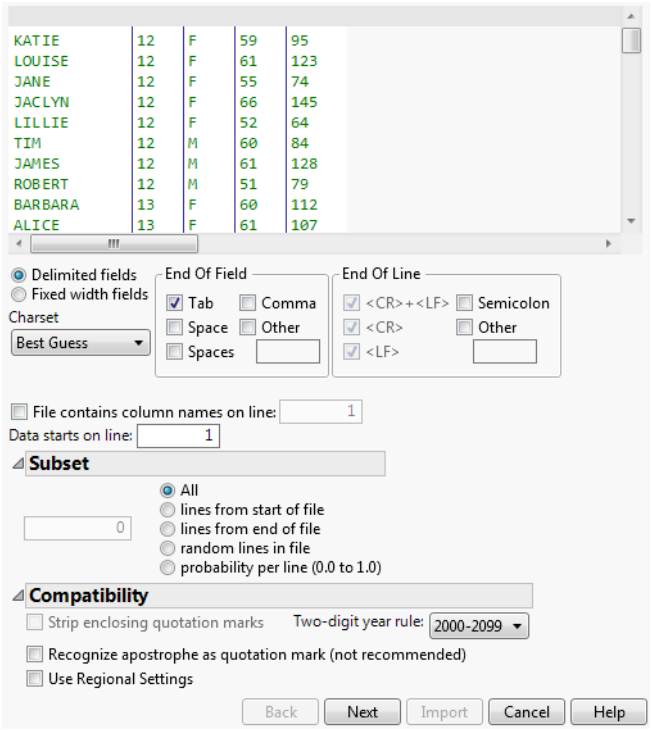


Figure 3.15 Text Import Wizard for Delimited Files



Charset Select the character set used in the imported file, or let JMP detect the character set. If incorrect characters are displayed in the imported file, open the file again and select another character set.

End of Field (Available only in the Delimited Import window.) Select the check boxes beside the character that marks the end of a field. Alternatively, select the check box beside **Other** and enter a character if the appropriate character is not listed.

End of Line (Available only in the Delimited Import window.) Select the check boxes beside the character that marks the end of a line (row). Alternatively, select the check box beside **Other** and enter a character if the appropriate character is not listed. Note that when JMP finds double quotation marks, the delimiter rules change to look for an end double quotation mark. Other text delimiters, including spaces embedded within the quotes, are ignored and treated as part of the text string.

File contains column names on line Tell JMP where to find data to use as column names. For example, if the column names in your text file are on line (row) 3, select this option and type 3 in the check box. Otherwise, JMP uses the data in the first line of the imported file as the column name in the JMP data table or takes the first line as data.

Data starts on line Specify the number of the first line that contains data.

Number of Lines Specify the number of lines (rows) that you want to import.

Strip enclosing quotation marks (Available only on fixed-width imports.) Select this check box when you want JMP to remove quotation marks that enclose data in the text file.

Two-digit year rule Specify how to display dates that have two-digit years. Select the 100-year range in which your dates fall. For example, if the earliest date is 2/2/79, and the year is 1979, select **1970-2069**. If the earliest date is 2/2/12, and the year is 2012, select **2000-2099**. If dates span centuries, you must recode the dates with four-digit years before importing the data.

Recognize apostrophe as quotation mark (not recommended) (Available only in the Delimited Import window). Use this option only if your data comes from a nonstandard source that places apostrophes around data fields rather than quotation marks.

Use Regional Settings Specifies whether the operating system's regional settings are used when importing a text file. If the option is deselected (the default setting), files that use a period for a decimal point and a comma for the value separator import correctly. You must specify the value separator in the Text Data Files import preferences.

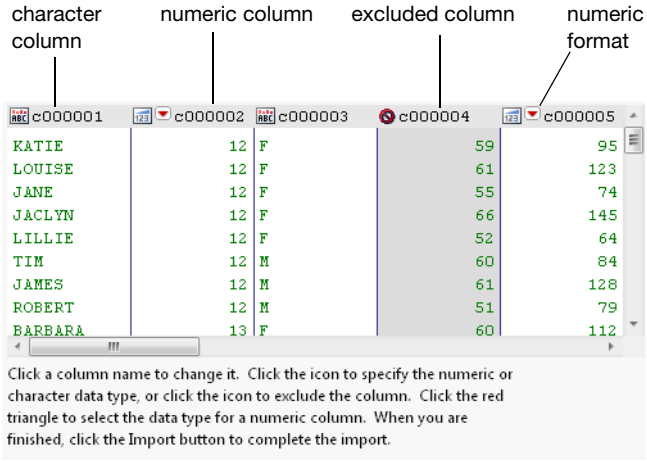
If the file uses a comma for a decimal point, you must also set the following options: the OS regional settings must use a comma for a decimal point (Windows and macOS), or the Windows Specific Display Language preference must be set to a language that uses commas for decimal points.

When you are finished selecting the settings, click **Next**. The next window shows each column's modeling type. To change the default modeling types, do one of the following:

- Click the data type icon to change the data type from numeric ( ) to character ( ). Clicking the icon cycles between the modeling type and exclude ( ). Exclude means that the column is not imported.
- To change a numeric column's data format, select the format from the red triangle menu.
- Click the column heading to modify the text.

The top of the text import window shows a preview of the text file as it appears when imported into a JMP data file. Click the **Import** button to import the data.

Figure 3.16 Text Import Wizard Window with Column Options



Note: To specify the format of dates, select the red triangle next to the column name, select **Date**, and then select the date format. To specify how the dates appear after you've imported the data, right-click the date column in the data table, select **Column Info**, and change the date next to Format.

7. When you are finished setting up the data, click **Import** to complete the text import.

Note: Both the Text Import Wizard and Multiple File Import import columns of numbers as numbers by default. If you have something like part numbers with leading zeros that you want to keep, or part numbers with more than 15 digits, you can force the column to be character. In the Text Import Wizard, you can do this column-by-column (on the second wizard pane, top of the column). In Multiple File Import, you can select the check box that causes *all* columns to be imported as character. (Otherwise leading zeros are not preserved, and digits beyond 15 are lost.) See "Import Multiple Files" for more information about Multiple File Import.

Edit Text Before Importing into JMP

You can open a text file in a Script window, where you edit the text. Then you can import the text as a data table. This feature is helpful when you want to reformat the text before importing it as a data table. For example, you might need to insert the correct delimiters or modify the text.

Another option is opening a JMP add-in definition (.def) file as text and then editing it in a Script window.

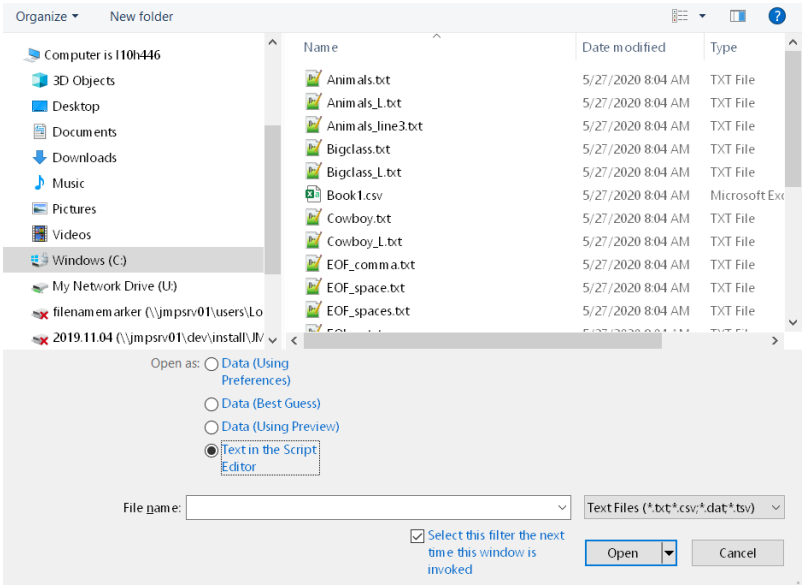
To open a text file in a text editing window (Windows)

Files that you recently opened are listed in the JMP Home window. For most files, right-click the text file and select **Open as Plain Text** to open the file in a text editing window. JMP add-in definition files cannot be opened as plain text from the JMP Home window.

When you are opening the file for the first time, follow these steps:

1. Select **File > Open**.
2. Select **Text Files** from the list next to **File name**.

Figure 3.17 Select Text Files



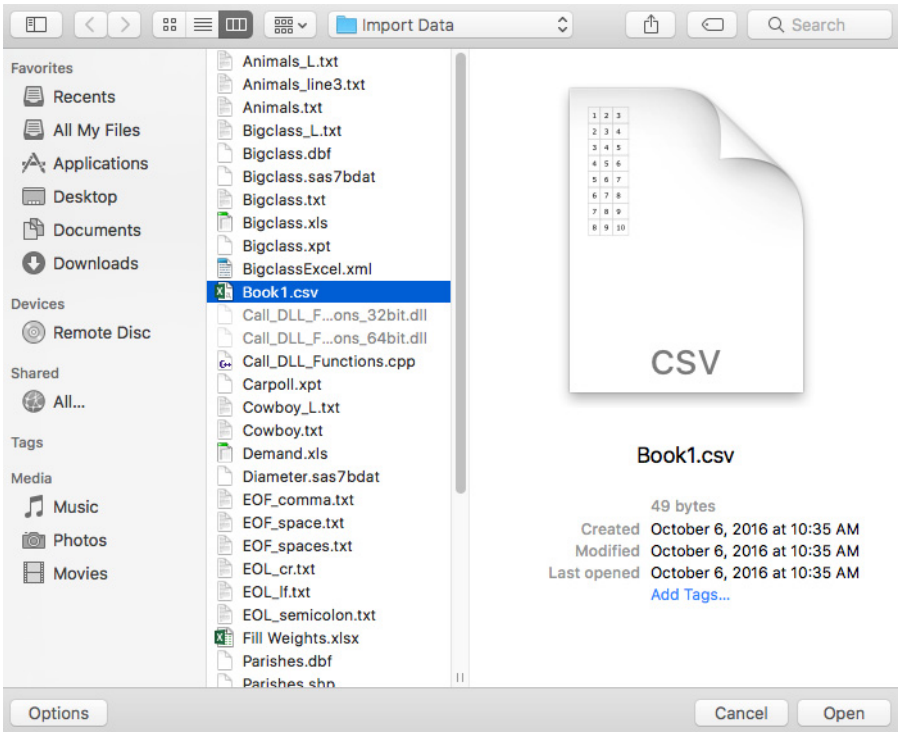
3. (Optional) To set the default option file type to **Text Files**, select the check box beside **Select this filter the next time this window is invoked**.
4. Select the file.
5. Select **Plain text into Script window** next to **Open as**.
6. Click **Open**.

   The text appears in a Script window.

To open a text file in a text editing window (macOS)

1. Select **File > Open**.

Figure 3.18 Opening a Text Document on macOS



2. Select the file.
3. Click **Options**.
4. Select **Text in the Script Editor** from the **Open As** list.
5. Click **Open**.

   The text appears in a Script window.

For more information about converting the text to a data table, follow step 3 in "Import Text from the Script Editor".

Import Text from the Script Editor

You can import text from the Script Editor as a data table. The text can be in a table format (for example, from a Microsoft Word document or Web page) or in plain text format. This feature is helpful when you want to reformat the text before importing it as a data table. For example, you might need to insert the correct delimiters or modify the text.

JMP uses the import settings in the preferences to determine how to structure and format the text. Some options include removing quotation marks around text and specifying the rows that contain column headings and data. See "Preferences for Importing and Exporting Text Files".

Note: You can also import an entire web page as a data table. See "Import Data from the Internet or a Remote Computer".

This section describes how to import text that you paste into the Script Editor. For more information about opening a text file in the Script Editor, see "Edit Text Before Importing into JMP".

To import text from the Script window

1. Open a new Script window by selecting **File > New > Script** (Windows) or **File > New > New Script** (macOS).
2. Copy and paste the text into the Script Editor.
3. Do one of the following:
   - To import all text from the Script Editor, select **Edit > Import as Data**.
   - To import specific text, select the text, and then select **Edit > Import as Data**.

   The text is imported into a JMP data table.

Note: To preview text that you import from the Script Editor, press Shift before you select File > Import as Data.

Import Multiple Files

Multiple File Import (MFI) imports CSV data and unstructured text files from one directory and stack them in a data table (or to import them in separate data tables). You can save the import settings as a script and then run the script when the data is updated in the future.

Multiple File Import supports the following file types:

- XLSX
- CSV
- TSV
- DAT
- TXT
- SAS7BDAT

- XML
- JMP
- JSON
- PNG
- JPG
- GIF

Multiple File Import can also import only one file. Table 3.3 can help you decide whether to use the standard Text Import, Multiple File Import, or JSL.

Table 3.3 Choosing Between the Text Import Wizard and MFI

Import Characteristics	One File	Multiple Files
CSV, TSV, DAT	Text Import has a preview wizard to help name the columns, pick date formats, and subset the data.	MFI can stack similar files automatically. You can change formatting later.
Unstructured text	MFI, possibly in JSL with `Load Text File()`.	MFI
Large files over 100MB	You can use the Source script from a previous Text Import to speed up the next import.	MFI imports in parallel and might be much faster than Text Import.
Unsure about format	Text Import might be able to guess. The preview wizard lets you see the results before the text is imported.	MFI does not guess. You specify the tab delimiters. There is no preview, but select **Keep dialog open**, and it is easy to try different import settings.
Pictures	Try the JSL `Open()` function.	MFI understands PNG, JPG, and GIF files. You'll get a table with a picture column.

Table 3.3 Choosing Between the Text Import Wizard and MFI *(Continued)*

Import Characteristics	One File	Multiple Files
JMP, XLSX, and JSON	Try the JSL `Open()` function.	MFI opens JMP, XLSX, and JSON files using JSL `Open()` and will stack them if requested. MFI does not keep the table and column scripts when it opens JMP tables, so you might need to use `Files In Directory()` and `Open()` if you need all of those scripts.

To import multiple files, follow these steps:

1. Place the files that you want to import in the same directory.

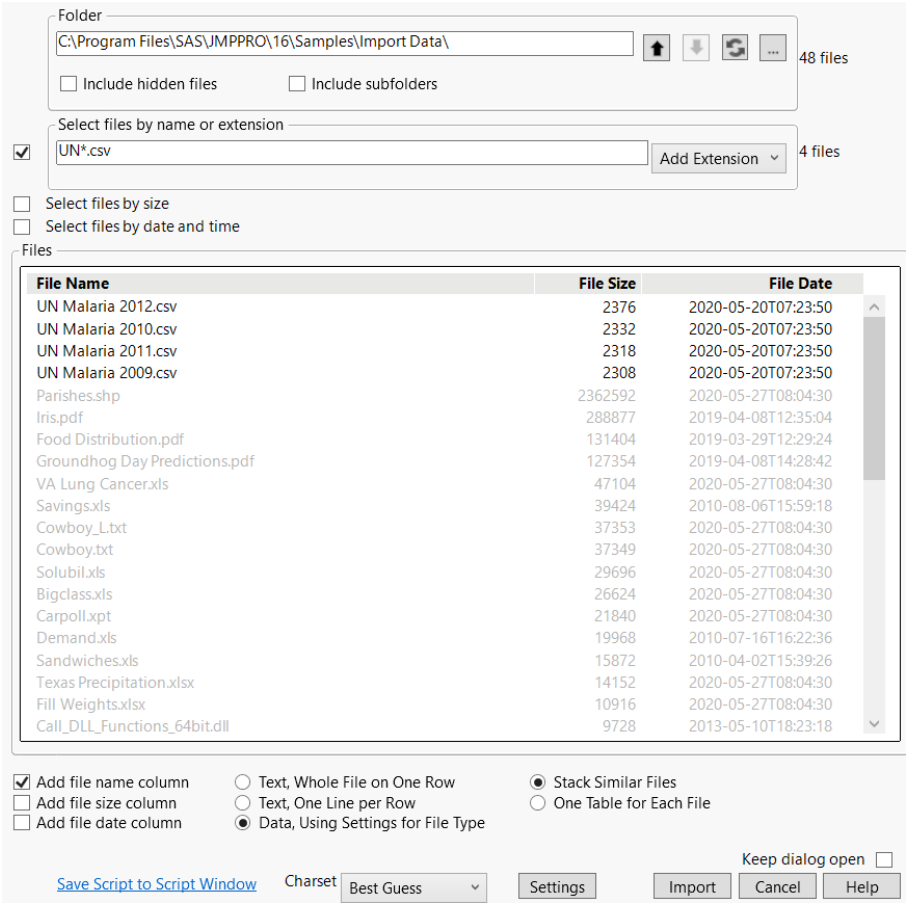
   This example uses import data found in the JMP Samples/Import Data folder.
2. Select **File > Import Multiple Files** (Windows) or **File > Open Multiple** (macOS).
3. Browse to select the JMP Samples/Import Data folder.
4. In the **Select Files by name or extension** box, type UN*.csv and press Enter.

   In this example, the files have the UN prefix and are CSV files.

   The Files list updates to show which files have been selected.

   Specify multiple file names and extensions by separating them with semicolons, or click **Add Extension** and select an extension.

Figure 3.19 Selected CSV Files



Notice that the Refresh button flashes if the files have changed in the directory since the last refresh.

5. Specify the columns that you want added to the data table.

   By default, no extra columns are added. You might like a File Name column to show the names of the imported files. Select **Add file name column**.

   Note: The Text Data Files preferences do not determine which options are selected in the CSV settings, although the settings are similar.

6. Keep **Stack Similar Files** selected because these CSV files have the same columns.
7. Click **Import**.

   JMP applies the file name, file size, and file date and time filters to stack the files. Data with columns that contain the same column headings are imported in the same data table.

If the columns do not match, the mismatched columns are imported into a separate data table.

Figure 3.20 Imported CSV Data

	Country or Area	Year	Number of Incidents	File Name
1	Afghanistan	2009	64880	UN Malaria 2009.csv
2	Algeria	2009	4	UN Malaria 2009.csv
3	Angola	2009	1573422	UN Malaria 2009.csv
4	Argentina	2009	86	UN Malaria 2009.csv
5	Azerbaijan	2009	78	UN Malaria 2009.csv
6	Bangladesh	2009	63873	UN Malaria 2009.csv
7	Belize	2009	256	UN Malaria 2009.csv
8	Benin	2009	889597	UN Malaria 2009.csv
9	Bhutan	2009	972	UN Malaria 2009.csv
10	Bolivia (Plurinatio...	2009	9743	UN Malaria 2009.csv
11	Botswana	2009	1024	UN Malaria 2009.csv
12	Brazil	2009	309316	UN Malaria 2009.csv
13	Burkina Faso	2009	182527	UN Malaria 2009.csv

Notes:

- To import hidden files, select **Include hidden files**.
- To import files that are in subdirectories, select **Include subfolders**.
- To import files in a certain size range, in the **Select files by size** box, type the lower and upper file size limits.

  Consider specifying the limits if you don't want to import large files over a certain limit.

  Specify file sizes in KB (kilobytes). One KB equals 1,000 bytes.
- To import files that were modified during a certain date and time range, in the **Select files by date and time** box, click the calendar button to specify the range.
- (Unstructured text files) Select **Text, Whole File on One Row** or **Text, One Line Per Row** based on how the file is structured.

  Notes:

  – These two options are not supported for XLSX, JSON, and JMP files.

  – When you select Text, Whole File on One Row, the import includes the whole file and does not skip lines before the first data line.
- JMP creates a data table that lists the files that could not be imported (for example, SAS datasets).

Import Structured Data with Different Quoting

Sometimes export filters place something other than quotation marks around fields. You specify the delimiter in the CSV settings.

Text in a CSV file might contain escape sequences such as a backslash instead of doubling the quotation mark. You also specify the delimiter in the CSV settings.

Tips:

- **Keep dialog open** is off by default, but you might turn it on while testing. This enables you to import files to see what the data looks like, edit the source files if necessary, and reimport them with the same settings.
- When you import files that are not supported, a data table called "@ Unused Files" is created. The file names are listed in the Files table script. Right-click the script and select Edit. To output the file names of unsupported files in this data table, select **Add file name column** on the Multiple File Import window.
- Right-click the Files list and select Sort by Column to change the way the Files list is sorted.
- The Import button is unavailable if no files match the filters that you selected.
- JMP automatically changes the CSV settings when you select either of the two options for Text Explorer-style data (the bottom center radio buttons). A delimiter is not specified, which loads all data in one column. JMP also changes the CSV settings to not use an end-of-line character, which loads the entire file in one row. You can also change the CSV settings. If you try the manual approach, be sure to set No Headers and First Data Line to 1.

Import PDF Files

The PDF Import Wizard enables you to preview a PDF file, adjust the import settings, and view the results in the data table preview before you import the data. JMP attempts to auto-detect the data structure in the PDF file, or you can customize the structure.

Note: A PDF file consists of tables that contain data. In the PDF preview, the tables are outlined and selected on each page.

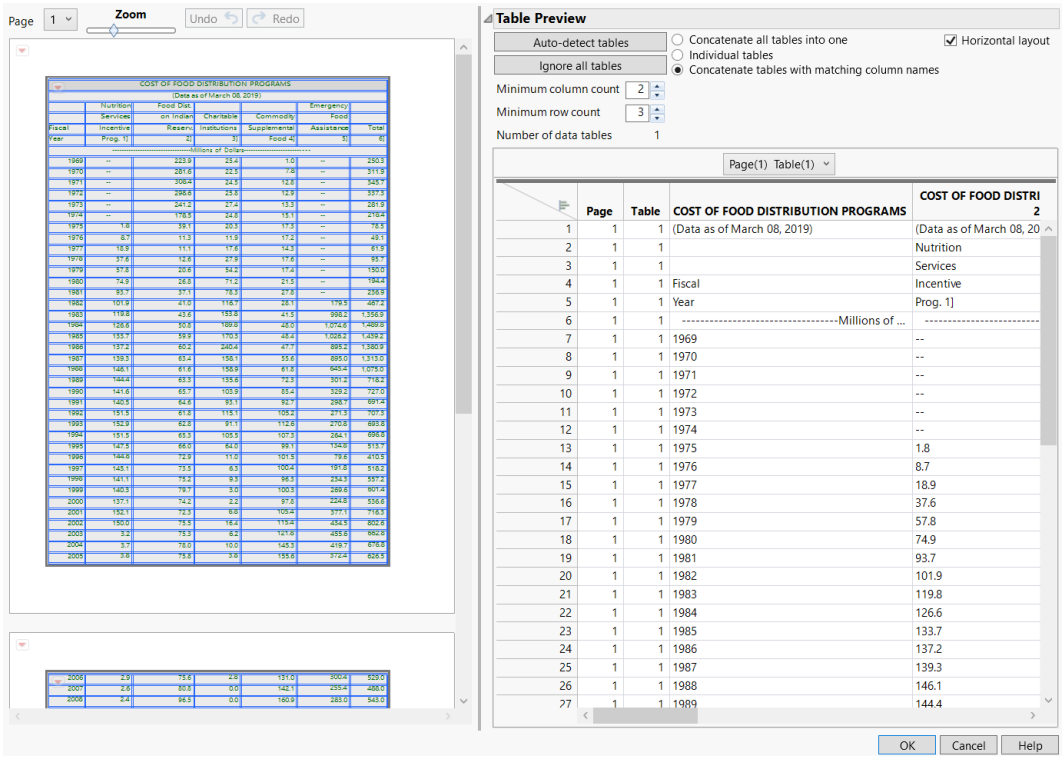
- "Concatenate Tables into One Data Table"
- "Concatenate Tables by Column Headers"
- "Import Tables that JMP Does Not Auto-Detect"
- "PDF Preview Red Triangle Options"
- "Data Table Preview Options"
- "PDF Preview Right-Click Options"

Concatenate Tables into One Data Table

1. Select **File** > **Open** and select **PDF Files** from the list next to File name.
2. Navigate to the JMP Samples/Import Data folder and double-click Food Distribution.pdf.
3. In the Import PDF Wizard, a preview of the PDF file is shown on the left. A preview of the data in a data table is shown on the right.

   The PDF file consists of one table on each page.

Figure 3.21 Initial PDF File and Data Table Preview



Note the following:

- There are seven rows at the top of the PDF that you don't want to import.
- You also don't want the footnotes in the last row of the PDF file.

4. In the corner of the first table, click the red triangle and select **Number of rows to use as header** > **Specify rows**.
5. Type 7 and click **OK**.

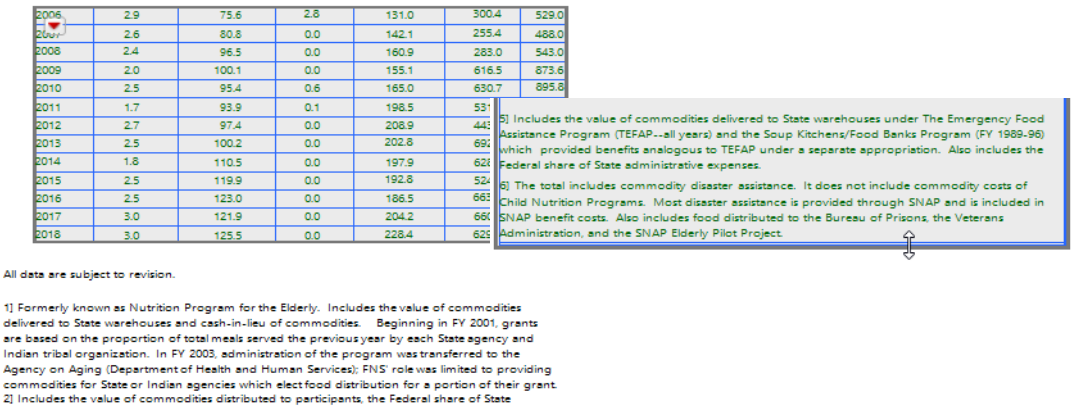
   In the PDF file preview, the first seven rows are selected.

   In the data table preview, the data begin on row 1.

6. In the PDF file preview, scroll down to page 2.
7. Drag the bottom of the PDF table upward until the footnotes are no longer selected (Figure 3.22).

Note: You can also simply delete the row from the final data table.

Figure 3.22 Dragging the Table to Remove Data



2006	2.9	75.6	2.8	131.0	300.4	529.0
2007	2.6	80.8	0.0	142.1	255.4	488.0
2008	2.4	96.5	0.0	160.9	283.0	543.0
2009	2.0	100.1	0.0	155.1	616.5	873.6
2010	2.5	95.4	0.6	165.0	630.7	895.8
2011	1.7	93.9	0.1	198.5	53	
2012	2.7	97.4	0.0	208.9	44	
2013	2.5	100.2	0.0	202.8	69	
2014	1.8	110.5	0.0	197.9	62	
2015	2.5	119.9	0.0	192.8	52	
2016	2.5	123.0	0.0	186.5	66	
2017	3.0	121.9	0.0	204.2	66	
2018	3.0	125.5	0.0	228.4	62	

5] Includes the value of commodities delivered to State warehouses under The Emergency Food Assistance Program (TEFAP--all years) and the Soup Kitchens/Food Banks Program (FY 1989-96) which provided benefits analogous to TEFAP under a separate appropriation. Also includes the Federal share of State administrative expenses.

6] The total includes commodity disaster assistance. It does not include commodity costs of Child Nutrition Programs. Most disaster assistance is provided through SNAP and is included in SNAP benefit costs. Also includes food distributed to the Bureau of Prisons, the Veterans Administration, and the SNAP Elderly Pilot Project.

All data are subject to revision.

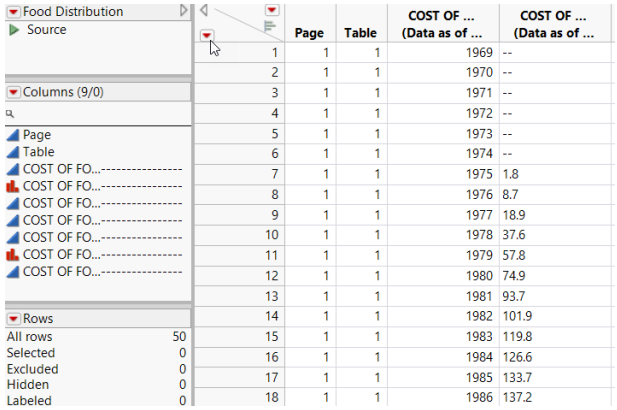
1] Formerly known as Nutrition Program for the Elderly. Includes the value of commodities delivered to State warehouses and cash-in-lieu of commodities. Beginning in FY 2001, grants are based on the proportion of total meals served the previous year by each State agency and Indian tribal organization. In FY 2003, administration of the program was transferred to the Agency on Aging (Department of Health and Human Services); FNS' role was limited to providing commodities for State or Indian agencies which elect food distribution for a portion of their grant.
2] Includes the value of commodities distributed to participants, the Federal share of State

8. In the Table report on the right, select **Concatenate all tables into one**.

   When you concatenate data tables, you combine rows from two or more data tables into one data table.
9. At the bottom, click **OK**.

   The data is imported into a data table.

Figure 3.23 Final Data Table (Partial View)



Food Distribution
Source

Columns (9/0)

Page
Table
COST OF FO...----------
COST OF FO...----------
COST OF FO...----------
COST OF FO...----------
COST OF FO...----------
COST OF FO...----------
COST OF FO...----------

Rows
All rows 50
Selected 0
Excluded 0
Hidden 0
Labeled 0

	Page	Table	COST OF ... (Data as of ...	COST OF ... (Data as of ...
1	1	1	1969	--
2	1	1	1970	--
3	1	1	1971	--
4	1	1	1972	--
5	1	1	1973	--
6	1	1	1974	--
7	1	1	1975	1.8
8	1	1	1976	8.7
9	1	1	1977	18.9
10	1	1	1978	37.6
11	1	1	1979	57.8
12	1	1	1980	74.9
13	1	1	1981	93.7
14	1	1	1982	101.9
15	1	1	1983	119.8
16	1	1	1984	126.6
17	1	1	1985	133.7
18	1	1	1986	137.2

Concatenate Tables by Column Headers

When a PDF file contains tabular data with the same column headers on each page, you can concatenate the tables based on the column headers.

1. Select **File** > **Open** and select **PDF Files** from the list next to File name.
2. Navigate to the JMP Samples/Import Data folder and double-click Iris.pdf.

   Scroll through the PDF preview and notice that the column headers are the same on each page.
3. In the Table report on the right, make sure that **Concatenate tables with matching column names** is selected.
4. At the bottom, click **OK**.

Figure 3.24 Final Data Table (Partial View)

	Page	Table	Sepal length	Sepal width	Petal length	Petal width	Species
1	1	1	5.1	3.5	1.4	0.2	setosa
2	1	1	4.9	3	1.4	0.2	setosa
3	1	1	4.7	3.2	1.3	0.2	setosa
4	1	1	4.6	3.1	1.5	0.2	setosa
5	1	1	5	3.6	1.4	0.2	setosa
6	1	1	5.4	3.9	1.7	0.4	setosa
7	1	1	4.6	3.4	1.4	0.3	setosa
8	1	1	5	3.4	1.5	0.2	setosa
9	1	1	4.4	2.9	1.4	0.2	setosa
10	1	1	4.9	3.1	1.5	0.1	setosa
11	1	1	5.4	3.7	1.5	0.2	setosa
12	1	1	4.8	3.4	1.6	0.2	setosa
13	1	1	4.8	3	1.4	0.1	setosa
14	1	1	4.3	3	1.1	0.1	setosa
15	1	1	5.8	4	1.2	0.2	setosa
16	1	1	5.7	4.4	1.5	0.4	setosa
17	1	1	5.4	3.9	1.3	0.4	setosa
18	1	1	5.1	3.5	1.4	0.3	setosa

Import Tables that JMP Does Not Auto-Detect

Sometimes tables in PDF files cannot be auto-detected. In these cases, you can drag boxes around the columns to group and divide tables.

Figure 3.25 Partial PDF File Preview

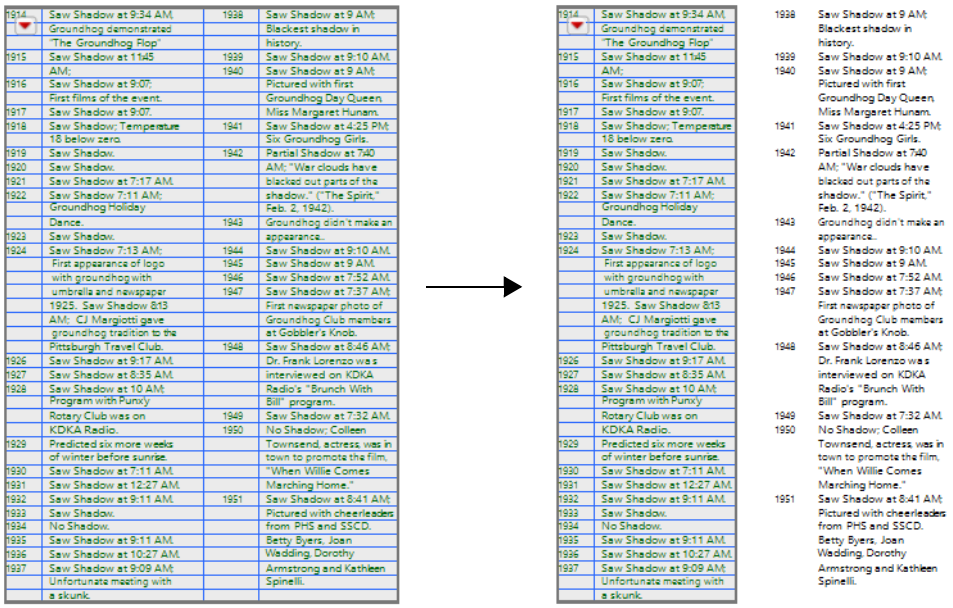
Notice that rows in the first column were auto-detected as three tables.

Drag the top edge of the third table up until it surrounds the two columns.

Figure 3.26 Creating One Table

In the first table shown in Figure 3.27, drag the far right border to the left until only columns one and two are in the same table,and columns 3 and 4 are outside the table (Figure 3.27).

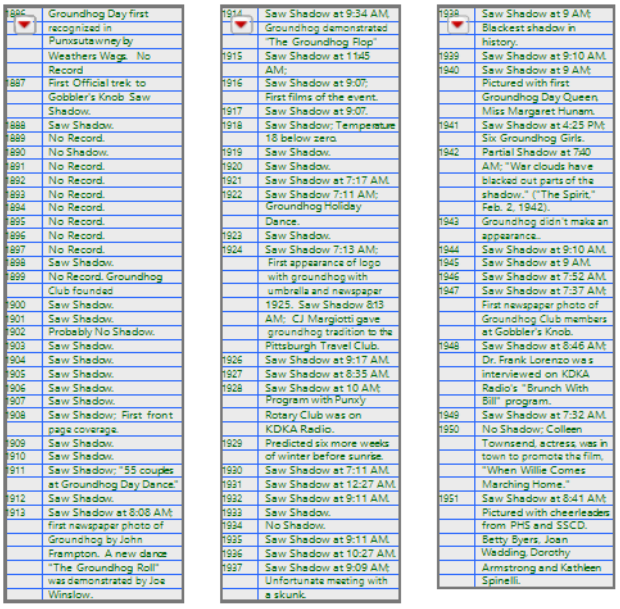
Figure 3.27 Splitting Up a Table



1914	Saw Shadow at 9:34 AM;
	Groundhog demonstrated
	"The Groundhog Flop"
1915	Saw Shadow at 11:45
	AM;
1916	Saw Shadow at 9:07;
	First films of the event.
1917	Saw Shadow at 9:07.
1918	Saw Shadow; Temperature
	18 below zero.
1919	Saw Shadow.
1920	Saw Shadow.
1921	Saw Shadow at 7:17 AM.
1922	Saw Shadow 7:11 AM;
	Groundhog Holiday
	Dance.
1923	Saw Shadow.
1924	Saw Shadow 7:13 AM;
	First appearance of logo
	with groundhog with
	umbrella and newspaper
	1925. Saw Shadow 8:13
	AM; CJ Margiotti gave
	groundhog tradition to the
	Pittsburgh Travel Club.
1926	Saw Shadow at 9:17 AM.
1927	Saw Shadow at 8:35 AM.
1928	Saw Shadow at 10 AM;
	Program with Punxy
	Rotary Club was on
	KDKA Radio.
1929	Predicted six more weeks
	of winter before sunrise.
1930	Saw Shadow at 7:11 AM.
1931	Saw Shadow at 12:27 AM.
1932	Saw Shadow at 9:11 AM.
1933	Saw Shadow.
1934	No Shadow.
1935	Saw Shadow at 9:11 AM.
1936	Saw Shadow at 10:27 AM.
1937	Saw Shadow at 9:09 AM;
	Unfortunate meeting with
	a skunk.
1938	Saw Shadow at 9 AM;
	Blackest shadow in
	history.
1939	Saw Shadow at 9:10 AM.
1940	Saw Shadow at 9 AM;
	Pictured with first
	Groundhog Day Queen
	Miss Margaret Hunam.
1941	Saw Shadow at 4:25 PM;
	Six Groundhog Girls.
1942	Partial Shadow at 7:40
	AM; "War clouds have
	blacked out parts of the
	shadow." ("The Spirit,"
	Feb. 2, 1942).
1943	Groundhog didn't make an
	appearance.
1944	Saw Shadow at 9:10 AM.
1945	Saw Shadow at 9 AM.
1946	Saw Shadow at 7:52 AM.
1947	Saw Shadow at 7:37 AM;
	First newspaper photo of
	Groundhog Club members
	at Gobbler's Knob.
1948	Saw Shadow at 8:46 AM;
	Dr. Frank Lorenzo was
	interviewed on KDKA
	Radio's "Brunch With
	Bill" program.
1949	Saw Shadow at 7:32 AM.
1950	No Shadow; Colleen
	Townsend, actress, was in
	town to promote the film,
	"When Willie Comes
	Marching Home."
1951	Saw Shadow at 8:41 AM;
	Pictured with cheerleaders
	from PHS and SSCD.
	Betty Byers, Joan
	Wadding, Dorothy
	Armstrong and Kathleen
	Spinelli.

Drag a box around the remaining two columns to put them in a table (Figure 3.28).

Figure 3.28 Final Tables on Page One



1886	Groundhog Day first
	recognized in
	Punxsutawney by
	Weathers Wags. No
	Record.
1887	First Official trek to
	Gobbler's Knob. Saw
	Shadow.
1888	Saw Shadow.
1889	No Record.
1890	No Shadow.
1891	No Record.
1892	No Record.
1893	No Record.
1894	No Record.
1895	No Record.
1896	No Record.
1897	No Record.
1898	Saw Shadow.
1899	No Record. Groundhog
	Club founded
1900	Saw Shadow.
1901	Saw Shadow.
1902	Probably No Shadow.
1903	Saw Shadow.
1904	Saw Shadow.
1905	Saw Shadow.
1906	Saw Shadow.
1907	Saw Shadow.
1908	Saw Shadow; First front
	page coverage.
1909	Saw Shadow.
1910	Saw Shadow.
1911	Saw Shadow; "55 couples
	at Groundhog Day Dance."
1912	Saw Shadow.
1913	Saw Shadow at 8:08 AM;
	first newspaper photo of
	Groundhog by John
	Frampton. A new dance
	"The Groundhog Roll"
	was demonstrated by Joe
	Winslow.

1914	Saw Shadow at 9:34 AM;
	Groundhog demonstrated
	"The Groundhog Flop"
1915	Saw Shadow at 11:45
	AM;
1916	Saw Shadow at 9:07;
	First films of the event.
1917	Saw Shadow at 9:07.
1918	Saw Shadow; Temperature
	18 below zero.
1919	Saw Shadow.
1920	Saw Shadow.
1921	Saw Shadow at 7:17 AM.
1922	Saw Shadow 7:11 AM;
	Groundhog Holiday
	Dance.
1923	Saw Shadow.
1924	Saw Shadow 7:13 AM;
	First appearance of logo
	with groundhog with
	umbrella and newspaper
	1925. Saw Shadow 8:13
	AM; CJ Margiotti gave
	groundhog tradition to the
	Pittsburgh Travel Club.
1926	Saw Shadow at 9:17 AM.
1927	Saw Shadow at 8:35 AM.
1928	Saw Shadow at 10 AM;
	Program with Punxy
	Rotary Club was on
	KDKA Radio.
1929	Predicted six more weeks
	of winter before sunrise.
1930	Saw Shadow at 7:11 AM.
1931	Saw Shadow at 12:27 AM.
1932	Saw Shadow at 9:11 AM.
1933	Saw Shadow.
1934	No Shadow.
1935	Saw Shadow at 9:11 AM.
1936	Saw Shadow at 10:27 AM.
1937	Saw Shadow at 9:09 AM;
	Unfortunate meeting with
	a skunk.

1938	Saw Shadow at 9 AM;
	Blackest shadow in
	history.
1939	Saw Shadow at 9:10 AM.
1940	Saw Shadow at 9 AM;
	Pictured with first
	Groundhog Day Queen
	Miss Margaret Hunam.
1941	Saw Shadow at 4:25 PM;
	Six Groundhog Girls.
1942	Partial Shadow at 7:40
	AM; "War clouds have
	blacked out parts of the
	shadow." ("The Spirit,"
	Feb. 2, 1942).
1943	Groundhog didn't make an
	appearance.
1944	Saw Shadow at 9:10 AM.
1945	Saw Shadow at 9 AM.
1946	Saw Shadow at 7:52 AM.
1947	Saw Shadow at 7:37 AM;
	First newspaper photo of
	Groundhog Club members
	at Gobbler's Knob.
1948	Saw Shadow at 8:46 AM;
	Dr. Frank Lorenzo was
	interviewed on KDKA
	Radio's "Brunch With
	Bill" program.
1949	Saw Shadow at 7:32 AM.
1950	No Shadow; Colleen
	Townsend, actress, was in
	town to promote the film,
	"When Willie Comes
	Marching Home."
1951	Saw Shadow at 8:41 AM;
	Pictured with cheerleaders
	from PHS and SSCD.
	Betty Byers, Joan
	Wadding, Dorothy
	Armstrong and Kathleen
	Spinelli.

Note: Groundhog Day Predictions.pdf in the Samples/Import Data folder shows a cleaned-up version of the data that is auto-detected in the PDF Import Wizard.

PDF Preview Red Triangle Options

The section describes the red triangle options on the PDF preview pane.

Global Red Triangle

The red triangle in the upper left corner of the PDF preview window contains two options:

Auto-detect tables on this page If you previously divided or combined tables, this option reverts the changes you made so that the tables are auto-detected again.

Ignore tables on this page Removes the tables from the page. That data is not imported.

Red Triangle on Each Table

Ignore this table Removes the table. The data is not imported.

Number of rows to use as header Specifies the number of header rows in the PDF file that will not be imported as data.

Auto-detect tables on this page If you previously divided or combined tables, this option reverts the changes you made so that the tables are auto-detected again.

Ignore tables on this page Removes the tables from the page. That data is not imported.

Data Table Preview Options

This section describes options that are on the data table preview pane.

Auto-detect tables If you previously divided or combined tables, this option reverts the changes you made so that the tables are auto-detected again.

Ignore all tables Removes the tables from the PDF file. The data is not imported.

Minimum column count Avoids importing tables that have fewer columns than specified.

Minimum row count Avoids importing tables that have fewer rows than specified.

Concatenate all tables into one Combines rows from two or more data tables into one data table.

Individual tables Imports the data into multiple tables.

Concatenate tables with matching column names Combines rows from two or more data tables that have matching column names into one data table.

PDF Preview Right-Click Options

See "Red Triangle on Each Table" for descriptions of most PDF preview right-click options. The following options are also available:

Add column divider Creates a column divider at the selected point. Choose this option to create a new column in the data table (for example, if the columns weren't auto-detected).

Remove column divider Deletes the column border at the selected point.

Add row divider Creates a row border at the selected point. Choose this option to create a new row in the data table.

Remove row divider Deletes the column border at the selected point (for example, if the rows weren't auto-detected).

Import XML and JSON Files

Preview and Import XML and JSON Data

The XML Import Wizard imports two types of nested text data: XML and JSON files. Both kinds of files contain structured text that can be nested to represent hierarchical relationships. This wizard helps you pick out the elements of the hierarchy that hold values and elements that determine rows to build a data table.

A useful way to imagine a nested text file: the document is a book of short stories. Some of the short stories might have chapters, others only have paragraphs. Chapters have paragraphs. Paragraphs have sentences. Here's an example of the full tree for such a book.

```
<book>
  <story name="car poll">
    <wheels>4</wheels>
    <para><name>chev</name></para>
    <para><name>ford</name></para>
    <para><name>volk</name></para>
  </story>
  <story name="big class">
    <para><name>ralph</name><height>6</height></para>
    <para><name>billy</name><height>5</height></para>
  </story>
  <story name="cheese" location="NC">
```

```
    <chapter name="american">
      <para>1<price>2.00</price><quantity>2</quantity></para>
      <para>2<price>2.10</price><quantity>4</quantity></para>
      <para>3<price>2.20</price><quantity>3</quantity></para>
    </chapter>
    <chapter name="swiss">
      <para>1<price>3.00</price><quantity>3</quantity></para>
      <para>2<price>3.10</price><quantity>2</quantity></para>
    </chapter>
  </story>
</book>
```

This XML document (the book) contains three short stories; each short story is a different data table. In the following example, you choose to import only one of them at a time. This particular XML file uses paragraphs to represent rows in a table in all of the tables. Arranging content by paragraph is not required. If you wanted to import the cheese table, which is slightly more complicated than the other two, you'd probably want a variety column, containing "american" or "swiss" for each row. You would also want price and quantity columns. And you might want a column that doesn't have a clearly defined name for the 1, 2, 3, 1, 2 values of <para>. You'd expect five rows. Finally, you might want a table variable or a column for the location NC value.

A JSON file consists of name and value pairs that are imported as column headings and data. In the following example, the first name in each string is appended to "Grocery Store Purchases" and turned into column headings, as in "Grocery Store Purchases.Item". "avocado" is the value in the first cell of the table. The next column is named "Grocery Store Purchases.Category", and "Produce" is the value in the first cell of the second column.

```
{"Grocery Store Purchases":[
    {"Item":"avocado", "Category":"Produce"},
    {"Item":"bread", "Category":"Bakery"},
    {"Item":"chocolate", "Category":"candy"}
]}
```

To import an XML or JSON file in the XML Import Wizard

1. Select **File > Open**.
2. Browse to JMP's Samples/Import Data folder.
3. On Windows, set the file type to **XML Data Files**.
4. Select Book.xml and click **Open**.

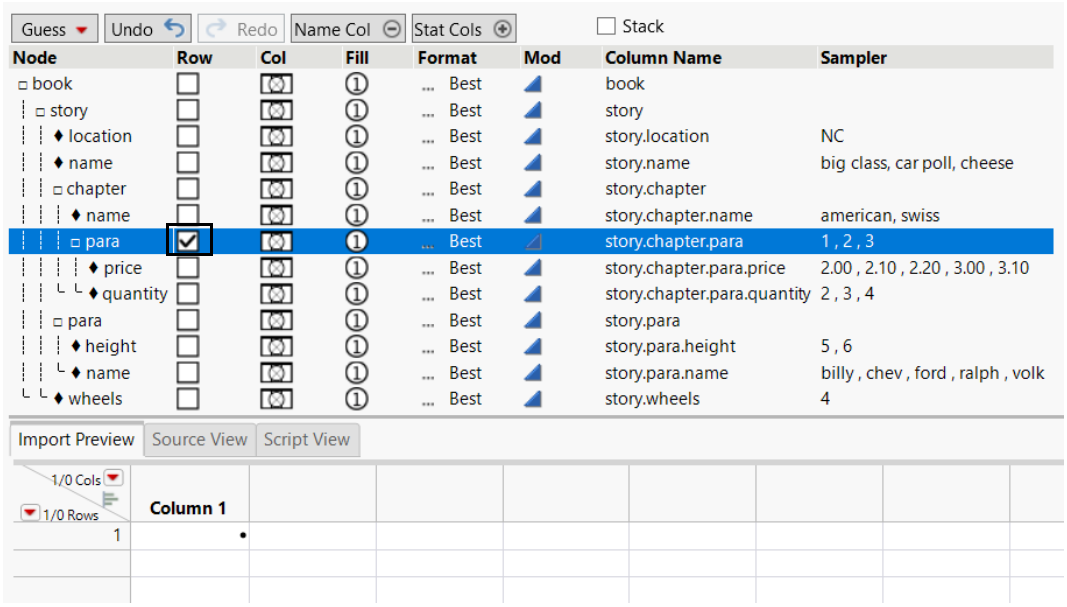
   The file opens in the XML Import Wizard.

Note: "Data (Using Preview)" is selected on the Open Data File window so that the XML file opens in the wizard automatically. You can also import XML and JSON files directly. See "Import XML Files Directly".

Determine How the Data is Imported

1. In the Row column, select the element that should create a new row in the data table. In this example, the row is created when `</para>` is processed, and the columns are written for that row with their current values.

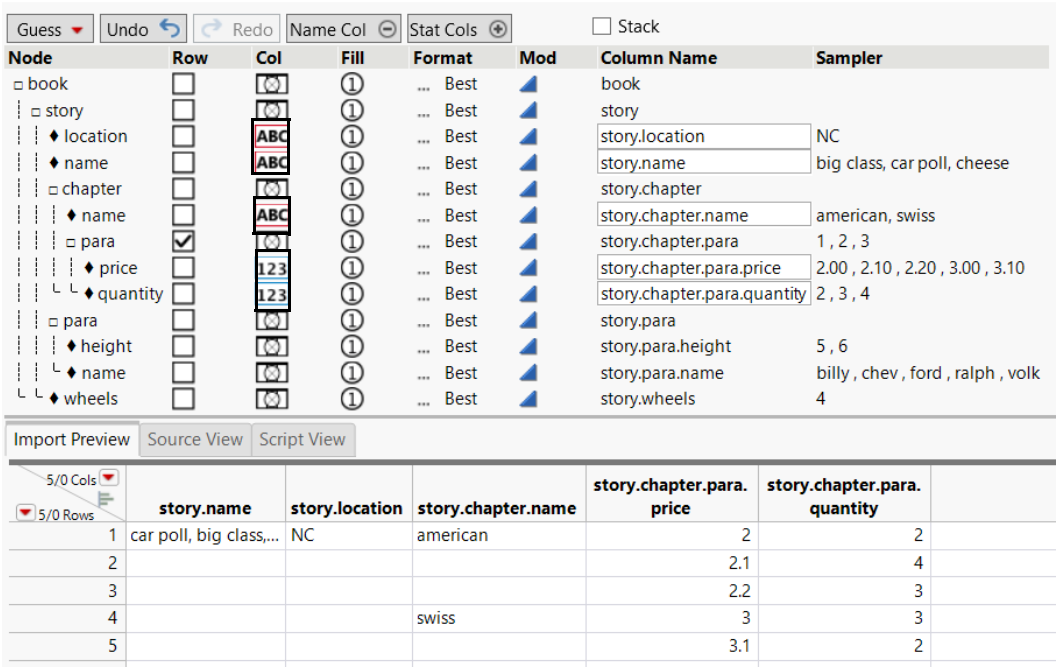
Figure 3.29 Row Element Selected in the Row Column



2. In the Col column, select the data type for that column next to the Col circle.
   - location: **Character**
   - name: **Character** (both instances)
   - price: **Numeric**
   - quantity: **Numeric**

Tip: The values in the Sampler column give an idea of what the data type should be.

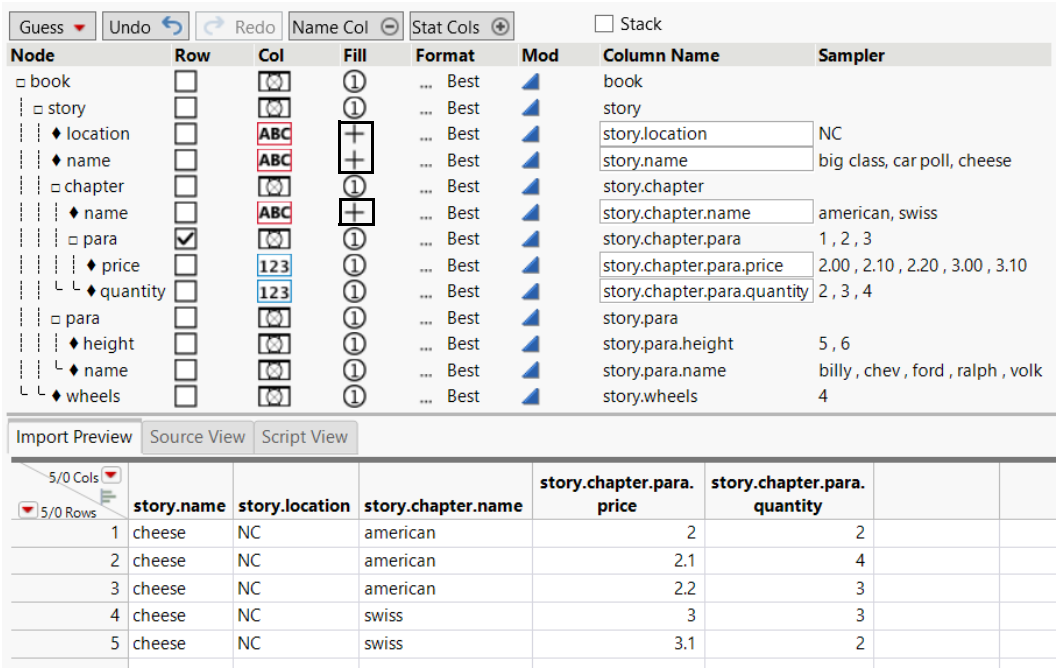
Figure 3.30 Data Types Assigned in the Col Column



3. In the Fill column, specify which values fill the cells.

   For location and the both instances of name, click the Fill circle and select **+book/story**. This setting fills cells with the values until the node `<book>/<story>` begins again.

Figure 3.31 Fill Values Selected in the Fill Column

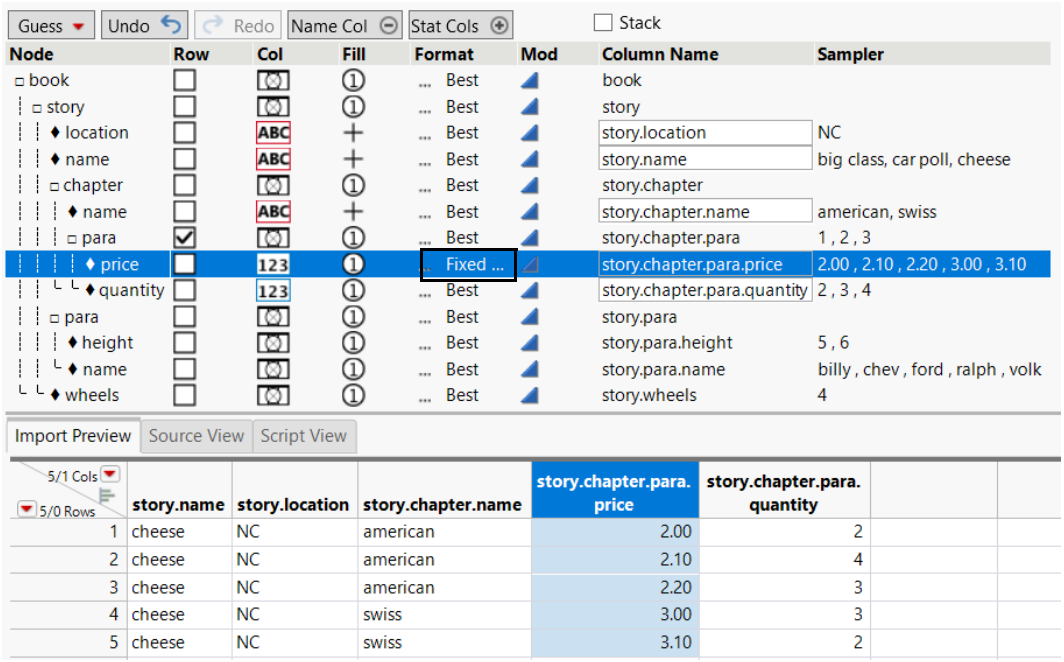


When adding each row to the data table, JMP decides what to do on the next row if no new value is found for a column. For example, the chapter name is not specified for each type of cheese, Missing values follow "american" and "swiss".

- The Use Once option means use the value once and forget it; the second, third, and fifth rows would be missing if the data is not given again.
- The Fill Forever option means never forget the value but replace it with new values when found.
- You select the +book/story option in this example. This option indicates that, when the opening `<book>/<story>` elements are found, JMP fills in the cells until the `<book>/<story>` node begins again. The `-<element>` option refers to the closing tag of the element, which will be the point at which the node ends.

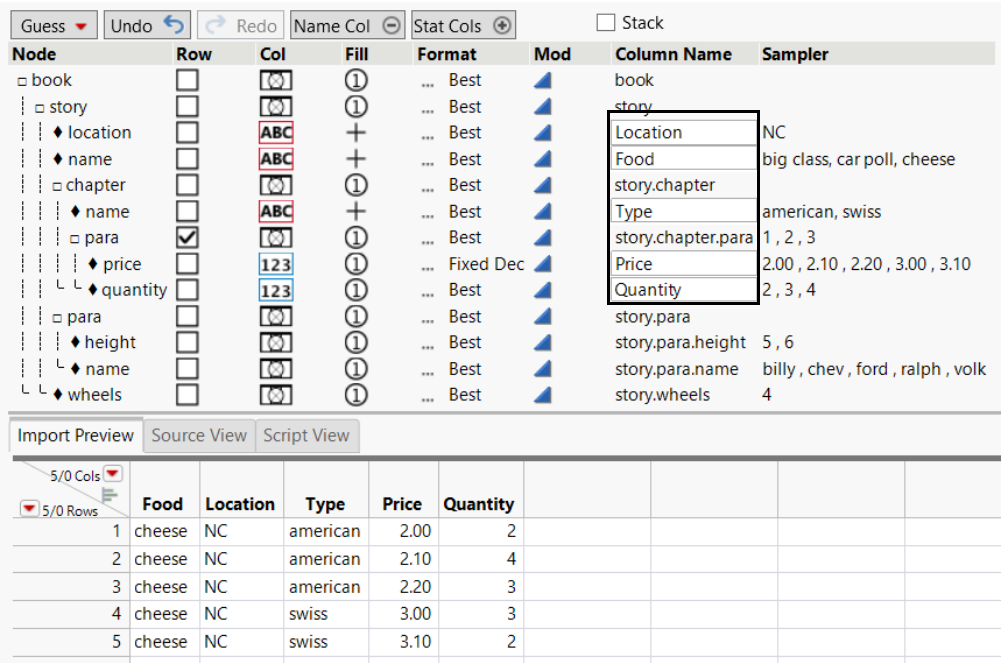
4. In the Format column, click **Best** next to price, select **Fixed Dec**, type "2" for the decimal place, then click **OK**.

Figure 3.32 Format Type Selected in the Format Column



5. In the Column Name column, rename columns as shown in Figure 3.33.

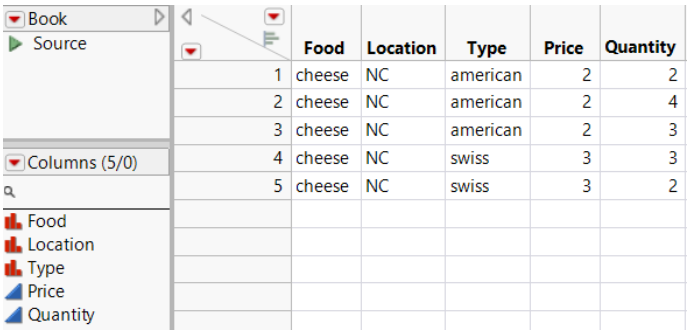
Figure 3.33 Renamed Columns in the Column Name Column



6. At the bottom of the window, click **OK**.

   The data is imported in a data table as you specified.

Figure 3.34 Final Data Table



View the Statistics in the XML Import Wizard

Click **Stat Cols** at the top of the XML Import Wizard to show statistics for the elements.

Occurs The number of times the element occurs. The root element occurs once and is usually not the element to make rows from. You'll get as many rows from the element as occur.

If you pick multiple rows, then the actual number might be less than the total of the "occurs"; elements that end right after another element ends don't generate an extra row. For example, if you select both <para> and <chapter> as row makers, then you don't get an extra row when the chapter ends right after the last paragraph. But you could get an extra row if there were a <footnote> for <chapter> after the last paragraph and before the end of the chapter. You'd have to select the footnote column to make that happen.

Writes The number of times the source has a value that could be written. If it is zero, you probably don't want to make a column from this element, because the values would be all missing.

Unique The number of unique values (less than or equal to the Writes value).

Repeats The maximum number of times this element repeats within its parent element. Bigger repeat values are likely candidates to make rows. If this element repeats and is used as a column, a character column will make a list; an expression column might be even better because it uses {lists} (within curly brackets) to indicate where the values stop and start. A numeric column will get only one of the repeating values.

Sampler The values for the element.

XML Import Wizard Options

Guess JMP selects the settings based on the size of data table that you select. **Tall guess** selects an element that will make as many rows as possible (the default setting). **Wide guess** selects an element to make rows that will make as many columns as possible. **Huge guess** selects as many elements to make rows as it needs to get all of the data into the table; it might be bigger than you expect.

Undo Reverses the last change made to the window.

Redo Recalls the last change made to the window.

Name Col Shows or hides the Column Name column.

Stat Cols Shows or hides statistics for the elements in the tree.

Stack Applies to nodes that repeat within a parent node that is creating rows. By default, extra values are stored in a single table cell separated by commas. If this option is selected, repeating values are stacked in extra rows.

Import Preview Shows a preview of the imported data according to the options that you select.

Source View Shows the source XML document.

Script View Shows a JSL script of the settings that you select.

Import XML Files Directly

JMP can detect the structure of an XML or JSON file and import it directly into a data table. If you don't want to see a preview of the data in the XML Import Wizard, use this method.

1. On Windows:
   - Select **File > Open**.
   - Set the file type to **Data Files**.
   - Select the file.
   - Next to Open as, select **Data (Best Guess)**.
2. On macOS:
   - Select **File > Open** and select the file.
   - Click **Options** on the lower left corner of the window.
   - Select **Data (Best Guess)**.
3. Click **Open**.

The file opens as a data table.

If you are not satisfied with the results, consider using the XML Import Wizard to customize the import. See "Preview and Import XML and JSON Data".

Note: You can import XML or JSON text from a script editor window by selecting Edit > Import as Data. If nothing is selected in the file, then the import is based on the file extension. If text is selected, then JMP attempts to parse the text as JSON or XML, Otherwise, the text is imported into a CSV file.

Import JSON Files Directly

A JSON file consists of name and value pairs that are imported as column headings and data. You can import a JSON files with the XML Import Wizard ("Preview and Import XML and JSON Data") or directly.

To open a JSON file directly, follow these steps:

1. On Windows:
   - Select **File > Open**.
   - Set the file type to **JSON Data Files**.

  - Select the file.
  - Select **Data (Best Guess)** next to **Open as**.
2. On macOS:
  - Select **File > Open** and select the file.
  - Click **Options** on the lower left corner of the window.
  - Select **Data (Best Guess)**.
3. Click **Open**.

The file opens as a data table.

Notes:

- The imported JSON file does not contain date, time, currency, geographic, percent, and scientific information to determine the column format. If you want to use any of these formats, right-click the numeric column, select **Column Info**, and change the format in the Data Type list.
- Member names in name-value pairs are case insensitive.
- The JSON import code does not use the JSON file's numeric, Boolean, or string distinction. Instead, it determines the entire column can be numeric or not. You can explicitly specify the column as character if the guess is not what you want.
- The file cannot have an empty array element. The name must be quoted before the colon in the name-value pair.
- The JSON standard accepts decimal numbers with decimal points.

Build SQL Queries in Query Builder

Query Builder is the preferred method for selecting and importing data from a SQL database without writing SQL statements. You can preview the data before importing it into a data table. Share your queries so that other users can customize and run the queries.

Query Builder provides an alternative to writing your own queries using the File > Database > Open Table feature. However, you can also start building a query in Query Builder and then add your own SQL statements.

SAS Query Builder is also available for querying SAS data sets on SAS servers. See "Open SAS Data Sets with SAS Query Builder".

Notes:

- Database table names that contain the characters `$# -+/%()&|;?` are not supported.

- The Value Order column property is not supported. Consider writing a script in the Post-Query Script window to sort the rows.

Connect to a SQL Database

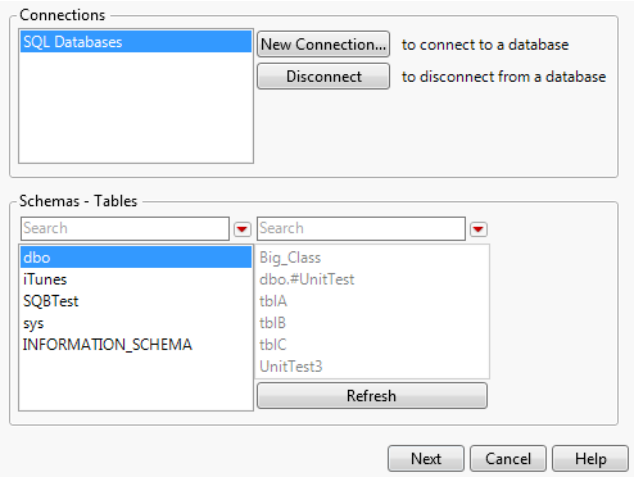
Set up the ODBC connection through the Windows Control Panel or inside JMP.

1. Select **File > Database > Query Builder** to display the Select Database Connection window.

   The Connections box lists data sources that you connected to in the current JMP session.
2. If the desired data source is *not* listed in the Connections box, click **New Connection** to choose a data source. The method of choosing a data source depends on your operating system and the ODBC driver. See "Connect to a Database".
3. Select a table or schema from the Schemas - Tables box and click **Next**.

   Query Builder examples are based on a table named SQBTest, which contains movie rental data.

Figure 3.35 Select the Database Schema



Tip: To find the table in a long list, enter the name in the search box above the schemas. You can also search for tables above the list of tables. The red triangle menu provides options for matching case and searching with regular expressions.

Select Tables from a SQL Database

After connecting to the SQL database, select the tables that you want to query. Either select a primary table or join several tables to query them all.

By default, JMP attempts to join tables based on key relationships that are assigned in the tables.

- A primary key identifies a column that uniquely describes the data (for example, a customer ID number). All rows from the primary table are included in your query.
- A foreign key in a secondary table matches the primary key in one of the joined tables. Only matching rows from the secondary table are included in your query.

If there are no keys, data are matched by column name, which joins the two tables. By default, only matching rows from the secondary tables are included in the query.

This example shows how to join multiple tables. However, you can also build a query using a single table. In this case, joining is not necessary.

Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
2. Select the schema from the Schemas - Tables list.

   If you also select a table, that table will be the primary table after you click Next.
3. In the Select Tables for Query window, select **g6_Customers** from the Available Tables list, and then click **Primary**.

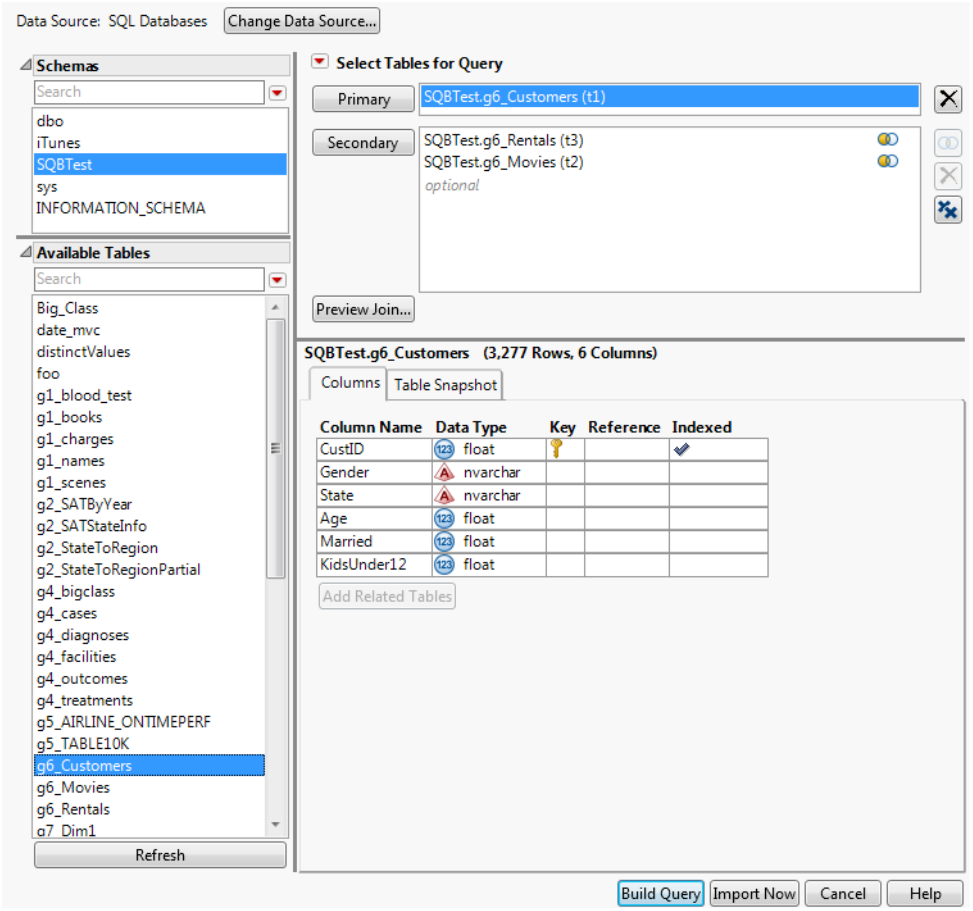
   The Columns tab shows that CustID is the primary key. The data is indexed, which speeds up the query.
4. Select **g6_Movies** and **g6_Rentals** from the Available Tables list, and then click **Secondary**.

   The Left Join icon indicates that the tables were automatically joined (Figure 3.36). CustID is the primary key in g6_Customers and matches a foreign key in one of the other tables.

   Tip: After you add a primary or secondary table, click **Add Related Tables** to add tables that have matching columns. The button is unavailable when no related tables are found or when a primary or secondary table is not selected.

Figure 3.36 shows the completed window.

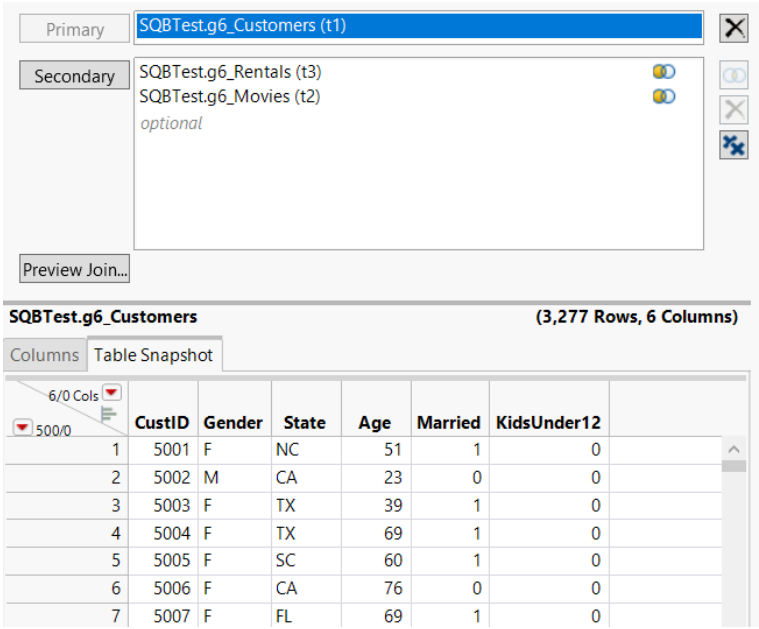
Figure 3.36 Selecting Primary and Secondary Tables



Tip: To find the schema in a long list, enter the name in the search box above the schemas. You can also search for tables above the list of tables. The red triangle menu provides options for matching case and searching with regular expressions.

5. Click the **Table Snapshot** tab for each table to preview the data.

Figure 3.37 Table Snapshot Tab



6. Below the primary and secondary tables, click **Preview Join** to see a preview of the table that was created from the specified joins.

Tips:

- The icon next to a secondary table indicates that the table is not joined in the query. Click the **Edit Join** button to specify the columns to join. If you cannot find columns to join, click the **Remove** button to remove the table. See "Edit the Conditions for Joining Tables".
- On the Columns tab, the Key column might show multiple keys; some columns can be both primary and foreign keys. A unique key icon does not appear next to primary keys, because all primary keys are unique.
- On the Columns tab, the Reference is specified for foreign keys that match primary keys in another schema. The reference is the name of the schema and column.
- Click **Change Data Source** to query a different schema or database.
- To join data from different sources (for example, a database and Microsoft Excel), use Query Builder to import the database data into a data table; import the Excel data into a data table; use JMP Query Builder in the Tables menu to query and join the tables.
- When you import a table that contains a primary key, the Link ID column property is added to the column in the data table. The column property enables you to virtually join data tables. See "Virtually Join Data Tables".

Edit the Conditions for Joining Tables

In the Select Tables for Query red triangle menu, Auto join Database Tables is initially selected. JMP automatically joins database tables based on key relationships or matching column names.

If there are no keys, or when column names do not match, click **Edit Join** to specify the columns to join.

Note: The Query Builder examples are based on a database that is not installed with JMP.

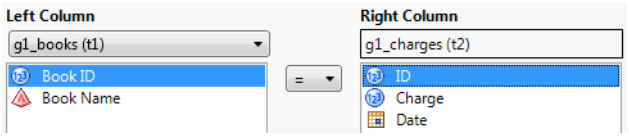
To edit the conditions for joining tables

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
2. In the Select Tables for Query window, select **g1_books** as the Primary table and **g1_charges** as the Secondary table.

   The icon next to the secondary table indicates that the table is not joined in the query.
3. Select **g1_charges** in the Secondary table pane and click **Edit Join** .

   The Add Condition window appears.
4. In the Left Column list, select **g1_books**.
5. Select **Book ID** from the Left Column box.
6. Select **ID** from the Right Column box.
7. Make sure that the equal sign is selected between the two boxes.

Figure 3.38 The Add Condition Window



8. Click **Next**.

   The Edit Join window shows that non-matching rows from g1_books will be included in the data table. Rows that are only in g1_charges will be omitted.

   To do a full join and import all rows, you would select **Include non-matching rows from g1_charges**. If only one of the non-matching options is available, the database does not support full joins.
9. Click **OK**.

Note: The **OK** button is unavailable until all of the secondary tables are joined.

To prevent tables from joining automatically

- Deselect **Auto join Database Tables** from the Select Tables for Query red triangle menu above the primary table.
- If you frequently query large databases, deselect **Automatically join tables added to a query** in **Preferences > Query Builder** to prevent memory issues.

Build an SQL Query

After selecting database tables, you either import the data or build a query. Query Builder enables you to interactively create the database query rather than write SQL expressions.

After selecting database tables (and joining them if necessary), click **Build Query** to open the Query Builder window. You can continue to refine the query by selecting which columns to include and specifying criteria for sampling and filtering. You can also save the query to edit and run again later.

The columns from all database tables appear in the Available Columns list. Prefixes such as t1 and t2 (also called *aliases*) associate each column with the corresponding database table.

To skip the Query Builder step and import all data, click **Import Now** instead.

Note: The JMP Query Builder in the Tables menu provides many of the same options but lets you query and join JMP data tables. See "Query and Join Data Tables with JMP Query Builder".

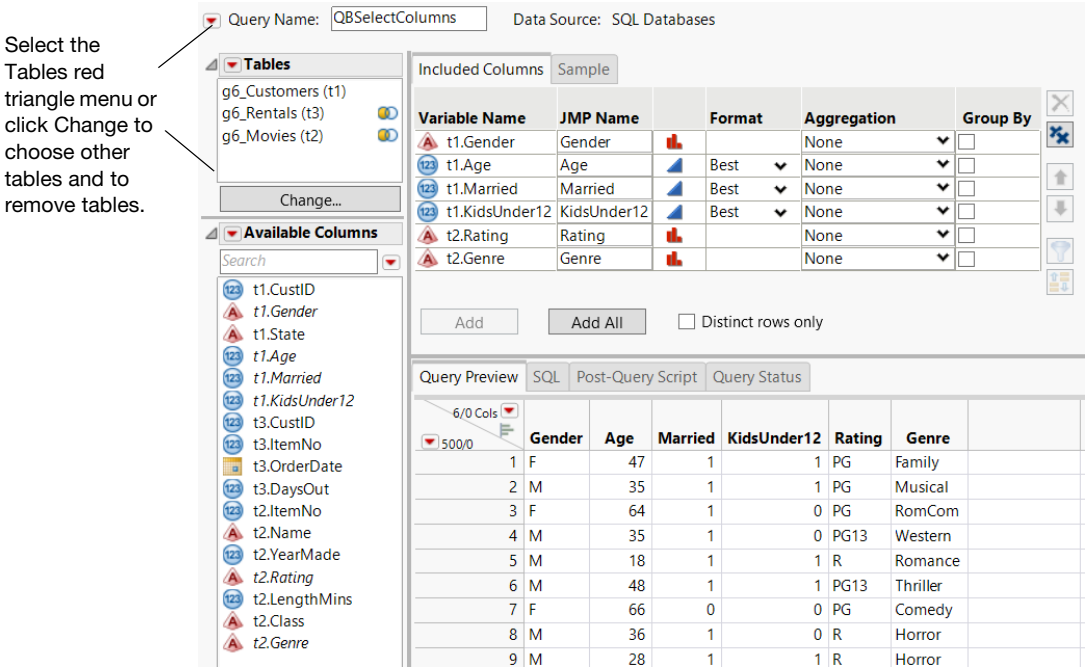
Select Columns from the Database Table

Suppose that you want to view movie rentals by movie genre, rating, and demographic data such as marital status and age.

Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
2. In the Select Tables for Query window, select **g6_Customers** and click **Primary.**
3. Select **g6_Movies** and **g6_Rentals** and click **Secondary**.
4. Click **Build Query** to show the Query Builder window.
5. In the Available Columns box, select t1.Gender, t1.Age, t1.Married, t1.KidsUnder12, t2.Rating, and t2.Genre.
6. Click **Add** on the Included Columns tab.

Figure 3.39 Selected Columns



Notes:

- When a query contains eight tables or more, a search box appears above the table names near the upper left corner. To find a column in a long list, enter the name in the search box below Available Columns. The red triangle menu provides options for matching case and searching with regular expressions. To hide the search box, deselect **Show Search Box** from the Available Columns red triangle menu.
- Reference columns no longer appear in the Available Columns list because they are not supported in queries.
- You can change the width of the column by selecting an option from the Format list in the Included Columns list.

7. Select the **SQL** tab below the columns to view the SQL statements for your query. This code is saved as a data table property after you run the query.
8. Click **Save** in the lower right corner.

   Your work is saved as g6_Customers.jmpquery, which you can open later to return to this point or to run the query.
9. Click **Run Query** to import the data.

   The data table includes the following scripts:

   - Run the Source script to reconnect to the database and import the data.

– Run the Modify Query script to open the query in Query Builder.
– Run the Update From Database script to re-import and refresh the data.

Tips:

- To rename a column, double-click the JMP Name in the Included Columns tab and enter a new name.
- To rename an alias, right-click the table in the Select Tables for Query window and select **Change Alias**. Aliases are not case sensitive.
- The query runs in the background unless you deselect **Run queries in the background when possible** from the Query Builder ODBC preferences. You can also check the progress of all ODBC queries by selecting **View > Running Queries.**

Note: For SAS Query Builder, all queries run in the foreground.

- Deselect **Update preview automatically** if the preview loads too slowly. Click **Update** below the Query Preview tab to update the data view. Consider changing the Preview options in the JMP Query Builder preferences if you frequently work with large databases. Consider limiting the maximum number of rows that can be previewed. In the JMP Query Builder preferences, change the value of **Maximum number of rows for previews**.
- To omit duplicate rows from the database, select **Distinct rows only** on the Included Columns tab.

Maintain Compatibility with JMP 12

If you add a JMP 16 feature to a query, that query will no longer load in JMP 12. If you are using JMP 13, but you need to create queries that will still run in JMP 12, select **Keep this query compatible with JMP 12** in the Query Builder Preferences. After you select the option, features that create compatibility problems are hidden in Query Builder.

When you are ready to move JMP 12 queries to a later version, deselect this preference.

Create a Computed Column

You can create a new column from existing columns. You might calculate the mean for two columns and store the mean in a new column. Date-time values might be in the wrong format. Click the Available Columns red triangle, select **Add Computed Column**, and create the new column in the formula editor.

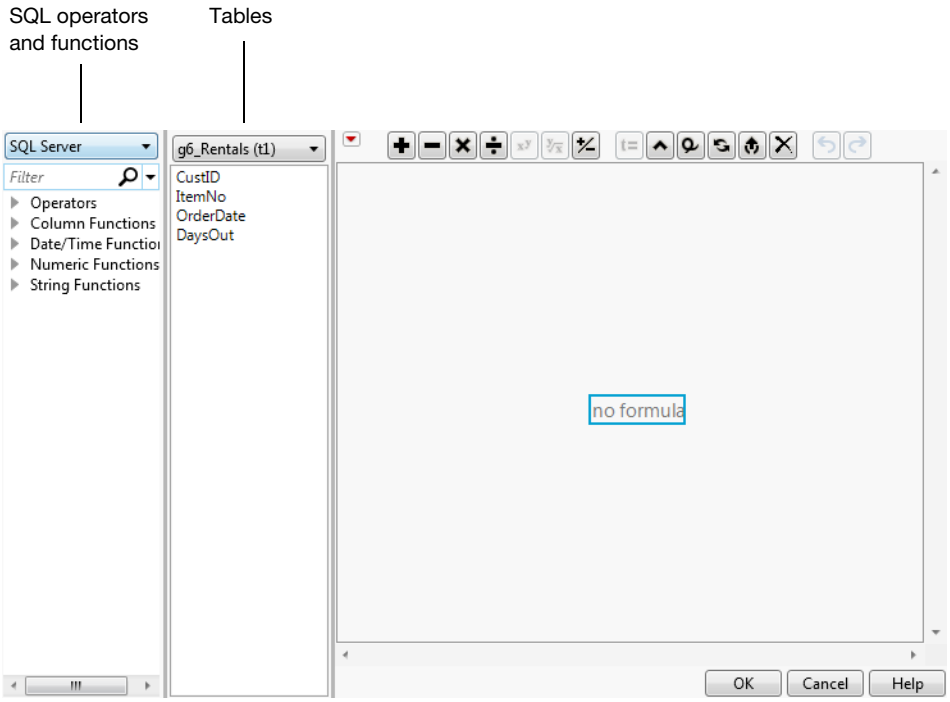
Suppose that you want to calculate the maximum number of times you can watch a movie during the rental period. You are querying a database that contains the length of each movie and number of days the movie was checked out. This example shows how to create a new computed column from these data.

Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
2. In the Select Tables for Query window, select **g6_Rentals** as the Primary table and **g6_Movies** as the Secondary table.
3. Click **Build Query** to show the Query Builder window.
4. Click the Available Columns red triangle and select **Add Computed Column**.

   The Computed Column window appears. The window contains the JMP Formula Editor.

Figure 3.40 Computed Column Window with Formula Editor



Notes:

- Operators and functions are provided in the list on the left side of the Formula Editor (Figure 3.40). In some instances, you might need to change the server type based on your database.
- The Operators list does not provide a Concatenate (||) operator. You must type the formula in the Formula Editor box.

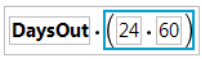
5. From the g6_Rentals list on the left, select **Days Out** and click the multiplication ✖ button.

Figure 3.41 Computed Column

6. Select the blank box, type (24 * 60, and press Enter.

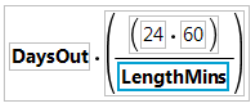
   This formula multiplies the number of minutes in an hour by the number of minutes in day. Notice that when you type the first parenthesis, then second one is automatically inserted.

Figure 3.42 First Portion of the Formula



7. Click the outer box to select the entire equation and click the division button.
8. Select **g6_Movies** from the list on the left and then select **LengthMins**.

Figure 3.43 Second Portion of the Formula



9. Click **OK**.

   A new column named Calc1 is created.
10. Right-click the column and select **Rename Column**.
11. Type MaximumTimesWatched and click **OK**.
12. In the Available Columns list, select MaximumTimesWatched and click **Add**.
13. Select t2.Name and click **Add**.

    On the Query Preview tab, notice that Nanny McPhee can be watched 160 times while the movie is rented.

Group the Common Values

You can combine (or group) common values in a column before importing the data into JMP. To group common values, select an Aggregation function to determine how the common values are calculated.

Note: Aggregation support is based on your database. See the database documentation.

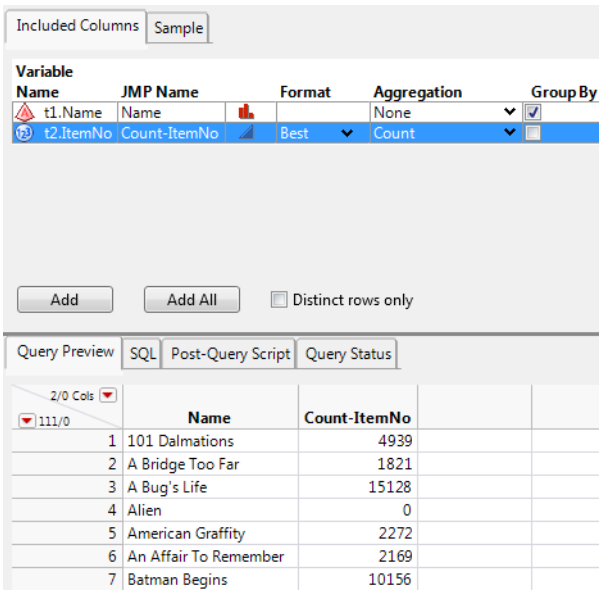
Suppose that you are interested in the number of times a specific movie was rented. In this example, the count for each item number is calculated, and common movie values are grouped into single rows.

Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database" for.
2. In the Select Tables for Query window, select **g6_Movies** as the Primary table and **g6_Rentals** as the Secondary table.
3. Click **Build Query** to show the Query Builder window.
4. In the Available Columns box, select t1.Name and t2.ItemNo and click **Add**.
5. Select t2.ItemNo and select **Count** from the Aggregation list.

   The Group By check box is selected for t1.Name. All instances of a specific movie name will be grouped into one row.

Figure 3.44 Grouped Columns



6. Click **Run Query** to import the data.
7. In the data table, right-click the Count-ItemNo column and select **Sort > Descending**.

   Scarface was rented most frequently.

Figure 3.45 Partial View of the Sorted Count-ItemNo Column

	Name	Count-ItemNo
1	Scarface	15991
2	Rambo	15643
3	Nanny McPhee	15289
4	Marmaduke	15255
5	A Bug's Life	15128
6	Diary of a Wimpy...	14989
7	Cars	14905

Tips:

- To clear the grouped rows, select **None** from the column's Aggregation list.
- The DISTINCT Aggregation functions show only rows that contain distinct values. Rows with duplicate values are omitted. These functions are useful when a database contains many duplicate values.

Import a Sample of the Data

With large databases, consider sampling the data. Sampling returns a subset of rows and decreases the query time. The database query runs, and a smaller portion of data are imported based on options that you select on the Sample tab.

Sampling methods differ based on the database vendor.

- SQL Server supports block sampling by default. A block sample takes an entire page of rows (such as all rows on pages 1 and 5). If you select 1,000 rows, approximately 1,000 rows are imported.
- Oracle and other databases support row sampling. If you select 5,000 rows, between 4,800 and 5,200 rows per sample are typically imported, based on how Oracle cycles through the data.

For major database vendors, JMP detects the capabilities and provides vendor-specific options when possible. Features that are unsupported by the vendor are unavailable on the Sample tab.

Suppose that you want to import a sample of the data. In this example, you select 5,000 random rows.

Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database" for.

2. In the Select Tables for Query window, select **g6_Rentals** as the Primary table and **g6_Movies** as the Secondary table.
3. Click **Build Query** to show the Query Builder window.
4. Click the **Add All** button on the Included Columns tab.
5. Click the **Sample** tab and select **Sample this result set**.
6. Select **Random N Rows** and type 5,000.

   In the Sample By area, Blocks or Pages is the only option based on which type of sampling the database supports.
7. Click **Run Query** to import the data.

   The new data table consists of approximately 5,000 rows. With block or page sampling, you might get a sample of 4,900 rows one time and 5,600 rows the next time.

Tips:

- To re-create the same sample set each time you run a query, set the Seed value to any positive integer up to 64,000. Suppose that you want to query movie rentals by gender. Type 1 as the Seed value and run the query. The distribution of male customers in the results is low. Type 2 as the Seed value and run the query again. Repeat this process to find the Seed value that results in a similar distribution of males and females.
- To add individual columns to the Included Columns tab, right-click the column and select **Include Column** or click the **Add** button.

Select Filters to Import a Subset of the Data

Add filters to import a subset of values from the selected filters into the data table. Some filters are not available if the query is compatible with JMP 12. See "Maintain Compatibility with JMP 12".

Filters for Both Continuous and Categorical Columns

Simple Comparison Matches values using the specified operator.

`Age > 14` matches ages that are greater than 14.

Range Matches a range of values using the specified operator.

`12 ≤ Age ≤ 17` matches ages that are between 12 and 17.

Is NULL or Is Not NULL Matches missing values.

Either NULL or not NULL matches missing values and non-missing values.

Custom Expression Enables you to write your own SQL expression.

```
( ( ( t2.Gender IN ( 'F' ) ) AND ( (t2.Age >= 20) AND (t2.Age <= 50)
  ) ) )
```

matches the F Gender. It also matches Age between 20 and 50.

Filters that are Only for Categorical Columns

List Box Displays a list box from which you select one or more values. List Box is the default filter for categorical columns based on the Query Builder preferences.

Manual List Enables you to enter the column values.

Check Box List Displays a check box list.

Note: List Box, Manual List, and Check Box List include a **Not in list** option that enables you to retrieve rows that do not match the selected values.

Contains Matches a string that contains or does not contain the specified value. **Contains Comedy OR Romance** matches Comedy and Romance.

For most categorical columns, the filter is a List Box by default. For columns that contain over 1,000 levels, the Contains filter is automatically selected. You can change the number of levels in the Query Builder preferences.

Like or Not Like Matches a string that is similar to or not similar to the specified value. Supports the % wildcard (zero or more characters) and _ wildcard (exactly one character).

Genre Like %com matches any number of characters before "com", as in "RomCom". To also match "Comedy", use **%com%** or **Contains com**.

Match Column Values Matches the specified column value. Select the table and then select the columns. The **Select non-matching** option enables you to filter all rows except for the selected rows. See "Import Matching Data from an Existing Data Table" for an example.

This example shows how to import data for age 30 and over customers, and movies in the RomCom and Comedy genres.

Note: The Query Builder examples are based on a database that is not installed with JMP.

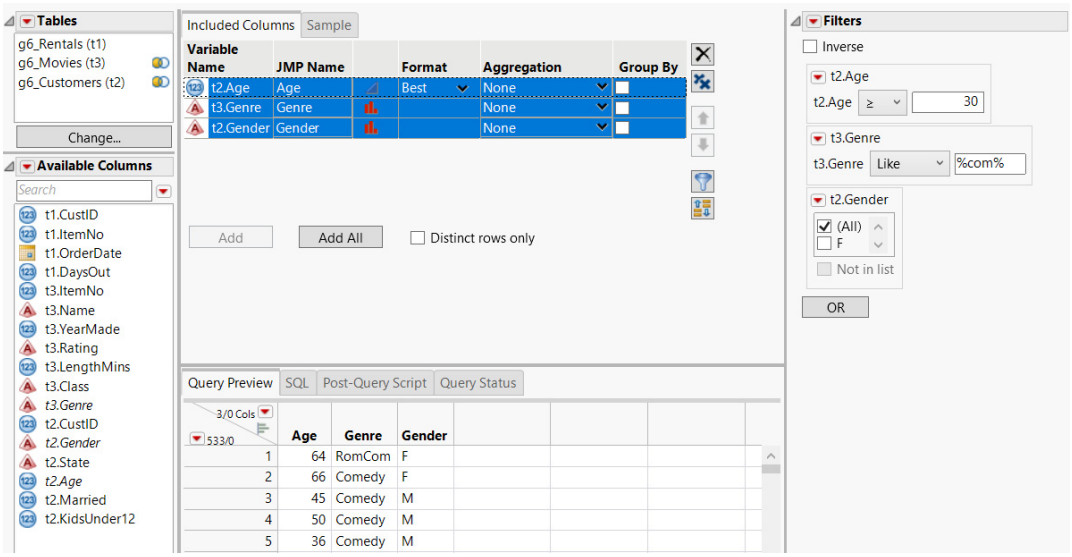
1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
2. In the Select Tables for Query window, select **g6_Rentals** from the Available Tables list, and then click **Primary**.
3. Select **g6_Customers** and **g6_Movies** and then click **Secondary**.
4. Click **Build Query** to show the Query Builder window.

5. In the Available Columns box, select t2.Gender, t2.Age, and t3.Genre, and then click **Add** on the Included Columns tab.
6. Select all columns on the Included Columns tab and click **Add Selected Items to Filters** .

   Filters for the columns appear in the Filters outline.
7. Set the t2.Age filter to ≥ 30.
8. Click the t3.Genre red triangle and select **Like**, type %com%, and press **Enter**.

   The % wildcards match any number of characters before and after "com". On the Query Preview tab, notice that movies in both the RomCom and Comedy genres are shown.

Figure 3.46 Selecting Filters



9. Click the Filters red triangle and select **All Prompt on Run**.

   Users who run the query can customize the filters.
10. Click **Run Query**.
11. In the Query Prompts window, click **OK** to apply the preselected filters and import the data.

Notes:

- For most categorical columns, the filter is a list box by default. For columns that contain over 1,000 levels, the Contains filter is automatically selected. You can change the number of levels in the Query Builder preferences.
- The Conditional option in a filter's red triangle menu enables you to filter data within hierarchical categories. For example, suppose that you have a State filter and a City filter.

To select a state and then display only cities that are in that state, click the City red triangle and select Conditional.

- The Inverse option at the top of the Filters list enables you to select all but the specified rows for all filters. The option is unavailable for filters that select all rows.
- "<Blank>" in the filter list indicates that the database contains a missing value for that column.
- To create a filter for large columns of categorical data, JMP attempts to determine the number of rows in the table.
  - The Query Builder preference called **Retrieve category levels for tables whose size cannot be determined** is selected by default so that JMP automatically retrieves the levels. If you deselect the preference, the Contains fallback filter type in the Query Builder preferences is selected.
  - If the categorical column has more than 1 million rows, JMP does not automatically retrieve the unique category levels for the filtered column. The Query Builder preference called **Maximum rows in table for which category levels will be automatically retrieved** supports a minimum of -1 (no limit) and a maximum value of 1 billion rows.
- The default filter for categorical columns is a list box unless the **Keep this query compatible with JMP 12** Query Builder preference is selected.

Import Matching Data from an Existing Data Table

You can also select rows from an open data table that match a column in your query. Consider a database of airline data. The database includes data such as flight duration and tail number. You also have a data table that includes tail number data. Use the Match Column Values filter to import only data for matching tail numbers.

Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **Help > Sample Data Library** and open Air Traffic.jmp.
2. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
3. In the Select Tables for Query window, select **g5_AIRLINE_ONTIMEPERF** from the Available Tables list, and then click **Primary**.
4. Click **Build Query** to show the Query Builder window.
5. Click **Add All** on the Included Columns tab.
6. Select t1.TailNum on the Included Columns tab and click **Add Selected Items to Filters** .

7. Click the t1.TailNum red triangle in the Filters column, select **Filter Type**, and then select **Match Column Values**.
8. Select **Air Traffic** below **Match values from table**.
9. Select the Tail Number column and then select **All rows (38,118)** from the list.

   The data view on the Query Preview tab updates to show the filtered values.
10. Click **Run Query** to import the data.

    The data table includes only data for rows that are in the Tail Number column.

Write a Custom Expression to Import a Subset of the Data

In addition to selecting filters to subset the data, you can write custom SQL expressions if you do not want to use the filters that are provided.

Note: The Query Builder examples are based on a database that is not installed with JMP.

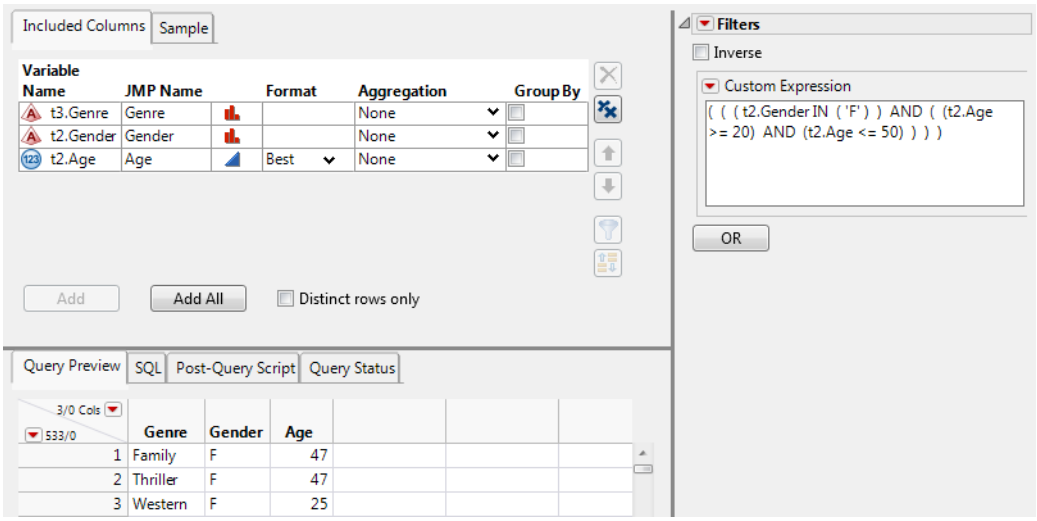
1. Select the columns that you want to filter (described in "Select Filters to Import a Subset of the Data").
2. Click the Filters red triangle and select **Add Custom Expression**.
3. Type the following text in the Custom Expression box:

```
( ( ( t2.Gender IN ( 'F' ) ) AND ( (t2.Age >= 20) AND (t2.Age <= 50)
  ) ) )
```

4. Click outside the Custom Expression box to update the Query Preview tab.

   This expression matches the F Gender. It also matches Age between 20 and 50.

Figure 3.47 Writing a Custom Filter Expression



Sort the Selected Data

You can sort the rows in specific columns by values to control how the data appear in the data table. In this example, you sort the Married column in descending order and sort the data by age and then height.

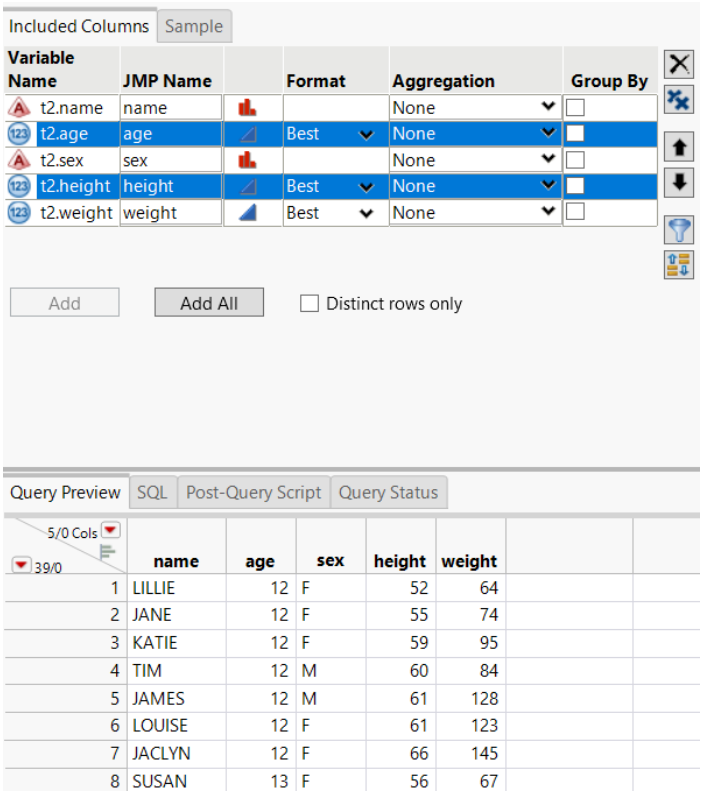
Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
2. In the Select Tables for Query window, select **g4_bigclass** as the Primary table**.**
3. Click **Build Query** to show the Query Builder window.
4. On the Included Columns tab, click **Add All**.
5. Select t1.age and t1.height and click **Order by the Selected Items** .

   The columns appear in the Order By outline in the right column.

   The columns are sorted by age first (youngest to oldest) and then height (shortest to tallest).

Figure 3.48 Selecting the Order By Columns



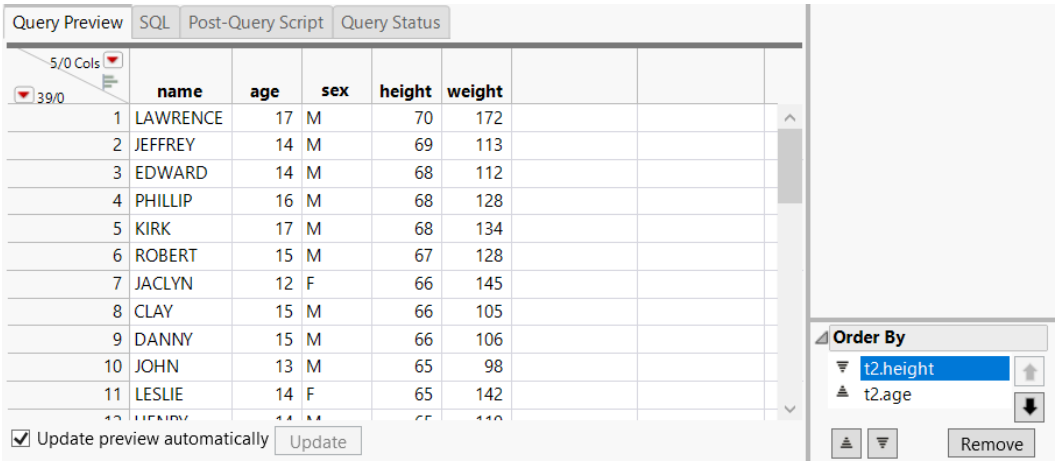
6. In the Order By outline, select t1.height and then click **Sort the values in descending order** below the columns.

   The height column is sorted from tallest to shortest.

7. Select t1.height and click **Move the Selected Items Up in the List** .

   The height column is sorted first. Values in the age column are sorted within each level of height. For a height of 68, age is sorted from 14 to 17.

Figure 3.49 Result of Reordering Columns



View the Query Status

On the Query Status tab, view the status of a query as it runs in the background. The query name, SQL statements, and number of processed records appear. You can stop a query at any time and view only the processed records. To view background queries from other JMP windows, select **View > Running Queries**. The status details are unavailable if you deselect **Run queries in the background when possible** from the Query Builder preferences.

Note: For SAS connections in Query Builder, all queries and query previews run in the foreground.

Write a Post-Query Script

On the Post-Query Script tab, write a JSL script that runs after you run the query. For example, you might want to import the data and then create a distribution.

```
Distribution( Column( :age, :gender ) );
```

This script is part of the Source script in the final data table.

Save and Run a JMP Query

Save your query as a .jmpquery file to modify or run the query later. You are prompted to enter the password if the server connection string does not specify it. The .jmpquery file can also be opened and run by a JSL script.

After you build a query, click **Save** in the lower right corner to save the settings as a .jmpquery file. Clicking **Save** again overwrites the file with your latest changes. Clicking **Save As** saves the query in a new .jmpquery file.

Run a JMP Query

A query file opens in Query Builder by default, where you modify the query and then run it as needed to import the data. After you are satisfied with the query, you can configure it to run when opened instead of opening in Query Builder.

When you are building the query, select **Run on Open** from the red triangle menu next to the Query Name box in the upper left corner.

To override the Run on Open setting, press Ctrl (Windows) or Command (macOS) before opening the file from the file system. On Windows, you can also right-click the query in the JMP Home Window and select **Edit Query**.

Use a Saved Query as a Template

You can also use the .jmpquery file as a starting point for a new query. This option prevents you from overwriting your original query if you accidentally click **Save**. It works the same as clicking **Save As** after you modify the query.

1. Right-click the .jmpquery file in the JMP Home Window's Recent Files list.
2. Select **Open as Template**.
3. Modify the query and click **Save** to create a new .jmpquery file.

Note: The .jmpquery file contains database login information. You must have set up the database connection before running the query. See "Connect to a Database".

Open the Selected Data

After you specify the columns and data to import from the SQL database, click **Run Query** to open the data in a data table. The SQL statements are saved as a table variable. The following scripts are available:

Source Runs the query.

Modify Query Opens the query in Query Builder, where you can change which columns and data are imported and further customize the query.

Update From Database Connects to the database to refresh the data and then run the query. If the data table and database table contain the same number of columns, values in the existing data table are updated when the database is refreshed.

The existing data table is also updated when only formula columns follow the last column that is refreshed.

Otherwise, the updated data appears in a new data table.

Query Builder Red Triangle Options

The Query Builder red triangle menu provides scripting and custom SQL options. The modify and run scripts are always automatically saved in the final data table.

Copy Modify Script Copies a script to the computer's clipboard that lets you modify the query.

Copy Run Script Copies a script to the computer's clipboard that lets you refresh the data and run the query.

Save Modify Script to Script Window Saves the Modify Query script to the script window.

Save Run Script to Script Window Saves the Update From Database script to the script window.

Run on Open Runs the query when you open the query file.

Convert to Custom SQL Shows the query statements in a new script editor window. You must remove prompting filters before selecting this option.

When you save the query from the Custom SQL window, the custom SQL is saved. Interactive components that were present before you customized the query are not saved. Revert to Interactive is also unavailable on the red triangle menu.

Keep this query compatible with JMP 12 Makes subsequent changes to the query compatible with JMP 12. Features that cause compatibility issues with JMP 12 are hidden while you edit the query. After you deselect this option and make a change that is supported only in JMP 13 and later, this option is no longer available.

The JMP Query Builder preferences include the same option, which is deselected by default. When you open a query in JMP 16 that is marked as JMP 12 compatible, features that create compatibility problems are hidden regardless of how the preference is set.

Revert to Interactive Displays the interactive query in the Query Builder window. Changes that you made on the Custom SQL tab are not saved when you revert.

Write SQL Statements in Query Builder

Query Builder enables you to interactively create SQL queries without writing SQL statements. You can also build a query in Query Builder and then add custom statements to the query.

Note: The Query Builder examples are based on a database that is not installed with JMP.

1. Select **File > New > Database Query**, connect to the database, and select the **SQBTest** schema. See "Connect to a Database".
2. In the Select Tables for Query window, select **g6_Rentals** as the Primary table.
3. Select **g6_Movies** and **g6_Customers** as the Secondary tables.
4. Click **Build Query** to show the Query Builder window.
5. Click the **Add All** button on the Included Columns tab.
6. Click the Query Builder red triangle and select **Convert to Custom SQL** and click **OK**.

   The SQL that Query Builder generated appears on the Custom SQL tab.
7. Click before the semicolon and type the following SQL statement:

   ```
   WHERE ( ( ( t2.Gender IN ( 'F' ) ) AND ( (t2.Age >= 20) AND (t2.Age
   <= 50) ) ) )
   ```
8. Click **Run Query** to import the data into JMP.

   The data table scripts include the custom query.

Note: If you select **Revert to Interactive** from the red triangle menu, the changes that you made on the Custom SQL tab are not saved. If you save the custom query and reopen it, Revert to Interactive is not available.

See "Structured Query Language (SQL): A Reference" for a brief primer of SQL statements.

Import Data from SAS

You can connect to a SAS server and work directly with SAS data sets.

Java Runtime Environment (JRE) Requirements

- On Windows, Java Runtime Environment (JRE) 7 or later must be installed on your computer to access SAS. However, JRE 7 does not need to be specified as the current version.
- On macOS, JRE 7 or later must be installed for SAS integration.

Note: On Windows, the bitness of the JRE must match that of JMP. For example, 64-bit JMP communicates with both 32-bit and 64-bit SAS if the 64-bit JRE is installed.

SAS Menu Options

The following options are in the File > SAS menu.

Browse Data Browse and import data residing on a SAS Server.

SAS Query Builder Select and import data on a SAS server without writing SQL statements. See "Build SQL Queries in Query Builder".

Export Data to SAS Export JMP data tables to a SAS Server.

Browse SAS Folders Browse and run SAS stored processes or open Metadata-defined data tables.

New SAS Program Opens a script window for writing and submitting SAS code.

Submit to SAS Sends SAS code directly from JMP to the currently active SAS server.

Open SAS Log Window Opens a SAS log window for the active SAS server.

Open SAS Output Window Opens a SAS output window for the active SAS server. This window shows recent SAS output.

Server Connections Administer connections to SAS servers.

You can also find shortcuts for SAS options on the SAS page of the JMP Starter, and there is a SAS toolbar. You can save certain settings pertaining to SAS Integration on the SAS Integration page of the Preferences window (**File > Preferences**). For more information about setting your SAS Integration preferences, see "Preferences for SAS Integration".

Import SAS Data Sets

SAS data sets are saved in one of many SAS formats:

- Windows supported formats are .sas7bdat and .sas7bxat.
- macOS supports reading and writing .sas7bdat files.
- Windows and macOS support reading and writing .xpt files

When you open a data set in JMP, the file opens as a data table. JMP uses SAS variable names as column names by default. To use variable labels in a specific file on Windows, select the option when you open the file (see step 5 below).

The following ISO date formats are supported for .sas7bdat files: B8601DA, B8601DN, B8601DT, B8601DZ, B8601LZ, B8601TM, B8601TZ, E8601DA, E8601DN, E8601DT, E8601DZ, E8601LZ, E8601TM, E8601TZ, and IS8601DA. If the date format is not supported, the data is imported with the Scientific Notation format.

The following date formats are supported for .xpt files: BEST, DATE, DATETIME,DAY, MMDDYY, DOWNAME, HHMM,HOUR, TIME,YYMMDD, DDMMYY, MMSS, DTDATE, IS8601DA, E8601DA,PERCENT, and DOLLAR.

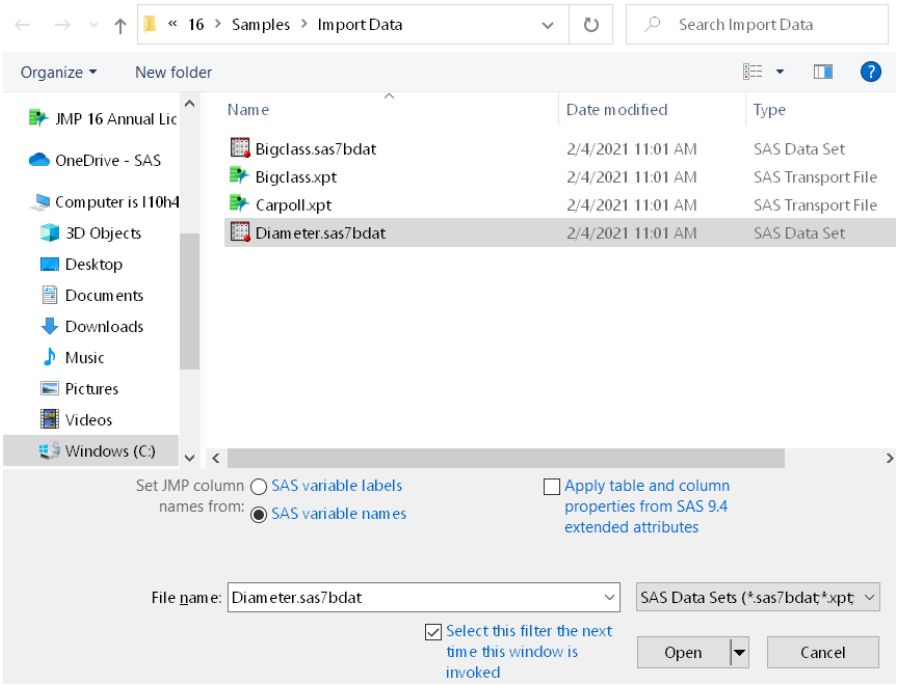
To open a SAS data set from JMP:

1. Select **File > Open**.
2. (Windows only) Select **SAS Data Sets** from the list next to **File name** as shown in Figure 3.50.

   Note: SAS variable names and formats are preserved and can be saved after changes are made to the SAS data set. See "Save JMP Data as a SAS Data Set".

3. Select the file.

Figure 3.50 Open SAS Data Set



Note: To select the columns to import, click the arrow on the Open button and select **Select columns**.

4. (Optional) Select any of the following options:

   SAS variable labels Uses the SAS variable labels (instead of variable names) as the column names in the JMP data table.

   SAS variable names Uses the SAS variable names (instead of the labels) as the column names in the JMP data table.

5. (Optional on Windows) Select any of the following options:

   Apply table and column properties from SAS 9.4 extended attributes If the SAS server supports extended attributes (SAS 9.4), includes the extended attributes when storing JMP metadata. This setting overrides the SAS 9.4 Extended Attributes preference on the SAS Integration page.

   Select this filter the next time this window is invoked Sets the default file type choice to the option that you select next to the **File name** list If selected, the default file type will be **SAS Data Sets** the next time you reach this window.

6. (Optional) Select any of the following for a SAS Transport (.xpt) file:

   Select member Lets you enter the name of a specific member, or table, for JMP to open. On macOS, select **Member Tables > Specified** and then enter the name.

   Open all members Opens all members, or tables, in the transport file. On macOS, select **Member Tables > All**.

   Save all members Saves the file as a JMP file as soon as you open it. The file is saved to the same directory where the SAS transport file was opened. On macOS, the option is **Save all**.

   Select columns Tells JMP to open only certain columns from the transport file. Select the columns that you want to import from the list that appears. On macOS, the option is **Select columns before opening**.

7. Click **Open**.

Note: If you are importing date variables from a SAS file, JMP looks for a SAS date format and translates it to a JMP date column.

Create SAS Transport Files in SAS from JMP

JMP can open SAS transport files that were saved using the SAS XPORT engine. For example, below is sample SAS code that creates a transport file called test.

Note: `misc` and `work` are SAS `libref` names.

```
data test;
input name $ age weight;
cards;
Susan 12 72
Melanie 10 68
Jonathan 11 77
Sheila 13 67
;
libname misc xport 'C:/test.xpt';
proc copy in=work out=misc;
run;
```

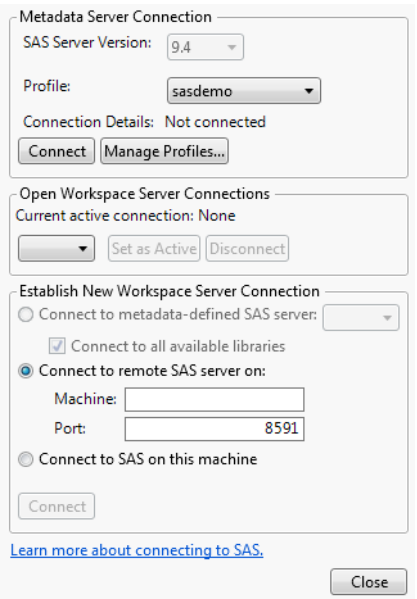
Connect to SAS from JMP

You can either connect to a SAS Metadata Server or directly to a SAS Workspace Server. Once connected to a SAS Metadata Server, you can browse through SAS servers, libraries, and data sets.

Note: SAS Server version 9.4 is the default setting in the JMP SAS Integration preferences though you can change the version to 9.3 if necessary.

To begin, select **File > SAS > Server Connections**. The SAS Server Connections window shown in Figure 3.51 appears. All connections are made in this window.

Figure 3.51 SAS Server Connections



The following sections describe how to connect to a SAS server.

- "Connect to a SAS Metadata Server from JMP"
- "Connect to a Remote SAS Workspace Server from JMP"
- "Connect to a SAS Environment from JMP (Windows Only)"
- "Connect to SAS on Your Local Machine from JMP (Windows Only)"

Connect to a SAS Metadata Server from JMP

Note: You can be connected to only one Metadata Server at a time. If you make a second connection, your first one is disconnected.

To connect to a Metadata Server

1. Select **File > SAS > Server Connections**. The SAS Server Connections window shown in Figure 3.51 appears.
2. Select the version for the SAS Server. Your SAS Metadata Server administrator should have this information.

   SAS Server version 9.4 is selected by default based on the JMP SAS Integration preferences.
3. Select the profile that you want to use.

   If you do not have a profile set up, see "To create or modify a SAS Metadata Server profile".
4. Click **Connect**.

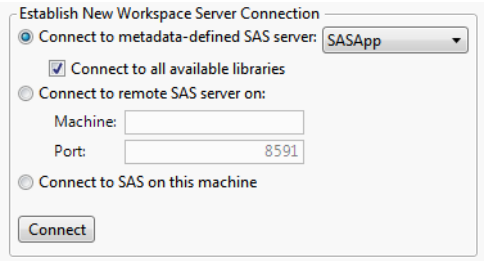
   If JMP is unable to establish a connection, an error message appears. Common reasons are invalid user names or passwords. If you need to update the information for the profile, see "To create or modify a SAS Metadata Server profile".
5. Click **Close**.

Once you are connected to a SAS Metadata Server, you can connect to any SAS Workspace Servers that the Metadata Server offers.

To connect to a SAS Workspace Server (Windows only)

1. Select **File > SAS > Server Connections**. The SAS Server Connections window shown in Figure 3.51 appears.
2. Select the Workspace Server to connect to.

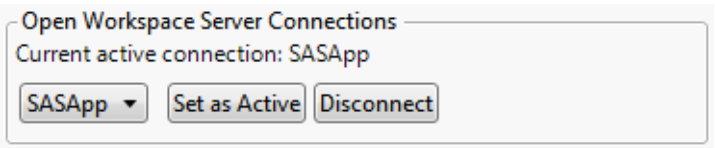
Figure 3.52 Open a Connection to a Workspace Server



3. Click **Connect**.

   Under Open Workspace Server Connections, the Workspace Server is shown as the current active connection.

Figure 3.53 Current Active Connection



4. Click **Close**.

To change the active connection

Note: The active connection is what is used to submit SAS code or handle SAS script commands.

To change the active connection, you first need to be connected to more than one server. Follow the instructions in "To connect to a SAS Workspace Server (Windows only)" to add two or more server connections.

1. In the Open Workspace Server Connections section, click the drop-down menu and select the desired server.
2. Click **Set as Active**.
3. Click **Close**.

Tip: You can change the active server at any time.

To disconnect from a SAS Workspace Server

1. In the SAS Server Connections window, select the Workspace Server to disconnect under Open Workspace Server Connections.

2. Click **Disconnect**.

To disconnect from a SAS Metadata Server

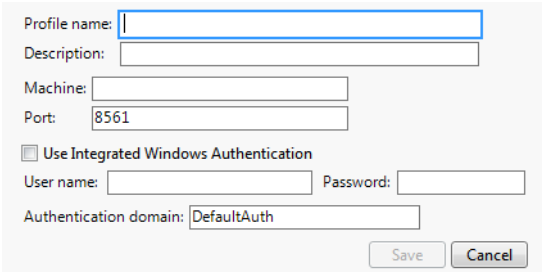
1. In the SAS Server Connections window, select the Metadata Server to disconnect.
2. Click **Disconnect**.

To create or modify a SAS Metadata Server profile

1. In the SAS Server Connections window, select the **SAS Server Version**.
2. Click **Manage Profiles**.
3. Click **Add** to add a new profile, or click **Modify** to change a profile's settings.

   The Create Profile or Modify Profile window appears. If you are adding a new profile, all fields are empty except the Authentication domain field, which contains DefaultAuth, and the Port field. If you are modifying a profile, the fields contain the current information.

Figure 3.54 Create or Modify a Metadata Server Profile



4. Fill in the information needed to connect to a SAS Metadata Server. Your SAS Metadata Server administrator should have this information.

   Profile name Select a name for this profile. This name is shown in the list of profiles.

   Description (Optional) You can enter a short description of this profile.

   Machine The name of the machine that hosts the Metadata Server. (Example: myserver.mycompany.com)

   Port The port through which you should connect to the machine. (Example: 8561)

   Use Integrated Windows Authentication Select this option to use your Windows log in ID and password to access the server. When enabled, the User name and password fields are disabled. This option is disabled by default.

   User name Your user name for the Metadata Server.

   Password Your password. This is always displayed as asterisks.

Authentication domain The domain you, as a user, belong to.

5. Click **Save**.

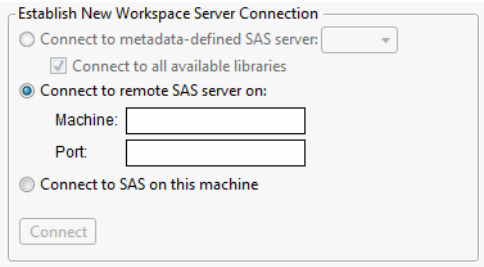
Connect to a Remote SAS Workspace Server from JMP

You can also connect directly to a SAS Workspace Server, instead of going through a Metadata Server.

To connect to a Remote SAS Workspace Server

1. Select **File > SAS > Server Connections**. The SAS Server Connections window shown in Figure 3.51 appears.
2. Under Establish New Workspace Server Connection, select **Connect to remote SAS server on**.

Figure 3.55 Open a Connection to a Remote SAS Server



3. Enter the machine name and the port number. Your SAS server administrator has this information.
4. Click **Connect**.
5. Enter your user name and password in the window that appears.
6. Click **OK**.
7. Click **Close** in the SAS Server Connections window.

To disconnect from a Remote SAS Workspace Server

1. In the SAS Server Connections window, select the server to disconnect under Open Workspace Server Connections.
2. Click **Disconnect**.

Connect to a SAS Environment from JMP (Windows Only)

On Windows, you can connect to a SAS mid-tier (or *SAS environment*) if SAS Server version 9.3 or 9.4 is selected in JMP's preferences and your computer or JMP has been configured correctly. (SAS Server version 9.4 is the default setting in the JMP SAS Integration preferences.)

The SAS installer should have set up your computer to find the SAS environment definition file. If not, you can enter the path to the file in the JMP preferences.

To configure your JMP preferences

1. Select **File > Preferences > SAS Integration**.
2. Select **I want to connect to a SAS Environment** and then click **Configure**.
3. To connect to an environment that JMP has already detected, click **Automatic discovery,** and then select the URL from the list if necessary.
4. To enter the path to the SAS environment definition file, click **Manual configuration** and enter the URL.
5. Click **OK**.

To connect to a SAS Environment

1. Select **File > SAS > Server Connections** to open the SAS Server Connections window.
2. In the Metadata Server Connection area, select **Connect to a SAS Environment**.

   If this option is not available, either your computer or JMP is not configured to find the environment. See "To configure your JMP preferences".
3. Select the name of the environment from the Environment list if necessary.
4. Click **Connect**.
5. Enter your user name and password if prompted.

Connect to SAS on Your Local Machine from JMP (Windows Only)

You can also connect directly to SAS on your local machine.

To connect to SAS on your computer

1. Select **File > SAS > Server Connections** to open the SAS Server Connections window.
2. Under Establish New Connection, select **Connect to SAS on this machine**.

   This option is disabled if SAS is not installed on the computer.
3. Click **Connect**.
4. Click **Close** in the SAS Server Connections window.

To disconnect from SAS on your computer

1. In the SAS Server Connections window, select **Local** under Open Connections.
2. Click **Disconnect**.

Open SAS Data Sets with SAS Query Builder

SAS Query Builder is the preferred method for selecting and importing data from a SAS server. You can preview the data before importing it into a data table. You can also save the queries to modify and run later or to reference in a JSL script.

SAS Query Builder provides an alternative to opening SAS data sets with the File > SAS > Browse Data feature.

To open a SAS data set with SAS Query Builder, follow these steps:

1. Select **File > SAS > SAS Query Builder**.
2. In the Connect to SAS Server window, select a Metadata server or a remote server.
   - To connect to a Metadata server that you have already set up in JMP, select the server from the **Connect to metadata-defined SAS server** list.

     Connect to all available libraries is selected by default based on the SAS Integration preferences. This option connects metadata-defined SAS libraries automatically across all JMP sessions. When the Workspace Server contains a large number of metadata-defined SAS libraries, consider deselecting this option to speed up your connection to the server.
   - To add or configure a Metadata server, click **Manage Profiles** and follow steps in "To create or modify a SAS Metadata Server profile".
   - To connect to a remote server, enter the machine name and the port number. Your SAS Metadata Server administrator should have this information.
3. Click **OK**.

   The SAS Query Builder window appears.

For more information about using Query Builder, see "Build SQL Queries in Query Builder".

Notes:

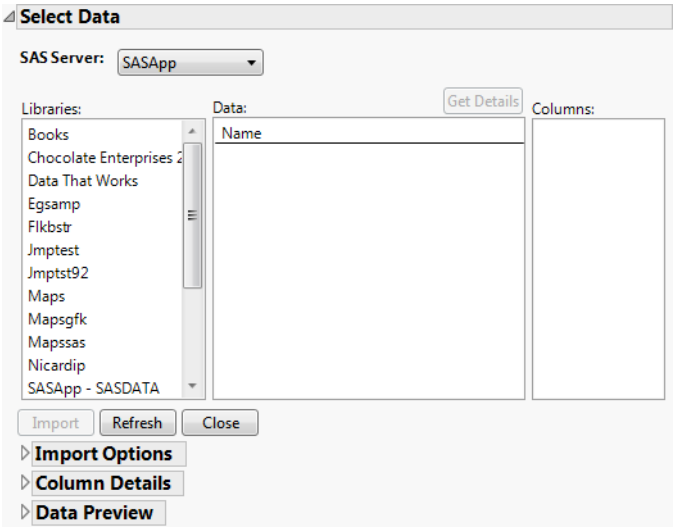
- All Query Builder queries run in the foreground.
- Extended attributes are not imported by default. To import them, modify the JMP SAS Integration preferences. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS). Select **SAS Integration** and then select **On import, apply table and column properties from extended attributes**.
- SAS Query Builder does not support local server connections.

Open SAS Data Sets through a SAS Server

Once you connect to a SAS Workspace Server, you can browse through the SAS libraries on that server and import data into JMP.

To browse the data sets on the SAS server, select **File > SAS > Browse Data**. The Browse SAS Data window appears.

Figure 3.56 Browse SAS Data



The window is initially populated with a list of servers the SAS Metadata Server provides (if connected). Any physical and local connections are also shown (as listed in Figure 3.55).

- Select a server to see a list of libraries that server contains.
- Select a library to see a list of data sets within that library.
- Select a data set to see a list of columns within that data set.

When you close and reopen the Browse SAS Data window, the previously viewed library and data set appear in the window. However, at any time, you can select a different server from the SAS Server list and then select a library and data set.

Tip: If a server is unavailable, or if the connections failed, the server's name is shown in light, italic text. Click it to try to re-establish the connection.

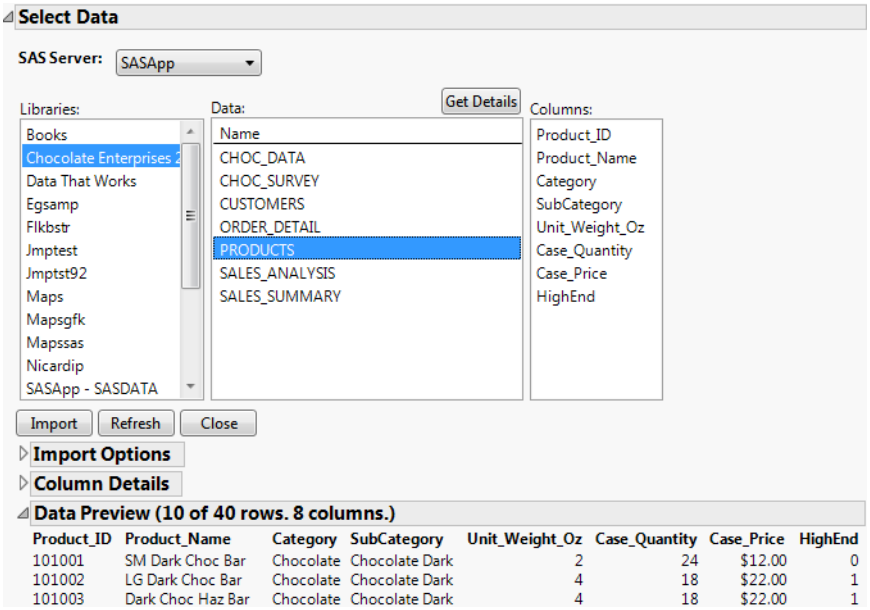
Browse SAS Data Information

You can select a SAS data set and see information about its contents before opening it using the Get Details, Column Details, and Data Preview options.

Data Preview

When you select a data set, the Data Preview outline shows you the first ten rows and columns in the data set.

Figure 3.57 Data Preview



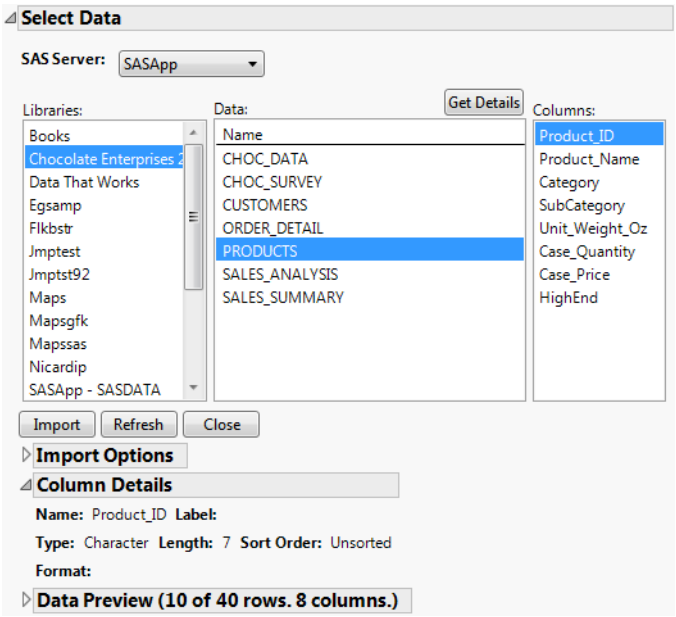
Data Set Details

Click **Get Details** in the Browse SAS Data window to see the size and last modification date for each data set in the library. This option helps you estimate whether your computer can process the entire data set.

Column Details

To see information about a particular column in the data set, select it. The Column Details outline shows you some basic information about the data column.

Figure 3.58 Column Details



Name Column name from the SAS data set.

Label Descriptive column label. The label can be longer than the name, and is often helpful to determine what the column name means.

Type Specifies whether the column has a character or numeric data type.

Length The length in bytes of data in the column.

Sort Order How the column is sorted in SAS.

Format The format for the SAS column, such as DOLLAR. This format field also contains information about the width of formatted values and the number of decimal places.

Open a SAS Data Set

You can import SAS data sets directly into JMP.

1. From the Browse SAS Data window, select a data set.

   By default, JMP specifies **All rows** for import.

2. Click **Import**.

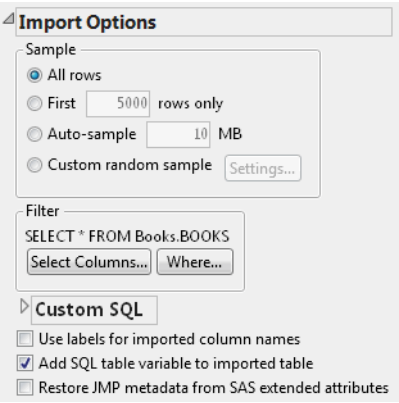
The entire SAS data set is imported into a JMP data table. When SAS data is imported, JMP attempts to make the best match to the SAS format.

Import a Sample of a SAS Data Set

You can import a sample of a SAS data set directly into JMP.

1. From the Browse SAS Data window, select a data set.
2. Open the Import Options outline.

Figure 3.59 Import Options



3. If you want to import only a portion of a data set, you can do any of the following:
   - Select the first *x* number of rows only. See "To import the first x number of rows only".
   - Select to auto-sample a specified file size. See "To import an auto-sample file of a specified size".
   - Select a subset of the columns. See "To select a subset of columns".
   - Construct a WHERE clause to filter the data. See "To import using a WHERE clause".
   - Take a custom sample of the data. See "Importing a Random Sample of the Data".

The following options are also available:

Use labels for imported column names Converts SAS labels to column names.

Add SQL table variable to imported table Saves the SQL table variable as a data table variable.

Restore JMP metadata from SAS extended attributes Includes JMP metadata (table and column properties) that were saved in the SAS data set as extended attributes.

To import the first x number of rows only

1. In the Import Options section, select **First x rows only** and specify the number of rows to import.
2. In the Browse SAS Data window, click **Import**.

JMP imports the specified number of rows.

To import an auto-sample file of a specified size

1. In the Import Options section, select **Auto-sample** and specify the number of MB to import.
2. In the Browse SAS Data window, click **Import**.

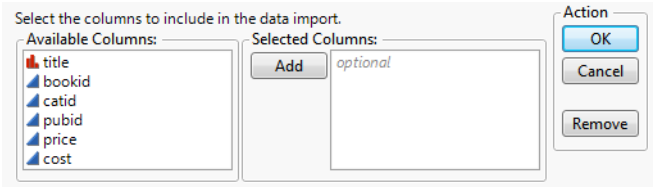
   JMP imports the specified number of MB.

To select a subset of columns

1. In the Import Options section, click **Select Columns**.

   The Select Columns window appears.

Figure 3.60 Select Columns



2. Select the columns that you want to import.

   To select more than one column at a time, press Ctrl and click each column.
3. Click **Add**.
4. When you have added all the columns that you want, click **OK**.
5. In the Browse SAS Data window, click **Import**.

Only the columns that you selected from the SAS data set are imported into a JMP data table.

To import using a WHERE clause

1. Click **Where**.
2. Use the WHERE clause editor to construct your WHERE clause.
3. Click **OK** to return to the Browse SAS Data window.
4. Click **Import**.

Only the data that matches your WHERE clause are imported into a JMP data table.

For information about constructing WHERE clauses and using the WHERE clause editor, see "Use the WHERE Clause Editor".

Note: If you import data using both a WHERE clause and sampling, the WHERE clause is applied first, and then a sample of the filtered data is taken.

You can also write your own SQL statements.

To import using a custom SQL statement

You can also open a SAS data set using a custom SQL statement.

1. Open the Custom SQL outline under the Import Options outline.

Figure 3.61 Custom SQL

2. Enter your SQL statement in the window.
3. Click **Execute Custom SQL**.

Note: Your SQL is run on the selected server but is not restricted to any selected library or data set.

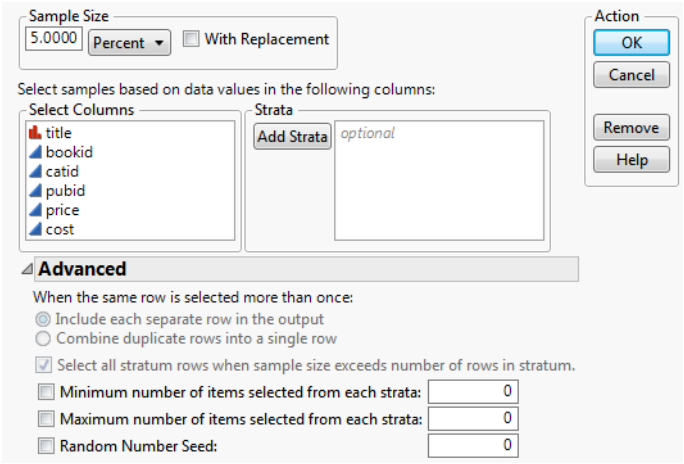
Importing a Random Sample of the Data

You can also import a random sample of the rows of the SAS data set.

Note: The sampling feature requires that the SAS server has the SAS/STAT product licensed and installed. If SAS/STAT is not present, sampling is disabled.

In the Sample Imported Data area of the Import Options outline, select the **Custom random sample** check box. By default, 5% of the rows are imported. To change the random sample import settings, click the **Settings** button.

Figure 3.62 Sampling Settings



In this window, you specify any of the following:

Sample Size You can set the sample size be percentage or by number of rows. To ensure that each row is sampled only once, de-select the **With replacement** option. To ensure that any row can be sampled and appear more than once in the imported data, select the option.

Selecting by Column You can select strata by moving columns into the Strata list.

Handling Multiple Row Sampling If **With replacement** is selected, you can specify to either add each duplicated row as a separate row or combine all duplicated rows into one row. If the second option is selected, a column is added to the table that contains a count of how many times each row was sampled.

Setting minimum and maximum numbers of items selected Select the option and enter a number.

Setting the random number seed Select the option and enter a seed. Specifying the seed lets you reproduce the exact same sample multiple times.

Note: If you import data using both a WHERE clause and sampling, the WHERE clause is applied first, and then a sample of the filtered data is taken.

Import Options

There are additional options that you can use to specify how SAS data is imported into JMP.

Use labels for imported column names When selected, this option switches the column name, which has a limited length and might be difficult to decipher, with the column label. This option is turned off by default. To use the SAS data column names as column names in JMP, deselect this box.

Add SQL table variable to imported table When selected, this option adds SQL queries to the data table panel as a variable. This option is turned on by default. If you turn off this option, only two variables are added when you import the data table: the SAS server and the data set.

Restore JMP metadata from SAS extended attributes

Tip: If your data is password-protected, you might want to turn this option off, because your password might be shown in the SQL.

Table Variables

After you import the JMP data table, table variables appear in the upper left panel of the data table. These variables show the SAS server, data set, and the SQL query and sampling settings if applicable. There is also a source script added that lets you re-do the import at any time.

Open Password-Protected Data Sets

JMP can open SAS version 7 or higher data sets that are password protected. The passwords are not case sensitive.

To open password-protected data sets

1. Select **File > Open**.
2. Select **SAS Data Sets** from the **Files of type** list.
3. Select the file.
4. Click **Open**.
5. Enter the password and then click **OK**.

   When the password is incorrect, you are prompted to enter it again until you get it right.

Using SAS Extended Attributes to Import Metadata

SAS extended attributes are metadata that you define in SAS code to import information such as table scripts, labels, length, and type. You associate the extended attributes with a data set or variable and define them in name-value pairs, such as `_JMP_TABLESCRIPTNAME_2="OnOpen"`. See the SAS documentation on extended attributes at https://sas.com.

On export from JMP to SAS, items such as column properties and table scripts are preserved automatically. You map SAS extended attributes to JMP attributes only when importing SAS code into JMP.

Here is an example of defining extended attributes in SAS code. The attributes define a table script name and the table script itself.

```
/* specify two table scripts. */
%LET _DS_ATTRIBUTES=_JMP_TABLESCRIPTCOUNT=2
/* define the table script names */
_JMP_TABLESCRIPTNAME_1="Favorite Movie By State"
_JMP_TABLESCRIPTNAME_2="OnOpen"
_JMP_TABLESCRIPTVALUE_2="CurrentDataTable() << RunScript(""Favorite Movie By
   State"");"
/* define the OnOpen table script */
_JMP_TABLESCRIPTVALUE_1=
/* a portion of the OnOpen table script */
"
   Current Data Table();
   New Column(""max_name"",
      Character,
      Width(128)
   );
```

The following example shows how to combine PROC DATASETS with extended attributes:

```
%LET _DS_ATTRIBUTES = attr-name=attr-value...attr-name=attr-value;
%LET _VAR_ATTRIBUTES= var-name (attr-name=attr-value...)...var-name
   (attr-name=attr-value...);

PROC DATASETS NOLIST LIB=WORK;
  MODIFY AttributeReferenceTable;
  XATTR OPTIONS MAXCHUNK=100;
  XATTR SET DS &_DS_ATTRIBUTES;
  XATTR SET VAR &_VAR_ATTRIBUTES;
RUN;
QUIT;
```

See "Conventions for Mapping JMP Attributes to SAS Extended Attributes" for more information about SAS extended attributes and their corresponding JMP attributes.

Two SAS Integration preferences determine whether extended attributes are imported or exported. Select **On export, store table and column properties in extended attributes to export extended attributes**. Select **On import, apply table and column properties from extended attributes.** The options are deselected by default.

To see an example of exporting extended attributes, run the following script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
sd = dt << Make SAS Data Step Window( SaveJMPMetadata( 1 ) );
```

Run Stored Processes

Stored processes are SAS DATA step code saved on the SAS server that you are connected to. You can run them from JMP and see the results of the script.

Note: Depending on the preferences that you have set for SAS, error messages are sent either to the JMP log or to a separate SAS log window.

You must be connected to a Metadata Server to view and run stored processes. If you select **File > SAS > Browse SAS Folders** without such a connection, you are prompted to either make a connection or cancel your action.

To select and run a stored process

1. Select **File > SAS > Browse SAS Folders**.

   The Browse SAS Folders window appears.
2. Browse through the stored processes to find the one that you want to run.
3. Select it and click **Run**.

   The data opens as a JMP data table.

On Windows, you can also right-click a stored process and select **Copy Metadata Path**. This option copies the path to the clipboard. You can then paste it into a script window to include it as a parameter for the JSL operator `Meta Get Stored Process()`. See the *JSL Syntax Reference*.

Note: Static graphs might not appear in the results returned from a SAS stored process when streaming output is selected.

Stored processes send reports to HTML by default, but you can select RTF or PDF instead on the SAS Integration page of the JMP preferences. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS) to view the JMP preferences.

Submit SAS Code

You can submit SAS code directly from JMP to the currently active SAS server. If the submitted SAS code generates SAS Listing output, that output is automatically retrieved from the SAS server and displayed in JMP. Also, the generated SAS Log is retrieved, and, if there are any errors in the submitted code, the SAS Log is automatically displayed in the SAS Log window.

All analyses in JMP are run natively within JMP without any dependency on the SAS System. The SAS code that JMP generates is intended to enable you to perform a separate but similar analysis in the SAS System after the initial JMP analysis, or to score new observations in the SAS System using a model that was fit within JMP.

Figure 3.63 SAS Code Submission Example

SAS Code

```
data a;
x=1;

proc print;
run;
```

JMP Code to Submit SAS Code

```
SAS Connect("sasmain");
SAS Submit("data a; x=1; proc print; run;");
```

The following JMP platforms generate SAS code:

- Standard Least Squares - PROC GLM
- REML - PROC MIXED
- Stepwise - PROC GLM
- Nominal and Ordinal Logistic - PROC LOGISTIC
- GLM - PROC GENMOD
- Time Series (ARIMA and TFM) - PROC ARIMA
- Neural - SAS Data Step scoring code
- Partition (Decision Tree, Bootstrap Forest, Boosted Tree) - SAS Data Step scoring code

Note: Use the JSL function `As SAS Expr( formula );` to turn any prediction formula into an expression that can be used in a SAS Data Step. See the *JSL Syntax Reference*.

To run SAS code directly from JMP

1. Either open an existing SAS program using **File > Open**, or create a new SAS program. (Create a new SAS program by selecting **File > SAS > New SAS Program** and typing in the SAS code.)
2. Click the **Submit to SAS** icon .

   You can also right-click in the Program Editor window and select **Submit to SAS**. The menu item also includes the name of the active SAS server that the SAS code will be submitted to.

   You can also press F8 (press Command+Shift+R on macOS).

To run SAS code using a JSL script

Write and run a JSL script that uses either the `SAS Submit` or `SAS Submit File` JSL functions. For more information about writing JSL scripts that submit SAS code, see the *Scripting Guide*.

To view the SAS Listing output

If the submitted SAS code generates SAS Listing (textual) output, that output is automatically be displayed in a SAS Output window when the job is completed. If you need to view the SAS Listing output again later in the JMP session, select **File > SAS > Open SAS Output Window**. The SAS Output Window retains the listing output from the previous 25 submits to the active SAS server.

To view the SAS log

If the submitted SAS code contained errors, the SAS Log window for the active SAS server is automatically opened, displaying the SAS Log for the job. However, you can view the SAS Log for the most recent 25 submits to the active server at any time by selecting **File > SAS > Open SAS Log Window**.

If you prefer that SAS Log information is appended to the JMP log after a submit completes:

1. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS).
2. Open the SAS Integration category.
3. In the Show SAS Log section, select **JMP Log** rather than **Separate Window**.

Also, in the Show SAS Log section, you can set whether the SAS Log should be displayed **Always**, **Never**, or **On Error** (the default).

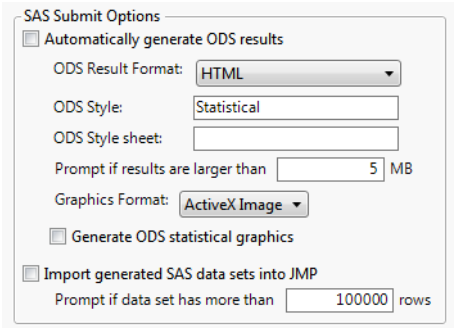
Generate ODS Results

The SAS Output Delivery System (ODS) is a powerful mechanism for generating reports in HTML, RTF, PDF, and other formats. ODS output is generally much more attractive and customizable than plain-text SAS Listing output. You can set your submitted SAS code generate ODS results rather than SAS Listing output using Preferences.

To generate ODS results from your submitted SAS code

1. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS).
2. Open the SAS Integration category and find the large SAS Submit Options group.

Figure 3.64 SAS Submit Options in Preferences



3. Select the **Automatically generate ODS results** option.
4. From the **ODS Result Format** list, select the format in which to generate the ODS results: HTML, PDF, RTF, or a JMP report.
5. (Optional) You can use other options to specify a style or style sheet to format the results or set the format for generated graphics. See "Preferences for SAS Integration".

Performing the previous steps causes JMP to generate additional SAS code, including an ODS statement, that is wrapped around the SAS code that you submit. The SAS code that you submit then automatically generates ODS results in the specified format. Those results are downloaded to your computer and displayed either within JMP, when possible, or in an appropriate external application.

Import Generated SAS Data Sets

SAS code that you submit might generate SAS data sets. You can have them automatically imported into JMP for further analysis.

1. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS).
2. Open the SAS Integration category.
3. Select the **Import generated SAS data sets into JMP** option.

Export JMP Data Tables to SAS

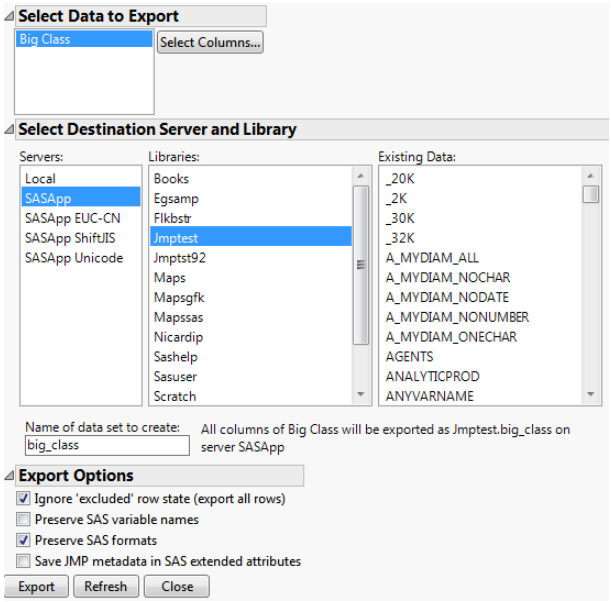
You can export JMP data tables to a SAS Workspace Server.

1. Connect to the SAS Workspace Server.
2. Open the file that you want to export.
3. Select **File > SAS > Export Data to SAS**.

   If necessary, you are connected automatically using your profile's user name and password.

4. Select the data table that you want to export to SAS from the list of open data tables under Select Data to Export.

Figure 3.65 Export Data to SAS



5. (Optional) To export only some of the columns in the data table, click **Select Columns**. See "To select columns to export".
6. Select the Destination Server.
7. Select the Library.

Tip: If your libraries do not appear, see "Show Libraries in the Export Data to SAS Window".

A list of the data sets in the library appears.

8. Enter the name as you want it to appear in the SAS library.
9. (Optional) Set the export options that you want to use. See "Export Options".
10. Click **Export**.

To select columns to export

1. To export only some of the columns in the data table, click **Select Columns**.
2. In the window that appears, select the columns to export and click **Add**.
3. When all the columns have been added to the Selected Columns list, click **OK**.

Export Options

Ignore 'excluded' row state (export all rows) Select this option to export all rows in the data table. Deselect this option to export only those rows that are not excluded. This option is on by default.

Preserve SAS variable names This option is useful for data tables that were imported originally from SAS. When importing a SAS data set, the original SAS variable name is saved in a column property for each column. Select this option to use the SAS variable name for each column when exporting to SAS. Deselect this option to export the JMP variable names. This option is off by default.

Preserve SAS formats This option is useful for data tables that were imported originally from SAS. When importing a SAS data set, the original SAS format and informat is saved in a column property for each column. Select this option to use the SAS format and informat for each column when exporting to SAS. Deselect this option to export the JMP formats instead. This option is on by default.

Show Libraries in the Export Data to SAS Window

If your libraries do not appear in the Export Data to SAS window, define the library in one of the following ways:

- Using JSL, submit code to the SAS server. The code defines a library using a `libname` command.
- Define an autoexec.sas file that runs a snippet of SAS code every time SAS is invoked. This creates the same `librefs` every time you connect to SAS. For more information about autoexec.sas files, see the SAS documentation.

Libraries that are defined in metadata (such as libraries defined in the SAS Management Console under the Data Library Manager) cannot be accessed from the Export Data to SAS window.

Import Data from a Database

You can import data from a database if you have an ODBC (Open Database Connectivity) driver for the database and then save the data back to the database.

You can use Structured Query Language (SQL) statements to control what you import from an external database. When you open a database file in JMP, you are actually sending a SQL statement to the database. By default, this statement gets all columns and records in the database table. See "Build SQL Queries in Query Builder" for more information about interactively building queries. See "Write SQL Statements to Query a Database" for more information about writing queries yourself.

This section describes how to connect to a database and import the data.

Notes:

- Database table names that contain the characters `$# -+/%()&|;?` are not supported.
- Multiple connections to a datasource are not permitted.
- Use a 64-bit version of the ODBC driver to match the bitness of JMP.
- SQL Native 11 and ODBC for SQL Server 17 don't return the column length. When a driver does not return a column length, the limit is 4MB.

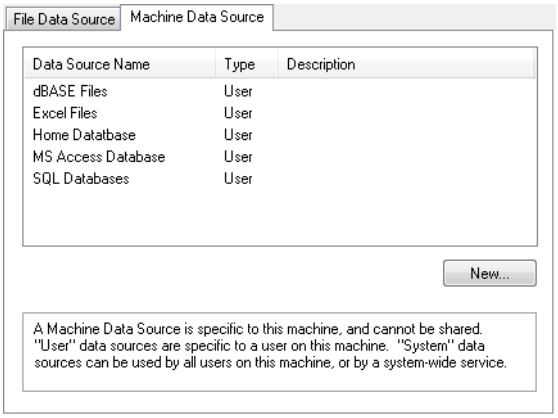
Connect to a Database

JMP can communicate with databases using ODBC data sources using an ODBC manager. The ODBC manager and associated drivers depend on your operating system. After you create the data source in the operating system software, follow these steps to connect to the database in JMP.

1. Select **File > Database > Open Table**. The Connections box lists data sources that you have connected to in the current JMP session.
2. Click **New Connection**.
3. (Windows) In the Select Data Source window (Figure 3.66), click the Machine Data Source tab, select the data source, click **OK**, enter the user name and password, and then click **OK.**

   (macOS) In the Choose DSN window, select the data source, enter the user name and password, and then click **Choose DSN**.

Figure 3.66 Select a Data Source (Windows)



The new connection is shown in the Database Open Table window.

Open Data from a Database

After you connect to the ODBC database and select a table to import, the data is opened in a data table. Several table scripts are included in the data table.

- Run the Source script to reconnect to the database.
- Run the Update from DB script to re-import and refresh the data. If the database table has the same number of columns as the original JMP table, the values in that existing JMP table are updated in place. However, if the number of columns is not the same, then a new data table opens where the updated data is stored.
- Run the Save to DB script to save the data table to the database. The existing data in the database is replaced. This script might contain the user name and password. The preference called `ODBC Hide Connection String` can be set to prevent including this possibly sensitive information. Select **File > Preferences > Tables** (Windows) to find the preference (or **JMP > Preferences > Tables** on macOS).

To import data from a database

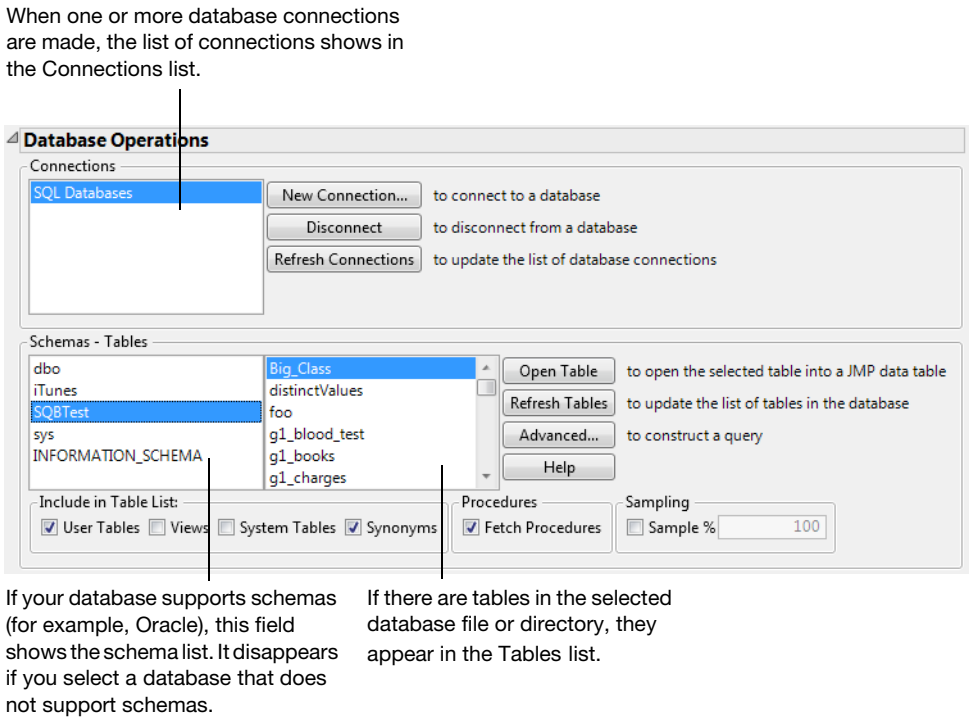
1. Select **File > Database > Open Table**.

   The Database Open Table window appears (Figure 3.67).

2. If you are already connected to the database, select it in the Connections box. Follow the steps in "Connect to a Database".

   The Connections box lists data sources to which JMP is connected. The Schemas - Tables box lists schemas for those databases that support them.

Figure 3.67 Database Open Table Window



Note: The **Fetch Procedures** check box is disabled if the ODBC driver does not support fetching procedures.

3. If the desired data source is *not* listed in the Connections box, click **New Connection** to choose a data source. The method of choosing a data source depends on your operating system.
4. Select the desired data source in the Connections box. The tables list in the Tables box updates accordingly. The update might take a several seconds, depending on the number of tables and the speed of the connection to the database. If your database supports schemas, tables are loaded for the first schema in the list, and on other schemas as you click them.
5. Control which tables are listed by choosing the options in the **Include in Table List** group of check boxes.

Note: Different drivers interpret these labels differently.

User Tables When clicked, displays all available user tables in the **Tables** list. User tables are specific to which user is logged on to the computer.

Views When clicked, displays "views" in the **Tables** list along with all other file types that can be opened. "Views" are virtual tables that are query result sets updated each time you open them. They are used to extract and combine information from one or more tables.

System Tables When clicked, displays all available system tables in the **Tables** list. System tables are tables that can be used by all users or by a system-wide service.

Synonyms When clicked, displays all available ORACLE synonyms in the **Tables** list.

Sampling Enter the percentage of rows that you want to appear in the list of tables. Selecting this option speeds up queries in large databases. JMP uses the sampling method supported by the database. The check box is unavailable when the database does not support sampling.

6. Select the desired table from the Tables list.

Note: If you are connected to a dBase database, select the database *folder* to which you would like to connect. Individual files are grayed out and cannot be selected.

7. Click **Open Table** to import all the data in the selected table, or click **Advanced** to specify a subset of the table to be imported. Some databases require that you enter the user ID and password to access the data.

   You might see a short delay when opening large tables. To see the status of all active ODBC queries, select **View > Running Queries.**

Note: If the data were previously exported to a database in JMP and contained an Expression column, the column will be imported as a Character column. Select **Cols > Column Info** and change the Data Type to Expression.

Write SQL Statements to Query a Database

You can use Structured Query Language (SQL) statements to control what you import from a database. When you open a database file in JMP, you are actually sending a SQL statement to the database. By default, this statement gets all columns and records in the database table. In some cases, this is too much data. When you are interested only in a subset of the table's data, you can customize the SQL request to only request the data that you want. After you execute a SQL query, the code for the query is stored in the data table in the SQL table variable.

This section describes how to write SQL statements to retrieve data. To interactively query data without writing SQL statements, use Query Builder. You can also start creating a query in Query Builder and then add your own SQL. See "Write SQL Statements in Query Builder".

1. Select **File > Database > Open Table**.

The Database Open Table window appears (Figure 3.67).

2. Connect to the database if necessary or select an existing database connection. Follow the steps in "Connect to a Database".

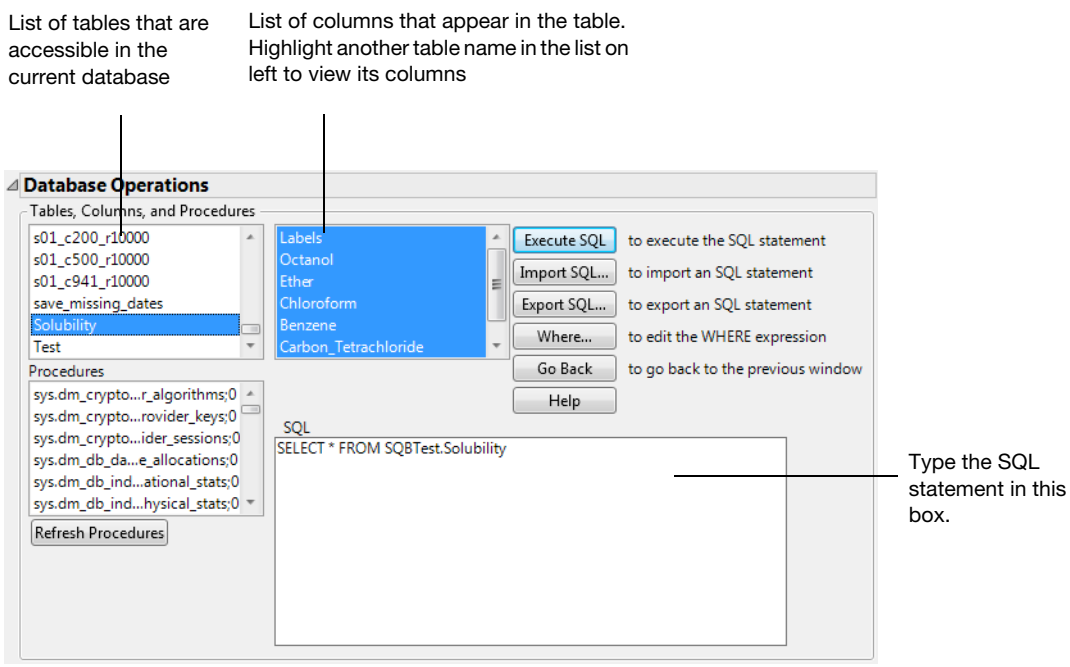
   The Connections box lists data sources to which JMP is connected. The Schemas - Tables box lists schemas for those databases that support them.

   Note: The SQL Query that you run in this window operates only on the tables and procedures that are displayed in the left panes of the window. Running unrelated SQL here has no results.

3. From the Database Open Table window, click the **Advanced** button to open specific subsets of a table.
4. Either type in a valid SQL statement, or modify the default statement. Figure 3.68 shows a default SQL `Select` statement appropriate for the selected file. See "Structured Query Language (SQL): A Reference", for a description of SQL statements that you can use.

   Instead, you can add expressions by clicking the **Where** button and using the WHERE Clause editor to create expressions. See "Use the WHERE Clause Editor".

Figure 3.68 Reading All Variables from the Solubility Table Stored in an Excel File



5. Click **Execute SQL**. A JMP data table appears with the columns that you selected. See "Use Data Table Variables".
6. To see the status of all running queries, select **View > Running Queries.**

Note that you can enter any valid SQL statement and click **Execute SQL** to execute the command. Valid SQL varies with the data source and ODBC driver.

Structured Query Language (SQL): A Reference

The following sections are a brief introduction to SQL. They give you insight to the power of queries, and they are not meant to be a comprehensive reference.

Use the SELECT Statement

The fundamental SQL statement in JMP is the SELECT statement. It tells the database which rows to fetch from the data source. When you completed the process in "Write SQL Statements to Query a Database" with the Solubility.jmp sample data table, you were actually sending the following SQL statement to your data source:

```
SELECT * FROM "Solubility"
```

The * operator is an abbreviation for "all columns." So, this statement sends a request to the database to return all columns from the specified data table.

Rather than returning all rows, you can replace the * with specific column names from the data table. In the case of the Solubility data table example, you could select the ETHER, OCTANOL, and CHLOROFORM columns only by submitting this statement:

```
SELECT ETHER, OCTANOL, CHLOROFORM FROM "Solubility"
```

Note: JMP does not require you to end SQL statements with a semicolon.

JMP provides a graphical way of constructing simple SELECT statements without typing actual SQL. To select certain columns from a data source, highlight them in the list of columns.

To highlight several rows

- Press Shift and then click to select a range of column names
- Press Ctrl and then click (Windows) or press Command and then click (macOS) to select individual column names.

Note that the SQL statement changes appropriately with your selections.

Sometimes, you are interested in fetching only unique records from the data source. That is, you want to eliminate duplicate records. To enable this, use the DISTINCT keyword.

```
SELECT DISTINCT ETHER, OCTANOL, CHLOROFORM FROM "Solubility"
```

Sort Results

You can have the results sorted by one or more fields of the database. Specify the variables to sort by using the `ORDER BY` command.

```
SELECT * FROM "Solubility" ORDER BY LABELS
```

selects all fields, with the resulting data table sorted by the `LABELS` variable. If you want to specify further variables to sort by, add them in a comma-separated list.

```
SELECT * FROM "Solubility" ORDER BY LABELS, ETHER, OCTANOL
```

Use the WHERE Statement

With the `WHERE` statement, you can fetch certain rows of a data table based on conditions. For example, you might want to select all rows where the column `ETHER` has values greater than 1.

```
SELECT * FROM "Solubility" WHERE ETHER > 1
```

The `WHERE` statement is placed after the `FROM` statement and can use any of the following logical operators.

Table 3.4 WHERE Operators

Operator	Meaning
=	Equal to
!= or < >	Not equal to
>	Greater than
<	Less Than
>=	Greater than or equal to
<=	Less than or equal to
NOT	Logical NOT
AND	Logical AND
OR	Logical OR

When evaluating conditions, `NOT` statements are processed for the entire statement first, followed by `AND` statements, and then `OR` statements. Therefore

```
SELECT * FROM "Solubility" WHERE ETHER > -2 OR OCTANOL < 1 AND CHLOROFORM > 0
```

is equivalent to

```
SELECT * FROM "Solubility" WHERE ETHER > -2 OR (OCTANOL < 1 AND CHLOROFORM >
  0)
```

Use the IN and BETWEEN Statements

To specify a range of values to fetch, use the IN and BETWEEN statements in conjunction with WHERE. IN statements specify a list of values and BETWEEN lets you specify a range of values. For example,

```
SELECT * FROM "Solubility" WHERE LABELS IN ('Methanol', 'Ethanol', 'Propanol')
```

fetches all rows that have values of the LABELS column Methanol, Ethanol, or Propanol.

```
SELECT * FROM "Solubility" WHERE ETHER BETWEEN 0 AND 2
```

fetches all rows that have ETHER values between 0 and 2.

Use the LIKE Statement

With the LIKE statement, you can select values similar to a given string. Use % to represent a string of characters that can take on any value. For example, you might want to select chemicals out of the Solubility data that are alcohols, that is, have the OL ending. The following SQL statement accomplishes this task.

```
SELECT * FROM "Solubility" WHERE LABELS LIKE '%OL'
```

The % operator can be placed anywhere in the LIKE statement. The following example extracts all rows that have labels starting with M and ending in OL:

```
SELECT * FROM "Solubility" WHERE LABELS LIKE 'M%OL'
```

Use Aggregate Functions

Aggregate functions are used to fetch summaries of data rather than the data itself. Use any of the following aggregate functions in a SELECT statement.

Table 3.5 SELECT Statement Functions

Function	Meaning
SUM( )	Sum of the column
AVG( )	Average of the column
MAX( )	Maximum of the column
MIN( )	Minimum of the column

Table 3.5 SELECT Statement Functions *(Continued)*

Function	Meaning
COUNT( )	Number of rows in the column

Some examples include:

- The following statement requests the sum of the ETHER and OCTANOL columns:

  ```
  SELECT SUM(ETHER), SUM(OCTANOL) FROM "Solubility"
  ```

- This statement returns the number of rows that have ETHER values greater than one:

  ```
  SELECT COUNT(*) FROM "Solubility" WHERE ETHER > 1
  ```

- The following statement lets you know the average OCTANOL value for the data that are alcohols:

  ```
  SELECT AVG(OCTANOL) FROM "Solubility" WHERE LABELS LIKE '%OL'
  ```

Note: When using aggregate functions, the column names in the resulting JMP data table are Expr1000, Expr1001, and so on. You probably want to rename them after the fetch is completed.

The GROUP BY and HAVING Commands

The GROUP BY and HAVING commands are especially useful with the aggregate functions. They enable you to execute the aggregate function multiple times based on the value of a field in the data set.

For example, you might want to count the number of records in the data table that have ETHER=0, ETHER=1, and so on, for each value of ETHER.

- SELECT COUNT(ETHER) FROM "Solubility" GROUP BY (ETHER) returns a single column of data, with each entry corresponding to one level of ETHER.
- SELECT COUNT(ETHER) FROM "Solubility" WHERE OCTANOL > 0 GROUP BY (ETHER) does the same thing as the above statement, but only for rows where OCTANOL > 0.

When using GROUP BY with an aggregate function of a column, include the column itself in the SELECT statement. For example,

```
SELECT ETHER, COUNT(ETHER) FROM "Solubility" GROUP BY (ETHER)
```

returns a column containing the levels of ETHER in addition to the counts.

Use Subqueries

Aggregate functions are also useful for computing values to use in a WHERE statement. For example, you might want to fetch all values that have greater-than-average values of ETHER. In other words, you want to find the average value of ETHER, and then select only those records that have values greater than this average. Remember that `SELECT AVG(ETHER) FROM "Solubility"` fetches the average that you are interested in. So, the appropriate SQL command uses this statement in the WHERE conditional:

```
SELECT * FROM "Solubility" WHERE ETHER > (SELECT AVG(ETHER) FROM
  "Solubility")
```

Save and Load SQL Queries

After constructing a query, you might want to repeat the query at a later time. You do not have to hand-type the query each time you want to use it. Instead, you can export the query to an external file. To do this, click the **Export SQL** button in the window shown in Figure 3.68. This brings up a window that lets you save your SQL query as a text file.

To load a saved query, click the **Import SQL** button in the window shown in Figure 3.68. This brings up a window that lets you navigate to your saved query. When you open the query, it is loaded into the window.

Use the WHERE Clause Editor

JMP provides help building WHERE clauses for SQL queries during ODBC import. It provides a WHERE clause editor that helps you build basic expressions using common SQL features, allowing vendor-specific functions. For example, you do not need to know whether SQL uses '=' or '==' for comparison, or `avg()` or `average()` for averaging.

In addition, string literals should be enclosed by single quotes (`'string'`)rather than double quotes (`"string"`).

To open the WHERE clause editor

1. Connect to a database by following the steps in "Connect to a Database".
2. From the **Database Open Table** window, click the **Advanced** button.
3. Click the **Where** button.

USE the WHERE Clause Editor to add any of the following from the work panel: expressions, functions, and terms. They are applied to the highlighted blue box.

1. Click the Table Name Browser to select a table. The columns in that table appear in the list.
2. Click the SQL Vendor Name Browser to select the type of SQL that you want to use: GenericSQL, Access, DB2, MySQL, Oracle, SQL Server, or all of the above. Perform an

action by clicking a function or operator in the list and selecting an operator from the list that appears.

Note: The following SQL Server data types are not supported: Binary, Geography, and Geometry.

3. Select an empty formula element in the formula editing area by clicking it. It is selected when there is a red outline around it. All terms within the smallest nesting box relative to the place that you clicked become selected. The subsequent actions apply to those combined elements.
4. Add operators to an expression by clicking buttons on the keypad.
5. (Optional) To customize your WHERE clause, select one of the options from the red triangle menu in the formula editor:

Show Boxing Show or hide boxes around the WHERE clause terms.

Larger Font Increase the font size of the formula.

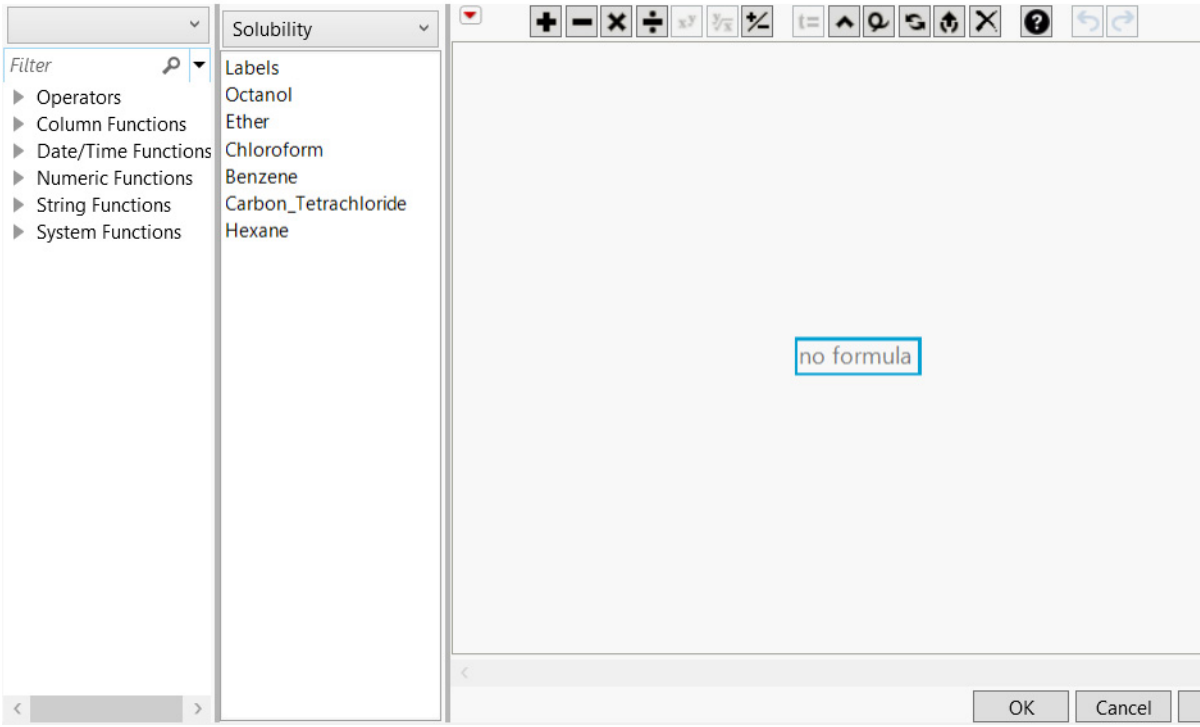
Smaller Font Decrease the font size of the formula.

Simplify Simply the WHERE clause statement as much as possible.

Reset panel layout Display the panels as shown in Figure 3.69.

The WHERE clause editor works similarly to the Formula Editor, which is described in "Create Formulas in JMP".

Figure 3.69 The WHERE Clause Editor



Import Data from Google Sheets

A Google Sheet is a spreadsheet (which consists of sheets, like tabs) that is saved in your Google account. This enables you to continue to edit the spreadsheet and to share it with others.

Import data from a Google spreadsheet by logging into your Google account, copying the URL for the spreadsheet, and specifying how the data is structured. For example, you must know which row the data begin on if there are column headers.

To import data from a Google spreadsheet, follow these steps:

1. Log in to https://sheets.google.com and open the spreadsheet.
2. Copy the URL in the browser's address bar.
3. In JMP, select **File > Internet Open > Google Sheets**.
4. Type your Google account name. (@gmail.com is unnecessary.)
5. Paste the URL in the spreadsheet URL or ID field.
6. Click **Import**.

The data are imported into a JMP data table.

If you are importing data using the specified Google account for the first time, you are prompted to give JMP access to your Google account.

a. Click **Allow**.

The data are imported into a JMP data table.

About Importing Google Sheets

- Information that Google uses to authenticate your account is stored in C:/Users/<user name>/AppData/Roaming/SAS/JMP/OAuthTokens.jmp on Windows. On Mac, oAuthTokens.jmp is stored in ~/Library/Application Support/JMP. (Passwords are not stored on the computer, and access tokens are encrypted.) Storing tokens in a data table enables you to import and export data using the same Google account without providing the authentication information again. Note that a user with access to your computer's user account has access to all of your Google spreadsheets. If the spreadsheets contain confidential information, reconsider sharing your user account.
- @gmail.com is not required for your account name. However, if you do use it, *<user name>*@gmail.com and *<user name>* are treated as separate accounts. You must enter the authentication code for both accounts when importing or exporting.
- If other users have access to your computer's user account, they can copy OAuthTokens.jmp to their computer and import your spreadsheets.
- If **Display intranet sites in Compatibility View** in Internet Explorer on Windows is selected, the Google Sheet cannot be imported. To avoid this issue, follow these steps:
  1. In Internet Explorer, select the Tools button and then select **Compatibility View settings**.
  2. Deselect **Display intranet sites in Compatibility View**.

Note: In Google Sheets, you can select **File > Publish to the web** to publish the sheet for an audience to view. In this case, the link to the published sheet includes the pubhtml suffix. Use File > Internet Open to open these sheets. See "Import Data from the Internet or a Remote Computer".

Google Sheet Settings

Change the following settings, if necessary, and then click **Import** to import the sheet data.

Sheet contains column headers Select if the worksheet contains rows with column headers.

Column headers start on row Indicates which row the column headers begin on in the worksheet. Click the up arrow until the headers begin on the correct row, or enter the row number and press Enter.

Data starts on row Indicates which row the data start on in the worksheet.

Select Sheets to Open Deselect any sheets you don't want to import by pressing Ctrl and clicking a sheet.

Advanced Options

Cell Range Imports cells within the specified range (for example, G5:G17).

Import cell colors Applies the cell coloring from the worksheet to the data table.

Suppress empty columns Prevents empty columns from appearing in the data table.

Import SPSS Files

JMP opens SPSS files as data tables and maintains several SPSS features:

- General numeric and character data with minimal formatting are supported.
- SPSS date, datetime, and time formats are supported.
- By default, labels are converted to column headings. When you select this option, and the data contains no labels, the columns are named *Column 1*, *Column 2*, and so on.

  You also have the option of selecting the conversion method for column headings when opening an SPSS file. The method that you select then overrides the preferences.

  To change the default conversion method, select **File > Preferences** (or **JMP > Preferences** on macOS). On the General page, deselect **Use SPSS labels for column names during import**. Variable names are then imported automatically as column headings.
- The value labels that you defined in the SPSS file are saved as Value Labels column properties. The value label then appears in each data table cell instead of the original value. For more information about Value Label properties, see "Value Labels".
- SPSS can assign certain values in a variable to be treated as missing for analyses. For example, the value *64* could be regarded as missing for a Height variable. Then, the calculation of the distribution of height would ignore values of *64*. When you import SPSS, these values are included in the Missing Value Codes column property for the appropriate variable.

  Missing value ranges of up to 20 numbers are supported. If the range begins with a negative number, the numbers count down to the maximum value (for example, -10

through -5). If the range begins with a positive number, the numbers count up from the minimum value (for example, 1 through 12).

Custom currency formats selected in an SPSS file are not maintained on import. In addition, JMP does not read SPSS data that contains double-byte characters, such as non-Unicode Japanese characters.

To open an SPSS file (Windows)

1. Select **File > Open**.
2. From the list next to **File name**, select **SPSS Data Files (*.sav)**.
3. Select the SPSS file.
4. (Optional) To specify the column headings, select one of the following **Set JMP column names from** options:
   - **SPSS Labels** creates column headings from SPSS labels.
   - **SPSS Variable Names** creates column headings from variable names.
5. Click **Open**.

   JMP opens the file as a data table.

To open an SPSS file (macOS)

1. Select **File > Open**.
2. Select the SPSS file.
3. (Optional) To specify the column headings, do one of the following
   - Deselect **Use SPSS Labels as Headings** to convert variable names to column headings.
   - Select **Use SPSS Labels as Headings** to convert labels to column headings.
4. Click **Open**.

   JMP opens the file as a data table.

Notes:

- As with importing other files, you might experience a delay when opening and saving large SPSS files.
- A pipe (|) now separates words within a multiple response in the .sav file's SPSS label (as in "detergent|liquid"). To change the delimiter, run the following JSL script: `Preference(SPSSMultiResponseDelimiter(":"))`, where the colon is the new delimiter.

Import Triple-S Survey Data

Triple-S (SSS) is a specification for survey data. The survey data is stored in a pair of files: .xml or .sss, and a .csv, .dat, or .asc file. The .xml or .sss file defines the variables and describes the survey. The accompanying .csv, .dat, or .asc file contains the data gathered from each respondent.

Variable labels are converted to column headings by default. To convert variable names to column headings, deselect **Use Triple-S Labels as Headings** in the JMP General preferences.

To import a Triple-S file, follow these steps:

1. Place the pair of Triple-S files in the same folder and use the same root name for both files.
2. Select **File > Open**.
3. On Windows, select **Triple-S Survey Files** from the File name list.
4. Select the SPSS file.
5. (Optional on Windows) To override the Triple-S column heading preference, select Variable Labels or Variable Names next to **Set JMP column names from:**.
6. Click **Open**.

   JMP opens the file as a data table.

Note: The default encoding is UTF-8. Triple-S files that are encoded as Windows-1252 will be imported as ISO-8859-1.

Import HDF5 Data

Hierarchical Data Format, Version 5 (HDF5) is a portable file format for storing data. An HDF5 file consists of groups and datasets. When you import the file, JMP opens a group to present the names of the inner datasets. For example, if the file contains data for a group of musical recordings with songs and artists tables, JMP prompts you to open either “songs” or “artists”. If the file does not contain a group, JMP prompts you to open a single table.

JMP handles only tables with numeric (integer, float, double) and string types, and compound files with three or fewer dimensions that contain only simple types. If a data set does not appear in the import window, the file is most likely unsupported.

You can import up to 1,000,000 columns and an unbounded number of rows.

Note: Only ASCII characters are supported in HDF5 files. Non-ASCII characters such as French and Chinese characters are not supported.

To import an HDF5 file, follow these steps:

1. Select **File > Open**.
2. On Windows, make sure that **All JMP Files** or **HDF5 Data File** is selected from the File name list.
3. Select the HDF5 file.

   A list of groups or datasets in the file appears.
4. Select the data set that you want to import and click **Import**.

   The data are opened as data tables.

Import Data from MATLAB

JMP provides a scripting interface to MATLAB. See the *Scripting Guide* and the *JSL Syntax Reference*.

Import Data from R

JMP provides a scripting interface to R. See the *Scripting Guide* and the *JSL Syntax Reference*.

Import Data Using the Excel Add-In

The add-in for Excel provides new capabilities to JMP and Excel users on Windows:

- Transfer selected cells in Excel to JMP data tables. See "Transfer Excel Data to a Data Table".
- Use the JMP Profiler with calculation models in Excel workbooks. The profiler tool is designed to bring the power of the JMP profiler to models residing in Excel worksheets. You do not have to recreate your Excel models in JMP, verify that they are correct, and maintain the model in both JMP and Excel. See *Profilers*.

  Note: During the JMP installation, select the Excel Add-In. This installs the add-in for your version of Microsoft Excel.

- Microsoft Excel 2010, Microsoft Excel 2013, Microsoft Excel 2016, Microsoft Excel 2019, and Office 365 Desktop Excel are supported.

Transfer Excel Data to a Data Table

You can use the JMP Add In for Excel to transfer a worksheet from Excel to the following JMP destinations:

- a data table
- Graph Builder
- Distribution platform
- Fit Y by X platform
- Fit Model platform
- Time Series platform
- Control Chart platform

You can also create models for profiling in JMP. See *Profilers*.

To transfer data from Excel to a JMP data table and platform

1. In your Excel worksheet, show the **JMP** add-in on the ribbon.
2. Click the **Preferences** button.
3. Accept the default **Data Table Name** (File name_Worksheet name) or type a name.
4. Select **Use the first rows as column names** if the first row in the worksheet contains column headers.
5. If you selected to use the first rows a column headers, type the number of rows used.
6. Select **Transfer Hidden Rows** if the worksheet contains hidden rows to be included in the JMP data table.
7. Select **Transfer Hidden Columns** if the worksheet contains hidden columns to be included in the JMP data table.
8. Click **OK** to save your preferences.
9. Select the cells to transfer into JMP, including any cells that you want to use as column names.

   If you are using cells as column names, they need to be the first rows in your selection.

   If only one cell (or no cell) is selected, the entire Excel worksheet is transferred to JMP.
10. Select the JMP destination from the toolbar:
    - Data Table
    - Graph Builder
    - Distribution platform
    - Fit Y by X platform

– Fit Model platform
– Time Series platform
– Control Chart platform

JMP opens, and the selected data is placed in a new JMP data table and the selected launch window appears.

Notes:

- Empty cells are imported as missing data, and dates, numbers, and strings are recognized correctly.
- Your JMP windows might be hidden behind your Excel window, especially if you maximize Excel.

For more information about using the various JMP platforms, refer to the proper JMP documentation.

About the JMP Add-In for Excel

The JMP add-in commands are in two groups:

Transfer to JMP

Preferences Set preferences for transferring data from Excel to JMP.

Data Table Transfer the selected data in your Excel file to a JMP data table.

Graph Builder Transfer the selected data in your Excel file to a JMP data table and launch the Graph Builder platform.

Distribution Transfer the selected data in your Excel file to a JMP data table and launch the Distribution platform.

Fit Y By X Transfer the selected data in your Excel file to a JMP data table and launch the Fit Y by X platform.

Fit Model Transfer the selected data in your Excel file to a JMP data table and launch the Fit Model platform.

Time Series Transfer the selected data in your Excel file to a JMP data table and launch the Time Series platform.

Control Chart Transfer the selected data in your Excel file to a JMP data table and launch the Control Chart platform.

Profile in JMP

Create/Edit Model Set up preferences for using the JMP profiler with Excel data.

Run Model Run the JMP profiler.

Uninstall the JMP Excel Add-In

To uninstall the Excel Add-in, double-click the JMP installer, click **Modify**, and deselect **Excel Add-In**.

Import Data from the Internet or a Remote Computer

Import data from the Internet, FTP sites, or other computers by selecting **File > Internet Open**. A file path can start with *http*, *ftp*, or *file*; a drive letter; or, the path to a network drive (relative or absolute).

Using Internet Open, you can import data and images from the Internet or a remote computer and save it as a data table, web page, or text. You can also open a SAS stored process report using this option. See "Import Data from Google Sheets" for more information about using Internet Open to import a Google spreadsheet.

Suppose that you want to import data (with images) from a table on a web page and save it as a data table. The specific web page lists countries by population.

1. Select **File** > **Internet Open > Web Page**.
2. Enter the following URL: https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population.
3. Keep **Open As** set to **Data**, and click **OK**.

   JMP finds several tables on the specified page.
4. Click **Rank country (or dependent territory)**... and click **OK**.

   JMP opens a data table that contains the information from the web page table.
5. (Optional) To add the images to the data table, run the *Load pictures* script.

Notes:

- You can open a remote file as one of the following formats:

  Data Opens the file as a data table.

Web page Opens the web page in the JMP browser (Windows) or the default browser (macOS).

From the File menu on Windows, you can then choose to save the file or import the data as a data table. Select this option to import data that is generated by web page scripts and server-side requests.

Text Opens the file in the script editor. If you imported an HTML file, the HTML tags of a web page are displayed.

- When you open a file from an FTP server, an FTP login window appears. For an anonymous account, click **OK**. For an authenticated login, enter your user ID and password. The file then opens as you specified.

Note: Some anonymous FTP servers require a user ID. If the data table does not open, try typing either "ftp" or "anonymous" in the User ID text box. Leave the Password text box empty and click **OK**.

- JMP waits 60 seconds before stopping the import due to an error. You can change the Internet Time Out setting on the General preferences page.
- If the table that you are importing contains images, the images are first imported as text. To display the images in your data table, run the automatically generated table script named *Load pictures*. A new expression column that contains the images is added to the data table.
- Running the Source script in a data table enables you to re-import and refresh the data.

To open a SAS stored process report as a data table (Windows only)

1. Select **File** > **Internet Open** and select the file.

   The file opens in the JMP browser.
2. In the JMP browser, select **File > Import Table as Data Table**.

   A window appears that lists the tables found in the web page.
3. Select the table or tables that you want to import.
4. Click **OK**.

   Each table is opened as a new data table.

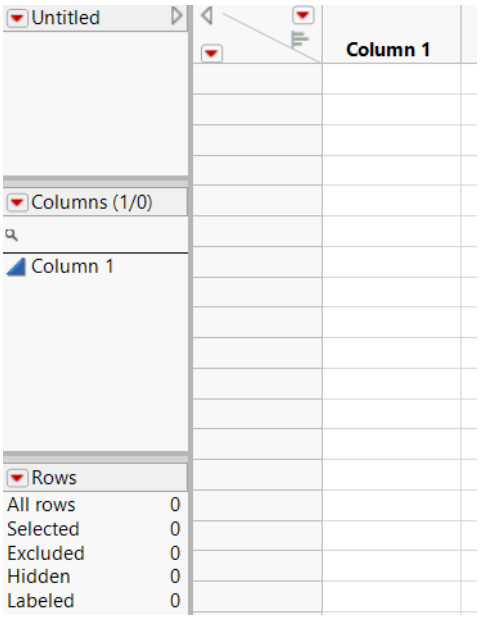
See "Run Stored Processes" for more information about stored process reports.

Create New Data Tables

To create a new data table by entering data manually, follow these steps:

1. Select **File > New > Data Table**. This shows an empty data table with no rows and one numeric column, labeled Column 1.
2. Move the cursor onto a cell.
3. Click in the cell. The cursor appears as a line in the cell.

Figure 3.70 A New Data Table



4. Enter a value.

There are several ways to fill a table with values:

- Create new rows and columns and type or paste data into the data grid. (See "Enter Data in Data Tables".)
- Construct a formula to calculate column values. (See "Build Formulas".)
- Import data from another application. (See "About Importing Data".)
- Copy values from another application and paste them into the table.
- Use a measuring instrument to read external measures. See in the *Scripting Guide* for more information about data feeds.
- Drag columns from one table to another.

See “Enter and Edit Your Data” for more information about how to format, edit, and work with data tables.

Chapter 4

Enter and Edit Your Data

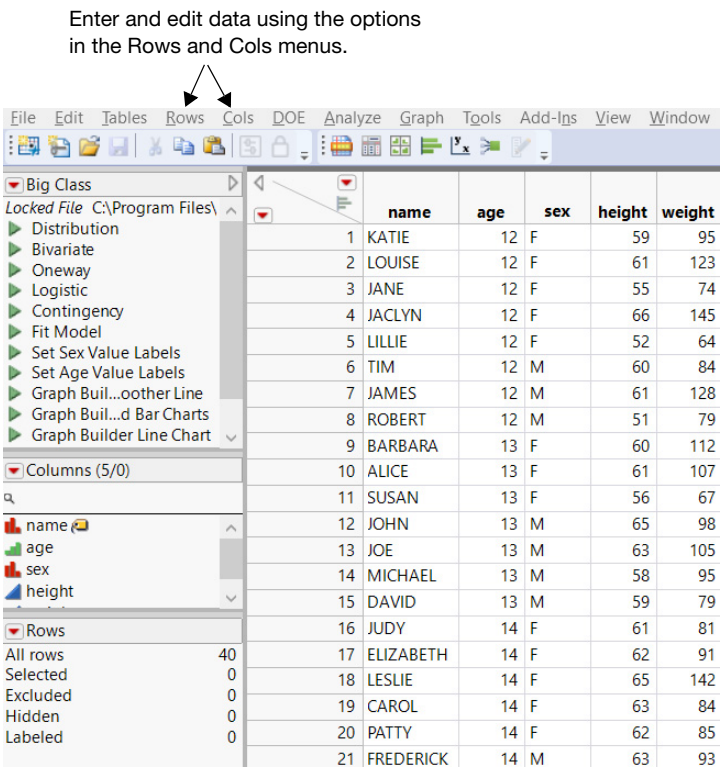
Perform Basic Data Table Tasks

After you import data into JMP or create a new data table, you can format your data to prepare it for analysis.

This chapter contains the following information:

- Change formatting for numeric values
- Add, delete, and select rows and columns
- Use the Row Editor to navigate within rows and edit rows
- Create scripts that are saved to the data table

Figure 4.1 The Rows and Cols Menus



	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107
11	SUSAN	13	F	56	67
12	JOHN	13	M	65	98
13	JOE	13	M	63	105
14	MICHAEL	13	M	58	95
15	DAVID	13	M	59	79
16	JUDY	14	F	61	81
17	ELIZABETH	14	F	62	91
18	LESLIE	14	F	65	142
19	CAROL	14	F	63	84
20	PATTY	14	F	62	85
21	FREDERICK	14	M	63	93

Contents

Enter Data in Data Tables . . . 192
Copy and Paste Data into Data Tables . . . 192
Add Rows to Data Tables . . . 193
Add Columns to Data Tables . . . 193
Fill Data Table Columns with Sequential Data . . . 195
Replace Missing Values with Previous Values in Data Table Columns . . . 196
Enter Cell Formulas into Data Table Columns . . . 197
Select Rows in Data Tables . . . 198
Find Matching Cells in Rows . . . 200
Select Rows That Contain Specific Values . . . 201
Locate the Next Selected Row . . . 202
Select Columns in Data Tables . . . 203
How to Select Columns . . . 203
Select Columns in the JMP Columns Viewer . . . 204
Locate Next and Previously Selected Columns in Data Tables . . . 210
Resize Rows and Columns . . . 210
Organize Data in Data Tables . . . 211
Delete Rows in Data Tables . . . 211
Delete Columns in Data Tables . . . 212
Rearrange Columns in Data Tables . . . 212
Group Columns in Data Tables . . . 213
Move Values in Data Tables . . . 213
Color Cells in Data Tables . . . 214
Edit or Delete Cells in Data Tables . . . 214
Edit Column Names in Data Tables . . . 214
Hide and Exclude Rows in Data Tables . . . 215
Exclude Rows in Data Tables . . . 216
Hide Rows in Data Tables . . . 217
Exclude Columns in Data Tables . . . 218
Hide Columns in Data Tables . . . 218
View Patterns of Missing Data in Data Tables . . . 219
Find and Replace Cell Values in Data Tables . . . 221
Edit Cells in Data Tables . . . 223
Right-Click Menus for Rows and Columns . . . 225
Compare Data Tables . . . 225
Assign Characteristics to Rows and Columns . . . 232
How to Find Options for Rows and Columns . . . 232
Label Rows and Columns . . . 232
Assign Colors or Markers to Rows . . . 233

Create Color Themes in Graphs . . . 235
Delete Custom Color Themes in Graphs . . . 241
Select Matching Row State Cells . . . 241
Delete Row Characteristics . . . 242
Lock Columns . . . 242
Restructure Data . . . 242
Split a Data Table Column into Multiple Columns . . . 243
Create a New Column by Text Matching . . . 243
Make Indicator Columns . . . 245
Combine Columns in Data Tables . . . 246
Compress Selected Columns in Data Tables . . . 247
Make a Binning Formula in a Data Table Column . . . 248
Convert Labels to Codes and Codes to Labels . . . 253
Make a New Formula Column . . . 254
Transform Columns in a JMP Platform . . . 254
Recode Data in a Column . . . 264
Edit Data Tables . . . 281
Rename Data Tables . . . 281
Lock Data Tables . . . 281
Compress Data Tables . . . 282
Copy Table Scripts . . . 282
Use Data Table Variables . . . 283
Create and Save Scripts in Data Tables . . . 284

Enter Data in Data Tables

This section describes how to add rows and columns, fill columns with sequential data, and enter cell formulas.

- "Copy and Paste Data into Data Tables"
- "Add Rows to Data Tables"
- "Add Columns to Data Tables"
- "Fill Data Table Columns with Sequential Data"
- "Replace Missing Values with Previous Values in Data Table Columns"
- "Enter Cell Formulas into Data Table Columns"

Tip: Consider setting the Autosave Timeout value in the General preferences to automatically save open data tables at the specified number of minutes. This autosave value also applies to journals, scripts, projects, and reports.

Copy and Paste Data into Data Tables

When you paste data from another document into a data table, JMP adds rows and columns as needed. Click a cell and paste the data.

Multiple Lines of Data

When you copy one data table cell and paste several lines of data, each line is pasted into its own cell.

To paste all lines into one cell, do one of the following:

- Double-click an existing cell first and then paste.
- Right-click an existing cell and select **Paste into Cell**. The Paste into Cell option is available only on character and expression columns.

Note: If you try to paste into a cell that does not exist, the text will paste into separate cells.

The data type of the copied content needs to match the data type of the cells that you're pasting into. You can add new columns and paste the data, paste the data into cells of the correct data type, or change the data type of the cells if necessary.

Note: To copy the entire numeric value, select **Edit > Copy With Full Precision**. The option applies only to the fixed decimal format.

Add Rows to Data Tables

To add any number of rows to the data table

1. Select **Rows > Add Rows**.
2. Enter the number of rows to add.
3. Specify where to add the new rows (at the start or end of the data table, or after a specific row).
4. Click **OK**.

To add a single row to the end of the data table

- Below the last row, click anywhere in a cell and begin typing.
- Below the last row, double-click in the empty row number area.

Add Columns to Data Tables

There are several ways to add new columns to a data table:

- Double-click the empty space to the right of the last data table column to add one column. See "About the Column Info Window" for more information about changing attributes after you add the column.
- Select **Cols > New Columns**. Use this option to add one or more columns and change their attributes.
- Double-click the upper triangle in the data grid above the row numbers. The Add Multiple Columns window appears.

Notes:

- When you initially create a column, you can choose to fill it with initial data values. See "Initialize Data in a Column". However, after you modify the cells, this option no longer appears.
- If any rows are selected in the data table, double-clicking to create a new column deselects the rows. Using Cols > New Columns to create a new column will not change the row selection.
- You cannot edit the data in the columns of a By group or a linked subset.

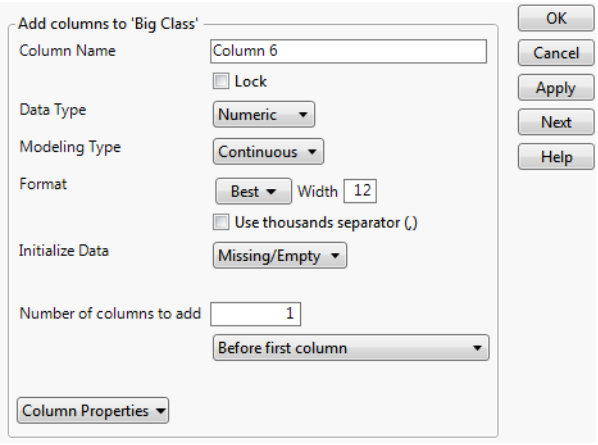
To add one or more columns using New Columns

1. (Optional) Select the column before or after which the new column or columns are added.
2. Select **Cols > New Columns**.
3. Change the column name.

By default, the new column names are Column 1, Column 2, and so on.

4. Select the data type, modeling type, and format for all of the columns. See "About the Column Info Window".
5. (Optional) Select initial data values for all of the columns. See "Initialize Data in a Column".
6. Specify where you want to put the new columns. If you do not select an option, the column is added after the last column.
7. (Optional) For multiple columns, specify whether the columns should be grouped. See "Group Columns in Data Tables".
8. To add more columns, follow these steps:
   a. Click **Apply** to add the new column.
   b. Click **Next**, change the column properties, and then click **Apply**.
   c. Click **Next** to add another column.
9. Click **OK** to save your changes.

Figure 4.2 The New Column Window

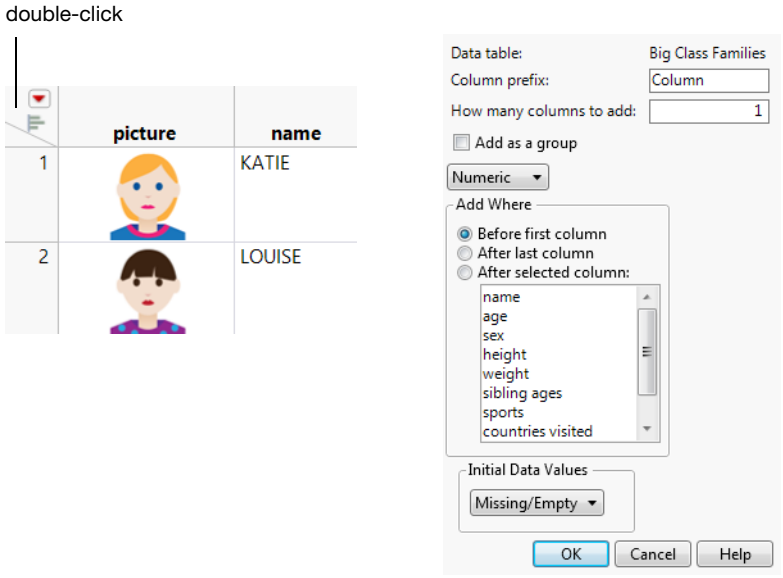


Tip: To change the modeling type after the columns are created, click the modeling type icon in the Columns panel and select a different type.

To add one or more columns using Add Multiple Columns

1. Double-click the upper triangular area in the data grid. The Add Multiple Columns window appears.

Figure 4.3 Adding Multiple Columns



2. Change the prefix for each column.

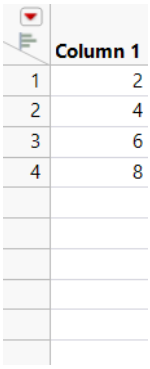
   By default, the new column names are Column 1, Column 2, and so on.
3. Enter the number of columns to add.
4. Specify whether the columns should be grouped. See "Group Columns in Data Tables".
5. Select the data type for all of the columns. See "About the Column Info Window".
6. Specify where you want to put the new columns.
7. (Optional) Select initial data values for all of the columns. See "Initialize Data in a Column".
8. Click **OK** to save your changes.

Fill Data Table Columns with Sequential Data

To fill columns with a repeating sequence of data or with a continuation of values, follow these steps:

1. Create a sequence of data in a column.

Figure 4.4 Example of a Sequence of Data



	Column 1
1	2
2	4
3	6
4	8

2. Highlight the cells containing the sequenced data. The cells can be in different columns.
3. Right-click the selected cells and select an option under **Fill**.

Fill Options

Repeat sequence to end of table Specifies that the cells below the selection are filled with repeats of the selected cells.

Continue sequence to end of table Specifies that the cells below the selection are filled with a continuation of the pattern found in the selected cells. For example, if the selected cells contain the numbers 1 and 2, then the remaining cells are filled with 3, 4, 5, 6, and so on. If the selected cells contain the numbers 2 and 4, then the remaining cells are filled with 6, 8, 10, 12, and so on.

Repeat sequence to JMP repeats the pattern found in the selected cells to the row number that you specify.

Continue sequence to JMP continues the pattern found in the selected cells to the row number that you specify.

Replace Missing Values with Previous Values in Data Table Columns

When a column contains missing values, you can replace those values with the value in the preceding cell. For example, if "4, .,., ., ., 6, ., ., 7, ." is selected, the values are changed to "4, 4, 4, 4, 4, 6, 6, 6, 7, 7".

1. Select the cells that contain the missing and replacement values.

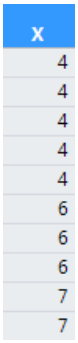
   The first cell that you select must immediately precede a missing value. After that, you can select non-adjacent cells.

Figure 4.5 Missing Values

x
4
•
•
•
•
6
•
•
7
•

2. Right-click and select **Fill > Replace Missing with Previous Value**.

Figure 4.6 Replaced Values



x
4
4
4
4
4
6
6
6
7
7

Enter Cell Formulas into Data Table Columns

In numeric data table columns, you can enter cell expressions preceded by an equal sign (=). JMP evaluates the expression and stores the new number as the cell's value. Unlike column formulas, a cell expression is not stored. Cell expressions can contain operators, constants, and global and column variables.

To enter an expression

1. Click the cell where you want to enter the expression.
2. Type an equal sign (=), and then type the expression (Table 4.1).
3. Press Enter.

Table 4.1 Examples of Expressions in Table Cells

Example expression	Cell value
=sqrt(2)	1.41
=456+890	1346
=height+weight	Sums the values of cells in columns height and weight located in the same row as the cell that you entered the expression.
=height[1]	Displays the value found in row 1 of the height column

Select Rows in Data Tables

To select one entire row

- Click in the empty space that contains the row number.

To select a specific row number

- Select **Rows > Row Selection > Go to Row** and type in the desired row number.

To select multiple rows

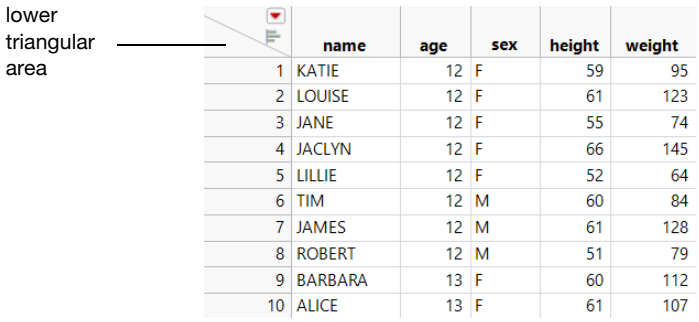
- For continuous selection:
  - Click and drag the cursor over the row numbers.
  - Press Shift and click the first and last rows of the desired range.
  - Press Shift and press the up or down arrow key.
- For discontiguous selection:
  - Press Ctrl and click each row.

To select or deselect all rows

- To select all rows, select **Rows > Row Selection > Select All Rows**.

  or
- Press Shift and click the lower triangular area in the upper left corner of the data grid to select. Click again in the same area to deselect all rows (Figure 4.7).
- To clear all highlighted areas in the data table, press Esc.

Figure 4.7 Lower Triangular Area



	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107

To select random rows

1. Select **Rows > Row Selection > Select Randomly**.
2. You can randomly select either a specific number of rows, or a proportion of the total number of rows:
   - Enter a whole number to select that number of rows.
   - Enter a value between 0 and 1 to select that proportion of rows.

   For example, enter 10 to select 10 rows. Enter 0.1 to select 10% of the rows.

To invert the row selection

- Select **Rows > Row Selection > Invert Row Selection**.

To select dominant rows

1. Select **Rows > Row Selection > Select Dominant**.
2. Choose the column(s) whose values you want to use to determine dominance.
3. Select the high or low values to dominate by for each column.
4. Click **OK**.

Note the following about dominant values and rows:

- A value is dominant over another value if it is higher or lower (based on your specification).
- The Select Dominant option selects each row that is not dominated by any other row. A row dominates another row only if *all* of its values are dominating the other row's values.
- The resultant set of rows is called the Pareto Frontier.

To save the current row selection in a new column

1. Select **Rows > Row Selection > Name Selection in Column.**

2. Type a column name.
3. Label the selected and deselected rows.
4. Click **OK**.

Tip: If you repeat this process after creating the new column and indicate the same column name, the original column is overwritten. Changes that you made in the data table to Selected or Unselected values are lost. Avoid overwriting those values by clearing the corresponding box in the Name Selection in Column window. For example, to avoid overwriting the Unselected values, make sure that the Unselected box is cleared.

To select excluded, hidden, or labeled rows

1. Select **Rows > Row Selection**.
2. Select from the following options:
   - Select Excluded
   - Select Hidden
   - Select Labeled

Note: For more information about excluded, hidden, or labeled rows, see "Assign Characteristics to Rows and Columns".

To select rows that contain duplicate values

1. Select **Rows > Row Selection > Select Duplicate Rows**.

   All duplicate rows except for the first are selected. If no columns are selected, duplicate rows are selected using all the columns.

   If columns are selected, duplicate values are found in the rows of those columns.

   The match is case sensitive.

Find Matching Cells in Rows

Find cells that have matching values using the Select Matching Cells option.

- Select **Rows > Row Selection > Select Matching Cells**

  or

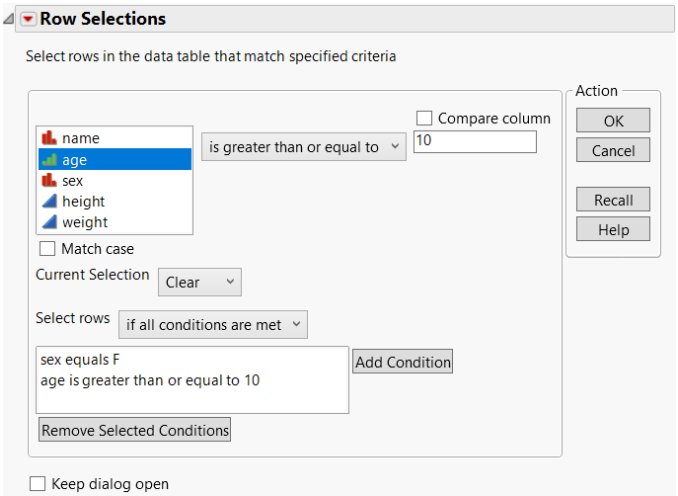
- Right-click one of the highlighted row numbers and select **Select Matching Cells**.

Select Rows That Contain Specific Values

JMP can search for a specific value (or text string) and highlight all of the cells in the data table that contain the specific value.

1. Select **Rows > Row Selection > Select Where**.

Figure 4.8 Specify Criteria for Selecting Rows



2. From the column list, highlight the name of the column whose rows you want to select.
3. Use the drop-down menu to select a condition from the list (equals, does not equal, and so on). See Figure 4.8.
4. Type the search value.
5. Click **OK**.

You can also specify the following optional features:

- To compare the values of two columns, click the **Compare column** check box. Select from the list of columns for comparison.
- To make the search case-sensitive, click the box beside **Match Case**.
- You can save a Where clause from the Select Rows window to the clipboard, data table, or script window. Select **Keep dialog open**, select rows and other options, and select one of the Save Script options from the Row Selections red triangle menu. Add JSL to the script that reruns the query:

```
dt = Open("$SAMPLE_DATA/Big Class.jmp");
Current Data Table() << Row Selection(
   Select where( :age == 15 ),
   Dialog( Edit( Source Column( :age ) == "" ), Keep dialog open )
```

```
);
```

If rows are currently selected in the data table, you can specify the following features:

- Click an option under **Current Selection** to tell JMP how to handle that current selection:
  - **Clear** removes the highlight from currently selected rows and selects all rows that contain the specified value.
  - **Extend** keeps the currently selected rows selected and also selects the rows in which the specified value has been found.
  - **Restrict** selects the rows in the currently selected array that contain the specified values.
- Click **Add Condition** to add a condition to the list.
- To add more conditions to the search, repeat the previous steps. Click the appropriate item in the **Select Rows** area to specify if you would like JMP to select rows conditionally: if *all* conditions are met, or if *any* of the conditions are met.

Locate the Next Selected Row

You can locate the next selected row after the current row and cause it to flash by selecting **Rows > Next Selected**.

Each time you select **Rows > Next Selected**, the next selected row is found and flashes. A beep signals when the last selected row is located.

You might want to use this feature when you have selected rows intermittently in a large data set and want to look through the selected rows in the data table.

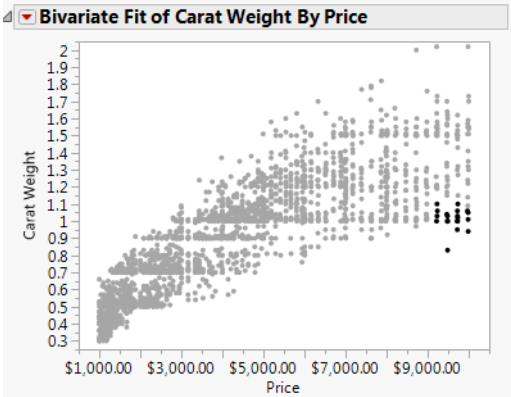
Example of Locating Next Selected Rows

1. Select **Help > Sample Data Library** and open Diamonds Data.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Carat Weight and click **Y, Response**.
4. Select Price and click **X, Factor**.
5. Click **OK**.
6. Select **Tools > Lasso**.

   If you cannot see the menu bar, hover over the blue bar below the title bar to reveal it.
7. Lasso some of the points near the 10,000 dollar price at the bottom of the plot (Figure 4.9).
8. In the data table, select **Rows > Next Selected** (or you can press F7).

You can easily navigate through the selected rows to see the data for each.

Figure 4.9 Points Selected



Select Columns in Data Tables

There are several ways to select columns:

- Select columns in the data table itself. See "How to Select Columns".
- In a data table that has many columns, select columns by attributes, properties, and statistics in the Columns Viewer. See "Select Columns in the JMP Columns Viewer".
- Find the next or previously selected column. See "Locate Next and Previously Selected Columns in Data Tables".

How to Select Columns

Choose one of several methods for selecting columns in a data table.

To select one entire column

- In the data grid, click in the empty space around the column name.

  or

- In the Columns panel, click the column name.

To select a specific column number

1. Select **Cols > Column Selection > Go to**.
2. Enter the column number or name and click **OK**.

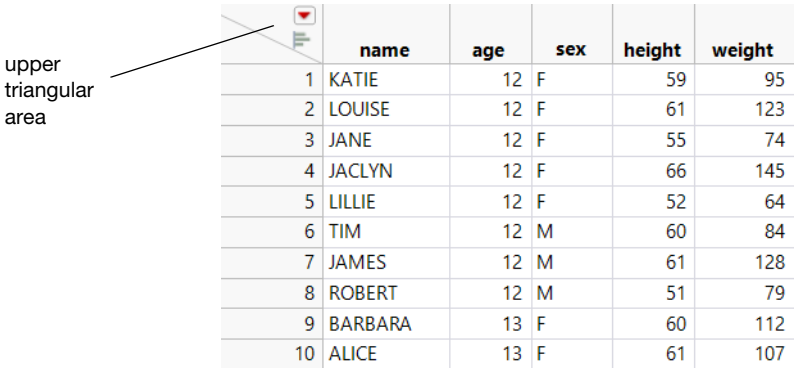
To select multiple columns

- For continuous selection:
  - Click and drag the cursor over the column name.
  - Press Shift and click the first and last columns of the desired range.
  - Press Shift and press the left or right arrow key.
- For columns that are not next to each other:
  - Press Ctrl and click each column.

To select or deselect all columns

- Press Shift and click the upper triangular area in the upper left corner of the data grid to select. Click again in the same area to deselect all columns.

Figure 4.10 Upper Triangular Area



	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107

Tip: To clear all highlighted areas in the data table, press Esc.

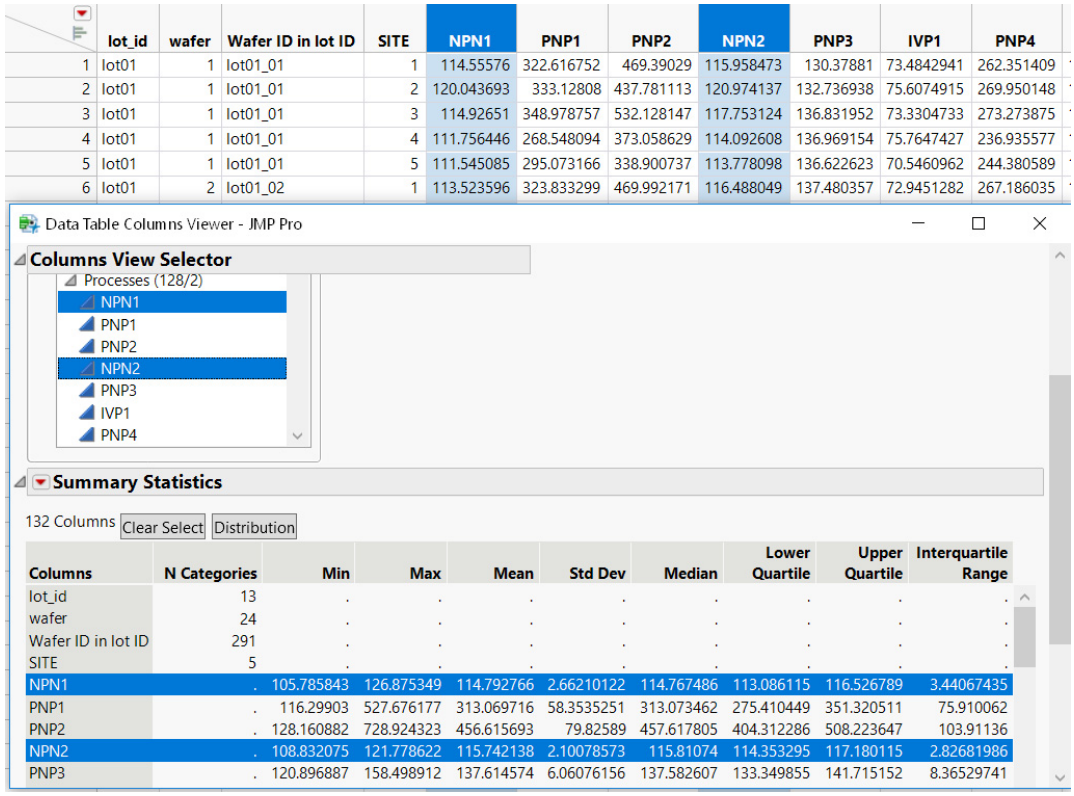
To invert the column selection

Select **Cols > Column Selection > Invert Column Selection**. Selects any column that is currently deselected and deselects any column that is currently selected.

Select Columns in the JMP Columns Viewer

The Columns Viewer option in the Cols menu helps you quickly select columns by attributes, properties, and statistics, particularly in a data table that has many columns. You can view summary statistics and properties for those columns, view quartiles in the summary statistics, subset the data, and more. And columns in the Columns Viewer window are also linked to the data table columns.

Figure 4.11 Linked Columns in the Column Viewer

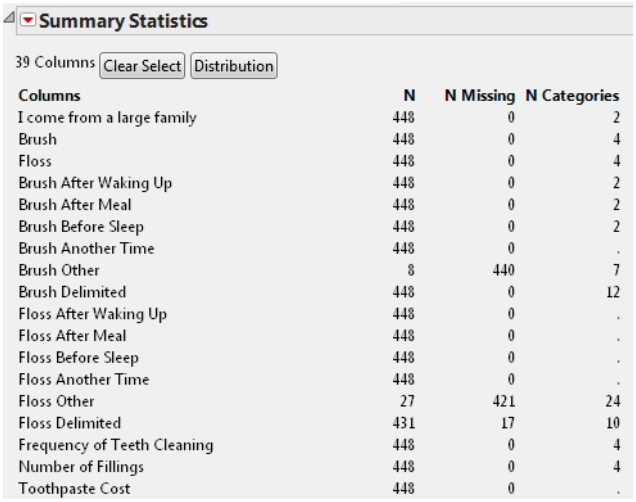


	lot_id	wafer	Wafer ID in lot ID	SITE	NPN1	PNP1	PNP2	NPN2	PNP3	IVP1	PNP4
1	lot01	1	lot01_01	1	114.55576	322.616752	469.39029	115.958473	130.37881	73.4842941	262.351409
2	lot01	1	lot01_01	2	120.043693	333.12808	437.781113	120.974137	132.736938	75.6074915	269.950148
3	lot01	1	lot01_01	3	114.92651	348.978757	532.128147	117.753124	136.831952	73.3304733	273.273875
4	lot01	1	lot01_01	4	111.756446	268.548094	373.058629	114.092608	136.969154	75.7647427	236.935577
5	lot01	1	lot01_01	5	111.545085	295.073166	338.900737	113.778098	136.622623	70.5460962	244.380589
6	lot01	2	lot01_02	1	113.523596	323.833299	469.992171	116.488049	137.480357	72.9451282	267.186035

Columns	N Categories	Min	Max	Mean	Std Dev	Median	Lower Quartile	Upper Quartile	Interquartile Range
lot_id	13	.	.	.	.	.	.	.	.
wafer	24	.	.	.	.	.	.	.	.
Wafer ID in lot ID	291	.	.	.	.	.	.	.	.
SITE	5	.	.	.	.	.	.	.	.
NPN1	.	105.785843	126.875349	114.792766	2.66210122	114.767486	113.086115	116.526789	3.44067435
PNP1	.	116.29903	527.676177	313.069716	58.3535251	313.073462	275.410449	351.320511	75.910062
PNP2	.	128.160882	728.924323	456.615693	79.82589	457.617805	404.312286	508.223647	103.91136
NPN2	.	108.832075	121.778622	115.742138	2.10078573	115.81074	114.353295	117.180115	2.82681986
PNP3	.	120.896887	158.498912	137.614574	6.06076156	137.582607	133.349855	141.715152	8.36529741

The Columns Viewer gives you a quick view of data table characteristics. For example, the Summary Statistics report shows which columns contain missing values. You can select those columns in the report and then exclude them in the data table.

Figure 4.12 Identify Missing Values



Summary Statistics

39 Columns | Clear Select | Distribution

Columns	N	N Missing	N Categories
I come from a large family	448	0	2
Brush	448	0	4
Floss	448	0	4
Brush After Waking Up	448	0	2
Brush After Meal	448	0	2
Brush Before Sleep	448	0	2
Brush Another Time	448	0	.
Brush Other	8	440	7
Brush Delimited	448	0	12
Floss After Waking Up	448	0	.
Floss After Meal	448	0	.
Floss Before Sleep	448	0	.
Floss Another Time	448	0	.
Floss Other	27	421	24
Floss Delimited	431	17	10
Frequency of Teeth Cleaning	448	0	4
Number of Fillings	448	0	4
Toothpaste Cost	448	0	.

The Summary Statistics report shows the following information:

- the total number of rows (N)
- the number of rows with missing values (N Missing)
- the number of categories (N Categories)
- for continuous data, the Min, Max, Mean, and Std Dev

Other options include the following:

Clear Select Deselects columns in the data table and in the Columns Viewer. This option ensures that no columns are selected before you begin selecting columns.

Subset Creates a linked subset data table from the selected columns.

Show Summary Creates a linked Summary Statistics report for the selected columns. Right-click to select options such as sorting by column or creating a data table. Select **Show Quartiles** to include lower quartiles, upper quartiles, and interquartile ranges. To create a linked data table from all columns in the report, click the Summary Statistics red triangle and select Data Table View.

Find Columns with Properties Shows a list of column properties in the Columns with Properties report. Select the properties that you want to find and then click **OK** to create a linked report from all columns. Or you can select columns first in the Select Columns list and then show the list of properties just for those columns.

Notes:

- Each time you click the **Show Summary** or **Find Columns with Properties** buttons, a new report is added to the window. To delete a report, select **Remove** from the report's red triangle menu.
- Deleting rows from a data table view created from Columns Viewer is not undo-able.

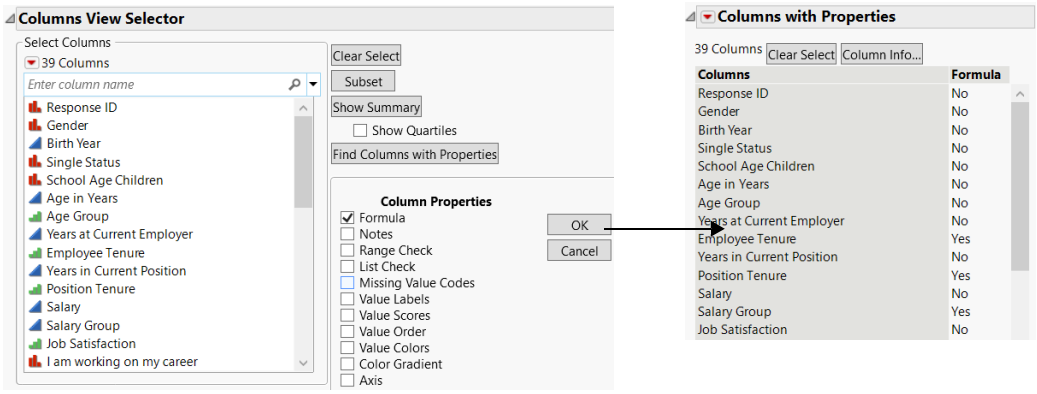
Example of Finding Columns with a Specific Property

This example shows how to find columns that have a Formula property and then view all formulas at once.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Select **Cols > Columns Viewer** to open the Data Table Columns Viewer window.
3. Select **Find Columns with Properties**, select **Formula**, and click **OK**.

   The Columns with Properties report appears. Several columns include the Formula property. Because the list is so long, you want to view all formula columns together.

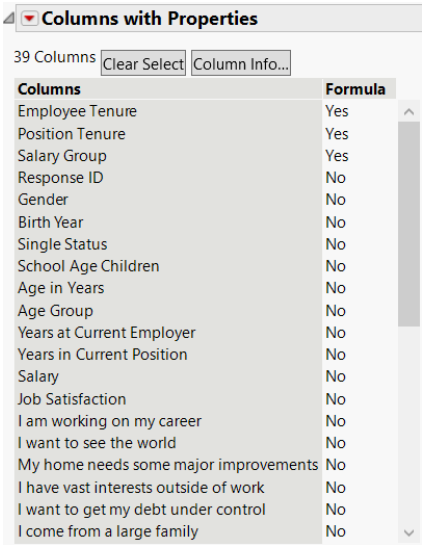
Figure 4.13 Select the Formula Column Property



4. Right-click the report, select **Sort by Column**, **Formula**, and then click **OK**.

   Columns that have a Formula property appear at the top of the report.

Figure 4.14 Sort by Column



5. Select the Employee Tenure, Position Tenure, and Salary Group columns and select **Column Info**.

   Formulas for the selected columns appear in the data table's Column Settings window.

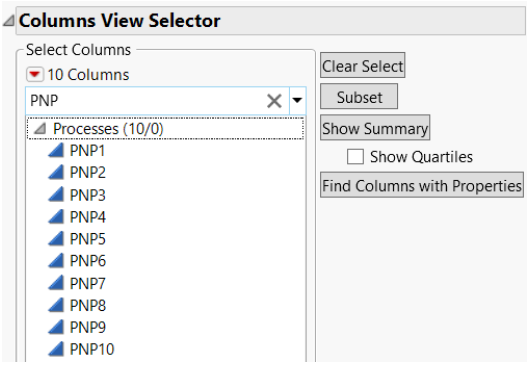
Example of Showing Summary Statistics

This example shows how to find columns with a low standard deviation. This can be helpful if you want to delete or exclude that data from an analysis.

1. Select **Help > Sample Data Library** and open Semiconductor Capability.jmp.
2. Select **Cols > Columns Viewer** to open the Data Table Columns Viewer window.
3. Type PNP in the box and press Enter.

   The grouped processes are shown.
4. Expand the processes group to show the PNP columns.

Figure 4.15 Filter Columns by Name



5. Click **Show Summary** to add the Summary Statistics report.

   The rows show the minimum, maximum, mean, and standard deviation for each column.

Figure 4.16 Summary Statistics for Selected Columns

Summary Statistics

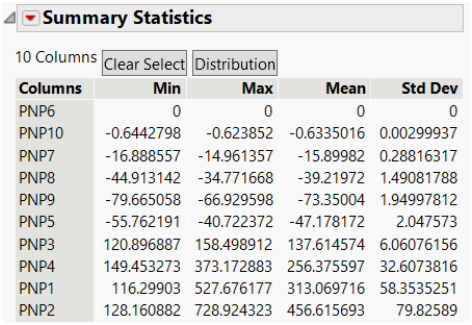
10 Columns Clear Select Distribution

Columns	Min	Max	Mean	Std Dev
PNP1	116.29903	527.676177	313.069716	58.3535251
PNP2	128.160882	728.924323	456.615693	79.82589
PNP3	120.896887	158.498912	137.614574	6.06076156
PNP4	149.453273	373.172883	256.375597	32.6073816
PNP5	-55.762191	-40.722372	-47.178172	2.047573
PNP6	0	0	0	0
PNP7	-16.888557	-14.961357	-15.89982	0.28816317
PNP8	-44.913142	-34.771668	-39.21972	1.49081788
PNP9	-79.665058	-66.929598	-73.35004	1.94997812
PNP10	-0.6442798	-0.623852	-0.6335016	0.00299937

6. Right-click in the report and select **Sort by Column**.
7. Select **Std Dev** and **Ascending**, and then click **OK**.

   Notice that PNP6 has no standard deviation, because the minimum, maximum, and mean values are 0.

Figure 4.17 Sorted Std Dev Column



Summary Statistics

10 Columns | Clear Select | Distribution

Columns	Min	Max	Mean	Std Dev
PNP6	0	0	0	0
PNP10	-0.6442798	-0.623852	-0.6335016	0.00299937
PNP7	-16.888557	-14.961357	-15.89982	0.28816317
PNP8	-44.913142	-34.771668	-39.21972	1.49081788
PNP9	-79.665058	-66.929598	-73.35004	1.94997812
PNP5	-55.762191	-40.722372	-47.178172	2.047573
PNP3	120.896887	158.498912	137.614574	6.06076156
PNP4	149.453273	373.172883	256.375597	32.6073816
PNP1	116.29903	527.676177	313.069716	58.3535251
PNP2	128.160882	728.924323	456.615693	79.82589

8. In the Summary Statistics report, select the row for PNP6 and then display the data table.
9. View the data table and press Delete to remove the selected column.

   The column is instantly removed from the data table.
10. To close the Columns Viewer, click the X button in the upper right corner (Windows) or upper left corner (macOS) of the window.

Locate Next and Previously Selected Columns in Data Tables

You can find the next selected column after the current column by selecting **Cols > Column Selection > Next Selected Column**. Similarly, you can find the previously selected column before the current column by selecting **Cols > Column Selection > Previous Selected Column**.

Each time you select one of these options, the next or previously selected column appears and flashes. The options are available only when columns are selected.

You might want to use this feature to look at intermittently selected columns in a large data table.

Resize Rows and Columns

To resize rows

Resize rows by dragging one of the row borders. All rows are resized to the same height. Graphics that appear inside each cell shrink based on the row height.

To resize columns

To resize a column, drag the column border to the right or left.

To resize several columns to the same width, follow these steps:

1. Select the columns that you want to resize.
2. Press Alt (Windows) or Option (macOS) and drag one of the column borders. All columns are resized to the same width.

Organize Data in Data Tables

Choose one of many methods to rearrange data in a data table.

- "Delete Rows in Data Tables"
- "Delete Columns in Data Tables"
- "Rearrange Columns in Data Tables"
- "Group Columns in Data Tables"
- "Move Values in Data Tables"
- "Color Cells in Data Tables"
- "Edit or Delete Cells in Data Tables"
- "Edit Column Names in Data Tables"
- "Hide and Exclude Rows in Data Tables"
- "Exclude Rows in Data Tables"
- "Hide Rows in Data Tables"
- "Exclude Columns in Data Tables"
- "Hide Columns in Data Tables"
- "View Patterns of Missing Data in Data Tables"
- "Find and Replace Cell Values in Data Tables"
- "Edit Cells in Data Tables"
- "Right-Click Menus for Rows and Columns"
- "Compare Data Tables"

Delete Rows in Data Tables

To delete rows in a JMP data table, follow these steps:

1. Highlight the rows that you want to delete.
2. Press Delete, or right-click the row numbers and select **Delete Rows**.

Caution: When you try to delete thousands of rows, an alert might appear if your computer has insufficient memory to save data for undo. Either select fewer rows to delete or select **Disable Undo** from the Table panel red triangle menu. This option removes all actions from the undo history and does not record future actions. When the **Disable Undo** option is selected, it is in effect only while the data table is open; the setting is not saved with the data table.

Delete Columns in Data Tables

To delete columns in a JMP data table, follow these steps:

1. Highlight the columns to delete.
2. Press Delete, or right-click and select **Delete Columns**.

Rearrange Columns in Data Tables

You can rearrange or sort data table columns by their name, data type, or modeling type, or reverse the current order. To reorder columns, select **Cols > Reorder Columns** and select from one of the following options:

Move Selected Columns Moves the selected columns to a particular place in the data table. Specify where to place the selected columns in the Move Selected Columns window:

To first Moves the selected columns so that they are in the left-most position in the data table.

To last Moves the selected columns so that they are in the right-most position in the data table.

After Moves the selected columns so that they are after a column that you identify.

Original Order Returns the columns to the order they were in when data table was last saved.

Reorder by Name Arranges the columns from left to right in alphabetical order by column name.

Reorder By Data Type Arranges the columns from left to right in alphabetic order by data type (expression, row state, character, numeric).

Reorder By Modeling Type Arranges the columns from left to right in alphabetic order by modeling type (none, continuous, ordinal, nominal, multiple response, unstructured text, vector). Row state columns have no modeling type, and are shown last.

Reverse Order Reverses the order of the data table columns.

Group Columns in Data Tables

Group columns within a single heading to manage large numbers of columns and facilitate analysis role assignment. Grouped columns appear in an outline format within the Columns panel.

To group or ungroup columns

1. Within the data grid, select the columns that you want to group.
2. From the main menu, select **Cols > Group Columns** or **Cols > Ungroup Columns**.

   or

1. From the Columns panel, select the columns that you want to group.
2. Right-click the selected columns and select **Group Columns** or **Ungroup Columns**.

Note: Grouped columns are automatically retained for data tables generated from the following commands: **Subset**, **Sort**, **Summary**, **Join**, **Stack**, and **Split**. For the **Stack** command, if all the columns in the stack group belong to the same columns group, then the group's name is used for the column name.

Move Values in Data Tables

To move values in a data table, select the values, click and pause, and then drag and drop the values into the new location.

Tip: Clicking and dragging on a selection without pausing extends the selection.

When dragging and dropping values, note the following:

- Cells retain all of their characteristics and column properties.
- After you move cells, missing values appear in the cells that you initially selected.
- The selected cells and the destination cells must have the same data type.
- If you drag a set of cells to an empty area of the table, new columns are automatically created.
- New columns have the original columns' display format and modeling types.

To specify where to move rows

1. Highlight the rows that you want to move.
2. Select **Rows > Move Rows**.
3. Specify where you would like to move the rows in the Move Rows window:

- To the beginning of the table (**At start**)
- To the end of the table (**At end**)
- After a specific row number (**After row:**)

Move Content into Another JMP Window

On Windows, you can drag selected content over a minimized window. The minimized window moves to the front and you can paste your content into it. You can do the same thing in JMP. For example, you can drag selected content over the Home Window button (located in the bottom right corner of most windows). Then in the Window List, drag the content over the window that you want to move the content into. That window moves to the front and you can drop in the content.

Tip: If you cannot see the JMP Home Window button, select **View > Status Bars.**

For example, you can drag a selected column, row, or cell from one data table into another; drag selected text from one script window into another; or drag selected content from a report into a journal.

Color Cells in Data Tables

You can select one or more data table cells and change the color to highlight a specific value. Right-click the cell or cells, select **Color Cells**, and select a color. To remove color from cells, select the cells, right-click, and select **Cols > Clear Color**.

Edit or Delete Cells in Data Tables

To edit or delete the contents of a cell, follow these steps:

1. Click the cell containing the value that you want to edit or delete.
2. Press Delete.
3. To edit the value, click the cell a second time, and then edit the cell's value.

Edit Column Names in Data Tables

To edit a single column name, select the column and begin typing. You can also edit the column name in the Column Info window, or select the header and press Enter.

To edit all of the column names at once, click **Cols > Column Names** and select one of the following options:

Move up Moves the first row of data up to replace the column names.

Move up and append Moves the first row of data up and appends the entries to the existing column names.

Move down Moves the columns names down to become a row of data. The columns are then named Column 1, Column 2, and so on.

Recode Column Names Opens the Recode window to recode the column names. See "Recode Data in a Column".

Hide and Exclude Rows in Data Tables

Apply Hidden and Excluded row states to rows that you do not want to include in analysis or plot calculations and that you do not want to display in plots. These rows are not included in subsequent analyses.

Notes:

- Plots found under the Graph menu, with the exception of Profilers, are immediately recalculated to reflect hidden and excluded rows.
- For many platforms, reports and plots are not updated immediately to reflect excluded and hidden observations. You need to rerun the analysis to recalculate analysis results and the related plots. For exceptions, see "Platforms that Update Immediately for Hide and Exclude".
- When you Exclude and Hide rows, they are excluded and hidden in all open reports and plots that update automatically.
- When you Exclude and Hide rows, a circle with a strikethrough (, for Exclude) and a mask icon (, for Hide) appear beside the row number.
- Rows remain hidden and excluded until you highlight the rows and select **Hide and Exclude** again

To hide and exclude one or more rows

1. Select the rows that you want to hide and exclude.
2. Do one of the following:
   - Right-click the highlighted area next to the row numbers and select **Hide and Exclude.**
   - From the **Rows** menu, select **Hide and Exclude**.

Platforms that Update Immediately for Hide and Exclude

When you apply Hide and Exclude to rows, for most platforms under the Analyze menu, points or plot elements that correspond directly to rows are usually hidden immediately in plots. However, calculations shown in accompanying reports usually are *not* updated. For the following platforms, applying Hide and Exclude results in immediate updates to plots *and* calculations:

- Tabulate
- Control Chart Builder

Exclude Rows in Data Tables

Apply the Excluded row state to rows that you do not want to include in subsequent calculations. Statistics are recalculated without the excluded rows.

To exclude one or more rows, follow these steps:

1. Select rows that you want to exclude.
2. Do one of the following:
   - Right-click the highlighted area next to the row numbers and select **Exclude/Unexclude.**
   - From the **Rows** menu, select **Exclude/Unexclude**.

Notes:

- In most cases, when a row is represented by a point or an individual plot element, excluding the row does *not* hide it in plots.
- In Graph Builder, when a plot element is represented by a group of rows, the plot is reconstructed using only the unexcluded rows. The plot also does not display plot elements corresponding to the excluded rows. Excluded rows are treated as if they are both hidden *and* excluded.
- Plots found under the Graph menu (with the exception of Profilers, Parallel Plot, and Scatterplot Matrix) are immediately recalculated to reflect excluded rows.
- For many platforms, reports and plots are not updated immediately to reflect excluded observations. You need to rerun the analysis to recalculate plots and analysis results.
- When you Exclude rows, they are excluded in all open plots and reports that are updated immediately.
- A circle with a strikethrough ( ) appears beside the row number of an excluded row.
- Rows remain excluded until you highlight the rows and select **Exclude/Unexclude** again.

Hide Rows in Data Tables

Apply the Hidden row state to rows that you do not want to include in plots.

Note: Not all plots hide the elements that correspond to hidden rows. This is often the case when the calculations used to construct the plot elements use the data from the hidden rows. As a general rule, if there is a one-to-one correspondence between rows and plot elements, the points or elements corresponding to the hidden rows are hidden.

To hide one or more rows, follow these steps:

1. Select rows that you want to hide.
2. Do one of the following:
   - Right-click the selected area next to the row numbers and select **Hide/Unhide.**
   - From the **Rows** menu, select **Hide/Unhide**.

Notes:

- When you Hide rows, they are hidden in all open plots that are updated immediately.
- Hiding rows does *not* exclude them from calculations and analyses.
- When each hidden row corresponds to a point or an individual plot element, plots are updated immediately when you apply the Hidden row state to rows.
- Depending on the platform, plot elements are calculated using multiple observations, but they do not correspond directly to a row. In these cases, the plot elements generally are not updated to reflect hidden observations unless those rows are also excluded.
- For some plots where plot elements correspond to groups of rows, when you Hide rows, the plot is updated to hide plot elements corresponding to hidden rows. See "JMP Plots Where Grouping Elements are Updated for Hide".
- A mask icon ( ) appears beside the row number of a hidden row.
- Rows remain hidden until you select the rows and select **Hide/Unhide** again.

JMP Plots Where Grouping Elements are Updated for Hide

When you apply Hide to rows, in plots with a one-to-one correspondence between rows and plot elements, the points or plot elements are usually hidden immediately. However, for the following platforms and options, applying Hide results in immediate updates to plot elements that correspond to groups of rows:

- Control Chart Builder. See *Quality and Process Methods*.
- Graph Builder. The bar chart, heat map, and pie chart are updated to reflect hidden rows. See *Essential Graphing*.
- Bubble Plot. See *Essential Graphing*.

Exclude Columns in Data Tables

When your data table contains columns that you do not want to consider for analysis, you can exclude those columns so that they do not appear in selection lists in launch windows. Note the following:

- Excluded columns remain visible in the data grid.
- A circle with a strikethrough ( ) appears to the right of the column name in the Columns panel.

To exclude columns from launch windows

1. Select one or more columns that you want to exclude.
2. Do one of the following:
   - Right-click the highlighted column name in the Columns panel and select **Exclude/Unexclude.**
   - From the **Cols** menu, select **Exclude/Unexclude**.

To unexclude columns

1. Select the columns in the Columns panel.
2. Do one of the following:
   - Right-click the highlighted column name in the Columns panel and select **Exclude/Unexclude.**
   - From the **Cols** menu, select **Exclude/Unexclude**.

Hide Columns in Data Tables

When your data table contains columns that you do not want to see in the data grid, you can hide those columns. Note the following:

- Hidden columns appear in launch windows and are available for analyses.
- A mask icon ( ) appears to the right of the column name in the Columns panel.

To hide columns

1. Select one or more columns that you want to hide.
2. Do one of the following:
   - Right-click the highlighted column name in the Columns panel and select **Hide/Unhide.**
   - From the **Cols** menu, select **Hide/Unhide**.

To unhide columns

1. Select the columns in the Columns panel.
2. Do one of the following:
   - Right-click the highlighted column name in the Columns panel and select **Hide/Unhide.**
   - From the **Cols** menu, select **Hide/Unhide**.

View Patterns of Missing Data in Data Tables

If your data table contains missing data, you might want to determine whether there is a pattern to the missing data. The pattern might help you make discoveries about your data.

To view patterns of missing data, follow these steps:

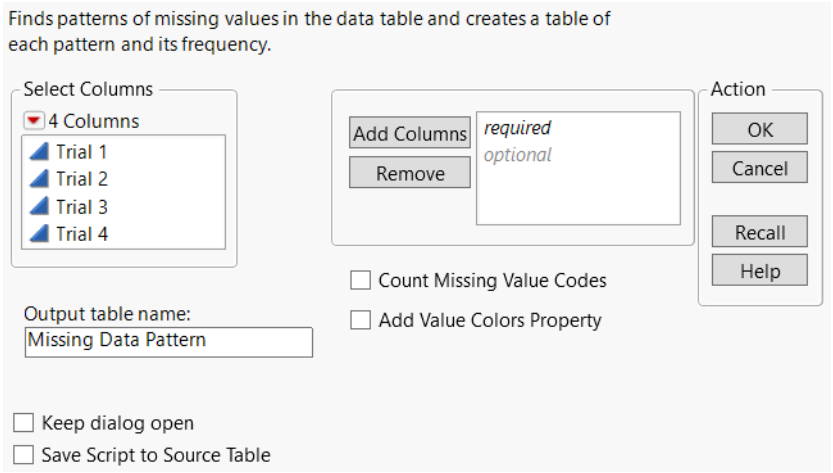
1. With your data table open, select **Tables > Missing Data Pattern**.
2. Select the columns for which you would like to find patterns of missing data.
3. Click **Add Columns**.
4. Select **Count Missing Value Codes** to count missing value codes as missing values.
5. Select **Add Value Colors Property** to consistently color columns that contain 0 or 1 (missing value indicators).

   For example, nonmissing is always colored blue, missing is always colored red. This is especially helpful for data tables that might have similar values across columns.
6. Click **OK**.

Example of Viewing Patterns of Missing Data

1. Select **Help > Sample Data Library** and open Missing Data Pattern.jmp.
2. Select **Tables > Missing Data Pattern**.

Figure 4.18 The Missing Data Pattern Window

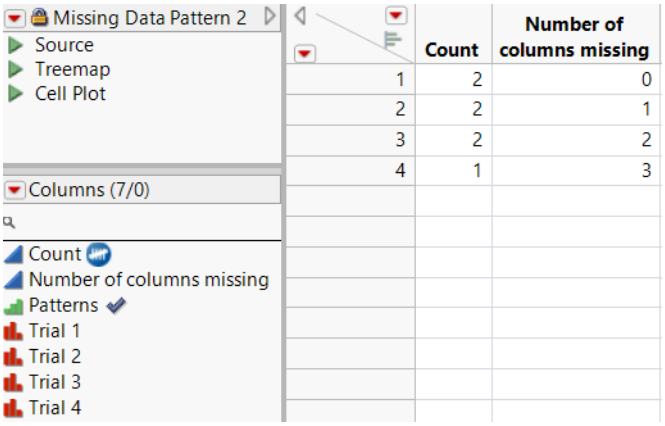


3. Highlight all of the columns.

   Note: For more information about the options in the red triangle menu, see "Column Filter Menu".

4. Click **Add Columns**.
5. Click **OK**.

Figure 4.19 A Missing Data Pattern Table



	Count	Number of columns missing
1	2	0
2	2	1
3	2	2
4	1	3

Tip: To quickly create a Treemap or Cell Plot of the data, click the green triangle next to the Treemap or Cell Plot script in the table panel.

Figure 4.19 shows the following patterns:

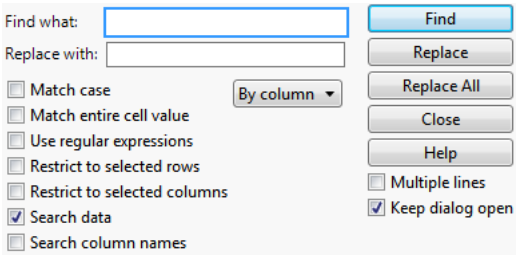
- Row 1 shows that there are two instances where all rows in Trial 1, Trial 2, Trial 3, and Trial 4 have no missing values.
- Row 2 shows that there are two rows in the source table whose one missing value is in the Trial 4 column.
- Row 3 shows that there are two rows in the source table whose missing values are in the Trial 3 and Trial 4 columns.
- Row 4 shows that there is one row in the source table whose three missing values are in the Trial 2, Trial 3, and Trial 4 columns.

The Count column is assigned the frequency role. If you now use the Missing Data Pattern data table to run an analysis, JMP automatically uses Count as a frequency. See "Assign a Preselected Analysis Role to a Column".

Find and Replace Cell Values in Data Tables

You can find and replace cell values by selecting the **Edit > Search > Find** options.

Figure 4.20 The Find Window



The following rules apply to searching for values:

- To find values in hidden columns, unhide the column.
- Values found in locked columns cannot be modified.
- The **Undo** command works only with **Replace**. You cannot undo **Replace All**.
- If your data table contains value labels, using the **Search** commands searches for actual values, but does not search for labels. See "Value Labels".
- If your data table contains formatted values (such as dates, times, or durations) using the Search command searches for the formatted values, not the actual values.

Find Window Options

Refine your search with the following options:

Match Case Performs a case-sensitive search, which can be useful for locating proper nouns or other capitalized words.

Match entire cell value Detects empty spaces, which lets you search for a series of words in a character column, or locate strings with unwanted leading or trailing empty spaces.

> **Tip:** To find missing character values, leave the **Find what** box empty and check **Match entire cell value**. To find missing numeric values, insert a period into the **Find** box and check **Match entire cell value**.

Use regular expressions Assumes the find string to be a regular expression instead of the literal string that you enter in the **Find what** box. The regular expressions follow standard semantics.

Restrict to selected rows Restricts the search to selected rows.

Restrict to selected columns Restricts the search to selected columns.

Search data Searches only data cells (omitting column names).

Search column names Searches only column names (omitting data cells).

By column Searches the table column by column, from top to bottom, until it reaches the last cell in the rightmost column, or until you stop the search.

By row Searches the data table row by row from left to right, to the rightmost cell in the last row or until you stop the search.

Multiple lines Increases the Find and Replace boxes to 3 lines long instead of 1. The Enter key inserts a return into the field.

> **Tip:** You can alternatively click and drag on the Find and Replace boxes to make them larger. If you copy and paste, the boxes resize to 1 line long, but all of your text is still there.

Keep dialog open Keeps the Find window open during your search.

Search Actions

This section describes some common searches that you might perform.

Begin by searching for a value in the data table. The search begins with the first cell in the first column and searches every cell until it locates the value or reaches the end of the table.

To replace the currently highlighted cell value

Enter a value in the Replace with box and click **Replace**. Or, if the Search window is closed, select **Edit > Search > Replace**. If the replace value is a missing value, the currently highlighted cell content becomes a missing value.

To replace all occurrences of the specified value

Enter a value in the Replace with box and click **Replace All**. Or, if the Search window is closed, select **Edit > Search > Replace All**.

To replace the value and search for the next value

Enter a value in the Replace with box and click **Replace**. Or, if the Search window is closed, select **Edit > Search > Replace and Find Next**. Or, press Ctrl+L.

To use a selected value as the Find what *value*

In the data table, select a value. Select **Edit > Search > Use Selection for Find**. Next, select **Edit > Search > Find**. The value that you selected in the data table is already entered in the **Find what** field.

To use a selected value as the Replace with *value*

In the data table, select a value. Select **Edit > Search > Use Selection for Replace** to populate the **Replace with** field.

To find the next value when the Search window is closed

Select **Edit > Search > Find Next**. Or, press Ctrl+G, or F3, on Windows.

To find a missing value:

- To find missing character values, leave the Find what field empty and select **Match entire cell value**.
- To find missing numeric values, type a period into the Find what text box.

Edit Cells in Data Tables

Use the Row Editor to browse or edit cells one row at a time. Open the Row Editor in one of the following ways:

- Select **Rows > Row Editor**.
- In a data table, double-click in the row number area. The row that you use is the row that first appears in the Row Editor.
- In a report window, right-click in a plot or graph and select **Row Editor**.

Figure 4.21 Row Editor

Note the following:

- If you have a report window open, and you want edited data to be automatically reflected there, make sure that Automatic Recalc is turned on. See "Automatic Recalc".
- If your data table contains value labels, the Row Editor displays the label, and when the cell is highlighted for editing, it shows the actual value. See "Value Labels".

Row Editor Buttons

Click the arrow buttons to browse through selected rows or the entire data set if no rows are selected.

- Shows the previous row.
- Shows the previously selected row.
- Makes the row blink in graphs.
- Shows the next selected row.
- Shows the next row.
- Searches for a row. See "Resize Rows and Columns".
- Creates a new row at the end of the data table.

Note: Changes made to a row using the Row Editor are written to the data table when you change fields in the Row Editor. You still need to save the changes to the data table.

Row Editor Options

The red triangle menu in the Row Editor contains the following options:

Next Selected Displays information for the selected row that is located after the current one.

Prev Selected Displays information for the selected row that is located before the current one.

Next Displays information for the row that is located after the current one, regardless of whether the row is selected.

Prev Displays information for the row that is located before the current one, regardless of whether the row is selected.

Save Saves the data table and any changes that you have made to it via the Row Editor.

New Row Creates a new row in the data table.

Find Displays the same window as if you had selected **Rows > Row Selection > Select Where**. Select one of the options from the **Current Selection** menu, and then highlight the column whose rows you want to select. Type in the value for which you want JMP to search. See "Resize Rows and Columns".

Blink Causes the current row's highlight to flash at a rapid rate.

Note: Text in a locked column or a locked data table cannot be edited. See "Lock" and "Lock Data Tables".

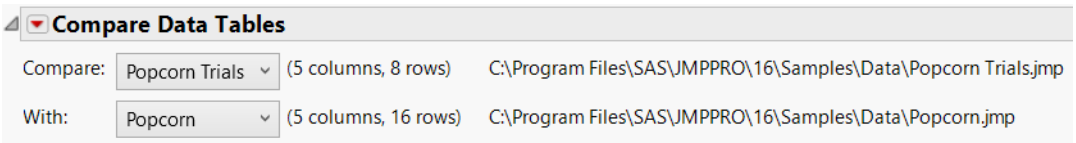
Right-Click Menus for Rows and Columns

When you right-click in the row number area, or at the top of a column in the column name area, pop-up menus appear. These menus provide quick access to selected commands in the Rows and Columns menus. For more information about these options, see "Right-Click Options for Columns" and "Right-Click Options for Rows".

Compare Data Tables

JMP can compare two open data tables and report the differences between data, scripts, table variables, column names, column properties, and column attributes. The numbers of columns and rows in each data table are shown at the top of the Compare Data Tables window. In this example, the sample data tables Popcorn Trials.jmp and Popcorn.jmp are compared.

Figure 4.22 Basic Information about Data Tables



Specify Matching Columns

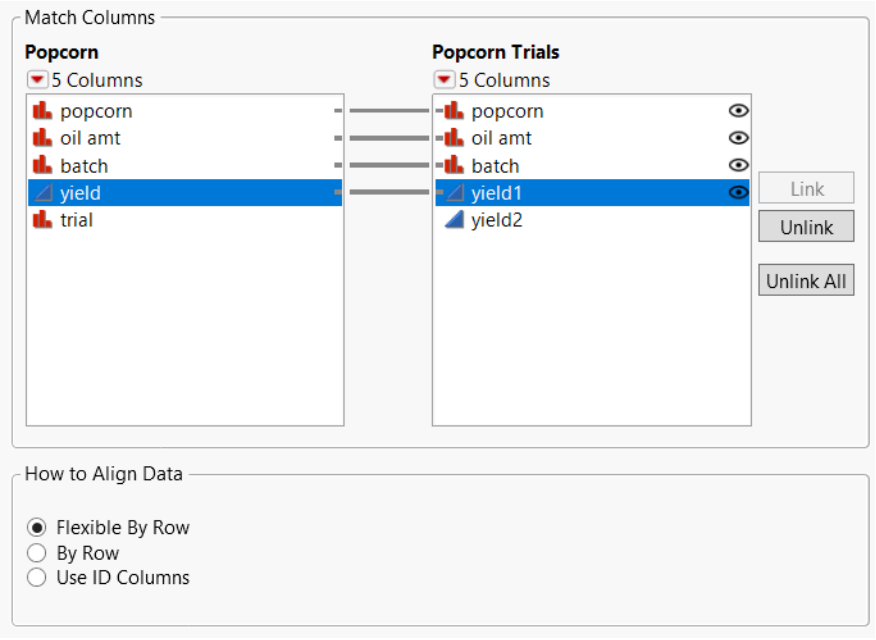
Columns with the same name are automatically matched. Lines are drawn between each matched column. You can also manually link two columns.

1. Select **Help > Sample Data Library** and open Popcorn.jmp and Popcorn Trials.jmp.

2. Show Popcorn Trials.jmp and select **Tables > Compare Data Tables**.
3. From the With list, select Popcorn.jmp.

   Popcorn Trials.jmp should automatically be selected from the Compare list.
4. In the Match Columns pane, select yield and yield1 and then click **Link**.

Figure 4.23 Manually Linked Columns



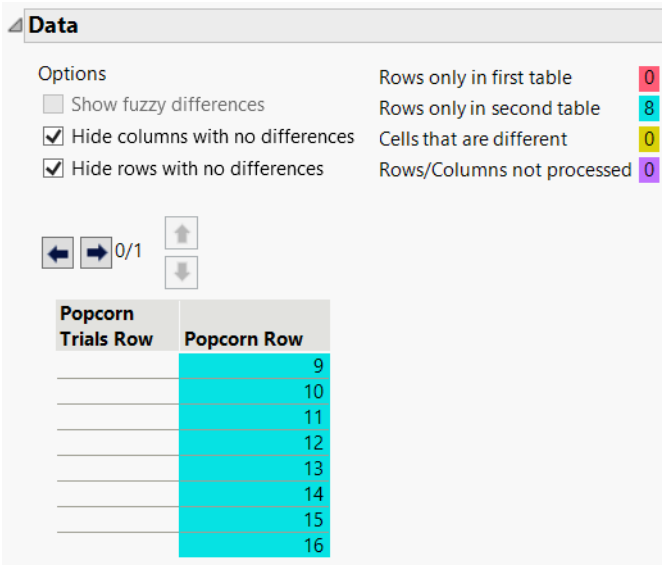
Columns that have the same name are automatically linked.

Note: If you click the Compare Data icon, the linked columns will not be compared. The values for that pair of columns will show up in the results, but they won't be compared.

5. Click **Compare**.

   The results are shown in the Data report.

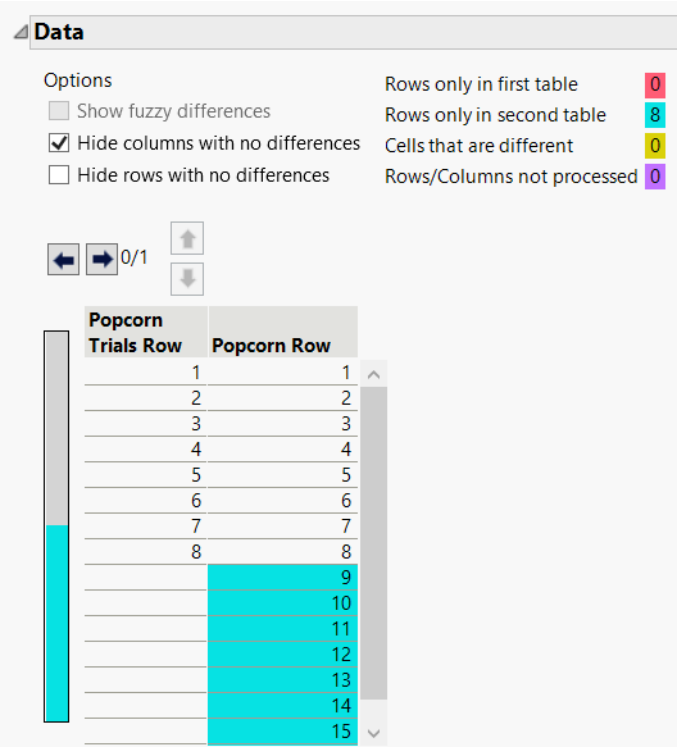
Figure 4.24 Comparing Columns



The first eight rows are the same so they are not shown. The remaining rows (colored blue) are only in the second table, Popcorn.jmp.

6. In the Data report, deselect **Hide rows with no differences**.

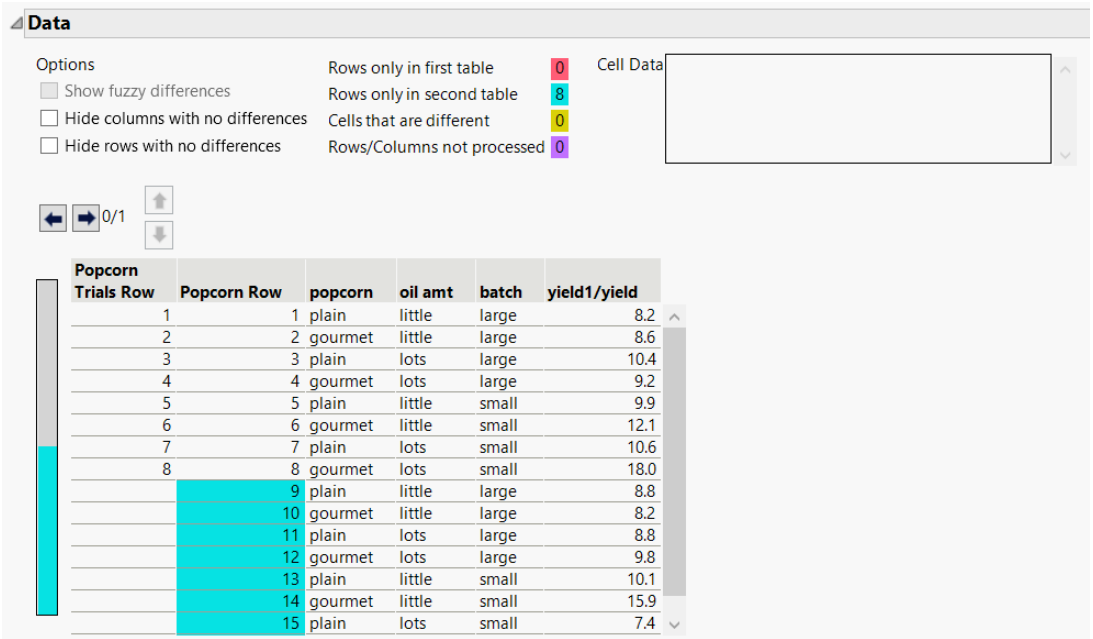
Figure 4.25 Showing Rows with No Differences



The first eight rows are shown because they are the same in both data tables.

7. Deselect **Hide columns with no differences.**

Figure 4.26 Showing Columns with No Differences



Popcorn Trials Row	Popcorn Row	popcorn	oil amt	batch	yield1/yield
1	1	plain	little	large	8.2
2	2	gourmet	little	large	8.6
3	3	plain	lots	large	10.4
4	4	gourmet	lots	large	9.2
5	5	plain	little	small	9.9
6	6	gourmet	little	small	12.1
7	7	plain	lots	small	10.6
8	8	gourmet	lots	small	18.0
	9	plain	little	large	8.8
	10	gourmet	little	large	8.2
	11	plain	lots	large	8.8
	12	gourmet	lots	large	9.8
	13	plain	little	small	10.1
	14	gourmet	little	small	15.9
	15	plain	lots	small	7.4

Columns that are in both data tables and that match are shown. You might select this option to give more context to the matched data.

Note: If the data in a cell isn't 'completely shown, select the text and then view the data in the Cell Data box.

Data Table Compare Options

Align Data Options

Flexible by Row Searches for common rows to align. Consider this option for smaller data tables if you think that the data tables are nearly the same.

By Row Compares rows one by one. Consider this option if you already know that the rows should line up row by row. The comparison will run much faster. However, if the default Flexible by Row was selected, and the comparison is taking too long, you might want to select this option.

Use ID Columns Uses the selected ID column to compare rows. The rows of the data table are uniquely identified by the values of the ID column. Consider this option if the data tables are large, sorted differently, or have missing rows. You can select more than one column.

Fuzzy Compare Options

Ignore missing Ignores missing data.

Allow Relative Error Enables you to specify the relative error rate for numeric data. The numeric values are considered equal if they are within the relative error rate that you specify. The smaller the relative error rate, the more precise the comparison.

Ignore case Disregards case when comparing text.

Ignore whitespace Disregards whitespace when comparing text.

Data Options

Show fuzzy differences Shows differences in numeric and string data that are approximately the same. Works with the value in the Relative Error field to remove insignificant differences.

Hide columns with no differences Shows or hides all matched columns.

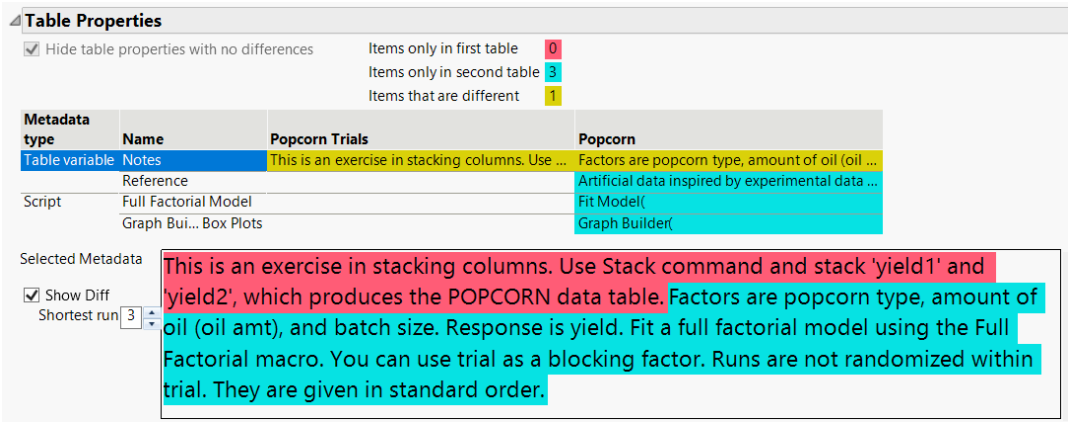
Hide rows with no differences Shows or hides rows that contain matching data.

Compare Table Properties

Click the red triangle and select **Compare Table Properties** to see differences in table scripts and variables. Figure 4.27 shows that the table variables and scripts are different. To see the full variable or script, select the line and view the selected metadata.

In this example, both notes differ, and the reference variable and scripts are only in Popcorn.jmp. The Notes variable is selected so that you can see it contents in both data tables in the Selected Metadata box. The red shading indicates that the text is only in Popcorn.jmp. The blue shading indicates that the text is only in Popcorn Trials.jmp.

Figure 4.27 Different Table Variables



Deselect **Show Diff** to see the name of each data table and the complete contents of each Notes variable instead of only showing the differences.

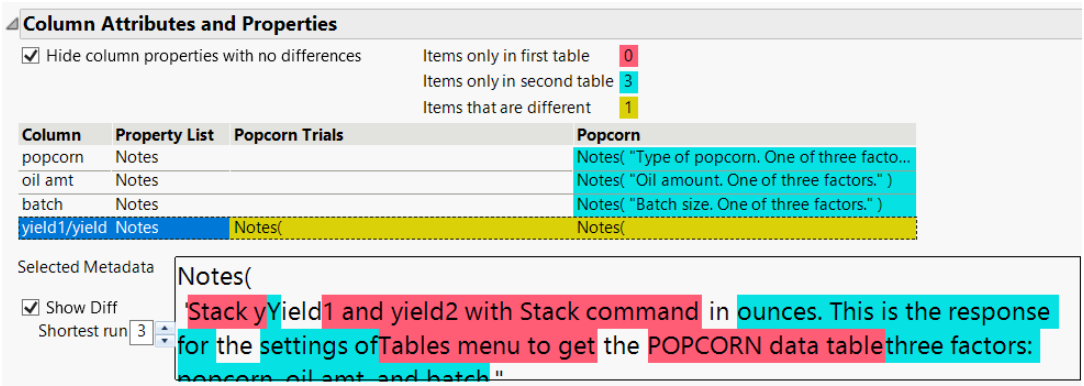
Shortest run is the smallest number of consecutive characters that are required to be the same (between the two files) before you can declare the characters as a *common subsegment*. A common subsegment has no background color because it is present in both files. The shortest run is set to 3 to prevent subsegments that are too short from showing up as common, which is typically unhelpful. For example, a shortest run of 1 means that any single character that is in both files could match. This leads to many very short segments of common text and differences, which is usually not a good reading experience.

Compare Column Attributes and Properties

Click the red triangle and select **Compare Column Attributes and Properties** to see differences in column notes, value colors, and the like.

Figure 4.28 shows that column notes are different in Popcorn.jmp and Popcorn Trials.jmp. The yield1/yield column is selected so that you can see the complete note and differences between the two notes in the Selected Metadata box.

Figure 4.28 Comparing Column Attributes and Properties



Assign Characteristics to Rows and Columns

This section describes how to exclude, hide, label, color, or mark rows and columns in order to customize the appearance of points in scatterplots and graphs.

- "How to Find Options for Rows and Columns"
- "Label Rows and Columns"
- "Assign Colors or Markers to Rows"
- "Create Color Themes in Graphs"
- "Delete Custom Color Themes in Graphs"
- "Select Matching Row State Cells"
- "Delete Row Characteristics"
- "Lock Columns"

How to Find Options for Rows and Columns

The menu for row actions can be accessed from the following places:

- the **Rows** menu in the main menu
- right-click a row
- the red triangle in the Rows panel
- the left red triangle in the upper left corner of the data grid

Similarly, the menu for columns actions can be accessed from the following places:

- the **Cols** menu in the main menu
- right-click a column
- the red triangle in the Columns panel
- the right red triangle in the upper left corner of the data grid

Label Rows and Columns

When you position the arrow cursor over a point in a plot, the point's label appears. By default, row numbers are used as labels. You can customize the labels:

- You can change the label to display column values instead of the row number.
- You can enable the label to always appear, not only when you position the cursor over points.

- A label or yellow tag icon ( ) appears beside the column name in the Columns panel, indicating that points on plots are identified by the column value. If there are multiple columns that are labeled, their values appear on plots separated by a comma.
- Data remain labeled until you select **Label/Unlabel** again.

To change the label to display column values

1. Highlight one or more columns whose values you want to appear as the label in plots.
2. Select **Cols > Label/Unlabel** from the menu or right-click and select **Label/Unlabel**.

To enable the label to always appear (not just when you position the cursor over points)

1. Highlight one or more rows whose label you want to always appear in plots.
2. Select **Rows > Label/Unlabel** from the menu.

To turn off labeling for rows or columns

1. Highlight the labeled rows or columns that you no longer want labeled.
2. Select **Label/Unlabel** from the **Rows** menu or **Cols** menu. You can also right-click columns or rows and select **Label/Unlabel.**

Tip: A photo can be displayed in the label when you hover over data. For example, the SAS Offices.jmp sample data table contains an Expression column that is labeled. A photo of each office is stored in the column. When you hover over a data point on the map, the photo for that row appears. See "Expression Role".

Assign Colors or Markers to Rows

Assign colors or markers to rows to differentiate rows in a data table.

- If you assign a color to a row, the points representing the values in that row are colored in the plot.
- If you assign a marker to a row, the point is replaced with the marker in the plot.
- You can also assign colors or markers based on column values.

Assign a Color to Rows

Assigning a color to selected rows means that the points in plots appear in the color that you select. In the data grid, the active color assigned to a row appears next to the row number.

To assign rows a color

1. Highlight one or more rows that you want to assign a color to.

2. Right-click the highlighted rows and select **Rows > Colors**.
3. Select one of the available colors.

Tip: To clear an assigned color from the selected rows, assign the color black.

Add Markers to Rows

To replace the standard points in plots with a marker, use the JMP markers palette. In the data table, these markers also appear next to row numbers.

1. Highlight one or more rows that you want to apply the marker to.
2. Right-click the selected rows and select **Markers**, and then select the marker shape.

   Select **Other** to create custom markers. You can type alphabetic characters, numerals, and other keyboard symbols.

Tip: To return to the default marker, select the initial dot marker.

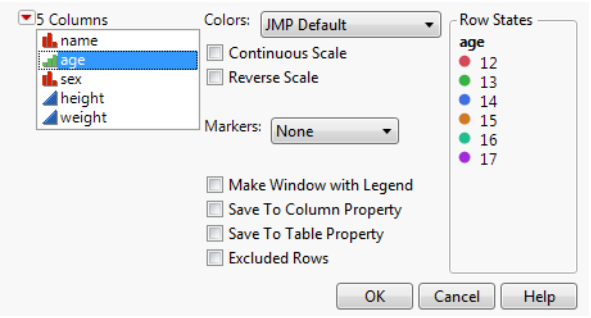
Assign Colors or Markers to Rows Based on Column Values

You can assign colors or markers to your data table rows based on the values found in a particular column. For example, in a column called Sex, you could assign all rows whose value is F a red circle marker. All rows whose value is M could have a green plus marker. These colors and markers replace the default black dot in plots and appear next to its row number in the data table.

To assign colors or markers to rows based on column values

1. Select **Rows > Color or Mark by Column**.
2. Select the column to color and or mark.

Figure 4.29 Color or Mark by Column



3. Select the **Colors** and **Markers** schemes to apply.

A preview of your selection appears under Row States.

4. (Optional) Select any additional options. See "Color or Mark by Column Options".
5. Click **OK**.
6. (Optional) To shade all rows according to their row state, right-click in the row numbers area within the data grid and select **Color Rows by Row State**.

   From then on, the rows are shaded with the color that you assign to the rows.

Color or Mark by Column Options

Colors Select a color theme to assign different colors to the rows in your data table. Color assignment is based on the values of the selected column.

Continuous Scale Assigns colors in a chromatic sequence based on the values in the highlighted column.

Reverse Scale Assigns colors in a reversed chromatic sequence based on the values in the highlighted column.

Markers Assigns a different marker to each row in your data table based on the values found in the column that you highlighted.

Make Window with Legend Includes a legend with your new characteristics so that you can easily identify which colors and markers correspond with which row.

Save To Column Property Saves the color and marker information as a column property. The rows in the selected column of the data table are colored, based on the color theme.

Save To Table Property Saves the color and marker information as a table property.

Excluded Rows Assigns colors or markers to rows that are excluded.

Create Color Themes in Graphs

JMP includes several color themes that can distinguish a range of values in a graph. You can also create your own color themes based on an existing color theme or create custom themes.

Note: When you select a default color theme, the colors are not applied to reports that are open. You need to rerun the existing reports to format them with the default color theme.

See "Delete Custom Color Themes in Graphs" for more information about deleting custom color themes.

To create a color theme

1. Select **File > Preferences > Graphs**.

2. To either create a new Continuous Color Theme or Categorical Color Theme, click the appropriate color theme.

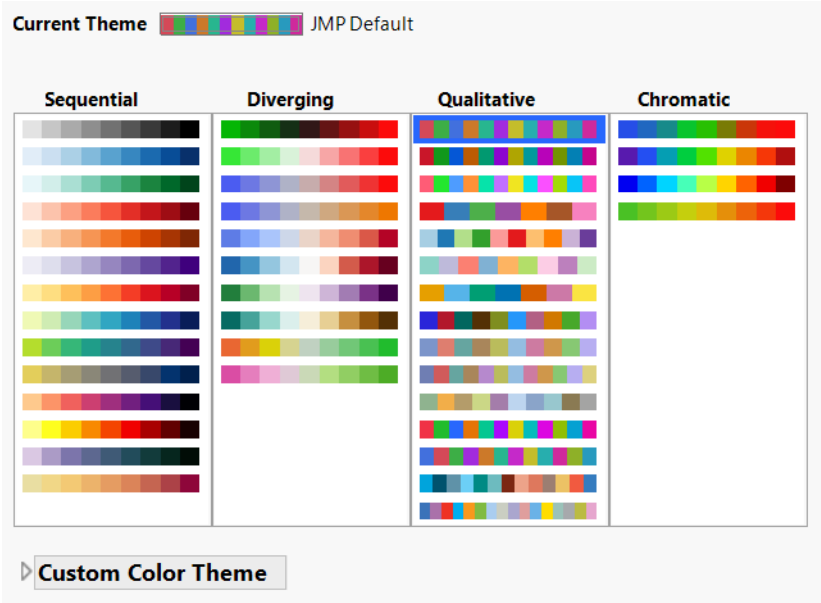
   If you are creating a new continuous color theme, the Continuous Color Themes window appears.

Figure 4.30 Continuous Color Themes Window

The sequential color “Cividis” is a form of Viridis that is designed for viewing with color limitations.

If you selected to create a new categorical color theme, the Categorical Color Themes window appears.

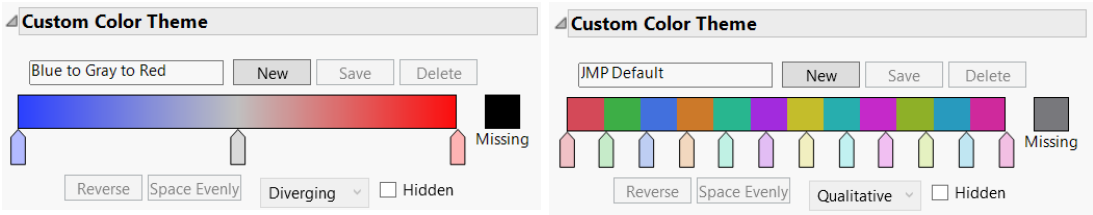
Figure 4.31 Categorical Color Themes Window



3. (Optional) To base the theme on an existing theme, select a color themes from the available themes.
4. Click the Custom Color Theme disclosure icon to show the Custom Color Theme panel.

   Figure 4.32 shows the color theme panels for both continuous and categorical themes (respectively).

Figure 4.32 Custom Color Theme Panel



5. Click **New** to create a new theme.

   A new color theme is created based on the selected color theme. A temporary name is assigned to the theme.
6. Type a new name in place of the temporary label. On Windows, do not press Enter. The window closes if you do so.
7. To modify the color theme, do any of the following:

- To modify the missing color, click the missing color to open a color palette. Select the desired missing value color and click **OK**.
- To modify the gradient of continuous color, move the sliders left or right.
- To add more colors to the gradient, click the color bar to choose a color. A new slider is displayed under the color bar.
- To change the color of a slider, click the slider to display the Color window and choose another color.
- To reverse the order of the colors on the gradient, click **Reverse**.
- To distribute the colors evenly on the gradient, click **Space Evenly**.
- To list the custom theme in the Sequential pane, select **Sequential** from the list.
- To list the custom theme in the Diverging pane, select **Diverging** from the list.
- To list the custom theme in the Chromatic pane, select **Chromatic** from the list.
- To prevent a theme from appearing in lists of color themes, select **Hidden**.
- To remove a color from the color theme, click the color's slider and drag the slider above or below the color bar.
- To discard your changes, click **Cancel**.

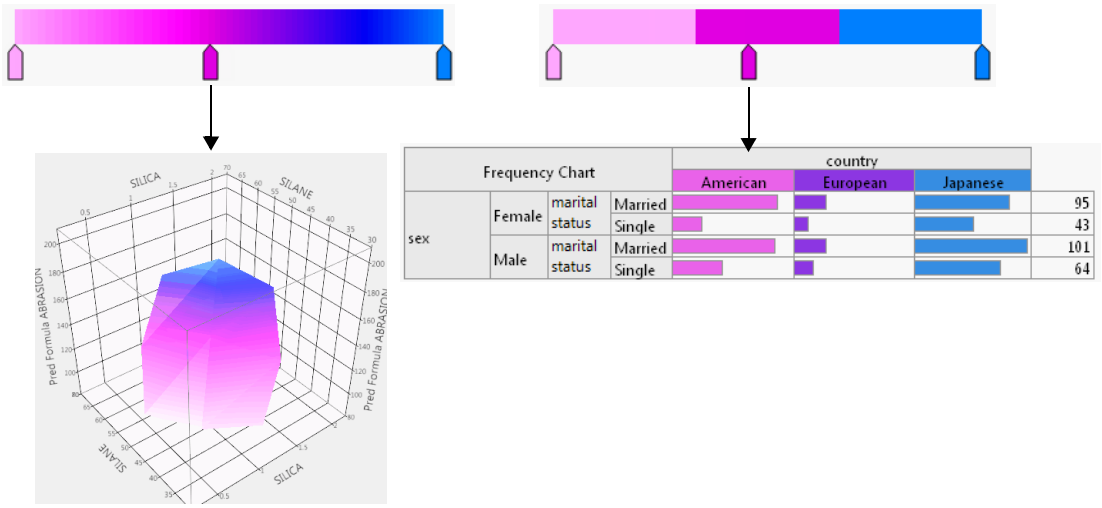
8. Click **Save** to save the custom color theme.

   The new custom color theme is appended to the contents of the selected pane.
9. Click **OK** to close the color theme window.

Continuous and Categorical Color Themes

The following figure shows examples of the two types of color themes in JMP, continuous and categorical. When a color theme is selected for continuous data, the colors are graduated (as shown on the left). When the same color theme is selected for categorical data, the color consists of distinct blocks of color. (as shown on the right).

Figure 4.33 Examples of Continuous and Categorical Color Themes



Custom Color Themes

Custom color themes can be applied in the same way as built-in color themes:

- You can select custom color themes as defaults from the **Continuous Color Theme** and **Categorical Color Theme** drop-down menus in the Graphs preferences. Only continuous color themes are available for continuous data. All color themes are available for categorical data.
- You can apply the custom color themes to components such as markers and data table rows. See "Assign Colors or Markers to Rows Based on Column Values".
- In certain reports, such as treemaps and surface plots, you can select specific custom color themes. See *Essential Graphing*.

Use Custom Color Themes on Multiple Computers

In Windows, the color themes that you create are defined in the JMP preferences file called JMP.PFS. If you use JMP on more than one computer (for example, at home and at work), you can copy the color theme definitions from one JMP preferences file to another. Custom colors are then available on both computers.

In the preferences file, the code for a custom color theme looks like this:

```
Add Color Theme(
{"Pink to Blue", 2051, {{255, 168, 255}, {255, 0, 255}, {0, 128, 255}}}
),
```

In this example, the name of the color theme is "Pink to Blue." The Red/Green/Blue (RGB) values for each color slider are located in brackets. The first slider defines the RGB values 255, 168, and 255. The second and third groups of brackets define colors for the second and third sliders.

Note: The second value (2051) is the category of the color theme, such as Categorical or Continuous. You can omit the category, but JMP always includes the information in the preferences file or when `Show Preferences()` is run.

In a text editor (such as Microsoft Notepad) add this color theme to the preferences file on your other computer. The preferences file is located in your Users folder within the JMP or JMPPro folder.

C:/Users/<user_name>/AppData/Roaming/SAS/JMP/16

C:/Users/<user_name>/AppData/Roaming/SAS/JMPPro/16

Note: To see the preceding folders, you must configure Windows Explorer to show hidden files and folders. For more information, refer to the Windows help.

To transfer color themes to another Windows computer

1. On the computer that contains the customized JMP preferences, select **File > New > Script**.

   The Script window appears.
2. Type the following JSL function:

   ```
   Show Preferences()
   ```
3. Click **Run Script** .

   Your customized preferences are written to the log.
4. Select **View > Log** (or display the open log).

   The custom color theme that you created appears, for example:

   ```
   Add Color Theme(
   {"Pink to Blue", 2051, {{255, 168, 255}, {255, 0, 255}, {0, 128, 255}}}
   ),
   ```

   This definition might be in the middle of other customized preferences that appear in the log.
5. Save the log as Log.jsl and open the file on the computer whose preferences you are updating.
6. On the computer whose preferences you are updating, close JMP.
7. Make a backup of JMP.PFS, and then open the original JMP.PFS in a text editor.
8. Copy and paste the custom color definition from Log.jsl to JMP.PFS. The definition goes after `Preferences(` as shown in the following example:

```
Preferences(
Add Color Theme(
{"Pink to Blue", 2051, {{255, 168, 255}, {255, 0, 255}, {0, 128, 255}}}
),
);
```

Note: Be sure to include the closing parenthesis and comma. The code does not need to be indented. You can put the code in any valid location. Pasting it after `Preferences(` helps ensure that you do not delete any necessary parentheses or commas.

9. Save the file.

   If you open JMP and the new color definition is not displayed in the preferences, delete the updated preferences file and add the definition to the original preferences file. Make sure that you copy and paste the definition in the correct location.

Delete Custom Color Themes in Graphs

1. Select **File > Preferences > Graphs**.
2. To either delete a color theme, select either the Continuous or Categorical Color Theme.

   The relevant Color Themes window appears.
3. Click the Custom Color Theme disclosure icon to show the Custom Color Theme panel.
4. From the appropriate pane, select the custom color to delete.

   Note: You can delete only custom color themes.
5. Click **Delete**.
6. Click **OK** to save your changes and close the Color Themes window.

Select Matching Row State Cells

You can select a cell in a row state column and then select all cells that contain the selected cell's attributes. This option enables you to select cells with any combination of row state attributes, such as Exclude, Hide, color, or marker. For example, if a cell has a red marker and is excluded, and you select matching cells, all cells in excluded rows with red markers are selected.

1. Right-click the cell whose row state you want to select.
2. Select **Select Matching Cells**.

   The cells with matching row states are selected.

Delete Row Characteristics

To clear all row states in the data table, select **Rows > Clear Row States**. To clear row states only in selected rows, select **Rows > Clear Selected Row States**.

All rows become included, visible, unlabeled, and show in plots as black dots. The **Clear Row States** command does not affect row states saved in row state columns.

Lock Columns

You can lock a column in place so that when you scroll horizontally, the column remains visible. Highlight the columns and select **Cols > Scroll Lock/Unlock**. Note the following:

- Hidden columns cannot be scroll locked.
- The name of a locked column appears in italics in the Columns panel.
- Scroll locked columns are moved to the left in the data grid. Once you unlock them, they are not moved back to their original locations in the data table, but remain on the left.
- Columns remain scroll locked until you highlight the columns and select **Scroll Lock/Unlock** again.

Restructure Data

JMP provides several ways to restructure data in a data table, such as creating a column from multiple columns, creating indicator columns, and recoding data.

- "Split a Data Table Column into Multiple Columns"
- "Create a New Column by Text Matching"
- "Make Indicator Columns"
- "Combine Columns in Data Tables"
- "Compress Selected Columns in Data Tables"
- "Make a Binning Formula in a Data Table Column"
- "Convert Labels to Codes and Codes to Labels"
- "Make a New Formula Column"
- "Transform Columns in a JMP Platform"
- "Recode Data in a Column"

Split a Data Table Column into Multiple Columns

Use the **Text to Columns** option to make a character column with delimited fields into multiple columns. Highlight a column from a data table and select **Cols > Utilities > Text to Columns.** The maximum number of delimited fields across all rows determines the number of new columns created.

Note: Text to Columns is case-sensitive.

The **Text to Columns** window has the following options:

Delimiter Specify text, such as a comma, to indicate how the data in the source column is organized into new columns. For example, if the original cell reads "NY, NJ, PA," and the delimiter is a comma, three new columns contain "NY", "NJ", and "PA".

TAB Specifies tabs as delimiters.

NEWLINE Specifies newline characters (\r, \n, and \r\n) as delimiters.

Make Indicator Columns Makes new columns that are named after the distinct fields in the source column with cell values of either 0 or 1.

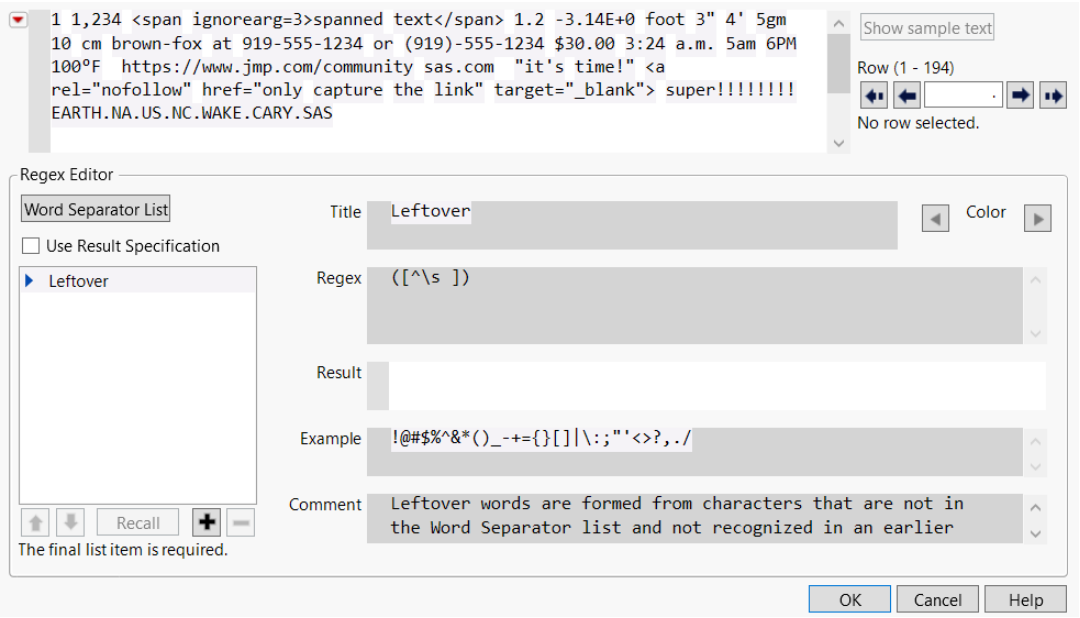
Include Missing Allows any empty rows to be counted as a category. An additional column named Missing is added to the data table. A value of 1 indicates an empty row.

Create a New Column by Text Matching

The New Column by Text Matching option enables you to use regular expressions (which identify patterns in text) to match unstructured text in a column and then create a new column from the matched text. This feature is an alternative to using the Text Explorer platform to write regular expressions and then save the results to a column.

1. Select **Help > Sample Data Library** and open Pet Survey.jmp.
2. Select the Survey Response column.
3. Select **Cols > Utilities > New Column By Text Matching**.

Figure 4.34 Regular Expressions Editor



4. Below the Leftover box, click the Add button ✚.
5. Select **Blank** from the list and click **OK**.
6. Next to Title, enter *Cat*.
7. Next to Regex, enter *(cat(?:s?))([\s\.])*.

   This expression finds “cat”, optionally followed by “s”, with either a space or period afterward.

   Note: If the regular expression has an error, an X icon ✘ appears to the left of Cat.

8. Click the Add button ✚ again.
9. Select **Blank** from the list and click **OK**.
10. Next to Title, enter *Dog*.
11. Next to Regex, enter *(dog(?:s?))([\s\.])*.
12. (Optional) To save the script so that you do not have to repopulate the Regular Expressions Editor after closing and reopening it, select one of the Save Script options from the red triangle menu.
13. Click **OK**.

Figure 4.35 Data Table with New Columns

	Survey Response	Survey Response 2	Survey Response 3
1	The tabby cat do...	cat cat	
2	In the winter I ma...		dog
3	My dogs are not ...		dogs
4	The dog chases c...		dog
5	My dog barks at ...		dog
6	My dogs are so s...		dogs dog
7	I have two dogs t...		dogs
8	My cat slid back...	cat	
9	The dogs bark lik...		dogs

The Use Result Specification option in the Regular Expressions Editor gives you control over which matched text appears in the new column:

- If you don't select Use Result Specification (the default setting), the result in the new column is whatever is produced by the regular expression in the right side of the editor.
- If you select Use Result Specification, then the result in the new column is whatever is produced by the Result expression in the right side of the editor. If your regular expression is *Air(port)*, then the new column contains the word *port* when found in the original column.

For more information about the Regular Expression Editor, see *Basic Analysis*.

To learn more about regular expressions, visit Regular-Expressions.info.

Make Indicator Columns

Convert a categorical column into multiple indicator columns based on each distinct category. The columns are named after each level in the column. For example, when you make an indicator column from sex in Big Class.jmp, the indicator columns are named F and M.

Highlight a column in a data table and select **Cols > Utilities > Make Indicator Columns**. You can then indicate whether you want to prepend the original column name to the indicator column names (as in sex_F and sex_M) and include missing values.

Multiple columns with values of either 0 or 1 are created. A value of 1 indicates that the original column contains that specific category.

If the given column has the Multiple Response modeling type or Multiple Response column property, the categories are determined from the set of responses.

Combine Columns in Data Tables

The Combine Columns option is the opposite of Text to Columns. Instead of making multiple columns, you can combine a set of columns into one character column with delimited fields.

To combine indicator columns, follow these steps:

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Select the columns, Floss After Waking Up, Floss After Meal, and Floss Before Sleep.
3. Select **Cols > Utilities > Combine Columns.**
4. Type "Combined Floss" for the column name, and keep the default delimiter as a comma.
5. Select **Selected Columns are Indicator Columns** and click **OK**.

Figure 4.36 Combined Floss Column

Combined Floss	Floss After Waking Up	Floss After Meal	Floss Before Sleep
Floss After Waking Up	1	0	0
Floss After Meal	0	1	0
Floss After Meal	0	1	0
Floss After Meal	0	1	0
Floss After Meal	0	1	0
Floss Before Sleep	0	0	1
Floss After Meal	0	1	0
Floss Before Sleep	0	0	1
	0	0	0
Floss After Meal	0	1	0
Floss Before Sleep	0	0	1
Floss After Meal,Floss Before Sleep	0	1	1
Floss After Waking Up,Floss Before Sleep	1	0	1
Floss After Waking Up,Floss After Meal,Floss Before Sleep	1	1	1
Floss After Waking Up,Floss Before Sleep	1	0	1
Floss After Waking Up,Floss After Meal	1	1	0

The selected columns are represented in the Combined Floss column with each field separated by a comma. Only the columns that have a value of 1 are represented in the combined column for each given row.

Note: Value labels show a label in the data table instead of a value. A label appears for each instance of the value in the combined column. You can show the original values by double-clicking a label within a cell. To avoid using value labels, select **No Value Labels** when you combine the columns.

Compress Selected Columns in Data Tables

JMP lets you compress columns in a data table to minimize the size of the file and reduce the amount of memory required to analyze data. This feature is helpful when numeric columns contain many small integers or when any column contains fewer than 255 unique values. For example, compressing columns in a data table with 389 columns and 85,000 rows might decrease the file size from 250MB to 33MB, depending on the type of data.

When you compress columns, JMP verifies whether the data can be stored in a more compact form based on the data type:

- In character columns with fewer than 255 unique values, the List Check property is added to the column where appropriate (Figure 4.37). When the preference Allow 16 Bit List Check Compression is selected, the List Check property is also added to character columns that have more than 255 unique values.

  The List Check property restricts the values in the selected column to valid values. The List Check property is not applied when the number of values in the selected column is too great. For example, if the number of values is almost the same as the number of rows, the data table does not add the List Check property to the column.
- For numeric columns, only those with the Best, Fixed Dec, or Data format are compressed. Data is compressed to 1-byte, 2-byte, or 4-byte integers when possible (Figure 4.38). For more information about short integers, see "The Short-Integer Format".

  A numeric column with non-integer values can also be compressed if there are fewer than 255 unique values. In this case, the List Check property is added to the column.

Notes:

- To automatically compress a column that has less than or equal to 65,535 unique values, select the **Allow 16 Bit List Check Compression** preference in the General group. The List Check property is also added to the column.
- In a column with the List Check property, you can enter only a value that is in the list. Otherwise, JMP warns that the cell contains invalid data when you try to enter the new value. See "List Check".

Figure 4.37 List Check Property Added to a Compressed Character Column

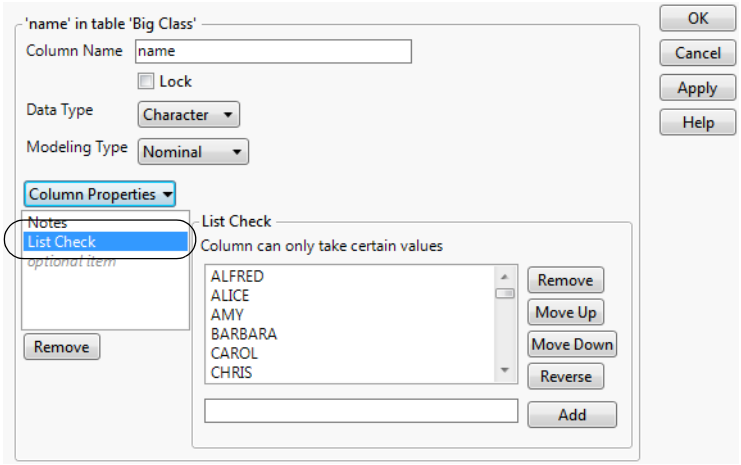


Figure 4.38 Column Info Window Showing Numeric Column before and after Compression

To compress columns, select one or more columns and select **Cols > Utilities > Compress Selected Columns**. (Select all columns if you do not know which columns can be compressed.)

The column or columns are compressed if possible. The log shows which columns were compressed and how they were compressed. (Select **View > Log** to show the log.)

Note: To compress a numeric column manually, set your Tables preferences to allow short numeric data and then change the column's data type to **1-byte integer**, **2-byte integer**, or **4-byte integer**. For more information about this preference, see "Preferences for Data Tables".

Make a Binning Formula in a Data Table Column

You can distribute your data into bins using the Make Binning Formula option. Select the column or columns that you want to divide into bins, and select **Cols > Utilities > Make Binning Formula**. A histogram of the bins and a Value Label legend are shown, along with a red triangle menu that contains options for defining the cutpoints. There are several ways to add cutpoints, remove cutpoints, and adjust the bin sizes.

- Right-click in the histogram to add or remove cutpoints.
- Click and drag the vertical lines in the histogram to adjust the bin sizes.

- Edit the boxes next to the cutpoint value labels to edit the cutpoint values and adjust the bin sizes.
- Use the add and remove buttons above the Value Label legend to add and remove rows, which is equivalent to adding and removing cutpoints.
- Use the undo and redo buttons to undo or redo actions done on the bin settings.

There is an option to add formula columns for the cutpoints to the data table.

Right-Click Options for Make Binning Formula

Right-click the histogram in the Make Binning Formula window to select the following options:

Add Cutpoint Adds a cutpoint line to the histogram in the location that was clicked.

Remove Cutpoint Removes a cutpoint line. This option is available only if you right-click on an existing cutpoint.

Show Counts Shows or hides the counts above the bars.

Show Percents Shows or hides the percentages above the bars.

Subset Creates a subset data table based on the current selection.

Fill Pattern Specifies the fill pattern for the bars in the histogram.

Red Triangle Options for Cutpoints

The Cutpoints red triangle contains the following options:

Fill using Equal Width Bins Specifies a start (offset) and a bin width. JMP fills in the table until it goes past the largest possible values. For example, if you enter offset=50 and width=10 and your largest data value is 95, you will get the following bins: 50, 60, 70, 80, 90, 100.

Fill using Bin Count Specifies a start, end, and a bin count. JMP divides the space between the start and end equally by the bin count. If you change the bin count, the reported bin width is automatically updated in the window.

Fill using Percentiles Specifies a percentage between 1 and 50. JMP divides the range of the data by that quantile. For example, if you specify a percentile of 20, you get five equal weight bins.

Fill using Mean/StdDev Specifies the bin cutpoints to be set at the mean of the data plus the first, second, or third standard deviation from the mean based on your selection in the window.

Fill using Jenks Natural Breaks Specifies a bin count and creates bin cutpoints so that the variance within each bin is minimized and the variance between the bins is maximized.

Bin Label Specifies whether value labels are shown instead of the data values.

Use Value Labels Shows a label instead of the value shown in the new formula column defining the bins.

Use Range Labels Includes the lower and upper values for each range in the label.

No Labels Uses the lower edge value as the label.

Character Applies the Character data type. The formula produces the labels.

See "Value Labels".

Tip: Value Labels are recommended in most platforms, many of which do not support range labels. In the Categorical platform, you must use value labels. On some axes, you might find that range labels more clearly identify the values, or you can try adjusting the offset and width for the axis.

Bin Label Style Specifies a preset that defines what the Value Labels or Character output look like in the table.

Low–High Defines the Value Labels by the lowest and highest points of each bin. The format is Low–High.

Low to High Defines the Value Labels by the lowest and highest points of each bin. The format is Low to High.

Low, High Defines the Value Labels by the lowest and highest points of each bin. The format is Low, High.

Low–High-1 (integers) Defines the Value Labels by the lowest point and one less than the highest point of each bin. The format is Low–High-1.

Low to High-1 (integers) Defines the Value Labels by the lowest point and one less than the highest point of each bin. The format is Low to High-1.

[Low, High) (mathematical) Specifies that the lower value is included in the range and the upper value is not. For example, if your range is [50,60), then a value of 50 would be in the bin, but a value of 60 would not be.

Low Defines the Value Labels by the lowest point of each bin.

Midpoint Defines the Value Labels by the midpoint of each bin.

Integer Sequence Defines the Value Labels by a sequence of consecutive integers. Define the starting integer in the Integer Sequence Labels window.

Uppercase Alphabetical Defines the Value Labels as uppercase letters in alphabetical order.

Lowercase Alphabetical Defines the Value Labels as lowercase letters in alphabetical order.

Custom Enables you to enter custom Value Labels.

Column Format Specifies how the Value Labels are formatted. By default, the format of the column is applied. However, you might want to override the setting (for example, changing the format to Dollar). This setting is most useful if the column format is something like Best, but you really want your bins to have no decimals. Instead of cleaning up all of the value labels, you specify the format.

Histogram Specifies whether counts or percentages appear above the bars.

Show Points Shows or hides the observed data points below the bars.

Rescale Axis Resets the axis to the original scale.

Recall Populates the Make Binning Formula window with the last binning actions that you performed.

Make All Like X (Appears only if multiple columns are selected.) Applies the choices made for the first column (X) to the remaining columns in the dialog.

Make Formula Columns Creates the formula columns and closes the window.

Tip: Once you create a formula column, you can edit the formula by right-clicking on the column header and selecting **Edit Binning Formula**. This reopens the Make Binning Formula window. To update the binning formula column, edit the cutpoints and click **Update Column**.

Example of Making a Binning Formula

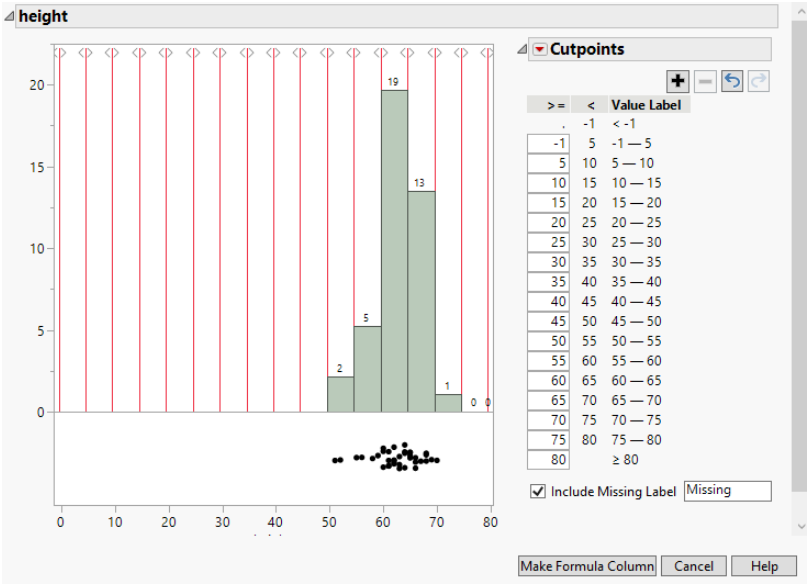
1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select the height column.
3. Select **Cols > Utilities > Make Binning Formula**.

   You want the range of values to appear as X-X, so in the Bin Label Style, keep the range set to **Low - High**.
4. Click the Cutpoints red triangle and select **Fill using Equal Width Bins**.
5. In the window that appears, change the **offset** to -0.5.

Tip: For integer data, setting the offset to -0.5 helps disambiguate values on the edge. In this example, one of the bins covers 59.5 to 64.5, so it is clear that 59 and 65 are not included in this bin.

6. Keep the **width** set to 5.
7. Click **OK**.
8. Keep the Bin Label setting of **Use Value Labels** so that you can see the range of values for the bin.

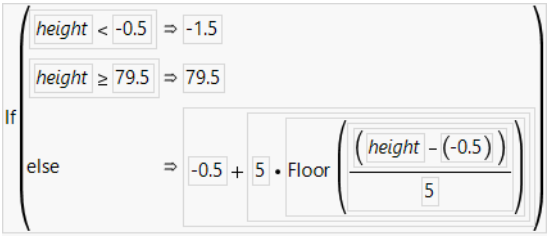
Figure 4.39 Completed Binning Window



9. Click **Make Formula Column**.

   A column called height Binned is added to the Big Class.jmp data table.
10. To see how the formula is calculated, right-click the height Binned column and select **Formula**.

Figure 4.40 Formula



Convert Labels to Codes and Codes to Labels

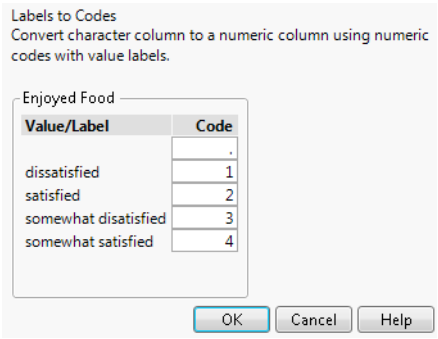
Suppose that you have a column of strings such as “dissatisfied”, “somewhat dissatisfied”, and “somewhat satisfied”, and “satisfied”. You can recode the values to “1”, “2”, “3”, and “4”. The original strings are placed in the Value Labels column property for that column. If the column has a List Check property, the property is removed. The column data type is changed to 1-byte Integer. Converting a column of strings to short numeric codes saves memory and can speed up processing.

You can also convert a numeric column of codes to labels. The column must have value labels. The column data type is changed to Character, and the Value Label column property is removed. This is, in effect, the opposite of converting labels to codes.

Convert Labels to Codes

1. Select the character column and select **Cols > Utilities > Labels to Codes**.

Figure 4.41 Converting Labels to Codes



   Default codes are assigned to the values.
2. Change the codes if desired and click **OK**.

   The column is recoded, and value labels are added to the column.

Convert Codes to Labels

1. Select the numeric column that contains value labels and select **Cols > Utilities > Codes to Labels**.
2. Change the labels if desired and click **OK**.

   The column is recoded with the original value labels.

Make a New Formula Column

To perform further analyses on your data, use the New Formula Column menu options from your existing data table. Formula columns use formulas or calculations to define column values.

Right-click a column heading in your data table and select **New Formula Column**. Choose from Transform, Character, Combine, Pairwise, Aggregate, Distributional, Date Time, Random, Row, or Vector to calculate column values. A new formula column is added to the data table. See "Transform Columns in a JMP Platform" for a description of these options.

Note: The same options exist in both the New Formula Column menu, and the right-click column menu in the launch window. However, performing these tasks in a launch window results in a temporary column, and New Formula Column adds a new column to the original data table.

Right-click options depend on the selected column's data type and the number of columns selected. If the selected column is a Character column, Character and Row options appear. See "Character Menu" and "Row Menu".

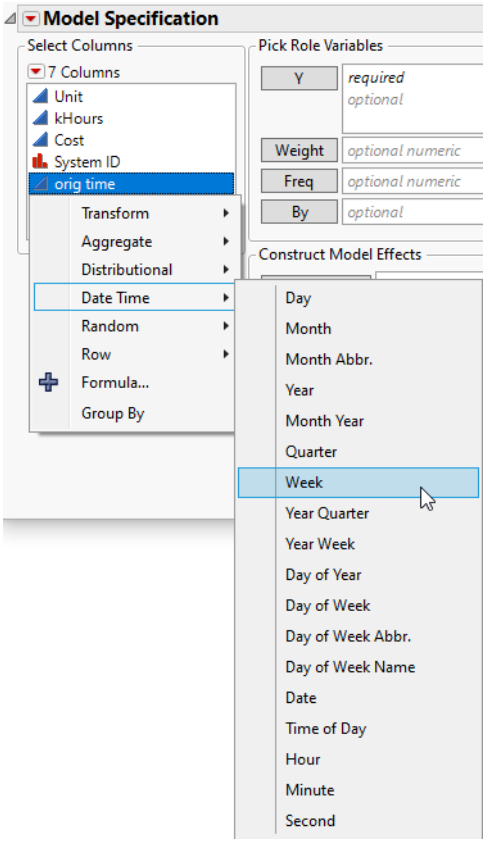
Transform Columns in a JMP Platform

Each launch window in JMP enables you to create one or more temporary transform columns for use in performing analyses. These transform columns are not part of the source data table and only can be used within the context of the current launch window. Transform columns use formulas or calculations to define the column values. Closing the launch window deletes any transform columns.

Each column listed in the Select Columns pane of the launch window includes an icon representing the column's modeling type (continuous, ordinal, or nominal) and the column name. Right-click a column name to create a transform column using Transform, Character, Combine, Pairwise, Aggregate, Distributional, Date Time, Random, Row, Vector, or Formula to calculate the column's values.

Right-click options depend on the selected column's data type and number of columns selected.

Figure 4.42 Example of Transform Column Menu



Group By For ordinal and nominal data, specifies the column to use for grouping data. A separate analysis is computed for each level of the specified column.

Notes:

- The transform column is available only in the current launch window. To make the transform column available outside of the current launch window, right-click the transform column and select **Add to Data Table**. The transform column is added to the source data table.
- You can paste a transform column into a Roles box on the launch window. For example, you might copy a transform column from a script. Right-click in the appropriate launch window Cast Selected Columns into Roles box and select Paste. This is an alternative to right-clicking the column in the Select Columns list, selecting the transform, and adding the transform column to a role.
- If the transform function is invertible, the following platforms use the original scale for the prediction profiler and to save the predicted values and formulas.

  - Fit Y by X
  - Standard Least Squares
  - Partition
  - Boosted Tree
  - Bootstrap Forest
- Define your own transform by writing a JSL script. See the *Scripting Guide*.

Transform Menu

Select a function from the Transform menu to create a transform column containing the calculations based on the selected function. See the Scripting Index in the Help menu or *Fitting Linear Models*.

Note: You can apply unary functions to multiple columns resulting in multiple transform columns.

Table 4.2 Descriptions of the Transform Menu Options

Round	Rounds date values. For example, in a chart of weeks, Round converts the date values into the first date of each week that they occur on.
Scale Offset	Enables you to specify a general linear transform. For example, you might multiply the value by 1.8 and apply an offset of 32 to convert Celsius temperatures to Fahrenheit.
Custom Binning	Opens the binning window which enables you to distribute the data into bins.
Square Root	Takes the square root of the values of the selected column.
Square	Calculates the square for the selected column values.
Log	Applies the natural logarithm transformation to the selected column.
Log x+1	Calculates Log(col+1).
Exp	Applies the exponential transformation to the selected column.
Log10	Applies the base-10 logarithm transformation to the selected column.
Pow10	Calculates 10 raised to the power of the selected column values.

Table 4.2 Descriptions of the Transform Menu Options *(Continued)*

Cube Root	Calculates the cube root for the selected column values.
Cube	Calculates the cube for the selected column values.
Reciprocal	Calculates the reciprocal (1/column) for the selected column values.
Absolute Value	Calculates the absolute value for the selected column values.
Negation	Calculates the negative for the selected column values.
Arrhenius	Applies the Arrhenius transformation to the variable T (temperature in degrees Centigrade): $X = \frac{11604.5181215503}{T + 273.15}$ This is the component of the Arrhenius relationship that is multiplied by the activation energy.
Arrhenius Inverse	Applies the inverse of the Arrhenius transformation to the variable X: $T = \frac{11604.5181215503}{X} - 273.15$
Logit	Calculates the inverse of the logistic function for the selected column (where p is in the range of 0 to 1): $\text{Logit}(p) = \log\left(\frac{p}{1-p}\right)$
Logistic	Calculates the logistic (also known as Squish and Logist) function for the selected column (where the result is in the range of 0 to 1): $\text{Logistic}(x) = \frac{1}{(1 + e^{-x})}$

Table 4.2 Descriptions of the Transform Menu Options *(Continued)*

Logit Percent	Calculates the logit as a percent for the selected column (where *pct* is a percent in the range of 0 to 100): $\text{LogitPct}(pct) = \log\left(\frac{\left(\frac{pct}{100}\right)}{1-\left(\frac{pct}{100}\right)}\right)$
Logistic Percent	Calculates the logistic (or logist) as a percent for the selected column (where the result is in the range of 0 to 100): $\text{LogisticPct}(x) = \frac{100}{(1+e^{-x})}$

Combine Menu

Select multiple columns to access the Combine menu. The Combine menu creates a transform column containing the calculations based on the selected function.

The following functions are included in the menu:

Sum Calculates the sum of the first and second columns (A + B).

Difference Calculates the difference between the first and second columns (A - B).

Difference (reverse order) Calculates the difference between the second and first columns (B - A).

Product Calculates the product of the first and second columns (A X B).

Ratio Calculates the ratio of the first column to the second column (A / B).

Ratio (reverse order) Calculates the ratio of the second column to the first column (B / A).

Minimum Returns the minimum value of the selected columns.

Maximum Returns the maximum value of the selected columns.

Average Returns the average value of the selected columns.

Standard Deviation Calculates the standard deviation of the values in the selected column.

Median Calculates the median value for the selected column.

Quantile Calculates the quantile of the specified percentage for the selected column.

Geometric Mean Returns the *n*th root of the product of the data.

Pairwise Menu

Select a function from the Pairwise menu to create pairwise results for two or more columns. This menu appears only when the number of selected columns is even and greater than or equal to four.

Sum Calculates the sum of the first and second columns (A + B).

Difference Calculates the difference between the first and second columns (A - B).

Difference (reverse order) Calculates the difference between the second and first columns (B - A).

Product Calculates the product of the first and second columns (A X B).

Ratio Calculates the ratio of the first column to the second column (A / B).

Ratio (reverse order) Calculates the ratio of the second column to the first column (B / A).

Minimum Returns the minimum value of the selected columns.

Maximum Returns the maximum value of the selected columns.

Average Returns the average value of the selected columns.

Geometric Mean Returns the *n*th root of the product of the data.

Aggregate Menu

Select a function from the Aggregate menu to create a transform column containing the statistics calculated from the selected column (or part of a column if you specified a Group By column).

Note: The **Group By** option is useful for these functions.

The following functions are included in the menu:

Mean Returns the average value of the selected column.

Sum Calculates the sum of the values in the selected column.

Count Calculates the number of values in the selected column.

Median Calculates the median value for the selected column.

Quantile Calculates the quantile of the specified percentage for the selected column.

Minimum Returns the minimum value of the selected column.

Maximum Returns the maximum value of the selected column.

Standard Deviation Calculates the standard deviation of the values in the selected column.

Distributional Menu

Select a function from the Distributional menu to create a transform column containing the statistics calculated from the selected column. See the Scripting Index in the Help menu.

The following functions are included in the menu:

Center Subtracts the column mean from each value across all rows of the selected column.

Standardize Calculates the column value minus the mean divided by the standard deviation across all rows of the selected column.

Range 0 to 1 Scales the data up or down so that the minimum value is greater or equal to 0, and the maximum value is less than or equal to 1.

Box Cox Transforms the data using the Box-Cox equation. See *Fitting Linear Models*.

Johnson Normalizing Transforms the data using one of the Johnson equations. The new column name indicates either Johnson Su, Johnson Sb, or None, depending on which equation was used to calculate the new data.

Informative Missing Creates two columns. The Informative column replaces missing values with the column mean. The Is Missing column indicates 1 for missing values, and 0 otherwise.

Rank Returns the rank, ranging from 1 as the lowest, with row-order tie-breaking.

Rank (reverse order) Returns the rank, ranging from 1 as the highest, with row-order tie-breaking.

Cumulative Probability Calculates the cumulative probability: Col Rank(col) / (Col Number(col) + 1).

Normal Quantile Calculates the quantile from a Normal distribution.

SHASH Calculates the cumulative distribution function (cdf) evaluated at x of the sinh-arcsinh (SHASH) distribution.

Random Menu

For numeric columns, select a function from the Random menu to create columns with random values.

The following functions are included in the menu:

Random Uniform Generates random numbers uniformly between 0 and 1.

Random Normal Generates random numbers that approximate a normal distribution with a mean of 0 and standard deviation of 1 if no arguments are used, or with the mean and standard deviation entered as arguments.

Sample without Replacement Shuffles the values randomly each time it's evaluated. The result for the first value affects the result for the second value.

Sample with Replacement Shuffles the values randomly each time it's evaluated. The result for the first value does not affect the result for the second value.

Date Time Menu

For column values containing date or time values, select a function from the Date Time menu to create a transform column containing values calculated from the selected column.

The following functions are included in the menu:

Day Returns the day of the month for the date in the selected column.

Month Returns the month number for the date in the selected column.

Month Abbr. Returns the abbreviated month for the date in the selected column.

Year Returns the year for the date in the selected column.

Month Year Returns the month number and year for the date in the selected column.

Quarter Returns the year's quarter (1, 2, 3, or 4) for the date in the selected column

Week Returns the number of the week in the year for the date in the selected column.

Year Quarter Returns the year and the year's quarter (1, 2, 3, or 4) for the date in the selected column.

Year Week Returns a string representing the ISO-8601 week of year format (for example, June 12, 2013 results in "2013W24").

Day of Year Returns the day of the year for the date in the selected column.

Day of Week Returns the day of the week for the date in the selected column.

Day of Week Abbr. Returns the abbreviated day of the week for the date in the selected column.

Day of Week Name Returns the full day of the week for the date in the selected column.

Date Returns the month, day, and year for the date in the selected column.

Time of Day Returns the time for the date in the selected column.

Hour Returns the hour part of the date in the selected column.

Minute Returns the minute part of the date in the selected column.

Second Returns the seconds part of the date in the selected column.

Character Menu

Select a function from the Character menu to create a transform column containing strings formed by the selected Character function.

The following functions are included in the menu:

Length Calculates the number of characters in each string in the selected column or columns.

Concatenate Concatenates the strings in the selected column or columns into a new string.

Concatenate with Space Concatenates the strings in the selected column or columns into a new string with each sub-string separated by a whitespace character.

Concatenate with Comma Concatenates the strings in the selected column or columns into a new string with each sub-string separated by a comma character.

Titlecase Converts the strings to title-case (for example, “Sheila Smith”).

Uppercase Converts the strings to uppercase (for example, “SHEILA SMITH”).

Lowercase Converts the strings to lowercase (for example, “sheila smith”).

First Word Extracts the first word from a character string in the selected column or columns.

Last Word Extracts the last word from a character string in the selected column or columns.

Rank Returns the rank, ranging from 1 as the lowest, with row-order tie breaking.

Rank (reverse order) Returns the rank, ranging from 1 as the highest, with row-order tie-breaking.

Row Menu

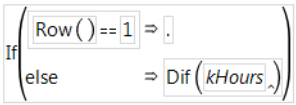
Select a function from the Row menu to create a transform column containing calculations determined by the selected Row function.

In addition to the functions described in the appendix, the following functions are included in the menu:

Row Returns the current row number.

Selected Returns the selected index.

Difference Calculates the difference of each value in the selected column using the formula:

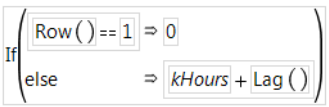


Note: The Difference function also supports the Group By option.

Lag Returns the value in the previous row for the selected column.

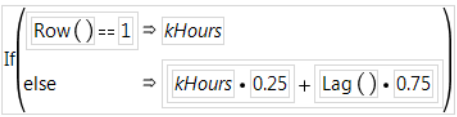
Lag Multiple Returns the values from multiple previous rows for the selected column.

Cumulative Sum Calculates the cumulative sum for each value in the selected column using the formula:



Note: The Cumulative Sum function also supports the Group By option.

Moving Average Calculates the exponentially weighted moving average, EWMA (using a smoothing parameter between 0 to 1.0) for each value in the selected column. The following example uses a smoothing parameter of 0.25:



Weighting Determines how the values are weighted. Incremental weighting is a ramp or triangle. The exponential moving average is EWMA or EMA.

Items Before Controls the size of the range (or window) by including the specified number of items before the current item in the average (in addition to the current item). -1 means all prior items.

Items After Controls the size of the range (or window) by including the specified number of items after the current item in the average (in addition to the current item). -1 means all following items.

Report missing values for partial window Controls how missing values are treated. By default, missing values are ignored.

Note: JMP evaluates the formula entered on-demand; complex formulas might require a lot of processing time.

Vector Menu

Select a transform from the Vector menu to create transform columns from vector data.

To Columns Expands the vector into individual number columns.

Sum Returns the sum of the arguments or of the values within the vector.

Min Returns the minimum value among the values within the vector.

Max Returns the maximum value among the values within the vector.

Loc Min Returns the first position of the minimum value.

Loc Max Returns the first position of the maximum value.

Transform Column Options

After creating a transform column, you can perform the following actions:

Rename Renames the transform column.

Add to Data Table Adds the transform column to the data table as a formula column.

Recode Data in a Column

Use the recoding tool to change all of the values in a column at once. For example, suppose you are interested in comparing the sales of computer and pharmaceutical companies. Your current company labels are Computer and Pharmaceutical. You want to change them to Technical and Drug. Going through all 32 rows of data and changing all the values would be tedious, inefficient, and error-prone, especially if you had many more rows of data. Recode is a better option.

Notes:

- To recode similar values within multiple columns, use the Recode option in **Cols > Standardize Attributes**. See "Standardize Column Attributes".
- To recode only column headings, select **Cols > Column Names > Recode Column Names**.

To recode data, follow these steps:

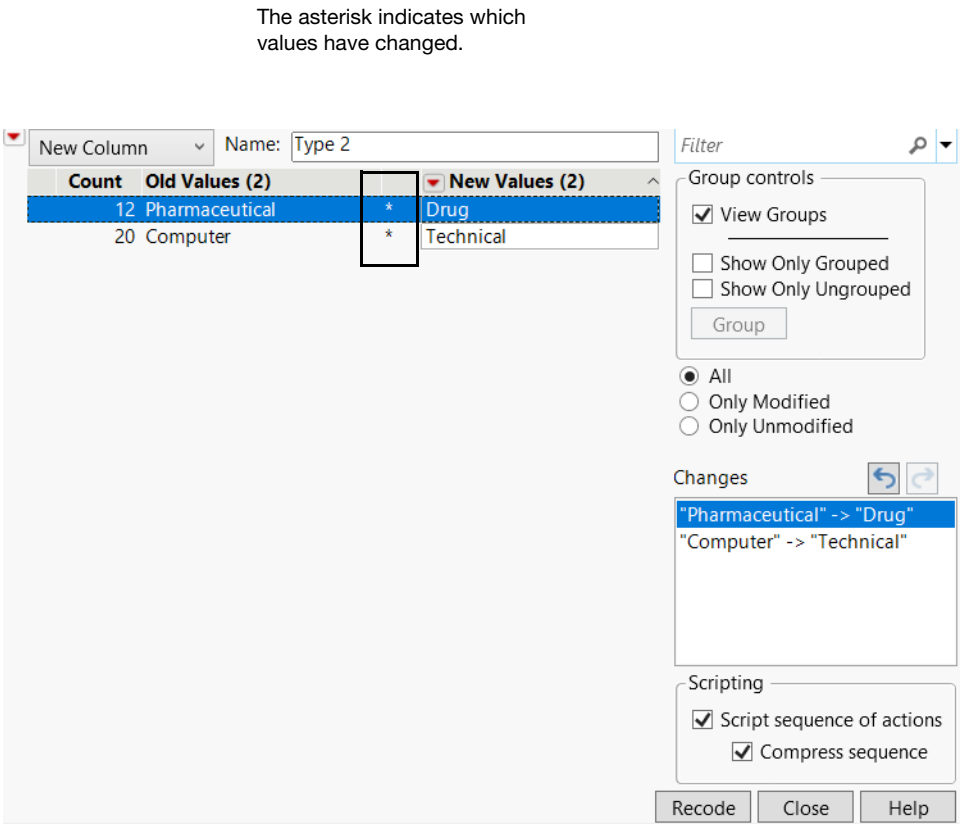
1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select the Type column by clicking once on the column heading.
3. Select **Cols > Recode**.

   Notice that New Column is selected by default to place the recoded values in a new column. The new column is named Type 2.

4. In the Recode window, enter the desired values in the **New Value** boxes. For this example, enter Technical in the Computer row, and Drug in the Pharmaceutical row.

   Notice that the recoded values appear in the Changes box.

Figure 4.43 Recode Window



5. Click **Recode**.

   The new values appear in the data table under a new column called Type 2.

Notes:

- By default, the recoded data appears in a new column. To always recode the column that you selected, select **File > Preferences** (Windows) or **JMP > Preferences** (macOS), select **Recode**, and then select **In Place** from the Commit Style list.
- By default, the data in the Recode window is sorted by new value in ascending order. This applies to both Recode and Recode Column Names. To change how data is sorted, select **File > Preferences** (Windows) or **JMP > Preferences** (macOS), select **Recode** and change the Sort Style preference. Select **Old Value** to sort column names by the natural order they

appear in the data table and data by the order in the value order column property, if specified. Select **Count** to sort by count in descending order.

- If the data table contains value labels, the labels appear in a column in the Recode window called New Labels. Editing the labels also modifies the Value Labels column property. You can also select **Replace values with value labels** to modify the data in the New Values column.
- The following column properties are updated when you recode a column: Value Labels, Value Scores, Value Order, Value Colors, Supercategories, Coding, and Missing Value Codes.
- If you enter a non-numeric value in a column with a Numeric data type, you are prompted to convert the data type to Character. Click **Yes** to convert the column and display the new value. Click **No** to keep the column Numeric and display a missing value.
- Selecting rows on the Recode window also selects them in the data table for quick editing.

Recode Options

The following options are available in the list next to the Recode red triangle:

New Column Creates a new column for the changed data and retains the original column.

Formula Column Creates a new column with the changes as a formula instead of values. Changing a value in the original column in the data table causes the formula column to update that value automatically.

In Place Applies any change to the original data column.

The remaining options are available on the Recode window:

Filter Searches for specific values. Click the down arrow to select search options. For more information about each option, see "Search Filter Options in Recode". Note that you can type a regular expression to customize the filter.

View Groups Deselect to view ungrouped values. Selected by default.

Show Only Grouped Shows recoded values that have been grouped.

Show Only Ungrouped Shows values that have not been grouped.

Group Becomes active when multiple values are selected. Click **Group** to make highlighted values part of the same group. If you previously edited a value before grouping, the edited value becomes the group representative in the New Value column. Otherwise, the group representative is the value that occurs most often.

All Shows all data values.

Only Modified Shows only modified recoded values.

Only Unmodified Shows only unmodified recoded values.

Changes Shows a list of changes made in the Recode window.

Reverses the last change made to the window.

Recalls the last change made to the window.

Scripting

To save a script to re-create the recoded data table, select Script from the red triangle menu and then select an option.

The Recode window includes the following script options:

Script sequence of actions Attempts to save your actions to a script and omit the data. If you convert all strings to uppercase using the red triangle option, the generated script contains code that converts input strings to uppercase without including the data.

Compress sequence Compresses the script of sequenced actions by removing unnecessary operations. Consider this option to speed up recoding that contains multiple edits.

Multiple Response

Remove empty items In a Multiple Response column, excludes empty items in the column when recoding the column in the data table.

Remove duplicate items In a Multiple Response column, excludes duplicate items. For example, if the original data were a, b, b, c, and no changes to those particular items were made, the recoded column in the data table would contain a, b, c.

Sort items In a Multiple Response column, sorts items alphabetically. For example, if the original data were b, a, c, and no changes to those particular items were made, the recoded column in the data table would contain a, b, c.

Red Triangle Options for Recode

The red triangle menu contains options for the Recode window.

Convert to Titlecase Converts the first letter of each word to uppercase, and the remaining letters to lowercase.

Convert to Uppercase Converts all values to uppercase.

Convert to Lowercase Converts all values to lowercase.

Trim Whitespace Removes leading and trailing whitespace characters. For example, if an extra space was imported before and after the name John, this command would delete the spaces.

Remove Whitespace Removes trailing, leading, and interior whitespace characters.

Collapse Whitespace Removes leading and trailing whitespace characters and removes duplicate interior whitespace characters. That is, if more than one whitespace character is present, the **Collapse Whitespace** command replaces the two spaces with one space.

Remove Punctuation Removes special characters (such as quotation marks and ampersands) from the beginning, middle, and end of words.

Replace String Replaces all occurrences of specified characters with a new string or with nothing. Note that you can type a regular expression to customize the replacement.

First Word Groups values based on the first word of the string. For example, if "John Smith" and "John Adams" were values, this command would group them under "John."

Last Word Groups values based on the last word of the string.

Group Similar Values Enables you to customize how data is grouped. Choose from the grouping options list. See "Group Similar Values Options for Recode".

Advanced

All but First Word Splits values based on the remaining value after the first word is excluded.

All but Last Word Splits values based on the remaining value after the last word is excluded.

Apply Mapping from Table Enables you to use columns in a mapping data table to modify values in the target table. See "Example of Applying Values from Mapping Columns".

Extract Segment Enables you to extract specific strings. See "Example of Splitting on a Comma".

Choose Closest Enables you to map values in the column of a mapping data table to the column that you are recoding.

Parse as Numbers Converts the data to numbers.

Labels to Codes Converts labeled values to codes.

Convert to Character In a numeric column, converts the column to a character column.

Add Value Labels Enables you to assign value labels to each value.

Replace Values with Value Labels Replaces the values specified in the Value Label column property with the value labels that you specify. This option is available only when the column has a Value Label property.

Recall Recalls previous changes made in the Recode window.

Start Over Returns the window to the default condition.

Script View options to import, merge, or save Recode scripts.

- **Import from File** imports a JSL script to recode previously recoded data. Run the same script on different data to recode data the same way.
- **Import from Data Table** imports a JSL script saved to a data table.
- **Import from Recode Result Column** imports data from the column that you select, usually the column in which you saved previous recode results.
- **Save to File** saves Recode changes to a JSL script. After selecting Save, you are prompted to name and save the file.
- **Save to Data Table** saves a Recode script to current data table.

  To modify the recoding in a saved script, right-click the script in the data table and select **Edit with Recode**. After making changes, select **Script > Save to Data Table** from the red triangle menu again to save your changes in a new script.
- **Merge with Data Table Script** merges changes made in the Recode window to the current Recode script saved to the data table. If there are multiple scripts, you are prompted to choose which script to merge your recoded data with.
- **Save to Script Window** appends the Recode script to the script window.

JMP PRO **Publish to Formula Depot** Publishes the recode script to the Formula Depot, which enables you to reuse the code in other scripts.

Right-Click Options for Recode

The first six options are available for grouped values:

Group to New Value (Available only when you select multiple values) Enables you to specify a new value for the two or more selected values.

Group to (Available only when you select multiple values) Right-click selected values to select a different grouping value, or group representative. The list shows the first eight possible group representatives.

Swap New Values When two values are highlighted, select **Swap New Values** to make the new value of the first value adopt the new value of the second value, and vice versa.

Make Representative Right-click a single value from a group and select **Make Representative** to make the selected value the new value.

Remove From Group After values are grouped, right-click one or more values to remove them from that group.

Group Similar Values Right-click a single string to find values that are similar. The Grouping Options window appears. See "Group Similar Values Options for Recode".

Open All Groups Opens the groups so that you can see all of the values in each group.

Close All Groups Collapses the values within each group.

The following options are available for all data:

Select Rows Selects the selected rows in the data table. When you select a row in the Recode window, the corresponding rows are automatically selected in the data table. However, if you select rows in the data table, the rows are not selected in Recode. If you want to select only the rows shown in the Recode window, right-click the row, and select this option.

Make into Data Table Creates a data table of the results shown in the Recode window.

Copy Selected Column Rows Copies the data only from the column that you right-clicked. You can then paste the data into a data table.

Copy Selected Table Rows Copies the data for all columns. You can then paste the data into a data table.

Group Similar Values Options for Recode

Select the following Group Similar Values commands to increase the accuracy of grouping:

Ignore Case Item case is ignored.

Ignore Non-Printable Characters Non-printable characters are ignored. Some data can include non-printable characters (such as file separators) that only the computer can read.

Ignore Whitespace White space is ignored.

Ignore Punctuation Punctuation is ignored.

Allow Character Edits Enables characters to be replaced by the new value when similar values are grouped.

Max Difference Ratio Groups values according to the maximum proportional difference. For example, type ".25" to group values that are at most 25% different.

Max Character Difference Groups values according to a maximum number of nonadjacent character differences. For example, type "5" to group values that differ by five characters or less.

Min Source String Length Prevents strings from being matched if they are too short. Experiment with this option and the Difference Ratio and Max Character Difference options to specify a value that works for your data.

Search Filter Options in Recode

Click the down arrow button next to the search box to refine your search.

Contains Terms Returns items that contain a part of the search criteria. A search for "ease oom" returns messages such as "Release Zoom".

Contains Phrase Returns items that contain the exact search criteria. A search for "text box" returns entries that contain "text" followed directly by "box" (for example, "Context Box" and "Text Box").

Starts With Phrase Returns items that start with the search criteria.

Ends With Phrase Returns items that end with the search criteria.

Whole Phrase Returns items that consist of the entire string. A search for "text box" returns entries that contain only "text box".

Regular Expression Enables you to use the wildcard (*) and period (.) in the search box. Searching for "get.*name" looks for items that contain "get" followed by one or more words. It returns "Get Color Theme Names", "Get Name Info", and "Get Effect Names", and so on.

Invert Result Returns items that do not match the search criteria.

Match All Terms Returns items that contain both strings. A search for "t test" returns elements that contain either or both of the search strings: "Pat Test", "Shortest Edit Script" and "Paired t test".

Ignore Case Ignores the case in the search criteria.

Match Whole Words Returns items that contain each word in the string based on the Match All Terms setting. If you search for "data filter", and Match All Terms is selected, entries that contain both "data" and "filter" are returned.

Example of Grouping by Similar Values

You can group similar values according to the number of characters that differ between them.

1. Select **Help > Sample Data Library** and open Candy Bars.jmp.
2. Select the Name column.
3. Select **Cols > Recode**.
4. Click the red triangle and select **Group Similar Values**.
5. Make sure that Allow Character Edits is selected.
6. Next to Max Different Ratio, type ".5".

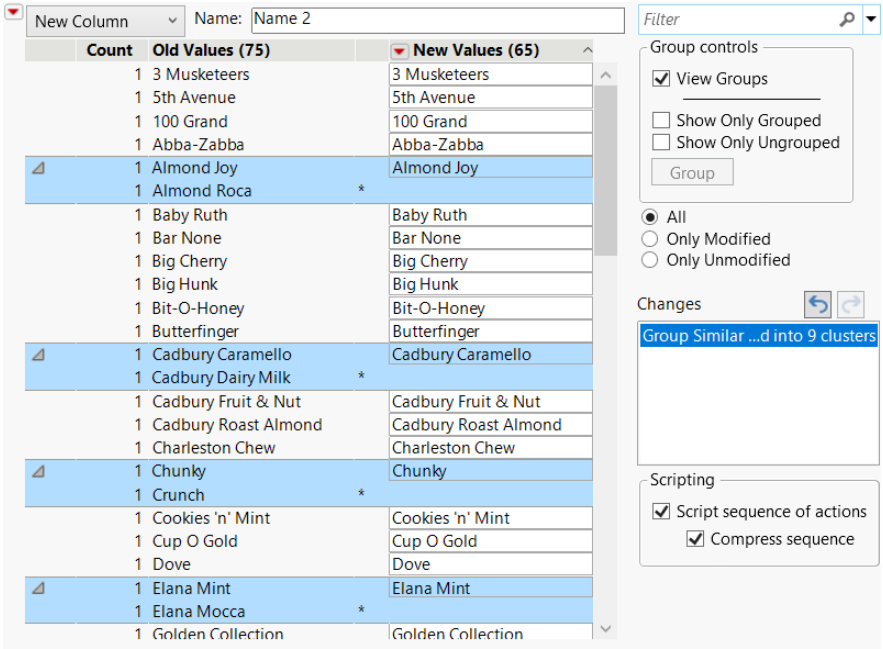
This allows JMP to group values that differ by 50% or less. In other words, values that share at least 50%, or half, of the same characters. The Difference Ratio is determined by comparing the total number of characters of each value and the total amount of unique characters between two given values.

7. Next to Max Character Difference, type “6”.

   Values that have no more than six characters are grouped.

8. Click **OK**.

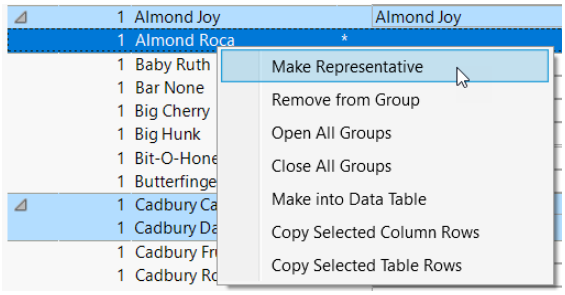
Figure 4.44 Grouping Similar Values



The values shown in the New Value column represents the grouped values in the recoded data table.

9. (Optional) To change the new value to represent a different old value in a group, right-click Almond Roca and select **Make Representative.**

Figure 4.45 Make Representative



10. (Optional) To replace the original data in the data table with the recoded data, select **In Place** from the red triangle menu, and then click **Recode.**

Example of Splitting on a Comma

The Aircraft Incidents.jmp sample data table contains a column of city and state names separated by a comma. You can split the values in that column on the comma and recode the column with only state names.

1. Select **Help > Sample Data Library** and open Aircraft Incidents.jmp.
2. Select the Location column and select **Cols > Recode**.

Figure 4.46 Original Data

Old Values (1537)	New Values (1537)
Abilene, TX	Abilene, TX
Adelanto, CA	Adelanto, CA
Afton, VA	Afton, VA
Aieia, HI	Aieia, HI
Aiken, TX	Aiken, TX
Ainsworth, NE	Ainsworth, NE
Akiachak, AK	Akiachak, AK
Akron, CO	Akron, CO
Alamogordo, NM	Alamogordo, NM
Alamosa, CO	Alamosa, CO
Albion, IL	Albion, IL
Albuquerque, NM	Albuquerque, NM
Alcova, WY	Alcova, WY
Alexander City, AL	Alexander City, AL
ALEXANDER, AK	ALEXANDER, AK
Alice, ND	Alice, ND
Allamuchy, NJ	Allamuchy, NJ
Almond, WI	Almond, WI
Alton, IL	Alton, IL
Alturas, CA	Alturas, CA
Altus, OK	Altus, OK
Alvarado, TX	Alvarado, TX
Amargosa Valley, NV	Amargosa Valley, NV
Amarillo, TX	Amarillo, TX
AMHERST, NH	AMHERST, NH

3. Click the red triangle and select **Advanced > Extract Segment**.

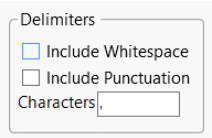
In the Preview pane, notice that the data contain cities followed by a comma and abbreviated state names. You will extract the last two characters of each string.

4. Deselect **Include Whitespace** and **Include Punctuation** at the top of the window.

   Whitespace and punctuation will not be considered in the extraction.

5. Type a comma in the Characters box.

Figure 4.47 Specifying a Delimiter



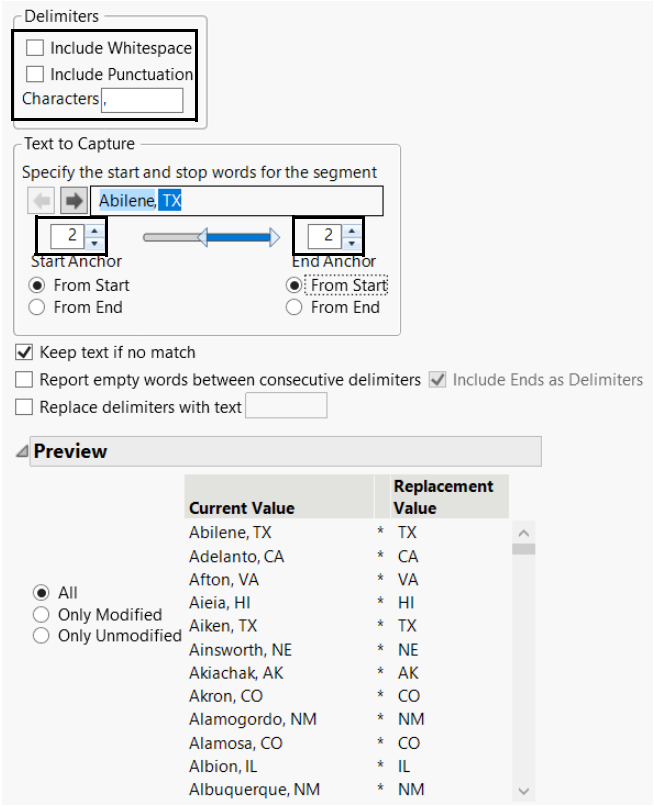
The comma is used to extract the last two characters.

6. In the Extract Segment window, type “2” above Start Anchor and End Anchor.

   The segment extraction begins and ends with the second word.

7. Make sure that From Start is selected for the Start Anchor and End Anchor options.

Figure 4.48 Extract Segment Window



Tip: In the Text to Capture section, click the left or right arrow to scroll through the values. This enables you to preview your settings.

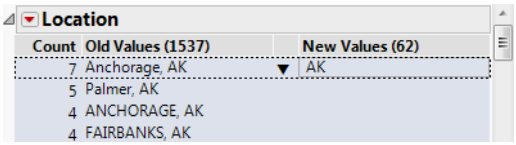
8. In the Preview pane shown in Figure 4.48, notice that the replacement values contain only the state abbreviations.
9. Scroll down in the Preview pane. Notice that the current value and replacement value for Atlantic Ocean are the same. The data couldn't be split on a comma.

Figure 4.49 Data with No State

10. Click **OK**.

    Entries with a matching state name are grouped.

Figure 4.50 Recoded City, State Name



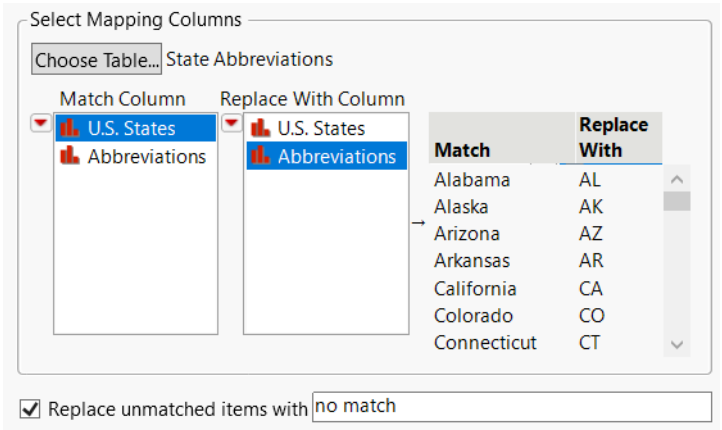
Example of Applying Values from Mapping Columns

A column in a mapping data table contain values that you can use to recode a column in another data table. In this example, U.S. states in State Capitals.jmp are replaced with the state abbreviations defined in State Abbreviations.jmp (the mapping data table).

1. Select **Help > Sample Data Library** and open State Abbreviations.jmp and State Capitals.jmp.
2. In State Capitals.jmp, select the U.S. States column.
3. Select **Cols > Recode**.
4. Click the red triangle and select **Advanced > Apply Mapping from Table**.
5. Click **Choose Table**, select State Abbreviations.jmp, and click **OK**.
6. In the Match Column box, select **U.S. States**.
7. In the Replace With Column box, select **Abbreviations**.

   The Match and Replace With columns show the result of the mapping.

Figure 4.51 Matching and Replacing Columns



8. Select the **Replace unmatched items with** box and type “no match”.

   The Preview section shows how the values are mapped. Scroll down and notice that the “no match” states are misspelled, so they don’t map to any state in the Abbreviations column.

Figure 4.52 Preview of Unmatched Columns

Current Value		Replacement Value
Kentucky	*	KY
Louisiana	*	LA
Maine	*	ME
Maryland	*	MD
Massachusets	*	no match
Michigan	*	MI
Minnesota	*	MN
Missisippi	*	no match

9. Click **OK** and then click **Recode**.

   A new column of abbreviated state names appears in State Capitals.jmp in a new column called U.S. States 2.

Figure 4.53 Abbreviated State Names

U.S. States	U.S. States 2
Alabama	AL
Alaska	AK
Arizona	AZ
Arkansas	AR
California	CA
Colorado	CO
Connecticut	CT
Delaware	DE
Florida	FL
Georgia	GA
Hawaii	HI
Idaho	ID
Illinois	IL
Indiana	IN
Iowa	IA
Kansas	KS
Kentucky	KY
Louisiana	LA
Maine	ME
Maryland	MD
Massachusets	no match
Michigan	MI
Minnesota	MN
Missisippi	no match
Missouri	MO

To fix the entries that weren't matched, see "Example of Fixing Misspelled Words".

Example of Fixing Misspelled Words

You can use a data table as a dictionary to fix misspelled words. State Abbreviations.jmp contains columns of U.S. state names and abbreviations. State Capitals.jmp contains a few misspelled state names to mimic real data. This example shows how to fix the misspelled state names.

1. Select **Help > Sample Data Library** and open State Abbreviations.jmp and State Capitals.jmp.
2. In State Capitals.jmp, select the U.S. States column.
3. Select **Cols > Recode**.
4. Click the red triangle and select **Advanced > Choose Closest**.
5. Click **Choose Table**, select State Abbreviations, and click **OK**.
6. In the Match Column box, select **U.S. States**.
7. In the Preview pane, scroll down and notice that "Massachusets" will be replaced with "Massachusetts", "Missisippi" will be replaced with "Mississippi".

Figure 4.54 Misspelled State Replacement

Current Value		Replacement Value
Kentucky		Kentucky
Louisiana		Louisiana
Maine		Maine
Maryland		Maryland
Massachusets	*	Massachusetts
Michigan		Michigan
Minnesota		Minnesota
Missisippi	*	Mississippi

8. Click **OK** and then click **Recode**.

   A new column that contains the updated state names is added to the data table.

Example of Parsing as Numbers

The Parse as Numbers option turns character data into numeric data. This option is comparable to the JSL `Num()` character function.

Many formats are supported, such as date and times, currency, and scientific notations. If a format is unsupported, a missing value is displayed.

1. Open your data table and select a column that contains both numeric and character values.

Figure 4.55 Column of Numeric and Character Data

	X
1	a
2	4,033.67
3	$8000.56
4	5/4/2019
5	10/21/2017 5:10:33
6	12Jan2018
7	1.07E-6

2. Select **Cols > Recode**.
3. Select **Parse as Numbers** from the red triangle menu.

   The old and new values are shown.

Figure 4.56 Preview of New Values

Count	Old Values (7)		New Values (7)
1	a	*	.
1	1.07E-6	*	1.07e-6
1	4,033.67	*	4033.67
1	$8000.56	*	8000.56
1	10/21/2017 5:10:33	*	3.591e+9
1	12Jan2018	*	3.599e+9
1	5/4/2019	*	3.64e+9

4. Click **Recode**.

   A new column with recoded values is added to the data table. The data type of the column is Numeric.

   "a" couldn't be parsed as a number, so a missing value is displayed in the new column for that value.

Figure 4.57 Parsed Numbers

	X	X 2
1	a	•
2	4,033.67	4033.67
3	$8000.56	8000.56
4	5/4/2019	3.6398e+9
5	10/21/2017 5:10:33	3.5914e+9
6	12Jan2018	3.5986e+9
7	1.07E-6	1.07e-6

Example of Applying Recoded Values to a Different Column

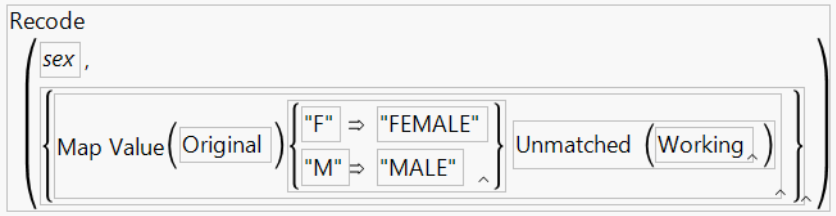
When you recode a column, you can save the recoded values in a script. When you import or run the script, the column that you originally recoded is recoded again.

Suppose that you want to recode a different column with the same values. Save the recoded values in a formula column and then edit the formula.

1. Enter the recoded values in the New Values column of the Recode window.
2. Select **Formula Column** from the list next to the red triangle menu.
3. Click **Recode**.
4. Right-click the formula column and select **Formula**.
5. In the formula, select the column that was recoded.

   In this example, the sex column was recoded.

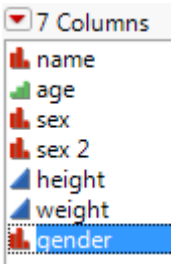
Figure 4.58 Recoded Column



6. In the Columns list, select the column that will replace sex.

   In this example, you want to replace the sex column with gender, so select gender.

Figure 4.59 Selecting the gender Column



Edit Data Tables

JMP provides many ways to change the basic properties of a data table.

- "Rename Data Tables"
- "Lock Data Tables"
- "Compress Data Tables"
- "Copy Table Scripts"
- "Use Data Table Variables"
- "Create and Save Scripts in Data Tables"

Rename Data Tables

A data table's name appears at the top of its window, in the table panel, and on all related analysis reports. You can change a data table's name in any of the following ways:

- Select **File > Save As** and save as the new name.
- In the table panel, click table name, type the new name, and then press Enter.
- On Windows, select **Window > Set Title**.

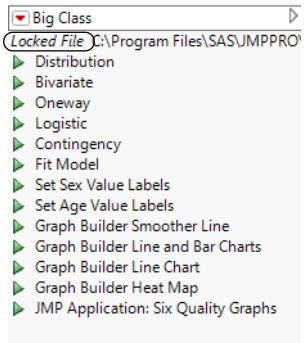
Lock Data Tables

Locking a data table prevents data and column properties from being added or edited. You can still assign row states, run analyses, and so on. To lock a data table, click the red triangle next to the table name in the table panel and select **Lock Data Table**.

A lock icon appears next to the data table name. To unlock the file, select **Lock Data Table** again.

If you make a data table read-only outside of JMP (for example, by changing its properties on Windows), the data table contains a note informing you that it is locked (Figure 4.60). This type of lock allows users to edit the data table, but not save the changes.

Figure 4.60 A Read-Only File



Compress Data Tables

Compressing a data table reduces the size of the stored file. You can still run analyses, assign characteristics, and so on. To compress a data table, click the red triangle next to the table name in the table panel and select **Compress File When Saved** and save the data table.

After saving the data table, a compressed icon appears next to the data table name. To decompress the file, select **Compress File When Saved** again.

In addition, you can configure JMP to always use GZ compression when saving by selecting **Preferences > General > Save Data Table Columns GZ Compressed**.

Note: The **Compress File When Saved** option only decreases the file size. This command does not affect the memory required to analyze the data. To reduce both the file size and memory required for analyzing, use **Cols > Utilities > Compress Selected Columns**. See "Compress Selected Columns in Data Tables".

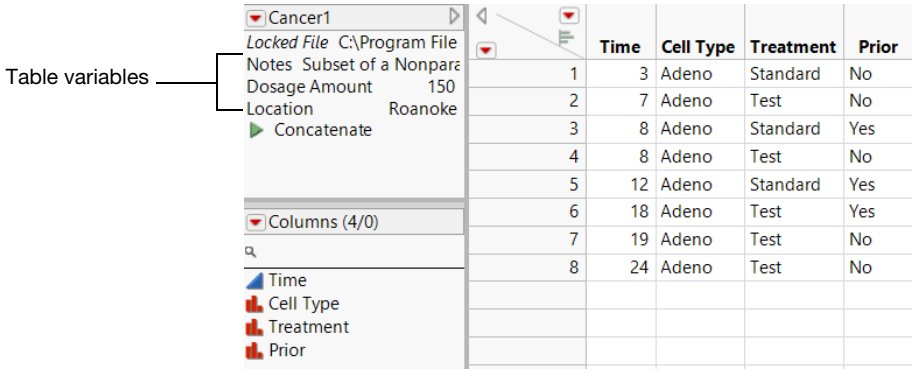
Copy Table Scripts

Copying a table script enables you to paste the script into another script to recreate the table

Use Data Table Variables

A data table variable can contain textual information (for example, source information for the data), or a value that can be used by column formulas or JSL scripts. Data table variable names appear in the table panel at the left of the data grid.

Figure 4.61 Table Variables in the Table Panel



	Time	Cell Type	Treatment	Prior
1	3	Adeno	Standard	No
2	7	Adeno	Test	No
3	8	Adeno	Standard	Yes
4	8	Adeno	Test	No
5	12	Adeno	Standard	Yes
6	18	Adeno	Test	Yes
7	19	Adeno	Test	No
8	24	Adeno	Test	No

Uses for Table Variables

Use table variables in the following situations:

- To document tables
- In formulas
- In JSL scripts

Use Table Variables to Document Tables

Table variables are used primarily to document tables. Many sample data tables installed with JMP contain a table variable named Notes. This variable provides details about the data (for example, the source of the data). The example in Figure 4.61 shows a data table that contains Notes as one of its table variables. JMP also automatically creates table variables when you create a design table using the Design of Experiments commands in JMP. The design table has a table variable named Design with the name of the design type as its value.

Reference Table Variables in Formulas

Table variables can also be incorporated in formulas that you build using the Formula Editor. These formulas calculate values for a column by referring to a table variable. For more information about constructing a formula that uses table variables, see “Refer to Data Table Values in Formulas”.

Use Data Table Variables in JSL Scripts

You can also incorporate table variables into JSL scripts. See the *Scripting Guide*.

Data Table Variable Actions

To add new data table variables

1. In the Table panel, click the red triangle to the left of the data table name.
2. Select **New Table Variable**.
3. Give the variable a name and value in the boxes labeled **Name** and **Value**.
4. Click **OK**.

   The table variable appears in the Table panel.

To view or edit table variables

1. Double-click the content of an existing table variable.
2. Edit the content.

To edit a table variable name

1. Double-click the table variable name.
2. Edit the name.

To delete table variables

Select one or more table variables and press Delete, or right-click the selected variables and select Delete.

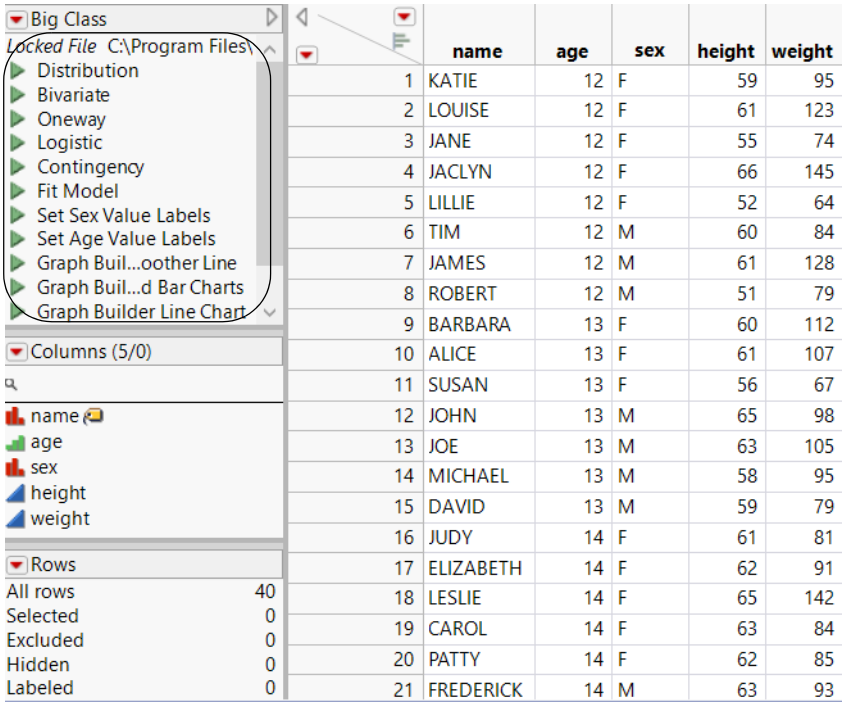
Concatenating Data Tables with Table Variables

See the "Example of Concatenating Data Tables and Table Variables".

Create and Save Scripts in Data Tables

To automatically complete various analyses and tasks, you can create a JSL script and save it to the data table (Figure 4.62). For more information about writing data table scripts, see the *Scripting Guide*.

Figure 4.62 Scripts Saved With the Data Table



	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107
11	SUSAN	13	F	56	67
12	JOHN	13	M	65	98
13	JOE	13	M	63	105
14	MICHAEL	13	M	58	95
15	DAVID	13	M	59	79
16	JUDY	14	F	61	81
17	ELIZABETH	14	F	62	91
18	LESLIE	14	F	65	142
19	CAROL	14	F	63	84
20	PATTY	14	F	62	85
21	FREDERICK	14	M	63	93

Save a Report Script to a Data Table

Once you have run an analysis and you are in the report window, you can add a script to the data table. This script generates the JSL that reproduces your analysis.

To save a script to the data table

From the report window, click the red triangle for the platform and select **Save Script > To Data Table**.

Example of Saving a Report Script to a Data Table

First, you create your analysis, then you save the script.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response**.
4. Select height and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Bivariate Fit of weight By height and select **Fit Line**.

7. Click the red triangle next to Bivariate Fit of weight By height and select **Save Script > To Data Table**.
8. In the Save Script As window, enter the name of the script.
9. To replace an existing script with the same name, select it from the Name list and select **Replace existing script**.
10. To save a script that has the same name as an existing script and use the same name, select **Append unique suffix**.

    A number is added to the end of the script name.
11. Click **OK**.

The script is added to the bottom of the Table panel.

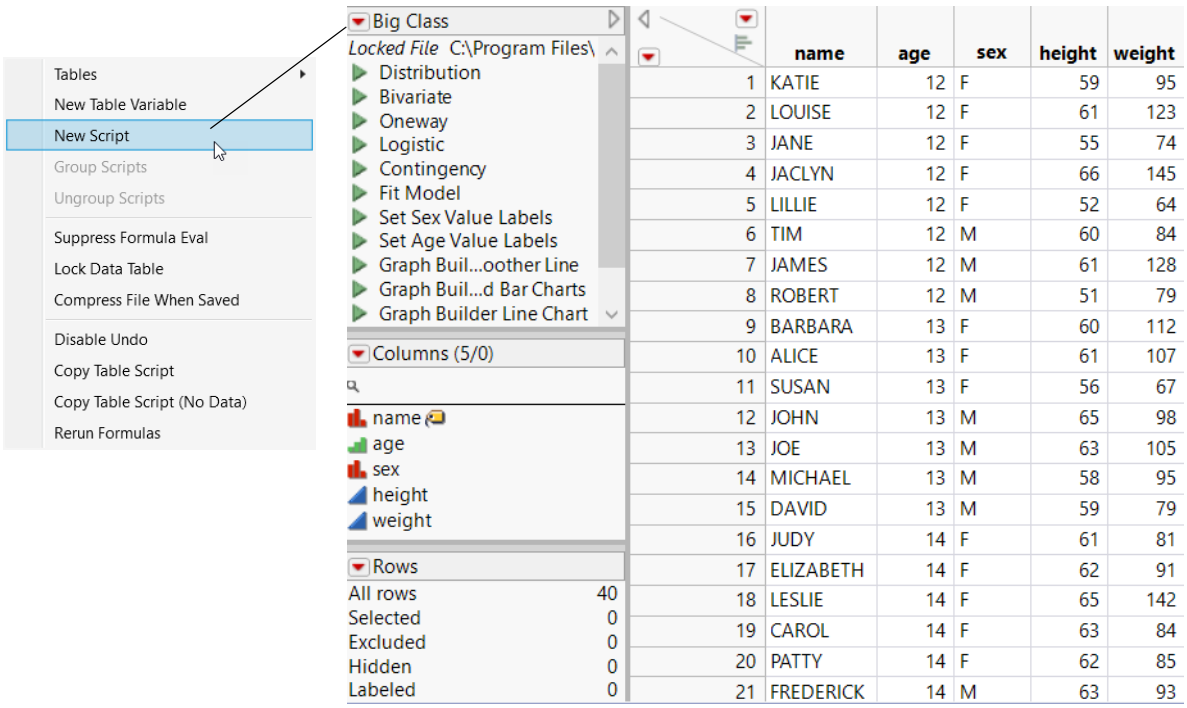
Tip: If you want a particular script to run automatically every time the data table is opened, name the script OnOpen. Only one script saved in the data table can be set to run automatically. If you name the script Model (or model) in a Fit Model script, the launch window is automatically filled in based on the script when you select **Analyze > Fit Model**.

Write a JSL Script for the Data Table

To add a script to a data table using JSL

1. Click the red triangle to the left of the data table name in the Table panel.

Figure 4.63 Creating a Script



2. Select **New Script**.
3. Give the script a name by typing it into the box beside **Name**.
4. Add the script by entering JSL code into the box beside **Script**.
5. Perform one of the following actions:
   - If you want to run the JSL Debugger on the script to check it for errors, click **Debug Script**.
   - If you are finished editing the script, click **OK**. The script appears in the Table panel and the window closes.
   - If you are not finished editing the script and want to save it, click **Save**. The script appears in the Table panel and the window remains open for further editing.
   - If you want to run the script, click **Run**.

Run, Edit, Delete, or Copy Scripts

To run, edit, delete, or copy a script that is saved to the data table

1. In the Table panel, right-click the green triangle beside the script's name or right-click the script name.

2. Select one of the following commands:
   - Run Script
   - Edit
   - Delete
   - Copy

Once you copy a script, you can then paste it into a script window or into the Table panel of another data table.

Chapter 5

Set JMP Column Properties

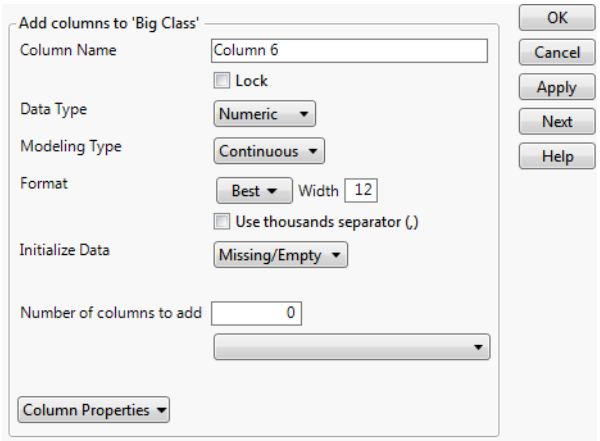
The Column Info Window

Use the Column Info window to set specific properties on a column in a data table. Here are some examples of the actions that you can perform on a column:

- Change data and modeling types
- Change numeric formats
- Add formulas
- Specify restrictions on values or missing values
- Order categorical values or row data
- Save specification, control, or response limits
- Enter a known value for sigma

You can also standardize attributes and properties across multiple columns, assign a preselected analysis role to columns, and compress columns in a data table.

Figure 5.1 The Column Info Window



Contents

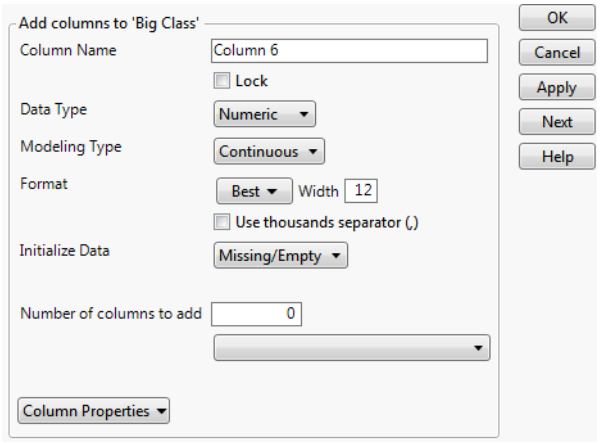
About the Column Info Window . . . 291
About Modeling Types . . . 294
The Short-Integer Format . . . 294
Numeric Formats . . . 295
Store Information in Row State Columns . . . 300
Initialize Data in a Column . . . 302
Column Properties in JMP . . . 304
Basic Column Properties . . . 305
Properties That Validate Column Values . . . 305
Properties That Attach Information to Column Values . . . 307
Properties That Control the Display of Columns . . . 312
Properties Used in Modeling and DOE . . . 313
Properties Associated with Control Charts and Capability . . . 314
Properties That Control How Columns Are Used in Platforms . . . 316
SAS Properties . . . 324
Link Data Tables with Virtual Join Properties . . . 324
Additional Column Properties . . . 326
Properties Assigned and Controlled by JMP . . . 329
Standardize Attributes and Properties Across Columns . . . 330
Recode Column Values . . . 331
Standardize Column Attributes . . . 331
Standardize Column Properties . . . 332
Delete Column Properties . . . 332
Example of Standardizing a Formula . . . 332
Assign a Preselected Analysis Role to a Column . . . 333

About the Column Info Window

Use the Column Info window to specify all of the attributes and properties of a column. You can access the Column Info window in any one of the following ways:

- Select **Cols > New Columns.**
- Click an existing column heading and select **Cols > Column Info**.
- Right-click an existing column heading and select **Column Info**.
- Double-click directly above a column name.
- In the Columns panel, right-click a column and select **Column Info**.

Figure 5.2 The Column Info Window



The Column Info window contains the following information:

Column Name Type or edit the name of the column.

Lock Lock the column so that none of its values can be edited. After you lock a column, the lock icon ( ) appears next to the column name in the data table's Columns panel. If you add a formula to a column, the column automatically locks.

Data Type Select or change the data type of a column, which determines the following:

- How the column's values are formatted in the data grid
- How the column's values are saved internally
- Whether the column's values can be used in calculations

See "About Modeling Types".

Choose from the following data types:

Numeric The columns contain only numbers (with or without a decimal point).

Character The columns contain any characters, including numbers. In character columns, numbers are seen as characters and are treated as discrete values instead of continuous values. The maximum field width for character values is 32,766 bytes.

Row State The columns contain row state information, which indicates whether the rows are excluded, hidden, labeled, colored, or marked. See "Store Information in Row State Columns".

Expression The columns contain JSL expressions that can range from simple entries to complicated statement sequences. In particular, an expression can correspond to a picture, a matrix, or an expression. You can drag-and-drop expressions, such as an image, from your desktop to cells in an expression column. See "Expression Role".

Notes:

- If you change a column's Data Type from Character to Numeric, any character values become missing data values and are not recoverable.
- If you enter non-numeric data in a column with a Numeric data type, a warning window is displayed, prompting you to change the data type to Character. Click **Try Again** to return to the data table to re-enter the value. Click **Change** to change the data type. Click **Revert** to cancel your edit and return the cell's value.
- Short-integer formats might also be available. See "The Short-Integer Format".

Modeling Type (Numeric or Character data types only) Select or change the modeling type of a column, which tells JMP how to treat the column's values during analyses. You can change the modeling type to look at a variable in different ways. See "About Modeling Types".

Choose from the following modeling types:

Continuous Only numeric data types. Continuous values are treated as continuous measurement values. JMP platforms use the numeric values directly in computations.

Ordinal Either numeric or character data types. JMP platforms treat ordinal values as discrete categorical values that have an order. In Graph Builder, if there are more than 50 discrete values, JMP treats the column as continuous.

- If the column has a numeric data type, the values are ordered according to their numeric magnitude.
- If the column has a character data type and a Value Order column property is saved to the column, values are ordered according to the Value Order property. Otherwise, values are ordered according to their alphanumeric data value ordering. However, in

special cases where the values have an obvious order, values are automatically ordered appropriately. See "Value Order".

Nominal Either numeric or character data types. JMP platforms treat all values as discrete values with no implicit order.

Multiple Response Only character data types. Distinct entries in a single cell must be separated by commas. JMP platforms that support multiple response columns treat each entry in the comma-separated list as a separate data value. If your entries are separated by a character other than a comma, use the Multiple Response column property instead. See "Multiple Response".

Unstructured Text Only character data types. These values are generally unique and are therefore not appropriate for categorical analysis. The Text Explorer platform is ideal for analyzing unstructured text values.

Vector Only expression data types. The entries in the cells are column or row vectors. JMP platforms that support the Vector modeling type recognize the vectors and treat them appropriately in calculations.

None Any data type. Use None as a modeling type when a column is not well represented by the other modeling types. For example, a column of pictures or ID values might be assigned the None modeling type. JMP launch windows do not allow you to assign roles to columns with the None modeling type.

Tip: You must use the Column Info window to change a column's modeling type to Multiple Response, Unstructured Text, or Vector.

Format (Numeric data types only) Select or change the display format of a numeric column. See "Numeric Formats".

Initialize Data (Appears only during new column creation) Specify the type of initial data values that you want to appear in the column. See "Initialize Data in a Column".

Column Properties (Contains a list of different properties) Assign properties to columns. See "Column Properties in JMP".

Tip: Click the **Next** button to continue adding columns.

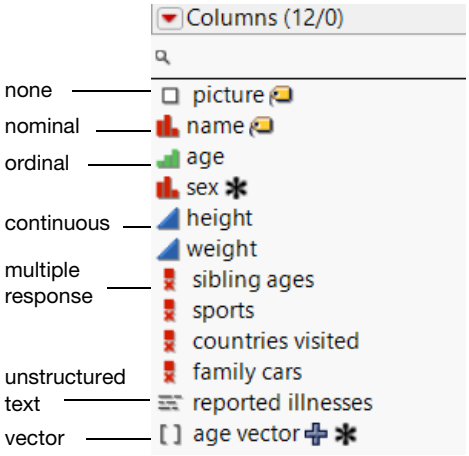
About Modeling Types

A column in a JMP data table can contain different types of information. However, all information in a single column must have the same data and modeling types.

- When you import data, JMP guesses which data and modeling types to use. Therefore, you should verify that JMP has guessed correctly.
- When you manually insert data, you should assign a data type and a modeling type at that time.

Figure 5.3 illustrates the icons that identify the different modeling types.

Figure 5.3 Modeling Type Icons in the Columns Panel



Click an icon to change the modeling type to **Continuous**, **Nominal**, **Ordinal**, or **None**.

See "Modeling Type".

Tip: You can select Continuous only if your data type is numeric. If the Continuous option is dimmed on the menu and you want to make the column continuous, you must first change the column's data type in the Column Info window.

The Short-Integer Format

When you use the correct short-integer format for your data, you do not see any difference in how the numbers appear. However, the numbers occupy less disk space and use less memory. Short-integer formats must be activated in preferences to appear in the Column Info window.

To make short-integer formats available in the Column Info window

1. Select **File > Preferences** and click **Tables**.
2. Select the **Allow short numeric data format** option.
3. Click **OK** to return to the data table.

To store numeric data in short-integer format:

1. Double-click above the column name whose values you want to be short-integer.

   The Column Info window appears.
2. From the Data Type menu, select **1-byte integer**, **2-byte integer**, or **4-byte integer**.

JMP stores values as integers in the range that you selected. The following numbers are examples:

- For 1-byte integer, the range of numbers that you can enter is from -126 to 127.
- For 2-byte integer, the range of numbers that you can enter is from -32,766 to 32,767.
- For 4-byte integer, the range of numbers that you can enter is from -2,147,483,646 to 2,147,483,647.

Numeric Formats

For numeric columns, the Format menu appears in the Column Info window. Specify the format to tell JMP how to display numbers in the column. For all format options, you can specify the number of total characters that you want the cells in the column to accommodate. See "Specify Width".

For descriptions of the format options, see "Numeric Format Options".

Tip: To add commas to values that equal a thousand or more, select the **Use thousands separator** option. You must account a space for each comma in the Width box, or else they might not appear. This option is available for the Best, Fixed Dec, Percent, and Currency formats.

Specify Width

When you specify a number in the Width field, be sure to include the total number of possible characters. Characters include: numbers, decimal points, commas, and currency symbols.

Numeric Format Options

Choose from the following numeric format options:

Best Allows JMP to consider the precision of each cell value and select the best way to show it. By default, the physical width of the column is 12 characters.

Fixed Dec Shows all values in the column rounded to the number of decimal places that you specify.

- To see only whole numbers, set the number of decimal places to zero.
- If the number of integers following the decimal point is smaller than the number of decimal places that you specify, zeros are added to reach the number of decimal places. For example, if the value is 1.23 and you type **5** in the Dec box, JMP shows the number with five decimal places: 1.23000.

Percent Multiplies numeric values by 100 and shows the number followed by a percent sign.

PValue Shows probability values. The default value of the width is 12. If a number is less than 0.0001, the number is displayed as <.0001. The format is mostly used in JSL scripts and rarely needed for a data table column.

Scientific Shows a number in standard scientific notation. If you enter the number 123456, it appears as 1.23456e+5. Select Dec to show the decimal points and enter the number of points.

Engineering Similar to Scientific, but the exponent is always a multiple of 3 and the Dec field represents significant digits.

Engineering SI Same as Engineering, but the exponent is replaced with an SI symbol.

Precision Rounds the number to a given number of significant digits. Specify the number of significant digits in the Dec field.

Currency Formats values with two decimal positions, thousands separators, and the currency sign that is specified in your computer's locale settings. The default width of the **Currency** format is 15. If you have a number that requires a wider field width, the format defaults to the **Best** format. Once assigned, the currency symbol appears in the column and in graphs that contain the column.

Date Shows all values in the column as a date. See "Date Formats".

Time Shows all values in the column as a specific instance in time, such as 12/2/03 at 2:23 PM. See "Time Formats".

Duration Shows all values in the column as a duration of time, such as hours, minutes, and seconds.

:day:hr:m, :day:hr:m:s Shows a duration of time, such as 52:03:01:30, or fifty-two days, three hours, one minute, and thirty seconds.

hr:m, hr:m:s, min:s Shows a duration of time, such as 17:37, or seventeen hours and thirty-seven minutes.

Geographic Shows latitude and longitude number formatting for geographic maps. Latitude and longitude options include the following:

DDD Degrees

DMM Degrees and minutes

DMS Degrees, minutes, and seconds

In each format, the last field can have a fraction part. You can specify the direction with either a signed degree field or a direction suffix. To show a signed degree field, such as *-59°00′00″*, deselect **Direction Indicator**. To show the direction suffix, such as *59°00′00″ S*, select **Direction Indicator**.

To use spaces as field separators, deselect **Field Punctuation**. To use degrees, minutes, and seconds symbols, select **Field Punctuation**.

Custom Enables you to define a custom format for a numeric column, including customization of column width and number of decimal places. Select **Custom**, click **Set Custom Format**, and define the format in the Formula Editor window. For example, if you don't want to display the percentage sign next to a number, select this option and multiply the number by 100. This is an alternative to selecting the Percent format. Or if your data is in meters and you want to show it in kilometers with the unit, add the formula `Char(:value / 1000) || " km"`. You might also want to apply a date/time format that isn't available in the Format Date or Time lists, such as D/M/Y.

Format Pattern Enables you to customize the date-time formatting of numeric values. See "Customize Date-Time Formats with Format Patterns" for details.

Date Formats

When you choose a Date format, you can also specify an Input Format. The Date format indicates how the date appears in the data table cells, and the Input Format indicates how you enter the date.

If you assign a date format to a numeric column that already contains data, then the numeric values are treated as the number of seconds since *January 1, 1904*. For example, if you have a numeric column with a cell value of 1,234,567,890 and you change the format to **Date > m/d/y**, the cell value appears as 02/13/1943.

The examples in Table 5.1 use the date of December 31, 2004.

Table 5.1 Date Formats

Format	Appears As
m/d/y	12/31/2004
mmddyyyy	12312004
m/y	12/2004
yyyyQq	2004Q4
d/m/y	31/12/2004
ddmmyyyy	31122004
ddMonyyyy	31Dec2004
Monddyyyy	Dec312004
y/m/d	2004/12/31
yyyymmdd	20041231
yyyy-mm-dd	2004-12-31
Date Long	Friday, December 31, 2004
Date Abbrev	Dec 31, 2004
Locale Date	Varies based on local OS setting. Here is an example: in the United States, the local OS setting is mm/dd/yyyy (12/31/2004).

Note: To change how a date appears in a graph without changing how it appears in a data table, see "Change the Numeric Format of an Axis".

Time Formats

When you choose a Time format, you can also specify an Input Format. The Time format indicates how the time appears in the data table cells, and the Input Format indicates how you enter the time.

- You can add the number of hours, minutes, and seconds after midnight of the prepended date for the following date formats:
  - m/d/y
  - d/m/y

  - y/m/d
  - ddMonyyyy
  - Monddyyyy
  - Locale Date

  For example, December 31, 2004 has a numeric value of 3,187,296,600, which represents 12/31/2004 12:10 AM.

- **:day:hr:m** and **:day:hr:m:s** show the number of days, hours, minutes, and seconds since *January 1, 1904*. For example, the results for December 31, 2004 are :36890:00:10: and :36890:00:10:00.
- **h:m:s** and **h:m** show the hours, minutes, and seconds portion of the date in the date field. For example, the results for December 31, 2004 at 12:10 AM are 12:10:00 AM and 12:10 AM.
- **yyyy-mm-ddThh:mm** and **yyyy-mm-ddThh:mm:ss** show the year, month, day, and time. For example, 2004-12-31T12:10:00. T is a literal value, representing itself.

Note: To change how a time appears in a graph without changing how it appears in a data table, see "Change the Numeric Format of an Axis".

Customize Date-Time Formats with Format Patterns

Format patterns are strings that define a date-time format, such as <YYYY></><MM></><DD> <hh><:><mm><:><ss><ampm>. The parts of the pattern in angle brackets are called *field descriptors*. The field descriptors represent a value (such as <YYYY>, which is a four-digit year) or other date-time text (such as </>, which is a locale-specific date separator).

A format pattern enables you to build formats that aren't provided in JMP. These formats can be used for both formatting and inputting data. You can type the pattern yourself, use a built-in format as a starting place, or select field descriptors from a list.

You can use format patterns anywhere built-in date-time formats can be selected.

Build a Format Pattern

1. Create a numeric column in a data table.
2. Right-click the column and select **Column Info**.
3. Select **Format Pattern** from the Format list.
4. Click **Set Format Pattern**.
5. To use a built-in format as a starting place, select **Builtins**, select a date-time format, and then edit the format.
6. To edit the format, type the field descriptors in the Format Pattern field, or select a format pattern from the Field Descriptors list.

7. Click **OK**.

Build an Input Format Pattern

1. In an existing numeric column, right-click and select **Column Info**.
2. Select **Format Pattern** from the Format list, and then deselect **Use Output Format**.
3. Click **Set Format Pattern**.
4. Build the pattern as described in the preceding task.

Notes:

- Some formats cannot be used for input. For example, when a variable-length number field descriptor is followed immediately by another number, some dates are ambiguous. The format <YYYY><M><D> cannot process input because <M> is a single digit month; dates such as 2020111 would be ambiguous. It's unclear whether the date is Jan 11 or Nov 1.
- ISO date formats were previously parsed as non-ISO YMD formats, and the opposite was also true. Now YMD date formats are compatible with each other; you can paste one YMD date into a column that is formatted using another YMD date.

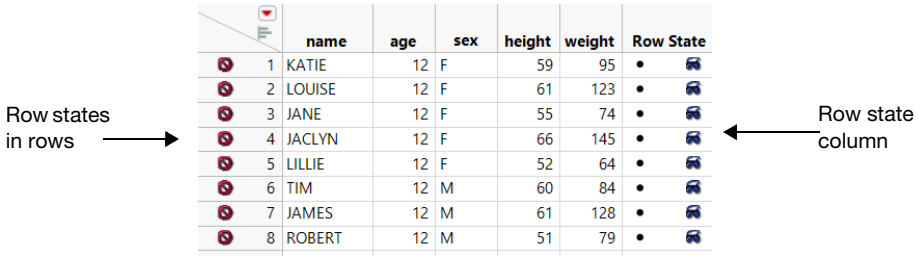
International Formats

If you are importing or entering data that contains formatting specific to country standards, you might need to make sure that your number formats are interpreted correctly. On Windows, access the Control Panel's region and language option, and select the country for which the number should be formatted. On macOS, from the Apple menu, select **Language & Region > Formats**, and select the correct country. On later versions of macOS, this option may appear under System Preferences > Language and Region.

Store Information in Row State Columns

Similar to assigning row states to rows, you can create a column that contains only row state information. A row state column stores information about whether rows are excluded, hidden, labeled, colored, marked, or selected. To designate a column as a row state column, in the Column Info window next to Data Type, select **Row State**.

Figure 5.4 Row States in Rows and a Row State Column



	name	age	sex	height	weight	Row State
1	KATIE	12	F	59	95	•
2	LOUISE	12	F	61	123	•
3	JANE	12	F	55	74	•
4	JACLYN	12	F	66	145	•
5	LILLIE	12	F	52	64	•
6	TIM	12	M	60	84	•
7	JAMES	12	M	61	128	•
8	ROBERT	12	M	51	79	•

Since row state columns store the row states, you can apply them again later. Populate row state columns by copying them from the current row states or with column formulas.

To create a row state column

1. Select **Cols > New Columns**.
2. Next to Data Type, select **Row State**.
3. Click **OK**.

Populate the cells with new row state information or copy existing row state information from rows.

To populate cells with new row state information

1. To populate only certain rows in the row state column, highlight those rows. Or, to populate all rows in the column, highlight the row state column.
2. Right-click and select **Row States Cells**.
3. Select the row state that you want to apply.

To copy existing row state information

1. To populate only certain rows in the row state column, highlight those rows. Or, to populate all rows in the column, highlight the row state column.
2. Click the star icon ( ) beside the column name in the Columns panel.
3. Select one of the following:

Copy from Row States Replaces the row states in the column with the row states from the rows.

Add from Row States Adds the row states from the rows to the row state column.

Copy to Row States Replaces the row states in the rows with the row states from the column.

Add to Row States Adds the row states from the row state column to the row states in the rows.

Permanently Select Cells

You can save a selection in a row state column just like you save other row state characteristics (hide, exclude, color, and so on). This places a "permanent" highlight on a cell.

To permanently select cells

1. Right-click a cell and select **Row States Cells > Select/Deselect**.
2. Repeat this for as many cells as you would like to select.
3. To remove the highlight, right-click the cell and select **Row States Cells > Select/Deselect**.

Initialize Data in a Column

When you first add a new column to a data table, the Initialize Data menu appears in the Column Info window. Specify the type of initial data values that you want to appear in the new column. Initializing data is not available for Row State data types.

Select one of the following options:

Missing/Empty Places missing values in the column, represented by a black dot (•) for numeric data and a blank space for character data.

Constant Places one number or character in all of the column's rows. Enter the number or character into the box that appears. Enter any number of characters.

Today Places today's timestamp in the column for each row. This option is relevant only for the **Date** or **Time** formats.

Sequence Data Inserts sequential data based on the parameters that you specify. See "Numeric or Character Sequence Data".

Random Inserts randomly generated data values into the column. Select a method for generating the random values:

Random Integer Enter minimum and maximum integer values. Integer values within this range are generated so that each occurs with approximately equal frequency.

Random Uniform Enter a range for continuous values. Random uniform values within this range are generated.

Random Normal Enter the mean and standard deviation for a normal distribution. Random normal values from this distribution are generated.

Random Indicator Enter up to three values and corresponding desired proportions. The proportions should sum to 1. Values are generated to have a distribution that corresponds as closely as possible to the specified proportions.

Note: This method generates values in the exact proportions that you specify unless the number of rows multiplied by one of the proportions results in a fractional value.

Suppose that there are n rows and that the three proportions are p_1, p_2, and p_3. Here is how the values are generated:

- Each row is assigned a random uniform value between 0 and 1.
- The rows are ordered according to their random uniform values.
- The rows corresponding to the smallest Round(np_1) random uniform values are assigned the first value, the rows corresponding to the next smallest Round(np_2) random uniform values are assigned the second value, and the remaining rows are assigned the third value.

Numeric or Character Sequence Data

To insert sequential data for numeric data

1. Next to **Data Type**, make sure **Numeric** is selected.
2. Next to **Initialize Data**, select **Sequence Data**.
3. In the **From** and **To** boxes, assign a starting and ending point.
4. In the **Step** box, assign the sequence.
5. (Optional) In the **Repeat each value N times** box, enter the number of times that you want each numeric value repeated.
6. Click **OK**.

For example, if you want the column to contain even numbers from 2 to 60, type 2 in the **From** box, 60 in the **To** box, and 2 in the **Step** box.

To insert sequential data for character data

1. Next to **Data Type**, make sure **Character** is selected.
2. Next to **Initialize Data**, select **Sequence Data**.
3. In the box next to **Add**, enter the character data and click **Add**.
4. (Optional) In the **Repeat each value N times** box, enter the number of times that you want each character value repeated.
5. Click **OK**.

Column Properties in JMP

A column property is information that is attached to a column. Once that information is entered, it is saved as part of the data table. You can select column properties and apply them to columns. There are some column properties that JMP creates for your convenience. Often, you can modify these.

Column properties are listed in the Column Info window's Column Properties menu. The area below the Column Properties menu shows column properties that have been saved to a column. To view or edit a property, select it from this list.

In the Columns panel of the data table, an icon appears next to the name of each column that contains a property, other than the Notes property. The Notes property is not denoted by an icon. Icons include the following:

Indicates that the range or list check property is applied

Indicates that the column contains a formula

Indicates that the column contains a property other than the Note, Range Check, or List Check column property

For information about more column property icons, see "Icons Representing Column Characteristics and Properties".

To assign a column property to one or more columns

1. Select the column or columns to which you want to assign a property.
2. Do one of the following:
   - Right-click the header area, select **Column Properties**, and select the property.
   - Right-click the header area, select **Column Info**, and select the property from the Column Properties menu.
   - Select **Cols > Column Info** and select the property from the Column Properties menu.
3. In the column property panel that appears, specify values and select options as appropriate.

   Click **Apply** to add the column property or click **OK** to add the column property and close the column properties window.

A column might already contain a property that you want to apply to other columns. Use the Standardize Attributes command to apply that property to other columns. See "Standardize Attributes and Properties Across Columns".

The following sections describe the properties that you can add to columns.

Basic Column Properties

Basic column properties that apply generally include Formula and Notes.

Formula

Insert a formula into a column to compute the values for that column. After a formula is added, the column is locked so that its data values cannot be manually edited (preventing invalidation of the formula).

- Click **Edit Formula** to create a formula. For more information about creating a formula, see "Create Formulas in JMP".
- If you do not want JMP to evaluate the formula, click **Suppress Eval**.
- If you do not want JMP to alert you about errors in your formula, click **Ignore Errors**.
- Once you have created a formula:
  - In the Column Info window, a visual of the formula appears at right. However, if your formula is long, only a portion of it might appear. Click and drag the borders of the formula box to resize it.
  - From the data table, edit the formula by clicking ( ) next to the column name in the Columns panel.

Tip: To bypass the Column Info window when creating a formula, right-click the column and select **Formula**.

Notes

Adds notes to the selected column.

Properties That Validate Column Values

The following properties help validate values in a column: Range Check, List Check, and Missing Value Codes.

Range Check

Range checking validates the data in a column. Set up the column to accept only numbers that fall within a specified range.

Select which formula to use to set up the range. x is the value entered into the column, a is the beginning of the range, and b is the end of the range.

- a = the lowest value that the column accepts

- b = the highest value that the column accepts
- For a single-sided range check, leave either a or b empty.
- From the data table, modify the range check by clicking ( ) next to the column name in the Columns panel.

List Check

List checking validates the data in a column. Set up the column to accept only the individual values that you specify. List checking is useful when you want to specify how to order the data in your graphs or plots.

- You can use up to 127 values.
- Use the buttons to add new values, change, or reverse the order of values, and remove values.
- Once a list check is set on a column, the cursor changes to when positioned over the cells. If you try to enter a value not included on the validation list, a warning message appears.
- To see a menu of acceptable values, right-click a cell and select **List Check Values**. You can select the cell value from the menu instead of entering it into the cell.
- From the data table, modify the list check by clicking next to the column name in the Columns panel.

Missing Value Codes

Use missing value codes to specify column values that should be treated as missing. For example, sometimes the value 99 is used as a placeholder to represent missing values, or perhaps several values are used to represent different types of missing values.

Note: The Missing Value Codes column property is an extended attribute. If you plan to export to a SAS 9.4 server or Save As a SAS 9.4 file format, you should enable extended attributes in the SAS Integration Preference page. See "Preferences for SAS Integration". Alternatively, you should select **Store table and column properties in SAS 9.4 extended attributes** in the Save As window. If extended attributes are not selected, JMP exports or saves Missing Value Codes as missing values. Extended attributes are supported only by SAS 9.4. Earlier versions of SAS disregard any extended attributes.

To remove missing value codes for a single column

1. In the Columns panel of the data table, next to the column, click > **Missing Value Codes**.
2. Missing Value Codes is highlighted in the list. Click **Remove**.

3. Click **OK**.

To remove missing value codes across multiple columns

1. Select the columns containing the missing value codes that you want to delete.
2. Select **Cols** > **Standardize Attributes**.
3. Under Delete Properties, click **Column Properties** > **Missing Value Codes**.
4. Click **OK**.

Properties That Attach Information to Column Values

The following column properties attach information that is used in reports and plots to the values in a column:

- Value Labels
- Value Scores
- Value Order
- Row Order Levels
- Value Colors
- Color Gradient
- Short Name
- SPSS Name
- SPSS Label
- SAS Name
- SAS Label
- Long Name
- Question

Value Labels

Use value labels to show a label in the data table instead of a value. A label appears for each instance of the value. You can show the original values by double-clicking a label within a cell.

- Enter the value that you want to assign a label to in the Value box.
- Enter the label that you want to appear in the Label box.
- To use ranges, click **Allow Ranges** then specify the lower and upper values. If Allow Ranges is selected, Value Labels can be non-integers. Graph Builder can use the Value Labels for the x- and y-axes.

Tip: To assign a label to missing values, enter a period (**.**) for the lower bound and leave the upper bound empty. To assign a label to all other values, enter three periods (**...**) for the lower bound and leave the upper bound empty.

- Add, change, or remove labels.

Tips:

- To turn off value labels in the data table without deleting the value labels that you have set up, in the Column Properties window, deselect **Use Value Labels**. Or, to quickly show or hide value labels in a data table, right-click a column and check or uncheck **Use Value Labels**.
- When your data table contains value labels, using the **Search** commands searches for actual values, but does not search for labels.
- When your data table contains value labels, the **Row Editor** displays the label, and when the cell is highlighted for editing, it shows the actual value.
- If you copy and paste a cell with a value label, the actual value is pasted.
- In a formula, when you reference a column using value labels, hover over the value label to see the actual data value.

Value Scores

Use value scores to indicate a value-score pair for categorical data columns. The value is a data value and the score is a number. This property associates a data value with a score (for example, the column's data value could be "not satisfied", "satisfied", "very satisfied"). The user could assign a score of 0 to "not satisfied", 50 to "satisfied", and 100 to "very satisfied". Those scores are then used for computation purposes like computing the mean. See *Consumer Research* for an example.

To use value scores

1. In the data table, right-click the column and select **Column Info**.
2. From the **Column Properties** list, select **Value Scores**.
3. Enter the value that you want to assign a label to in the Value box (for example, "satisfied").
4. Enter the label that you want to appear in the Score box (for example, "50").
5. Click **Add**, **Change**, or **Remove**.
6. Click **OK**.

Value Order

Value Order assigns an ordering to values in a column that is then used in reports and plots. The default ordering for reports and plots is alphanumeric data value ordering, except in the following cases, where values are automatically ordered appropriately:

- January, February, March, April, May, June, July, August, September, October, November, December
- Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
- Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday
- Very Low, Low, Medium Low, Medium, Medium High, High, Very High
- Strongly Disagree, Disagree, Neutral, Indifferent, Agree, Strongly Agree
- Failing, Unacceptable, Very Poor, Poor, Bad, Acceptable, Average, Good, Better, Very Good, Excellent, Best

Tip: You can anonymize Value Order so that the Value Order properties are renumbered. To do this, highlight the column with the Value Order column property and select **Tables > Anonymize.**

When you construct a design using a DOE platform, the Value Order column property is automatically assigned to categorical factors. For examples of assigning and editing the Value Order column property, see the *Design of Experiments Guide*.

Note: When you're comparing character data, select **Use Locale Comparisons for Characters** to consider locale information when determining the ordering.

Custom Order

By default, Custom Order is selected. This option allows you to order data values in a user-defined order. Initially, character data is ordered alphabetically and numeric data is sorted numerically. You can use the arrows to move the values and the plus sign to add values.

Sorted Order

Deselect **Custom Order** to move the values to the Sorted Order box. Then you indicate the rules for sorted items.

Items that are not custom ordered are affected by the sorted order. Drag items down to the Sorted Order area to remove them from the custom order and have the sorting rules applied to those items. Deselect **Custom Order** to apply sorting rules to all items.

The sort order rules are applied when you select or deselect the check boxes.

Row Order Levels Sorts the column values by their occurrence in the data table, rather than sorting by value. This option does not apply to sorting data table columns.

Note: If both Row Order Levels and Common Order are selected, the row order levels apply.

Common Order If all values have a predefined sort order (for example, the days of the week), remaining items are sorted on that basis. Common order is the typical ASCII ordering. This option is available only for character data.

Numerical Order Sorts character column values by splitting the data into string and numeric segments.

Here are the numerical order rules:

- Strings are broken into segments of numeric (not including +/-, but including dots in the middle) and non-numeric characters.
- Segments are compared between two strings, comparing numbers by magnitude as long as the numbers don't start with leading zeros.

  Numbers with leading zeroes are sorted before any numbers without leading zeros.

  If both numbers have leading zeroes, then the numbers are compared digit-by-digit. For example, "0CH2" is sorted before "0 CH2".
- If two strings differ by segment type, a numeric segment sorts before a string segment.
- The locale compare affects symbols; characters such as hyphens and lowercase or uppercase are ignored in locale sorting. (Note that lowercase is sorted before uppercase in ASCII ordering. The rules are different for each language. The goal is to sort the same way that the operating system sorts file names.

  To use ASCII ordering, deselect **Numerical Ordering** in the Value Order column property.
- The string segments are compared using locale compare by default. This comparison is language-aware. The comparison is provided by the operating system, so there are subtle differences between behavior on Windows and macOS.

Value Colors

Use value colors to assign the values of a nominal or ordinal column a certain color or range of color themes. The column's values appear with the assigned color in all applicable graphs, such as mosaic plots and plots with color-coded legends. You can also color the values in the data table column.

- To change the color of a specific value, right-click a color circle and select a color.

- To use a color theme, click **Color Theme** below the list of value colors and then select the color.
- To create a custom color theme, see "Create a Custom Color Theme".
- To also color the cells in the data table, select **Color Cell by Value**.
- Select from the options in the Macros menu. See "Macros Options".

Create a Custom Color Theme

To create a custom color theme

1. Click **Color Theme** below the list of value colors in the column properties.
2. Open the Custom Color Theme outline.
3. Click **New**.
4. Create the color using the sliders.
5. Name the color and click **Save**.
6. Click **OK**.

You can access the new custom color in the Color Theme menu the next time you use the Value Colors property. The color is also saved in your preferences. See "Create Color Themes in Graphs" for more information about creating color themes.

Macros Options

The Macros menu contains the following options:

Gradient between ends Sets the colors of the top and bottom values. JMP applies a color gradient across the entire range of values. Use this command to make all of the colors in between for the other levels.

Gradient between selected points Sets the colors of the top and bottom values so that JMP can apply a color gradient to a range of values that you have highlighted in the Value Colors list.

Reverse colors Reverses the color of the values from top to bottom or bottom to top.

Revert to old colors Sets the colors back to their original color values.

Color Gradient

Select a color gradient to color a continuous column in a plot. Color gradients are supported in the Graph Builder, Bubble Plot, Treemap, and Cell Plot platforms.

- To also color the cells in the data table, select **Color Cell by Value** above the list of value colors.

- Select a color gradient from the menu.
- Enter the minimum, maximum, and center values:
  - Minimum values reflect the color at the left of the gradient.
  - Maximum values reflect the color at the right of the gradient.
  - Center values reflect the color in the middle of the gradient.

Note: To see color gradients in Graph Builder, you must assign the column to the Color zone. To see color gradients in Bubble Plot and Treemap, you must assign the column to the Coloring role.

Short Name, SPSS Name, and SAS Name

These column properties insert the specified short name in a Categorical structured report when the Show Columns Used in Report red triangle option is selected.

SAS Name is also a column property that is added when you import SAS data. It contains the name of the variable in the SAS data set. Modifying it will affect the SAS data set if the data are exported back to SAS.

SPSS Label, Long Name, SAS Label, and Question

These column properties insert the specified long name in a Categorical structured report when the Show Columns Used in Report red triangle option is selected.

SAS Label is also a column property for imported SAS data. It contains the description of the column in the SAS data set. Modifying it will affect the SAS data set if the data are exported back to SAS.

Properties That Control the Display of Columns

The Axis and Units column properties control how column values are displayed.

Axis

Use the Axis property to change the default axis settings for a column. JMP automatically uses your settings when the column appears in an analysis. See "Customize Axes and Axis Labels in Graphs" for more information about the properties.

When you add the Axis column property, you can save all column properties or only the properties that you changed. For example, you might want the axis to start at 0 and are not concerned about the upper bounds of the axis. Set the minimum axis value and select **Changed Properties Only** from the Save to column list.

To set default axis properties for a column from within a graph

1. Create the graph.
2. Change the axis to your preferred specifications.
3. Right-click the axis and select **Save to Column Property**.

Units

Use the Units property to specify the measurement units that were used to collect the data for the column. The units appear in parentheses after the column name in the data table and when the column appears in plots. For example, you might want a column to indicate that age values are measured in months, or that a monetary value is in thousands of dollars.

Properties Used in Modeling and DOE

When you construct a design using the DOE platforms, column properties are saved to the resulting design table. However, some of these column properties are useful in general modeling situations. To use the properties more generally, you can specify them yourself.

Note: the *Design of Experiments Guide* describes the properties associated with DOE in detail and presents examples. Only a brief description of these properties is provided here.

The following column properties are used in modeling and DOE:

- Response Limits
- Detection Limits
- Design Role
- Coding
- Mixture
- Factor Changes

Response Limits

The Response Limits column property defines a desirability function for the response. The Prediction and Contour Profilers use desirability functions to find optimal settings. See the *Design of Experiments Guide*.

JMP PRO Detection Limits

The Detection Limits column property defines bounds beyond which the response cannot be measured. You can use these limits to specify a censored response in the Generalized Regression platform. See *Fitting Linear Models*.

Design Role

The Design Role column property indicates how a column is used in both a designed experiment and in the model used to fit the data. For example, a column could represent a continuous factor, a categorical factor, a blocking factor, and so on. See the *Design of Experiments Guide*.

Coding

The Coding column property applies a linear transformation to the data in a numeric column. The data are transformed to –1 and +1 based on bounds that you specify. JMP then uses the transformed data values whenever the column is entered as a model effect in the Fit Model platform. See the *Design of Experiments Guide*.

Mixture

The Mixture column property is useful when a column in a data table represents a component of a mixture. The components of a mixture are constrained to sum to a constant. The Mixture column property identifies a column as a mixture component and defines a coding for that column. See the *Design of Experiments Guide*.

Factor Changes

The Factor Changes column property indicates how difficult it is to change factor settings in a designed experiment. It is used to create and analyze split-plot, split-split-plot, and two-way split-plot designs. See the *Design of Experiments Guide*.

Properties Associated with Control Charts and Capability

The Specification Limits, Control Limits, Sigma, and Process Capability Distribution properties are associated with control charts and capability.

Spec Limits

Specification limits are used when you perform a capability analysis using the Distribution and Capability platforms. Note that in old data tables, the column property might be named Capability Analysis.

You can specify any combination of a Lower Spec Limit, an Upper Spec Limit, or a Target. The **Show as Graph Reference Lines** option displays the specification limits and target that you specify as reference lines on appropriate plots.

Tip: To add specification limits to several columns at once, see *Quality and Process Methods*.

Control Limits

The Control Limits column property enables you to specify control limits for a column, rather that having JMP calculate control limits from the values in the column. This is useful when you are comparing process data to historical control limits.

The Control Limits column property can be assigned only to a numeric column. To assign the Control Limits column property, select the control chart type and then enter the requested values. If any of these entries are missing, JMP replaces it with a calculated value in the control chart.

Sigma

Use the Sigma property to enter a known sigma value. This value is used by applications such as control charts or any application that requires a sigma value to complete computations. If no sigma value is supplied, sigma is calculated from the sample.

Process Capability Distribution

The Process Capability Distribution column property specifies a process distribution for the column. The Process Capability Distribution column property is used only in the Process Capability platform. See also "Distribution and Process Capability Distribution".

QC Alarm Script

You can generate automatic alarms by writing a script and storing it as a column property or data table property named QC Alarm Script.

The following QC Alarm Script automatically writes a message to the log whenever a test fails:

```
Write(
        "Out of Control for test ",
        qc_test,
```

```
            " in column ",
            qc_col,
            " in sample ",
            qc_sample
        )
```

Properties That Control How Columns Are Used in Platforms

The following properties control how columns are used in platforms:

- Distribution
- Time Frequency
- Map Role
- Supercategories
- Multiple Response
- Target Level
- Profit Matrix
- Informative Missing

Distribution

For a column that contains continuous numeric data, use the Distribution property to select a distribution type to fit to the column. This distribution is used in the Distribution platform and is used in the Process Capability platform under certain conditions. See "Distribution and Process Capability Distribution".

When you obtain a Distribution report (by selecting **Analyze > Distribution**) for the column, JMP automatically estimates a fit using the specified distribution. A curve representing the fitted distribution is superimposed on the histogram.

If you set both the Distribution property and the Spec Limits property, then the Distribution platform produces a Capability Analysis report that is based on the distribution specified in the Distribution column property.

Note: In older data tables, the Capability Analysis column property might have been assigned to a column to specify spec limits. We recommend that you use the Spec Limits property instead.

Distribution and Process Capability Distribution

If you are using the Process Capability platform to analyze a column that does not contain a Process Capability Distribution property, the distribution specified in the Distribution column property results in a nonnormal fit in the Process Capability platform. If this distribution is supported in the Process Capability platform, then it is used. However, if this distribution is not supported, the platform uses a Johnson fit.

If both the Distribution and Process Capability Distribution column properties are saved for a given column, then the distribution specified in the Distribution column property is used in the Distribution platform, and the distribution specified in the Process Capability Distribution column property is used in the Process Capability platform.

Time Frequency

When using the Time Series platform, you can assign the Time Frequency property to data. The Time Frequency property specifies the frequency with which the data is reported (such as annually, quarterly, monthly, and so on). Specifying a time frequency allows JMP to take things like leap years and leap days into account. If no frequency is specified, the data is treated as equally spaced numeric data.

Map Role

If you have created a data table that contains boundary data (such as countries, states, provinces, or counties), to see a corresponding map in Graph Builder, use the Map Role property.

Note the following:

- If the custom boundary files reside in the default custom maps directory, then you need to specify only the Map Role property in the -Name file.
- If the custom boundary files reside in an alternate location, then you must specify the Map Role property in the -Name file and in the data table that you are analyzing.
- The columns that contain the Map Role property must contain the same boundary names, but the column names can be different.

For an example using the Map Role property, see *Essential Graphing*.

To add the Map Role property into the -Name data table

1. Right-click the column containing the boundaries and select **Column Properties > Map Role**.
2. Select **Shape Name Definition**.
3. Click **OK**.
4. Save the data table.

To add the Map Role property into the data table that you are analyzing

Note: Perform these steps only if your custom boundary files do not reside in the default custom maps directory.

1. Right-click the column containing the boundaries and select **Column Properties > Map Role**.
2. Select **Shape Name Use**.
3. Next to **Map name data table**, click to browse to a -Name map data table. You can enter the relative or absolute path.

   If the map data table is in the same folder, enter only the filename. Quotation marks are not required when the path contains spaces.
4. Next to **Shape definition column**, enter the name of the column in the map data table whose values match those in the selected column.
5. Click **OK.**
6. Save the data table.

When you generate a graph in Graph Builder and assign the modified column to the Shape zone, your boundaries appear on the graph.

Supercategories

When a data set contains ratings (for example, on a five-point scale), you might want to know the percent of the responses in a subset of those ratings. Add a Supercategories column property to group specific categories into one category.

Supercategories are supported only in the Categorical platform.

To add the Supercategory property to a data column

1. Right-click the column that contains categories that you want to group.
2. Select **Column Properties > Supercategories**.

   The column properties window shows the Supercategories options (Figure 5.5).
3. Select the categories in the Column's Categories list that you want to group.
4. Enter a descriptive name next to Supercategory Name.

   Leave the name blank, and JMP names the supercategory after the categories that you selected.
5. Click **Add** to create the supercategory.
6. Click the **Supercategories** red triangle and select from the following options:

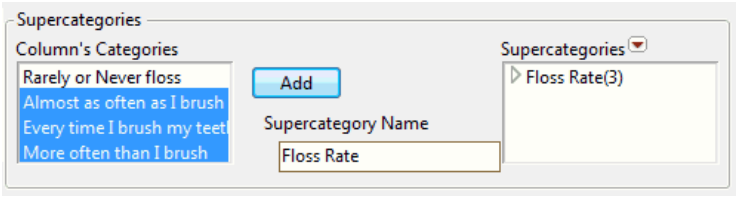
   Options > Hide Hides data in the selected supercategory from reports and graphs.

Add All Creates a supercategory from all of the categories in the column.

Add Mean and Add Std Dev Calculate statistics for value scores. See *Consumer Research*.

7. Click **OK** to add the property to the column.

Figure 5.5 Example of a Supercategories Configuration



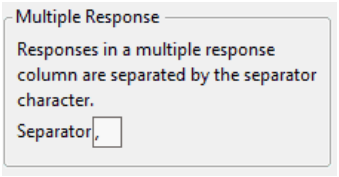
Multiple Response

The term *multiple response* refers to the situation where the cells in a column contain more than one response value. For example, many cells in the Brush Delimited column in the Consumer Preferences.jmp sample data table contain multiple values. For example, row 6 contains "Wake, After Meal, Before Sleep".

Add the Multiple Response column property if you want to specify a delimiter other than the comma. Otherwise, change the column's modeling type to Multiple Response in the Column Info window. See "About Modeling Types" for more information about the Multiple Response modeling type.

JMP automatically assigns the Multiple Response modeling type to data tables saved in JMP 12 or lower. The column must contain the Multiple Response column property, and the delimiter must be a comma for the automatic assignment to take place. JMP does not remove the Multiple Response column property, though you might choose to do so.

Figure 5.6 Multiple Response Configuration Window



Note: You can use the Multiple Response property in the Categorical platform. See *Consumer Research*. You can also use this property in the Data Filter. See "The Data Filter". If the delimiter is a comma, consider using the Multiple Response modeling type instead.

Target Level

Use the Target Level column property to specify the level of interest of a response column for logistic regression platforms. This level is used as the positive response level in logistic regression models, and it impacts the statistics, probability curves, and other measures of fit. This column property is designed for binary responses, but it can be used for any categorical column.

Note: The Target Level column property is used by the following platforms when a binary response column is specified: Fit Y by X (Logistic) and the Nominal Logistic personality of Fit Model.

Profit Matrix

Use the Profit Matrix column property to assign weights to the levels of a nominal or ordinal response variable for a predictive model. For a nominal response, you can specify the profit matrix entries using a probability threshold.

Note: The Profit Matrix column property is used by the following platforms: Model Comparison, Partition, PLS, and Fit Model. The personalities in Fit Model that use the Profit Matrix column property are Generalized Regression, Nominal Logistic, and Ordinal Logistic.

Profit Matrix

When you select **Column Properties > Profit Matrix**, a matrix template appears, with a row and column for each value in the selected column. The Actual levels are shown as rows and the predicted levels are shown as columns. Correct decisions are those on the diagonal, where the predicted level equals the actual level.

- For diagonal entries, enter values that reflect profits or weights for correct decisions.
- For non-diagonal entries, enter values that reflect profits (or losses) or weights for incorrect decisions.
- For situations where no prediction is made, use the Undecided column to indicate associated profits or losses.

Probability Threshold Specification for Profit Matrix

When the response is binary, additional options appear beneath the profit matrix template. These options enable you to specify a probability threshold instead of entering weights directly into the profit matrix.

Specify the Target Level and Probability Threshold. Then click Set to update the profit matrix.

Target Level The level whose probability is modeled.

Probability Threshold A threshold for the probability of the target level. If the probability that an observation falls into the target level exceeds the probability threshold, the observation is classified into that level.

Set Enters values into the profit matrix template that reflect your specifications for Target and Probability Threshold. See "Probability Threshold Calculations".

Probability Threshold Calculations

Denote the threshold probability by *t*. When you click Set, the entries in the profit matrix are assigned as follows:

- 0 for each diagonal entry, reflecting no loss from correct decisions
- -1 for a prediction of the target level when the actual value is the non-target level
- $-t/(1-t)$ for a prediction of the non-target level when the actual value is the target level

This implies that the profits for classifying into the two levels are given as follows:

$$\text{Profit for Target Level} = \left(\frac{-t}{1-t}\right)\text{Prob[Non-Target Level]}$$

$$\text{Profit for Non-Target Level} = -\text{Prob[Target Level]}$$

The Most Profitable Prediction is the level whose profit is the larger of these two values. It follows from the two Profit equations above that an observation is assigned to the target level whenever Prob[Target Level] is at least *t*.

Profit Matrix and Predictive Models

For a nominal or ordinal column with the Profit Matrix column property, most modeling platforms enable you to save formula columns that reflect profit matrix entries. Fit your model and then select the Save Prediction Formula or Save Probability Formula option. In addition to saving the usual prediction formulas to the data table, JMP saves the following analogs of the usual formula columns:

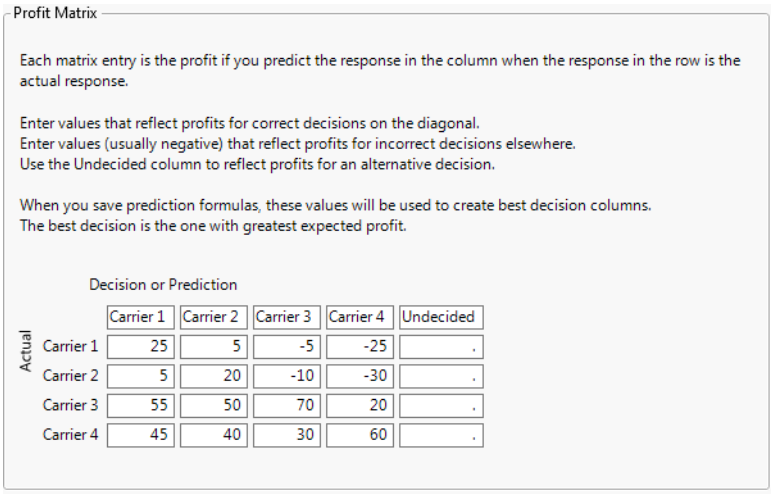
- Profit for <level>: For each level of the response, a column gives the expected profit for classifying each observation into that level.
- Most Profitable Prediction for <column name>: For each observation, gives the level of the response with the highest expected profit.
- Expected Profit for <column name>: For each observation, gives the expected profit for the classification defined by the Most Profitable Prediction column.
- Actual Profit for <column name>: For each observation, gives the actual profit for classifying that observation into the level specified by the Most Profitable Prediction column.

See "Example of a Profit Matrix for More Than Two Levels". For an example of using a profit matrix in modeling, see *Predictive and Specialized Modeling*.

Example of a Profit Matrix for More Than Two Levels

The example below shows a profit matrix for the Airline column in the Travel Costs.jmp sample data table.

Figure 5.7 Example of Profit Matrix Window



To see how the values in this profit matrix were assigned, consider a travel agency that uses four airlines, Carrier 1 to Carrier 4, to service its customers. For each ticket sold, the agency realizes a profit that depends on the carrier selected by the customer. When the agency recommends, or *predicts*, a carrier, it reserves a ticket for a small fee. If the customer decides to use the predicted carrier, the agency profits by a certain amount less the reservation fee. However, if the customer decides to take a different carrier, the agency loses the reservation fee and must pay another reservation fee. The agency's profit is lower due to the incorrect prediction.

Suppose that the reservation fees for Carriers 1 through Carrier 4 are $15, $20, $30, and $50, respectively, and that the profits from ticket sales are $40, $40, $100, and $110, respectively.

If the agency recommends Carrier 1 to a customer who then decides to purchase the ticket, the agency reserves a ticket for $15 and then receives $40 for a net profit of $25. If the agency predicts that a customer will choose Carrier 4 but the customer chooses Carrier 1, the agency loses $50 for the Carrier 4 reservation and must also pay $15 for the reservation on Carrier 1. This gives the agency a net loss of $40 - $50 - $15 = -$25.

Example of Using Probability Threshold to Define the Profit Matrix

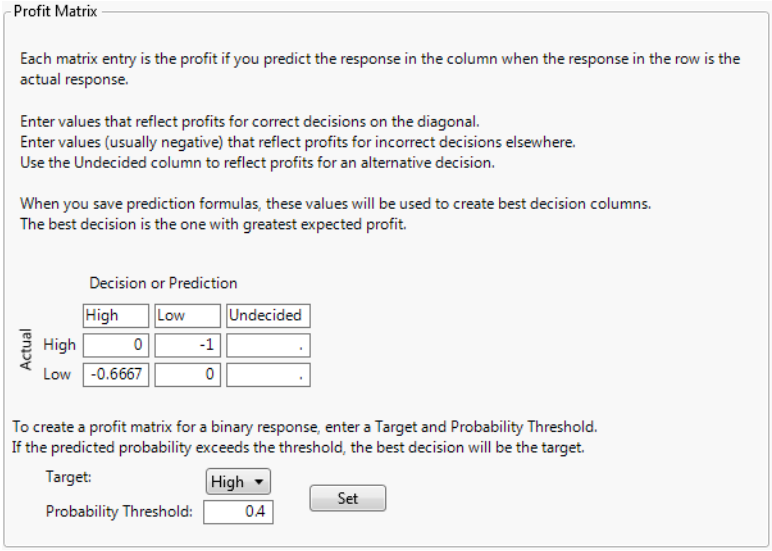
The sample data table Liver Cancer.jmp gives disease Severity ratings for 136 patients. You are interested in modeling Severity using the predictors given in the columns from BMI to Jaundice. The usual prediction formulas for a model classify a patient into the Severity level that is most probable. However, classifying a patient as having Low severity when in actuality the patient's severity is High is a more costly error than classifying a patient as having High severity when in actuality the patient's severity is Low. As a result, you want to assign a higher cost to misclassifying a patient as Low, when the patient's severity is actually high.

You can assign this higher cost by setting a probability threshold. With input from experts, you determine that the following is a good strategy: Classify into the High level of Severity any patient whose predicted probability of being in the High level exceeds 0.4.

1. Select **Help > Sample Data Library** and open Liver Cancer.jmp.
2. Select the Severity column and select **Cols > Column Info**.
3. Select the Profit Matrix column property.
4. Change the Target to **High**.
5. Enter 0.4 as the Probability Threshold.
6. Click **Set**.

   The profit matrix updates to show the corresponding weights.

Figure 5.8 Profit Matrix Showing Weights Corresponding to Probability Threshold



The profit matrix shows that the loss for misclassifying a patient with High severity as having Low severity is -1, while the loss for misclassifying a patient with Low severity as having High severity is smaller, -0.6667.

Informative Missing

The Informative Missing column property directs most fitting platforms to use a coding system for columns that contain missing values. For continuous columns, the coding system consists of two columns. The first column is a column of the original values with missing values replaced by the mean of the nonmissing values; the second column is an indicator column to denote which rows are missing. For categorical columns, missing values are treated as a distinct level of the column.

SAS Properties

SAS Label, SAS Name, SAS Format, and SAS Informat are assigned to columns when you import a SAS data set. You can change the value in the column property to control the label, name, format, or informat of the column that gets created in the SAS data set if the data is exported back to SAS.

Link Data Tables with Virtual Join Properties

Virtual Join links a main data table to one or more auxiliary data tables. The feature enables the main data table to access data from the auxiliary data tables without physically joining the tables. See "Virtually Join Data Tables".

The Link ID and Link Reference column properties make the linking possible.

Notes:

- The data types of the Link ID and Link Reference columns must match.
- Link ID and Link Reference do not work for Row State and Expression data types.
- Consider adding the Link ID column property before adding the Link Reference column property so that the referencing column in the main table has a column to link to.

Link ID

The Link ID column property marks a column in the auxiliary data table as the ID column. That is, the rows of the data table are uniquely identified by the values of the ID column. The data table that has a Link ID column property is referred to as the *referenced* data table.

To add the Link ID column property, select the column, select **Cols > Column Info** and then select **Link ID** from the Column Properties list. Make sure that the Link ID checkbox is selected and click **OK**. The selected checkbox indicates that the column is the ID column for the data table.

Link Reference

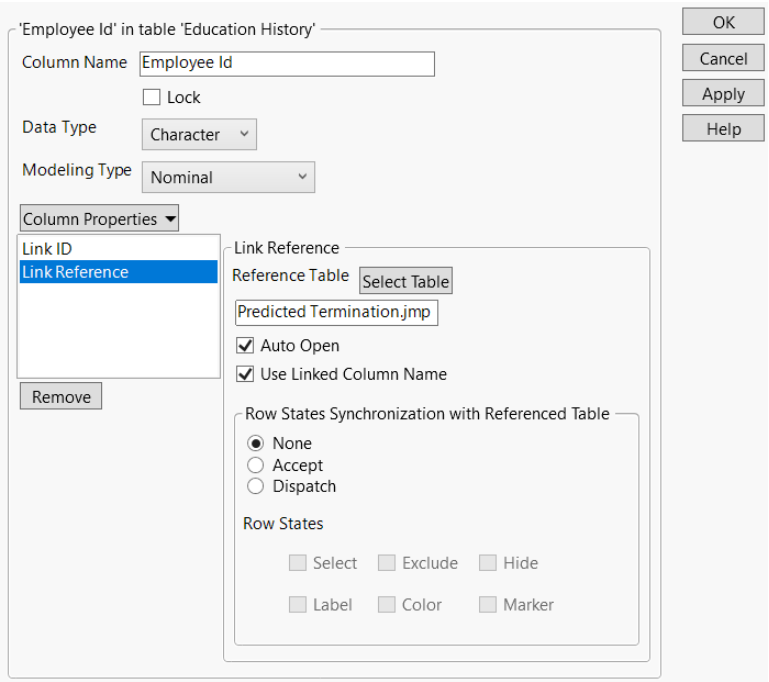
The Link Reference column property maps a column in the main data table to the ID column in the referenced data table. The column that has a link reference is referred to as the *referencing* column. The referencing column can look up the data of the auxiliary data tables through the ID column.

To add the Link Reference column property, select the column, select **Cols > Column Info**, and then select **Link Reference** from the Column Properties list. Enter the path to the main table and click **OK**. If you enter the path correctly, the referenced columns show up in the Columns panel of the main table after you click OK.

Notes:

- The path is case sensitive.
- Consider putting the data tables in the same folder. Then you can omit the directory name from the Link Reference property and include only the data table name. Otherwise, include the full path to the data table (such as c:/users/marie/My Data.jmp).
- You can also increase portability by using a JMP path variable in the Link Reference path. Figure 5.9 shows an example. Enter a forward slash in the path to share the data tables with both Windows and macOS users. See the *Scripting Guide* for more information about path variables.
- You can avoid entering the path by right-clicking a column to add the Link Reference property. See "Example of Virtually Joining Columns".
- Select Row State Synchronization options to synchronize row states in data tables that contain virtually joined columns. See "Example of Virtually Joining Columns and Showing Row States".
- Select Auto Open to open the linked table automatically when the main data table is opened. See "Update Data Tables".
- You can specify that the linked column name be used in virtually joined columns. This option lets you specify a shorter column name (for example, Cheese instead of Cheese[Choice]). In the Link Reference column property, select **Use Linked Column Name**.

Figure 5.9 Link Reference Column Property Definition



Additional Column Properties

There are three additional properties: Expression Role, Event Handler, and Other.

Expression Role

When you import a column of data that contains image data, the data type is set to Expression to show the data as pictures. Or you can assign the Expression data type in the Column Info window.

To customize how the image looks in the data table cell, or to display matrices and expressions as images, add the Expression Role column property. The default is set to Picture. If the Expression Role is set to Picture and the expression is a picture, then the expression evaluates to a picture and the picture appears in the data table. Otherwise, the expression shows as a JSL expression. Change this setting to alter how expressions are used.

Note: Expression columns that contain images greater than 32KB in size are truncated when saved to Microsoft Excel. Those expression columns are not restored when imported back into JMP.

Select from the following options:

Picture If the expression contains picture data, the column evaluates and the picture is displayed.

Matrix If the expression contains a matrix, the column evaluates and the matrix is displayed. Otherwise, the data is displayed as an expression.

Expression If the expression is simply an expression, use this option to display it in the column.

Use Expression when you want to work with an expression that might contain picture data, but you want to prevent JMP from displaying the image.

Picture expressions To customize how images appear in a data table cell, select one of the following options after selecting Picture from the list.

maximum width and height Specifies the maximum width and height of the image in the data table cell.

Stretch Enlarges a small image to fill the data table cell.

Preserve aspect ratio Maintains the image's ratio of width to height in the data table cell. Selected by default.

Add a decorative frame to small images in a data table Displays a beveled edge around small images in the data table cell.

To drag images into an expression column

1. Create a new column in a data table.
2. Right-click the column, select **Column Info**, and then select the **Expression** data type.
3. Right-click the column, select **Column Properties**, and then select **Expression Role**.
4. Select options as described above.
5. Navigate to the website that contains the image that you want in your data table. You can also use an image that is on your computer.
6. Select the image, and drag it into an empty cell in the expression column.

   Resize the cell to make the image larger.

   The SAS Offices.jmp and Big Class Families.jmp sample data tables show an example of an expression column.

Tips:

- You can also import data that contains a column of images to create an expression column.
- For more information about what you can do with images in an expression column, see "Use Images as Markers" and "Add Images from Data Tables to Hover Labels".

To add an expression column using Summary

1. Select **Help > Sample Data Library** and open CrimeData.jmp.
2. Select **Tables > Summary**.
3. Select State, and click **Group**.
4. Select Total Rate, and click **Statistics > Histogram**.
5. Click **OK**.

   A new data table appears that contains a new expression column with images of histograms. Resize the row to make images larger.

   For more information about Summary, see Chapter 8, "Summarize Your Data".
6. Right-click the Histogram(Total Rate) column, and select **Column Info**.

   Notice the column's data type is set to Expression.
7. Select **Expression Role** from the Column Properties menu.

   The Expression Role is automatically set to Picture. In the rare case that a picture needs to be displayed as an Expression, change the setting to Expression.
8. Click **OK**.

   Expression Role is assigned to the Expression column.

Event Handler

Use the Event Handler column property to define hyperlinks in a text column. The hyperlinks can be used to open web pages and other documents.

Add the Event Handler to a column of URLs to open the URLs in a browser. You can also modify the placeholder scripts on the column property to open different kinds of resources. For example, in a data table of geographical data, you might edit the Click script to display a Google map when the user clicks the hyperlink in the data table. The SAS Offices.jmp sample data table provides examples.

See the *Scripting Guide* for more information about editing the scripts.

Other

Use the Other column property to create your own column property and assign it any name that you choose. This property is then available for JSL programming.

1. Right-click the column and select **Column Properties > Other**.
2. Enter a name for the new property.
3. Enter a value for the property.

Properties Assigned and Controlled by JMP

To control how information flows between platforms, JMP assigns some properties that you cannot control. These properties do not appear on the Column Properties menu.

Response Probability

In Profiler reports, the Response Probability property makes all levels of the categorical response variable appear in a single row. JMP automatically assigns the Response Probability property when certain probability formulas are saved to the data table.

Follow these steps to save the property:

1. Fit a logistic regression model using the Fit Model platform.
2. Click the report red triangle and select **Save Probability Formula**.

   JMP automatically assigns the Response Probability property to the new probability columns.

For more information about the prediction profiler, see *Profilers*.

Expand Formula

In the Profiler and Nonlinear platforms, if an ingredient column to the model might have a formula, JMP substitutes the inner formula, as long as it refers to other columns. Add the Expand Formula property to prevent an ingredient column from expanding. Assign the value of 0.

See *Profilers*.

Predicting

JMP automatically assigns the Predicting column property when you fit a model to a continuous response and save the prediction formula or prediction values. The Predicting column property identifies the platform used to create the prediction formula. That platform is listed as the Creator in the Model Comparison platform.

Informative Missing Terms

Missing levels of a categorical variable are treated as informative missing in certain JMP platforms, either by default or if you request an informative missing fit. In these cases, JMP automatically assigns the Informative Missing Terms column property to a prediction formula column that includes the categorical variable. This column property ensures that the missing value category is treated as a distinct level of the categorical variable in plots and analyses that involve the prediction formula. In particular, profiler plots show the missing values as a level.

JMP PRO Column ID

JMP automatically assigns the Column ID column property to the column that is used to split the initial Bootstrap results data table. This column property assists the split operation and ensures that the column names in the split Bootstrap results table are meaningful.

ConstraintState

When you select Save Constraints, the coefficients of each linear constraint appear in a column in a data table. Each constraint column is assigned the ConstraintState column property. This property specifies the direction of the inequality that defines the constraint. When you select Load Constraints from a design platform, the ConstraintState column property tells JMP the direction of the inequality.

RunsPerBlock

The RunsPerBlock property indicates the maximum allowable number of runs in each block. This property is used by the Evaluate Design and Augment Design platforms to indicate the blocking structure for the factor.

DF

The DF column property is added by many modeling platforms to formula columns that calculate the standard error of the predicted values. The value of this column property is used by the Prediction Profiler to draw confidence intervals when both a prediction formula column and a standard error formula column are provided.

Caution: We do not recommend that you remove or change the value of this property. Doing so could adversely affect the confidence intervals in the Prediction Profiler.

Standardize Attributes and Properties Across Columns

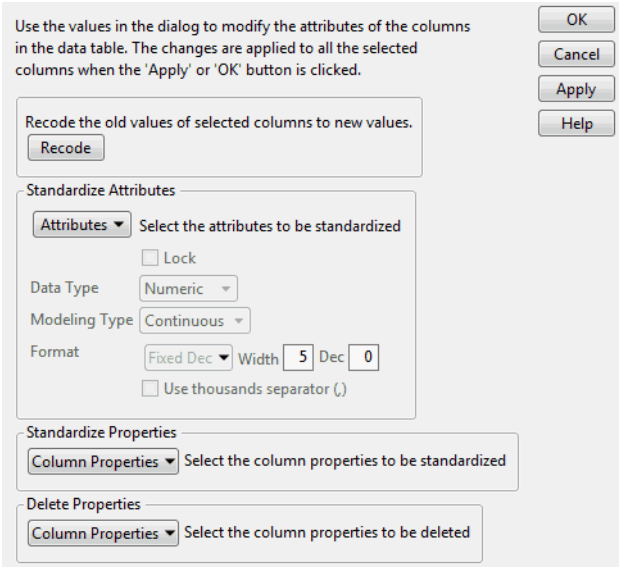
A column might contain attributes (data types, modeling types, numeric formats, and so on) or properties (formulas, notes, list and range checks, and so on) that you want other columns to have. You can use the existing column to *standardize* the attributes and properties across columns. This includes both adding and deleting attributes and properties.

To apply an existing column's attributes and properties to multiple columns

1. Select the column or columns containing the desired attributes or properties.

2. Select **Cols > Standardize Attributes**.

Figure 5.10 Standardizing Attributes across Columns



Recode Column Values

Use the Recode option if you need to recode similar values within multiple columns in the same way. After you click **Recode**, the values that appear are generated from the union of all of the selected columns. For example, suppose that you have two columns with user responses. One column contains the values Agree and Disagree. The other column contains the values Agree, Disagree, Unsure, and Strongly Disagree. You want to simplify all of the values in both columns to be A, D, U, and SD.

Note: If you want to recode values only in a single column, you can also use the Cols > Recode option. See "Recode Data in a Column".

Standardize Column Attributes

By default, the items within the Standardize Attributes panel are dimmed. To access an item, click the **Attributes** button and select the items to be duplicated across columns.

Note: The Input Format item is applicable only for the Date, Time, and Duration formats.

To change the values of any of the attributes, use the menus in the Standardize Attributes panel.

Standardize Column Properties

To standardize properties across columns

1. Select the columns that you want to standardize. One of those columns must contain the property that you want to add to the other columns.
2. Select **Cols > Standardize Attributes**.
3. Select **Column Properties** in the Standardize Properties area.
4. Select the property that you want to standardize. Modify the property settings if necessary.
5. Click **Apply** to standardize the property across selected columns. Changes are shown in the data table, open reports, and open graphs.

   Or click **OK** to standardize the property across selected columns and close the column properties window.

Delete Column Properties

To delete the same properties across multiple columns

1. Select the columns containing the attributes or properties that you want to delete.
2. Select **Cols > Standardize Attributes**.
3. Under Delete Properties, click **Column Properties** and select the properties that you want to delete.
4. Click **OK**.

Note: Make sure you click OK and not Remove. Clicking Remove simply removes the property from the list but does not delete it.

Example of Standardizing a Formula

If you have applied a formula to a column, and you want to apply that same formula to additional columns in the data table, use the Substitute Column Reference option.

Note: This option is dependent upon the location of the column that is referenced in the original formula. For example, if your original formula is based on the previous column, then any other formulas applied to additional columns are based on their previous columns.

For example, the Blood Pressure.jmp sample data table contains blood pressure measurements taken on five subjects three times each day, over a period of three days. You want to find the log of each blood pressure (BP) column.

1. Select **Help > Sample Data Library** and open Blood Pressure.jmp.

Create nine new columns, one for each existing BP column.

2. Select **Cols > New Columns**.
3. Add nine columns.
4. Click **OK**.

Apply your original formula:

5. Right-click Column 12 1 and select **Formula**.
6. Select BP 8M.
7. Select **Transcendental > Log**.
8. Click **OK**.

   Column 12 1 now contains the log of the BP 8M column. You want the rest of the empty columns to contain the log of the remainder of the BP columns.
9. In the data table, select all of the new columns that you created, including the one with the original formula (columns 12 1 through 12 9).
10. Select **Cols > Standardize Attributes**.
11. In the Standardize Properties panel, click **Column Properties** and select **Formula**.
12. Select the check box next to **Substitute Column Reference**.
13. Leave BP 8M selected.

    You want that column reference to be substituted in the subsequent column formulas.
14. Click **OK**.

    Now all of the new columns are populated with the log of the BP columns, in the order in which they appear. Column 12 1 contains the log for BP 8M, Column 12 2 contains the log for BP 12M, and so on.

See "Copy and Replicate Formulas".

Assign a Preselected Analysis Role to a Column

You can assign an analysis role, such as *x*, *y*, weight, or frequency, to a selected column and save the role with the data table. When you do this and then run an analysis, JMP uses the preselected role to automatically fill in the role boxes in windows. Then you do not have to specify these roles each time you run an analysis. For example, you might want a column named height to take the *x* role in every analysis of that data table. To enforce the *x* role, you assign the preselected role of *x* to the column.

When you select Freq, the values in that column are what JMP uses as the frequency of the observation. If *n* is the value of the Freq variable for a given row, then that row is used in computations *n* times. If it is less than 1 or is missing, then JMP does not use it to calculate any analyses.

When you select Weight, the values in that column provide weights for each observation in the data table. The variable does not have to be an integer, but it is included only in analyses when its value is greater than zero.

JMP PRO When you select Validation, that column is specified as a Validation column in platforms that support a Validation column role.

To assign a preselected role to a column

1. Highlight the column.
2. Select **Cols > Preselect Role**.
3. Select a role: **No Role**, **X**, **Y**, **Weight**, **Freq**, or **Validation**.

After you select the appropriate roles, icons in the Columns panel signify what roles have been assigned. Click the icon to access a list of roles and select a different one. See "Icons Representing Column Characteristics and Properties".

Chapter 6

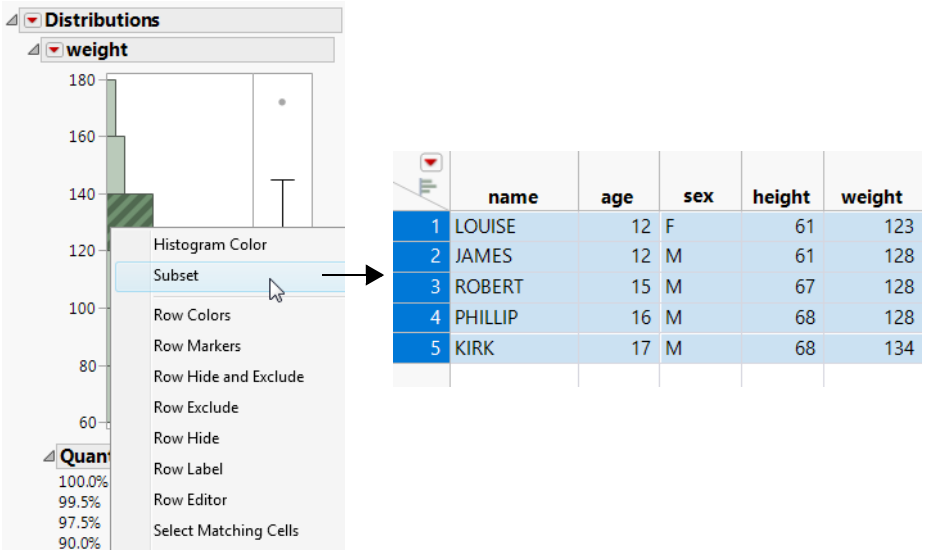
Reshape Your Data

Create Subsets, Sort Data, and More

This chapter covers the following tasks that you can perform on JMP data:

- create a new data table from a subset of rows and columns
- sort by any number of columns
- stack multiple columns into a single column
- split a column into two or more columns
- transpose rows and columns
- concatenate multiple tables end to end
- join two tables side by side
- virtually join tables without physically joining them
- update columns in a table with values from another table
- anonymize the data

Figure 6.1 Creating a Subset Data Table from a Report



	name	age	sex	height	weight
1	LOUISE	12	F	61	123
2	JAMES	12	M	61	128
3	ROBERT	15	M	67	128
4	PHILLIP	16	M	68	128
5	KIRK	17	M	68	134

Contents

Create a Subset Data Table . . . 337
Stratified Subsets . . . 339
Create a Subset Data Table from a Report . . . 339
Sort Data Tables . . . 340
Stack Columns . . . 343
Example of Stacking Columns into a Single Column . . . 345
Example of Stacking Columns into More Than One Column . . . 346
Split Columns . . . 348
Examples of Splitting Columns . . . 350
Transpose Rows and Columns . . . 352
Examples of Transposing Rows and Columns . . . 354
Concatenate Data Tables . . . 356
Example of Concatenating Data Tables . . . 357
Example of Concatenating Data Tables and Table Variables . . . 358
Join Data Tables . . . 359
Examples of Joining Data Tables . . . 362
Query and Join Data Tables with JMP Query Builder . . . 371
About Links to Data Tables in Queries . . . 374
Virtually Join Data Tables . . . 374
Example of Virtually Joining Columns . . . 376
Example of Virtually Joining Columns and Showing Row States . . . 379
Example of Merging Referenced Data in Virtually Joined Data Tables . . . 381
Example of a Link ID and a Link Reference on a Single Column . . . 384
Update Data Tables . . . 387
Example of Updating a Data Table . . . 390
Anonymize Data . . . 391

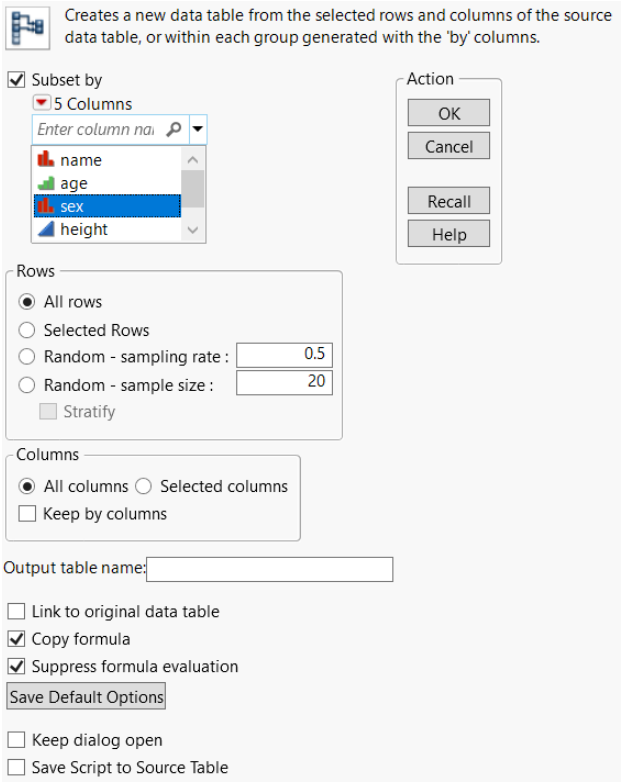
Create a Subset Data Table

You can produce a new data table that is a subset of all rows and columns, only highlighted rows and columns, or randomly selected rows from the active data table.

To create a subset

1. Select **Tables > Subset**.

Figure 6.2 The Subset Window



2. Specify the content that you want to subset. Select any combination of the following:
   - Subset by (the levels within selected columns)
   - Rows (all, selected, or random)
   - Columns (all or selected)

   In Figure 6.2, the sex column will be subset.
3. Customize your subset table further using the additional options.

4. Click **OK** to create the subset table.

Subset Options

Subset by Subsets by the levels of a column. Select **Subset by** and then select the columns that you want to categorize for the subset.

Consider the fact that many new data tables might be created. A new data table appears for each level of the column that you specified in the Subset window.

All Rows Creates a subset table that contains all rows from the active table.

Selected Rows Creates a subset table that contains only the selected rows from the active table. Selected by default.

Random - sampling rate Creates a subset table whose data is a random proportion of the active data table. Enter the proportion of the sample that you want in the text box. For example, if you want a random 50% of the data to be included in the new table, enter 0.5 in the text box.

Random - sample size Creates a subset table whose data is a random sample of the active data table. Enter the size of the sample that you want in the text box. For example, if you want 16 random rows to be included in the new table, enter 16 into the text box.

If you select a random sample that is the entire source table, the result is a random shuffle of the rows of the data table. If you specify columns to stratify, the result is a random shuffle of each of the rows for each group. See "Stratified Subsets".

All columns Creates a subset table that contains all columns from the active table. Selected by default.

Selected columns Creates a subset table that contains only the selected columns from the active table.

Keep by columns Retains the column that you subsetted by in the output data tables.

Output table name Specifies the name of the subset table.

Link to original data table Links the subset table to the original table. When you change values in one table, the other table is updated.

Copy formula Includes formulas from the original table in the output columns. Include all columns needed for the calculation of the formula. Selected by default.

Suppress formula evaluation Prevents JMP from evaluating columns' formulas when the new table is created. Selected by default.

Save Default Options Saves your current settings.

Note: Save Default Options only saves the settings for Selected Rows, Selected Columns, Linked to original data table, Copy formula, and Suppress formula evaluation.

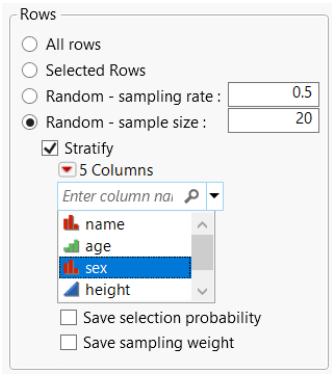
Keep dialog open Keeps the Subset window open after you click OK.

Save Script to Source Table Saves a script to the original data table that enables you to subset the data again using the same settings.

Stratified Subsets

If you specify a sample size and add stratification columns, the sample size represents the size per stratum, rather than the size of the whole subset.

Figure 6.3 Stratified Subsets



For stratified random samples with a specified sample size, two columns can be saved: Selection Probability and Sampling Weight. Check the corresponding check box to save these columns.

Create a Subset Data Table from a Report

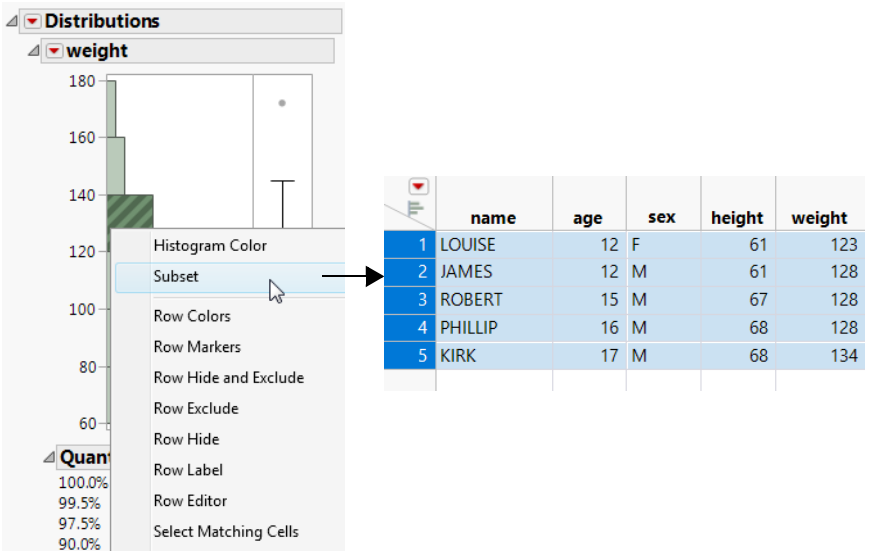
Histograms and Pareto plots produce linked subsets of a data table.

Use a Histogram

Once you have produced output that contains a histogram (by selecting **Analyze > Distribution**), you can use the histogram to create a new data table. The new data table contains the data in the histogram's highlighted bars.

To create a subset, double-click a highlighted bar. Or, right-click anywhere in the histogram and select **Subset** from the menu. The subset table appears.

Figure 6.4 Subset Created from a Histogram



Note: The subset data table contains a view of the data, not the actual data. You must explicitly save the table if you want a copy. Note that the saved copy of the data table is no longer linked to the source table.

Using a Pareto Plot

Once you have produced output that contains a Pareto plot (by selecting **Analyze > Quality and Process > Pareto Plot**), you can use the Pareto plot to create a new data table. The new data table contains the data in the Pareto plot's highlighted bars. To create a subset, double-click a highlighted bar.

Sort Data Tables

You can sort a JMP data table by columns in either ascending or descending order. By default, columns sort in ascending order. You can either create a new table that contains the sorted values, or you can replace the original table with the sorted table.

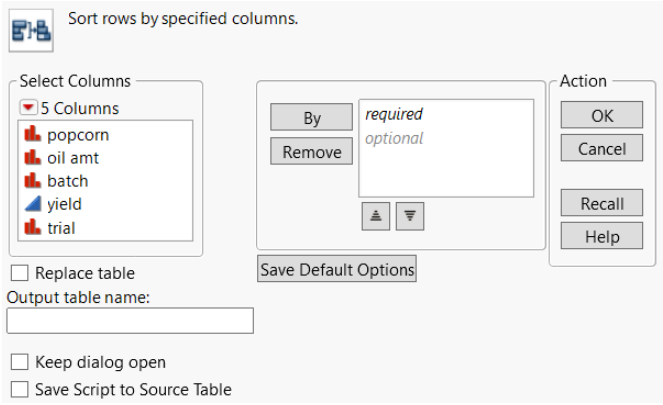
If columns contain value labels, sorting is based on the actual data values, not the value labels. (See "Value Labels".) However, the value labels are displayed in the sorted data table.

If your sorted column uses either the Value Order property or the List Check property, the column is sorted according to that order.

Example of Sorting Data Tables

1. Select **Help > Sample Data Library** and open Popcorn.jmp.
2. Select **Tables > Sort**.

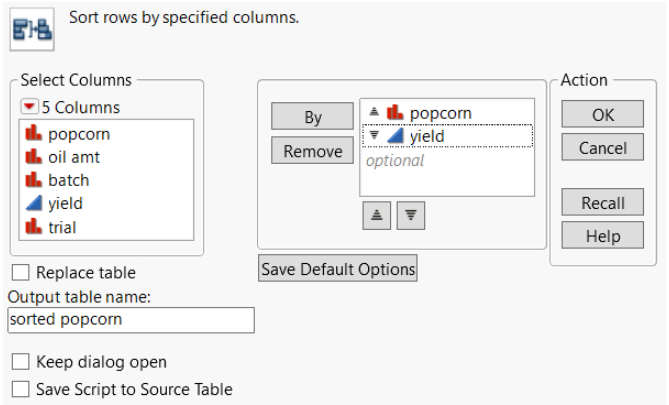
Figure 6.5 The Sort Window



3. Highlight the names of the columns that you want to sort by. For this example, select popcorn and yield.
4. Click **By** to add the columns to the sort list.

   The columns that you add to the list establish the order of precedence for sorting. The first column in the list is the major sort field. Each variable thereafter is sorted within the previous variable in the sort list. You can drag and drop within the By list to change the sort order.
5. Customize your sort further using the additional options. For this example, highlight yield and click the descending button.
6. Enter a name for the new sorted table in the box beside **Output table name**. For this example, enter *sorted popcorn*.

Figure 6.6 Completed Sort Window



7. Click **OK**.

Figure 6.7 Sorted in Ascending and Descending Order

	popcorn	oil amt	batch	yield	trial
1	gourmet	lots	small	18.0	1
2	gourmet	lots	small	16.0	2
3	gourmet	little	small	15.9	2
4	gourmet	little	small	12.1	1
5	gourmet	lots	large	9.8	2
6	gourmet	lots	large	9.2	1
7	gourmet	little	large	8.6	1
8	gourmet	little	large	8.2	2
9	plain	lots	small	10.6	1
10	plain	lots	large	10.4	1
11	plain	little	small	10.1	2
12	plain	little	small	9.9	1
13	plain	little	large	8.8	2
14	plain	lots	large	8.8	2
15	plain	little	large	8.2	1
16	plain	lots	small	7.4	2

Sort Options

Select Columns Filter Menu Contains options to search and filter through columns. See "Column Filter Menu".

Replace Table Replaces the original data table with the sorted table instead of creating a new table with the sorted values. This option is not available if there are any open report windows generated from the original table.

Output table name (Optional) Specifies the name of the sorted table.

Keep dialog open Keeps the Sort window open after you click **OK**.

Save Script to Source Table Saves a script to the original data table that enables you to sort the data again using the same settings.

By Adds the columns that you want to sort by. The columns that you add to the list establish the order of precedence for sorting. The first column in the list is the major sort field. Each variable thereafter is sorted within the previous variable in the sort list.

Remove Removes any highlighted columns.

Ascending and descending buttons ( ) You can change the ascending or descending list order of the values for the grouping variables. In the By variable list, select a variable and click the appropriate ascending or descending button. The icon beside the variable changes to indicate the sorting order.

Save Default Options Saves the current settings as the new default options.

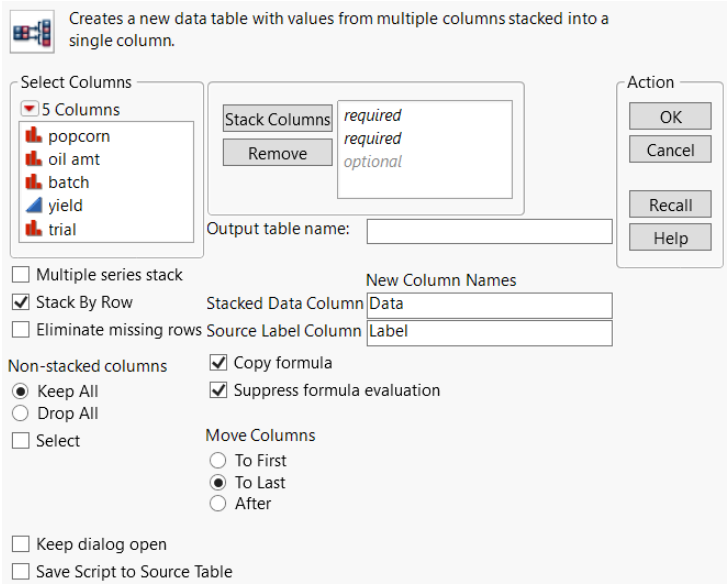
Stack Columns

You can rearrange your data table by stacking two or more columns into a single new column, preserving the values from the other columns. Or, you can stack a set of columns into multiple groups. The various ways that you can stack columns are explained in "Stack Options".

To stack columns, follow these steps:

1. Select **Tables > Stack**.

Figure 6.8 Stack Window



2. Highlight the names of the columns that you want to stack and click **Stack Columns**.
3. Customize your stacking further using the additional options.
4. Click **OK**.

Stack Options

Select Columns Filter Menu Contains options to search and filter through columns. See "Column Filter Menu".

Multiple series stack Stacks selected columns into two or more columns. Specify the number of columns into which you want the selected columns to be stacked by entering the number into the Number of Series box. This box appears when you check the box beside **Multiple series stack**.

Select the **Contiguous** option if the series consists of adjacent columns.

See "Example of Stacking Columns into More Than One Column".

Note: The order in which you add columns to the box on the right determines the group to which they belong.

Stack by Row Stacks columns by rows. Deselect the option to stack one column underneath another. Selected by default.

Eliminate missing rows Eliminates missing data from the new table. If Stack by Rows is checked also, only rows with *all* data missing are eliminated.

Non-stacked columns Includes or drops non-stacked columns from the new data table. Select one of these options:

Keep All Includes all of the non-stacked columns from the original table in the new table. Selected by default.

Drop All Omits the non-stacked columns from the new table.

Select Select the non-stacked columns that you want to include or drop in the new table.

Keep dialog open Keeps the Stack window open after you click **OK**.

Save Script to Source Table Saves a script to the original data table that enables you to stack the data again using the same settings.

Stack Columns Adds the columns that you want to stack.

Remove Removes any highlighted columns.

Output table name (Optional) Specifies the name of the new table.

Stacked Data Column Assigns a name to the column that will contain the data for the stacked columns. The default name is Data. Leave the box empty if you do not want this column to appear in the new table.

Source Label Column Assigns a name to the column that will contain the original table's column names. The default name is Label. Leave the box empty if you do not want this column to appear in the new table.

Copy formula Includes formulas from the original table in the output columns. Selected by default.

Suppress formula evaluation Prevents JMP from evaluating columns' formulas when the new table is created. Selected by default.

Move Columns Specifies where the stacked columns are moved to. To Last is the default value.

Example of Stacking Columns into a Single Column

A researcher has two columns in their data table representing yield, and they want to stack the two columns into a single column. (This new single column is called Data by default.)

1. Select **Help > Sample Data Library** and open Popcorn Trials.jmp.
2. Select **Tables > Stack**.
3. Select yield1 and yield2 and click **Stack Columns**.
4. Click **OK**.

Figure 6.9 Stacked Data Table

	popcorn	oil amt	batch	Label	Data
1	plain	little	large	yield1	8.2
2	plain	little	large	yield2	8.8
3	gourmet	little	large	yield1	8.6
4	gourmet	little	large	yield2	8.2
5	plain	lots	large	yield1	10.4
6	plain	lots	large	yield2	8.8
7	gourmet	lots	large	yield1	9.2
8	gourmet	lots	large	yield2	9.8
9	plain	little	small	yield1	9.9
10	plain	little	small	yield2	10.1
11	gourmet	little	small	yield1	12.1
12	gourmet	little	small	yield2	15.9
13	plain	lots	small	yield1	10.6
14	plain	lots	small	yield2	7.4
15	gourmet	lots	small	yield1	18.0
16	gourmet	lots	small	yield2	16.0

The Label column represents the Source Label Column that identifies the source of the data. Its values are the column names in the original table from which the stacked values originated.

Example of Stacking Columns into More Than One Column

Suppose that a researcher has data on blood pressure readings. The readings were taken over three days: Monday, Wednesday, and Friday. Three readings were taken each day, at 8am, 12pm, and 6pm.

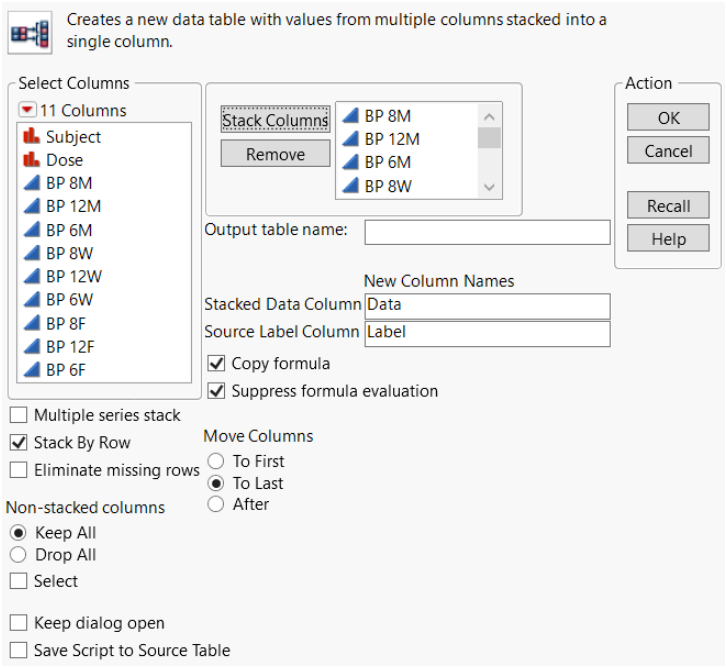
1. Select **Help > Sample Data Library** and open Blood Pressure.jmp.

   Each BP (blood pressure) column is delineated according to the date and time. The BP 8M column corresponds to readings that were taken at 8am on Monday. The BP 12W column corresponds to readings that were taken on 12pm on Wednesday, and so on. The researcher wants to stack all of the blood pressure columns into three columns that correspond to each day: Monday, Wednesday, and Friday.
2. Select **Tables > Stack**.
3. Select all of the BP readings and click **Stack Columns**.

   The order of the columns reflects how the columns in the series should be grouped.
4. Select **Multiple series stack**.
5. Next to Number of Series, type 3.
6. Because you want to stack the columns vertically, select **Contiguous**.
7. Rename the Stacked Data Column from Data to BP (for blood pressure).

8. Rename the Source Label Column from Label to Day.

Figure 6.10 Completed Stack Window



9. Click **OK**.

Figure 6.11 Stacked Data Table

	Subject	Dose	Day	BP	Day 2	BP 2	Day 3	BP 3
1	1	A	BP 8M	183	BP 8W	174	BP 8F	171
2	1	A	BP 12M	174	BP 12W	178	BP 12F	178
3	1	A	BP 6M	180	BP 6W	181	BP 6F	171
4	2	A	BP 8M	173	BP 8W	170	BP 8F	175
5	2	A	BP 12M	181	BP 12W	179	BP 12F	185
6	2	A	BP 6M	181	BP 6W	176	BP 6F	188
7	3	A	BP 8M	181	BP 8W	188	BP 8F	183
8	3	A	BP 12M	189	BP 12W	175	BP 12F	183
9	3	A	BP 6M	177	BP 6W	182	BP 6F	180
10	4	A	BP 8M	181	BP 8W	176	BP 8F	183
11	4	A	BP 12M	177	BP 12W	173	BP 12F	187
12	4	A	BP 6M	182	BP 6W	184	BP 6F	183
13	5	A	BP 8M	184	BP 8W	172	BP 8F	170

In the stacked data table, note the following:

- The first Day column represents Monday.
- The Day 2 column represents Wednesday.

- The Day 3 column represents Friday.

Split Columns

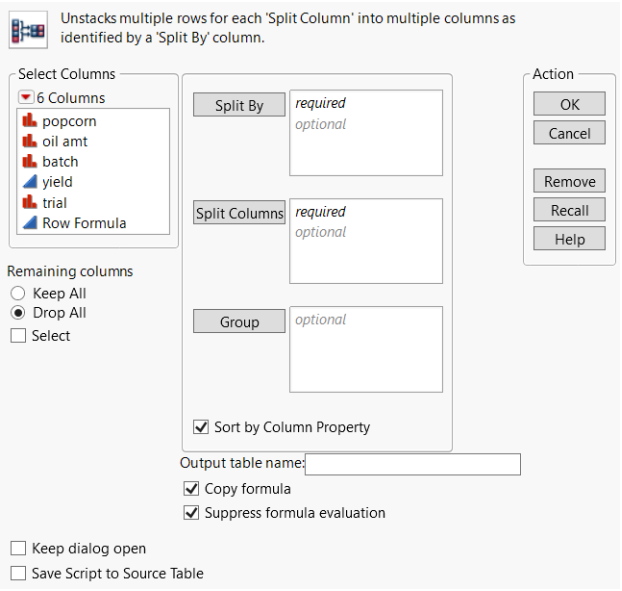
You can create a new data table from the active table by splitting one column into several new columns. This column is split according to the values found in another column, referred to as the **Split By** column. You can also split columns according to the values of one or more grouping variables.

Note: If the split is on a categorical column that contains a missing value, the column name for the missing category is "." for a numeric column.

To split columns, follow these steps:

1. Select **Tables > Split**.

Figure 6.12 Split Window



2. Highlight the names of the column or columns that you want to split and click **Split Columns**.
3. Highlight a column whose values you want to use as the basis for splitting the column.
4. Click **Split By**.
5. Customize your splitting further using the additional options.

6. Click **OK**.

Split Options

Select Columns Filter Menu Contains options to search and filter through columns. See "Column Filter Menu".

Keep All Includes all columns in the new table.

Drop All Includes only columns used in the split in the new table. Selected by default.

Select Selects which columns to keep in the new table.

Keep dialog open Keeps the Split window open after you click **OK**.

Save Script to Source Table Saves a script to the original data table that enables you to split the data again using the same settings.

Split By Adds the column whose values you want to use as the new column names, and as the basis for splitting the column.

Split Columns Adds the column or columns that you want to split.

Group Specifies a Group variable when you want your data to be split within each group of the selected variable. Each group results in a row in the output table. You must also specify the required variables, Split By, and Split Columns.

Note: If the variable that you want to group by contains unequal groups or is in a random order, specifying it as the Group variable ensures that your data is restructured properly, and any missing values are assigned in the appropriate places.

Sort by Column Property Sorts the order of the output columns by the Value Order column property. If the column does not have a Value Order property, and the data has an implied order (such as days or months), the implied order is applied.

Output table name (Optional) Specifies the name of the new table.

Copy formula (Appears if there is a formula in the data table.) Includes formulas from the original table in the output columns.

Suppress formula evaluation (Appears if there is a formula in the data table.) Prevents JMP from evaluating columns' formulas when the new table is created.

Examples of Splitting Columns

This section contains two examples using the Split command:

- In the first example, one column is split by a second column. See "Split a Column: Basic Example".
- The second example uses a Group variable. See "Split a Column: Grouping Rows Example".

Split a Column: Basic Example

In the Popcorn.jmp data table, the data in the trial column shows that there are two trials, 1 and 2. In this example, split the yield column into two new columns: one for trial 1 and one for trial 2.

1. Select **Help > Sample Data Library** and open Popcorn.jmp.
2. Select **Tables > Split**.
3. Select the yield column and click **Split Columns**.
4. Select the trial column and click **Split By**.
5. Under **Remaining columns**, select **Keep All**.

   The default is Drop All, which omits any columns that are not in the Split By, Split Columns, or Group fields. Selecting **Keep All** includes these columns in the new table.
6. (Optional) Type Yield column split by Trial column in the Output table name field.
7. Click **OK**.

   A new data table is created (Figure 6.13). Notice the following:
   - The yield and trial columns are gone.
   - The data table has two new columns, named after the unique values (1 and 2) from the original trial column.
   - The values from the original yield column are now split into the new columns named 1 and 2.
   - The columns other than trial and yield are exactly the same as they were in the original table.
8. (Optional) Rename the new columns to give them meaningful names. For example, rename 1 to yield (trial 1) and rename 2 to yield (trial 2).

Figure 6.13 New Table Created By Splitting yield Column by trial Column

	popcorn	oil amt	batch	1	2
1	plain	little	large	8.2	8.8
2	gourmet	little	large	8.6	8.2
3	plain	lots	large	10.4	8.8
4	gourmet	lots	large	9.2	9.8
5	plain	little	small	9.9	10.1
6	gourmet	little	small	12.1	15.9
7	plain	lots	small	10.6	7.4
8	gourmet	lots	small	18.0	16.0

Split a Column: Grouping Rows Example

The Drug Measurements.jmp sample data table contains measurements of three different drugs (a, b, and c) administered to 12 different subjects. You want to split the measurement into different columns, one for each drug type. You also want to group the measurements by subject.

1. Select **Help > Sample Data Library** and open Drug Measurements.jmp.
2. Select **Tables > Split**.
3. Select Drug Type and click **Split By**.
4. Select Measurement and click **Split Columns**.

   Notice that the Subject variable contains unequal groups. Most of the subjects were given all three drugs, but subject 2 was given only one drug, and subjects 7 and 12 were given only two drugs. In this situation, to ensure that the correct measurements are associated with the correct subject, specify Subject as the Group variable.
5. Select Subject and click **Group**.
6. Click **OK**.

Figure 6.14 Drug Measurements.jmp Split by a Grouping Variable

	Subject	a	b	c
1	1	91	99	91
2	2	•	96	•
3	3	95	99	93
4	4	97	93	96
5	5	95	100	94
6	6	92	96	95
7	7	93	96	•
8	8	100	94	92
9	9	98	99	97
10	10	91	96	99
11	11	98	100	93
12	12	•	95	95

You can see that the appropriate missing values appear for subjects 2, 7, and 12.

Transpose Rows and Columns

You can create a new JMP table that is a transposed version of the active data table. The columns of the active table are the rows of the new table, and its rows are the new table's columns.

When you transpose columns, you do the following:

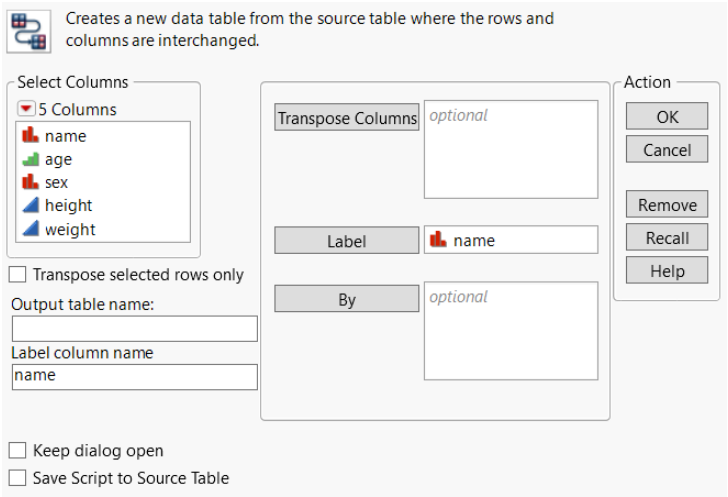
- Select the columns to be transposed.
- Specify a "label" column, from which the new columns get their names (optional).
- Specify "by" columns, which tells JMP to transpose data within groups (optional).

Note: Columns that you want to transpose must have the same data type. Also, if columns contain value labels, transposing uses the actual data values, not the value labels. (See "Value Labels".)

To transpose rows and columns, follow these steps:

1. Open a data table that contains the rows and columns that you want to transpose.
2. Select **Tables > Transpose**.

Figure 6.15 Transpose Window



3. Highlight the column name(s) you want to transpose in the **Select Columns** box on the left.
4. Click **Transpose Columns**.
5. (Optional) Customize your transposed table further using the additional options.
6. Click **OK**.

Transpose Options

Select Columns Filter Menu Contains options to search and filter through columns. See "Column Filter Menu".

Transpose selected rows only Transposes only rows that are currently highlighted in the active table.

Output table name (Optional) Specifies the name of the subset table.

Label column name (Applicable only if you have specified a Label column.) Specifies an alternative name for the Label column. Otherwise, the default column name is Label.

Keep dialog open Keeps the Transpose window open after you click **OK**.

Save Script to Source Table Saves a script to the original data table that enables you to transpose the data again using the same settings.

Transpose Columns Adds the columns that you want to transpose.

Label Uses the data from a column in the original table as the column names in the new table. Follow these steps:

1. Highlight a column from the Select Columns box on the left.
2. Click **Label**. The column name appears in the Label box.

The default column name is Label. You can specify an alternative name for the column using the **Label column name** option. Only one column is created for each distinct value in the label column. Therefore, if there are duplicate values in the label column, JMP creates only one column for the duplicated value using the value from the last duplicated row.

By Organizes the transposed columns into groups based on the columns that you put into the By box. Follow these steps:

1. Highlight the column name(s) in the Select Columns box whose values you want to see as a group.
2. Click **By**.

Table 6.1 describes the rules that apply to transposing.

Table 6.1 Rules for Transposing

If	Then
The original table has columns but no rows	The new table contains one column that lists those column names.
The original table has one column and it is assigned to Label	Its values become the column names in the transposed table.
The original table has multiple columns and contains a label column	JMP automatically inserts the label column into the Label box when the window appears. You can remove this column if you do not want it to appear.
There is no label column in the original table	The column names in the transposed table are Row 1, Row 2, …, Row n where *n* is the number of rows in the original table.

Examples of Transposing Rows and Columns

This section contains three examples: a simple example of transposing, an example using the Label option, and an example using a By group.

Simple Example of Transposing

1. Select **Help > Sample Data Library** and open Materials1.jmp.
2. Select **Tables > Transpose**.
3. Select plastic, tin, and gold and click **Transpose Columns**.

4. Click **OK**.

Figure 6.16 Simple Transposed Table

	Label	Row 1	Row 2
1	plastic	1	4
2	tin	2	5
3	gold	3	6

The original table in Figure 6.16 has two rows and three continuous columns called plastic, tin, and gold. The transposed table has a row for each of the three columns in the original table, and columns named Row 1 and Row 2 for the original table's rows. The additional column called Label has the column names (plastic, tin, and gold) from the original table as values.

Example Using the Label Option

1. Select **Help > Sample Data Library** and open Materials2.jmp.
2. Select **Tables > Transpose**.
3. Select plastic, tin, and gold and click **Transpose Columns**.
4. Select item and click **Label**.
5. Click **OK**.

Figure 6.17 Transpose with a Label

	Label	nails	hooks
1	plastic	1	4
2	tin	2	5
3	gold	3	6

The values from the item column in the original table are used as column labels in the transposed table.

Example Using a By Group

1. Select **Help > Sample Data Library** and open Animals Subset.jmp.
2. Select **Tables > Transpose**.
3. Select subject and miles and click **Transpose Columns**.
4. Select season and click **Label**.
5. Select species and click **By**.
6. Click **OK**.

Figure 6.18 Transpose Using a By Group

	species	Label	fall	winter	spring	summer
1	COYOTE	subject	1	1	1	1
2	COYOTE	miles	4	2	7	8
3	FOX	subject	3	3	3	3
4	FOX	miles	4	3	6	2

The transposed table contains values that have been transposed in groups.

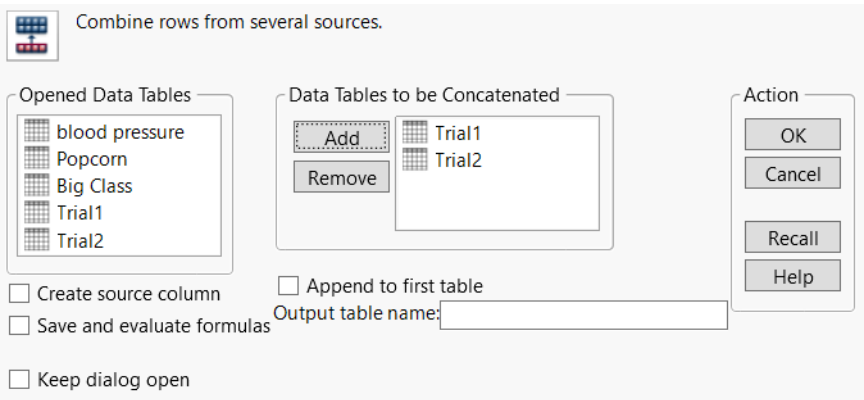
Concatenate Data Tables

When you concatenate data tables in JMP, you combine rows from two or more data tables. You can create a new data table *or* you can append rows to the first data table. A column name might be the same in the data tables that you want to concatenate. If so, then the column in the new data table lists the values from all of the data tables in the order of concatenation. If the two original data tables have columns with different names, those columns are included in the new data table showing missing values.

To concatenate two data tables with the same column names, follow these steps:

1. Select **Tables > Concatenate**.

Figure 6.19 Concatenate Window



2. Highlight the names of the data tables that you would like to combine, and click **Add**.

   You can concatenate as many data tables as you choose, and you can also add the same data tables multiple times. The number of rows in the new data tables is the sum of the number of rows in all the data tables.

3. (Optional) Click the **Save and evaluate formulas** choice to request that JMP include all formulas.

   If you do not select this option, no formulas are included in the new data table.

   Note: Columns with the same name can have different formulas. The formula from the first data table that contains a formula for that column is saved in the concatenated data table. This situation can occur when more than two tables are being concatenated and the second or third on has a formula for the column in question.

4. (Optional) Click the **Create source column** choice to add a column called Source Table to the new data table.

   This column identifies the name of the source data table in the corresponding rows.

5. (Optional) Select the **Append to first table** choice to append rows to the data table listed first in the **Data Tables to be Concatenated** field. This option is an alternative to creating a new data table.

6. (Optional) Enter a name for the new data table in the **Output table name** field.

   If you do not enter a name, JMP names the data table Untitled# (for example, Untitled1). The **Output table name** field is not available if you selected the **Append to first table** choice.

7. Click **OK**.

Example of Concatenating Data Tables

Suppose that you want to concatenate two data tables (Trial1 and Trial2) into a new data table.

1. Select **Help > Sample Data Library** and open Trial1.jmp and Trial2.jmp.
2. From the Trial1.jmp table, select **Tables > Concatenate**.
3. In the **Opened Data Table** list, select Trial2 and click **Add**.
4. Click **OK**.

The data tables combine into a new concatenated table with all of the rows from the first data table followed by all of the rows from the second data table.

Figure 6.20 Result of Concatenating Two Data Tables

	popcorn	oil amt	batch	yield
1	plain	little	large	8.2
2	gourmet	little	large	8.6
3	plain	lots	large	10.4
4	gourmet	lots	large	9.2
5	plain	little	small	9.9
6	gourmet	little	small	12.1
7	plain	lots	small	10.6
8	gourmet	lots	small	18.0
9	plain	little	large	8.8
10	gourmet	little	large	8.2
11	plain	lots	large	8.8
12	gourmet	lots	large	9.8
13	plain	little	small	10.1
14	gourmet	little	small	15.9
15	plain	lots	small	7.4
16	gourmet	lots	small	16.0

Concatenated data tables always contain a column for every column name found in the original data tables. However, if the column names do not match exactly, they are not merged. For example, if the yield column was instead named yield1 and yield2, a separate column would be created for each in the concatenated data table.

When you concatenate two or more data tables containing table variables, separate columns are created for each table variable. This ensures that important distinctions are not lost when concatenating data tables.

Note: Columns are not created for table variables that begin with the name Notes.

Example of Concatenating Data Tables and Table Variables

For example, suppose that two cancer trials were conducted at two different hospitals. One of the trials' data is in the Cancer1.jmp data table, and the other trial's data is in the Cancer2.jmp data table.

To consolidate the data and the variables into one table, follow these steps:

1. Select **Help > Sample Data Library** and open Cancer1.jmp and Cancer2.jmp.

   Notice that there are two distinct table variables: Dosage Amount and Location. In the concatenated table, columns are created for these two table variables.
2. From the Cancer1.jmp data table, select **Tables > Concatenate**.
3. Select Cancer2 and click **Add**.

4. Click **OK**.

Figure 6.21 Data and Variables Concatenated

	Time	Cell Type	Treatment	Prior	Dosage Amount	Location
1	3	Adeno	Standard	No	150	Roanoke
2	7	Adeno	Test	No	150	Roanoke
3	8	Adeno	Standard	Yes	150	Roanoke
4	8	Adeno	Test	No	150	Roanoke
5	12	Adeno	Standard	Yes	150	Roanoke
6	18	Adeno	Test	Yes	150	Roanoke
7	19	Adeno	Test	No	150	Roanoke
8	24	Adeno	Test	No	150	Roanoke
9	3	Adeno	Standard	No	250	Asheville
10	7	Adeno	Test	No	250	Asheville
11	8	Adeno	Standard	Yes	250	Asheville
12	8	Adeno	Test	No	250	Asheville
13	12	Adeno	Standard	Yes	250	Asheville
14	18	Adeno	Test	Yes	250	Asheville
15	19	Adeno	Test	No	250	Asheville
16	24	Adeno	Test	No	250	Asheville

The data and the variables are concatenated. The variables appear as columns in the concatenated table. The notes from each data table are added to the new data table as table variables.

Join Data Tables

You can combine two data tables into one new table by selecting **Tables > Join**. For an overall description of how to join two data tables, see "To join two data tables into a new data table, follow these steps:". Tables can be joined in three different ways:

- By combining them according to row number. See "Example of Joining by Row Number".
- In a Cartesian fashion, where you form a new table consisting of all possible combinations of the rows from two original tables. See "Examples of a Cartesian Join".
- By matching the values in one or more columns that exist in both data tables, or in a single data table. See "Examples of Joining By Matching Columns".

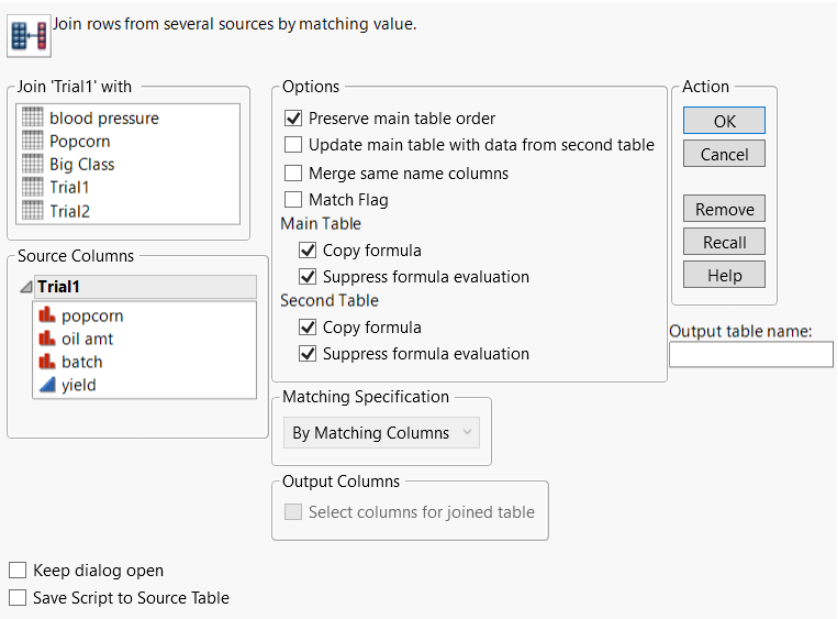
Note: The JMP Query Builder option in the Tables menu provides an option to query data before performing a simple join. See "Query and Join Data Tables with JMP Query Builder".

To join two data tables into a new data table, follow these steps:

1. Open the two data tables that you want to join.
2. Select **Tables > Join**.

In the window that appears, the names of all open tables appear below **Join...with**.

Figure 6.22 The Join Window



3. In the **Join...with** box, select the table to join with the active table.
4. From the Matching Specification area, select the option that specifies how to join the tables.
5. Enter the name of the new table in the text box beside **Output table name**.
6. (Optional) Customize the join procedure further using the additional options.
7. Click **OK** to create the joined data table.

Note: Joining on matching columns that have value labels correctly joins with the column's data values, not the value labels.

Join Options

Keep dialog open Keeps the Join window open after you click **OK**.

Save Script to Source Table Saves a script to the original data table that enables you to join the data again using the same settings.

Preserve main table order Maintains the order of the original data table in the joined table, instead of sorting by the matching columns. Selected by default.

Update main table with data from second table Data in the original table are updated with data in columns that have the same names. The results are displayed in a new data table.

Note the following:

- JMP does not replace data with missing values.
- The output table uses the same columns as the original table. Thus, when you use **Update main table with data from second table**, **Select Columns for joined table** is not applicable.
- The **Update main table with data from second table** option is available only when joining by row number or by matching columns.

Merge same name columns Data from the second table replaces the data of the same name columns in the original table. Note that missing values in the first table are replaced by nonmissing values in the second.

If you are matching by column, Match Flag is selected when you select **Merge same name columns**. The new joined table contains a nominal column named Match Flag:

- If Main appears in this column, the data originated from the first (active) table.
- If With appears in this column, the data originated from the second table.
- If Both appears in this column, the data was found in both the first and second tables.

Match flag Determines whether the Match Flag column is created when you are matching by column. Selected automatically when you select **Merge same name columns**.

Copy formula (Main Table and Second Table) Includes formulas from the main table and/or the second table in the output columns. Selected by default.

Suppress formula evaluation (Main Table and Second Table) Prevents JMP from evaluating columns' formulas during the creation of the new table. Selected by default.

By Matching Columns To join rows, select columns in both tables whose values and data types match. Follow these steps:

1. Highlight a column name from each list in the **Source Columns** area. The first highlighted column in the top list pairs with the first highlighted column in the bottom list, the second columns are paired, and so on. Rows join only if values and data types match for all the column pairs.
2. Click **Match**. The selected pair of columns appears in the **Match columns** box. Matching columns do not have to have the same names and do not have to be in the same relative column position in both tables.
3. (Optional) To include only the first match found, check the boxes associated with **Drop multiples** in both tables. Only the first match found is written to the new table. If you specify this option for one table, the first match value is joined with all matches in the

other table. If you do not check the boxes associated with **Drop multiples** in either table, a Cartesian join is performed within each group of matching column values.

4. (Optional) To include all rows from the data table, even when there is no matching value, check the boxes associated with **Include non-matches**. You can specify this option for either or both data tables being joined.

By Row Number Joins the two tables side by side.

Cartesian Join Joins two tables using a Cartesian fashion, where it forms a new table consisting of all possible combinations of the rows from two original tables. JMP crosses the data in the first table with the data in the second to display all combinations of the values in each set.

Select Columns for joined table Selects a subset of columns from either table for inclusion in the output table. Follow these steps:

1. In the Source Columns area, highlight the columns from each table that you want to include in the new table.
2. Click **Select** in the Output Columns area.

Output table name Specifies the name of the joined table.

Examples of Joining Data Tables

The following sections provide examples using the Join command.

Example of Joining by Row Number

Joining tables by row number joins the two tables side by side. The new table has all of the columns from both tables, unless you specify to include only certain columns.

To join tables with an unequal number of rows

If the two tables that you want to join have an unequal number of rows, the new table contains values for the rows found in both tables.

1. Select **Help > Sample Data Library** and open Species1.jmp and Species2.jmp.

   Notice that the Species1.jmp table has two rows, and the Species2.jmp table has four rows.
2. From the Species1.jmp table, select **Tables > Join**.
3. In the **Join...with** box, select Species2.
4. From the Matching Specification area, select **By Row Number**.
5. Click **OK**.

Figure 6.23 Joined Tables by Row Number

	species	season
1	FOX	spring
2	COYOTE	summer
3		fall
4		winter

If one table with two rows is joined with a table with four rows, then the new table contains four rows.

To join columns with the same name

If the two tables have column names that are the same, the names of these columns in the new table appear as "column name of table name." For example, suppose that you want to combine the eight rows from the Trial1.jmp and Trial2.jmp data tables shown in Figure 6.24 into a single table. You want to combine them so that the new table contains all of the columns from both tables.

1. Select **Help > Sample Data Library** and open Trial1.jmp andTrial2.jmp.
2. From the Trial1.jmp data table, select **Tables > Join**.
3. In the **Join...with** box, select Trial2.
4. From the Matching Specification menu, select **By Row Number**.
5. Click **OK**.

Figure 6.24 Original Tables and the Joined Table

	popcorn of Trial1	oil amt of Trial1	batch of Trial1	yield of Trial1	popcorn of Trial2	oil amt of Trial2	batch of Trial2	yield of Trial2
1	plain	little	large	8.2	plain	little	large	8.8
2	gourmet	little	large	8.6	gourmet	little	large	8.2
3	plain	lots	large	10.4	plain	lots	large	8.8
4	gourmet	lots	large	9.2	gourmet	lots	large	9.8
5	plain	little	small	9.9	plain	little	small	10.1
6	gourmet	little	small	12.1	gourmet	little	small	15.9
7	plain	lots	small	10.6	plain	lots	small	7.4
8	gourmet	lots	small	18.0	gourmet	lots	small	16.0

A column name can be the same in the two original tables. The output column name is then qualified by the source table name. For example, the column names in the new table appear as <variable name> of table name.

To join only specified columns

Suppose that you do not want all of the columns from the original data tables to be in the joined table:

1. Select **Help > Sample Data Library** and open Trial1.jmp and Trial2.jmp.
2. From the Trial1.jmp data table, select **Tables > Join**.
3. In the **Join...with** box, select Trial2.
4. From the Matching Specification menu, select **By Row Number**.
5. Click **Select columns for joined table** to specify the subset of columns that you want to include.
6. In the Source Columns list, select popcorn and yield from the Trial1 list and select yield from the Trial2 list.

   Because identical data exists in the popcorn column of both tables, you need to select only one column.
7. Click **Select**.
8. Click **OK**.

Figure 6.25 Joining Only Specified Columns

	popcorn	yield of Trial1	yield of Trial2
1	plain	8.2	8.8
2	gourmet	8.6	8.2
3	plain	10.4	8.8
4	gourmet	9.2	9.8
5	plain	9.9	10.1
6	gourmet	12.1	15.9
7	plain	10.6	7.4
8	gourmet	18.0	16.0

Examples of a Cartesian Join

When doing a Cartesian join, JMP joins two tables in a Cartesian fashion: the new table consists of all possible combinations of the rows from two original tables. This creates cases in the output table where there are one case for each combination of column values.

Simple Example of a Cartesian Join

1. Select **Help > Sample Data Library** and open Species1.jmp and Species2.jmp.
2. From the Species1.jmp table, select **Tables > Join**.
3. In the **Join...with** box, select Species2.
4. From the Matching Specification menu, select **Cartesian Join**.

5. Click **OK**.

Figure 6.26 Joining Tables Using Cartesian Join

	species	season
1	FOX	spring
2	FOX	summer
3	FOX	fall
4	FOX	winter
5	COYOTE	spring
6	COYOTE	summer
7	COYOTE	fall
8	COYOTE	winter

The data in Species1.jmp is crossed with the data in Species2.jmp to produce the joined table, which shows all combinations of the values in each set.

Complex Example of a Cartesian Join

In this example, use the **Tables > Join** command twice:

- The first join combines the Oil Amount.jmp table with the Batch.jmp table using the **Cartesian** option.
- The second join combines the resulting table (Cartesian oil amount + batch) with the Popcorn Type.jmp table and produces a final table with all tables joined.

1. Select **Help > Sample Data Library** and open Oil Amount.jmp, Batch.jmp, and Popcorn Type.jmp.
2. From the Oil Amount.jmp table, select **Tables > Join**.
3. In the **Join...with** box, select Batch.
4. From the Matching Specification menu, select **Cartesian Join**.
5. Under **Output table name**, type Oil Amount and Batch.
6. Click **OK**.

Figure 6.27 Oil Amount and Batch Joined Table

	oil amount	batch
1	little	small
2	little	large
3	lots	small
4	lots	large

The joined table contains all of the columns from the Oil Amount.jmp and Batch.jmp tables. Add the Popcorn Type.jmp columns:

7. From the Oil Amount and Batch table that you just created, select **Tables > Join**.
8. In the **Join...with** box, select Popcorn Type.
9. From the Matching Specification menu, select **Cartesian Join**.
10. Click **OK**.

Figure 6.28 Oil Amount and Batch Joined with Popcorn Type

	oil amount	batch	popcorn
1	little	small	gourmet
2	little	small	plain
3	little	large	gourmet
4	little	large	plain
5	lots	small	gourmet
6	lots	small	plain
7	lots	large	gourmet
8	lots	large	plain

The final table contains all of the columns from all three original tables. Keep in mind that the number of rows produced by a Cartesian join is the product of the number of rows in the original tables.

Examples of Joining By Matching Columns

When you join data tables by matching columns, JMP aligns the rows in the two tables by matching column values.

Caution: If the matched columns do not result in unique rows, proceed carefully.

Notes:

- The columns used for matching must have the same data type (numeric, character, or row state).
- You can join a data table to itself, in order to remove duplicate values or rows from the data table.
- You can use a virtual join as an alternative to creating a new data table. See "Virtually Join Data Tables".

To join tables with a unique identifier

You have two files of information about hot dogs. The second file contains one column, Size, that is missing from the first file. Use Join with a matching column to combine the data tables.

1. Select **Help > Sample Data Library** and open Hot Dogs.jmp and Hot Dogs2.jmp.

   The Product Name column is a unique identifier for the rows in your data tables. Each table has 54 unique product names.
2. From the Hot Dogs data table, select **Tables > Join.**
3. In the Join 'Hot Dogs' with list, select Hot Dogs2.
4. Select **By Matching Columns** in the Matching Specification area.
5. From the Hot Dogs and Hot Dogs2 lists, select Product Name.
6. Click **Match**.
7. In the Options, select **Merge same column names**. When the merge columns is used the Match Flag is automatically selected.
8. Select **Keep dialog open** in the lower left corner of the window so that you can easily return to the Join window in the case that the joined data table is not as you expected.
9. Click Apply.

Figure 6.29 Joined Table by Matching Columns

	Match Flag	Product Name	Type	Taste	$/oz	$/lb Protein	Calories	Sodium	Protein/Fat	Size
1	Both	Happy Hill Supers	Beef	Bland	0.11	14.23	186	495	1	Jumbo
2	Both	Georgies Skinless...	Beef	Bland	0.17	21.70	181	477	2	Jumbo
3	Both	Special Market's ...	Beef	Bland	0.11	14.49	176	425	1	Jumbo
4	Both	Spike's Beef	Beef	Medium	0.15	20.49	149	322	1	Regular
5	Both	Hungry Hugh's J...	Beef	Medium	0.10	14.47	184	482	1	Regular
6	Both	Great Dinner Beef	Beef	Medium	0.11	15.45	190	587	1	Regular
7	Both	RJB Kosher Beef	Beef	Medium	0.21	25.25	158	370	2	Regular
8	Both	Wonder Kosher S...	Beef	Medium	0.20	24.02	139	322	2	Regular
9	Both	Happy Fats Jumb...	Beef	Medium	0.14	18.86	175	479	1	Regular
10	Both	Midwest Beef	Beef	Medium	0.14	18.86	148	375	1	Regular
11	Both	General Kosher B...	Beef	Medium	0.23	30.65	152	330	1	Regular

The resulting table now includes the Size column that was in the Hot Dogs2 table. The Match Flag column indicates that the data originated from both tables for all rows.

To join tables with different numbers of rows and different column names

Suppose that Sarah and Joe are performing a popcorn experiment. They are popping different types of popcorn (gourmet and plain) in different amounts of oil. They are recording the amount (yield) of popcorn that is produced. Sarah gave you the first trial data in a file named Trial1.jmp. Joe gave you the second trial data in a file named Little.jmp. You want to combine the two tables into one table.

1. Select **Help > Sample Data Library** and open Trial1.jmp and Little.jmp.
2. From the Trial1.jmp table, select **Tables > Join**.
3. In the **Join...with** box, select Little.

You can see that three of the columns (popcorn, oil amt/oil, and batch) contain the same values in both tables. Identify these columns as matches. Also, because Sarah and Joe gave the oil and oil amt columns different names, you can tell JMP that oil amt and oil match.

4. Deselect **Preserve main table order**.

   The joined table is sorted by matching columns, not by the order of data in Trial1.jmp,

5. Select **By Matching Columns** in the Matching Specification area.
6. From the Trial1 list, select popcorn, oil amt, and batch.
7. From the Little list, select popcorn, oil, and batch.
8. Click **Match**.

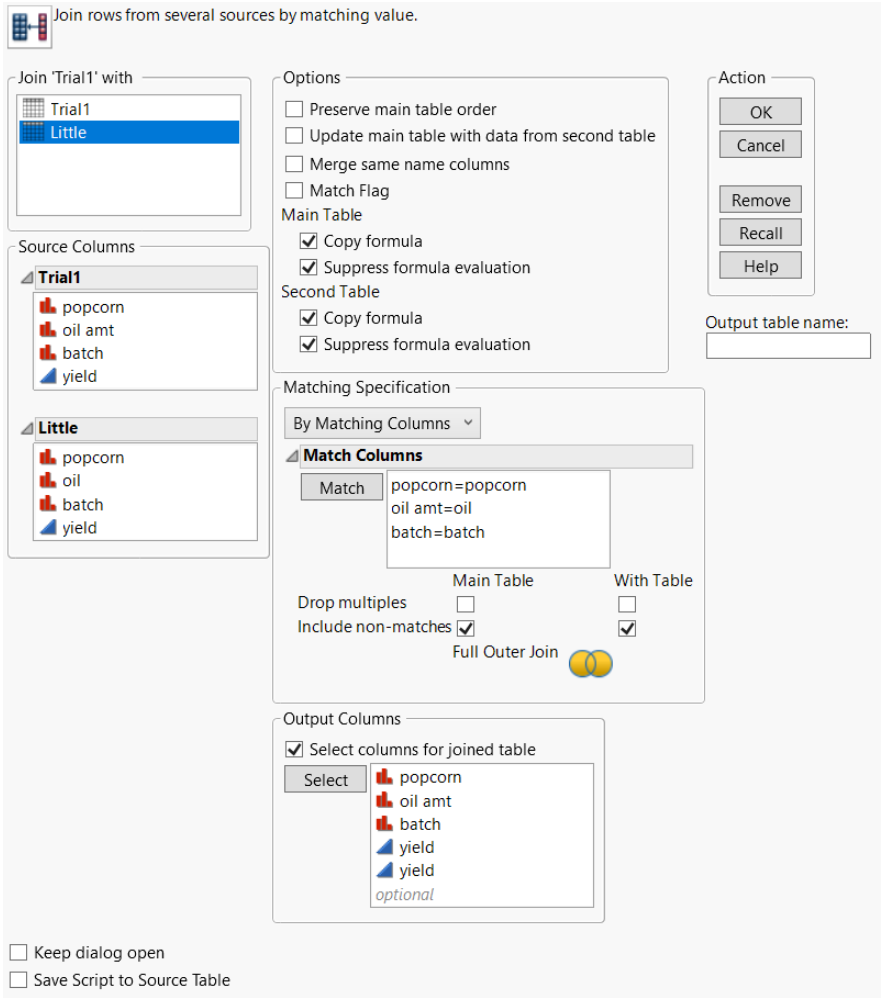
   Looking at the two data tables, you can see that they have different numbers of rows. Trial1.jmp has values for eight experimental conditions, and Little.jmp has values for only four of those conditions. Sarah completed her experiment, but Joe only partially completed his experiment. You want the joined table to contain all of the rows in Trial1.jmp. even if that row in the Little.jmp table contains a missing value.

9. Select the **Include non-matches** boxes for both tables.

   In the joined table, you want only one column for popcorn, one column for oil, and one column for batch. However, you want two columns for yield: one representing the yield from Trial1.jmp, and another representing the yield from Little.jmp.

10. Select the box beside **Select columns for joined table**.
11. From the Trial1 list, select all of the columns.
12. Click **Select**.
13. From the Little list, select yield.
14. Click **Select**.

Figure 6.30 Completed Join Window



15. Click **OK**.

Figure 6.31 Trial1.jmp and Little.jmp Joined

	popcorn	oil amt	batch	yield of Trial1	yield of Little
1	gourmet	little	large	8.6	8.2
2	gourmet	little	small	12.1	15.9
3	gourmet	lots	large	9.2	•
4	gourmet	lots	small	18.0	•
5	plain	little	large	8.2	8.8
6	plain	little	small	9.9	10.1
7	plain	lots	large	10.4	•
8	plain	lots	small	10.6	•

The joined table is sorted by the matching columns. Note that the yield column from the Little.jmp table (Yield of Little) has missing values indicating no matching values with the Trial1.jmp table.

To join a table to itself (to remove duplicate entries)

1. Select **Help > Sample Data Library** and open Coffee Shop Purchases.jmp.

   You can see that some of the customers had the same drink on the same date. You want to consolidate these duplicate rows using **Join**.
2. Select **Tables > Join**.
3. In the **Join...with** box, select Coffee Shop Purchases.
4. Select **By Matching Columns** in the Matching Specification area.
5. From both Coffee Shop Purchases lists, select all three columns: Date, Customer, and Beverage.
6. Click **Match**.
7. Select the **Drop multiples** boxes for both tables (the Main Table and the With Table).
8. Type Coffee Shop Purchases Final for the **Output table name**.
9. Click **OK**.

Figure 6.32 Original and Joined Coffee Shop Purchases Data Tables

	Date	Customer	Beverage
1	01/05/2013	Jeff	Espresso
2	01/05/2013	Mike	Chai tea
3	01/05/2013	Sergio	Latte
4	01/05/2013	Mike	Espresso
5	01/05/2013	Mike	Espresso
6	01/05/2013	Lisa	Espresso
7	01/05/2013	Lisa	Espresso
8	02/13/2013	Lisa	Chai tea
9	02/13/2013	Lisa	Chai tea
10	02/13/2013	Mike	Chai tea
11	02/13/2013	Sergio	Espresso
12	02/13/2013	Sergio	Espresso

	Date	Customer	Beverage
1	01/05/2013	Jeff	Espresso
2	01/05/2013	Mike	Chai tea
3	01/05/2013	Sergio	Latte
4	01/05/2013	Mike	Espresso
5	01/05/2013	Lisa	Espresso
6	02/13/2013	Lisa	Chai tea
7	02/13/2013	Mike	Chai tea
8	02/13/2013	Sergio	Espresso

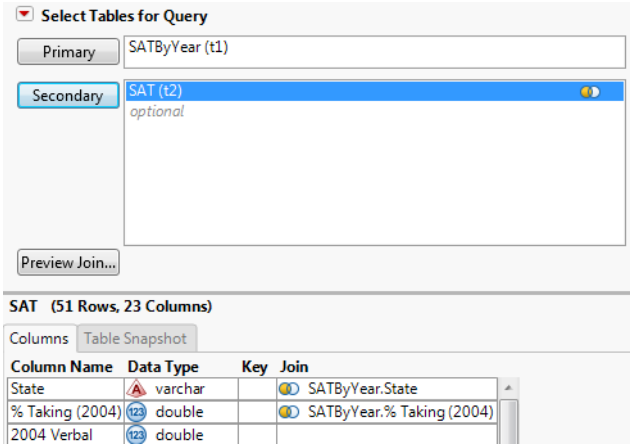
Query and Join Data Tables with JMP Query Builder

The JMP Query Builder option in the Tables menu enables you to query data tables and save selected data into a new data table. This feature is similar to using the Join command but lets you perform queries before saving the data. For example, you can query SAT data and save only data for 2004 test scores in a data table. You can also include a prompt that lets the user run the query and choose a subset of the 2004 test scores.

1. Select **Help > Sample Data Library** and open SAT.jmp and SATByYear.jmp.
2. Display SATByYear.jmp.
3. Select **Tables > JMP Query Builder**.

   The current data table, SATByYear.jmp, is selected as the Primary table.
4. Select SAT.jmp in the Available Tables list and click **Secondary**.
5. Select SAT.jmp next to the Secondary button and view the Columns tab. The Join column shows that two columns have the same name and were joined.

Figure 6.33 Joined Columns



6. Click **Build Query**.

Add Columns

1. In the Available Columns list, select t1.State, t1.%Taking (2004), t2.2004 Verbal, and t2.2004 Math.
2. Click **Add**.

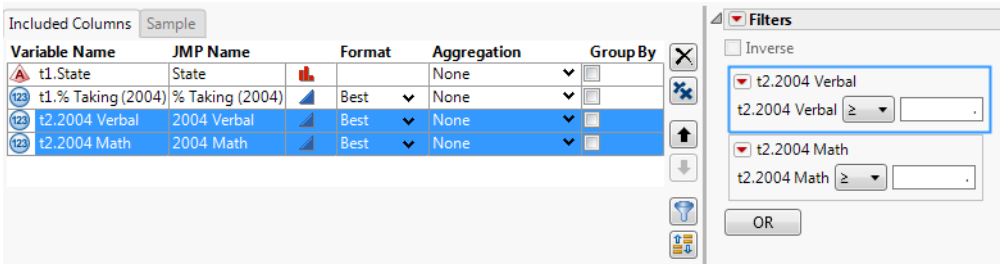
   The columns are added to the Included Columns tab.
3. Select **Distinct rows only** to avoid saving duplicate rows.
4. On the Query Preview tab, make sure that **Update preview automatically** is selected so that you can see the selected columns.

Add Filters

1. In the Included Columns list, select t2.2004 Verbal and t2.2004 Math, and then click **Add Selected Items to Filter** .

   The columns are added to the Filters list.

Figure 6.34 Selected Filters



2. Click the red triangle next to each filter and select **Prompt on Run**. Accept the default prompt message.

Run the Query

1. Click **Run Query**.

   You are prompted to indicate which math and verbal scores to save in the data table. To save all data, do not change the values in the prompt window.

2. Click **OK** to create the data table.

Figure 6.35 Queried Data

	State	% Taking (2004)	2004 Verbal	2004 Math
1	Alabama	0.1	560	553
2	Alaska	0.53	518	514
3	Arizona	0.32	523	524
4	Arkansas	0.06	569	555
5	California	0.49	501	519
6	Colorado	0.27	554	553
7	Connecticut	0.85	515	515
8	District of Colum...	0.77	489	476
9	Delaware	0.73	500	499
10	Florida	0.67	499	499
11	Georgia	0.73	494	493

See "Build SQL Queries in Query Builder" for more information about features that also appear in Query Builder for databases.

Tips:

- To join data from different sources (for example, a database and Microsoft Excel), use Query Builder to import the database data into a data table; import the Excel data into a data table; and use JMP Query Builder in the Tables menu to query and join the tables.
- When you open a query, data tables in the query open as hidden files that you can open from the JMP Home Window. You can also open the hidden files from the Table panel when you build the query.
- If you try to query a Microsoft Excel file that has more than 255 columns, only 255 columns are imported. Instead, select **File > Open** to open the Excel file in the Excel Import Wizard, and save the file as a data table.

About Links to Data Tables in Queries

When you query data tables, the resulting data table contains scripts for rerunning the query, updating the data, and editing the query. Data tables in these scripts have *absolute paths*. For example, the following portion of these scripts defines where two JMP sample data tables are stored.

```
JMP Tables(
     ["SAT" => "C:\Program Files\SAS\JMPPRO\16\Samples\Data\SAT.jmp",
     "SATByYear" => "C:\Program
  Files\SAS\JMPPRO\16\Samples\Data\SATByYear.jmp"]
  ),
```

However, when you save a query as a .jmpquery file, the file contains *relative paths* to the original data tables if it is possible to create them. The following example shows data tables that are stored in the FuelData subfolder that is relative to the .jmpquery file.

```
JMP Tables( ["Cars" => "FuelData\Cars.jmp",
  "Trains" => "FuelData\Trains.jmp"] )
```

In the .jmpquery file, if a relative path cannot be created, an absolute path is used with path variable substitutions if possible. When you run the query, you are prompted to select the data table if the table cannot be found. JMP then detects whether other missing data tables in the query are in the selected folder.

You can also use path variables to locate the data tables. For example, you might write a script that selects a data table in the $DOCUMENTS folder. Or you can define a path variable in a JSL script and then run a query from the script. See the *Scripting Guide* for more information about path variables.

Virtually Join Data Tables

Virtual Join links a main data table to one or more auxiliary data tables. This feature enables the main data table to access data from the auxiliary data tables without physically joining the tables. Virtually joining tables saves memory, because the same data are not replicated in every table that references them. And updating linked data is simpler; linked data can be independently updated in the source table without being updated in the referencing table.

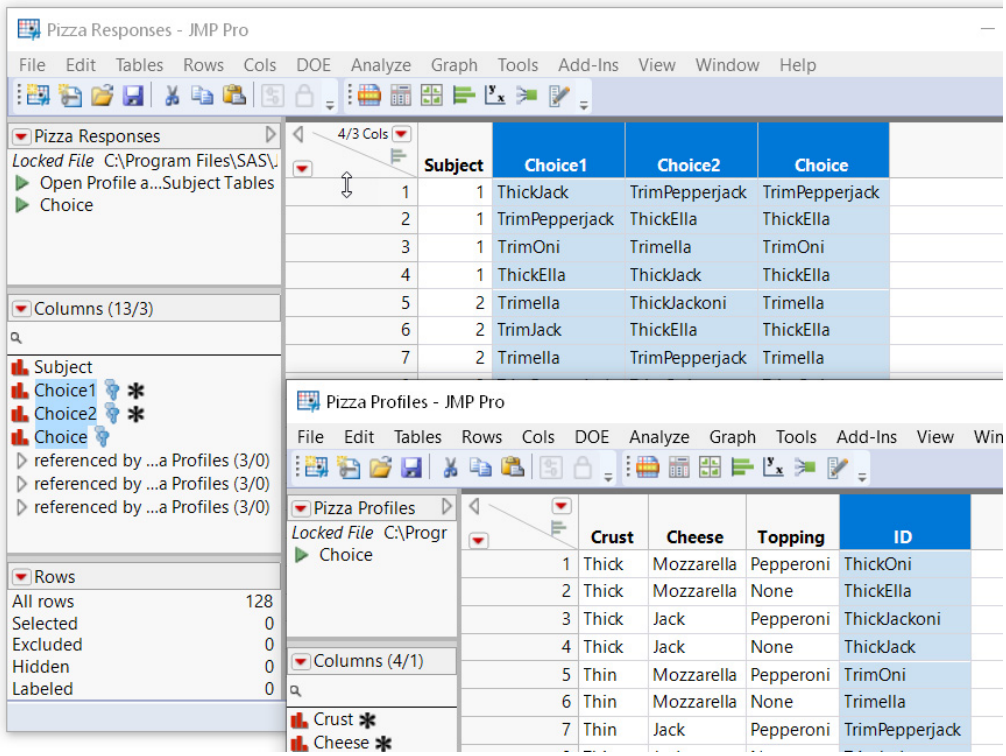
The Link ID and Link Reference column properties make the linking possible.

- The Link ID column property marks a column in the *auxiliary* data table as the ID column. That is, the rows of the data table are uniquely identified by the values of the ID column. The data table that has a Link ID column property is referred to as the *referenced* data table.
- The Link Reference column property links a column in the *main* data table to the Link ID column in the referenced data table. The column property specifies the path name of the

referenced data table. The column that has a link reference is referred to as the *referencing* column. The referencing column can look up the data of the auxiliary data tables through the Link ID column.

Figure 6.36 shows an example of virtually joined data tables. Pizza Profiles.jmp is the referenced data table because the ID column contains a Link ID column property. The ID column contains unique values that correspond to data in the referencing data table, Pizza Responses.jmp.

Figure 6.36 Virtually Joined Data Tables



Note: A column can have a Link Reference column property, but it does not necessarily have to link columns from the referenced data table. The virtual join icon next to the column in the Columns list is gray . This can happen if the referenced data table is not opened or the referencing column is excluded. See "Virtual Join Icons" for more information about the other virtual join icons.

With the link column properties setup, all columns from the referenced data tables become part of your main data table. Therefore, if the main data table has a referencing column, the columns of the referenced table automatically appear in the Select Columns list of the launch window.

Example of Virtually Joining Columns

Suppose that respondents in a pizza experiment chose their favorite crust, topping, and cheese. To see which cheese the respondents preferred, follow these steps:

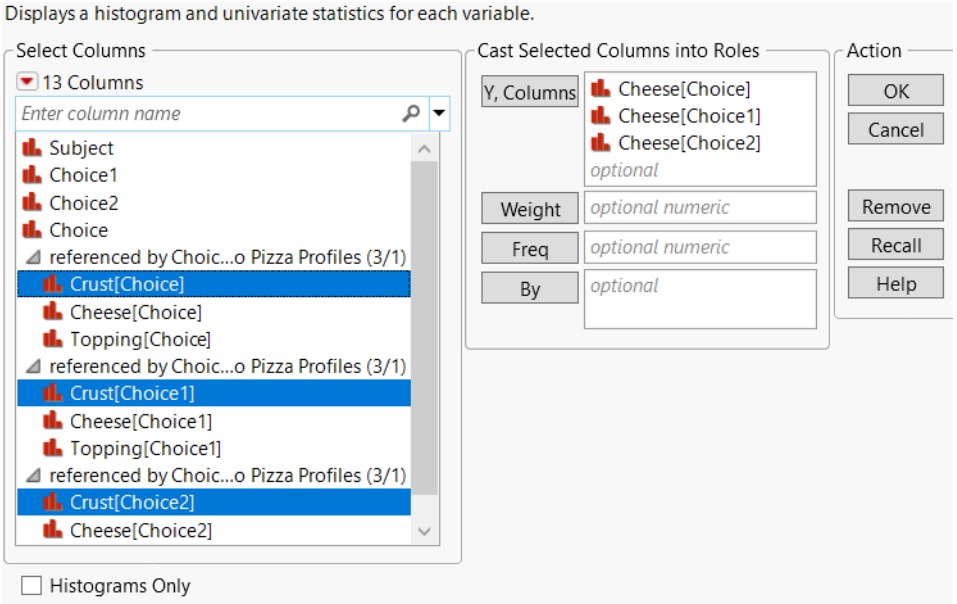
1. Select **Help > Sample Data Library** and open Pizza Profiles.jmp and Pizza Responses.jmp.
2. First, right-click the ID column in Pizza Profiles.jmp and select **Link ID**.

   This column contains unique values that correspond to values in Pizza Responses.jmp. For example, row one in Pizza Profiles.jmp indicates that pizza with thick crust, mozzarella cheese, and pepperoni toppings is nicknamed ThickOni. When all responses in Pizza Responses.jmp are "ThickOni", the subject chose pizza with these attributes.
3. In Pizza Responses.jmp, select the three columns that begin with "Choice".
4. Right-click and select **Link Reference** > Pizza Profiles.jmp.

   The selected columns are linked to the ID column in Pizza Profiles.jmp and appear in the main data table's Columns list (Figure 6.36).

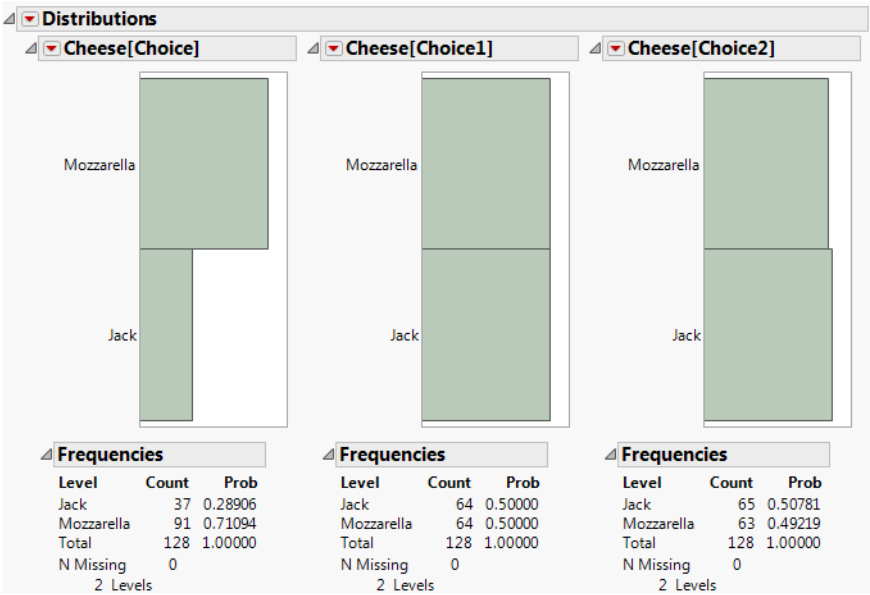
   Note: Look at one of the column's Link Reference properties again and notice that the Pizza Profiles.jmp data table is selected. See "Link Data Tables with Virtual Join Properties" for information about changing the link.
5. In Pizza Responses.jmp, select **Analyze > Distribution**.
6. Scroll down to select the three Cheese columns in the referenced column groups, click **Y, Columns**, and click **OK**.

Figure 6.37 Distribution Launch Window



More respondents chose Mozzarella in the Choice column. In Choice1 and Choice2, the proportion of Mozzarella to Jack is nearly even.

Figure 6.38 Distribution of Pizza Responses



Notes:

- See the Movie Rentals.jmp sample data table for another example of virtually joined columns. The data are already joined with Movie Customers.jmp and Movie Inventory.jmp, so you do not need to specify the Link Reference.
- A data table can have only one Link ID column property. The Link ID column cannot have duplicate values.
- A data table can have multiple referencing columns. The columns can reference different or the same data tables.
- A referenced data table can also have a referencing column (a column that references another data table).
- If the Link Reference column property is removed from the column, or the referenced data table is closed, the corresponding referenced columns are removed from the main data table.
- The data types of the linked columns must match.
- The Expression and Row State data types do not support the Link Reference or Link ID column properties.
- When the main data table is saved, the Link Reference column property is saved, but referenced columns are not saved. The data tables are relinked when you open them.
- When a referenced data table is renamed, the corresponding Link Reference column property is automatically updated.
- Referenced columns are not included when you select Copy Table Script from the Table panel red triangle menu.
- Select Row State Synchronization options to synchronize row states in data tables that contain virtually joined columns. See "Example of Virtually Joining Columns and Showing Row States".
- A column can have both Link ID and Link Reference column properties.

Tips:

- In addition to right-clicking a column to add the Link ID and Link Reference column properties, you can add the properties through the Column Info window. See "Link Data Tables with Virtual Join Properties".
- You can specify that the linked column name be used in virtually joined columns that are shown in the Columns list. This option lets you specify a shorter column name (for example, Cheese instead of Cheese[Choice]). In the Link Reference column property, select **Use Linked Column Name**. You can also manually rename the column as usual. The Table preference is called Virtual Join Use Linked Column Name. It is deselected by default.
- To share the data with another user, you might want to merge the data so that the data tables are permanently joined. This option is helpful because you provide one data table,

not the main and auxiliary data tables. To merge the joined columns into the table that has a link reference, select **Merge Referenced Data** from the red triangle menu in the data table's left pane. The data are replicated from the auxiliary data table to the main data table. When you save the main data table, the actual data are also saved. The Link Reference column property is removed from the referencing column in the main data table. See "Example of Merging Referenced Data in Virtually Joined Data Tables".

Example of Virtually Joining Columns and Showing Row States

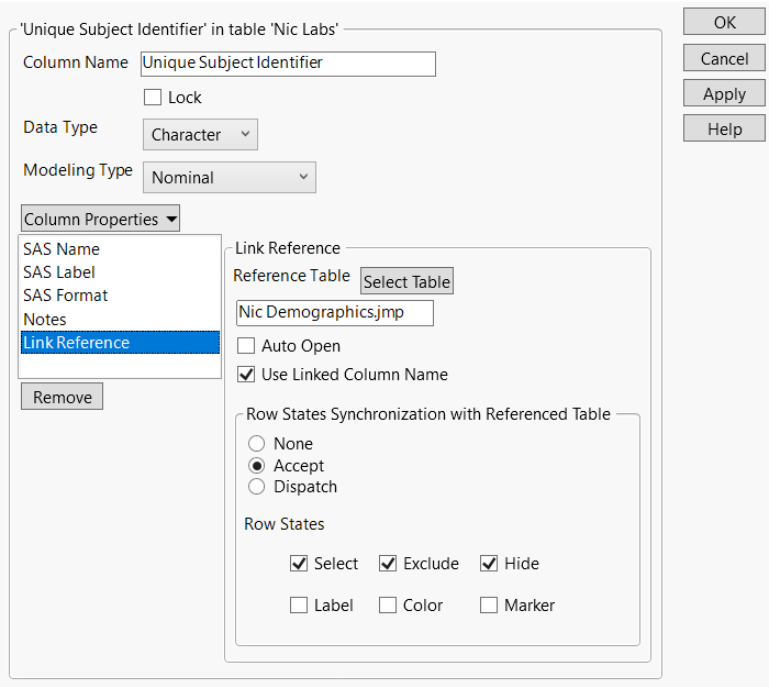
When row states change in one data table that has a virtually joined column, those row states can automatically show up in virtually joined columns in other linked data tables. Suppose that the main and auxiliary data tables have a subject ID column. The row states are changed in the data table that contains a Link ID property on the subject ID column. Then the row states are updated in the data table that contains virtually joined Link Reference columns. You can specify whether the Link Reference columns send (*dispatch*) row states to the other data tables or receive (*accept*) row states.

1. Select **Help > Sample Data Library** and open Nic Demographics.jmp, Nic Labs.jmp, and Nic Adverse Events.jmp.
2. View Nic Labs.jmp.

   Notice that, in the Columns list, the blue Virtual Join icon appears next to Unique Subject Identifier. The blue color indicates that the column has a Link Reference column property and the referenced data table (Nic Demographics.jmp) is open. The Link Reference column property links the Unique Subject Identifier column in the current data table to the Unique Subject Identifier column in the referenced data table. If the icon is gray , the referenced data table is not open, or the columns could be linked incorrectly.

   Nic Adverse Events.jmp is set up the same way.

Figure 6.39 Link Reference Column Property on Unique Subject Identifier



3. In Nic Labs.jmp, select the Unique Subject Identifier column and select **Cols > Column Info**.
4. Select the **Link Reference** column property.

   Notice that Accept is selected. This means that when rows in the referenced data table (Nic Demographics.jmp) contain a row state, the row state automatically appears in the current data table.

   Also notice that the Select, Exclude, and Hide row states are selected. When these row states are selected in Nic Demographics.jmp, the row states appear in Nic Lib.jmp and Nic Adverse Events.jmp.
5. In Nic Demographics.jmp, right-click the first row and select **Hide and Exclude**.

   The first row contains data for Unique Subject Identifier 101001.
6. View Nic Labs.jmp to see the hidden and excluded rows for Unique Subject Identifier 101001.

Cautions:

- Dispatching row states in multiple tables with multiple columns can be complex and should be done with caution.
- Dispatching row states is best used when no multiple row states are assigned to the same ID value. If a Link Reference column dispatches the color row state, ambiguity quickly develops when there are multiple rows with the same ID value but different color row

states. For example, the data table has two rows with an ID value of 001. You color one row red and the other row green, There is no way for JMP to know how to set the row state in the linked data table.

Notes:

- Multiple columns in table A can link to table B. And Table B can itself have multiple columns that link to table C. The row states of all of the rows appear in the data tables that contain linked columns. For example, if row 1 is excluded in table A and row 1 is hidden in table B, the main data table shows that the row as excluded and hidden.
- If a data table is accepting a specific row state, avoid manually applying that row state in the data table. Let the dispatching column control that row state.

Example of Merging Referenced Data in Virtually Joined Data Tables

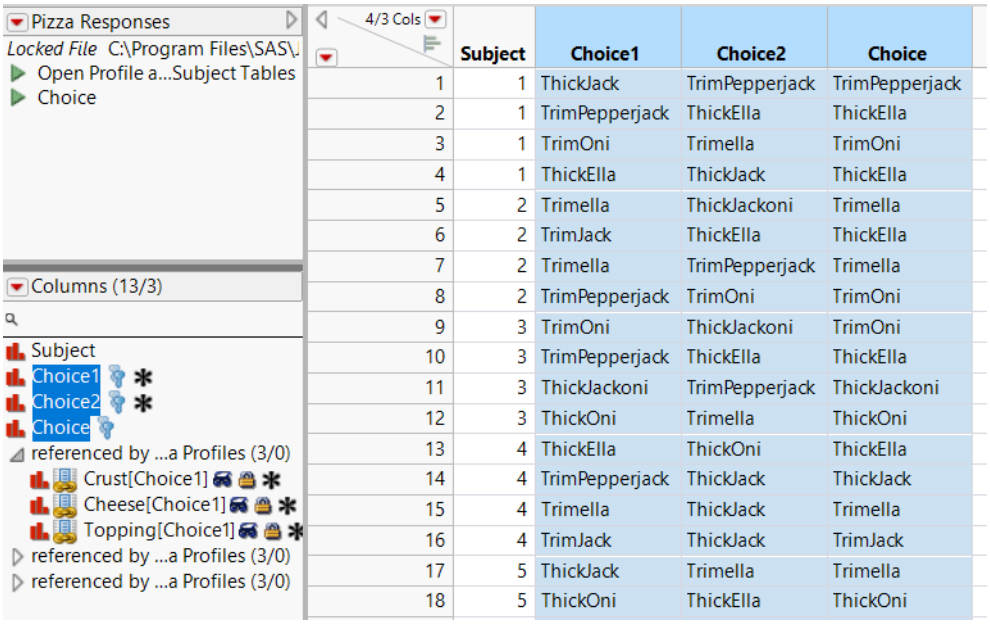
To share the data from virtually joined data tables with another user, you might want to merge the data so that the data tables are permanently joined. This option is helpful because you provide one data table, not the main and auxiliary data tables.

When you merge referenced data, the data are replicated from the auxiliary data table to the main data table. When you save the main data table, the actual data are also saved.

1. Select **Help > Sample Data Library** and open Pizza Profiles.jmp and Pizza Responses.jmp.
2. Right-click the ID column in Pizza Profiles.jmp and select **Link ID**.
3. Select the Choice1, Choice2, and Choice columns in Pizza Responses.jmp, right-click, and then select **Link Reference** > Pizza Profiles.jmp.

   The selected columns are linked to the ID column in Pizza Profiles.jmp.

Figure 6.40 Selected Columns in Virtually Joined Data Table



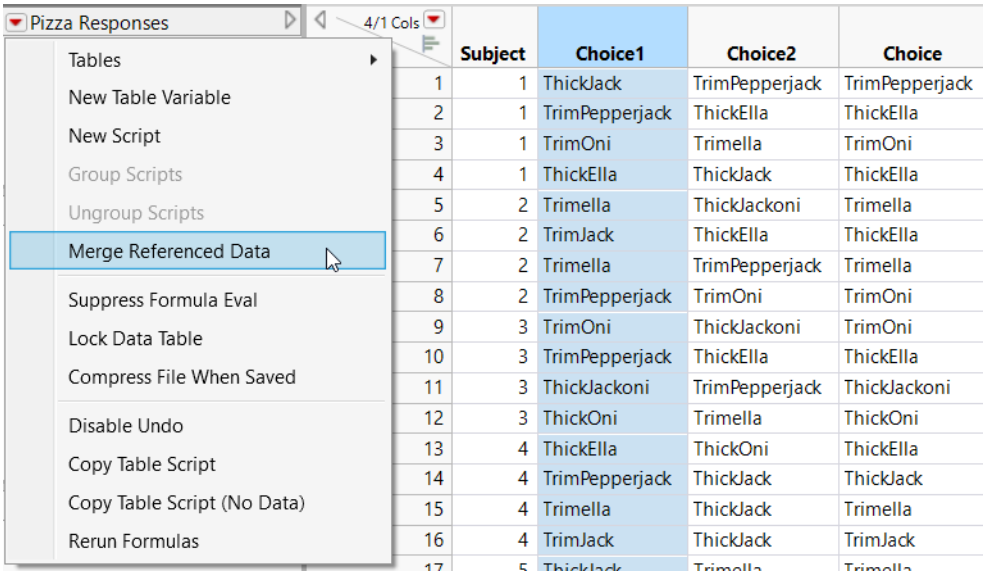
	Subject	Choice1	Choice2	Choice
1	1	ThickJack	TrimPepperjack	TrimPepperjack
2	1	TrimPepperjack	ThickElla	ThickElla
3	1	TrimOni	Trimella	TrimOni
4	1	ThickElla	ThickJack	ThickElla
5	2	Trimella	ThickJackoni	Trimella
6	2	TrimJack	ThickElla	ThickElla
7	2	Trimella	TrimPepperjack	Trimella
8	2	TrimPepperjack	TrimOni	TrimOni
9	3	TrimOni	ThickJackoni	TrimOni
10	3	TrimPepperjack	ThickElla	ThickElla
11	3	ThickJackoni	TrimPepperjack	ThickJackoni
12	3	ThickOni	Trimella	ThickOni
13	4	ThickElla	ThickOni	ThickElla
14	4	TrimPepperjack	ThickJack	ThickJack
15	4	Trimella	ThickJack	Trimella
16	4	TrimJack	ThickJack	TrimJack
17	5	ThickJack	Trimella	Trimella
18	5	ThickOni	ThickElla	ThickOni

4. In Pizza Responses.jmp, select the Choice1 column if it is not already selected.

   Note: If you do not select the columns, then all columns will be in the merged data.

5. Click the red triangle next to the name of the data table and select **Merge Referenced Data**.

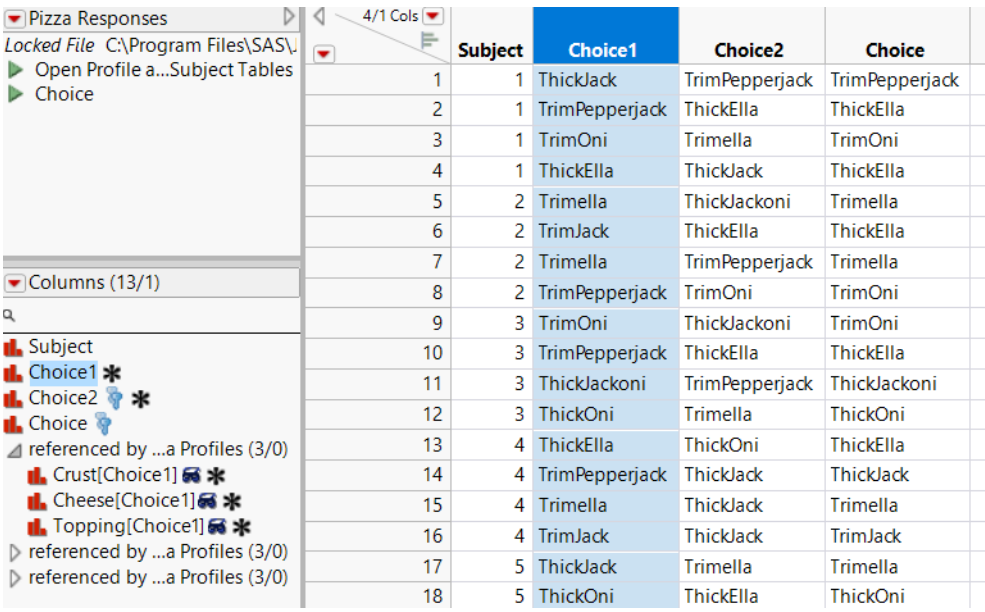
Figure 6.41 Merging Referenced Data



	Subject	Choice1	Choice2	Choice
1	1	ThickJack	TrimPepperjack	TrimPepperjack
2	1	TrimPepperjack	ThickElla	ThickElla
3	1	TrimOni	Trimella	TrimOni
4	1	ThickElla	ThickJack	ThickElla
5	2	Trimella	ThickJackoni	Trimella
6	2	TrimJack	ThickElla	ThickElla
7	2	Trimella	TrimPepperjack	Trimella
8	2	TrimPepperjack	TrimOni	TrimOni
9	3	TrimOni	ThickJackoni	TrimOni
10	3	TrimPepperjack	ThickElla	ThickElla
11	3	ThickJackoni	TrimPepperjack	ThickJackoni
12	3	ThickOni	Trimella	ThickOni
13	4	ThickElla	ThickOni	ThickElla
14	4	TrimPepperjack	ThickJack	ThickJack
15	4	Trimella	ThickJack	Trimella
16	4	TrimJack	ThickJack	TrimJack

A message confirms that the selected column was merged. Notice that the blue virtual join icon for Choice1 no longer appears in the Column list. The Link Reference column property was removed.

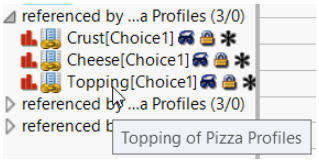
Figure 6.42 Merged Data



	Subject	Choice1	Choice2	Choice
1	1	ThickJack	TrimPepperjack	TrimPepperjack
2	1	TrimPepperjack	ThickElla	ThickElla
3	1	TrimOni	Trimella	TrimOni
4	1	ThickElla	ThickJack	ThickElla
5	2	Trimella	ThickJackoni	Trimella
6	2	TrimJack	ThickElla	ThickElla
7	2	Trimella	TrimPepperjack	Trimella
8	2	TrimPepperjack	TrimOni	TrimOni
9	3	TrimOni	ThickJackoni	TrimOni
10	3	TrimPepperjack	ThickElla	ThickElla
11	3	ThickJackoni	TrimPepperjack	ThickJackoni
12	3	ThickOni	Trimella	ThickOni
13	4	ThickElla	ThickOni	ThickElla
14	4	TrimPepperjack	ThickJack	ThickJack
15	4	Trimella	ThickJack	Trimella
16	4	TrimJack	ThickJack	TrimJack
17	5	ThickJack	Trimella	Trimella
18	5	ThickOni	ThickElla	ThickOni

If you hover over one of the linked columns in the Columns panel, a tooltip indicates which data table it is linked to.

Figure 6.43 Data Table that Contains the Linked Column



Example of a Link ID and a Link Reference on a Single Column

Suppose that employee data is available in multiple data tables and can be linked together by a single column that has both a Link ID and a Link Reference property. To see an example of how the data tables are set up, follow these steps:

1. Select **Help > Sample Data Library** and open Employee Master.jmp.

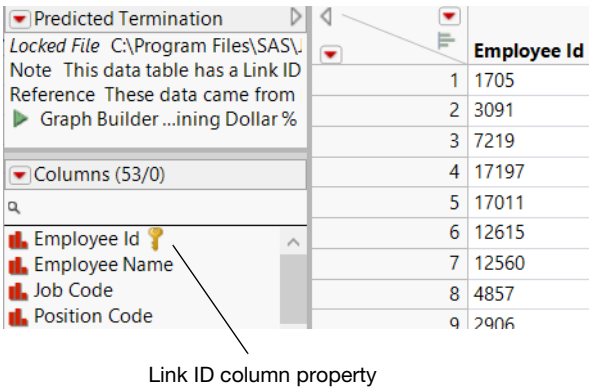
   Note: In the Employee Id column's Link Reference column property, the Auto Open option is selected to automatically open two other tables, Education History.jmp and Predicted Termination.jmp. They appear in the Home Window's Window List.

   – Employee Master.jmp has information about each employee.
   – Education History.jmp has the information about the education of each employee.
   – Predicted Termination.jmp has job performance information about each employee.

2. Open Predicted Termination.jmp from the Home Window's Window List.

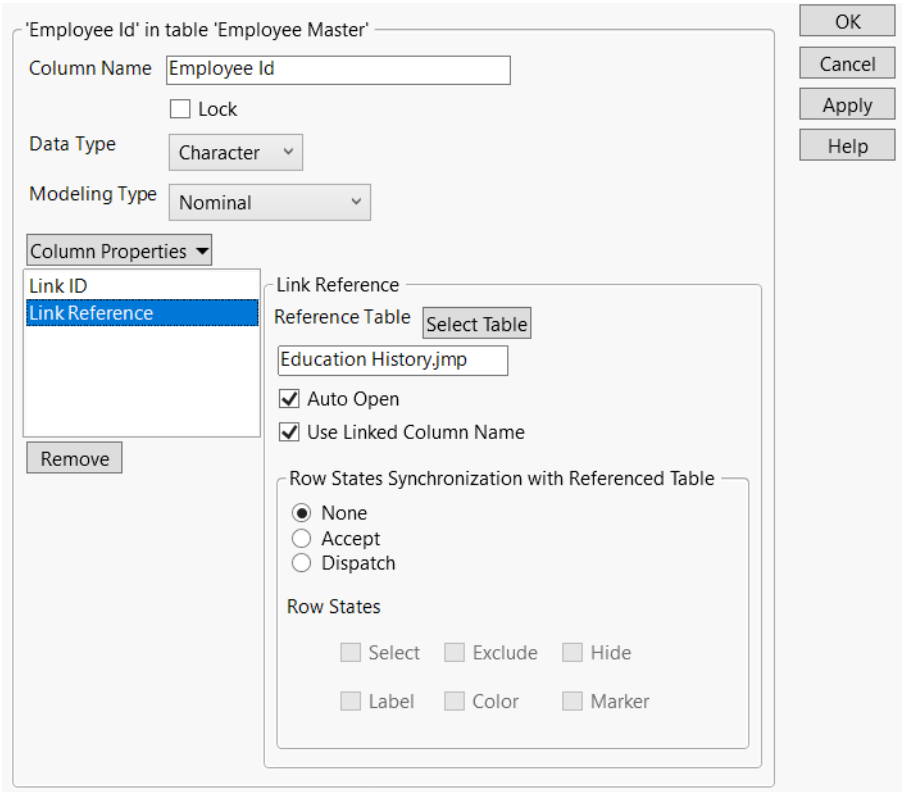
   Notice that the Employee Id column has a Link ID column property.

Figure 6.44 Link ID Column Property



3. In Employee Master.jmp, select the Employee Id column and select **Columns > Column Info**.

Figure 6.45 Link ID and Link Reference Column Properties in Employee Master.jmp

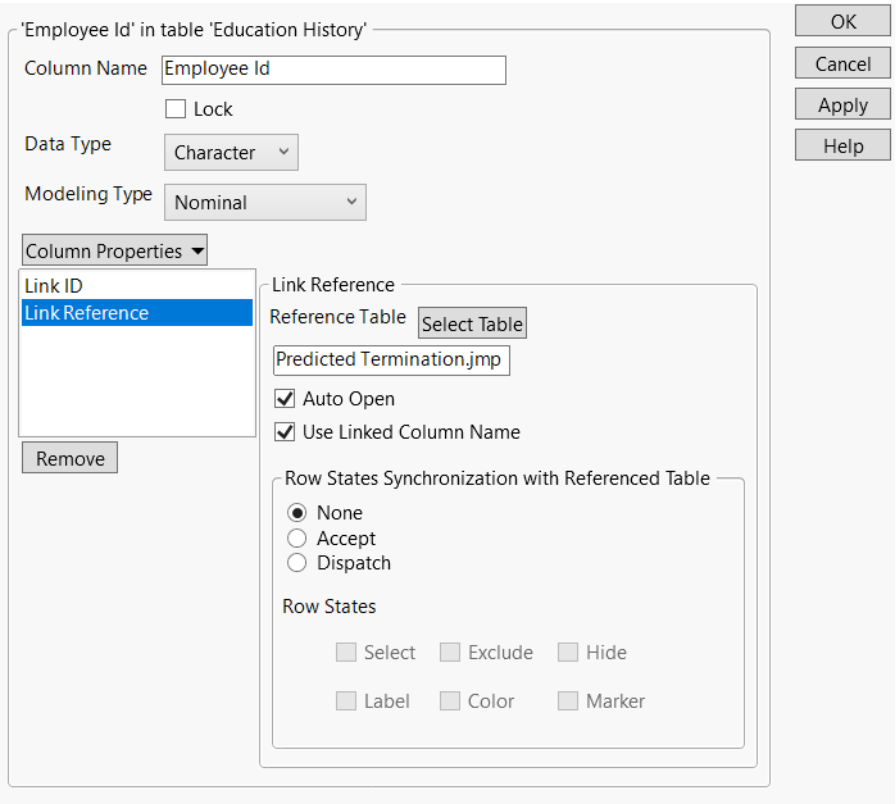


Notice that the column has a Link ID column property and a Link Reference to Education History.jmp.

In the data table's Columns list, the Link Reference icon and Link ID icon appear next to the Employee Id column.

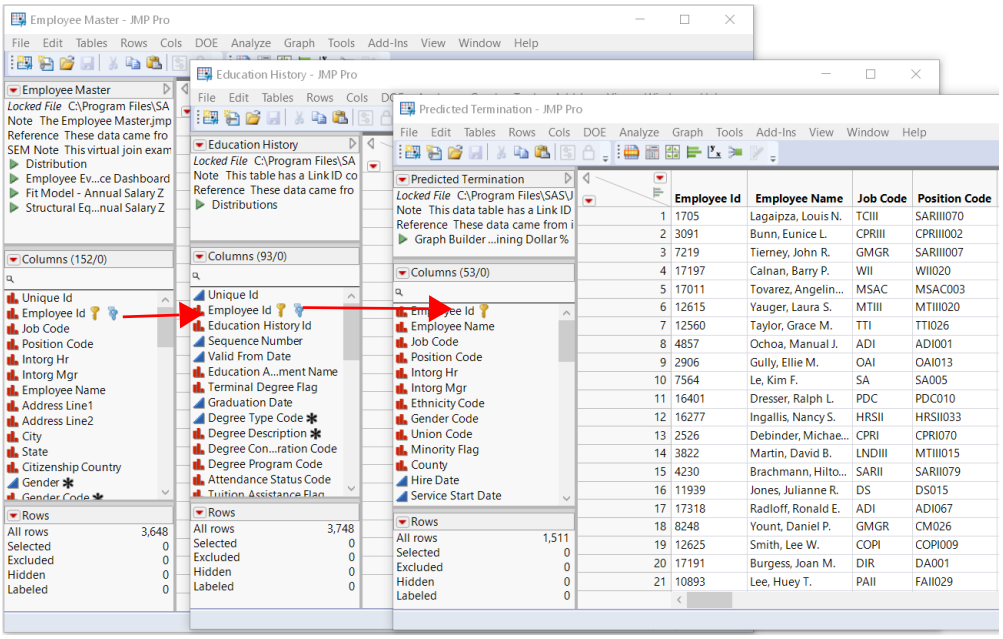
4. Open Education History.jmp from the Home Window's Window List.
5. Select the Employee Id column and select **Columns > Column Info**.

Figure 6.46 Link ID and Link Reference Column Properties in Education History.jmp



As shown in Figure 6.46, Employee Id has a Link ID column property and a Link Reference to Predicted Termination.jmp.

Figure 6.47 Linked Data Tables



The Employee Id column of Employee Master.jmp references Education History.jmp, which then references Predicted Termination.jmp. All data are available through virtual join from the main data table, Employee Master.jmp.

Notes:

- Auto Open in a column's Link Reference property specifies whether the linked table is automatically opened when the main data table is opened.
- If referenced columns are unhidden when the data table is saved, those referenced columns are automatically linked and unhidden when the table is opened.
- When you close the data tables, it is a best practice to close the main table first (Employee Master.jmp), because there is a link between that table and all the others. Closing them in a different order might display a prompt indicating that the table that you are closing has other windows open. You might want to cancel and consider saving and closing your tables in a different order.

Update Data Tables

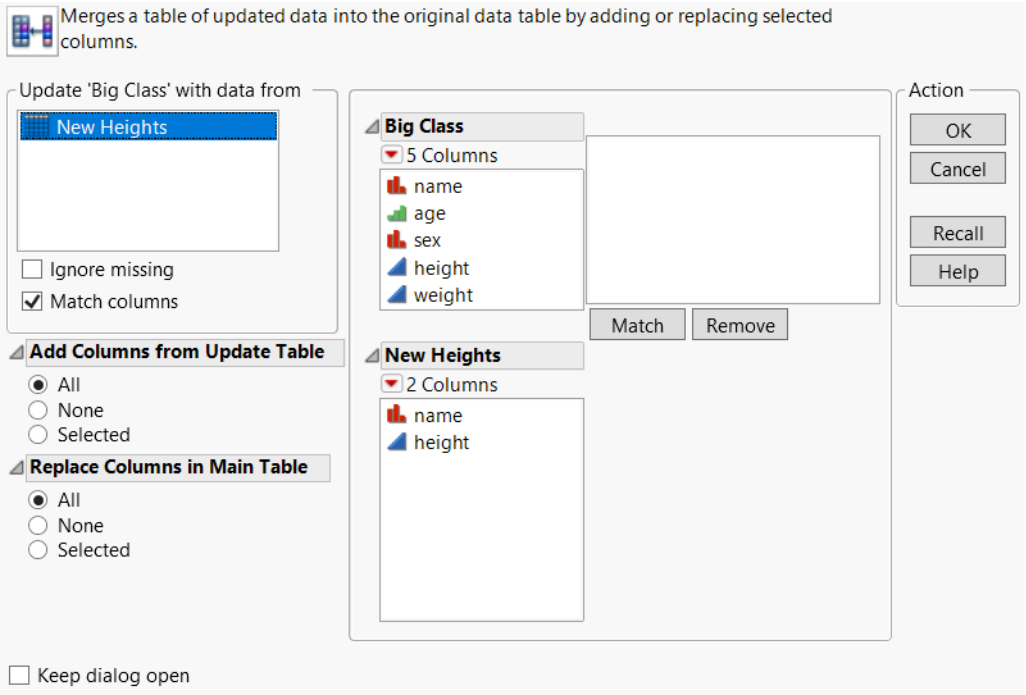
If you have two data tables and would like to update your original table with data from a new table, select **Tables > Update**. The Update command is a special case of Join in place. It is a Join with the Update option checked, and it does not result in a new table.

Before you update a table, make sure that the name of the column containing the values that you want to replace is the same as the name of the column containing the data that you want to replace it with.

To replace values in the active table with those found in another open table

1. Click the original table that you want to update (this is the table whose values you want to replace) to make it the active table.
2. Select **Tables > Update**.

Figure 6.48 Updating a Table



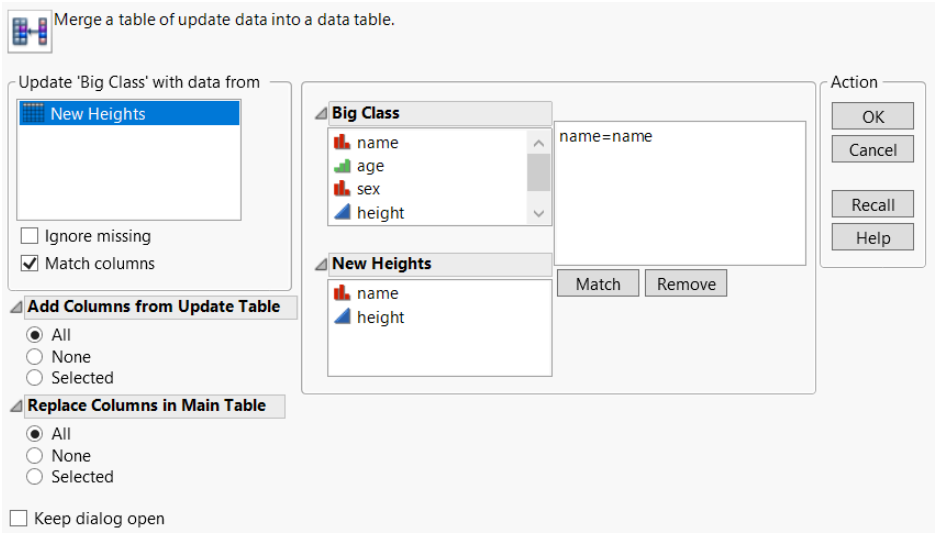
3. Highlight the new table containing the data that you want to transfer to the original table.
4. (Optional) If you do not want JMP to replace the values in the original table with any missing values found in the new table, select the box next to **Ignore missing**. The original table retains its original values if they correspond to missing values in the new table.
5. If the two tables have one or more columns whose values uniquely describe each row, JMP uses those columns as the *match column values*. That is, JMP updates the rows whose match column values coincide. JMP uses these columns to preserve the sorted order of the data. If your tables do not have matching column values, you can incorporate the updated values according to their row order by continuing here. To proceed with tables containing matching column values, see "To update a table using matching columns".

6. From the Add Columns from Update Table area, select an option. Using these options, you can add columns (that do not exist) from the new table into the original table.
   - Choose **All** to add all columns from the new table into the original table.
   - Choose **Selected** to add only columns that you have selected from the new table into the original table.
   - Choose **None** if you do not want to add any non-existent columns from the new table into the original table.
7. (Optional) From the Replace Columns in Main Table area, select an option, Using these options, you can replace the values in columns that have the same names.
   - Choose **All** to add all columns with the same names from the new table into the original table.
   - Choose **Selected** to replace only columns with the same names that you have selected from the new table into the original table.
   - Choose **None** if you do not want to add any non-existent columns that have the same names from the new table into the original table.
8. Click **OK**.

To update a table using matching columns

1. Follow the first three steps outlined in the previous section, "To replace values in the active table with those found in another open table".
2. Select **Match columns**.

Figure 6.49 This Window Appears When You Click Match Columns



3. Highlight the two column names (in the respective tables) that you want to match.
4. Click **Match**.
5. (Optional) Repeat to match more columns.
6. Click **OK**.

Note: Columns that have different names from the columns in the table that you are updating (and that have not been assigned matches) are appended as separate columns. To avoid this problem, select the None option in the Add Columns from Update table area.

Example of Updating a Data Table

Suppose a researcher has a data table containing height measurements for students. The researcher receives an updated table that contains more recent measurements of the students' heights. The researcher wants to avoid scrolling through the data tables to find the students whose height has changed, and copying and pasting the new values. Using the **Update** command, the researcher can quickly update the original data table with the new height values.

1. Select **Help > Sample Data Library** and open Big Class.jmp and New Heights.jmp.

   The Big Class.jmp table contains the original data, and the New Heights.jmp table contains the updated data.
2. From the Big Class.jmp table, select **Tables > Update**.
3. In the **Update...with data from** box, select New Heights.
4. Select **Match columns**.
5. In the Big Class and New Heights lists, select name.
6. Click **Match**.

   This tells JMP to use name as the match column value, since it is the column whose values uniquely describe each row.
7. Click **OK**.

Figure 6.50 The Updated Big Class.jmp Table

	name	age	sex	height	weight
1	KATIE	12	F	62	95
2	LOUISE	12	F	61	123
3	JANE	12	F	58	74
4	JACLYN	12	F	68	145
5	LILLIE	12	F	52	64
6	TIM	12	M	64	84
7	JAMES	12	M	63	128
8	ROBERT	12	M	70	79
9	BARBARA	13	F	62	112
10	ALICE	13	F	61	107
11	SUSAN	13	F	58	67
12	JOHN	13	M	68	98
13	JOE	13	M	64	105
14	MICHAEL	13	M	60	95

You can see that the height values in the updated table no longer match the values in the original table. The values have been updated to use the newer values from the New Heights.jmp table.

Anonymize Data

The Anonymize Data feature enables you to create a new data table in which certain unique identifiers have been removed:

- Column headings, character data, and value labels are modified.
- Data in nominal columns are modified.
- Modified data in ordinal columns appear in the same order as in the original data table.
- Data in continuous columns are not modified.
- Column names in column properties are modified.

Note: Table scripts and some table variables cannot be anonymized. If your data table contains content that cannot be anonymized, a warning message appears in the log.

“Notes” column properties are removed. Other column properties are modified based on changes in the data table. For example, Value Order properties are renamed. Formulas except for prediction formulas are updated. Some column properties might no longer have the desired effect on the column.

To rename data, do one of the following:

- To rename data in specific columns, select the columns and then select **Tables > Anonymize**.
- To rename data in the entire data table, select **Tables > Anonymize**.

The data appear in a new data table.

DISCLAIMER: The Anonymize Data utility is provided merely to assist Users in removing certain unique identifiers from a data table as described above. The Anonymize Data utility might not remove all sensitive or personally identifiable information from a data table. Users should not rely exclusively on the Anonymize Data utility to remove such information where complete anonymization, pseudo-anonymization, or de-identification of data is desired or required by law or policy (such as where data are or might be disclosed to other parties). SAS does not represent, and specifically disclaims, that use of the Anonymize Data utility will by itself result in the User's compliance with any national, state, local or international laws, regulations, or policies that pertain to the privacy, de-identification, or anonymization of sensitive or personally identifiable information.

Chapter 7

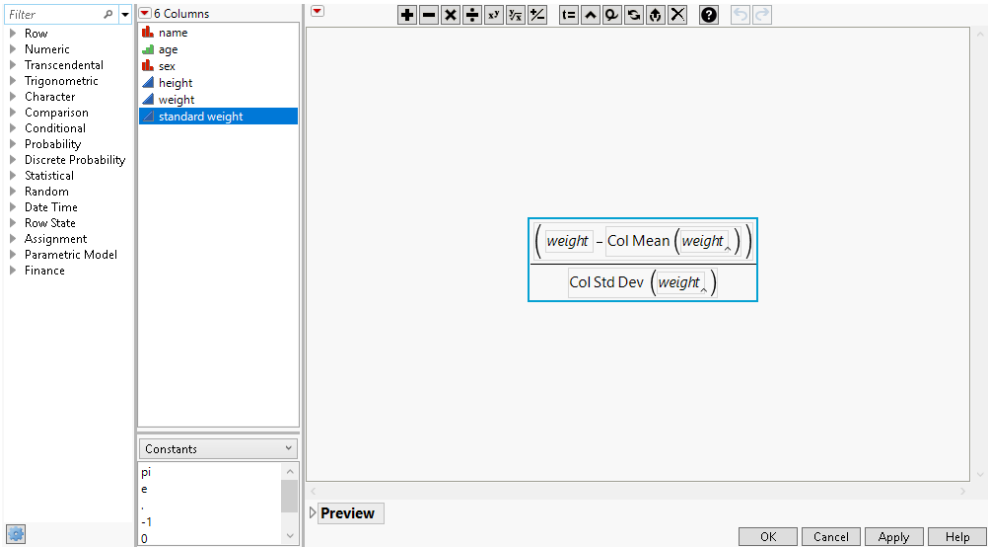
Create Formulas in JMP

Formula Editor

Use the JMP Formula Editor to create a column whose values are computed by a formula and store that formula as part of a column's information. Formulas can be simple assignments of numeric, character, or row state constants, or they can contain complex evaluations based on conditional clauses. The Formula Editor window operates like a calculator with buttons, displays, and a list of functions.

For more information about the functions that are available in the Formula Editor, see "Reference for JMP Functions in Formulas".

Figure 7.1 The Formula Editor



Contents

Overview of Formulas . . . 395
Build Formulas . . . 395
Refer to Data Table Values in Formulas . . . 397
Use Local Variables in Formulas . . . 400
Incorporate Parameters in Formulas . . . 401
Insert Constants in Formulas . . . 402
Add Operators to JMP Formulas . . . 404
Keypad Reference for the Formula Editor . . . 404
Use Functions in Formulas . . . 406
Find Frequently Used Functions in the Formula Editor . . . 408
Copy and Replicate Formulas . . . 410
Order Expressions in Formulas . . . 412
Build a Formula in Order of Precedence . . . 413
JMP Formula Editor Options . . . 414
Calculate Derivatives of JMP Functions . . . 414
Simplify Complex Formulas . . . 415
Evaluate Formulas . . . 416
Ignore Errors in Formulas . . . 416
View Formulas in JSL . . . 417
Edit Formulas . . . 417
Correct Mistakes in Formulas . . . 417
Select Expressions in Formulas . . . 418
Delete Functions Formulas . . . 418
Cut, Copy, and Paste Formulas . . . 419
Click and Drag Formulas . . . 419
Customize Formulas in the JMP Formula Editor . . . 420
Hide and Show Outlines Formulas . . . 420
Change the Font Size Formulas . . . 420
Specify the Matrix Size Formulas . . . 420
Change the Orientation of a JMP Formula . . . 421
Open and Close Arguments Formulas . . . 421
Examples of Building Formulas in the JMP Formula Editor . . . 421
Use Basic Features in the JMP Formula Editor . . . 422
Use Local Variables Formulas . . . 423
Use the Munger Function Formulas . . . 424
Use the Match Conditional Function Formulas . . . 425
Use the Delete Expression Button Formulas . . . 426
Keyboard Shortcuts for the JMP Formula Editor . . . 427
Glossary of Terms for the JMP Formula Editor . . . 428

Overview of Formulas

Formulas are an integral part of a data table for the following reasons:

- They are stored as part of a column's information when you save the data table.
- You can examine or change them at any time by opening the Formula Editor.
- Their values can be linked to, or dependent on, the values in other columns. Their values are automatically recomputed whenever you edit the values in the columns to which the formula is linked.
- Their values are locked in the data table so that they cannot be manually edited.

This chapter describes the Formula Editor and shows how to build formulas. For more information about each function, see "Reference for JMP Functions in Formulas".

Build Formulas

A formula is an expression stored in a column that performs operations in order to insert values into that column. Formulas can perform mathematical operations, such as addition and multiplication, or they can compare column values or join values by referring to other areas in the same data table. Formulas can consist of any JMP Scripting Language (JSL) command. Once you insert a formula into a column, the column is locked: its values can be edited only by changing or removing the formula.

There are three basic steps to building a formula:

1. Open the Formula Editor by right-clicking the column name to which you want to apply the formula and selecting **Formula**.

   or

   Double-click the column name to which you want to apply the formula, select **Formula** from the Column Properties menu, and then click **Edit Formula**.

   In the Formula Editor, the empty formula is selected (highlighted in blue). You can start typing or select a column or function.

   – When you type, a small text editor window appears, which enables you to edit the formula. You can click the **Maximize editor** button if you need more room. Long formulas open in the maximized editor by default.

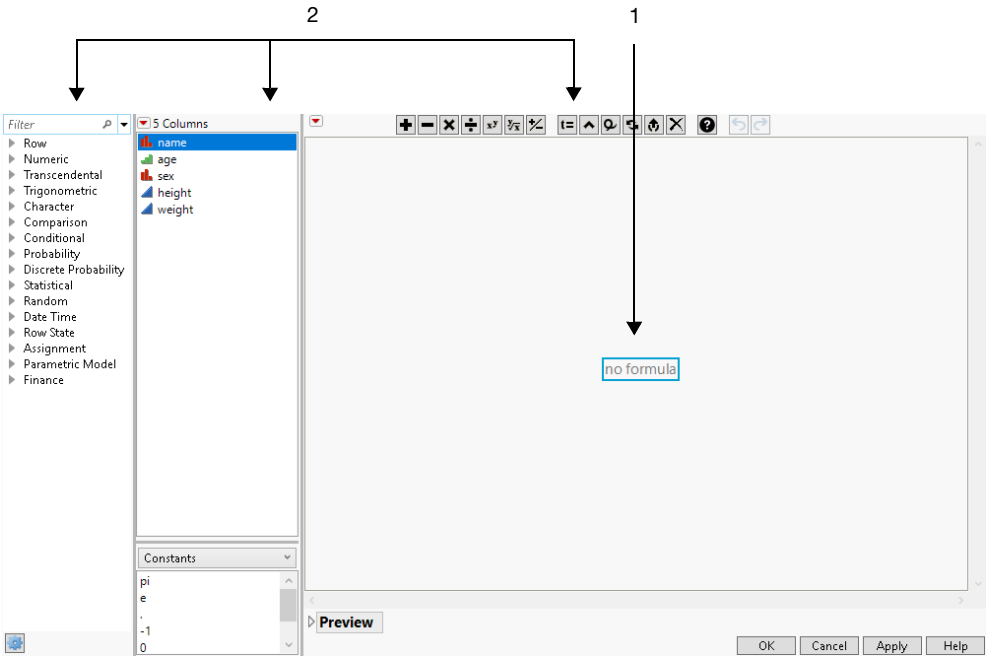
   – When you select a column or function, that item is added to the selected blue box.

   – You can also drag a column from the Columns list into the selected box.

Note: An element is selected when there is a blue outline around it. All terms within the smallest nesting box relative to the place that you clicked become selected. The subsequent actions apply to those combined elements.

2. Add expressions, functions, and terms. Then they appear in the highlighted blue box. The following sections in this chapter provide detailed instructions on how to add constants, elements, operators, and functions.

Figure 7.2 Building a Formula



See "Use Basic Features in the JMP Formula Editor", for an example of how to use the Formula Editor.

Notes:

- In a formula, when you reference a column using value labels, hover over the value label to see the actual data value.
- The Formula Editor recognizes the Excluded state and removes those rows from calculations. Here's an example using the `Col Maximum()` function:

  ```
  Col Maximum( :height, :sex, Excluded( Row State() ) )
  ```

- JMP automatically removes trailing spaces in formulas.
- You can also use the Formula Editor to define a custom format. For example, you might want to convert a number from inches to centimeters using the calculation `Char( :height`

`* 2.54 ) || " cm"`. Realize that the underlying data, which you see when double-clicking a data table cell, would still be the value of :height.

- To move the cursor from one field to another in a formula, select the field and press an arrow key.
- For aggregated functions such as Col Mean, the formula might evaluate faster if you separate the formula into two columns. One column should contain an expression that represents the target for the aggregated function. The second column should contain the aggregated function itself referencing the new target column. Here's an example of the original formula:

  ```
  :Y - Col Mean( If( :Status == "nonsmoker", :Y, . ), :Cycle )
  ```

  In a new Y2 column, create this formula:

  ```
  If( :Status == "nonsmoker",
     :Y,
     .
  )
  ```

  In another column, create this formula:

  ```
  :Y - Col Mean( :Y2, :Cycle )
  ```

Refer to Data Table Values in Formulas

You can create a formula that refers to values found in other parts of the data table, such as other columns and table variables.

When a formula uses values in other columns, the values in the column with that formula are dependent on the values in those other columns. Whenever a column that the formula refers to changes, the dependent column also changes. If you delete the referenced column, empty terms appear in the column containing the formula.

If you create a formula that refers to values found in table variables, those table variables must already exist in the data table. Table variables are character strings that are available to the entire table. Their names are displayed in the table panel at the left of the data table. See "Use Data Table Variables".

Tip: To create a table variable from a value in the Formula Editor, right-click the value and select **Create Table Variable from Value**.

The following example shows how to re-create a formula column in the Companies.jmp sample data table.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Right-click the last empty column and select **New Columns**.

The Column Info window appears.

3. Type “Profit by Sales” next to Column Name.

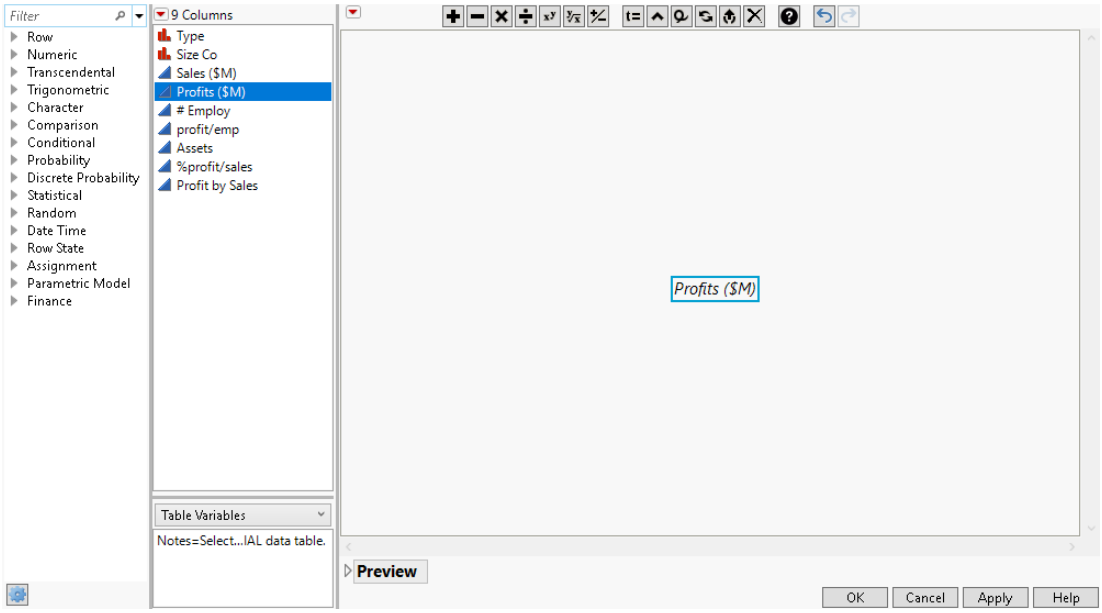
   You are re-creating the formula found in the existing %profit/sales column.

4. In the Column Properties list, select Formula.

   The Formula Editor opens. The blue outline around the box in the editing window indicates that the box is selected.

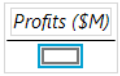
5. With the box selected, select **Profits ($M)** from the Columns list.

Figure 7.3 Adding the Profits ($M) Column



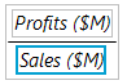
6. With the box selected, click the Divide button ÷.

Figure 7.4 Adding the Division Symbol



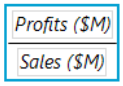
7. With the box selected as shown, select **Sales ($M)** from the Columns list.

Figure 7.5 Adding the Sales ($M) Column



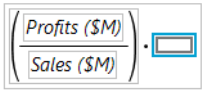
8. Select the outer box in the formula editing area by clicking it.

Figure 7.6 Selecting the Formula



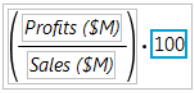
9. Click the Multiply button ×.

Figure 7.7 Adding the Multiplication Symbol



10. With the new box selected, type “100” and press Enter.

Figure 7.8 Multiplying the Formula by 100



11. Click **OK**.

    The column is filled with the calculated numbers.

12. Right-click the last column and select Column Info.
13. In the Column Info window, select **Fixed Dec** from the Format list, type “10” next to Width, and type “2” next to Dec.

    The column width and number of decimal places are specified.

14. Click **OK**.

    The new formula column matches the %profit/sales formula column.

Tips:

- Pressing Alt (Option on macOS) and clicking a column replaces both empty and non-empty selected fields with that column.
- The right-click menu for the Columns list has two replacement options: **Replace all occurrences of selected subexpression** replaces all occurrences of the selected expression with the currently selected column. **Replace the selected subexpression with columns** replaces the selected expression with multiple columns for functions that support it (for example, Sum or Plus).
- Define your own functions and transform columns by writing a JSL script. See the *Scripting Guide*.

Use Local Variables in Formulas

You can create and use temporary numeric variables in expressions. You can use ordinary local variables or you can use parameters, which are special types of local variables. Local variables exist only for the evaluation of the formula in which they are defined. They appear in formulas as bold italic terms.

Local variables are most often used with Assignment functions, which can assign expressions to local variables that are used in a complex equation. See "Assignment Functions". This technique can sometimes simplify building an equation and improve the efficiency of its evaluation.

To build a formula that references values found in local variables, follow these steps:

Step 1: Create the Local Variable

1. Open the Formula Editor by right-clicking a column name in the data table and selecting **Formula**.
2. Select a box in the formula editing area by clicking it. (It is selected when there is a blue outline around it.)
3. In the middle pane of the Formula Editor, select **Local Variables** from the list.
4. Click **New Local**.
5. Enter a name for the local variable. By default, local variables have the names *t0*, *t1*, and so on, and have missing values.
6. Assign a starting value, and click **OK**.
7. (Optional) To copy, edit, or delete a local variable, right-click (Ctrl-click on the macOS) its name and select **Copy**, **Edit**, or **Delete**.

Step 2: Insert a Local Variable into a Formula

1. Select a term in the formula editing area by clicking it. (It is selected when there is a blue outline around it.)
2. Click the local variable name in the Local Variables list. It appears in the formula as a bold italic term.

Notes:

- Double-clicking or pressing Alt and clicking the local variable replaces the portion of the formula that is selected.
- Another way to create local variables is to use the Make Temporary Variable t= button on the Formula Editor keypad. The button automatically creates and displays local variables and places a semicolon after it. See "Add Operators to JMP Formulas".

- You can also create a local variable from an expression. Right-click the expression in the Formula Editor and select **Create Local Variable from Expression**.
- Pressing Alt+Shift and then clicking enables you to edit the selected variable in place instead of in a new window.

See "Use Local Variables Formulas" for an example of referencing local variables in a formula.

Incorporate Parameters in Formulas

Parameters are special types of local variables that are recognized as model parameters in some platforms, such as Nonlinear fitting. They can be used in formulas just as ordinary local variables can.

To view examples of parameters

1. In the sample data folder that was installed when you installed JMP, open the Nonlinear Examples folder and then US Population.jmp.
2. Right-click the column name x-formula and select **Formula**.
3. In the middle pane of the Formula Editor, select **Parameters** from the list.

Notes:

- After completing a nonlinear fit or after clicking the Reset button in the nonlinear control panel, the parameter's value is the most recent value computed by the nonlinear platform.
- Each time the fitting algorithm takes a step, the updated parameter values are shown in the Nonlinear report.
- When you paste a formula with parameters into a column, the parameters are automatically created for that column unless it has existing parameters with the same names.
- Double-clicking or pressing Alt and clicking the parameter replaces the portion of the formula that is selected.
- Pressing Alt+Shift and clicking enables you to edit the selected parameter in place instead of in a new window.

To build a formula that references values found in parameters

Step 1: Create the Parameter

1. Open the Formula Editor by right-clicking a column name in the data table and selecting **Formula**.
2. Make sure the formula is selected in the formula editing area. (It is selected when there is a blue outline around it.)

3. In the middle pane of the Formula Editor, select **Parameters** from the list.
4. Click **New Parameter**.
5. Enter a name for the parameter. By default, parameters have the names *b0*, *b1*, and so on, and have missing values.
6. Assign a starting value. It is important to enter this value when using a parameter in a model for the nonlinear platform. After completing a nonlinear fit, the parameter's value is the most recent value computed by the nonlinear platform.
7. (Optional) To add several parameters (one for each level of a categorical variable, for example) at once, select **Expand into categories, selecting column**. Then select the column for which you want to expand the parameter.
8. Click **OK**.
9. (Optional) To copy, edit, or delete a parameter, right-click (Ctrl-click on the macOS) its name and select **Copy**, **Edit**, or **Delete**.

Step 2: Insert a Parameter into a Formula

1. Select a term in the formula editing area by clicking it. (It is selected when there is a blue outline around it.)
2. Click the parameter name in the Parameters box. The parameter appears in the formula as bold type.

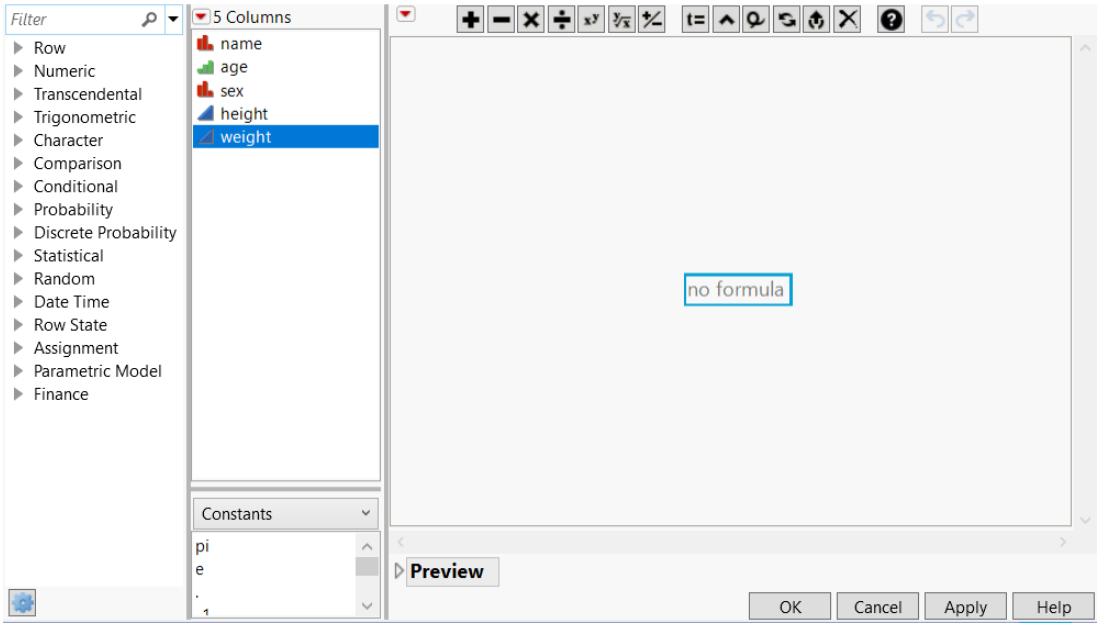
Tip: Pressing Alt+Shift and clicking enables you to edit the selected parameter.

Insert Constants in Formulas

Formulas can be simple assignments of numeric, character, or row state constants, or they can contain complex evaluations based on conditional clauses. Constants include commonly used numeric terms, such as e, pi, -1, 0, 1, and 2. There are two ways to add a constant value to a formula:

- Enter them manually using the keyboard
- Select them from the list in the middle pane, as shown in Figure 7.9.

Figure 7.9 The Formula Element Browser with Constants Selected



To add constants to a formula, follow these steps:

1. Open the Formula Editor by right-clicking a column name in the data table and selecting **Formula**.
2. Select a box in the formula editing area by clicking it. (It is selected when there is a blue outline around it.)
3. Either type in a number or select **Constants** in the middle pane, as shown in Figure 7.9. Then click a value in the list that appears: 0, 1, 2, -1, pi, e. The value appears in the outlined box.
4. Complete the remainder of the formula using the keypad and functions. (See "Refer to Data Table Values in Formulas", "Add Operators to JMP Formulas", and "Use Functions in Formulas".)

Note: Double-clicking or pressing Alt and clicking the constant replaces the portion of the formula that is selected.

Add Operators to JMP Formulas

You can add operators to a formula using the keypad, which contains buttons that help build formulas. It includes common operators (also referred to as functions).

To build a formula using keypad operators at the top of the window, follow these steps:

1. Open the Formula Editor by right-clicking a column name in the data table and selecting **Formula**.
2. Select a box in the formula editing area by clicking it. (It is selected when there is a blue outline around it.) The operator performs its action on the area that is highlighted.
3. Select the column or variable that you want to use in your formula.
4. Click the keypad button(s).

Keypad Reference for the Formula Editor

Table 7.1 describes the keypad buttons.

Table 7.1 Keypad Buttons in the Formula Editor

Name	Description
Arithmetic buttons	Work as they normally do on a pocket calculator, providing addition, multiplication, subtraction, and division operators.
Exponent	Raises a given value to a specified power. It has an exponent of two by default.
Root	Calculates the specified root of the radicand. It has an implied index of two (a square root), which is not displayed.
Unary sign function	Inverts the sign of the argument. Apply the unary sign function to variable expressions or use it to enter negative constants.

Table 7.1 Keypad Buttons in the Formula Editor *(Continued)*

Name	Description
Local variable (Make Temporary Variable)	Creates and displays a local variable and assigns it the value of the selected expression. The local variable has the default name *t0* in an expression and a semicolon after it. See "Use Local Variables in Formulas", for more information about creating and inserting local variables. See "Use Local Variables Formulas" for an example.
Insert	Inserts a new clause or function argument. First select the existing clause or argument that you want the new element to follow, and then click this button. The new clause appears and is selected. You can also insert a new clause or argument by typing a comma. Pressing ALT and the Insert icon inserts a new field before the selected field
Delete	Deletes an element's value, or deletes a clause. The Delete button functions the same as the Delete key on the keyboard.
Switch terms	Looks at the operator that is central to the selected expression and switches the expressions on either side of that operator.
Delete expression (Peel)	Removes the outermost expression with the first argument. You can repeat this process to delete a formula term by term. See "Use the Delete Expression Button Formulas", for an example.
Clear	Clear the entire formula.
Help	Opens the Scripting Index to the entry for the JSL function in the selected part of the formula. This button is applicable only when the selected box in the visual area of the formula is a known function.
Undo	Reverts the last change.
Redo	Redoes the last reverted change.

Use Functions in Formulas

Tip: Hover over a function to learn more about the function.

You can add many types of functions to a formula. All of these functions are organized in the Functions list. The browser groups collections of functions in lists organized by topic. The list contains functions that are commonly used in formulas. Use the Functions list to specify the type of calculation that you want to perform on the elements in a formula.

To create a formula that contains a function, follow these steps:

1. Open the Formula Editor by right-clicking a column name in the data table and selecting **Formula**.
2. Select an expression in the formula editing area by clicking it. (It is selected when there is a blue outline around it.) The function performs its action on the area that is highlighted.
3. In the Functions list, select a group of functions to view. See Table 7.2.

   The functions that belong to that group are then displayed in the list below the menu. The function groups are briefly described in the following list.
4. Click any function in the Functions list to apply it to the selected item. When you click some items, you reveal a submenu from which you should make a selection.
5. Continue to build the formula by highlighting terms and clicking items in the formula element browser, keypad, or Functions list.

Tips:

- Pressing Alt and selecting the function replaces the selected function in a formula with the function that you select from the Functions list or the button that you select from the keypad.
- Pressing Alt (Option on macOS) and clicking a column or other element replaces the item even if it is non-empty.
- Most functions give hints about appropriate arguments through gray words inserted in the boxes in the formula editing area. Functions also show a small caret in the argument area if additional arguments can be added.
- To find out what a function does, right-click the function in the Functions list or in the formula and select **Show in Scripting Index**.
- Define your own functions by writing a JSL script. See the *Scripting Guide*.

Table 7.2 Groups of Functions

Group	Functions Included
Row	Displays a list of functions that contains miscellaneous functions such as Lag, Dif, Subscript, Row, and NRow. See "Row Functions".
Numeric	Displays a list of functions that are terms commonly used in formulas. See "Numeric Functions".
Transcendental	Displays a list of functions that are functions such as natural log, common log, exponential, root, factorial, combinatorial, beta, and gamma. See "Transcendental Functions".
Trigonometric	Displays a list of functions that are the standard trigonometric functions: sine, cosine, tangent, inverse functions, and hyperbolic functions. See "Trigonometric Functions".
Character	Displays a list of functions that operate on character arguments such as trimming, finding the length of a string, converting between numbers and characters. See "Character Functions".
Comparison	Displays a list of functions that are the standard logical comparisons such as less than, less than or equal to, not equal to, and so on. See "Comparison Functions".
Conditional	Displays a list of functions that are programming-like functions, such as If, Match, and Choose. See "Conditional Functions".
Probability	Displays a list of functions that compute probabilities and quantiles for standard statistical distributions, such as normal, Student's *t*, Chi-squared, and *F*-distributions. See "Probability Functions".
Discrete Probability	Displays a list of functions that compute discrete probabilities, such as Poisson, Gamma Poisson, and Hyper geometric. See "Discrete Probability Functions".
Statistical	Displays a list of functions that calculate standard statistical quantities such as the mean or standard deviation. See "Statistical Functions".
Random	Displays a list of functions that generate random numbers based on predefined distributions such as the uniform, normal, Cauchy, and so on. There is also a function to randomize the order of table rows. See "Random Functions".

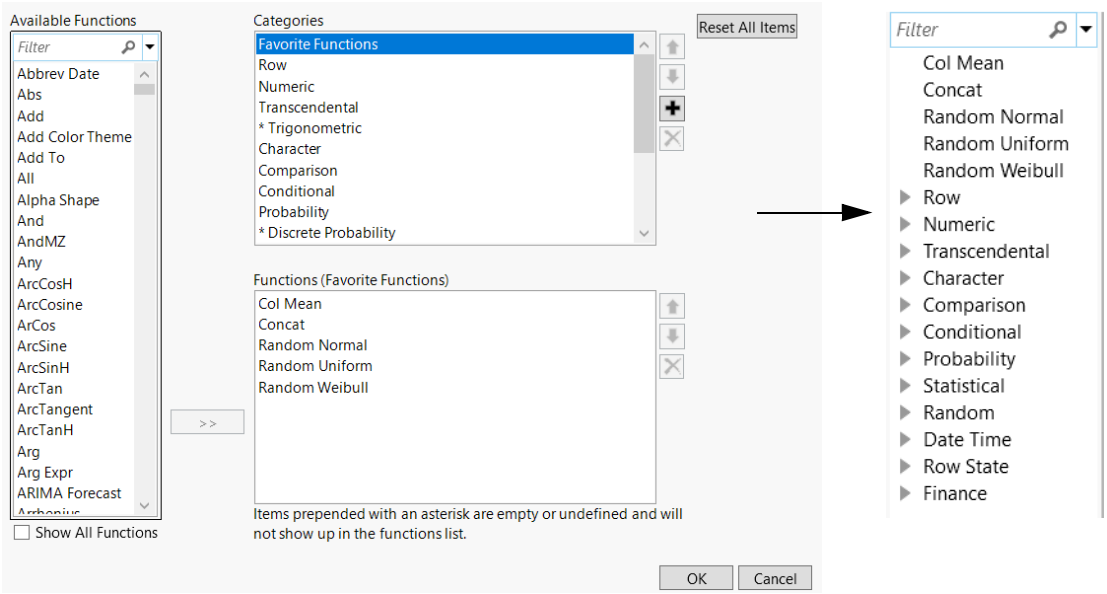
Table 7.2 Groups of Functions *(Continued)*

Group	Functions Included
Date Time	Displays a list of functions that require arguments with the *date* data type, which is interpreted as the number of seconds since January 1, 1904. Date Time functions return values such as day, week, or month of the year. They can also compute dates and can find data intervals. See "Date Time Functions".
Row State	Displays a list of functions that assign or detect row state status of color, marker, label, hidden, excluded, or selected. See "Row State Functions".
Assignment	Displays a list of functions that place the value on the right side of the assignment operator into the variable on the left side of the operator. See "Assignment Functions".
Parametric Model	Lets you construct a Linear, Interactions, or Full Quadratic model for columns that you choose. After choosing the type of model, a window appears that lets you select the variables from which to construct the model. Press Ctrl and click to select more than one. See "Parametric Model Functions".
Finance	Lets you create formulas to calculate principal payments, interest rate, rate of return, and so on. See "Finance Functions".

Find Frequently Used Functions in the Formula Editor

You can customize the Functions list to make it easier to find the functions that you typically work with. The functions appear at the top of the list. Show frequently used functions at the top of the list, create your own function categories, delete functions, delete categories, and move functions up and down in the list.

Figure 7.10 Customizing the Functions List



To quickly bookmark a favorite function

1. Right-click the function in the Functions list.
2. Select **Add as favorite**.

To bookmark a favorite and specify the location

1. Select **Customize Functions List** from the red triangle menu above the formula editor area.
2. Select the function in the Available Functions list.
3. Click the right arrow button to place it in the Functions list.
4. Click the up or down button to rearrange the favorite.

To rearrange or delete favorites

1. Select **Customize Functions List** from the red triangle menu above the formula editor area.
2. Select **Favorite Functions** in the Categories list.
3. Select the favorite function that you want to rearrange and click the up or down button.
4. Select the favorite function that you want to delete and click the X button.

To add functions to a category

If a function is in a formula but the function is not in the Functions list, you can add it to your list by adding it to a category.

1. Select the function that you want to add to a category in the Functions list.
2. Right-click the function and select **Add to Category**.
3. Select the category.
4. Click **OK**.

To rearrange, add, or delete categories

1. Select **Customize Functions List** from the red triangle menu above the formula editor area.
2. In the Categories list, select a category and rearrange, add, or delete the category.

To rearrange or delete functions

1. Select **Customize Functions List** from the red triangle menu above the formula editor area.
2. In the Categories list, select the category that contains the function you want to rearrange or delete.
3. Click the up, down, or X buttons to rearrange or delete the selected function.

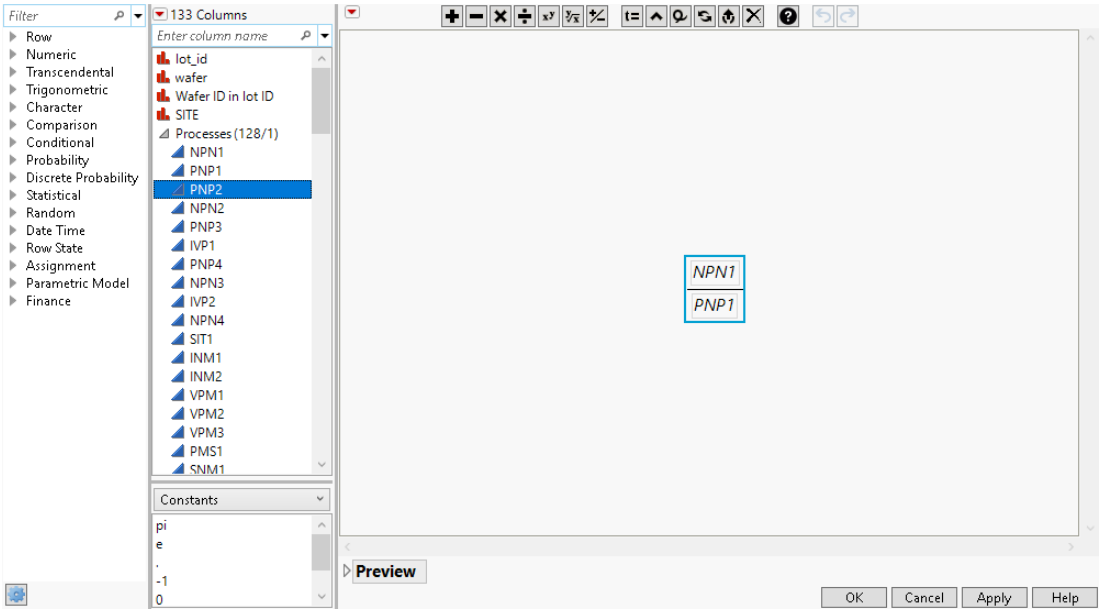
Note: Deleting a function only deletes it from the Functions list. You can still type the function in a formula.

Copy and Replicate Formulas

The Standardize Attributes option in the Cols menu provides a way to replicate a column formula while incrementing each column reference in the formula. This option is an alternative to manually creating new column formulas whose columns increment by one in each of the subsequent columns. For example, if column B contains the formula Ax2, then the standardized column C would contain the formula Bx2, and the standardized column D would contain the formula Cx2.

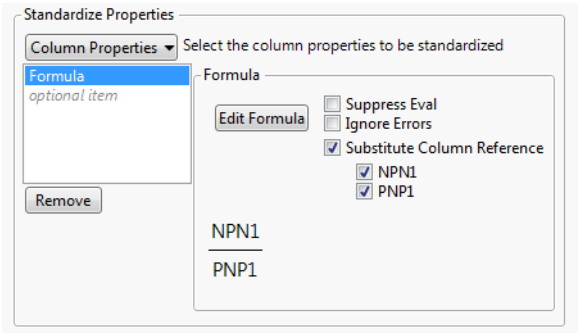
1. Select **Help > Sample Data Library** and open Semiconductor Capability.jmp.
2. Select **Cols > New Columns**.
3. Type 3 next to **Number of columns to add** and click **OK**.
4. Select the first new column, select **Cols**, and then select **Formula**.
5. Create the formula NPN1/PNP1 and click **OK**.

Figure 7.11 Create the Formula NPN1/PNP1



6. Select the three new columns and then select **Cols > Standardize Attributes**.
7. In the Standardize Properties section, select **Formula** from the list.
8. Select **Substitute Column Reference** and leave NPN1 and PNP1 selected.

Figure 7.12 Standardize Properties Options



Note: In this example, we want to change both NPN1 and PNP1. You can deselect the column that you do not want to change in the subsequent formulas.

9. Click **OK**.

   A formula that is relative to the formula in the first new column is copied to the second and third selected columns.

The formulas in the second and third selected columns increment by one column.

- Original formula in the first column: NPN1/PNP1
- Relative formula in the second column: PNP1/PNP2

  In the data table, PNP2 comes after PNP1.
- Relative formula in the third column: PNP2/NPN2

  In the data table, NPN2 comes after PNP2.

Figure 7.13 Resulting Data Table

NPN1	PNP1	PNP2	NPN2	X	Y	Z
114.55576	322.616752	469.39029	115.958473	0.3550831117	0.687310238	4.0479171331
120.043693	333.12808	437.781113	120.974137	0.3603529704	0.7609466694	3.618799212
114.92651	348.978757	532.128147	117.753124	0.3293223673	0.6558171354	4.5190151087
111.756446	268.548094	373.058629	114.092608	0.4161505811	0.7198549341	3.2697879024
111.545085	295.073166	338.900737	113.778098	0.3780251749	0.8706772615	2.9786113754
113.523596	323.833299	469.992171	116.488049	0.3505618367	0.6890184949	4.0346814413

Note: Some columns in Semiconductor Capability.jmp were deleted to show the new columns X, Y, and Z in Figure 7.13.

See "Example of Standardizing a Formula" for another example.

Order Expressions in Formulas

As you build a formula, keep in mind that all functions have an order of precedence shown in the following table, where level one is the highest order of precedence. Expressions with a high order of precedence are evaluated before those at lower levels. When an expression has operators of equal precedence, it is evaluated from left to right. You can use parentheses to override other precedence rules when necessary because any expression within parentheses is always evaluated first. Terms have no order of precedence because they cannot be evaluated further. Table 7.3 shows the first six levels of the order of precedence.

Table 7.3 Order of Precedence of Operators in Formulas

Level 1	Parentheses
Level 2	Functions in the Functions lists, And, Or, Not
Level 3	*, ÷, Modulo
Level 4	+, –
Level 5	Comparisons: <, ≤, =, ≠, ≥, >, ≤ *x* <, < *x* ≤, < *x* <

Table 7.3 Order of Precedence of Operators in Formulas *(Continued)*

Level 6	Logical Operators &, |, and !

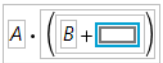
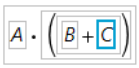
Note: When a function has an expression as its argument, the argument has a higher order of precedence than it would if enclosed in parentheses outside the function.

Build a Formula in Order of Precedence

It is best to build a formula starting with any expression that serves as an argument. This is because functions have a high order of precedence and are always grouped with their corresponding arguments. It is also a good idea to create expressions working from highest to lowest order of precedence when possible. If you need parentheses, be sure to enter the open parenthesis before entering the expression to be enclosed.

For example, given a data table with the columns A, B, and C, use the following steps to compose the expression $A(B + C)$. Note that this expression is not the same as $A \times B + C$, which evaluates as $(A \times B) + C$.

To enter the expression, follow these steps:

1. Click column A in the Columns list.
2. Click the multiplication button in the Formula Editor keypad.
3. Enter an open parenthesis.
4. Click column B in the Columns list.
5. Click the addition button in the Formula Editor keypad. 
6. Click column C in the Columns list. 

Because the order of precedence determines which arguments are affected by each functions, it also affects the grouping of expressions. Select functions in the formula to verify how the order-of-precedence rules have been applied.

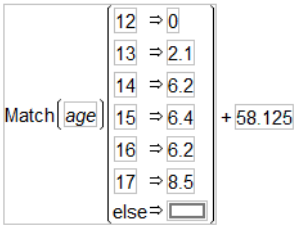
Structure Formulas for Efficient Evaluation

Usually, it is not necessary to structure formulas with efficient evaluation in mind. Most formulas evaluate almost instantaneously regardless of their structure. This is because statistical functions and constant expressions are evaluated only once when a column's values are calculated.

However, when you are creating conditional expressions, keep in mind that Match evaluates faster and uses less memory than an equivalent Condition function, If. (Note that Match ignores trailing spaces and If does not.)

For example, using Big Class.jmp, you can predict a child's height from his age as shown in Figure 7.14. There is a base height of 58.125 inches to which a quantity is added depending on the value of the age variable.

Figure 7.14 The Match Conditional Evaluates Faster Than the If Function



The Match conditional evaluates faster than the If function because the age variable is evaluated only once for each row in the data table. The If condition must evaluate the age variable at each If clause for each row until a clause evaluates as true.

JMP Formula Editor Options

The Formula Editor provides robust options for creating and managing formulas.

- "Calculate Derivatives of JMP Functions"
- "Simplify Complex Formulas"
- "Evaluate Formulas"
- "Ignore Errors in Formulas"
- "View Formulas in JSL"

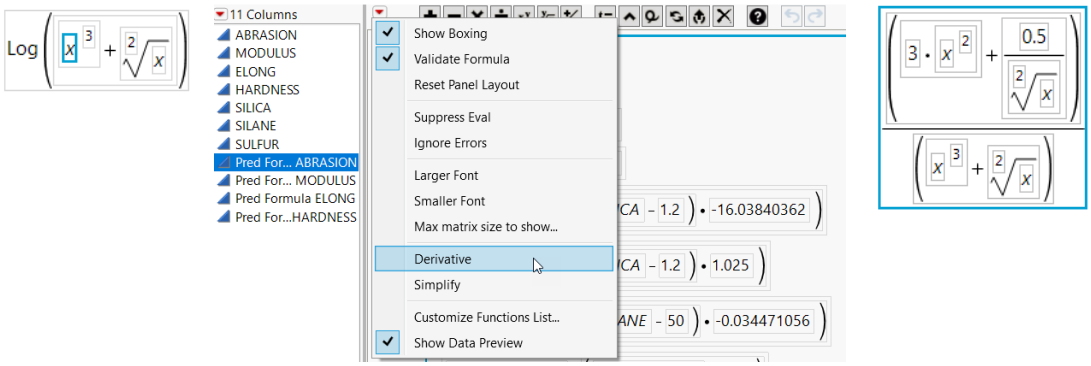
Calculate Derivatives of JMP Functions

The JMP Formula Editor can find and display the derivative of a function. The derivative is found with respect to the function argument (a single variable name) you highlight. Therefore, in order to differentiate with respect to x, x must be one of the arguments in the expression. The red triangle menu next to the keypad contains the Derivative command.

To calculate a derivative, follow these steps:

1. Enter a function.
2. Highlight a variable.
3. Select **Derivative** from the menu.

Figure 7.15 Using the Derivative Option



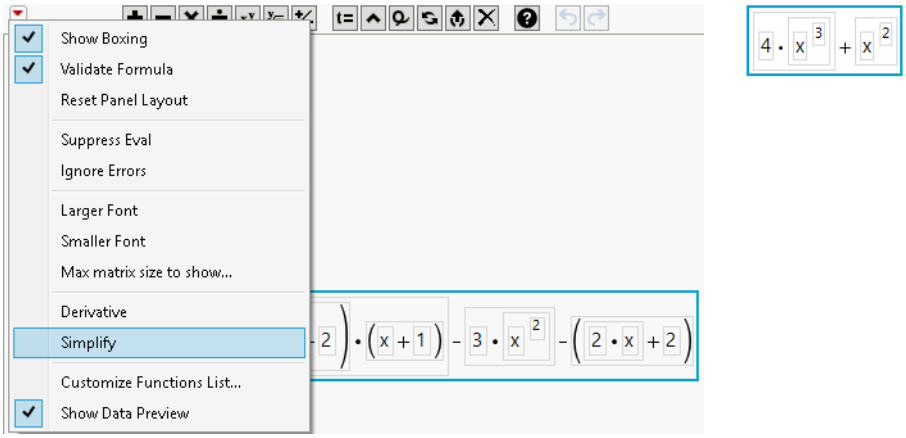
Simplify Complex Formulas

When the Formula Editor contains a complex formula, JMP can simplify it using various algebraic rules. It can find constant expressions, distribute multiplication over addition, combine terms, and more.

To simplify complex formulas, follow these steps:

1. Select a portion or all of the formula.
2. Click the red triangle next to the keypad.
3. Select **Simplify** from the list as shown in Figure 7.16.

Figure 7.16 Selecting Simplify (Left) Produces a Simplified Formula (Right)



Evaluate Formulas

By default, JMP evaluates each formula that you create. You can turn this evaluation off, or you can use it before you have finished creating a formula.

Suppress Evaluation

Turning off evaluation is a useful formula development mode for building complex formulas. You can turn off evaluation and build sections of a formula, and evaluate only to test it. In particular, you can close the Formula Editor and reopen it at a later time to continue building a formula without JMP evaluating it.

To suppress formula evaluation, click the red triangle next to the keypad and select **Suppress Eval**. When evaluation is suppressed, the formula icon appears dimmed.

Note the following:

- If the icon appears to the right of the red triangle in the Formula Editor, it indicates that formula evaluation is suppressed for that formula.
- If the icon appears beside the column name in the Columns panel, it indicates that the values in the column result from a formula. When formula evaluation is suppressed, this formula icon appears dimmed. See "Icons Representing Column Characteristics and Properties".
- To turn off formula evaluation for all formulas, click the red triangle next to the data table name and select **Suppress Formula Eval**. To re-evaluate formulas, click the red triangle next to the data table name and select **Rerun Formulas**. You can also click the **Apply** button in the Formula Editor for each column that contains a pending formula.
- When you turn off formula evaluation, edit a formula, and click **Apply**, you are prompted for whether to evaluate the formula. To allow formula evaluation once, click **Evaluate formula this time**.

Ignore Errors in Formulas

Once you construct a formula and click **OK**, JMP checks behind the formula for error and alerts you of any errors that it finds. An error message appears for each error and asks whether you want to ignore further errors.

Sometimes you might want to suppress error messages while a formula is under development. For example, you might want the evaluated values for some rows without seeing an error message for each row that causes errors.

To ignore errors, follow these steps:

1. Create a formula.

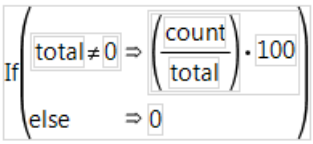
2. Click the red triangle next to the keypad and select **Ignore Errors**.

View Formulas in JSL

You have the option of entering or changing any part of a formula in *text mode*. Text mode displays the formula in the JMP Scripting Language (JSL). The entire formula (or any of its terms) appears in the Script Editor when you double-click the formula. If the formula is long, it opens in a maximized Script Editor window. Otherwise, you can click the **Maximize editor** button to maximize the window.

Any element of a formula can be displayed in the Script Editor and then edited. After editing formula scripting commands, click outside the formula (or click **OK** below the maximized window) to see its formatted form and to save the change. To enter an If statement in text form, add pairs of arguments for each If/Then clause in the statement, and a single last argument for the else clause if needed. In text form, the If statement in Figure 7.17 looks like this: `If(:total!=0, (:count/:total)*100, 0)`.

Figure 7.17 An If Statement in Formula Mode



Edit Formulas

Edit formulas by using the keypad; using arrows to select expressions; cutting, copying, and pasting; deleting functions, and dragging expressions.

- "Correct Mistakes in Formulas"
- "Select Expressions in Formulas"
- "Delete Functions Formulas"
- "Cut, Copy, and Paste Formulas"
- "Click and Drag Formulas"

Correct Mistakes in Formulas

If you make a mistake while entering a formula in the JMP Formula Editor, press Ctrl+Z. This reverses the effect of the last (undo-able) command.

Other commands to help modify formulas include the following:

- Click the delete button ( ) on the Formula Editor keypad to remove the selected expression.
- Use the cut, copy, and paste keyboard shortcut commands or right-click a highlighted part of the formula and select **Cut**, **Copy**, or **Paste** from the menu.
- Drag portions of the formula to rearrange them.

Select Expressions in Formulas

Use the keyboard arrow keys to select expressions for editing in the JMP Formula Editor. You can also use the arrow keys to view the formula's order of precedence when either parentheses or the boxing option are not present. (See "Hide and Show Outlines Formulas".)

Clicking an operator (+, –, *, ÷) in an expression selects the operator and its operands. A blue box appears around the items. Once an operator is selected:

- The left and right arrow keys move the selection across other associative operators having equal precedence within the expression.
- The up arrow extends the current selection by adding the operand and operator of higher precedence to the selection.
- The down arrow reduces the current selection by removing an operand and operator from the selection.

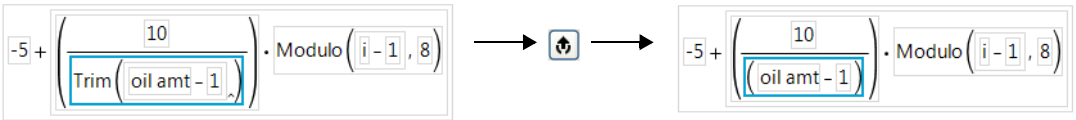
Delete Functions Formulas

Deleting a function also deletes its arguments; deleting a required argument or missing term from a function sometimes deletes the function as well. In the JMP Formula Editor, you can peel a function to delete it from its argument.

To peel a function from a single argument, follow these steps:

1. Select the function.
2. Click the peel button in the Formula Editor keypad. Or use the hand tool to drag the argument on top of its function.

Figure 7.18 Peeling an Argument



3. Complete formula changes.

4. Click **Apply**, and the new values fill the data table column automatically.
5. Once you have created a formula, you can change values in columns that are referenced by your formula. JMP automatically recalculates all affected values in the formula's column.

Cut, Copy, and Paste Formulas

You can cut or copy any expression or an entire formula and paste it into another formula in the JMP Formula Editor. Use the cut, copy, and paste shortcut commands or right-click a highlighted part of the formula and select **Cut**, **Copy**, or **Paste** from the menu. The following aspects apply when you cut, copy, and paste a formula:

- When you paste it into another formula display, the formula appears in formatted form.
- The formula is saved on the clipboard as a JSL statement. Thus, if you copy it into other applications, it appears as a JMP Scripting Language (JSL) statement.

Note: Press Shift and click the red triangle to reveal a command called **Copy As SAS Formula**.

Click and Drag Formulas

In the JMP Formula Editor, you can drag any part of a formula that can be selected to any other location that can be selected.

To click and drag in a formula, follow these steps:

1. Place the arrow cursor inside an expression.
2. Click the expression.
3. Drag across the formula. Destination expressions have a dotted outline.

Figure 7.19 Dragging an Expression

4. Drag the selected expression to the new desired location. The selected expression is moved to the new location, where it replaces the existing expression.

Customize Formulas in the JMP Formula Editor

There are several ways that you can customize formulas in the JMP Formula Editor.

- "Hide and Show Outlines Formulas"
- "Change the Font Size Formulas"
- "Specify the Matrix Size Formulas"
- "Change the Orientation of a JMP Formula"
- "Open and Close Arguments Formulas"

Hide and Show Outlines Formulas

By default, JMP outlines specific terms within the formula. This is called *boxing*. Boxing is useful when you want to select and modify a specific portion of a formula, or need to determine the order of evaluation that takes place.

To turn boxing on or off, follow these steps:

1. Build a formula.
2. Click the red triangle next to the keypad and select **Show Boxing**. When a check appears on the menu beside **Show Boxing**, the outline appears in the formula. When it does not, the outline does not appear.

Change the Font Size Formulas

To incrementally increase or decrease the font used to display the formatted formula, follow these steps:

1. Click the red triangle next to the keypad.
2. Select **Larger Font** or **Smaller Font**.
3. Repeat this process to further increase or decrease the font size.

Specify the Matrix Size Formulas

You can set the maximum width and height of matrices by selecting **Max matrix size to show** from the red triangle menu. If the width or height is greater than the specified size, the matrix dimensions show instead of the data. You can still see the data by editing the formula in the text editor window.

Change the Orientation of a JMP Formula

By default, JMP determines the size of a formula and displays it in the Formula Editor in the best orientation (horizontally or vertically). However, if you create a long formula, you might want to display it in a different layout.

To change a formula's orientation, follow these steps:

1. Build a formula.
2. Highlight an argument or formula.
3. Right-click what you have highlighted.
4. Select **Orientation**.
5. Select from the Orientation options: Best, Horizontal, or Vertical.

Open and Close Arguments Formulas

When a formula is too large to fit on the screen, you can close formulas and arguments.

To close an argument, follow these steps:

1. Build a formula.
2. Highlight an argument or formula.
3. Right-click what you have highlighted.
4. Select **Close** or **Close Arguments** from the menu that appears.

Examples of Building Formulas in the JMP Formula Editor

To better familiarize yourself with building formulas, review the following examples and tutorials.

- "Use Basic Features in the JMP Formula Editor"
- "Use Local Variables Formulas"
- "Use the Munger Function Formulas"
- "Use the Match Conditional Function Formulas"
- "Use the Delete Expression Button Formulas"

Use Basic Features in the JMP Formula Editor

The following example uses the Big Class.jmp sample data table to walk you through using the basic features of the Formula Editor. You can find Big Class.jmp in the sample data folder, which was installed when you installed JMP.

Big Class.jmp has a column called weight. Suppose you want a new column that computes standardized weight values.

To create this column using a formula to obtain its values, follow these steps:

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Cols > New Columns**.
3. Type the new name, standard weight, in the box beside Column Name.
4. Select **Formula** from the Column Properties menu.
5. Click the empty formula element in the formula editing area to select it.

   When you create a formula and Show Boxing is checked, the selected portion of the formula is outlined with a thin blue line. See "Hide and Show Outlines Formulas". All terms within the smallest nesting box relative to the place that you clicked become selected, and the subsequent action applies to those combined elements.

Next, enter the formula that standardizes the weight values.

1. While the initial missing term is selected, click weight in the Columns list.
2. Click the minus button ➖ in the Formula Editor keypad.

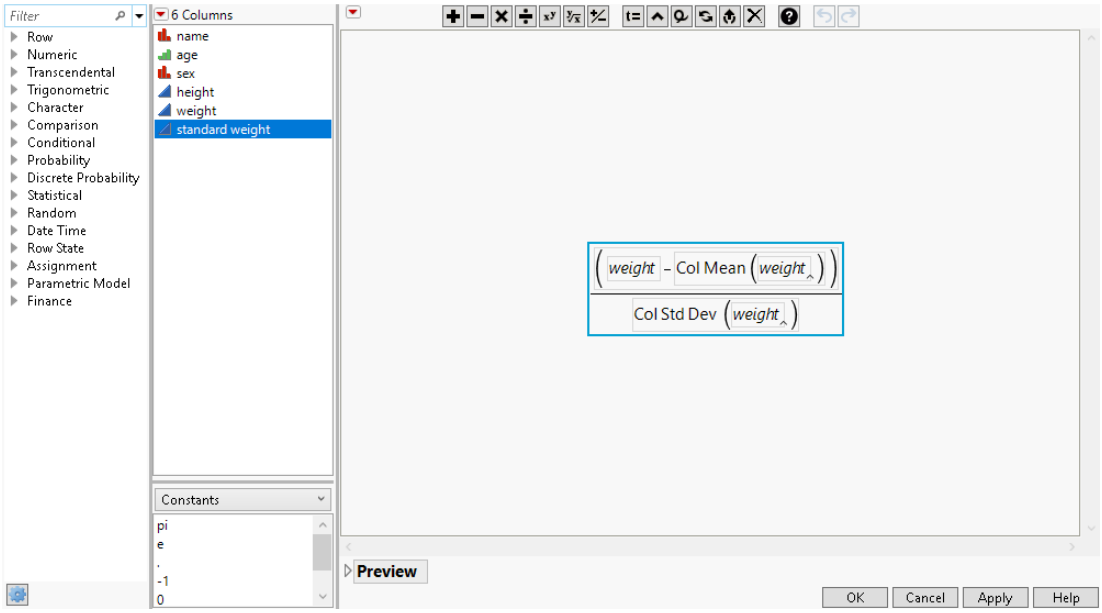
   A new missing term appears after the minus sign as shown in Figure 7.20.
3. Click weight again.
4. Click the Functions list, and select **Statistical**.
5. Select **Col Mean** from the Statistical list.
6. Select the entire expression.

   The blue box should now enclose the whole formula.
7. Click the division button ➗ in the keypad.

   The result gives a selected missing denominator for the whole expression.
8. Click weight again from the column selector list.

   It becomes selected in the denominator.
9. Select **Col Std Dev** from the Statistical list.

Figure 7.20 Building a Formula



10. Close the Formula Editor by clicking **OK**.
11. Close the Column Info window by clicking **OK**.

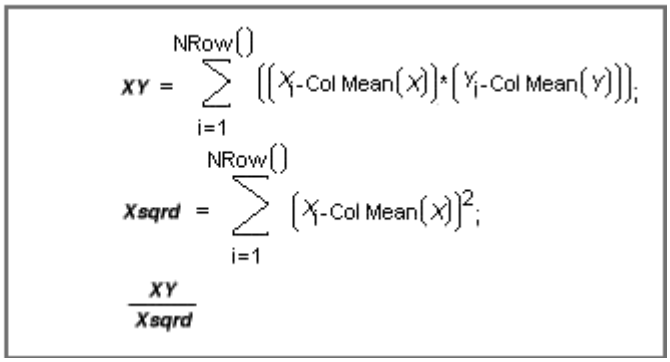
In the data table, the new Standard Weight column fills with values. When a weight value changes, the calculated Standard Weight value automatically recalculates.

Use Local Variables Formulas

Suppose you want to compute the slope in a simple linear regression of *y* on *x* using the standard formula shown in Figure 7.21. One way to do this is to create two local variables, and name them XY and Xsqrd. (See "Use Local Variables in Formulas".) Then assign them to the numerator and the denominator calculations of the slope formula. Delimit each assignment with a semicolon, as shown in Figure 7.21. (Statements in the Formula Editor are actually JSL programming statements. Multiple statements in a formula must be separated by a semicolons.) The slope computation is simplified to XY divided by Xsqrd.

Figure 7.21 Local Variables in a Standard Slope Formula

$$\frac{\sum (x-\bar{x})(y-\bar{y})}{\sum (x-\bar{x})^2}$$



Note: You can also create local variables using the t= button on the on-screen keypad. Clicking this button creates a local variable with a default name in an expression and places a semicolon after it. The default name is *t1*, and additional local variables are named *t2*, *t3*, and so on. You can change these default names by double-clicking and editing the name. However, you must be careful to rename each instance of the variable to avoid errors.

Use the Munger Function Formulas

The following examples show uses of the Munger function. In these examples, assume that there is a character column of names with “Veronica Layman” as one of its values. To simplify the examples, the literal name “Veronica Layman” is the search string instead of a column name.

For instructions on how to incorporate Character functions, such as Munger, into a formula, see “Character Functions”.

Insert Characters

This Munger example finds the blank between the first and last name, and inserts the middle initial “J.” The formula **Munger("Veronica Layman", 1, " ", " J. ")** inserts the middle initial J., and evaluates as Veronica J. Layman.

Double quotation marks are required by the Munger function for literal strings, including strings that consist of a blank or when leading or trailing blanks are part of a string.

Delete Characters

To delete one or more characters from a string, follow these steps:

1. Designate the characters to delete as the Find string in the Munger function.

2. Enter an empty Replace string: two quotation marks with nothing between them.

For example, the function, **Munger("Veronica Layman", 1, "onic", "")** removes the "onic" from Veronica and evaluates as Vera Layman.

Note: A Replace field with a null (no value) string enclosed in quotation marks is different from a Replace field with no value. If you delete the Replace string altogether, Munger shows the argument name ("Replace") in the Formula Editor window and behaves as if that optional argument does not exist. The resulting data type can also change from character to numeric, depending on the value of the Find/Length argument.

Find the Position (Index) of a Value

When the Find/Length field contains characters, Munger behaves like an index function and returns the numeric position of the first instance of the search string if it exists. For example, **Munger("Veronica Layman", 1, " ")** searches for a single blank and finds it in position nine. If the search string is not found, Munger returns a zero. This use of Munger produces the same result as the Contains function, as shown in "Character Functions".

Find a Substring

Munger can extract substrings. For example, to extract only the first name, **Munger("Veronica Layman", 1, 8,)** starts at position one and reads through position eight. The remaining characters are ignored because the replace argument is not defined. This yields "Veronica" and produces the same result as the Substring, as shown in "Character Functions".

An alternative way to find a substring is with a start value, any negative find value, and a **no replace** argument. **Munger("Veronica Layman", 9, –1,)** returns "Layman".

Use the Match Conditional Function Formulas

This example walks you through using the Match conditional function.

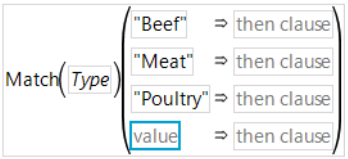
Suppose that you want a Match conditional for the nominal variable Type from the Hot Dogs.jmp sample data table.

1. In the Formula Editor, select Type from the Columns list.
2. Select **Conditional** from the Functions list.
3. Select **Match**.
4. Select **Add Match Arguments from Data**.

   The values are automatically filled in.

Note: Rather than complete step 2 through step 4, press Shift, select **Conditional** from the Functions list, and then select **Match**.

Figure 7.22 Automatically Filling a Match Conditional Statement



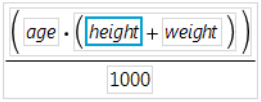
If you do not want the values filled in for you, select **Do Not Add** from the Match list instead of **Add Arguments from Data**.

Use the Delete Expression Button Formulas

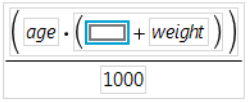
The following steps illustrate the results of repeatedly clicking the delete expression (or peel) button :

1. Start with a formula.

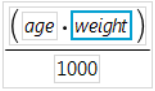
2. Select a formula element.



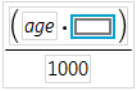
3. Click the delete expression button .



4. Click the delete expression again.



5. Click the delete expression again.

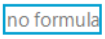


6. Click the delete expression again.

7. Click the delete expression again.

8. Click the delete expression again.

9. Click the delete expression again.



Keyboard Shortcuts for the JMP Formula Editor

Table 7.4 describes the keyboard shortcuts that you can use in the JMP Formula Editor.

Table 7.4 Keyboard Shortcuts in the Formula Editor

Insert This Item	Using This Keyboard Shortcut
A missing element (•)	Delete
Subscript	[
Exponent	^ (Press Shift and type 6.)
Set of parentheses ()	(
Set of quotation marks " "	"
New argument before the selected field	Alt + ^
* (multiplication symbol)	* on keypad or keyboard
+	+ on keypad or keyboard
−	– on keypad or keyboard
÷	/ on keypad or keyboard

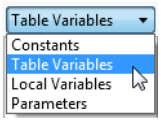
Table 7.4 Keyboard Shortcuts in the Formula Editor *(Continued)*

Insert This Item	Using This Keyboard Shortcut
+/−	Press Shift+- (Minus)
And	&
Not	!
Or	|
Power	^
New argument	,
x<y	<
x==y	=
x>y	>

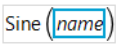
Glossary of Terms for the JMP Formula Editor

The following terms are used in the Formula Editor:

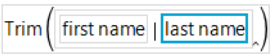
Element The name of a constant, table variable, local variable, or parameter that appears in the element browser list.



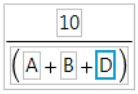
Argument Any element or an entire expression (including mathematical operands) that is operated on by a function. Arguments are always grouped with functions. To find which expressions serve as a function's arguments, select that function in the formula.



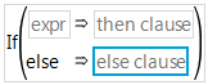
Term Indivisible parts of an expression, such as constants and variables.



Expression Any part of a formula that can be selected as a single unit, including terms, missing terms, and functions grouped with their arguments, as well as the entire formula.



Clause A complete segment in a conditional function.



Function A mathematical or logical operation that performs a specific action on one or more arguments. Functions include most items in the Functions list and all keypad operators. Functions always operate upon selected expressions, and arguments are always grouped with functions. To find which expressions serve as a function's arguments, select that function in the formula.

The boxed groupings also show how order-of-precedence rules apply and show which arguments are deleted when you delete a function. See "Order Expressions in Formulas".

Missing term Any empty place holder for an expression, represented by an empty box.

Missing value Excluded or null data consisting of the missing value mark (•) for numeric data or null character strings for character data.

	name	age	sex	height	weight
1	KATIE	•	F	59	95
2		12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84

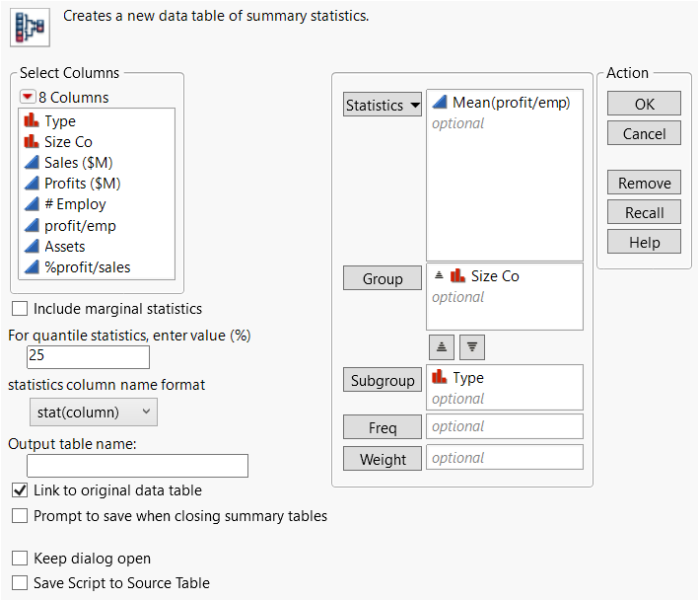
Chapter 8

Summarize Your Data

The Table Summary Command

This chapter describes how to create a summary data table, which includes summary statistics such as the mean and median, standard deviation, and minimum and maximum values.

Figure 8.1 Summary Table for Companies.jmp



	Size Co	N Rows	Mean(profit/emp, Computer)	Mean(profit/emp, Pharmaceutical)
1	big	9	4530.478	17140.699
2	medium	7	-3462.506	24035.115
3	small	16	7998.815	38337.191

Contents

Summarize Columns 433
Summary Launch Window 433
Create a Summary Table 433
Add a Statistics Column to an Existing Summary Table 437
Explanation of Summary Statistics 438
Example of Creating a Summary Table with Groups 440

Summarize Columns

The Tables > Summary command calculates various summary statistics, including the mean and median, standard deviation, minimum and maximum value, and so on.

Summary tables have the following characteristics:

- A single row exists for each level of a grouping variable that you specify. If no grouping variable is specified, a single row exists for the full data table.
- When there are several grouping variables, the table contains rows for each combination of levels of all the grouping variables.
- In addition to one column for each grouping variable, the table contains frequency counts in a column named N Rows with counts for each grouping level.
- The summary table can be linked to its source table. When you select rows in the summary table, the corresponding rows are highlighted in its source table.
- If the source table's column(s) contain value labels, the value labels are displayed in the new table.
- A summary table is not saved when you close it unless you select **File > Save As** to give it a name and location.

Summary Launch Window

Statistics Defines the summary statistics (such as mean, standard deviation, and median) for a numeric column in the source table. See "Add Summary Statistics".

Group Summarizes the statistics by group. See "Use One or More Grouping Columns".

Subgroup Summarizes the statistics by subgroup. See "Create a Two-Way Table of Summary Statistics by Adding a Subgroup Variable".

Freq Identifies a column whose values assign a frequency to each row.

Weight Identifies a column whose values assign a weight to each row.

Create a Summary Table

To create a summary table, follow these steps:

1. Open a data table.
2. Select **Tables > Summary.**
3. Highlight the columns that you want to summarize.

Note: For more information about the options in the red triangle menu, see "Column Filter Menu".

4. Add summary statistics, groups, subgroups, frequency variable, weight variable, and select any options needed:
   - "Add Summary Statistics"
   - "Use One or More Grouping Columns"
   - "Use Quantile Statistics"
   - "Change the Format of the Statistics Column Name"
   - "Link to the Original Data Table"
   - "Prompt to Save When Closing Linked Summary Tables"
   - "Keep the Summary Window Open"
   - "Create a Two-Way Table of Summary Statistics by Adding a Subgroup Variable"
5. Name the summary table by typing a name in the box beside **Output table name**.
6. Click **OK**.

Notes:

- When the summary table is linked to the original data table, you cannot edit the data in the summary table, because that would modify and compromise the original data. To edit the data, create a subset of the summary table or save the table, close the table, and reopen it.
- When you select a column that has a Units column property, that value is included in the column header.

Add Summary Statistics

You can add columns that display summary statistics (such as mean, standard deviation, median, and so on) for any numeric column in the source table.

1. In the Summary window, highlight the column that you want to use in calculating the statistics.
2. Click the **Statistics** button.
3. Select one of the standard univariate descriptive statistics from the Statistics list. The statistics are described in "Explanation of Summary Statistics".

Use One or More Grouping Columns

If you want the statistics summarized by group, highlight the column(s) that you want to be your grouping variables and click **Group** to move the variable into the grouping variables list. See "Example of Creating a Summary Table with Groups" for an example. If you add only grouping variables, the summary table shows a count for each group.

To change the order of the grouping variables

To change the order of the grouping variables (ascending or descending order), select a variable in the grouping variable list and click the ascending or descending button ( ). The icon beside the variable changes to indicate the sorting order.

You can also change the order of the grouping variables using the Value Order column property. See "Value Order".

To include marginal statistics

To add marginal statistics (for the grouping variables) to the output columns, click the box beside **Include marginal statistics**. In addition to adding marginal statistics for each grouping variable, JMP adds rows at the end of the table that summarize each level of the first grouping variable.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Tables > Summary**.
3. Select Profits ($M) and click **Statistics**.
4. Select **Mean**.
5. Select Type and Size Co and click **Group**.
6. Select **Include marginal statistics**.
7. Click **OK** (or **Create**). See Figure 8.2 at left.

Figure 8.2 Summary Table with and without Marginal Statistics

	Type	Size Co	N Rows	Mean(Profits ($M))
1	Computer	big	4	1089.93
2	Computer	medium	2	-85.75
3	Computer	small	14	44.94
4	Computer		20	240.87
5	Pharmaceutical	big	5	894.42
6	Pharmaceutical	medium	5	698.98
7	Pharmaceutical	small	2	156.95
8	Pharmaceutical		12	690.08
9			32	409.32

	Type	Size Co	N Rows	Mean(Profits ($M))
1	Computer	big	4	1089.93
2	Computer	medium	2	-85.75
3	Computer	small	14	44.94
4	Pharmaceutical	big	5	894.42
5	Pharmaceutical	medium	5	698.98
6	Pharmaceutical	small	2	156.95

Compare the summary table with marginal statistics (at left) to the summary table without marginal statistics (at right). You can see that the marginal statistics are added, and a row showing that there are 32 total Computer and Pharmaceutical companies.

Use Quantile Statistics

To add specific quantile statistics, follow these steps:

1. In the box under **For quantile statistics, enter value (%)** type the desired quantile value (%) for the first quantile (for example, 25).
2. Select the applicable column and click **Statistics**.
3. Select **Quantiles**.
4. (Optional) Repeat this process for any additional quantiles.

Change the Format of the Statistics Column Name

To change the format of the statistics column name in the summary table, select from one of the formats in the **statistics column name format** menu. Table 8.1 illustrates the available options. Assume that you are creating a summary table of the mean profits for a company. Your original column name is Profits ($M).

Table 8.1 Statistics Column Name Format Options and Examples

Option	Example
stat (column)	Mean (Profits ($M))
column	Profits ($M)
stat of column	Mean of Profits ($M)
column stat	Profits ($M) Mean
stat	Mean

Link to the Original Data Table

You can select whether to link the summary table to the original data table. By default, the **Link to original data table** option is selected. If you want to edit the data in the summary table, deselect the **Link to original data table** option. When the summary table is linked to the original data table, you cannot edit the data in the summary table, because that would modify and compromise the original data. To edit the data, create a subset of the summary table or save the table, close the table, and reopen it.

Within linked tables, if you drag columns from the summary table into the column heading of a new column in the original data table, the values are expanded as if they were matched by grouping columns.

Prompt to Save When Closing Linked Summary Tables

To be prompted to save an unsaved linked summary table when you close it, select **Prompt to save when closing summary tables**. When you select this option in the Summary window, it is selected in the Tables preferences. If you do not want to be prompted to save linked summary tables in the future, deselect the preference.

Keep the Summary Window Open

If you select the **Keep dialog open** option, the Summary window remains open after you click **Apply**. Notice that once you select this option, the OK button is replaced by a Create button.

Create a Two-Way Table of Summary Statistics by Adding a Subgroup Variable

1. Highlight the column(s) that you want to be the nested variable(s). These are your "subgroup variable(s)."
2. Click **Subgroup** to move the variable(s) into the subgroup list.
3. Highlight the column for which you want statistics summarized by subgroup.
4. In the Statistics list, select the specific statistic that you want.
5. Click **OK**.

For more information about the types of statistics, see "Explanation of Summary Statistics".

Add a Statistics Column to an Existing Summary Table

After you have created a summary table, you can add columns of descriptive summary statistics for any numeric column in the source table. To do so, from an existing summary table, click the upper red triangle in the data grid and select **Add Statistics Column**.

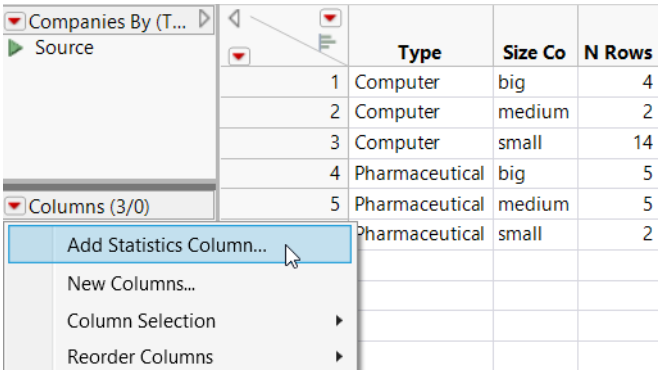
Example of Adding a Statistics Column to an Existing Table

Suppose that you have already created a summary table, and you want to add more statistics to the existing summary table.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Tables > Summary**.
3. Select Type and Size Co and click **Group**.
4. Click **OK**.

5. Click the red triangle in the Columns panel, select **Add Statistics Column**.

Figure 8.3 Creating a Summary Statistics Column from Within a Data Table



	Type	Size Co	N Rows
1	Computer	big	4
2	Computer	medium	2
3	Computer	small	14
4	Pharmaceutical	big	5
5	Pharmaceutical	medium	5
	Pharmaceutical	small	2

A modified version of the Summary window appears.

6. Select the column that you want, click **Statistics**, and select the specific statistic that you want. For this example, select profit/emp and click **Statistics**, and then select **Mean**.
7. Click **OK**.

Figure 8.4 Example of a Summary Table with a Summary Statistics Column

	Type	Size Co	N Rows	Mean(profit/emp)
1	Computer	big	4	4530.478
2	Computer	medium	2	-3462.506
3	Computer	small	14	7998.815
4	Pharmaceutical	big	5	17140.699
5	Pharmaceutical	medium	5	24035.115
6	Pharmaceutical	small	2	38337.191

Summary statistics column

The Mean(profit/emp) column is added to the existing summary table.

Explanation of Summary Statistics

You can add columns of descriptive summary statistics for any numeric column in the source table by clicking the Statistics button and making a selection from the menu.

The Statistics menu gives these summary statistics for numeric columns:

N The number of nonmissing values.

Mean The arithmetic mean of a column's values. It is the sum of nonmissing values (and if defined, multiplied by the weight variable) divided by the Sum Wgt.

Std Dev The sample standard deviation, computed for the nonmissing values and calculated with the specified weight variable. It is the square root of the sample variance. The formula follows:

$$s = \sqrt{s^2} \text{ where}$$

$$s^2 = \sum_{i=1}^{N} \frac{w_i(y_i - \bar{y}_w)^2}{N-1}$$

$$\bar{y}_w = \text{weighted mean}$$

Min The smallest nonmissing value in a column.

Max The largest nonmissing value in a column.

Range The difference between Max and Min.

% of Total The percent of the total count for each group. Or, if you have so specified, the percent of nonmissing values of the column to the total count for each group.

N Missing The number of missing values.

N Categories The number of distinct categories.

Sum The sum of all values in a column.

Sum Wgt The sum of all weight values in a column. (See "Column Properties in JMP".) Or, if no column is assigned the weight role, **Sum Wgt** is the total number of nonmissing values.

Variance The sample variance, computed for the nonmissing values. It is the sum of squared deviations from the mean, divided by the number of nonmissing values minus one.

Std Err The standard error of the mean. It is the standard deviation divided by the square root of **N**. If a column is assigned the role of weight, then the denominator is the square root of the sum of the weights.

CV (Coefficient of Variation) The measure of dispersion, which is the standard deviation divided by the mean multiplied by one hundred.

Median The 50th percentile, which is the value where half the data are below and half are above or equal to the 50th quantile (median).

Median Absolute Deviation The median of the absolute deviation from the data's median.

Geometric Mean The *n*th root of the product of the data. For example, geometric means are often used to calculate interest rates. The statistic is also helpful when the data contains a large value in a skewed distribution.

Note: Negative values result in missing numbers, and zero values (with no negative values) result in zero.

Interquartile Range The difference between the third and first quartiles.

Quantiles the value at which the specific percentage of the argument is less than or equal to. For example, 75% of the data is less than the 75th quantile. The summary window has an edit box for entering the quantile percentage that you want.

Histogram Generates histograms for different groups. Images of the histograms are saved in data table Expression columns.

Example of Creating a Summary Table with Groups

Suppose a researcher is working with Companies.jmp, which groups companies by Type and Size. Follow along with this next example by opening Companies.jmp from the sample data folder that was installed when you installed JMP.

Suppose that the researcher wants to perform the following tasks:

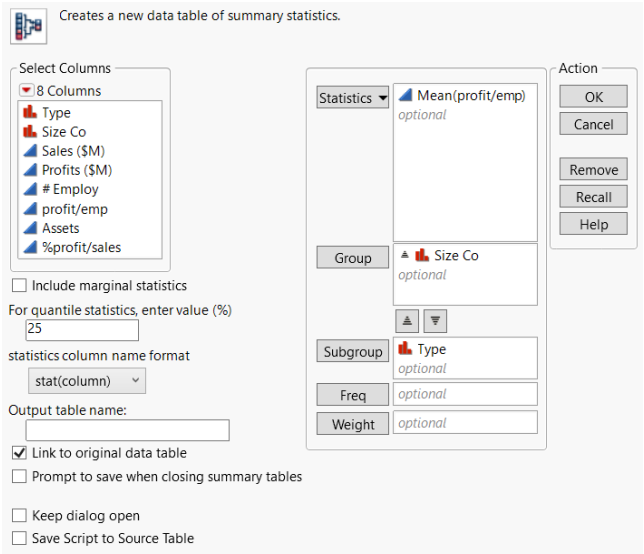
- Create a table that shows the average profit per employee for small, medium, and big computer and pharmaceutical companies. In other words, create a table that contains a row for each size company and a column for the mean profit per employee of each type of company.
- Create it so the cells hold the mean for the subgroup (defined by the intersection of the row and column).

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Tables > Summary**.
3. Select Size Co and click **Group**.

   The researcher selects Size Co as the grouping variable because he wants the values in that column to become rows in the new table.
4. Select profit/emp and click **Statistics**.
5. Select **Mean**.
6. Select Type and click **Subgroup**.

   This tells JMP to create a column for the average profit per employee (Mean(profit/emp)) for each level (computer, pharmaceutical) of subgroup variable (type).

Figure 8.5 Summary Statistics for a Subgroup



	Size Co	N Rows	Mean(profit/emp, Computer)	Mean(profit/emp, Pharmaceutical)
1	big	9	4530.478	17140.699
2	medium	7	-3462.506	24035.115
3	small	16	7998.815	38337.191

Note: The summary table is named after the Group column name. If the column name contains characters that are not allowed in file names, a hyphen is substituted for the special character.

Chapter 9

JMP Reports

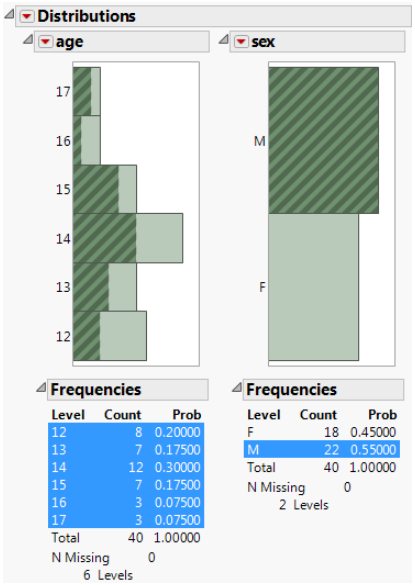
Navigate and Customize Report Windows

Most JMP platforms present the results of your analyses in a report window. The report window consists of graphs and statistical reports that are often linked to the data table. When you select data in the graph or report, the data is highlighted in the data table. Figure 9.1 shows an example of selected data in a report window and in a data table.

This chapter describes how to customize the colors, lines, data points, and other graphical elements in a report window.

Figure 9.1 Example of a JMP Report Window and Data Table

	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107
11	SUSAN	13	F	56	67
12	JOHN	13	M	65	98
13	JOE	13	M	63	105
14	MICHAEL	13	M	58	95
15	DAVID	13	M	59	79
16	JUDY	14	F	61	81
17	ELIZABETH	14	F	62	91
18	LESLIE	14	F	65	142
19	CAROL	14	F	63	84
20	PATTY	14	F	62	85
21	FREDERICK	14	M	63	93
22	ALFRED	14	M	64	99



Contents

Navigate JMP Reports . . . 446
Use the Hand Tool . . . 447
Display Options for JMP Reports . . . 447
Show and Hide Parts of a JMP Report . . . 450
Combine Several Reports . . . 450
Rename Titles in Reports . . . 450
Increase Font Sizes in Reports . . . 450
Resize a Report Window . . . 450
Red Triangle Options in JMP Reports . . . 451
Local Data Filters in Reports . . . 451
Redo Menus in Reports . . . 454
Save Script Menus in Reports . . . 455
The Data Filter . . . 458
Types of Filter Columns . . . 459
Filtering Modes . . . 459
The Data Filter Control Panel . . . 460
Red Triangle Options for the Data Filter . . . 463
Animation in Data Filters . . . 467
Red Triangle Options for Variables in Data Filters . . . 470
Reformat Tables in Reports . . . 473
Modify Display Box Properties in a JMP Report . . . 474
Turn a Table in a JMP Report Into a Matrix . . . 475
Use Conditional Formatting in Reports . . . 475
Select Points in Graphs . . . 481
Select Rows in Graphs . . . 482
Select a Rectangular Area of Points in a JMP Graph . . . 482
Select an Irregularly Shaped Area of Points in a JMP Graph . . . 483
Use Markers in Graphs . . . 484
Change the Shape of Markers . . . 484
Change the Color of Markers . . . 485
Change the Size of Markers . . . 485
Work with Crowded Markers . . . 485
Add Outlines Around Markers . . . 486
Change the Selection Mode of Markers . . . 487
Specify the Transparency of Markers . . . 488
Exclude and Hide Markers . . . 488
Use Images as Markers . . . 489
Change the Shape or Color of Markers Based On Values . . . 490
Change the Appearance of Graphs . . . 492

Resize Graphs . . . 492
Zoom In on Graphs . . . 494
Change Line Widths in Graphs . . . 494
Change the Background Color in Graphs . . . 494
Change the Color of Histogram Bars . . . 495
Display Coordinates and Temporary Reference Lines in Graphs . . . 495
Scroll and Scale Axes in Graphs . . . 495
Customize Axes and Axis Labels in Graphs . . . 496
Change the Order of Values in Reports . . . 507
Change the Pattern and Format of Selected Objects in Graphs . . . 508
Add a Graph to a Data Table . . . 508
Customize Markers, Lines, Text, and More in Graphs . . . 509
Add Images to a JMP Graph or Report . . . 515
Paste a Background Image into a JMP Graph . . . 515
Paste an Image at the End of a JMP Report . . . 516
Drag and Drop an Image into a JMP Graph . . . 516
Add Geographical Images and Boundaries to JMP Graphs . . . 518
Extract Data from an Image . . . 519
Add Annotations and Shapes to JMP Reports and Graphs . . . 520
Add Annotations to JMP Reports or Graphs . . . 520
Add Shapes to JMP Reports . . . 521
Customize Hover Labels in Graphs . . . 524
Add Graphs or Images to Hover Labels . . . 525
Add Rich Text to Hover Labels in Graphs . . . 540
Update Names and Values in Hover Labels . . . 543
Edit Hover Labels in Graphs . . . 553
Add Images from Data Tables to Hover Labels . . . 558
Pin Hover Labels in Graphs . . . 559
Right-Click Options for Hover Labels . . . 560

Navigate JMP Reports

JMP reports are displayed in standard windows with scroll bars and options to resize. They also have other special buttons and menus like those illustrated in Figure 9.2 and those discussed in the following sections.

Figure 9.2 Basics of the Report Window

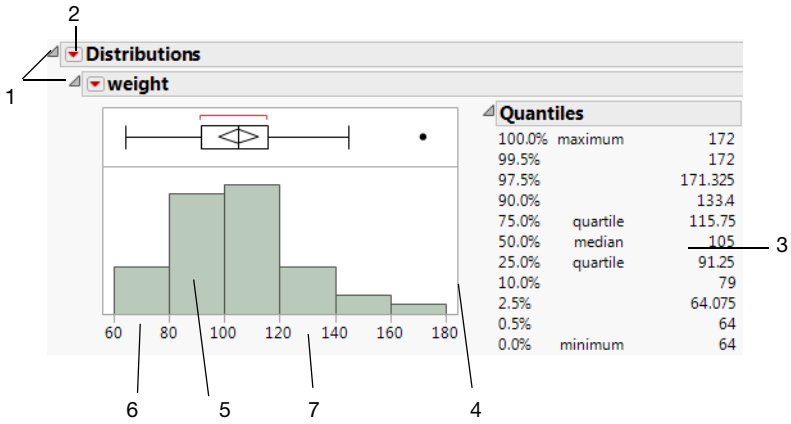


Table 9.1 Report Window Actions

Number	Action
1	Click the disclosure icons to hide or show sections of the report.
2	Click the red triangle to access report options.
3	Right-click in the table to access formatting options.
4	Click and drag on the borders to resize graphs.
5	Right-click anywhere in the graph to access formatting options.
6	Right-click within the axis to access formatting options.
7	The cursor turns into a hand when you position it over an axis. Click and drag the cursor to scroll along the axis or to rescale the axis. See "Scroll and Scale Axes in Graphs".

Tip: Consider setting the *Autosave timeout* value in the General preferences to automatically save open reports at the specified number of minutes. This autosave value also applies to data tables, journals, scripts, projects, and reports.

Use the Hand Tool

Select the hand tool using the Tools > Grabber option. There are many functions that you can use with the hand tool (also known as the grabber tool) in a report. Here are some examples of how the hand behaves in graphs and plots:

- On histograms, for continuous variables, use the hand tool to change the number of bars or to shift the boundaries of the bars.
- In all report tables, use the hand tool to click and drag columns for rearranging.
- Use the hand tool to change the displayed range of axis values. See "Scroll and Scale Axes in Graphs".

Display Options for JMP Reports

Right-click a disclosure icon ◿ to show a menu that lets you rearrange the report and gives you control over report outline levels. The resulting menu has the following report formatting options:

Close Closes (hides) that section of the report. This can also be accomplished by clicking the disclosure icon.

Horizontal If available, the option switches the outline of the report between a vertical and horizontal layout.

Open All Below Opens all outline levels beneath the level where this command is selected, including that level.

Close All Below Closes all outline levels beneath the level where this command is selected, including that level.

Open All Like This Opens all of the same type of reports as the one that is present in the analysis window. If you analyze several variables at a time, you often want to open many of the same type of report tables all at once. You might also want to open all of the same type of report tables at once when you select multiple options on a single analysis.

Close All Like This Closes all of the same type of reports as the one that is present in the analysis window.

Close Where No Outlines Closes all parts of the report that do not have sublevels. This command is usually used at the top level of the report outline. It is a quick way to see a nesting structure overview of a report.

Outline Close Orientation Specifies the orientation of the outline box when the disclosure icon is clicked closed. Available options are **Auto**, **Horizontal**, or **Vertical**. Default orientation is **Auto**.

Append Item Displays a submenu, which lists ways that you can add structural items to the report. Items include text, outline title bars, references to other JMP files and windows, a list of all open JMP files, URLs, and scripts.

Add Text Item Opens a window that enables you to add up to six text items. You can select whether the text items appear with a bullet or initially appear hidden. **Note:** You must click exactly on a hidden text box for the text item to become visible.

Add Outline Item Open a window that enables you to add a titled outline box. **Note:** The appended outline box contains a red triangle menu listing the append menu items.

Add Window Reference Opens a window that enables you to select an open JMP window for adding as a link to the selected outline box.

Add File Reference Displays the Open Data File window that enables you to select a file for adding as a link to the selected outline box.

Add Directory of Files Displays the **Choose directory** window that enables you to select a directory for adding as a link to the selected outline box. **Note:** Links to each file in the selected directory are appended to the outline box.

Add All Open Files For each file open in JMP, a new outline box is added. It contains a link to the file.

Add URL Reference Opens the Create Link to URL window that enables you to add a **Link Name** and **URL**. The URL reference link is added to the selected outline box.

Add Script Button Opens the Add Script Button window that enables you to add a named link to the selected outline box. Clicking the added link runs the specified script.

Size to Fit Resizes the outline to fit the width of the inside graph or report.

Show Properties Enables you to modify the formatting of display boxes in the report. Properties include graph axis elements such as tick lines,

---

Tip: You can modify some properties directly in the report by selecting one or more display boxes with the Selection tool and hovering over the display box until a popup property editor appears.

Edit Displays a submenu that contains the following options, which affect all reports at the outline level where they are used:

Select Highlights all reports for that outline level.

Deselect Deselects all selected reports for that outline level.

Journal Duplicates the report in a separate window titled Journal so that you can edit it or append other reports to it. See "Create Journals".

Copy Picture Copies the report to the clipboard. You can then open another application and paste it.

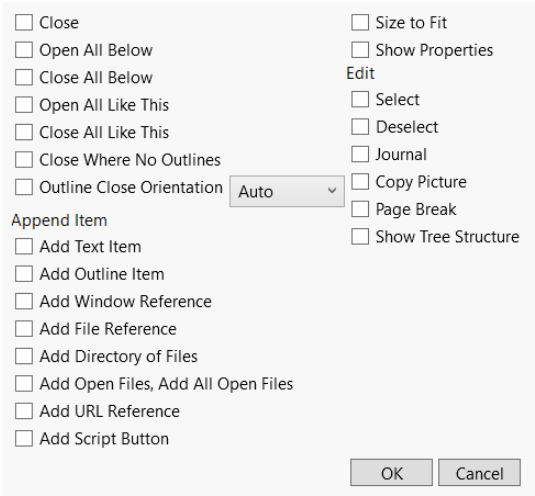
Page Break Inserts a page break for printing purposes.

Show Tree Structure Opens a window that shows the display boxes that make up the report. This is mainly used by JSL programmers who are manipulating or reading parts of the report. See the *Scripting Guide*.

Restore Window Size (Appears only if the window size has been modified from its original size.) Returns the selected window to its original size.

An alternative way to access these options is to press Alt and right-click the disclosure icon ◿. This displays a window, as shown in Figure 9.3, with check boxes for commands and options so that you can select multiple actions at the same time. You can also do the same for the menu under a red triangle menu.

Figure 9.3 Outline Box Menu Items Window



Show and Hide Parts of a JMP Report

JMP reports are organized in a hierarchical outline. Each level of the outline has a triangle-shaped disclosure icon . Click the disclosure icon to open and close that section of the report.

You can also change which columns appear in a report. Right-click the report and select the column from the Columns list. To select more than one column at once, press Alt before right-clicking.

Combine Several Reports

Suppose that you perform multiple analyses and want to show all of the results (and the data table) in one window. You can select and combine the reports and the data table in several ways. See "Example of Combining Windows to Create a Dashboard".

Rename Titles in Reports

To rename a title in a report, double-click any of the following titles:

- a title next to a red triangle menu
- a title next to a disclosure icon
- a column title

Increase Font Sizes in Reports

On Windows, change the font size that JMP uses in reports and data tables by selecting **Window > Font Sizes**. Then choose from one of the submenu items:

Increase Font Size Increases the font size. Select again to increase the font size again.

Decrease Font Size Decreases the font size. Select again to decrease the font size again.

On macOS, select **View > Make Text Bigger** or **View > Make Text Smaller**.

Resize a Report Window

Report windows are automatically sized to fit the contents of the window. On Windows, you can automatically resize a window after manually resizing it. Press Ctrl and click the lower right corner of the window.

Red Triangle Options in JMP Reports

Click the red triangle menu in a report to access a list of options that apply for that particular report. In addition to clicking the red triangle menu, you can also:

- Select multiple actions at the same time. Press Alt and click the red triangle menu. A panel of all commands and options appears with check boxes.
- Apply a command to all similar reports in the report window. Press Ctrl and click the red triangle menu. For example, in a One-way analysis, if you press Ctrl, click the icon, and select **Means/Anova/Pooled t Test**, an analysis of variance is performed for all One-way analyses in the active report window.

The red triangle options applicable to each platform in the Analyze and Graph menus are described in the following JMP documentation:

- *Basic Analysis*
- *Consumer Research*
- *Essential Graphing*
- *Fitting Linear Models*
- *Multivariate Methods*
- *Profilers*
- *Quality and Process Methods*
- *Reliability and Survival Methods*
- *Predictive and Specialized Modeling*

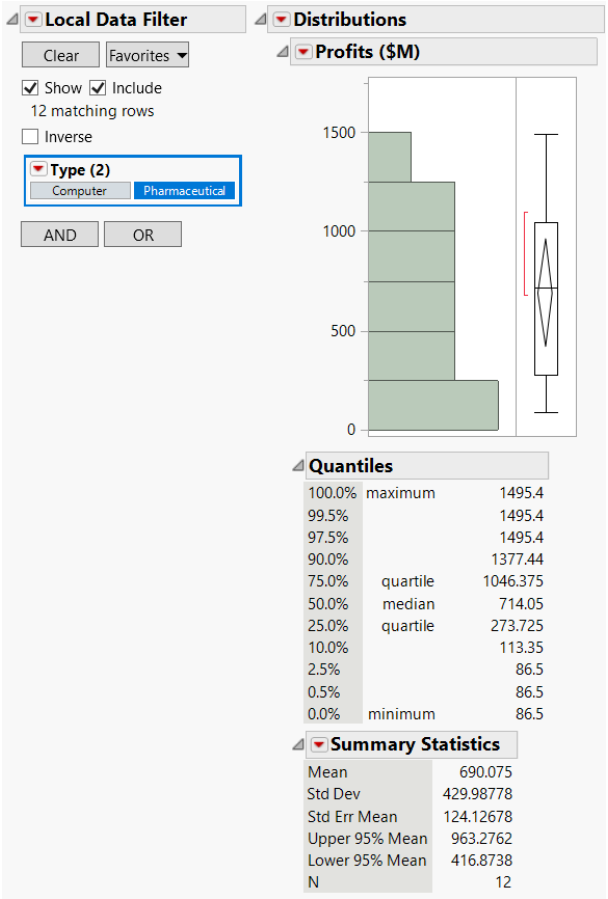
Local Data Filters in Reports

Local Data Filters limit the scope of the filter to the output of the platform that you are in. For example, suppose that you have data on the profits of 30 different computer and pharmaceutical companies. You want to examine profits by company type. You can run a distribution and then use the Local Data Filter to narrow the results in the Distribution platform to the type of company.

1. Select **Help** > **Sample Data Library** and open Companies.jmp.
2. Select **Analyze** > **Distribution**.
3. Select Profits ($M) and click **Y, Columns**.
4. Click **OK**.
5. Click the Distributions red triangle and select **Local Data Filter**.
6. Click Type and click **Add** .

7. In the Type box, click **Pharmaceutical** to see only their profits and then click **Computer** to see only their profits.

Figure 9.4 Local Data Filter Showing Pharmaceutical Profits



Note: The red triangle menu next to a variable contains options such as inverting the selection and displaying the values in a list of check boxes. See "Red Triangle Options for Variables in Data Filters" for more information.

Local Filtering

When you use a Local Data Filter, it only filters in the platform, not in the data table (as the Data Filter does). For example, if you exclude rows in a Local Data Filter, these rows are not excluded in the data table. And, if you have row states in your data table, they are not impacted by the Local Data Filter.

Saving a Filter

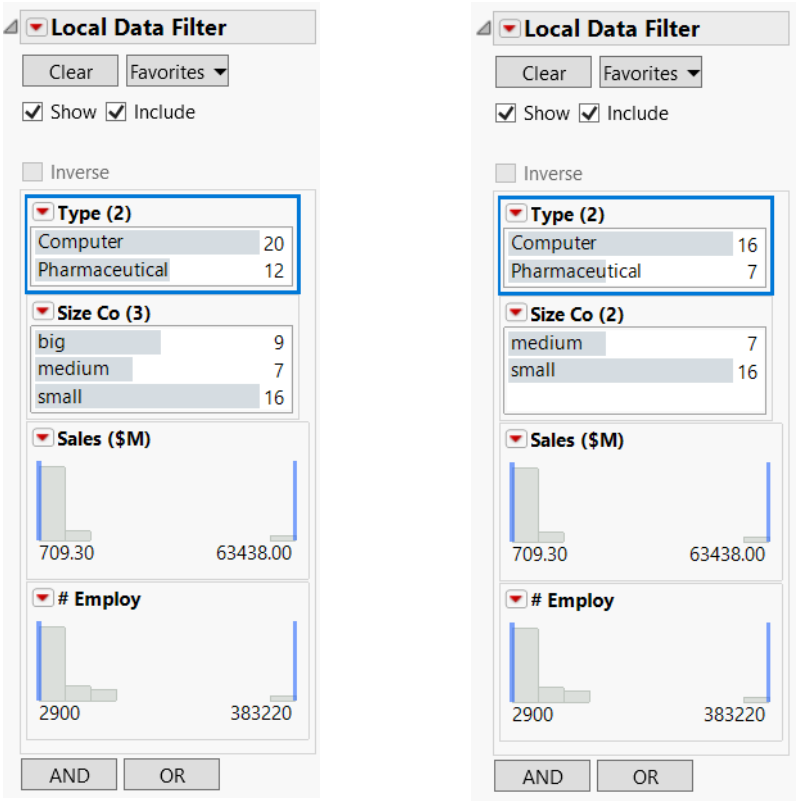
You can script and therefore save a Local Data Filter, so that you can access your filter settings again quickly. To save a filter, click the red triangle next to Local Data Filter and select one of the Save Script options.

Counting Excluded Rows

The counts for each level of a variable (including excluded rows) are shown in the Local Data Filter by default. To omit the excluded rows from the count, click the Local Data Filter red triangle and deselect **Count Excluded Rows**.

Figure 9.5 shows the difference between counting and omitting excluded rows. When the excluded rows are counted, the Local Data Filter shows 20 rows that contain the value "Computer". Also, there are nine "big" values. When the excluded rows are omitted, only 16 rows contain "Computer".

Figure 9.5 Counting (Left) and Omitting (Right) Excluded Rows



Note: The counts are not shown when Display Options is set to Blocks Display (Figure 9.4).

For more information about advanced scripting with Local Data Filters, see the *Scripting Guide*.

Using By-Groups and Subset Reports

If your analysis uses by-groups or combines together reports using linked subsets of the same table, a local data filter on the parent table can be shared between all the different by-groups and subset reports.

Using Local Data Filters in Dashboards and Applications

You can incorporate Local Data Filters into dashboards or applications:

- For an example using dashboards, see "Example of Creating a JMP Query Dashboard and Add-In".
- For an example using applications, see the *Scripting Guide*.

Zoom In on Time Series Data

After you filter time series data, you can zoom in closer on the Local Data Filter histogram to see smaller time increments. Suppose that one year of data are selected on the histogram. To view smaller increments of time within that year, select **Zoom to Selection** from the red triangle menu next to the variable. Select **Reset Zoom** from the Local Data Filter red triangle menu to reset the histogram.

Redo Menus in Reports

After performing an analysis, you might want to run the analysis again using different options or variables. You might also want to perform the analysis again if data in the data table have changed. The Redo red triangle menus contain the following options:

Column Switcher Lets you interactively exchange one column for another on a graph. See "Column Switcher".

Redo Analysis Duplicates the analysis based on the previous launch window settings. The analysis appears in a new report window. This option is helpful if the data have changed.

Relaunch Analysis Opens the launch window and recalls the settings used to create the report.

Automatic Recalc Automatically updates analyses and graphics when data table values change. See "Automatic Recalc".

Save Script Menus in Reports

After completing an analysis and receiving a report, you might want to save the process that you used to create the report as a JSL script. You do this by selecting an option from the Script red triangle menu. Most of the Save Script options are the same throughout JMP. A few platforms add extra options that are described in the specific platform chapters. The following Save Script menu options are common to all platforms.

To Clipboard Saves the script to your computer's clipboard. You can then copy the script to a script or another document.

To Data Table Saves the script to the data table that was used to produce the report. This enables you to run the script again from the data table to re-create the results. Select this option to save a script to the data table.

To Journal Saves a link in a journal window that runs the script for the analysis. If a journal window is already open, the script is added to that journal window. The script contains the path to the data table. Note that if the data table cannot be found, the script does not run.

To Script Window Opens a script editor window and adds the script to it. If a script window is already open, the script is added to the bottom of that open window.

To Report Adds the script to the top of the report window.

Script All By-Groups

If you specified a By variable in the launch window, the Save By-Group Script menu enables you to save a script for all levels of the By variable to the data table or script window.

In the Save Script menu, **To Data Table (All Objects)** and **To Script Window (All Objects)** save a script for all levels of the By variable.

Automatic Recalc

The Automatic Recalc feature immediately reflects changes that you make to the data table in the corresponding report window. You can make any of the following data table changes:

- exclude or unexclude data table rows
- delete or add data table rows

This powerful feature immediately reflects these changes to the corresponding analyses, statistics, and graphs that are located in a report window.

To turn on Automatic Recalc for a report window, click the platform red triangle and select **Redo > Automatic Recalc**. To turn it off, deselect the same option. You can also turn on Automatic Recalc using JSL.

Platforms that support Automatic Recalc include a preference in the Platforms preferences.

Note: For some platforms, the Automatic Recalc feature is not appropriate and therefore is not supported. These platforms include the following: DOE, Profilers, Choice, Partition, Nonlinear, Neural, Neural Net, Partial Least Squares, Fit Model (REML, GLM, Log Variance), Gaussian Process, Item Analysis, Cox Proportional Hazard, Response Screening, and Control Charts (except Run Chart).

Column Switcher

Within a report, use the Column Switcher to quickly analyze different variables without having to re-create your analysis. To activate the Column Switcher, from a report window, click the red triangle and select **Redo > Column Switcher**.

If you have multiple columns, use the buttons to animate the column switching or step through each column manually. Move the slider control to change the speed of the animation.

Note: You cannot copy or move the column switcher within a report. Also, a column switcher cannot be saved to a JMP journal.

Example of the Column Switcher

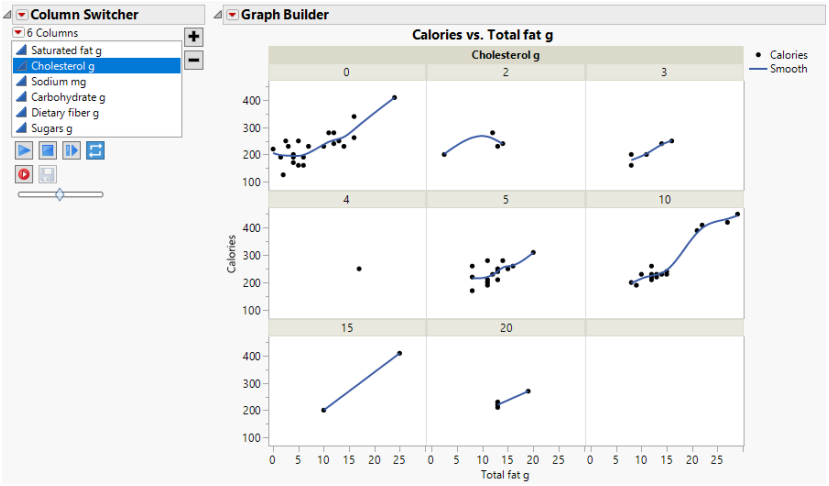
You have data about nutrition information for candy bars. You want to examine different factors, to see which factors best predict calorie levels.

1. Select **Help > Sample Data Library** and open Candy Bars.jmp.
2. Select **Graph > Graph Builder**.
3. Click Total fat g and drag to the **X** zone.
4. Click Calories and drag to the **Y** zone.
5. Click Cholesterol g and drag to the **Wrap** zone.
6. Click the Graph Builder red triangle and select **Show Control Panel**.
7. Click the Graph Builder red triangle and select **Redo > Column Switcher**.
8. Select Cholesterol g.

   This chooses the initial column to switch.
9. Select Saturated fat g, Cholesterol g, Sodium mg, Carbohydrate g, Dietary fiber g, and Sugars g.

   This chooses the replacement columns.
10. Click **OK**.

Figure 9.6 Column Switcher in Graph Builder Window



11. Click the **Play** button to cycle between the different factors. Use the slider to control the speed of the animation. Alternatively, you can step through each factor individually.

    Click the Looping button to specify the behavior of the animation when it reaches the last column in the Columns list.

You can see that the relationship between calories and fat is relatively strong for each level of carbohydrate. Therefore, Carbohydrate g appears to be the best predictor of calorie levels.

Notes:

- To keep axis settings constant when switching columns, select **Retain Axis Settings** from the Column Switcher red triangle menu. This is useful when you have customized an axis, for example, by adding a reference line.
- If a report has multiple column switchers, you can adjust the layout of the column switchers using the Layout options in the Column Switcher red triangle menu.
- To add new columns, do one of these options:
  - In the Column Switcher, click the plus button, choose columns, and click **OK**.
  - In Graph Builder, drag columns from the Columns list into the Column Switcher list.
  - From a data table, drag columns from the Columns panel into the Column Switcher list.
- To remove columns, click the minus button, choose the columns to remove, and click **OK**. Or, you can select the column and press the Delete key.
- To revert changes, select **Reset** from the Columns red triangle menu.
- The Record and Save buttons in the Column Switcher record the animation and then save it as an animated GIF. This feature is available only on Windows.

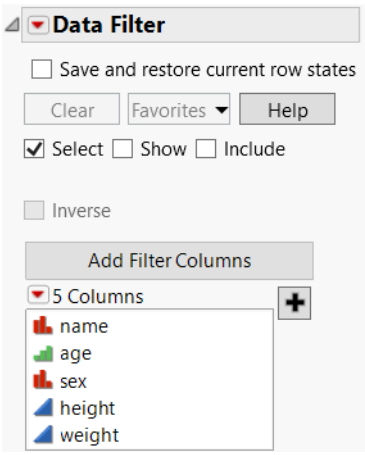
The Data Filter

The Data Filter in JMP provides a variety of ways to identify subsets of data. You can interactively select complex subsets of data, hide these subsets in plots, or exclude them from analyses.

1. Select **Rows** > **Data Filter**.

Tip: In addition to the main Data Filter, you can also launch a local Data Filter within a platform report. Click the Local Data Filter icon from the Report toolbar, or select **Local Data Filter** from the red triangle menu in a report. See "Local Data Filters in Reports" for more information.

Figure 9.7 Initial Data Filter Window



2. Select the columns that you want to use as filters, and then click **Add** .

Notes:

- To restore your current row states when the Data Filter window is closed, select the **Save and restore current row states option**.
- If you have a long list of columns, you can sort, show, hide, or search for columns in the list. Use the options in the Add Filter Columns red triangle menu.
- By default, the Data Filter window is attached to the data table. You can detach it temporarily or permanently:
  - Detach it temporarily by deselecting the **Use Floating Window** option from the Data Filter red triangle menu.
  - Detach it permanently by selecting **File > Preferences > Data Filter** and deselecting **Use Floating Window**.

Types of Filter Columns

There are four types of filter columns:

Continuous columns Filters numeric columns whose modeling type is set to continuous. A continuous filter column is represented by a histogram that spans the data range. You can adjust vertical lines on the histogram that represent the minimum and maximum filtered values. For options, see "Options for All Types of Variables" and "Options for Continuous Variables".

Categorical columns Filters nominal and ordinal columns. For each categorical column, the Data Filter generates a set of distinct categories. These categories can be displayed in different forms. For options, see "Options for All Types of Variables" and "Options for Nominal or Ordinal Variables".

Note: For categorical columns with value labels, if you want to include responses that are not present in the data, select the **Include Responses Not in Data** option from **File > Preferences > Platforms > Categorical**.

Multiple Response columns Filters character columns that have the Multiple Response column property or Multiple Response modeling type assigned. Each data cell of the column generally consists of multiple categories, separated by some common separator, like a comma. Since each data cell can contain more than one category, multiple response columns have a richer set of filtering options. For options, see "Options for All Types of Variables" and "Options for Variables with the Multiple Response Modeling Type or Column Property".

Unstructured Text Filters text columns whose modeling type is set to Unstructured Text. Type words or phrases into the filter text edit box to add them to the filter text list. You can then filter the rows based on these words and phrases. For options, see "Options for All Types of Variables" and "Options for Unstructured Text".

Filtering Modes

There are three modes of filtering: **Select**, **Show**, and **Include**. You can set and clear these modes using the corresponding check boxes in the Data Filter.

Select (Not available for the Local Data Filter.) Shows the selected rows in the data table in a highlighted state.

You can turn off the automatic selection of this option using the **Data Filter Select Check** preference. See also "Changing the Row State in the Data Table After Making Data Filter Selections".

Show Shows the unselected rows with the Hide icon .

You can turn on the automatic selection of this option using the **Data Filter Show Check** preference. For more information about row states, see the "Store Information in Row State Columns".

Include Shows the unselected rows with the Exclude icon .

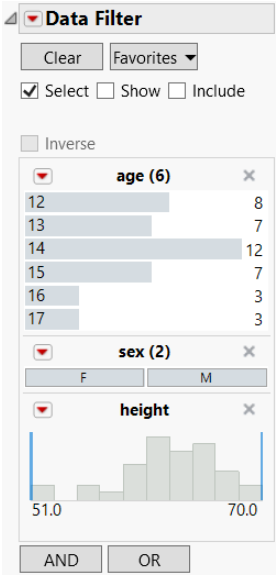
You can turn on the automatic selection of this option using the **Data Filter Include Check** preference. For more information about row states, see the "Store Information in Row State Columns".

There are two additional options when filtering: **Auto clear** and **Conditional**. These options are available from the red triangle menu for the Data Filter. See "Red Triangle Options for the Data Filter".

The Data Filter Control Panel

Once you have added columns in the initial window, the Data Filter control panel appears.

Figure 9.8 Data Filter Control Panel



The main controls in the Data Filter include the following:

Clear Clears all selections that you have made on variables in the Data Filter window.

Favorites Saves your current data filter criteria as a favorite. Once you have created a favorite, selecting it resets the current conditions to the criteria in the favorite. You can also

remove the favorite. To retain favorites once the current session ends, save the data filter script by selecting one of the Save Script options from the Data Filter red triangle menu.

Select, Show, and Include See "Filtering Modes".

Inverse Inverts the current selection state of the rows in the data table.

Note: Only the rows in the data table are inverted, not the selection in the Data Filter. To invert the selection in the Data Filter, from the column's red triangle menu, select **Invert Selection**.

AND The AND button opens the **Add to Selected Group** list. The *and* operator restricts the selection. You can add variables to the filter process at any time.

OR The OR button opens the **Add new OR Group list**. The *or* operator extends the selection. You can add variables to the filter process at any time.

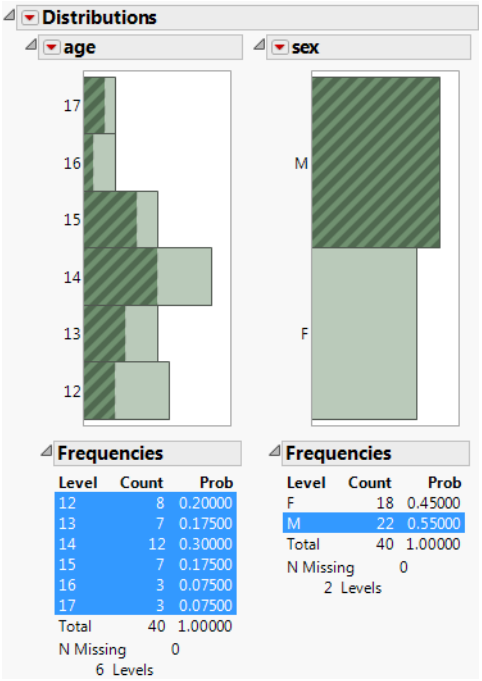
Changing the Row State in the Data Table After Making Data Filter Selections

If you have made selections in the Data Filter and subsequently alter row states in the data table, or select points in a graph or plot, the selections in the Data Filter might not match the selections in the data table. The Data Filter contains a warning message that says: "Your selection was changed in another window". The **Reset Selection** button appears. Clicking the **Reset Selection** button changes the data table selections back to reflect the selections in the Data Filter.

Example of Modifying Selections

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Distribution**.
3. Select age and sex and click **Y, Columns**.
4. Click **OK**.
5. Select **Rows > Data Filter**.
6. Select sex and click **Add**.
7. In the Data Filter control panel, select all of the males by clicking on the M block.

Figure 9.9 Rows Containing Males Highlighted in Data Table and Histograms



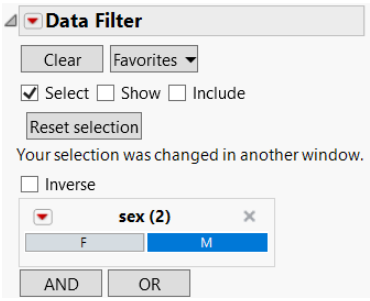
	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107
11	SUSAN	13	F	56	67
12	JOHN	13	M	65	98
13	JOE	13	M	63	105
14	MICHAEL	13	M	58	95
15	DAVID	13	M	59	79
16	JUDY	14	F	61	81
17	ELIZABETH	14	F	62	91
18	LESLIE	14	F	65	142
19	CAROL	14	F	63	84
20	PATTY	14	F	62	85
21	FREDERICK	14	M	63	93
22	ALFRED	14	M	64	99

You can see that all of the rows containing males are highlighted in the data table and in the histograms. Now you decide that you want to see only the students who are age 12.

8. In the age histogram, select the bar representing age 12.

   Now the selection does not match the Data Filter selection. A warning message and a **Reset selection** button appear in the Data Filter window.

Figure 9.10 Data Filter Warning Message and Reset Button



Red Triangle Options for the Data Filter

The red triangle menu next to Data Filter contains the following options:

Auto clear If you have more than one nominal or ordinal column selected in the Data Filter, this option clears any other selections before making a new selection. For example, using Big Class.jmp, suppose that you have the columns sex (nominal) and age (ordinal) in your Data Filter. If you have males (M) selected for sex, and you click an age group, say age 12, your selection of males will be automatically cleared. This means that selecting age 12 is not conditional on selecting males. Conversely, if you turn off Auto clear, you can then select both males and age 12 at the same time. Auto clear is off by default.

Conditional Limits the categories displayed for the selected filter column. See "Conditional Data Filters".

Grouped by AND Enables you to control the AND and OR behavior of multiple groups of column filters. See "Grouped by AND in Data Filters".

Show Histograms and Bars Shows or hides the histogram and bars in the data filter.

Show Counts Shows or hides the counts in the categorical column data filters.

Use Floating Window Keeps the Data Filter window on top of its associated data table. If you do not want the Data Filter window to remain on top, deselect this option.

Start Over Closes the current Data Filter session and shows the original Data Filter window.

Show Controls Shows or hides all buttons and checkboxes in the Data Filter window except Clear and Inverse.

Show Modes Shows or hides the filtering mode checkboxes in the Data Filter window.

Animation Sequentially highlights the values of a single variable in the data table. See "Animation in Data Filters".

Subset Creates a new data table that contains only the following:

- The rows identified by the Data Filter.
- The columns selected in the active data table. If no columns are selected, then all columns are included.

This option is similar to the Tables > Subset command, only without subsetting options.

Save Where Clause Builds a Where clause based on the value selections that you make.

Tip: Selecting **To Formula Column** creates an indicator column. You can use the indicator column in many platforms as a By role, Group or ID in Graph Builder, and so on.

Save Script Provides options for saving scripts. See "Save Script Menus in Reports".

Copy Local Data Filter (Available only for the Local Data Filter.) Copies the script for the local data filter to the clipboard. You can then apply this local data filter to a new report by using the Paste Local Data Filter option in the report's red triangle menu.

Remove Local Data Filter (Available only for the Local Data Filter.) Removes the Local Data Filter from the report.

Conditional Data Filters

For filter columns with hierarchy, you can use the Conditional option to filter what appears in the column lists. For example, you could filter by region so that only the states in the selected region appear in the list.

Note: If there is more than one OR group, the conditional filter is not available.

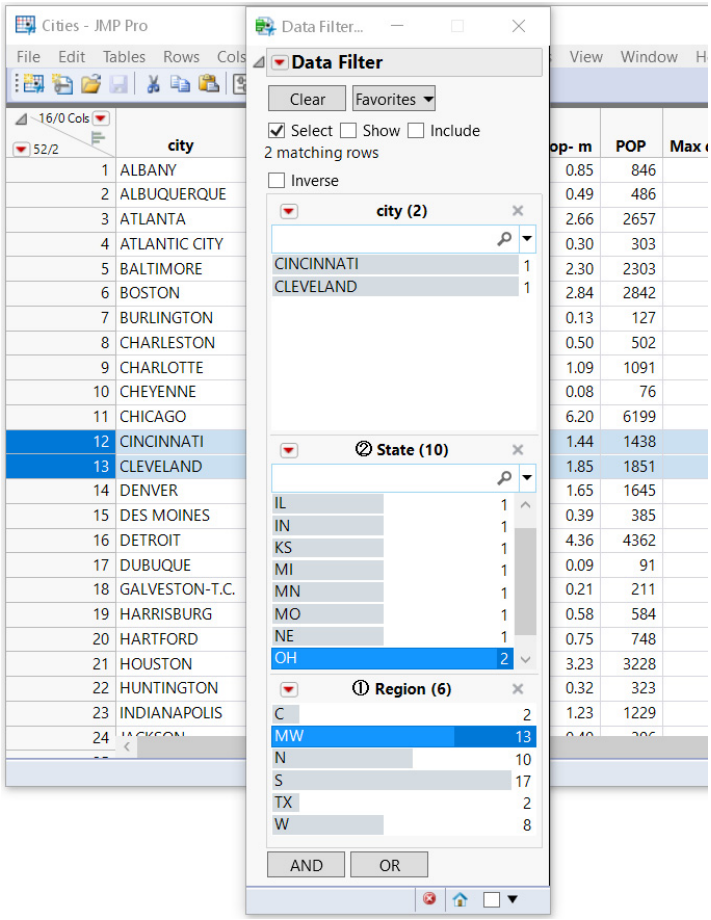
The following example illustrates how the Conditional option helps show the subcategories clearly, without the extra categories that do not belong.

1. Select **Help > Sample Data Library** and open Cities.jmp.
2. Select **Rows > Data Filter**.
3. In the Data Filter window, select city, State, and Region, and then click **Add**.

   The Data Filter window appears, showing a list for each variable.
4. Click the Data Filter red triangle and select **Conditional**.
5. Select MW in the Region list.
6. Select OH from the State list.

   The cities that are in Ohio and in the Midwest region are selected in the data table. In the Data Filter window, only Midwestern states appear in the State list, and only cities in Ohio appear in the cities list.

Figure 9.11 Using the Conditional Option



The circled number in front of the column name indicates the order in which the column values were selected. In Figure 9.11, Region was selected first, so it has a circled “1” in front of the column name. State was selected second, so it has a circled “2” in front of the column name.

When you rearrange hierarchical filters that are in ascending order, the filter number changes to match the ascending position in the hierarchy.

To clear the selections and reset the order of the hierarchy, click **Clear**.

Grouped by AND in Data Filters

When you add a filter column, click AND, and add another filter column, you create a filter group. Within that group, columns that meet all criteria are selected in the data table.

When you then click OR and add one or more filters, you create a second filter group. With Grouped by AND deselected, this is the default behavior.

If you select Grouped by AND, you reverse the behavior.

The following example shows what happens when Grouped by AND is selected.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Rows > Data Filter**.
3. Click the Data Filter red triangle and select **Grouped By AND**.
4. In the Data Filter window, select the age column and click **Add**.
5. Click **OR** and add the sex column.

   sex is added to the same group as age.
6. Select **14** from the age filter, and select **M** from the sex filter.

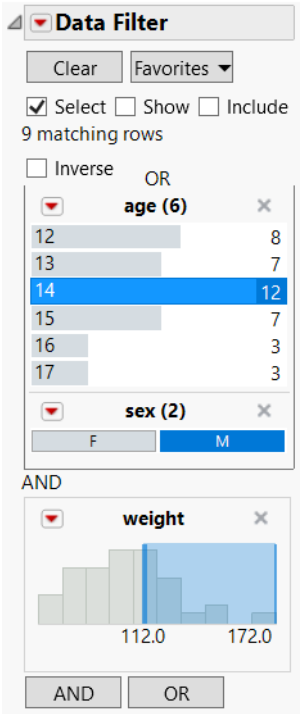
   age and sex are in the same group. In the data table, rows that meet either the age OR sex conditions are selected. That is, all 14-year-olds and all males are selected in the data table.
7. Click **AND** and add the weight column.

   A new group is added.
8. Move the *left* weight slider to approximately 112.

   Another filter group has been added. The second filter group limits the selection to those rows where weight is greater than 112.

   The two filter groups together select rows that satisfy the condition in the first group AND the condition in the second group, that is, a student who is 14 or any male but must weigh more than 112.

Figure 9.12 Data Filter with Grouped by AND Condition

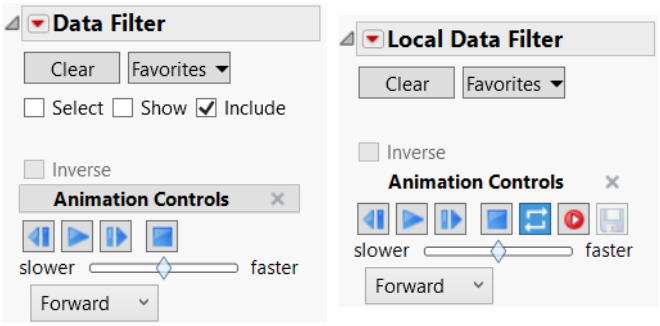


Animation in Data Filters

The animation feature sequentially highlights the values of a single variable. The variable's values are highlighted in the data table. However, patterns are more interesting if you first create a plot and then animate a variable using the Data Filter to see how it behaves on the plot.

To use the animation feature, click the Data Filter red triangle and select **Animation**. Then select the variable that you want to animate. The highlighted frame around the variable indicates which variable is selected for animation.

Figure 9.13 Animation Control Panel in the Data Filter and Local Data Filter



The Animation Control panel contains the following controls:

- The middle button ( ) starts and pauses the animation. After you start the animation cycles, the button changes to a pause button ( ). By default the animation begins with the first value of the topmost variable.
- The backward arrow ( ) moves the animation backward one cycle. Click more than once to go backward more than one cycle.
- The forward arrow ( ) moves the animation forward one cycle. Click more than once to go forward more than one cycle.
- The square button ( ) hides the Animation Control section on the Data Filter window. Select **Animation** from the menu on the Data Filter title bar again to see the Animation Control.
- The looping button ( ) enables looping and is on by default. If you turn the looping button off, the animation will cycle through the sequence of values one time and then stop.
- The record button ( ) starts recording the animation. After you start the recording, the button changes to a stop recording button ( ).
- The save button ( ) saves the recorded animation.

Use the slider to adjust the speed of the animation (slower to faster).

The Animate drop-down menu contains the following options:

Forward Highlights values from first to last.

Backward Highlights values from last to first.

Bounce Highlights forward and then backward repeatedly.

Save Where Clause in Data Filters

Once you have filtered variable values in the Data Filter, that information can be expressed as a JMP Where clause. The Where clause is used in JSL (JMP Scripting Language) programs to identify specific rows of data for processing or analysis. The Data Filter builds a Where clause based on the value selections that you make.

The options in the Save Where Clause menu include the following:

To Clipboard Creates a Where clause from the filter criteria and puts it on the clipboard.

To Row State Column Creates a row state column in the data table that has a formula equivalent to the filter criteria.

To Formula Column Creates a formula column in the data table that has a formula equivalent to the filter criteria.

To Data Table Creates a Where clause from the filter criteria and saves it as a JSL command with the current data table in a table property called Filter.

To Script Window Creates a Where clause from the filter criteria and appends it to the current script text window, or creates a new script if one does not exist already.

To Journal Creates the Where clause from the filter criteria and appends it to the current journal, or creates a new journal if one does not already exist.

Example of Saving a Where Clause

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Rows > Data Filter**.
3. Select age, sex, and height and click **Add**.

Select all females who are twelve and fourteen years old and whose height is between 56 and 60 inches:

4. Click the 12 and 14 blocks and the F block.
5. Click 51 and type 56.
6. Click 70 and type 60.
7. Click the Data Filter red triangle and select **Save Where Clause > To Script Window**.

The Where clause that is created from this example appears in a script window:

```
Select Where(
(:age == 12 | :age == 14) & :sex == "F" & (:height >= 56 & :height <= 60)
);
```

Red Triangle Options for Variables in Data Filters

After you add a column to the data filter, a red triangle menu appears. Some of the red triangle options for a variable can vary, depending on the type of variable.

Options for All Types of Variables

The red triangle menu next to any type of variable contains the following generic options:

Clear Selection Clears any selection in effect for that variable only.

Invert Selection Deselects any selected values, and selects all values previously not selected, for that variable only.

Modeling Type Shows a submenu of modeling types for the column filter. By default, the column filter type is set to the column's modeling type. You can change the filter type by selecting a different modeling type in the submenu.

Options for Continuous Variables

For continuous variables, values appear in a histogram with vertical lines at the minimum and maximum values. You can adjust the selected range by clicking on the vertical lines and dragging. The updated range is highlighted in blue.

The following options are available for continuous variables only:

Select Missing Selects rows that contain missing values.

Include Minimum Value Includes or excludes the minimum value in the selected range. When the minimum value is excluded, the solid vertical line in the histogram becomes a dashed vertical line.

Include Maximum Value Includes or excludes the maximum value in the selected range. When the maximum value is excluded, the solid vertical line in the histogram becomes a dashed vertical line.

Set Number of Bins Sets the number of bins for the histogram.

Zoom to Selection Shows only the selected range in the histogram. The menu item changes to Reset Zoom after you select it. Reset Zoom resets the display.

Options for Nominal or Ordinal Variables

For nominal and ordinal variables, values appear in blocks, in a list, or in a menu. If the variable contains only a small number of categories, the values appear in blocks. If the variable contains a large number of categories, the values appear in a list or in a menu. However, you can change these default settings.

The following options are available for nominal or ordinal variables only:

Display Options Changes the appearance of the display. Options can include the following:

Blocks Display Shows each level as a block.

List Display Shows each level as a member of a list, followed by its frequency.

Single Category Display Shows each level, followed by its frequency, in a menu.

Check Box Display Adds a check box next to each value. To make this the default setting, select the **Check Box Display** option in **File > Preferences > Data Filter**.

Radio Box Display Adds a radio button next to each value.

Order by Count Orders the values in decreasing sort order by count.

Find Provides a text box where you can enter a search string for the selected column. Press Enter to perform the search.

Options for Variables with the Multiple Response Modeling Type or Column Property

The following options are available for variables with the Multiple Response modeling type or column property:

Display Options Changes the appearance of the display. Options can include the following:

List Display Shows each level as a member of a list, followed by its frequency.

Single Category Display Shows each level, followed by its frequency, in a menu.

Check Box Display Adds a check box next to each value. To make this the default setting, select the **Check Box Display** option in File > Preferences > Data Filter.

Radio Box Display Adds a radio button next to each value.

Match None Selects only rows containing values that do not match any of the selections

Match Any Selects all rows that contain values that match any of the selected values. By default, this option is selected.

Match All Selects only rows with values that include all of the selected values.

Match Exactly Selects only those rows with values that exactly match the checked values.

Match Only Selects only those rows with contents exactly matching the checked values.

Match At Least Selects at least *n* of the selected values.

Match At Most Selects a maximum of *n* of the selected values.

Match Between Selects between *n* and *m* of the selected values.

Find Provides a text box where you can enter a search string for the selected column. Press Enter to perform the search.

Note: For more information about the Multiple Response property, see the "Multiple Response".

Options for Unstructured Text

The following options are available for variables with the Unstructured Text modeling type:

Case Sensitive If this box is checked, the filter based on the word or phrase typed in the filter text edit box is case sensitive.

Regular Expression Assumes the string typed into the filter text edit box is a regular expression instead of a literal string.

Display Options Changes the appearance of the display. Options can include the following:

List Display Shows each word or phrase as a member of a list, followed by its frequency.

Single Category Display Shows each word or phrase, followed by its frequency, in a menu.

Check Box Display Adds a check box next to each word or phrase. To make this the default setting, select the **Check Box Display** option in File > Preferences > Data Filter.

Radio Box Display Adds a radio button next to each word or phrase.

Add Missing Adds a filter for entries that do not contain any text.

Clear Filter Texts List Removes any words or phrases that had been added to the filter text list.

Show Filter Text Edit Box Shows or hides the filter text edit box.

Order by Count Orders the words or phrases in decreasing sort order by count.

Match None Selects only rows containing text that does not match any of the selected words or phrases.

Match Any Selects all rows that contain text that matches any of the selected words or phrases. By default, this option is selected.

Match All Selects only rows with text that includes all of the selected words or phrases.

Reformat Tables in Reports

There are many ways to change the formatting of a report table. Right-click a report table to access the following formatting options:

Table Style Adds borders, shading, and dividing lines to the table. Select from the following options:

Underline Headings Shows the preference setting for Underline Table Headings.

Shade Headings Shows the preference setting for Shade Table Headings.

Heading Column Borders Shows divider lines between columns in headings.

Column Borders Contains borders outside columns and divider lines between columns.

Row Borders Contains borders outside rows and divider lines between columns.

Shade Alternate Rows Shades alternate rows.

Shade Cells Shades the body of the table. When used with Shade Alternate Rows, a darker shade is used on alternate rows.

Note: Change the format of all report tables by selecting the preceding options in the JMP preferences. On Windows, the Report Tables options are in File > Preferences > Styles. On macOS, select the Report Tables options in **JMP > Preferences > Styles.**

Columns Shows or hides columns in the table.

Note: Columns whose names begin with a tilde (~), such as ~Bias, are not applicable to the analysis that you ran and do not appear in the table, even if you place checks next to their names.

Sort by Column Sorts the columns in descending or ascending order by the selected column. You can also select Numerical Order to sort a sorted string column numerically. After you do that, the Numerical Order option is available when you right-click the column. For more information about the sorting rules used by the Numerical Order option, see "Numerical Order".

Make into Data Table Creates a JMP data table from the report table. The data table script recreates the report and the data table generated by Make into Data Table.

Make Combined Data Table Searches the report for other tables like the one you selected and combines them into a single data table. The data table contains a column that identifies the original data table. The data table script recreates the report and the data table generated by Make Combined Data Table.

Make Into Matrix Creates a JMP matrix from a report table. See "Turn a Table in a JMP Report Into a Matrix".

Select Where (Available only for categorical columns) Creates a Where clause using the selected columns or a complex expression.

Show Properties Enables you to modify the formatting of display boxes in the report.

Format Columns Changes the numeric formatting of one or more columns in the table.

Copy Column Copies the contents of the right-click column to the clipboard for pasting into another window or application.

Copy Table Copies the right-click table to the clipboard for pasting into another window or application.

Simulate Provides simulated results for the selected column of statistics in a report.

JMP PRO **Bootstrap** Approximates the sampling distribution of a statistic. See *Basic Analysis*.

Modify Display Box Properties in a JMP Report

For each display box in a report, you can change properties such as the font, text color, padding, and alignment. Your changes are included when you save the report as a script, a graphic, or a journal.

To modify display box properties, follow these steps:

1. Right-click the disclosure icon next to the report and select **Show Properties**.

   The Properties pane appears.
2. Modify the report properties.
3. Click the report to view your changes.

Tips:

- In the Properties pane, use the arrow buttons to navigate among display boxes in the display tree.
- After you open the Properties pane, select a specific report and change its properties.

Modify the Format of Numeric Columns

In the Show Properties pane, you can modify the appearance of a report, including modifying the format of selected numeric columns. For example, you might want to specify two decimal places for all numeric columns. Your changes are included when you save the report as a script, a graphic, or a journal.

To modify the numeric formats, follow these steps:

1. Right-click the disclosure icon next to the report and select **Show Properties**.

   The Properties pane appears.
2. Press Shift and select the columns whose format you want to change.
3. From the Format list in the Properties pane, select the Format properties.

Tips:

- To select all display boxes of a specific type (for example, all NumberColBoxes), select one of the boxes in the report. From the Select list in the Properties pane, select **Boxes of type 'NumberColBox'**.
- To select all columns that have a specific format, select one of the boxes in the report. From the Select list in the Properties pane, select **Columns with matching format**.

Turn a Table in a JMP Report Into a Matrix

You can create a JMP matrix from a report table. For example, when working with JMP Scripting Language (JSL), you might want to access a report's table that has been stored into a JSL variable. Or, you might want to store a report table's values into a table property as either a table property or as a JSL assignment, which is stored within the data table and is accessible via a script or the Formula Editor.

To store a table in matrix form into a global variable, into a table property, or into a table property as an assignment:

1. Right-click anywhere in a report table.
2. Select **Make into Matrix**.
3. In the window that appears, tell JMP how you want to store the table.
4. (Optional) Rename the variable or property by entering a new name into the box beside Name.

Use Conditional Formatting in Reports

Note: You must enable **Show conditional formatting** in the Reports preferences for your conditional formatting to appear in JMP reports.

To configure reports to use conditional formatting, you must first set your report preferences to enable showing conditional formatting. See "Preferences for JMP Reports".

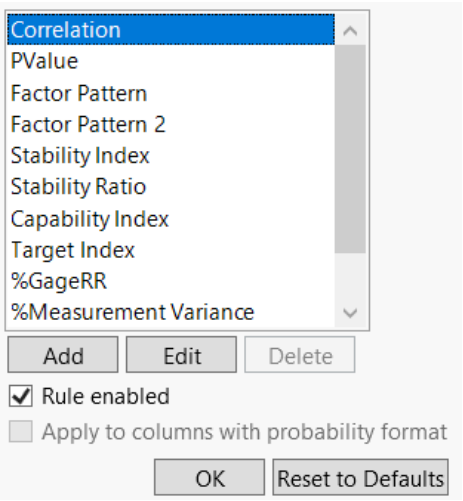
To configure conditional formatting, follow these steps:

1. Open the JMP Preferences window.

2. Select the Reports preference group.
3. Click **Manage Rules**.

   The Conditional Format Rule window appears.

Figure 9.14 Conditional Format Rule Window



All rules are enabled by default. The PValue rule is applied to columns that contain probability values.

From this window you can add, edit or delete a rule.

Disable or Enable a Rule

To disable or enable a rule:

1. Open the JMP Preferences window.
2. Select the Reports preference group.
3. Click **Manage Rules**.

   The Conditional Format Rule window appears.
4. Select the rule from the list.
5. If **Rule enabled** is selected, select to disable.
6. If **Rule enabled** is not selected, select to enable.

Add a Rule

To add a rule:

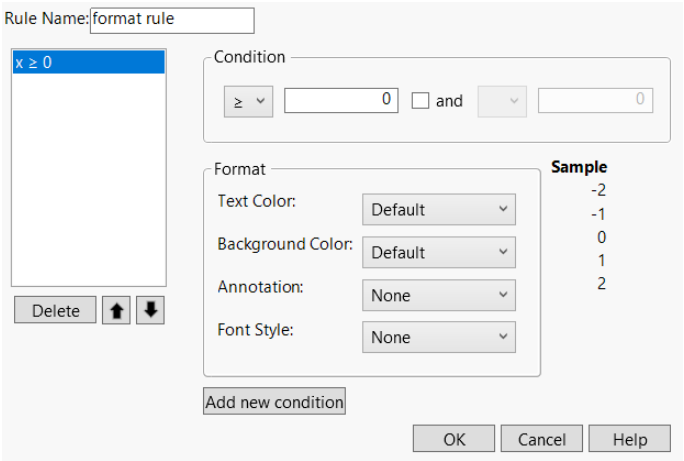
1. Open the JMP Preferences window.

2. Select the Reports preference group.
3. Click **Manage Rules**.

   The Conditional Format Rule window appears.
4. Click **Add**.

   The Conditional Format Rule window appears. A placeholder rule appears on the left.

Figure 9.15 Adding a Conditional Format Rule



5. Add and format conditions as described in "Add a Condition".
6. Click **OK** to save the new rule and return to the previous Conditional Format Rule window.
7. Select **Rule enabled** to enable the rule.

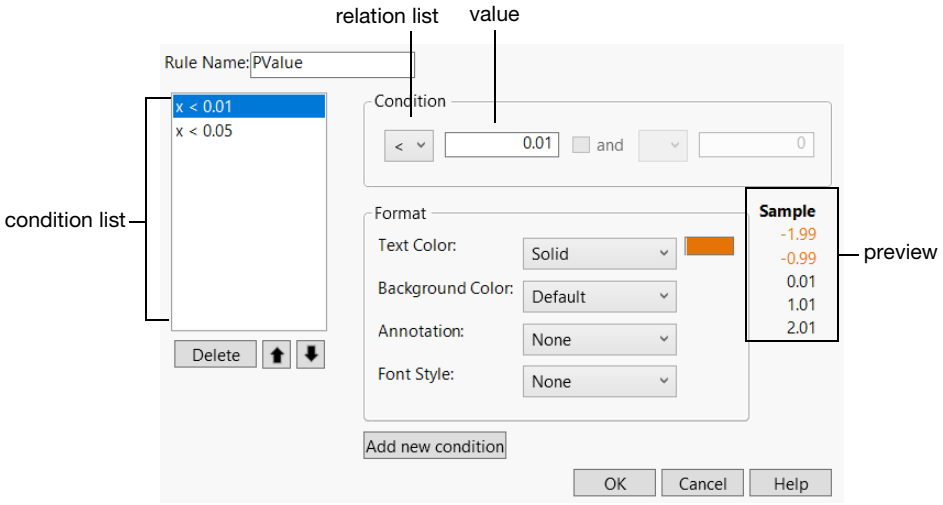
Edit a Rule

To change the conditional formatting of a rule:

1. In the Conditional Format Rule window, select the rule from the list.
2. Click **Edit**.

   The Conditional Format Rule window appears, showing the selected rule's current conditions.

Figure 9.16 Conditional Format Rule Window



The default rule for PValue includes two conditions. The order of the list indicates the order that the rules are applied. For example, a *p*-value of 0.04 would not invoke the first rule (x < 0.01) but would invoke the second rule (x < 0.05).

Note: You cannot edit an existing condition within a rule. To edit an existing condition, you must delete the condition and then add it. See "Edit an Existing Condition".

Add a Condition

To add a condition to a rule:

1. Open the rule to view the Conditional Format Rule window.
2. In the Condition area, select a relation from the list and enter a value in the text box.

   Or

   If you want the condition to include a value range:

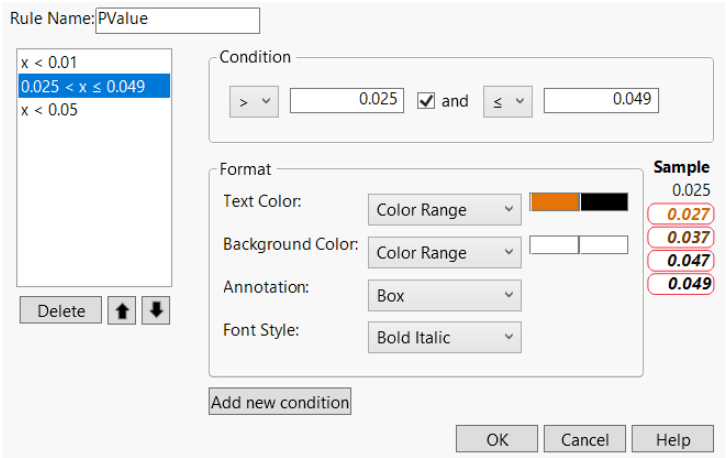
   – Select > or ≥ from the relation drop list and enter the value in the text box.

   – Select the check box, select the < or ≤ relation, and enter the value.
3. Format the condition using the procedure described in "Format a Condition".

Tip: When you click **Add new condition**, the condition and its format settings are immediately added to the bottom of the condition list and the Condition and Format areas return to their default settings. Verify your settings before you click **Add new condition**.

4. After verifying your settings, click **Add new condition** to add the condition to the rule.
5. Use the and buttons to position the condition within the list.

Figure 9.17 Example of Adding a Condition with Formatting Rules



6. Click **OK** to save your changes and return to the previous Conditional Format Rule window.

Delete a Condition

To delete a condition from a rule:

1. In the Conditional Format Rule window, select the condition and click **Delete**.
2. Click **OK** to save your changes and return to the previous Conditional Format Rule window.
3. Click **OK** to return to the Preferences window.

Format a Condition

To format values that fall within a condition:

1. With a Condition entered, select a Text Color from the drop list:

   Default The value appears without color.

   Solid The value appears in the specified color.

   Dimmed The value appears dimmed by the specified transparency.

   Color Range For ranged conditions, the values appear in color gradients within the specified color range.

   Dimmed Range For ranged conditions, values appear in dimmed gradients by the specified transparency range.

2. To specify a text color, click the color swatch and select a color.

3. To specify a transparency, enter the percent value (for example, '60%').

   The Sample area displays a preview of the appearance of the text column.

4. Select a Background Color from the drop list:

   Default The value appears without a color background.

   Solid The value appears with the specified color background.

   Color Range For ranged conditions, the values appear with background color gradients within the specified color range.

5. To specify a background color, click the color swatch and select a color.

   The Sample area displays a preview of the appearance of the text column.

6. To have condition values noted, select an Annotation from the drop list:

   None No notation is used on the value.

   Box The value appears in the table with a box around it.

   Circle The value appears in the table with an ellipse around it.

   *** (asterisk)** An asterisk (*) appears to the right of values in the table.

   The Sample area displays a preview of the appearance of the text column.

7. To style the value text, select the **Font Style** from the drop list:

   None No style is used to format the value.

   Plain No style is used to format the value.

   Bold The value appears bolded in the table.

   Italic The value appears italicized in the table.

   Bold Italic The value appears bolded and italicized in the table.

See Figure 9.17 for an example of condition formatting.

Note: After configuring the formatting, remember to click **Add new condition** to add the new condition to the rule.

Edit an Existing Condition

To change the formatting for an existing condition:

1. In the Conditional Format Rule window, select the condition and click **Delete**.
2. Re-create the condition by using the procedure described in "Add a Condition".
3. Format the conditional using the procedure described in "Format a Condition".

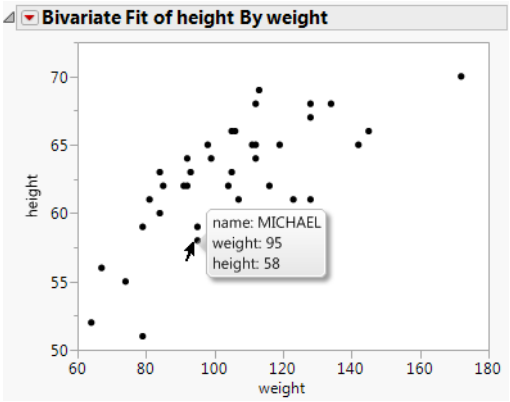
Tip: When you click **Add new condition**, the condition and its format settings are immediately added to the bottom of the condition list and the Condition and Format areas return to their default settings. Verify your settings before you click **Add new condition**.

4. After verifying your settings, click **Add new condition** to add the condition to the rule.
5. Use the Up and Down buttons to position the condition within the list.
6. Click **OK** to save your changes and return to the previous Conditional Format Rule window.

Select Points in Graphs

To select a point in a graph, click the point. This selects the point as well as the corresponding row in the current data table. To select multiple points, press Shift while you select points. A point's label appears when you place the cursor over the point with or without clicking.

Figure 9.18 Label for a Data Point



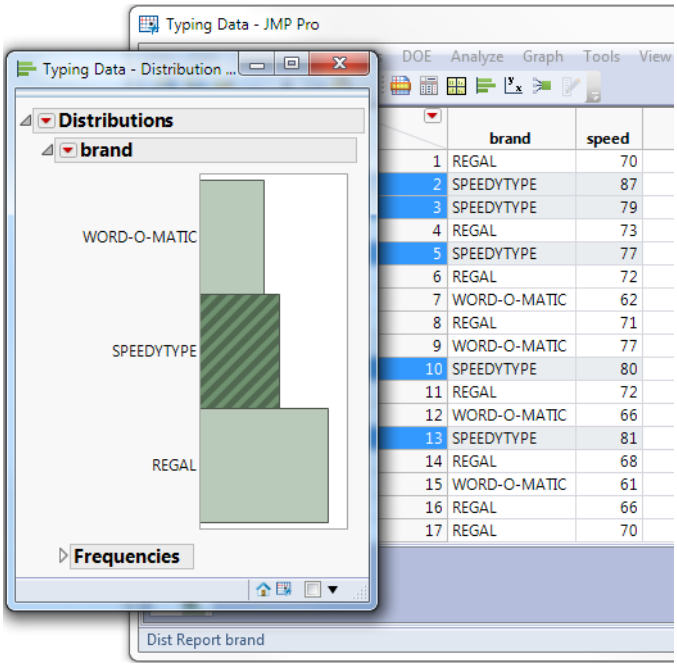
Tips:

- You can paste the contents of a label into another document. Right-click the label and select **Copy**. Paste the text into the other document.
- A photo can be displayed in the label if you store the image in an Expression role column. See "Expression Role".

Select Rows in Graphs

All graphs and plots that represent the same data table are linked to each other and to the corresponding data table. When you click points in plots or bars of a graph, the corresponding rows highlight in the data table. The example in Figure 9.19 shows a histogram with the SPEEDYTYPE bar highlighted, and the corresponding rows highlighted in the table. You can also extend the selection of bars in a histogram by pressing Shift and then making your selection.

Figure 9.19 Highlighting Rows in a Histogram



Select a Rectangular Area of Points in a JMP Graph

You can select all points that fall in a rectangular area using the cursor. Click and drag the arrow to highlight points. Alternatively, you can use the brush tool. As you move the brush over the graph, points that fall within the rectangle are selected. Any points marked in the data table as hidden are not selected. See "Hide Rows in Data Tables".

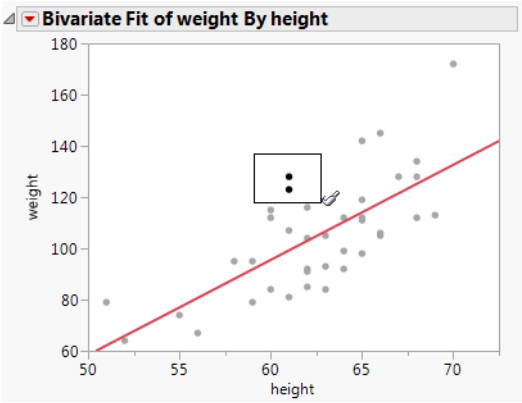
To select points using the brush tool, follow these steps:

1. Click the brush tool in the toolbar.
2. Click and hold the brush tool in a plot. A rectangle appears.

3. Move the rectangle over points. As it passes over them, they appear highlighted both in the plot and in the active data table.

   To keep all points selected as you move the brush tool over points, press Shift before you click a point in the plot. The selected points are also selected in the data table.

Figure 9.20 Using the Brush Tool



4. Release the mouse. The points within the rectangle and the data table remain selected.
   - To change the size of the selection rectangle, press Alt before you click in the plot. Drag the cursor to resize the selection box. This shape acts like a slicing tool that can traverse and highlight slices of points across either axis. **Note:** The size of the selection box is remembered for the next time you use the brush tool.
   - If you press Ctrl and click the brush tool on selected points, the points within the selection rectangle are deselected. Points outside the selection rectangle remain selected.

Select an Irregularly Shaped Area of Points in a JMP Graph

You can use the lasso tool to select points that fall in an irregularly shaped area. Any points marked in the data table as hidden are not selected. See "Hide Rows in Data Tables".

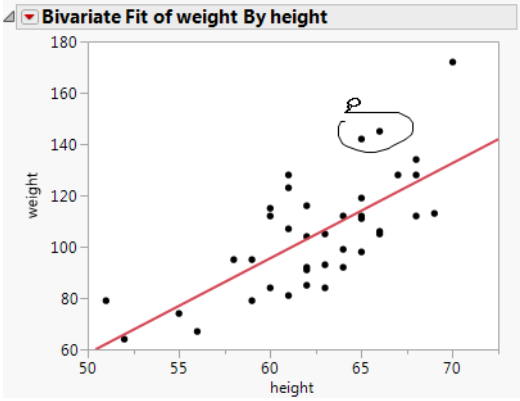
To select points within an irregularly shaped area, follow these steps:

1. Click the lasso tool in the toolbar.
2. Click and hold the lasso tool in a plot.

   Note: To keep all points selected as you drag the lasso around several sets of points, press Shift before you click in the plot.

3. Drag the lasso around any set of points.

Figure 9.21 Using the Lasso Tool



4. Release the mouse. JMP automatically closes the lasso and highlights the points within the enclosed area.

Use Markers in Graphs

Markers are points on a graph that represent data. Once they are changed from their default setting, they also appear next to rows in the data table.

- "Change the Shape of Markers"
- "Change the Color of Markers"
- "Change the Size of Markers"
- "Work with Crowded Markers"
- "Add Outlines Around Markers"
- "Change the Selection Mode of Markers"
- "Specify the Transparency of Markers"
- "Exclude and Hide Markers"
- "Use Images as Markers"
- "Change the Shape or Color of Markers Based On Values"

Change the Shape of Markers

You can assign a character from the JMP markers palette to replace the standard points in scatterplots. These markers also appear next to row numbers in the data table.

1. Highlight one or more markers whose shape you would like to change.

2. Right-click anywhere in the graph. In a histogram, right-click the box plot area on the right.
3. Select **Row Markers**.
4. Select a marker shape from the options that appear, or click **Other** to enter a character to use as a marker.

Change the Color of Markers

You can assign any color to highlighted rows. When you do this, the points in scatterplots appear in the color that you select from the colors palette. The active color assigned to a row appears next to the row number in the data grid.

To change the color of markers (or points) on a graph:

1. Highlight one or more markers whose color you would like to change.
2. Right-click anywhere in a graph. In a histogram, right-click the box plot area on the right.
3. Select **Row Colors**.
4. Select one of the colors, or click **Custom** to apply a custom color.

Change the Size of Markers

To increase or decrease the size of markers on a graph:

1. Right-click anywhere in a graph. Press Ctrl and right-click to broadcast the command and apply it to all plots of the same type located in the same window. In a histogram, right-click the box plot area on the right.
2. Select **Marker Size**.
3. Select one of the marker sizes listed.

   The default value, Preferred Size, is selected on the Graphs page in the JMP preferences.

Note: When the drawing speed is set to Fast, the smallest marker size (Dot) is applied. See "Work with Crowded Markers".

Work with Crowded Markers

A large number of markers on a graph can appear crowded on a graph. You might need to alter the transparency to gain a better view. Altering the transparency might also affect the marker drawing mode, which is the mode JMP uses when it refreshes a report window. As markers are drawn on a plot, JMP uses one of two speeds: normal or fast.

To change the marker drawing speed

1. Right-click anywhere in a graph. In a histogram, right-click the box plot area on the right.
2. Select **Marker Drawing Mode**, and then select either **Normal** or **Fast**.

Normal If JMP is in normal drawing mode and the number of markers on a graph are more than the specified threshold number, JMP automatically switches to fast mode. See "Preferences for JMP Reports" for more information about setting the marker threshold.

Fast Graphs that display a large number of markers appear faster if you set the marker drawing speed to Fast. Note that when the drawing speed is set to Fast, the smallest marker size (Dot) is applied, and marker transparency settings revert to the default opaqueness.

Outlined See "Add Outlines Around Markers".

To fill hollow markers

In the JMP preferences, select **Graphs** and then select **Fill Hollow Markers**. Hollow markers on all graphs are filled with an opaque background.

Add Outlines Around Markers

You can add a black outline, or frame, to markers in a plot. Outlined markers are available at the medium, larger, XL, XXL, and XXXL marker size. See "Change the Size of Markers".

To add outlines

1. Right-click a plot or graph.
2. Select **Marker Drawing Mode**.
3. Select **Outlined**.

To use an outline effectively, it is best if your marker is a color other than black.

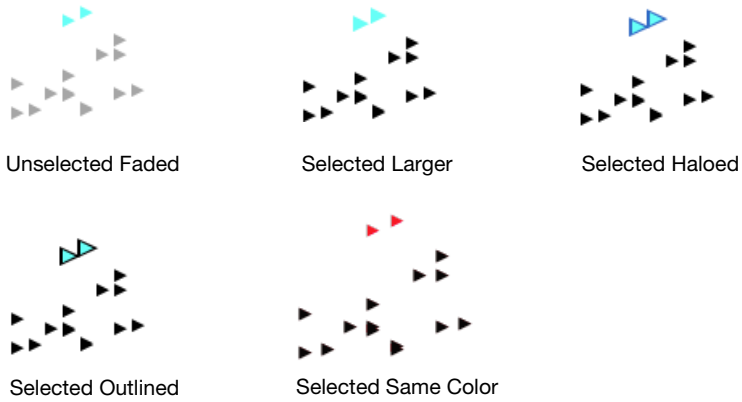
To change marker colors

1. Highlight the markers whose color you want to change.
2. Right-click anywhere in the graph.
3. Select **Row Colors**.
4. Select a marker color from the options that appear.

Change the Selection Mode of Markers

When you select markers on a graph, only the selected markers are highlighted. You can change how markers are highlighted on the current graph. The options are applied to the top two triangles in the following figure.

Figure 9.22 Examples of Highlighted Triangular Markers



To change the marker highlighting on the current graph, follow these steps:

1. Right-click anywhere in a graph and select **Marker Selection Mode**.
2. Select one of the following options:

Preferred Mode The Marker Selection Mode that is selected on the Graphs page in the JMP preferences. The default value is Unselected Faded.

Unselected Faded Only the selected markers are highlighted. Everything else is dimmed by the percent specified by **Faded amount for unselected markers** in Preferences > Graphs.

Selected Larger The selected markers are larger than the deselected markers.

Selected Haloed The edges of the selected markers are outlined in blue.

Selected Outlined The selected markers are outlined in black.

Selected Same Color The selected markers are shaded with the Marker Selection Color that is selected in the Graphs preferences.

Specify the Transparency of Markers

You can change the transparency of markers (points) on a graph. For example, this enables you to control the visibility of overlapping points.

Note: When the drawing speed is set to Fast, the default marker transparency is applied. See "Work with Crowded Markers".

1. In a graph, right-click anywhere and select **Transparency**. In a histogram, right-click the box plot area on the right and select **Transparency**.
2. Enter the level of transparency that you want the markers (points) to have on the graph, and click **OK**.

   A value of 1 indicates total opaqueness and 0 indicates invisibility. Values between 1 and 0 are semi-transparent.

Exclude and Hide Markers

Use the Rows > Hide and Exclude command to suppress the appearance and exclude from statistical analyses the highlighted rows. Data remains hidden and excluded until you select Rows > Hide and Exclude again.

Using the Exclude/Unexclude command, you can exclude highlighted rows from statistical analyses. Data remains excluded until you select Rows > Exclude/Unexclude for those highlighted rows.

Note: Excluded data are not automatically hidden in plots even though they are excluded from calculations in text reports and graphs.

Using the Hide/Unhide command, you can suppress (hide) the appearance of highlighted points in scatterplots. For example, you can exclude points from analysis and then hide those same points in scatterplots. The data remain hidden until you select Rows > Hide/Unhide for highlighted hidden rows.

Note: Hidden points are not automatically excluded from statistical computations that affect text reports and graphs, even though they are not displayed in the plots. To exclude hidden observations from analyses, you must highlight them and select Rows > Exclude/Unexclude characteristic.

To exclude or hide markers (points) from analyses, follow these steps:

1. Highlight the marker(s) that you would like to exclude or hide.
2. Right-click anywhere in a graph.

3. Select **Row Exclude** or **Row Hide**.

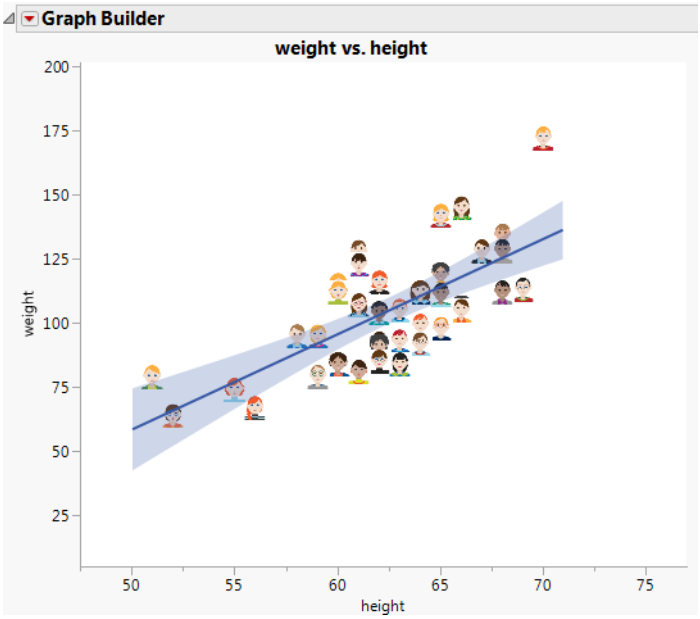
Use Images as Markers

Place images in an expression column to use those images as markers on graphs. Then you indicate which column to use for markers. This feature prevents the need to look at a marker color and refer to the legend for its meaning, which is particularly helpful with data that contain many levels.

If you use a character or numeric column as the marker, the data value is used for the marker.

1. Create a new column in your data table.
2. Right-click the column, select **Column Info**, and select the **Expression** data type.
3. Click **OK**.
4. Drag graphics from your computer into each cell of the column. You can also import data that contains a column of images.
5. Select the column, select the **Cols** menu, and then select **Use for Marker**.
6. Create a graph that uses markers. Figure 9.23 was created in Graph Builder.
7. (Optional) To resize the markers, right-click the graph, and select **Graph > Marker Size** (or right-click and select **Marker Size** in some platforms). Medium markers are selected in this example.
8. Click the Graph Builder red triangle and deselect **Show Legend**.

Figure 9.23 Pictures Used as Markers



Notes:

- In the Big Class Families.jmp sample data table, the picture column used for markers on graphs.
- This feature is not supported in Bubble Plot.

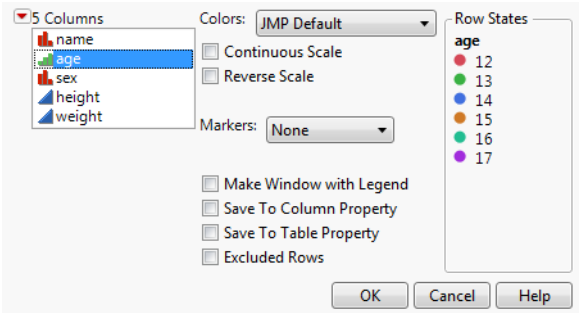
Change the Shape or Color of Markers Based On Values

In some plots, you can change marker shapes or colors based on the values of points by adding a *row legend*. It is called a row legend because JMP automatically inserts a legend using row color or row marker settings. When you assign markers or colors in this way, it assigns the characteristic(s) to all points in a graph, regardless of what points you have selected. All previous marker and color settings are overwritten.

To add shapes or colors based on column values, follow these steps:

1. Right-click anywhere in a graph. In a histogram, right-click the box plot area on the right.
2. Select **Row Legend**.
3. In the window that appears, highlight the column whose values you want to color or mark. A preview of the legend is shown on the right.

Figure 9.24 Adding a Row Legend



4. Refine your row legend using the following options:

Colors Lets you choose among several pre-defined color schemes.

Tip: If you already applied a color scheme in the data table, select None to maintain it. Colors should be consistently applied and uniquely grouped.

Continuous Scale Assigns colors on a spectrum that corresponds to the ascending or descending order of the values. Use this option when the highlighted column contains continuous values.

Reverse Scale Reverses the scale of colors.

Markers Lets you choose among several marker schemes.

Tip: If you already applied a marker scheme in the data table, select None to maintain it. Markers should be consistently applied and uniquely grouped.

Make Window with Legend Creates a separate legend window that tells you what colors and shapes correspond to which value.

Save to Column Property Adds a column property that stores the selected color theme.

Save to Table Property Adds a table property that preserves the selected color and marker configuration.

Excluded Rows Assigns colors or markers to rows that are excluded.

Most legends have one column. However, the following platforms have multi-column legends when there are more than 20 levels:

- Recurrence
- Oneway (for CDF Plot and all three Densities red triangle commands)
- Fit Model (for the Regression Plot in Standard Least Squares)

- Survival

Remove the Row Legend

Delete a row legend by right-clicking it and selecting **Remove**.

Change the Appearance of Graphs

There are many ways that you can format your report to meet your needs. The sections that follow detail how to make changes to the graphical portions of your output reports.

Tip: If you have a touchscreen, you can pan and zoom most graphs and axes in JMP. Graphs and axes change scale and zoom out if pinched, and zoom in if stretched.

- "Resize Graphs"
- "Zoom In on Graphs"
- "Change Line Widths in Graphs"
- "Change the Background Color in Graphs"
- "Change the Color of Histogram Bars"
- "Display Coordinates and Temporary Reference Lines in Graphs"
- "Scroll and Scale Axes in Graphs"
- "Customize Axes and Axis Labels in Graphs"
- "Change the Order of Values in Reports"
- "Change the Pattern and Format of Selected Objects in Graphs"
- "Add a Graph to a Data Table"
- "Customize Markers, Lines, Text, and More in Graphs"

Resize Graphs

There are two main ways to resize graphs in JMP: using the click and drag method or resizing it according to pixel size.

Note: You can also change the default size of a graph using the Graph Height option in File > Preferences > Graphs.

Use Click and Drag

To resize a plot or graph using the click and drag method:

1. Place the cursor on the right edge, bottom edge, or lower right corner of the plot frame. The cursor changes to a small double-arrow.
2. Click and drag to change the size of the plot. When you resize, the height and width of all plots in that frame adjust independently of other frames in the same report window. Table 9.2 describes how to adjust the plot.

Table 9.2 Resizing Actions

Action	Instructions
Adjust the plot frame but preserve the proportions (aspect ratio)	Press Shift and click and drag the corner of the frame.
Adjust a plot in 8-pixel increments	Press Alt and click and drag the corner of the frame.
Adjust all plots of the same type simultaneously	Press Ctrl and click and drag the corner of one of the plots. If you do this for one scatterplot, the action is broadcast to all scatterplots in the window, and they resize together. Any other types of plots remain unchanged.

Specify Graph Size in Pixels

To resize a plot or graph to a specific pixel size:

1. Right-click the plot or graph.
2. Select **Size/Scale** > **Frame Size**.
3. Enter the number of pixels for the frame's height and width.

Note: For more information about the other options in the Size/Scale menu, see "Scroll and Scale Axes in Graphs".

Zoom In on Graphs

The magnifier lets you automatically zoom in on any area of a plot. When you click the magnifier, the point or area where you click becomes the center of a new view of the data. The scale of the new view enlarges, giving you a closer look at interesting points or patterns. You can perform any of the following actions:

- Click and drag the magnifier to focus in on a particular region of the plot.
- On a ternary plot, drag the magnifier to zoom the triangular axes.
- Zoom repeatedly to look closer at the data.
- Press Ctrl and click the magnifier to return to your previous state before the last zoom.
- Double-click or press Alt and click the magnifier to restore the original plot.

On Windows, you can also zoom by pressing Ctrl and using the mouse wheel.

On macOS, zoom by pressing Option and clicking the area.

Note: You can zoom in and still see a center point by dragging from the axis to inside the graph. For a vertical axis, click a position and drag right into the graph. This zooms in on the scale, adjusting both endpoints of the scale but keeping the clicked value in about the same position. Dragging left zooms out.

Change Line Widths in Graphs

After fitting a line to a graph, or producing a graph with a line already present, you can adjust the width of the line:

1. Right-click anywhere in a graph.
2. Select **Customize.**
3. Select **Smoother** (for example, if the graph contains a smoother).
4. Change the line width. The value is in pixels.
5. Click **OK**.

Change the Background Color in Graphs

To change the background color in a graph, follow these steps:

1. Right-click anywhere in a graph. (To change only the color of a box plot, right-click the box plot area.)
2. Select **Background Color**.
3. Select one of the predefined colors, or create your own color.

4. Click **OK**.

Change the Color of Histogram Bars

To change the color of histogram bars, follow these steps:

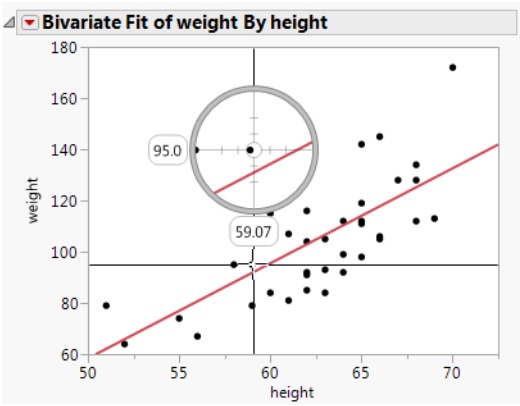
1. Right-click anywhere in a histogram and select **Histogram Color**.
2. Click a color, or click **Other** and create your own color.

See your operating system documentation for more information about creating your own colors.

Display Coordinates and Temporary Reference Lines in Graphs

You can measure points and distances in graphs, or easily find the exact value, or coordinates, of points and distances on plots and graphs. To do this, click the crosshairs tool and click and hold anywhere on a graph. The coordinate values appear where the crosshairs intersect the vertical and horizontal axis as you drag the crosshairs within a plot.

Figure 9.25 Using the Crosshairs Tool



On a fitted line or curve, the crosshairs identify the response value for any predicted value. On a ternary plot, this tool displays triangular crosshair lines.

Scroll and Scale Axes in Graphs

The hand tool (also known as the grabber tool) ( ) provides a way to change the axes and view of a plot:

- On a *y*-axis, dragging scrolls the *y*-axis; dragging or scales the *y*-axis.

- On an *x*-axis, dragging scrolls the *x*-axis; or scales the *x*-axis.

Tip: When you drag an axis to change its scale, JMP automatically updates the major and minor tick increments based on the new axis width. To prevent this (retaining your original increments), press Shift while dragging.

You can also right-click in a plot or graph, and select **Size/Scale** (or **Graph > Size/Scale**). Choose one of the following options:

- To adjust the scale of the X axis, select **X Axis**. To adjust the scale of the Y axis, select **Y Axis** or **Right Y Axis**. For more information about this window, see "Customize Axes and Axis Labels in Graphs".
- For more information about the Frame Size option, see "Specify Graph Size in Pixels".
- Select **Size to Isometric** when the *x*- and *y*-axes are measured in the same units and you want distances on the graph to be represented accurately regardless of direction.

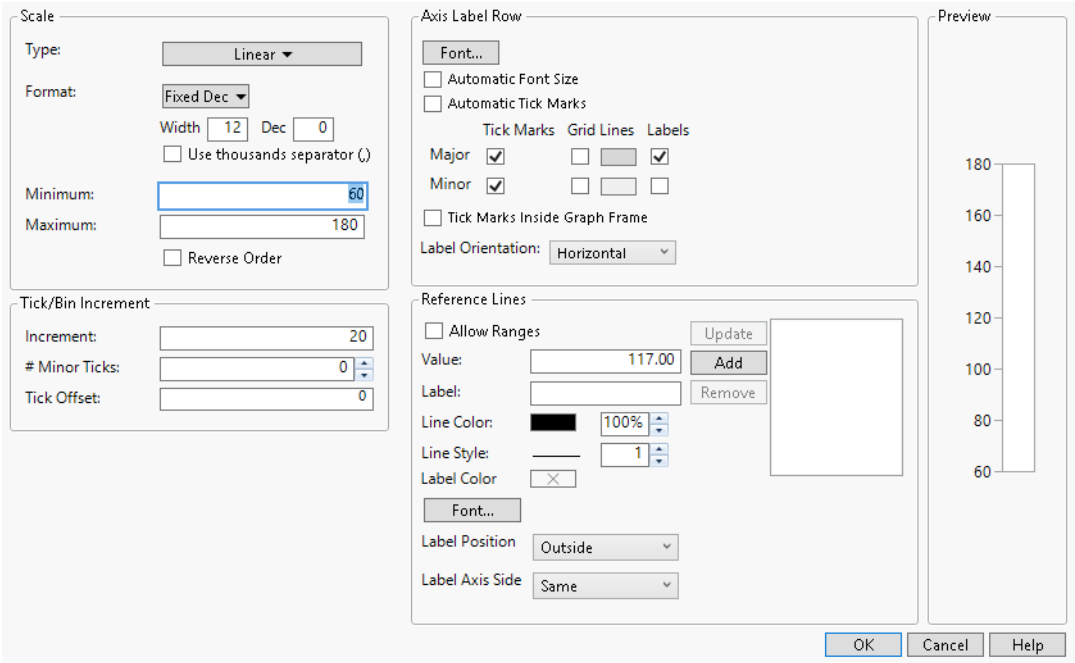
Customize Axes and Axis Labels in Graphs

Double-click a numeric axis to customize it using the Axis Settings window. Or, right-click the axis area and select **Axis Settings** to access the window.

Note: In the Graph Builder and Distribution platforms, you can also customize nominal axes using the Axis Settings window.

Customization features in the window depend on the data type of the axis and the specific platform JMP uses to create the plot or chart. Figure 9.26 shows a typical Axis Settings window for numeric (continuous) axes.

Figure 9.26 Axis Settings Window for a Numeric (Continuous) Axis



The Axis Settings window contains the following panels:

- "Scale"
- "Tick/Bin Increment"
- "Axis Label Row"
- "Reference Lines"
- The Preview panel shows how your current selections will appear on the axis.

Scale

In the Scale panel, you can do the following:

- "Change the Axis Scale Type"
- "Change the Numeric Format of an Axis"
- "Establish Minimum and Maximum Axis Values"

Change the Axis Scale Type

When viewing a graph with a numeric axis, you can change the axis scale to one of the following types:

- Linear

- Log
- Power
- Geodesic
- Geodesic US
- Probability Scales (Normal, Weibull, Fréchet, Logistic, Exponential, Gamma, Beta, and Mixtures)

Notes:

- Specific platforms might use other scale types that are fixed and cannot be changed. For example, Destructive Degradation, Degradation, and the X axis in Fit Y by X scatterplots use custom scales. Do not change the scale in these graphs through the Axis Settings window.
- If you selected a scale type of Log, enter the Base to use.
- If you selected a scale type of Power, enter the Power to use.
- Power works similarly to log scale, but instead of the scale conversion being log(x), it is x^p, where p is a customizable axis option. For example, Power = 2 creates an x^2 axis.

To set a default scale type for a variable, which avoids making this change every time you run an analysis, see "Axis".

Change the Numeric Format of an Axis

For plots and charts that contain a numeric axis area, you can change the format of the axis. For more information about numeric formats, see "Numeric Formats".

Notes:

- If you selected Date, Time, or Duration, you need to specify the format of the increments. See "Add and Remove Axis Labels". You can also specify label row nesting. See "Label Row Nesting".
- If you selected Fixed Dec, enter the number of decimal places that you want JMP to display in the Dec box.
- If you selected Precision, select whether you want to keep trailing zeros and all whole digits.
- To add commas to values that equal a thousand or more, select the **Use thousands separator** option. You must account a space for each comma in the Width box, or else they might not appear. This option is available for the Best, Fixed Dec, Percent, and Currency formats.
- For a numeric axis, you can adjust the width of the tick mark labels using the Width box.

Note: When you change the numeric format of an axis, you do not change the numeric format of the way the values appear in the corresponding data table. To change how a date or time appears in a data table, see "Numeric Formats".

Selecting a date interval from the date increment drop-down menu divides the JMP date (number of seconds) into the appropriate units. This gives the plot scale that you want for your data. The date axis must be a column with a JMP date value and appear in the Axis Settings window in the date format found in the Column Info window. However, you can use the Axis Settings window to format the date's appearance in the plot.

Establish Minimum and Maximum Axis Values

For plots and charts that contain a numeric axis area, you can change the minimum and maximum values that you want the graph to display.

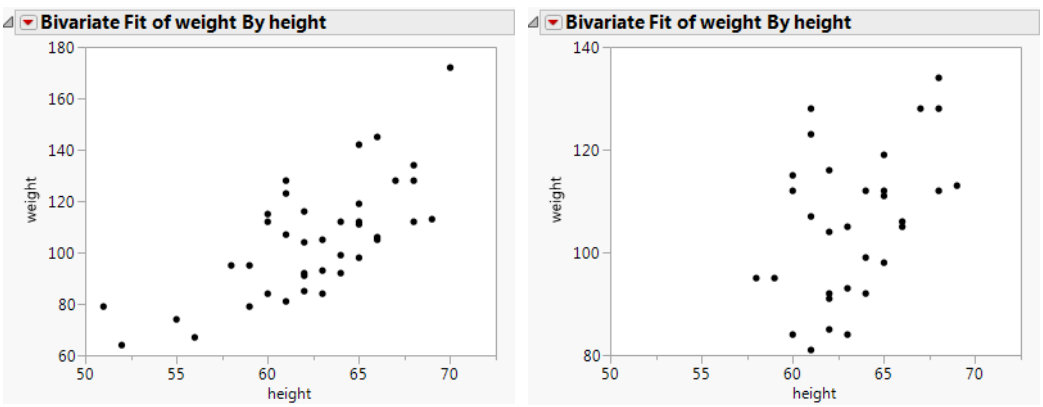
Note the following:

- To restore the default minimum, maximum axis settings, and tick increments of an axis, right-click on the axis labels and select **Revert Scale**. To restore all default axis settings, right-click the axis labels and select **Revert Axis.**
- Select **Reverse Order** to reverse the axes by reversing the minimum and maximum values. Note that this option is not supported for ternary plots.

Tip: To set a default minimum and maximum axis value for a variable, which avoids making this change every time you run an analysis, see "Axis".

The example on the right in Figure 9.27 is an enlargement of the point cluster that shows between 80 and 140 in the plot to the left. The enlarged plot is obtained by reassigning the maximum and minimum axis values.

Figure 9.27 Rescale Axis to Enlarge a Plot Section



Tick/Bin Increment

In the Tick/Bin Increment panel, you can do the following:

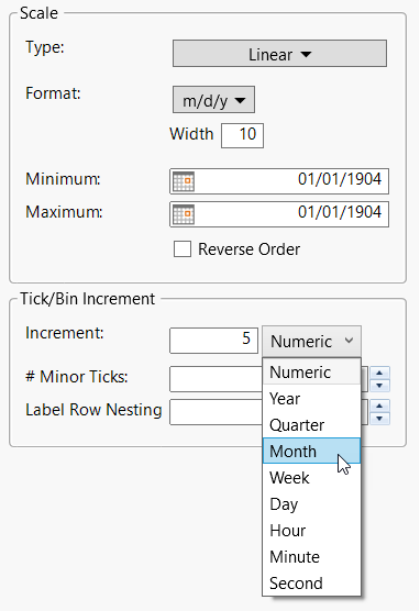
- "Change Axis Increments"
- "Add Minor Tick Marks"
- "Change the Tick Offset"
- "Label Row Nesting"

Change Axis Increments

While viewing a graph, you can change the axis increments.

If the axis Format is set to Date, Time, or Duration, a format menu appears beside Increment (Figure 9.28). Select the format for the increments.

Figure 9.28 Selecting the Format for Date and Time Increments



Tip: To set a default axis increment for a variable, which avoids making this change every time you run an analysis, see "Axis".

Add Minor Tick Marks

You can add tick marks to a numeric axis, or change the number of minor tick marks that appear on a numeric axis. In the Axis Label Row panel, be sure to select the **Minor** Tick Marks box so that the tick marks appear on the axis.

Tip: To set default minor tick marks for a variable, which avoids making this change every time you run an analysis, see "Axis".

Change the Tick Offset

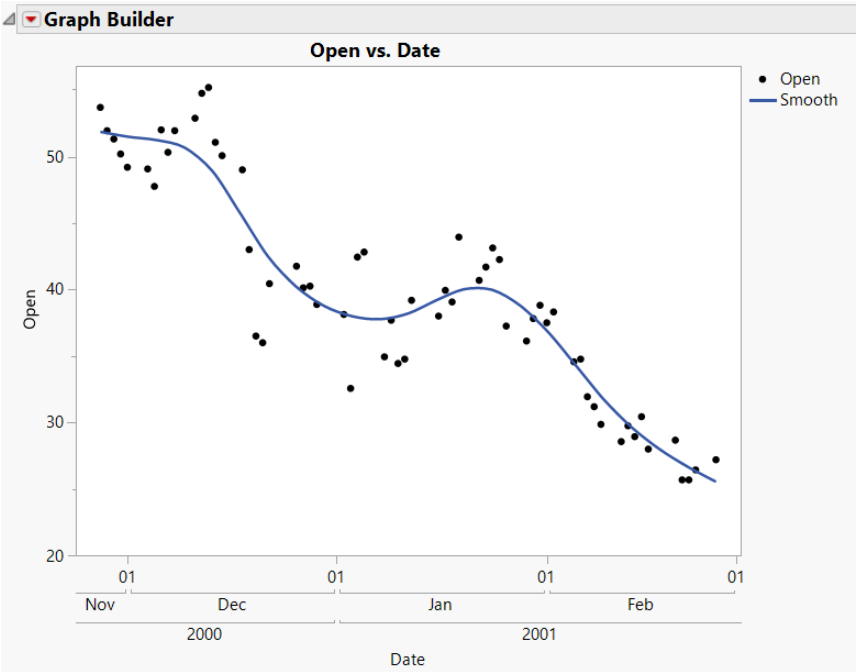
To change the starting point of the tick marks, enter a number in the Tick Offset box. For example, if the Tick Offset is currently set to 0, setting it to 1 will move all the values on the axes up by 1.

Label Row Nesting

This option appears only if you have selected Date, Time, or Duration as the axis Format. Label row nesting enables you to split a date axis into multiple rows based on the format. For example, you might put the year on the outermost row, then the month, then the day:

1. Select **Help > Sample Data Library** and open Stock Prices.jmp.
2. Select **Graph > Graph Builder**.
3. Drag Date into the **X** zone.
4. Drag Open into the **Y** zone.
5. Double-click the X axis.
6. For **Format**, select **Date > ddMonyyyy**.
7. Increase the value for **Label Row Nesting** to 3.
8. Click **OK**.
9. Click **Done** to see the finished graph.

Figure 9.29 Example of Label Row Nesting



Axis Label Row

In the Axis Label Row panel, you can do the following:

- "Change the Axis Label Font"
- "Add Tick Marks, Grid Lines, or Labels"
- "Change the Orientation for Tick Labels"

Note: In Graph Builder, if you have nested labels in an axis, tabs appear in the Axis Label Row panel. You can modify each row of labels separately in its own tab.

Change the Axis Label Font

You can modify the axis label font on any axis type. Your changes only apply to the active graph. To set the default axis label font, see "Fonts Preferences".

To change the current font type and size, double-click an axis, select **Axis Settings**, and then click **Font**.

Notes:

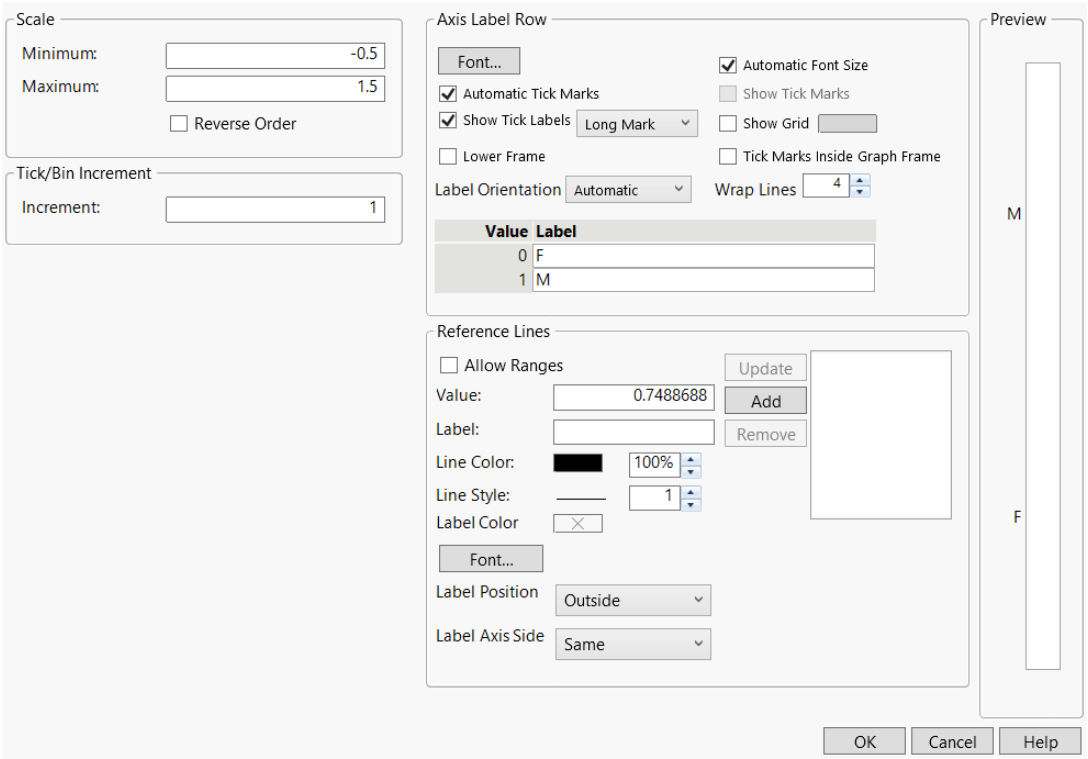
- Select **Automatic Font Size** to have JMP attempt to decrease the font size (down to a certain minimum) if all of the labels cannot fit at the default size.

- To change the axis label font color, right-click the axis and select **Show Properties**. In the Edit options on the right, change the text color.

Add Tick Marks, Grid Lines, or Labels

These options can vary depending on whether your axis is continuous or categorical. Figure 9.30 shows axis settings for a categorical axis. Figure 9.26 shows axis settings for a continuous axis.

Figure 9.30 Axis Label Row for a Categorical Axis



Options for *continuous* axes:

- Select **Automatic Tick Marks** to turn on tick marks only if one or more labels are hidden due to insufficient space.
- Add major or minor tick marks, grid lines, or labels, by selecting the corresponding check boxes.
- Click the color box to change the color of grid lines.
- Select **Tick Marks Inside Graph Frame** to move the tick marks inside the graph.

Options for *categorical* axes:

- The default is to not show tick marks if all of the labels are visible. To add tick marks, select **Automatic Tick Marks**. Change the tick mark style by selecting one of the options.
- Select **Automatic Tick Marks** to turn on tick marks only if one or more labels is hidden (due to insufficient space).
- Add grid lines by selecting **Show Grid**.
- Click the color box to change the color of grid lines.
- Select **Tick Marks Inside Graph Frame** to move the tick marks inside the graph.
- Select **Lower Frame** to add a frame around the labels.
- Change the number of lines that text wraps.
- Modify the labels by entering different values in the Label boxes.

Tips:

- To set default tick marks for a variable, which avoids making this change every time you run an analysis, see "Properties That Control the Display of Columns".
- For additional axis customizations, select **Show Properties** from the axis right-click menu.

Change the Orientation for Tick Labels

To change the orientation for tick labels, select an option from the Label Orientation list.

Automatic Tick labels are oriented automatically to be readable, depending on the tick increment and length of the labels.

Horizontal Baseline at the bottom.

Vertical Vertical with the baseline on the right.

Perpendicular Horizontal for vertical axes and vertical for horizontal axes.

Note: Parallel and Perpendicular are useful with paired axes, like in Multivariate plots.

Parallel Vertical for vertical axes and horizontal for horizontal axes.

Angled Angled at about 45 degrees.

Reference Lines

Add reference lines to graphs:

1. Double-click a numeric axis. Or, right-click a numeric axis and select **Axis Settings**.
2. In the Value text box, enter the value to which you want the reference line to correspond. This is the position on the graph at which the line is placed.

3. Enter a Label for the line.
4. You can further customize the reference line by choosing either of these options:

   Allow Ranges Enables you to enter a minimum and a maximum value, which define the beginning and the end of the reference line. The reference line appears over a range of data.

   Line Color Changes the line color. You can also specify the opacity.

   Line Style Specifies the line style (when Allow Ranges is not selected). You can also specify the line width.

   Font Enables you to adjust the font for the reference line label.

   Label Position Enables you to select a position for the reference line label. Options are outside or inside the graph frame.

   Label Axis Side Enables you to select the side of the graph for the reference line label.
5. Click **Add**. The value moves into the box to the right of the Add button, indicating that it will be placed on the graph. Your changes appear in the Preview window.
6. To add more lines, repeat the above steps.
7. To modify an existing reference line, make your changes, and then click **Update**.

Tip: To set a default reference line for a variable, which avoids making this change every time you run an analysis, see "Properties That Control the Display of Columns".

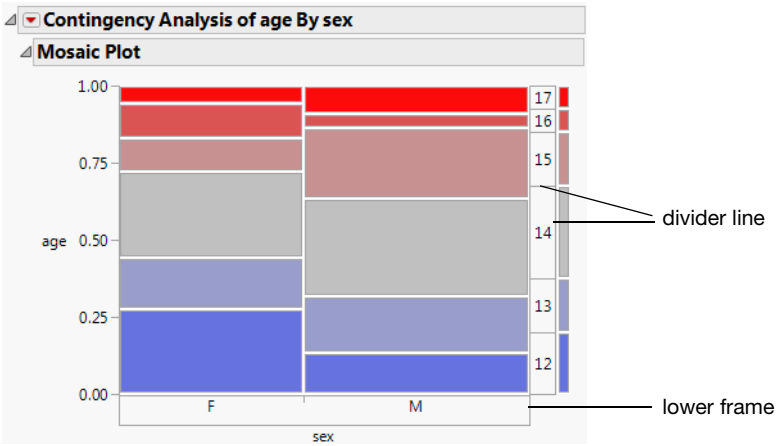
Extend Divider Lines and Frames for Categorical Axes

Extending the vertical divider line(s) between tick labels is useful when there are many levels of a nominal or ordinal (categorical) variable.

To extend the divider line to the *x*-axis labels, follow these steps:

1. Double-click a nominal or ordinal axis to show the axis settings.
2. Select **Show Tick Marks > Short Divider of Show Tick Marks > Long Divider** to add the divider lines, or **Lower Frame** to add a frame around the axis area.

Figure 9.31 Divider Lines



Add and Remove Axis Labels

You can add or remove labels in a numeric axis. To add an axis label:

1. Right-click a numeric axis and select **Add Axis Label**.
2. Enter a name for the axis label. The axis area enlarges to hold the number of label lines that you enter.

This command can be used multiple times to add multiple labels. To edit the label after it has been added to the axis, click it and it will turn into an edit box.

To remove an axis label, right-click a numeric axis and select **Remove Axis Label**. The last label added is removed.

Rotate Axis Labels and Tick Marks

You can modify the axis label on any axis type.

To rotate an axis label, follow these steps:

1. Double-click an axis to show the axis settings.
2. From the Label Orientation list, select which direction to rotate the text.

Tip: To set a default axis label position for a variable, which avoids making this change every time you run an analysis, see "Properties That Control the Display of Columns".

You can rotate tick marks on a nominal axis by right-clicking the tick label and selecting **Rotated Tick Labels**.

Copy and Paste Graph Contents

After customizing a graph by adding elements such as a fitted line, you can copy and paste the contents from one graph to another compatible graph:

1. Right-click the graph that you have customized.
2. Select **Edit > Copy Frame Contents**.
3. Right-click the graph to which you would like to copy the settings.
4. Select **Edit > Paste Frame Contents.**

Tip: If you are having difficulty pasting histogram bars, see the JSL workaround the *Scripting Guide*.

Copy and Paste Axis Settings

After customizing an axis (as described in "Customize Axes and Axis Labels in Graphs"), you can copy and paste your new settings to another axis:

1. Right-click the axis that you have customized.
2. Select **Edit > Copy Axis Settings**.
3. Right-click the axis to which you would like to copy the settings.
4. Select **Edit > Paste Axis Settings**.

Tip: To broadcast the axis settings to multiple graphs, press Ctrl when you right-click the axis and select **Edit > Paste Axis Settings**.

Change the Order of Values in Reports

Data in a JMP report might not appear in the order that you prefer. To give data a specific order so that it appears that way in a report, assign the column the Value Order property before running the analysis, as described in "Value Order".

If your values include any of the following, they automatically appear in the appropriate order in reports:

- January, February, March, April, May, June, July, August, September, October, November, December
- Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
- Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday
- Very Low, Low, Medium Low, Medium, Medium High, High, Very High
- Strongly Disagree, Disagree, Neutral, Indifferent, Agree, Strongly Agree

- Failing, Unacceptable, Very Poor, Poor, Bad, Acceptable, Average, Good, Better, Very Good, Excellent, Best

Change the Pattern and Format of Selected Objects in Graphs

Patterns appear on some objects such as bars and maps when you select them. You can change the pattern for a specific object in the legend.

Right-click the item in the legend and select **Fill Pattern** to select a new pattern.

To remove the pattern, right-click the item again and select **Revert**.

Modify the Format of Selected Objects in All Graphs

Select **File > Preferences > Graphs** (Windows) or **JMP > Preferences > Graphs** (macOS) to change the Filled Areas preferences.

- To change the highlight on selected objects, select one of the following Filled Selection Mode preferences:

  Selected Patterned Striped pattern (the default setting)

  Selected Darker Solid darker color

  Selected Outlined Outline

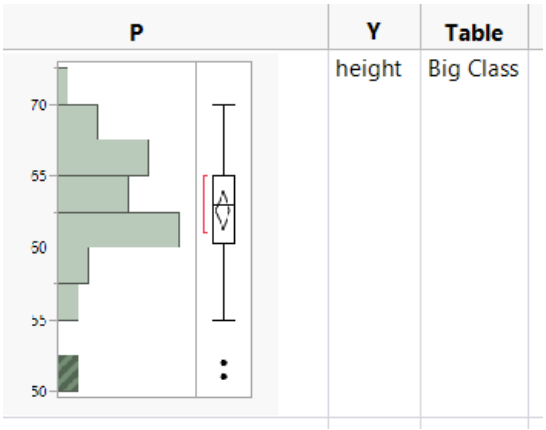
  Selected Same Color The Fill Selection Color in the preferences formats the selected object.

- To change the color of selected rows in graphic objects when the Fill Selection Mode is "Selected Same Color", click the Fill Selection Color bar. The default color is red.

Add a Graph to a Data Table

You can add a graph to a new data table. Right-click a graph and select **Edit > Make table of graphs like this**. The graph and the variables appear in a new data table. You then resize the row and column to enlarge the graph.

Figure 9.32 Histogram and Y Variable Added to Data Table



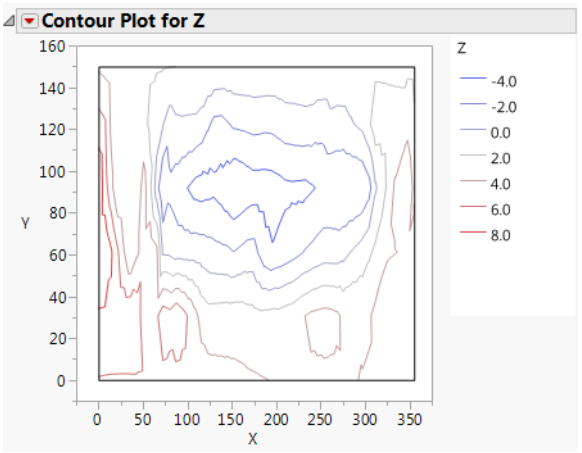
Customize Markers, Lines, Text, and More in Graphs

Graphs consist of markers, lines, text, and other graphical elements that you can customize. For example, in a Scatterplot Matrix graph, you might want to highlight data points in one of the bivariate graphs with a pink solid marker. In a Contour Plot graph, you can increase the width or transparency of the contour lines.

Example of Customizing a Contour Plot

1. Select **Help > Sample Data Library** and open Little Pond.jmp.
2. In the Table panel, click the green triangle next to the Contour Plot script.
3. Click the red triangle next to Contour Plot for Z and deselect **Fill Areas**.

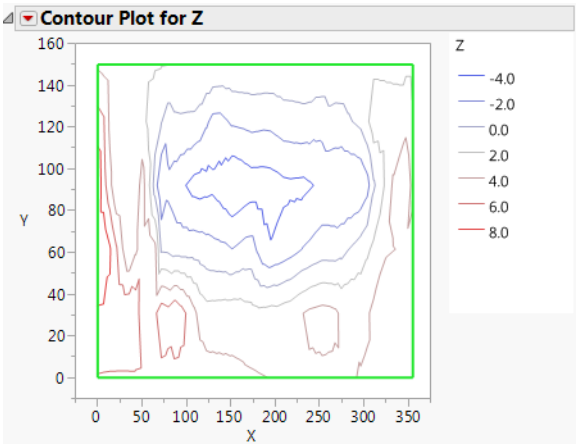
Figure 9.33 Contour Plot before Customization



4. Right-click the plot and select **Customize**.

   The Customize Graph window appears.
5. Click **Shape**.
6. Change the **Line Width** to 2.0.
7. Click the color next to **Line Color** and select a shade of green.
8. Click **OK**.

Figure 9.34 Customized Contour Plot



The graphical elements that you can customize differ on each graph. For example, in a Control Chart graph, there are three line elements (Lower Limit, Upper Limit, and Connect Line). In other graphs, the line element might be named Line or Custom.

To customize graphical elements in the current graph, follow these steps:

1. Right-click the graph and select **Customize**.
2. Select the element that you want to change, and then modify the properties.

   Your changes are immediately shown on the graph.
3. Click **OK** to save your changes to the current graph.

Note: The Customize settings do not override those shown in the legend. For example, if you set the fill area in a bar chart to red, and the legend is blue, the legend will remain blue.

Each element can include properties for line color, marker style and size, text style, and so on. Here are some common properties:

Line Color Changes the color of the line. Click to select any color in the window. Right-click to choose from more colors.

Line Style Changes the style of the line. Click to select one of five different styles.

Line Width Changes the width of the line. Click in the box and enter the desired line width in points.

Marker Changes the marker shape. For more information about markers, see "Use Markers in Graphs".

Marker Size Changes the size of the marker.

Arrow Adds an arrowhead to a line at either None, Start, End, or Both ends of the line.

Line of Fit Changes the color, width, or style of the line.

Text Color Changes the color of the text.

Font Changes the font, style, and point size of the text.

Text Style Changes the text alignment to centered, left-aligned, or right-justified. The **Fill** option applies the selected color to the background.

Fill Color Changes the color of objects such as box plots and Fit Confidence regions.

Fill Pattern Changes the pattern of filled objects.

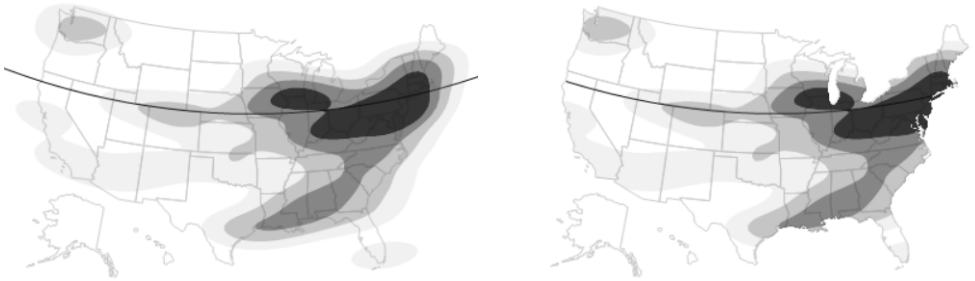
Transparency Changes the marker or label transparency. Enter the level of transparency to draw markers (points) on the graph. The degrees of opacity ranges from 0 (clear) to 1 (opaque). For more information about changing the transparency of markers, see "Specify the Transparency of Markers".

Clip Shape Crops the object (such as contours) around the boundary according to the selected Clip Shape. The display segs that make up the object then conform to the boundaries of the map. Select a sub-boundary to display the objects only over a portion of the boundary. For example, you might want to display the objects only over New York.

About Clip Shape Options

Graphical elements (display segs such as reference, grid lines, and contours) can be clipped to conform to the boundaries of a geographical map. The Customize Graph window provides Clip Shape options, where you can select a boundary around which to clip the shape. Figure 9.35 shows reference lines clipped around a map of the United States.

Figure 9.35 Unclipped (Left) and Clipped (Right) Reference Lines



Tips:

- See *Essential Graphing* for an example of creating the maps shown in Figure 9.35.
- The Clip Shape option in the Customize window was also set to clip the contours to the boundaries of the map. See "Clip Shape".
- Scripting provides additional options such as specifying the clipping path in a matrix or string. See the *Scripting Guide*.

Create Scripts for Graphical Elements

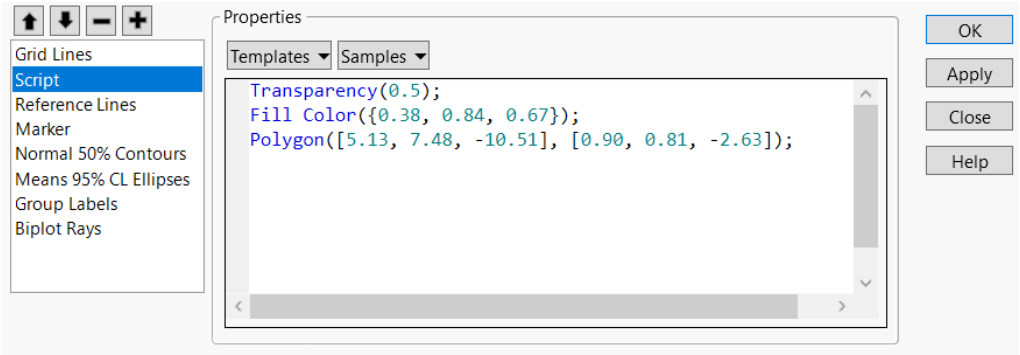
In addition to customizing graphical elements, you can write JSL scripts that add elements. These scripts run when you display the graph. You can write the scripts from scratch, or you can select from the following lists of commands and scripts, accessed by clicking on the **Add a new script button** ➕:

- Use the Templates list to insert a single JSL command. For example, the Polygon option inserts the Polygon command. The text enclosed in underscores are placeholders for point values, which you replace with your own values.

  `Polygon([_x0_, _x1_, ...], [_y0_, _y1_, ...]);`

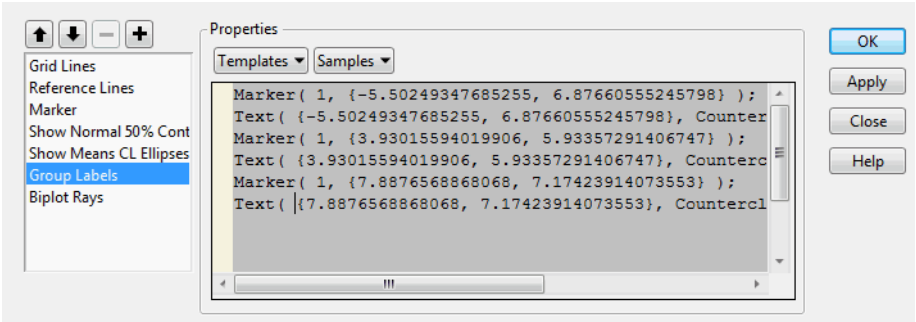
- Use the Samples list to insert a script that creates elements such as bubble plots and sine waves. In this list, the Polygon sample script shows examples of the Transparency, Fill Color, and Polygon values, which you replace with your own values.

Figure 9.36 The Polygon Sample Script



In some graphs, you can view the JSL that creates graphical elements. Figure 9.37 shows the Group Label script for a Discriminant Analysis graph. The script defines markers and text for group labels. The commands included in these built-in scripts cannot be modified or deleted. You can insert commands from the Templates or Samples list, but you cannot click in the window and type new commands.

Figure 9.37 Example of JSL That Creates Group Labels



For information about JSL, see the *Scripting Guide* and the *JSL Syntax Reference*.

To create a graphics script, follow these steps:

1. Right-click the graph and select **Customize**.
2. Click the **Add a new script** button (+) to create a new script.

   The default name, Script, is highlighted.
3. With the default name highlighted, type a more descriptive name, and then press Enter. (If you already moved the cursor and the name is no longer highlighted, double-click Script, and then enter the new name.)
4. Do one or more of the following:
   - Enter JSL in the Properties window.

- Select one or more JSL commands from the Templates list, and modify the placeholder text. For example, change the Pen Color option from “blue” to “red.”
- Select one or more sample scripts from the Samples list and modify, if necessary.

5. (Optional) Click **Apply** to update the graph with your changes without closing the window. Unlike other property changes, script changes to not take effect until you click **Apply** or **OK**.
6. Click **OK** to save your changes.

   One of the following occurs:

   - The element that you created appears on the graph.
   - An error message appears if the script contains an error. Select **View > Log** to read about the error, and then correct the script.

To delete a script that you created, select the script and then select the **Delete selected object** button (−).

Change the Drawing Order of Graphical Elements

The graphical elements are drawn in the order in which they are listed. The first element on the list is drawn first, so it appears *behind* all other graphical elements. If one element hides another, you can rearrange the order of the elements.

To rearrange graphical elements

1. Right-click the graph and select **Customize**.
2. Select the element that you want to move.
3. Click the **Move up in drawing order** button (⬆) or **Move down in drawing order** (⬇) button one or more times until the elements are in the order in which you want them drawn.
4. Click **OK**.

Save Your Customizations

The graph customizations apply to the current graph and are also used when you redo an analysis. To re-create your graph at a later time with its customizations, select **Save Script** from the red triangle menu, and then select one of the Save options. For example, you can save the script to the data table, which applies your customized properties each time you run the script. See “Save a JMP Analysis as a Script”.

Copy Your Customizations

You can copy your customizations from one graph to another. All objects that you created or modified, such as colored text or lines, are pasted to the other graph.

To copy lines from one graph to another, see "Copy and Paste Graph Contents".

To copy other objects

1. Right-click in the graph with custom elements and select **Edit > Copy Customizations**.
2. Right-click in the destination graph and select **Edit > Paste Customizations**.

   The objects appear on the current graph and are added to the list of customized elements.

Notes:

- The customizations are pasted into all similar graphs when you press Ctrl, click, and then paste.
- The copy customizations feature copies only the elements that you added or modified. It does not copy the other contents of the graph.

Add Images to a JMP Graph or Report

To enhance your analyses with logos and other types of graphics, paste or drag and drop graphics into the graph or report. .bmp, .emf, .jpeg, .png, and .rtf graphic formats are supported.

- "Paste a Background Image into a JMP Graph"
- "Paste an Image at the End of a JMP Report"
- "Drag and Drop an Image into a JMP Graph"
- "Add Geographical Images and Boundaries to JMP Graphs"
- "Extract Data from an Image"

Paste a Background Image into a JMP Graph

1. Open the graphic file in a graphics program and select the graphic. Copy it to the computer's clipboard.
2. Right-click inside a graph.
3. Select **Edit > Paste Background Image**. The graphic is inserted at the point in the graph that you right-clicked.

Paste an Image at the End of a JMP Report

You can also add the graphic to the end of a report window. Copy the graphic from a graphics program onto your computer's clipboard, display the report window, and select **Edit > Paste**.

Drag and Drop an Image into a JMP Graph

To add an image that you created, generate your report and then drag and drop the image from your computer's file system onto the graph in the report. After you add the image, JMP provides several options such as resizing, formatting, and rotating an image. Right-click the image and select **Image** to see these options.

The following example shows a bivariate plot of wind speeds in the Chicago area. The plot on the left includes arrows to illustrate the wind direction and speed. A map image was dropped onto the plot and resized to line up the markers with the stations that provided the wind data (each dot representing a station).

Figure 9.38 Example of a Custom Map Image

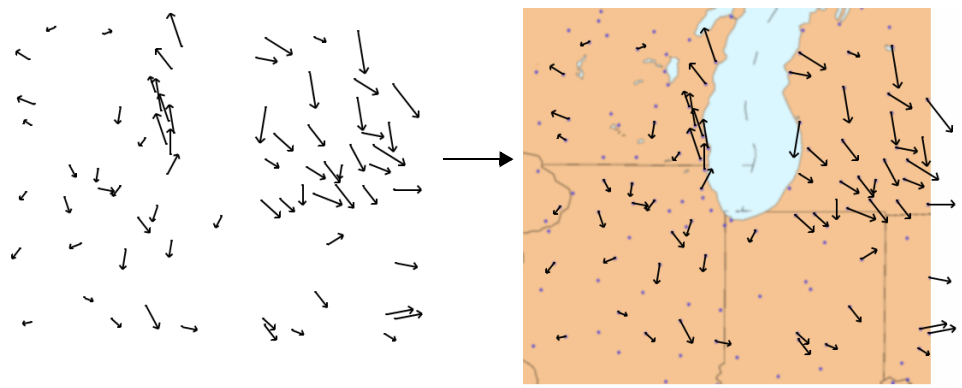


Image Processing

JMP includes a variety of image processing options. Image processing includes filters that can sharpen, blur, resize, and enhance. JMP also allows flipping, rotating, and locking images. This section describes the processing options.

Notes:

- Repeated changes to images can degrade the image quality. If you are not happy with the results of sizing, scaling, or applying filters to the image, remove the image and start again.
- All JMP image filters are supported at the operating system level; images might be processed differently on Windows and macOS.

Lock

Locks the image in place so that it cannot be moved.

Size/Scale

- **Fill Graph**: Resizes the image proportionately to fit the graph.
- **Specify Size**: Resizes the image according to the values that you enter. (The units for these values are the same as your graph axes.)
- **Crop**: Crops the image according to the values that you enter. (The units for these values are the same as your graph axes.) For example, the left edge might be positioned at 50. You enter 60 next to Left, and the portion of the image between 50 and 60 is removed from the image.

Flip

- **Flip vertical**: Flips the image top to bottom.
- **Flip horizontal**: Flips the image left to right.
- **Flip both**: Flips the image both horizontally and vertically.

Rotate

Rotates the image the specified number of degrees. Enter a negative value to rotate the image counterclockwise.

Transparency

Changes the image transparency level. Enter a value between 0 and 1 (where 1 is completely opaque).

Filter

Provides filters found in many graphic editing programs to change the appearance of the image. Select a filter repeatedly to increase its effects on the image.

- **Contrast**: Optimizes the light and dark colors. Larger values lighten the image.
- **Despeckle**: Removes pixels that do not blend with surrounding pixels. For example, a black pixel surrounded by white pixels is converted to a white pixel.
- **Edge**: Darkens everything but the outlines of objects.
- **Enhance**: Reduces the contrast between pixels in a noisy image.
- **Gamma**: Balances the brightness of an image and the red, green, and blue (RGB) ratios. Larger values create a lighter image.

- **Gaussian Blur**: Blurs pixels by the specified radius. Larger radii create a smoother image. (In JSL, you can also specify the sigma value. Larger sigma values create a smoother image.)
- **Median**: Replaces each pixel color value with the median value of the surrounding pixels.
- **Negate**: Converts each pixel to its complementary color (such as pink to green and white to black).
- **Normalize**: Removes a percentage of the top and bottom color values. The color values are then stretched to fill the remaining image. This process increases the intensity of the colors.
- **Reduce Noise**: Finds the minimum and maximum color values and replaces them with values more consistent with the surrounding pixels. Larger values create a smoother image.
- **Sharpen**: Makes the edges of pixels more distinct.

Remove

Removes the image from the report.

Add Geographical Images and Boundaries to JMP Graphs

Adding map images and political boundaries to graphs provides visual context to geospatial data. For example, you can add a map to the graph that displays an image of the U.S. Another option is displaying the boundaries for each state (when data includes the latitudes and longitudes for the U.S.).

The data should have latitudinal and longitudinal coordinates. Otherwise, the map has no meaning in the context of the data. The x and y axes also have range requirements based on the type of map. These requirements are described in the following sections.

Two tools are especially helpful when you are viewing a map:

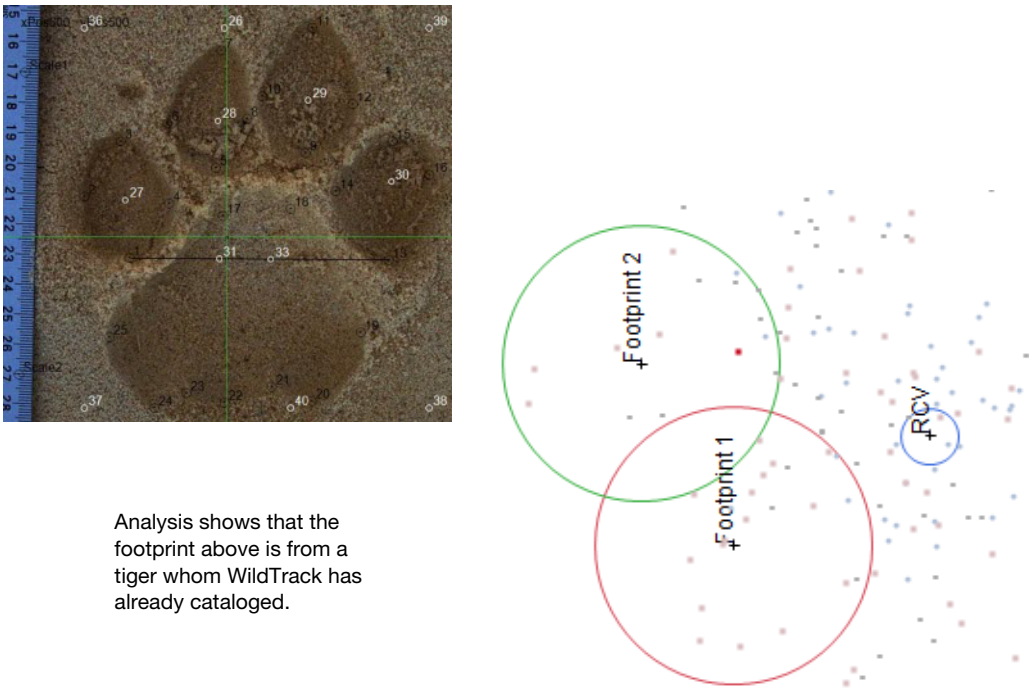
- The grabber tool ( ) lets you scroll horizontally and vertically through a map.
- The magnifier tool ( ) lets you zoom in and out.

For more information about adding maps, see *Essential Graphing*.

Extract Data from an Image

JMP provides the ability to extract information from images into a data table and then analyze that information. Researchers at WildTrack.org analyze digital footprint photos in JMP to track endangered species. They drag and drop a footprint image into a JMP report and draw data points to capture the size and shape of the print. A JMP Scripting Language (JSL) script extracts those measurements into a data table. At that point, the researchers can analyze the data and determine whether the footprint is from a new animal. This method helps them track populations of endangered species in specific regions of the world.

Figure 9.39 Example of Extracting and Analyzing Data



Analysis shows that the footprint above is from a tiger whom WildTrack has already cataloged.

The details for implementing this feature are beyond the scope of *Using JMP*, because the needs of the user vary widely. For more information about writing JSL scripts, see the *Scripting Guide* and the *JSL Syntax Reference*.

Add Annotations and Shapes to JMP Reports and Graphs

You can add text notes, lines, polygons, ovals, and rectangles to a report or graph using graphics tools found in the toolbar.

Add Annotations to JMP Reports or Graphs

You can add editable text notes to a JMP report or graph using the annotate tool . To add an annotation:

1. Select the annotate tool from the Tools menu or toolbar.
2. In the report or graph, click where you want to add the annotation. Or, click and drag to size the annotation note. A white editable text box appears.
3. Enter text.
4. Click outside the annotation.
5. (Optional) To add a tag line, right-click the annotation and select **Tag Line**.

   You can also edit any of the following options:

   Background Color Select the background color for the annotation.

   Text Color Select the text color. The color of the text also defines the color of the annotation outline.

   Font Change the font type, style, and size. To change the default font, see "Fonts Preferences".

   Tag Line Draws a line between the annotation and the point or text it is anchored to.

   Filled Removes the background color from the annotation so that it looks transparent. A transparent note is handy for putting titles and footnotes on a graph.

   Editable Makes the annotation editable by double-clicking on it.

   Fixed Size Avoids resizing the annotation box when you add text to the box.

   Reanchor (Available only when the annotation is moved.) Annotations are anchored to a location within the report, but can be moved around independently of the anchor location. The behavior of an annotation depends on the anchor position. For example, annotations anchored inside a graph move with the axes, and annotations anchored inside an outline show or hide when the outline is opened or closed. To reanchor an annotation, turn off the tag line, move the annotation to a new location, and right-click to reanchor it.

Delete Deletes the entire annotation.

Note: When adding multiple annotations, press Shift when selecting the annotation tool for the first time. This causes subsequent clicks to add an annotation, and you do not have to select the annotate tool from the toolbar before the addition of each annotation.

Once you have added an annotation, you can do the following:

Table 9.3 Using Annotations

Action	Instructions
Add to or edit an annotation	Click inside the text box.
Move an annotation	Click inside the annotation box and drag it. When an annotation is moved, it becomes selected, as indicated by a double blue line with handles around the perimeter.
Resize an annotation	Place the cursor on the handle of a selected note (showing in the middle of the edges and in the corners). The cursor appears as a single crossed arrow; drag to resize the annotation.
Delete an annotation	Highlight the annotation by clicking the handle of a note and then press Delete or Backspace.
Move an annotation's tag line	1. Right-click the annotation and deselect **Tag Line**. 2. Drag the annotation to where you want the tag line to start (the attachment point). 3. Right-click the annotation and select **Tag Line**. The tag line now starts at the point where you moved the annotation. 4. Move the annotation where you want it to appear.
Move the tag line with the annotation	Press Ctrl and drag the annotation to move the tag line endpoint with the annotation.

Add Shapes to JMP Reports

You can add editable lines, polygons, and simple shapes (ovals or rectangles) to a JMP report using the drawing tools . The following sections describe how each of these tools can be used.

Note: Each graphics tool remembers the most recent options chosen. This is useful if you need many annotations or other graphics with the same characteristics. For example, suppose you want many thick green lines with an arrow on one end. Create a line the way you want it, set the options, and subsequent lines appear with those options in effect. The options persist until you change them.

Add a Line

Add a line annotation to a report window:

1. Click the line tool in the tool palette.
2. Click and drag where you want to insert the line. The line appears selected, showing handles on both ends.
3. Click and drag the line to move it.
4. Click a handle and drag to rotate the line.
5. Right-click a line for options to customize the appearance of the line:

Point to and Point from Places arrows on either end (or both ends) of the line.

Thick Changes the width of a line to thick. Lines are thin by default.

Dashed Changes the line to dashed. Lines are solid by default.

Color Displays the JMP color palette to change the color of the shape.

Reanchor Line annotations are anchored to a location within the report but can be moved around independently of the anchor location. The behavior depends on the anchor position. For example, line annotations anchored inside a graph move with the axes, and line annotations anchored inside an outline show or hide when the outline is opened or closed. To reanchor a line annotation, move the line to a new location and right-click to reanchor it. Available only when the annotation is located outside the graph.

Delete Removes the shape from the report surface. You can also remove the shape by selecting it and then pressing Delete or Backspace.

Add a Polygon or Bezier Curve

To add a polygon (or Bezier curve) annotation to a report window:

1. Click the polygon tool in the tool palette.
2. Click to create the beginning point for the first side of a polygon.
3. Click again at the location where you want to complete the first side and begin an adjacent side. A square selection box with handles appears around the polygon area.

4. Click a third time to complete the second side.
5. Continue this process until the polygon is the way you want it. Each time a side is complete, the selection box adjusts to encompass the polygon sides.
6. Double-click to release the polygon tool.

Once you have added a polygon, you can perform the following actions:

Table 9.4 Working with Polygons

Action	Instructions
Select or deselect the polygon	Click the edge of a completed polygon.
Resize the polygon	Select it and drag one of the selection box's handles.
Move the polygon	Click between the box's handles and drag the selection box.
Change the number of sides of the polygon	Click and drag the sides to form the new shape.

Right-click a polygon for options to customize its appearance:

Filled Fills or empties the area of the shape.

Raised Displays thick shaded lines around the shape. If the shape is also filled, the lower edge of the figure appears raised, giving it a three-dimensional look.

Smooth Smooths the vertices of a polygon to produce a Bezier curve. The smoothed figure is reshaped and resized the same way as the polygon, and can be filled and raised.

Closed Opens or closes the last segment of a polygon.

Color Displays the JMP color palette to change the color of the shape's sides, and its fill color when the **Filled** option is in effect.

Reanchor Polygon annotations are anchored to a location within the report, but can be moved around independently of the anchor location. The behavior depends on the anchor position. For example, polygon annotations anchored inside a graph move with the axes, and polygon annotations anchored inside an outline show or hide when the outline is opened or closed. To reanchor a polygon annotation, move the polygon to a new location and right-click to reanchor it.

Delete Removes the shape from the report. You can also remove the shape by selecting it and then pressing Delete or Backspace.

Add an Oval or Rectangle

To add an oval or rectangle annotation to a report window:

1. Click the simple shape tool in the tool palette.
2. Click and drag where you want to insert the shape. An oval appears with a selection box around it.
3. (Optional) To turn the oval into a rectangle, right-click the oval and select **Shape > Rectangle**.
4. Double-click to release the simple shape tool.

Once you have added the shape, you can do the following:

- Select the shape and drag one of the selection box handles to reshape or resize.
- Click and drag an edge of the selection box (located between the handles) to move it.
- Select the shape, and then right-click it for options to customize its appearance:

  Filled Fills or empties the area of the shape.

  Raised Displays thick shaded lines around the shape. If the shape is also filled, the lower edge of the figure appears raised, giving it a three-dimensional look.

  Shape Displays a submenu whose options transform the shape into either an oval or a rectangle when selected.

  Color Displays the JMP color palette to change the color of the shape's sides and its fill color when the Filled option is in effect.

  Reanchor Shape annotations are anchored to a location within the report, but can be moved around independently of the anchor location. The behavior depends on the anchor position. For example, shape annotations anchored inside a graph move with the axes, and shape annotations anchored inside an outline show or hide when the outline is opened or closed. To reanchor a shape annotation, move the shape to a new location and right-click to reanchor it.

  Delete Removes the shape from the report. You can also remove the shape by selecting it and then pressing Delete or Backspace.

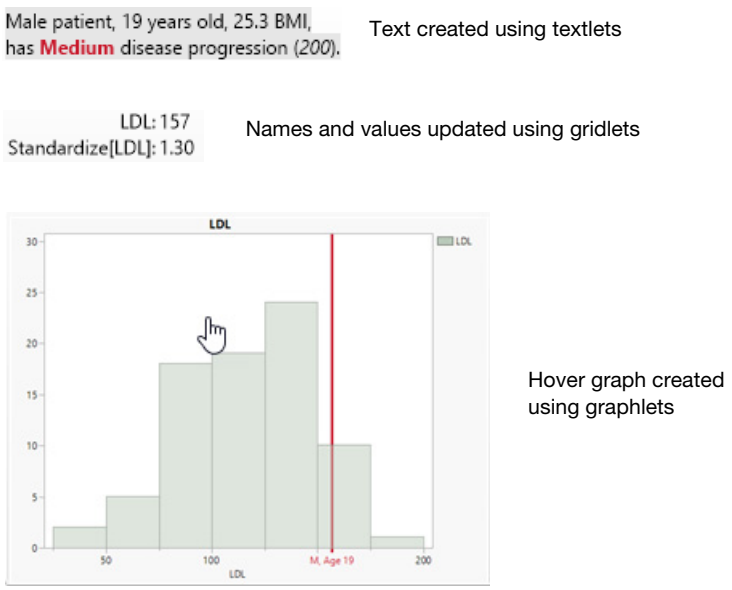
Customize Hover Labels in Graphs

When you hover over a visual element in a graph, usually you see a basic description of the data point in a hover label. Customize what appears in the hover label in any of these ways:

Table 9.5

Add interactive graphs or images using *graphlets*.	See "Add Graphs or Images to Hover Labels".
Add formatted text using *textlets*.	See "Add Rich Text to Hover Labels in Graphs"
Add, remove, or update names and values using *gridlets*.	See "Update Names and Values in Hover Labels".
Learn more about using the Hover Label Editor.	See "Edit Hover Labels in Graphs".
Add images from data tables.	See "Add Images from Data Tables to Hover Labels".
Pin hover labels on graphs.	See "Pin Hover Labels in Graphs".
Format hover labels and access additional options.	See "Right-Click Options for Hover Labels".

Figure 9.40 Examples of Graphs, Text, Names, and Values in Hover Labels



Add Graphs or Images to Hover Labels

You can add a thumbnail of a graph or an image to a hover label using graphlets.

- Preset graphlets: Preset graphlets are predefined and editable. They are easy to add, simply right-click on a graph and select **Hover Label** > *graph type*, or you can add them from the Hover Label Editor.

  Note: Graphs based on synthetic data (such as data used in simulations) that are not connected to a data table do not support preset graphs.

- Paste graphlet: Create a graph in Graph Builder. Copy the script by saving it to your clipboard, and then paste it as a graphlet into a baseline graph.

Once you add the hover graph, put your cursor over an element in the baseline graph. A thumbnail preview of the hover graph appears for that element. For example, for aggregate data, you can see a hover graph of the underlying data distribution.

Some JMP platforms have built-in hover label graphs, such as Principal Components, Process Screening, Model Driven Multivariate Control Charts, and Functional Data Explorer. You do not need to do anything to add these graphs to your visualizations. When you hover over them, the graph automatically appears in the hover labels.

You can also customize hover label graphs using JSL. For details, see the *Scripting Guide*.

Hover Graph Actions

- Click a hover graph to launch it in a new window, or click an image to open a related link.
- Press Ctrl and click on a hover label graph to replace the baseline graph with the hover label graph.
- Press Alt and click on a hover label graph to replace the contents of the last launched window, instead of opening a new one.

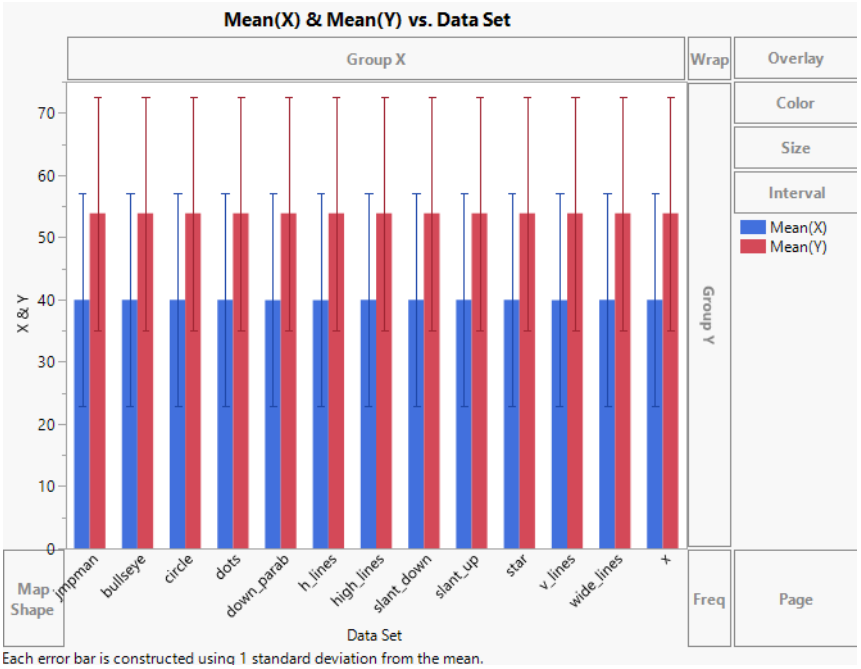
Example of Adding a Preset Graph to a Hover Label

In this example, you create a graph, add preset graphs to hover labels, and then customize the hover label graphs.

Create the Original Graph

1. Select **Help** > **Sample Data Library** and open JMP Man Dozen.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select X and Y and drag them to the **Y** zone.
4. Select Data Set and drag it to the **X** zone.
5. Click the Bar element .
6. For the Error Interval, select **Standard Deviation**.

Figure 9.41 Original Graph for JMP Man Dozen.jmp



You can see that the mean and standard deviation for these observations are the same, so this graph does not tell you much. Use preset graphs to get more insight into the data.

Add Preset Graphs

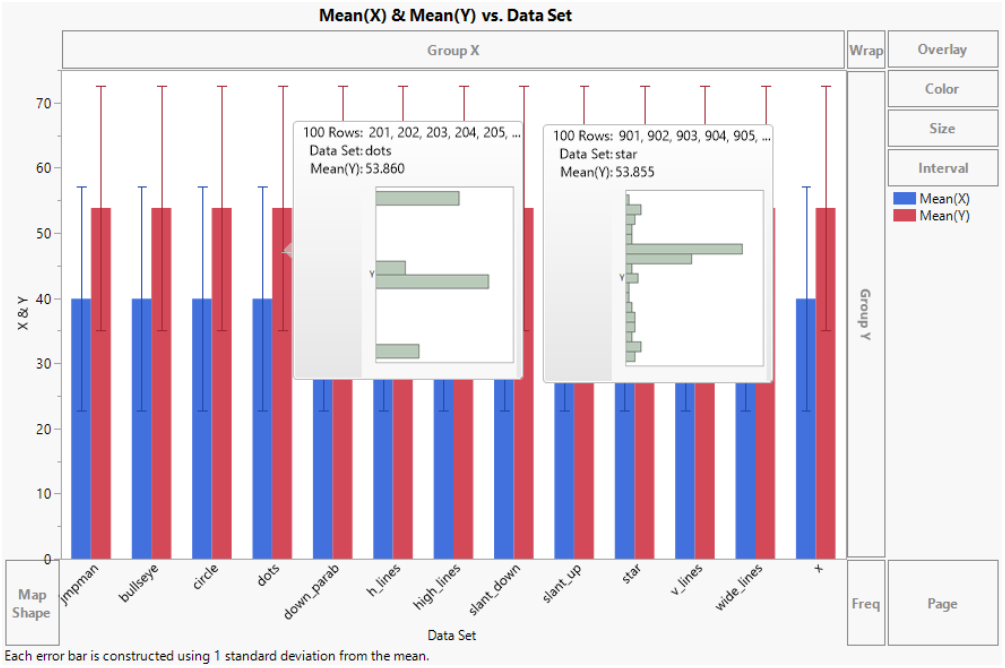
1. In the graph, right-click and select **Hover Label** > **Histogram**.

   This adds preset histograms to the hover labels.

2. Point to some of the bars. In this example, point to the red bars for dots and star.

Tip: To make the hover label remain so that you can compare it with other hover labels, click the Pin icon in the top right corner. See "Pin Hover Labels in Graphs".

Figure 9.42 Example of Preset Histograms



You can see that the distribution for the measurements is clearly different. But you want to see another type of graph, a scatterplot.

3. In the graph, right-click and select **Hover Label** > **Points**.

   This changes the preset graphs from histograms to scatterplots.

4. Hover over the blue bar for down_parab and the red bar for down_parab. Notice that the Y and X variables on the axes are reversed.

   You want to use the same X and Y role assignments for all graphs (assigning the X variable to the X axis, and the Y variable to the Y axis). You also want to change the smoother to a line of fit, and add a caption box with summary statistics to the scatterplot.

   You can customize a graph in Graph Builder and then copy the graph as a template and apply it to all of the hover labels in this graph.

Customize Graphs

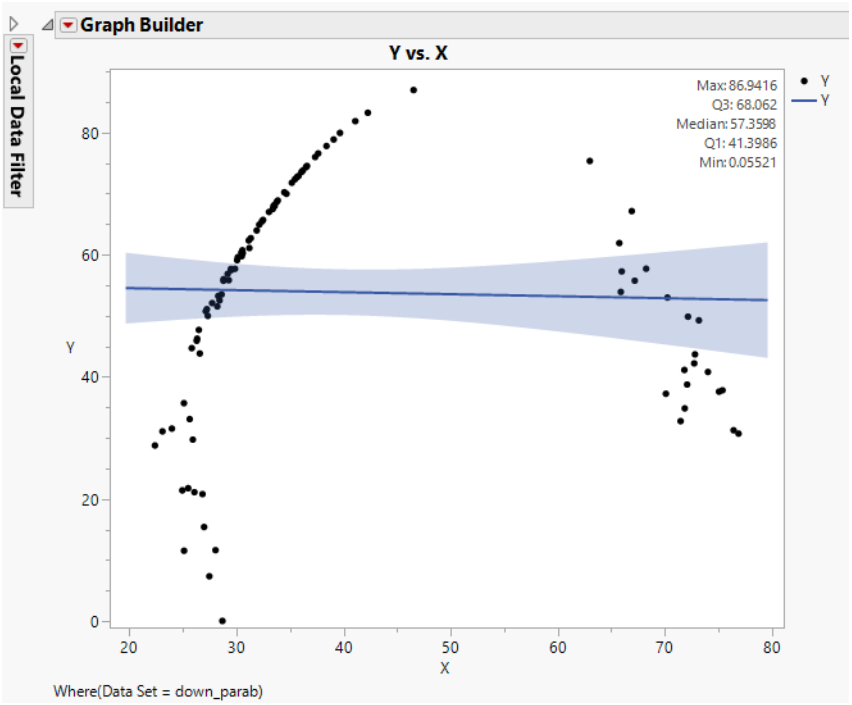
1. Hover over the blue bar for down_parab and click the graph to open it in a new window.

   The graph opens in Graph Builder.

   Note: Notice that when you launch a preset graph, you see a Local Data Filter at left. You can use the Local Data Filter to expand and modify the graph. For details, see "Local Data Filters in Reports".

2. Click the Graph Builder red triangle and select **Show Control Panel**.
3. Click the Line of Fit element .
4. Click the Caption Box icon .
5. Under Caption Box, change the Summary Statistic to **5 Number Summary**.
6. Click **Done**.

Figure 9.43 Customized Graph for down_parab



7. Click the Graph Builder red triangle menu and select **Save Script** > **To Clipboard**.
8. Close this Graph Builder window.
9. In your original graph, right-click and select **Hover Label** > **Paste Graphlet**.

   This applies the customized graph as a template to all hover label graphs. Hover over a few bars to see the changes.

Tip: To better compare original graphs and graphs launched from hover labels, use projects. See Chapter 10, "Save and Share Your Data".

Example of Specifying Drill-Down Column Hierarchies in Hover Labels

In preset graphs that use categorical columns, you can specify which columns to filter by.

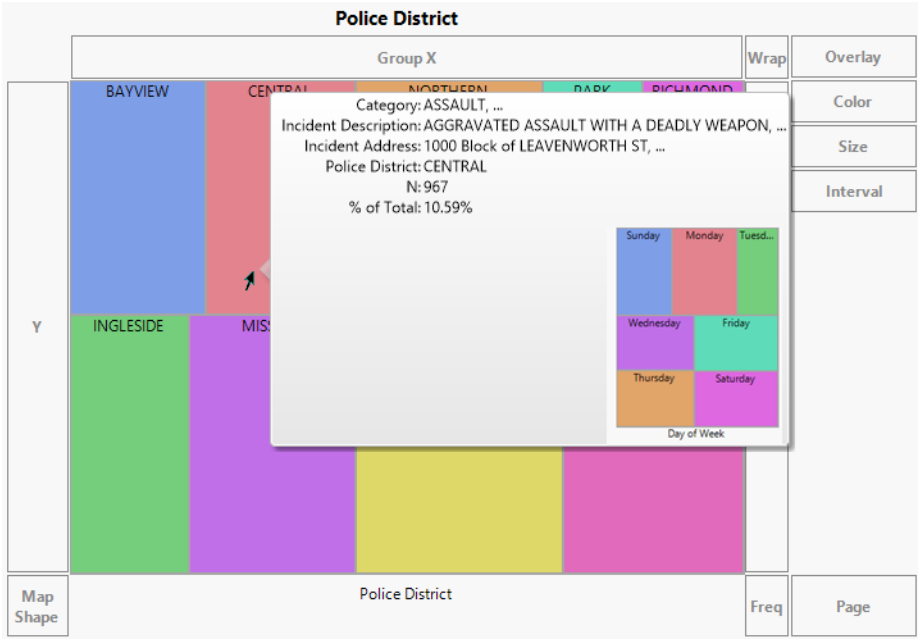
Create the Original Graph

1. Select **Help** > **Sample Data Library** and open San Francisco Crime.jmp.

   These data show crime statistics for police districts in San Francisco.
2. Select **Graph** > **Graph Builder**.
3. Select Police District and drag it to the **X** zone.
4. Click the Treemap element .

   You can see a treemap of the different police districts in San Francisco.
5. Right-click on the graph and select **Hover Label** > **Treemap**.
6. Put your cursor over Central to see the hover label graph.

Figure 9.44 Initial Hover Label Graph for Central

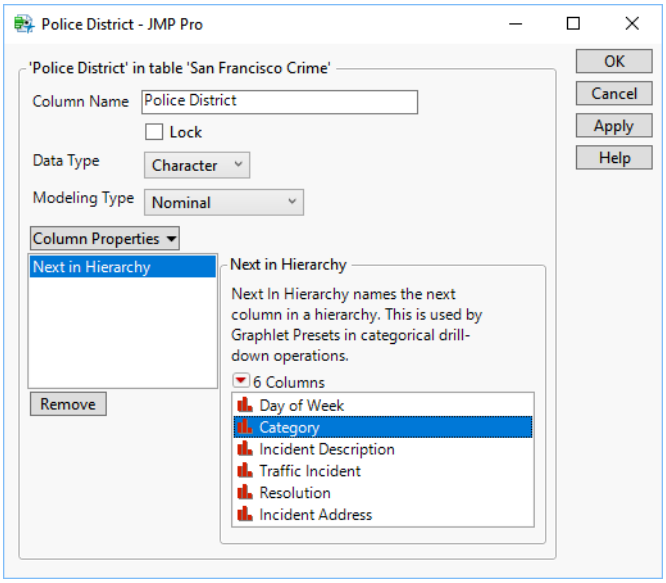


The hover label graph is showing the number of incidents in each district by the day of the week. Instead, in this example, you want to see the number of incidents in each district by category.

Specify the Drill-Down Column Hierarchy

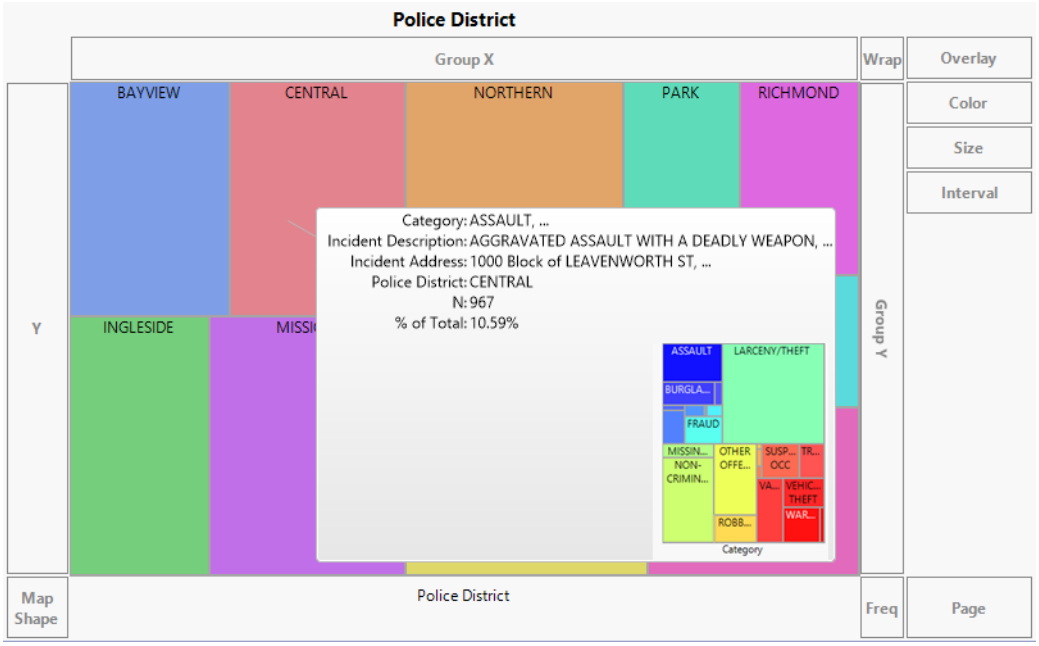
1. In the San Francisco Crime.jmp data table, right-click the Police District column heading and select **Column Properties** > **Next in Hierarchy**.
2. Under Next in Hierarchy, click Category.

Figure 9.45 Category Selected for Next in Hierarchy



3. Click **OK**.
4. Add a second level of the hierarchy that appears if the first hover label graph is launched:
   a. In the San Francisco Crime.jmp data table, right-click the Category column.
   b. Select **Column Properties** > **Next in Hierarchy**.
   c. Under Next in Hierarchy, click Incident Description.
   d. Click **OK**.
5. In Graph Builder, put your cursor over Central again to see the updated hover label graph.

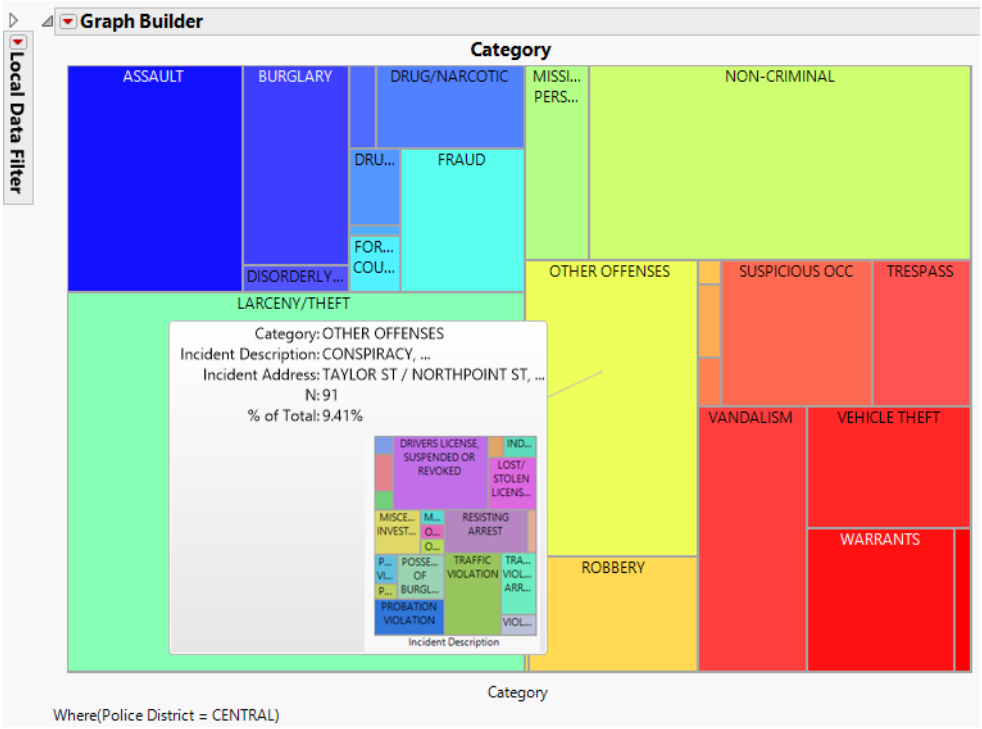
Figure 9.46 Updated Hover Label Graph for Central



The hover label graph is now showing the number of incidents in each district by category.

6. Click the hover label graph to launch it.
7. Put your cursor over Other Offenses.

Figure 9.47 Second Level Hover Label Graph for Other Offenses Category



The second level of the hover label graph is showing the number of incidents in the Other Offenses category by the incident description.

Example of Removing a Filter Column from a Hover Label Graph

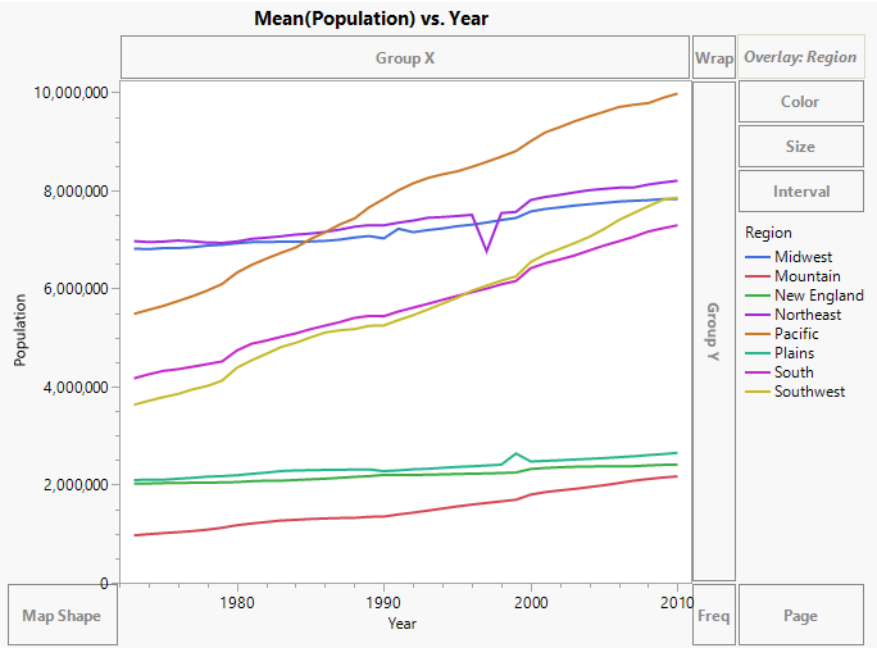
In some cases, the hover label graph might be filtering on a column that you want to remove. For example, the filter column might be too granular to be useful. You can remove it using the Hover Label Editor.

Create the Original Graph

1. Select **Help** > **Sample Data Library** and open CrimeData.jmp.

   These data show crime rates for regions and states in the U.S.
2. Select **Graph** > **Graph Builder**.
3. Select Population and drag it to the **Y** zone.
4. Select Year and drag it to the **X** zone.
5. Select Region and drag it to the **Overlay** zone.
6. Click the Line element .

Figure 9.48 Mean Population Crime Rates by U.S. Region Per Year



This graph shows the mean population in each U.S. region that were impacted by different crime rates over different years. In the hover label graphs, you want to see a more granular view that shows the impacts by U.S state instead of region.

7. Keep this window open.

Copy and Paste a Graph

1. Click the Graph Builder red triangle and select **Redo > Redo Analysis**.

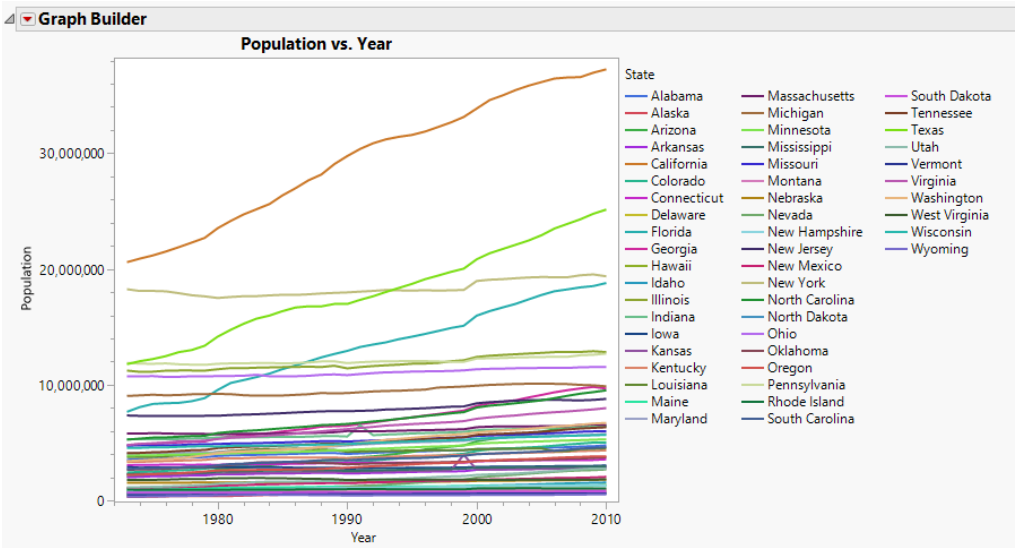
   A copy of the original graph appears.

2. In the copy of the graph, select State and drag it to the **Overlay** zone.

   This replaces Region with State.

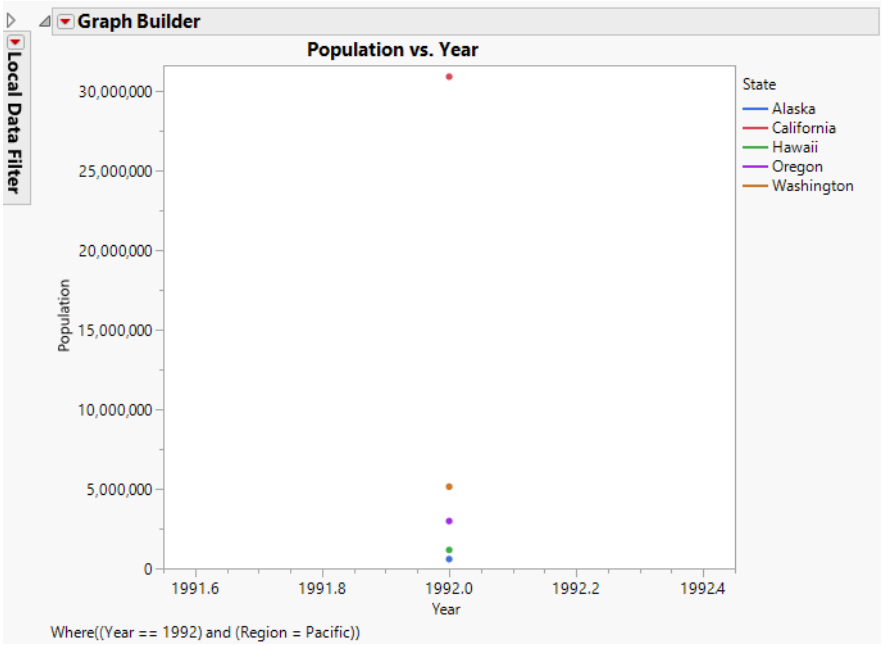
3. Click **Done**.

Figure 9.49 Mean Population Crime Rates by U.S. State Per Year



4. Click the Graph Builder red triangle and select **Save Script > To Clipboard**.
5. Close this graph.
6. In the original graph, right-click and select **Hover Label** > **Paste Graphlet.**
7. Hover over the Pacific line and click to launch the hover label graph.

Figure 9.50 Hover Label Graph Showing Crime Rates by State and Year for the Pacific Region



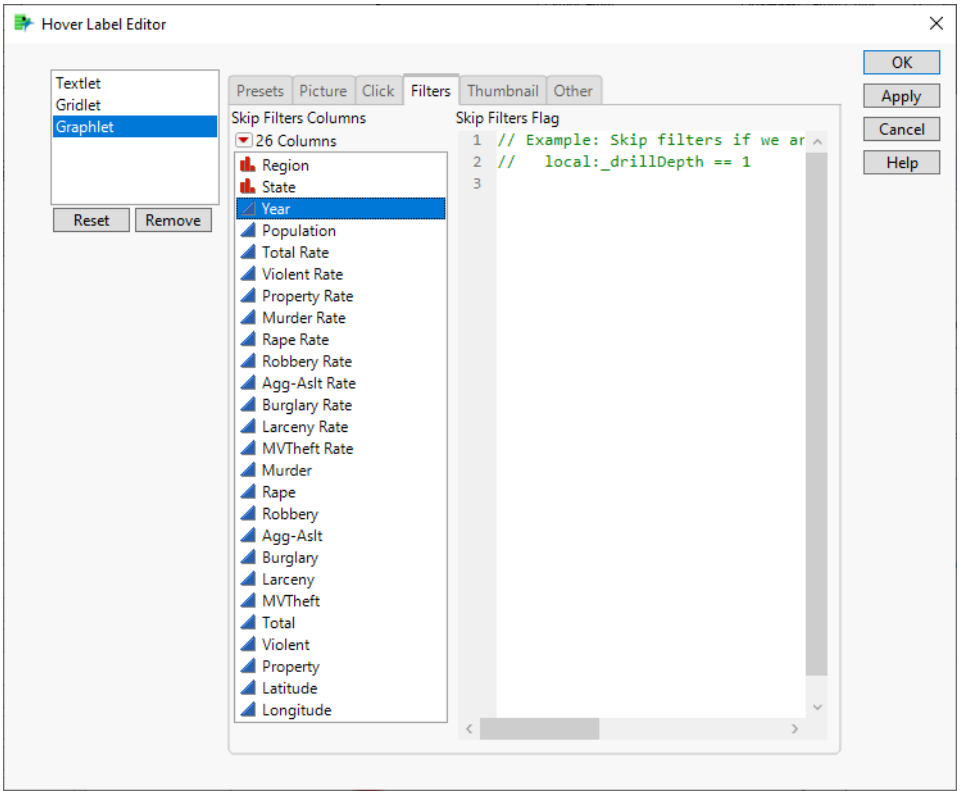
Because we are still filtering by year, this view is too granular to be useful. You want to remove the year filter, so the graph filters only by region.

8. Close the hover label graph for the Pacific region.

Remove a Filter Column

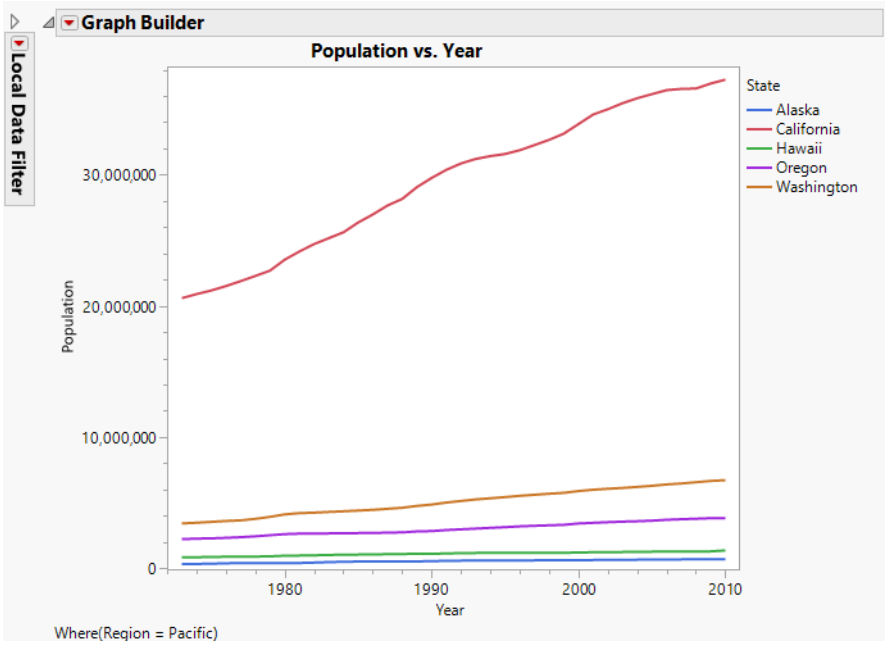
1. Right-click the original graph and select **Hover Label** > **Hover Label Editor**.
2. In the Graphlet window, click the **Filters** tab.
3. Select **Year**, the column you no longer want to filter by.

Figure 9.51 Skip Filters in the Hover Label Editor



4. Click **OK**.
5. Hover over the Pacific line and click to launch the hover label graph.

Figure 9.52 Hover Label Graph Showing Crime Rates by State for the Pacific Region



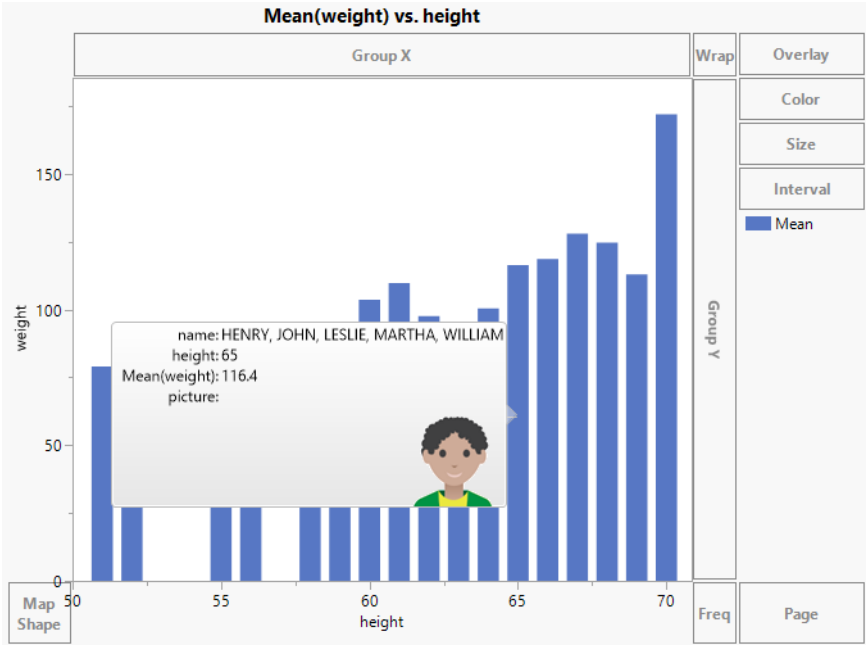
You can see that the hover label graphs are no longer filtering by year, but by region only, creating a more useful visualization.

Example of Displaying Multiple Images in a Hover Label

For labeled columns, hover labels can contain an image from the first labeled image column and up to four values from labeled text columns. To show more labeled images or text observations, use the Label Viewer.

1. Select **Help** > **Sample Data Library** and open Big Class Families.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select weight and drag it to the **Y** zone.
4. Select height and drag it to the **X** zone.
5. Click the Bar element .
6. Hover over the bar for the height of 65 inches and click Pin .

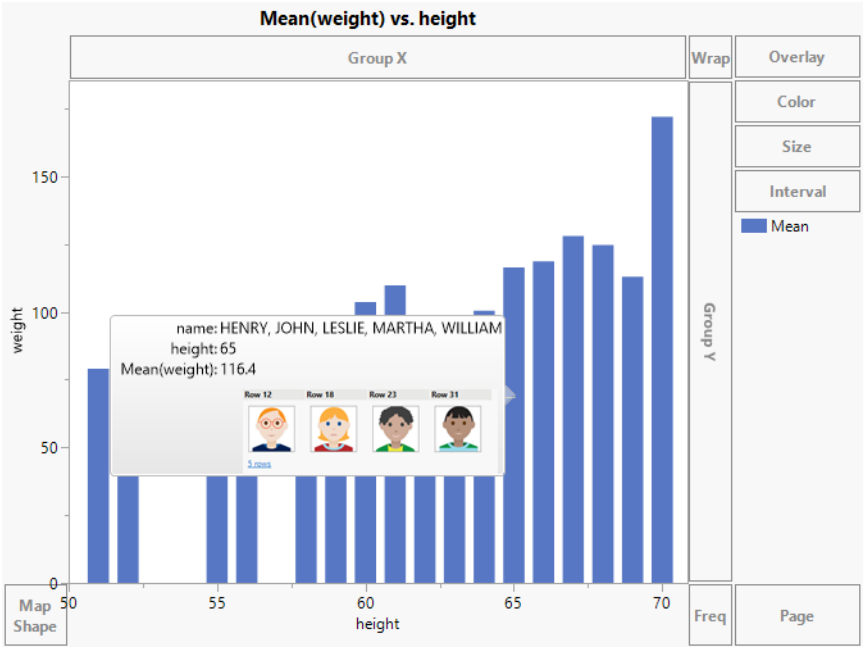
Figure 9.53 Hover Label with Only Initial Image



Although five names are included in this bar, only the first image (Henry) appears in the hover label. You want to see more images in the hover label.

7. Right-click in the graph and select **Hover Label** > **Label Viewer**.

Figure 9.54 Hover Label with Multiple Images



The maximum of four images now appear in the hover label.

8. To see the additional image, in the hover label, click a thumbnail.
9. In the Label Viewer, click the Forward arrow to see the fifth image.

Add Rich Text to Hover Labels in Graphs

A textlet is a hover label extension that adds formatted text to a hover label. You might use a textlet to describe data in an explanatory paragraph, or to call out important information by making it bold or drawing it in a different color.

Example of Adding Text to a Hover Label

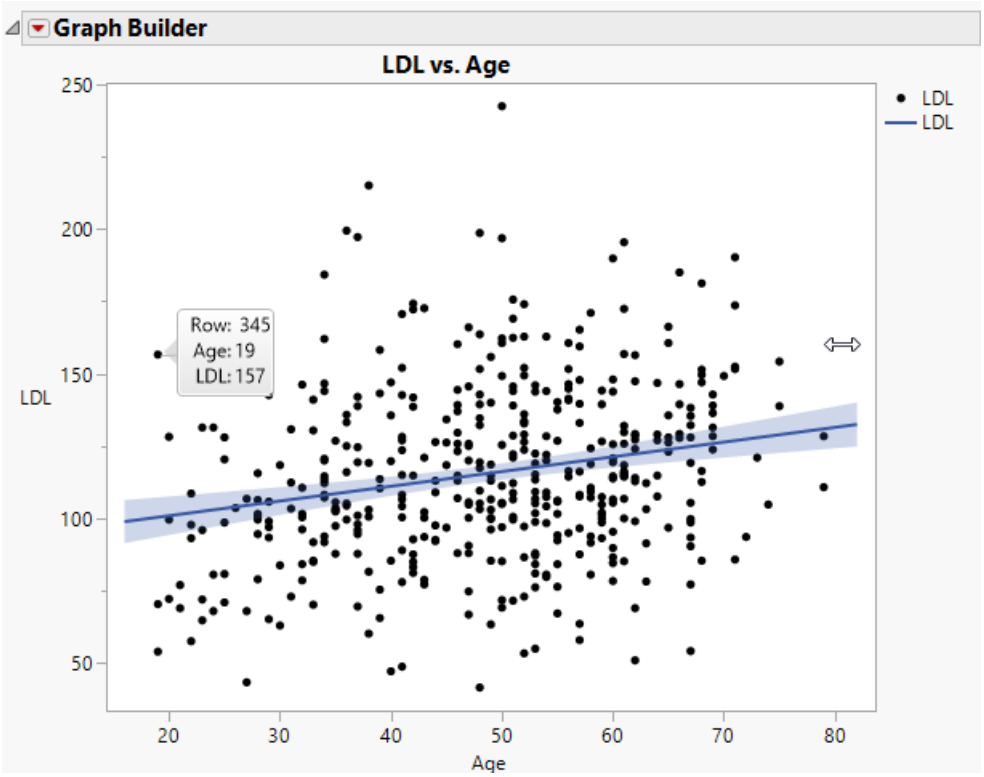
Create the Original Graph

1. Select **Help** > **Sample Data Library** and open Diabetes.jmp.

   These data show measurements for diabetes patients that track the progression of the disease.
2. Select **Graph** > **Graph Builder**.
3. Select LDL and drag it to the **Y** zone.
4. Select Age and drag it to the **X** zone.

5. Click the Line of Fit element .
6. Click **Done**.
7. Hover over the point in row 345.

Figure 9.55 Initial Hover Label for Content for Row 345



For the data point in row 345, you can see the row number, age of the person, and their LDL measurement. Change what appears in the hover labels:

- Add the patient's level of disease progression in a sentence.
- Assign different colors to the levels of the disease progressions, based on severity.

Add Text

1. Put your cursor over the hover label for row 345 and click Pin .

   Pinning the hover label enables you to see your changes as you apply them.
2. Right-click in the graph and click **Hover Label** > **Hover Label Editor**.
3. On the left, click **Textlet**.

4. In the JSL Variables box, define the variables using JSL. Copy and paste the following JSL script:

```
// gets the value for the Y Ordinal column for the underlying visual element
local:YOrdinal = As Column( "Y Ordinal" )[local:_firstRow];
// maps the value to HTML color names that are used to style the text
local:color = Match( local:YOrdinal,
"Low", "Medium Dark Green",
"Medium", "Medium Dark Blue",
"High", "Medium Dark Red");
```

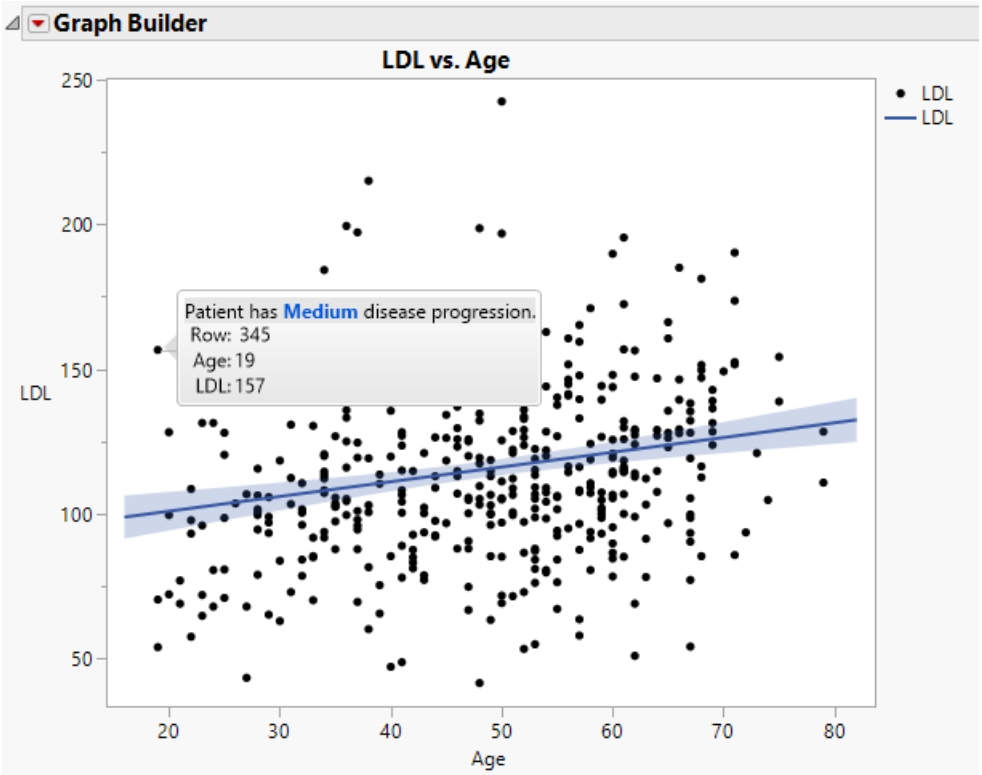
5. Under HTML Markup, write your text using HTML and the JSL variables that you just defined. Copy and paste the following code:

```
Patient has <font color='{local:color}'><b>{local:YOrdinal}</b></font>
   disease progression.
```

Figure 9.56 JSL Variables and HTML Markup for Textlet

6. Click **OK**.

Figure 9.57 Updated Hover Label Containing Rich Text



You can see the new sentence appearing in the hover label indicating the level of disease progression. Since the disease progression for this patient is medium, it is color coded blue.

Update Names and Values in Hover Labels

When you hover over an element in a graph (such as a point, bar, and so on), you see content in the hover label for that element. By default, the hover label contains name and value pairs that are in a grid. The value can come from the data table, or can be an arbitrary calculation or JSL expression. You can customize these names or values using gridlets. For example, here are some things that you can do:

- Remove a name and its associated value
- Add a name and its associated value
- Change a name
- Change the style of a name (for example, bold) or a value (for example, decimal places)

Tip: Another way to customize hover label grid entries is using labels. See "Add Images from Data Tables to Hover Labels".

Example of Updating Names and Values in a Hover Label

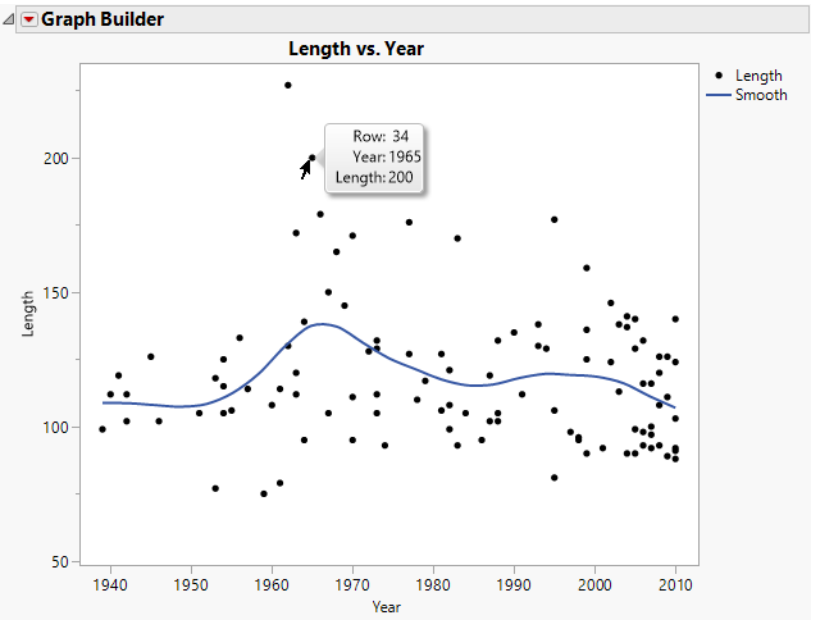
In this example, you create a graph and update content in the hover label using gridlets.

Create the Original Graph

1. Select **Help** > **Sample Data Library** and open Movie Inventory.jmp.

   These data show movie rental data.
2. Select **Graph** > **Graph Builder**.
3. Select Length and drag it to the **Y** zone.
4. Select Year and drag it to the **X** zone.
5. Click **Done**.
6. Hover over the point in row 34 (Figure 9.58).

Figure 9.58 Initial Hover Label Content for Row 34



For the data point in row 34, you can see the row number, the year in which the movie was made, and the length of the movie (in minutes). Change what appears in the hover labels:

- Remove the row number and the length entry.

- Add the movie name and genre.
- Create a new entry that shows the length of the movie in hours and minutes.
- Make the genre name bold.

Update Names and Values

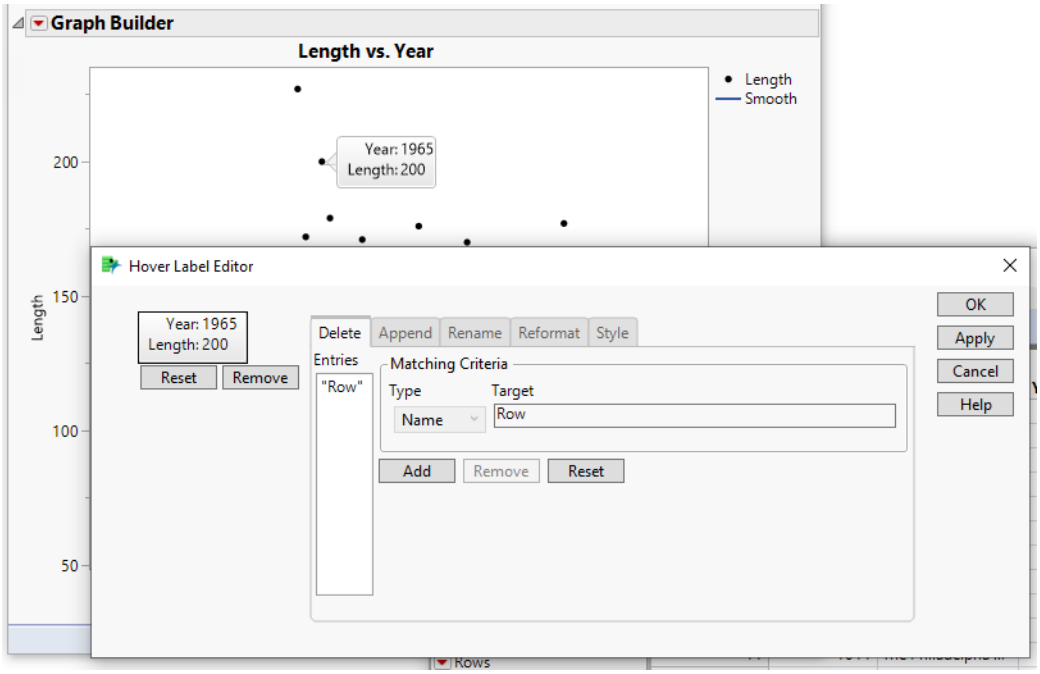
1. Hover over the hover label for row 34 and click Pin .

   Pinning the hover label enables you to see your changes as you apply them.
2. Right-click in the hover label and click **Edit**.
3. At left in the thumbnail, click Row.
4. In the Delete tab, click **Add**.

   "Row" appears under the entries to delete.
5. Click **Apply**.

   In the graph window, you can see that Row is removed from the hover label.

Figure 9.59 Hover Label with Row Removed



6. At left in the thumbnail, click Length (the other entry that you want to remove).
7. In the Delete tab, click **Add**.

   "Length" appears under the entries to delete.

8. Click **Apply**.

   In the graph window, you can see that Length is removed from the hover label, leaving only Year.

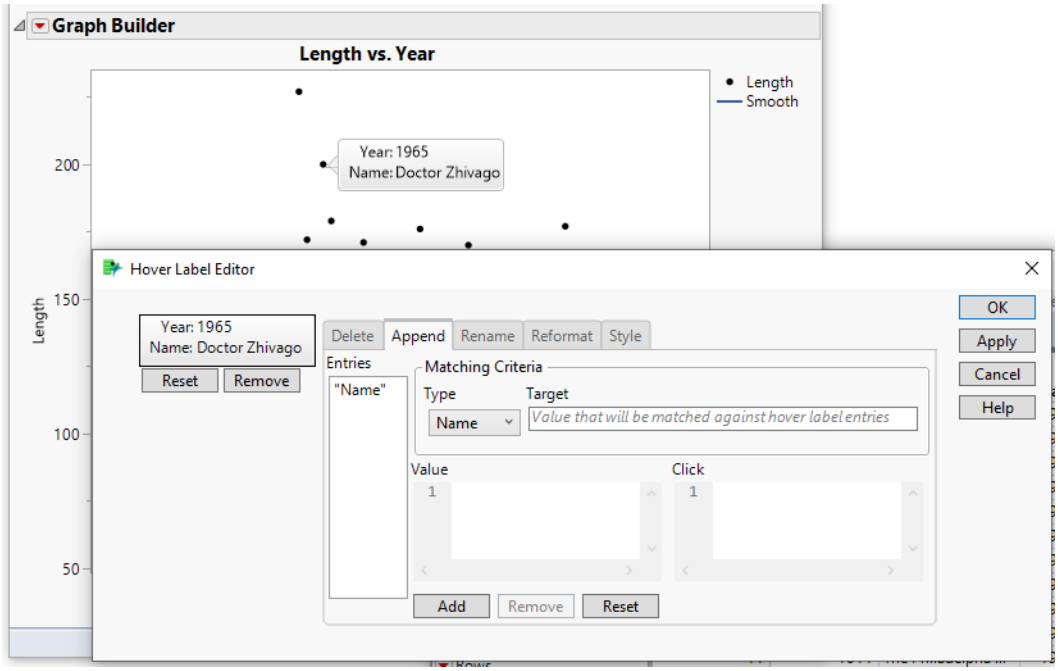
9. Add the movie name and genre:
   a. Click the **Append** tab. Type should be set to **Name**.
   b. Under Target, type Name (the name that you want to add).
   c. Under Value, copy and paste the following JSL expression: `:Name[local:_firstRow]`

      Name is the column reference. Brackets indicate to get the corresponding column value for the given index. `local:_firstRow` is the index, which is a Hover Label Execution Context (HLEC) variable. For details, see the *Scripting Guide*.

   d. Click **Add** and then click **Apply**.

      In the thumbnail and in the graph hover label, you can see that the name of the movie is added.

Figure 9.60 Hover Label with Name Added



   e. Under Target, type Genre (the next name you want to add).
   f. Under Value, copy and paste the following JSL expression: `:Genre[local:_firstRow]`
   g. Click **Add** and then click **Apply**.

In the thumbnail and in the graph hover label, you can see that the genre of the movie is added.

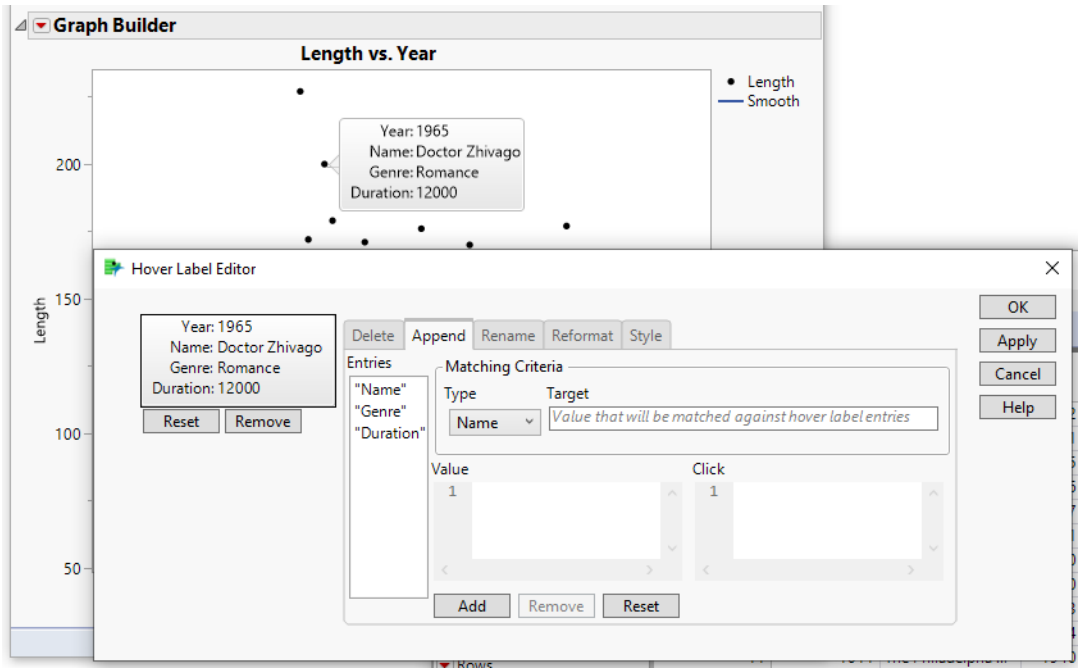
10. Convert the Length data from minutes to hours and minutes:
    a. On the **Append** tab, in the Target box, type Duration (the name of the new hover label that you want to add).
    b. Under Value, copy and paste the following JSL expression: `local:_Length * 60`

       local:_Length is an HLEC variable, with the value of the grid entry Length. Length is in the graph, so we have access to this value in the hover label context. For details, see the *Scripting Guide*.

       * 60 converts the duration from minutes to seconds. In order to apply the hours:minutes format, the duration needs to be in seconds.
    c. Click **Add** and then click **Apply**.

       In the thumbnail and in the graph hover label, you can see the Duration label is added. But the value needs to be formatted correctly.

Figure 9.61 Hover Label with Duration Added



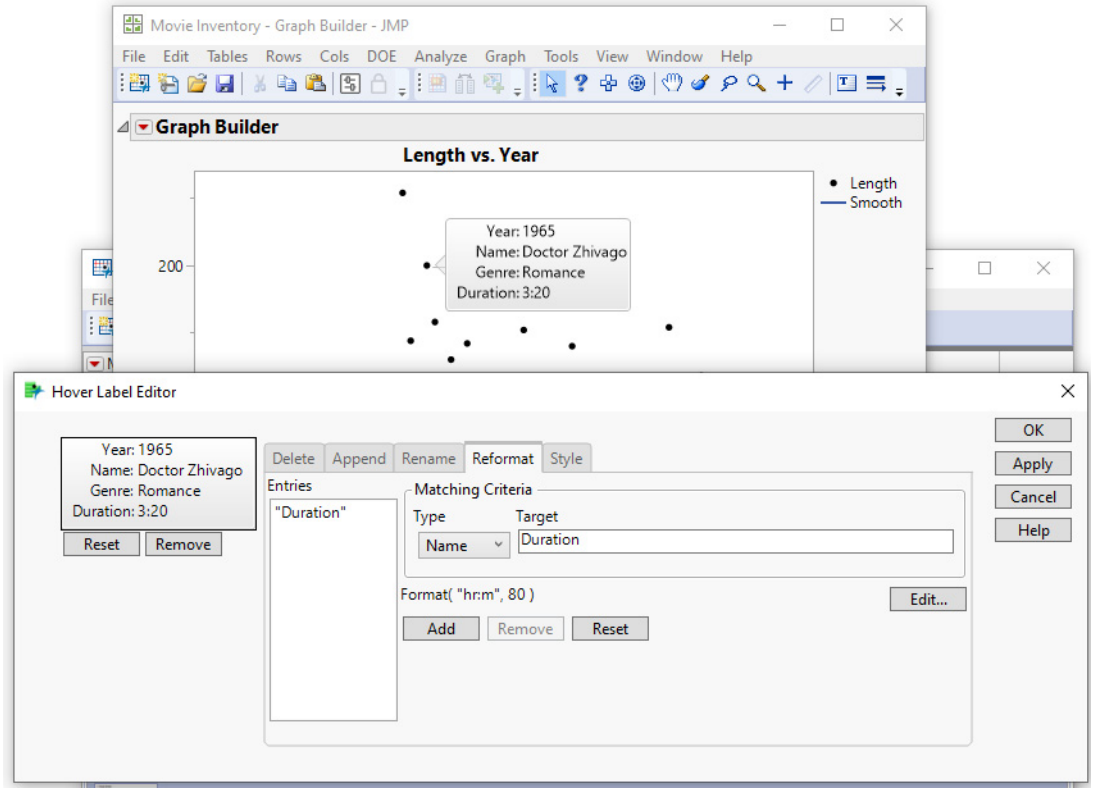
    d. Click the **Reformat** tab.
    e. Under Target, type Duration (the name of the hover label entry that you want to reformat).
    f. Click **Edit**.

g. Next to Format, select **Duration** > **hr:m** and then click **OK**.

h. Click **Add** and then click **Apply**.

On the hover label, you can see that the Duration is now in hours and minutes.

Figure 9.62 Hover Label with Duration in Hours and Minutes



11. Make the Genre values bold:

a. Click the **Style** tab.

b. Under Target, type Genre (the name of the hover label entry that you want to make bold).

Notice that Scope is set to **Values**. This bolds the Genre values instead of the name Genre.

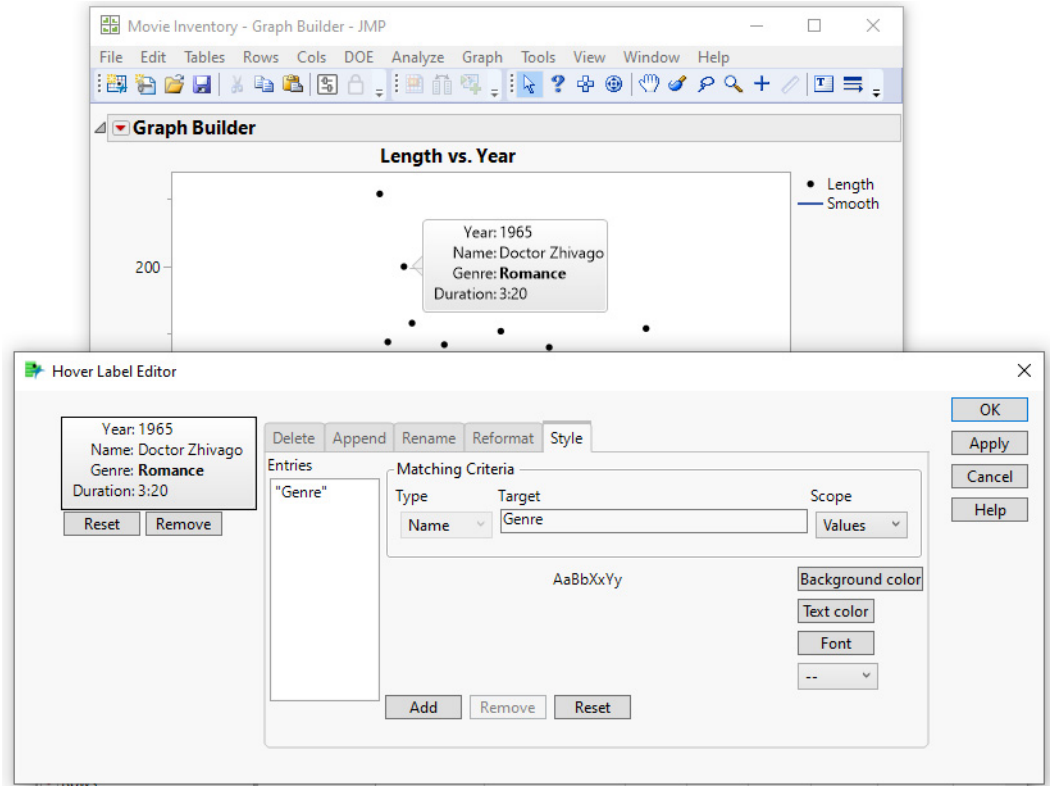
c. Click **Font**.

d. Under Font style, click **Bold** and then click **OK**.

e. Click **Add** and then click **Apply**.

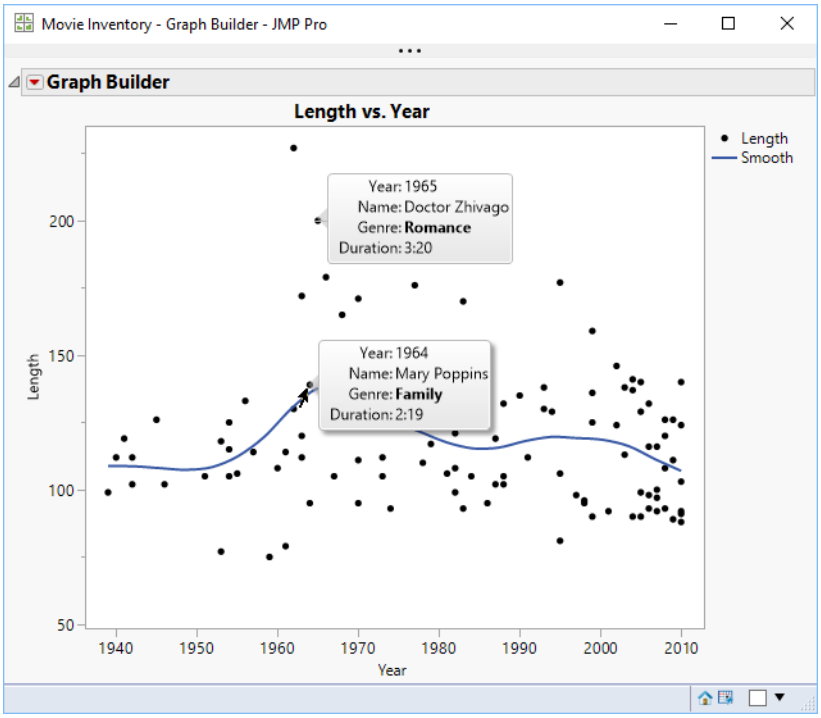
On the hover label, you can see that the Genre value, Romance, is now bold.

Figure 9.63 Hover Label with Genre Value in Bold



Hover over the other labels to see that they are all updated.

Figure 9.64 Updated Hover Labels



Use Labels in Data Tables

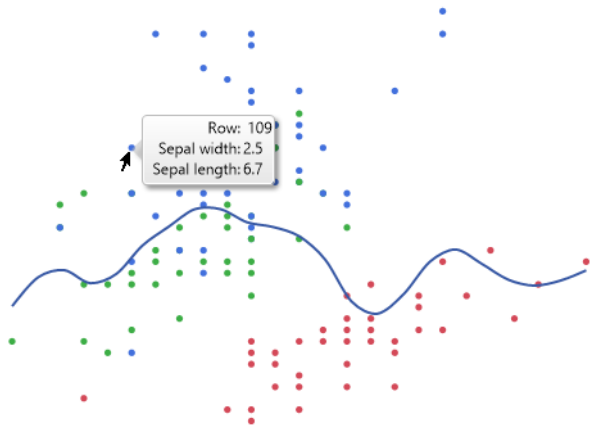
In addition to gridlets, you can also do some basic customization of hover labels by adding Labels to columns in data tables.

- If your hover label contains a Row entry, you can replace it with values in columns by adding a Label.
- Once you add a Label, you can make it persist on the graph. You can also move the label's location on the graph. A tag line might be added to connect the label to its point.

Example Using Labels in Data Tables

1. Select **Help** > **Sample Data Library** and open Iris.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select Sepal length and drag it to the **Y** zone.
4. Select Sepal width and drag it to the **X** zone.
5. Click **Done**.
6. Hover over the point in row 109.

Figure 9.65 Hover Label for Row 109



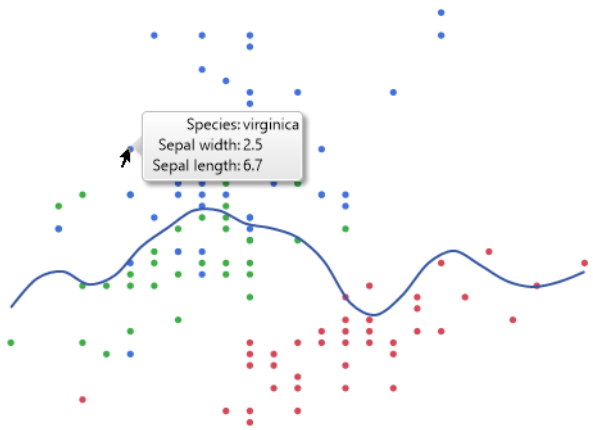
You can see that a Row entry appears in the hover label. You want to replace it with a column and its value from the data table called Species.

7. In the data table, right-click the Species column and select **Label/Unlabel**.

   A label icon appears beside the column name in the Columns panel.

8. In Graph Builder, hover over the same data point.

Figure 9.66 Hover Label Showing Species Column



You can see that the Row entry is replaced by the Species column.

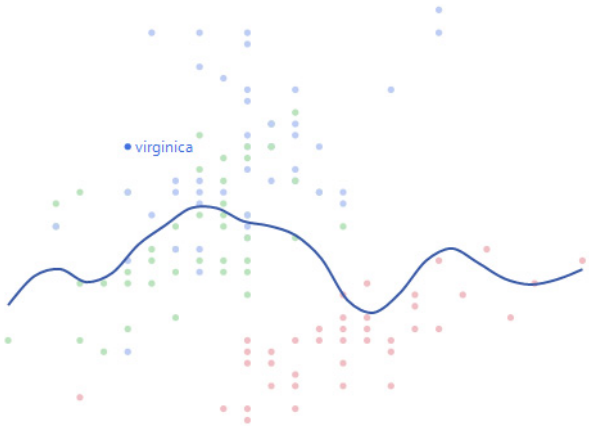
Note: If you specify multiple labeled columns, their values appear alphabetically in the hover label.

Now, you want to make some of the labels persist on the graph.

9. Click the same data point to highlight it.
10. Right-click anywhere and select **Rows** > **Row Label**.

Note: In a histogram, right-click the box plot area on the right.

Figure 9.67 Species Label Showing on Graph

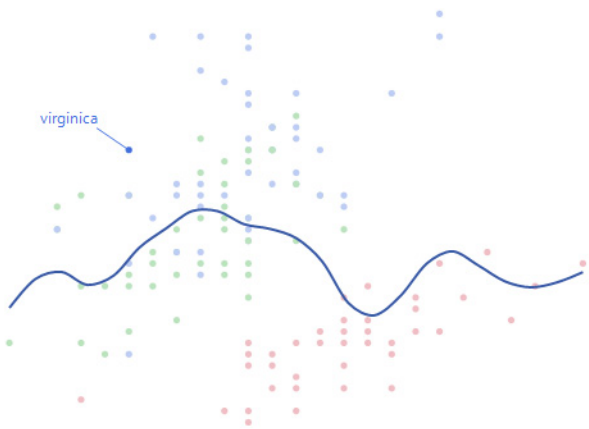


You can see that the label containing the Species value for this data point appears on the graph.

11. Click and drag the label to move its position.

    A tag line is added to connect the label to the point.

Figure 9.68 Reposition the Species Label



In the data table, a label icon appears beside the row number.

Data remains labeled until you highlight the column or row and select **Label/Unlabel** again.

Edit Hover Labels in Graphs

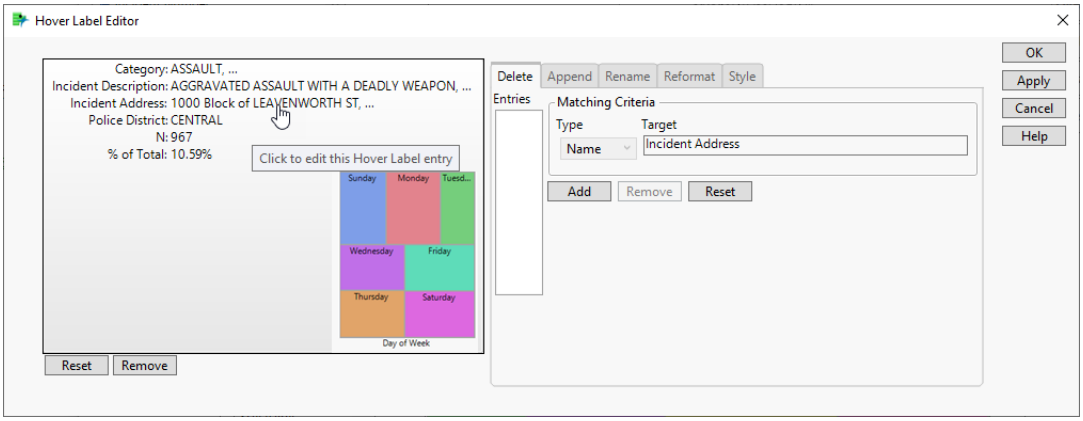
Use the Hover Label Editor to customize and add graphs, images, formatted text, names, and values in hover labels.

To interactively edit hover labels

1. In a graph, hover over data to show a hover label.
2. In a hover label, right-click and select **Edit**.

In the Hover Label Editor, a thumbnail of the hover label appears at left. Click an area in the thumbnail to edit it at right. Changes that you make apply to all hover labels in the graph.

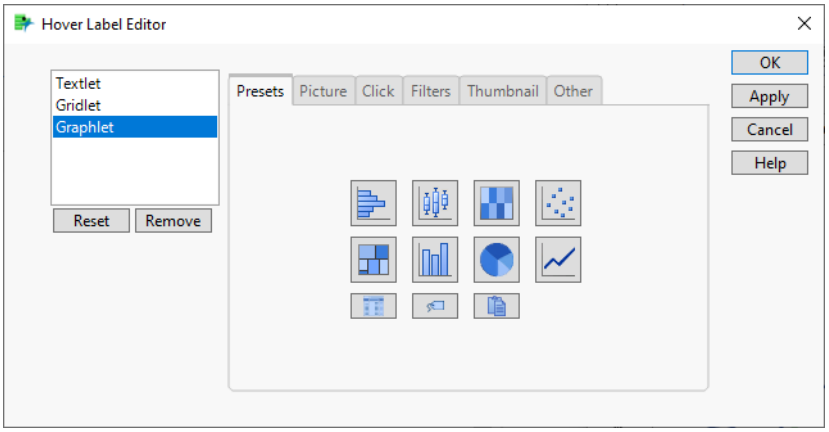
Figure 9.69 Interactively Edit a Hover Label



In this example, the Incident Address data is clicked and the Delete tab appears, so you could delete the Incident Address entry from all hover labels in the graph. You could also rename it, reformat it, or change the font style.

To add or edit hover labels generally, launch the Hover Label Editor. Right-click on a graph and select **Hover Label** > **Hover Label Editor**.

Figure 9.70 Hover Label Editor



On the left, you can choose **Textlet**, **Gridlet**, or **Graphlet**, depending on what you want to do in hover labels:

Textlet Add formatted text to hover labels using JSL. For an example, see "Example of Adding Text to a Hover Label". For details about this window, see the *Scripting Guide*.

Gridlet Delete, add, rename, reformat, or choose a style for names and values that appear in hover labels. For an example, see "Example of Updating Names and Values in a Hover Label". For details about this window, see "Gridlet Tabs".

Graphlet Add an interactive graph or image to hover labels. For an example, see "Example of Adding a Preset Graph to a Hover Label". For details about this window, see "Graphlet Tabs".

Textlet Tabs

Markup Define HTML markup used to create rich text on the hover label.

Presets Examples of a preconfigured textlet.

- The Showcase textlet is an example of how to add color and style to the textlet content based on the underlying data. "Add Rich Text to Hover Labels in Graphs" for an example.
- The Tabulate textlet is an example of how to add summary statistics for the underlying data to the textlet content.
- The Context textlet is an example of how to display the evaluation context variables in the textlet (for debugging purposes). Evaluation context variables define the environment in which your graphlet and textlet scripts are executed. See the *Scripting Guide* for details.

JSL Variables (Optional) Enables you to assign the result of expressions to local JSL variables. The variables are used in the HTML Markup field to create dynamic, data-driven rich text.

HTML Markup Specifies a rich text paragraph template based on a subset of HTML tags. The markup can contain JSL variable references enclosed in delimiters. The default delimiters are braces ({}). Table 9.6 describes the supported HTML tags:

Table 9.6 Supported HTML Tags in Textlets

HTML Tag	Description
	bold text
	italicized text
<u> </u>	underlined text
 	stylized font with any of the following attributes: • color=*color*: The quoted foreground color value. Can be a color name or a hexadecimal RGB color value. • face=*family*: The quoted name of an installed font family. • size=*size*: The quoted font size.
<background> </background>	background color value • color=*color*: The quoted color name or hexadecimal RGB color value.

Other Specify the delimiters and width of text on the hover label.

Delimiters Select the delimiter that indicates variables for replacement in the Markup area. For example, you might use square bracket delimiters for JMP variables if the script contains C++ code or other code that also uses braces.

Width Specify the preferred width in pixels for the hover label.

Note: After you make changes to the textlet configuration, you can click Apply to see your changes on a pinned hover label. When you are satisfied with the updated hover label, click **OK** to save your changes.

Gridlet Tabs

Gridlet tabs include Delete, Append (add), Rename, Reformat, and Style. In all of these tabs, you specify the Matching Criteria, which identifies the hover label entry that you want to modify.

Type and Target

Matching Criteria Specify the hover label entry that you want to modify:

Choose the type of entry to select and the target entry.

Note: Not all choices appear in all tabs.

Name Enter the exact name of the entry that you want to modify. For example, if the entry name is Mean(height), enter this target value: Mean(height).

Figure 9.71 Hover Label Example

Index Enter the index of the entry in the grid that you want to modify. For example, if the third entry in the hover label is Mean(height), enter 3 as the Target value.

Column Enter the column name of the entry that you want to modify. For example, if the entry is Mean(height), enter this target value: height.

Regex Enter a regular expression matched against the name that you want to modify. For example, if the entry is Mean(height), enter this target value: ^.*\(height\)$. (The Pat Regex, parenthesis, and quotations are automatically added.)

All Changes the style for all hover label entries.

Gridlet Buttons

Add Adds a specified Target value under Entries. Click **Apply** to apply your changes to the hover label.

Remove Removes a specified Target value under Entries. Click **Apply** to apply your changes to the hover label.

Replace Replaces a selected Target value under Entries. Click an entry, enter a new target value, and click **Replace**. Click **Apply** to apply your changes to the hover label.

Gridlet Tabs

Delete Specify entries to delete from hover labels.

Note: If you are using JSL, Delete is the `Expunge` command.

Append Specify entries to add to hover labels.

Value Specify the JSL to add the value to the hover label. For an example, see "Update Names and Values".

Click Add an action to the entry that will appear as a link. Specify the script to be executed when the link is clicked. For example, the script can create a visualization, open a web page, or launch a platform. Here are a few ways that you can copy and paste a script:

Create a visualization to link to: In Graph Builder, create the visualization that you want. Click the Graph Builder red triangle menu and select **Save Script** > **To Clipboard**. Paste the script under Click.

Link to a preset graph: In Graph Builder, from the underlying graph that has the preset hover label graphs, right-click > **Hover Label** > **Hover Label Editor**. Under Graphlet, in the Picture tab, copy the script. Paste the script under Click.

Click **Apply** to apply your changes to the hover label.

Note: If you are using JSL, Append is the `Annex` command.

Rename Specify entries to rename in hover labels. Enter a new name to replace the current Target value.

Reformat Specify a different format for entry values in hover labels. For example, you can change the date or duration format, or change the number of decimal points. Click **Edit** to choose a different format. For details about the Edit options, see "Numeric Format Options"

Style Specify a background color, text color, or font details (such as bold, and so on) for entries in hover labels.

Scope Specify if the style updates should be made to the names of the entries, the values, or both. For example, if you are applying a style change to all the matching criteria, you can easily align all the names to the left, or all the values to the right.

Background color Specifies the background color for the Target value, name, or both in hover labels.

Text color Specifies the text color for the Target value, name, or both in hover labels.

Font Specifies font details, such as font, font style, and size for the Target value, name, or both in hover labels.

Alignment Specifies how to align the text for the Target value, name, or both in hover labels.

Graphlet Tabs

Presets Click a preset graph to add it to hover labels. For details about preset graphs, see "Add Graphs or Images to Hover Labels".

Picture Add a JSL script to define the type of graph or image to add to hover labels.

Click Add a JSL script that is executed when a user clicks on the thumbnail (or hover graph) that is defined by the Picture script. If nothing is defined here, the Picture script is reused, so that the image or graph defined under Picture is launched in a new window.

Filters Remove a filter column from your hover graph. Click a column that you want to remove, or add a "1" under Skip Filters Flag. See "Example of Removing a Filter Column from a Hover Label Graph".

Thumbnail Specify the size of the graph thumbnail and whether to include axis values.

Other Use JSL to specify the following information:

Title Sets the title of the new window that is created when a hover graph is launched.

Reapply Defines whether the current hover label graph customizations are applied to a new window launched when clicking on a hover graph.

Add Images from Data Tables to Hover Labels

You can add images to hover labels using graphlets, but you can also add them from a data table.

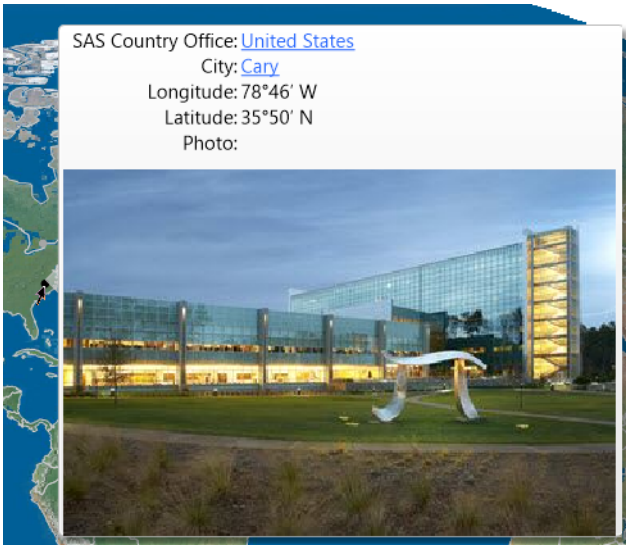
Tip: For details about adding JMP graphs or images using graphlets in a hover label, see "Customize Hover Labels in Graphs".

1. Create a new column in your data table.
2. Right-click the column, select **Column Info**, and select the **Expression** data type.
3. Click **OK**.

4. Drag graphics from your computer into each cell of the column. You can also import data that contains a column of images.
5. Right-click the column in the data table and select **Label/Unlabel** to apply a label.
6. Create a graph using your data table (for example, in Graph Builder or Distribution).
7. Hover over a data point or a histogram bar.

   The image appears on the label. To see an example, run the Graph Builder script in the SAS Offices.jmp sample data table.

Figure 9.72 Image on Hover Label



Tips:

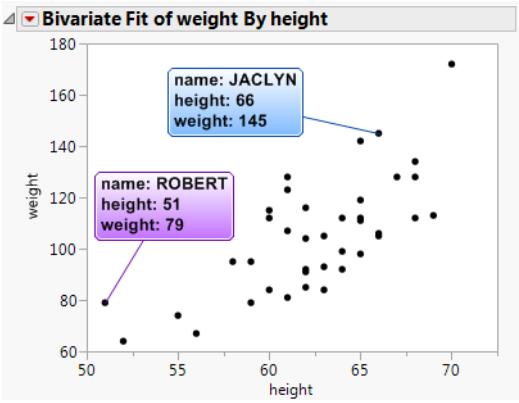
- Images in an expression column can also be used as markers. See "Use Images as Markers".
- You can pin the hover label so that the label always appears on the graph. See "Pin Hover Labels in Graphs".

Pin Hover Labels in Graphs

When you hover over data on a graph, a hover label that contains data values appears. You can pin the hover label so that it always appears on the graph as a label. This is helpful if you want to compare several labels, or if you want to show the label information. When you save the graph, the label is saved. When you print the graph, the label is printed.

Note: The hover label remains pinned after you redo an analysis. However, if the data changed, you might have to reposition the label.

Figure 9.73 Pinned Hover Labels



To pin a hover label, hover over the data point until the label appears. Click the pin icon in the upper right corner of the label. You can then drag the label to reposition it.

Right-Click Options for Hover Labels

Background Color Applies the selected color to the label background.

Frame Color Applies the selected color to the label border and tag line.

Text Color Applies the selected color to the text.

Filled Applies the background color and frame color to the label. Selected by default.

Use Gradient Shades the background color vertically from light to dark. Selected by default.

Font Applies the selected font family to the text. The default font is determined by your JMP Font preferences.

If a script with a pinned label refers to a font that is not on the computer, the script does not run.

Add Text Adds text to the bottom of the label.

Replace Text Change the text on the tooltip. If you redo the analysis, the tooltip text remains updated.

Tag Line Draws a line between the label and the data point. The Frame Color is applied to the line.

Copy Copies the hover label contents to the clipboard.

Subset Opens a data table with the subset data associated with the hover label.

Edit Opens the Hover Label Editor, where you can further edit the contents of the hover label. See "Edit Hover Labels in Graphs".

Close Removes the label. You can also remove the label by clicking the X in the upper right corner of the label.

Note: To return to the default label formatting, remove the label and add it again.

Chapter 10

Save and Share Your Data

Get Your Data Out of JMP

This chapter covers the following topics:

- saving data tables as text files, SAS data sets, Excel files, and other formats
- saving reports as interactive HTML and PowerPoint
- emailing reports and data tables
- saving analyses as scripts
- saving parts of a report as a graphic
- printing reports
- copying and pasting reports
- creating JMP journals or projects
- saving JMP sessions
- saving log windows

The method that you choose depends on how you want to interact with your analyses in the future:

- If you want to interactively present your data, you might choose to create a JMP journal.
- If you want to send data tables, linked reports, and other items to someone, you might choose to create a JMP project.
- If you want to continue to explore a report later, you might choose to create a JMP report.
- If you want to send a report to a non JMP user, you might convert the report to interactive HTML or a web report.

Contents

Save and Share Data Tables . . . 566
Save Data Tables as Microsoft Excel Files . . . 567
Save JMP Data as a CSV File . . . 568
Save JMP Data as a Text File . . . 569
Save JMP Data as a JSON File . . . 570
Save JMP Data as a SAS Transport File . . . 570
Save JMP Data as a Google Sheet . . . 572
Save JMP Data as a SAS Data Set . . . 573
Save Data Tables to a Database . . . 574
Replace a Database with a Data Table . . . 575
Save and Share Reports . . . 576
Share Reports as Interactive HTML . . . 578
Publish Reports to JMP Live or JMP Public . . . 582
Save Reports as PDF Files . . . 591
Save Reports as PowerPoint Presentations . . . 592
Setting the Graphic DPI for Exported Graphics . . . 594
Save a JMP Analysis as a Script . . . 594
Save Parts of a Report in a Graphic Format . . . 594
Print Reports . . . 596
Copy and Paste Reports . . . 596
Work with Projects . . . 598
Create a New Project . . . 599
Open Files in a Project . . . 599
Rearrange Files in Projects . . . 600
Save a Project . . . 601
Project Workspace . . . 602
Project Bookmarks . . . 603
Project Contents . . . 603
Recent Files in Projects . . . 604
Project Log . . . 605
Move Files into Projects . . . 605
Write a Project On Open Script . . . 606
Example of Creating a New Project . . . 606
Create Journals . . . 609
Example of Making a Journal for a Presentation . . . 610
Create New Journals . . . 612
Manage Links in Shared Journals . . . 613
Add and Edit Outlines to Journals . . . 614
Right-Click Options in Journals . . . 615

Control the Display of Outline Levels in Journals 616
Add a Graph or Graphic to Journals 616
Customize Items in Journals 616
Prevent Modifications to Journals 617
Append Reports to Journals 617
Save Journals in Other Formats 617
Delete Items from Journals 618
Save JMP Sessions 618
Save Sessions upon Exiting JMP 619
Save Sessions Manually 619
Save a Log Window 620

Save and Share Data Tables

Select **File > Export** or **File > Save As** to save a data table in another format such as Microsoft Excel or CSV.

Note: To save a data table as a journal, select **Edit > Journal** and then **File > Save As**.

JMP saves data tables in the formats listed in Table 10.1.

Table 10.1 Supported File Types for Saving Data Tables

File Type(s)	OS Availability
JMP Data Table (.jmp)	All
Excel Workbook (.xls, .xlsx)	All
Text Export File (.txt)	All
Comma-delimited File (.csv)	Windows
Tab-delimited File (.tsv)	Windows
SAS Data Set (.sas7bdat)	All
SAS Data Set with Extended Attributes (.sas7bxat) **Note:** JMP can save column properties and table properties as extended attributes in a SAS 9.4 .sas7bxat file. Select **File > Export**, select the **SAS** file type, and then select the check box for saving extended attributes. JMP names the file with the .sas7bxat extension. The SAS Integration preferences also include this option, which is off by default.	Windows
SAS Transport File (.xpt,.stx)	All
JSON Data File (.json)	All

You can also save files for types that have a corresponding ODBC driver. See "Save Data Tables to a Database".

Notes:

- The maximum length of the data table's name depends on your computer's operating system. See your operation system documentation for details.
- JMP tries to save any modified files when a crash is detected.

Save Data Tables as Microsoft Excel Files

There are two options for saving data tables as Microsoft Excel files:

- Save one data table in Microsoft Excel format (.xls or .xlsx). Data from a single table is saved in a workbook named after the data table.
- Generate a workbook in .xlsx format from multiple tables. Data from each table appear on a different worksheet in the workbook.

Saving an .xlsx file retains data table cell colors, hidden data, and excluded data. Formulas and most formatting are not included.

The maximum length of the Excel filename is determined by your computer's operating system.

Save One Data Table as a Microsoft Excel Workbook

To save a data table as a Microsoft Excel workbook, follow these steps:

Windows

1. Select **File > Export.**
2. Select **Microsoft Excel** and click **Next**.
3. Enter a name for your file in the File name box.
4. (Optional) To save the file as an .xls file, select **Use Excel 97-2003 Format (*.xls)**.
5. (Optional) To avoid opening the file in the default spreadsheet program after saving, deselect **Open the file after** saving on the Export window.
6. Click **Save**.

macOS

1. Select **File > Export.**
2. Select **Microsoft Excel** and select the format.
3. Click **Next**.
4. Enter a name for the file in the **Save As** box.
5. (Optional) To avoid opening the file in the default spreadsheet program after saving, deselect **Open the file after saving**.
6. Click **Export**.

Note: On Windows, data is split among several worksheets when the number of columns or rows exceeds the number that Excel supports.

Save Multiple Data Tables as a Microsoft Excel Workbook

To save multiple data tables in a new or existing Microsoft Excel workbook, follow these steps:

1. Open the data tables.
2. (Windows) Select **File > Export > Microsoft Excel**, select **Export Multiple Data Tables**, and then click **Next.**
3. (macOS) Select **File > Export**, select **Export Multiple Data Tables** from the Excel list, and then click **Next**.
4. In the JMP Data Table column, select only the data tables that you want to save.
5. In the Worksheet Name column, specify the name of the worksheet tab if necessary.
6. Enter a name for the workbook in the Workbook Name box.

   The workbook is named after the current data table by default.
7. Click **Choose** and browse to select a directory to save the file in.

   You can overwrite an existing workbook. Worksheets that already exist in the workbook are replaced. Worksheets that are not in the existing workbook appear at the end of the sheets.
8. Click **Save** (or **Export** on macOS).

   The file is saved as an .xlsx file.

Save JMP Data as a CSV File

On Windows, JMP can convert data from a JMP data table to a comma-separated values (CSV) file. To save a data table as a CSV file:

1. With the specific data table open in JMP, select **File > Save As**.
2. Select CSV (Comma delimited) (*.csv) from the **Save as type** list.
3. Enter the filename.
4. Select the location for the CSV file.
5. Click **Save**.

Note: Exporting a data table to a CSV file uses the comma and carriage return line feed character (CRLF) as file delimiters. Use a different extension and later rename the file with the CSV extension if you need to use different delimiters.

Save JMP Data as a Text File

JMP can convert data from a JMP data table to standard text format with rows and columns.

Text Formatting Options

Export Column Names to Text File To save column headings in the first line of text, and to save labels or header information with the data, select **Export Table Headers.**

Add quotation marks to all column names, Add quotation marks to all character values, Add quotation marks to all numeric values Select an option to add quotation marks even when they are not required. Some programs that import CSV files might need the quotation marks to display the text correctly. Quotation marks are always added to fields that contain special characters such as the field delimiter (typically a comma or tab), the line separator (typically a carriage return), or quotation marks.

End of Field Select the radio button next to the character that marks the end of a field (or cell). Select **Other** and enter a character if the appropriate character is not listed.

End of Line Select the radio button next to the character that marks the end of a line (or row). Select **Other** and enter a character if the appropriate character is not listed.

Notes:

- If a double quotation mark is found when importing text data, the delimiter rules change, and JMP looks for an end double quotation mark. Other text delimiters, including spaces embedded within the quotation marks, are ignored and treated as part of the text string.
- When you import text data, columns that contain image data are assigned the Expression data type.

Windows

1. Select **File > Expor**t.
2. Select **Text** and then click **Next**.
3. Enter the file name in the File name box.
4. Click the **Options** button and specify end-of-line and end-of-field characters, and choose whether to export column headings as text.
5. Click **OK**.
6. (Optional) To avoid opening the file in the default text editor after saving, deselect **Open the file after saving**.
7. Click **Save**.
8. Click **Yes**.

macOS

1. Select **File > Export**.
2. Select **Text** and then click **Next**.
3. Enter a name for the file in the Save As box.
4. Specify end-of-line and end-of-field characters, and choose whether to export column headings as text.
5. (Optional) To avoid opening the file in the default text editor after saving, deselect **Open the file after saving**.
6. Click **Export**.

Save JMP Data as a JSON File

You can save a data table as a JSON file, which consists of name and value pairs. JSON files are often used to exchange data between applications.

To save a data table as a JSON file, follow these steps:

Windows

1. Select **File > Export**.
2. Select **JSON Data** and click **Next**.
3. Enter the file name in the File name box.
4. Click **Save**.

macOS

1. Select **File > Export**.
2. Select **JSON Data** and click **Next**.
3. Enter a name for the file in the Save As box.
4. Click **Export**.

Save JMP Data as a SAS Transport File

You can save a JMP data table in SAS transport file format. JMP replaces spaces in filenames and column names with underscores, converts column headings to uppercase SAS variables, and makes other changes to follow the transport file specifications.

When you save the data table as a transport file, you can opt to append it to an existing transport file. When you reopen the transport file in JMP, the two data tables open in separate windows.

Notes:

- The maximum length for SAS transport filenames is 8 characters. JMP warns that characters beyond that limit are omitted.
- Images in Expression columns cannot exceed 200 characters, otherwise they are not supported.

Windows

1. Select **File > Export.**
2. Select **SAS Transport File**.
3. Click **Next**.
4. Enter the file name in the File name box.
5. Click **Save**.
6. Click **OK**.

   Alerts appear regarding other SAS format modifications.
7. Click **OK** on each alert.
8. (Optional) Append the data table to an existing SAS transport file. Select **File > Export**, select the **Append** check box, and then click **Next**. Select the file to which you want to append the data table and then click **Save**.

macOS

1. Select **File > Export.**
2. Select **SAS Transport**.
3. Click **Next**.
4. Enter a name for the file in the **Save As** box, and then click **Export**.

   Alerts appear regarding SAS format modifications.
5. Click **OK** on each alert.
6. (Optional) Append the data table to an existing SAS transport file. Select **File > Export**, select the **Append** check box, and then click **Next**. Select the file to which you want to append the data table and then click **Append**.

Save JMP Data as a Google Sheet

A Google Sheet is a spreadsheet (which consists of sheets, like tabs) that is saved in your Google account. Exporting a JMP data table to Google Sheets enables you to share the data with others and edit the data outside of JMP.

You can export a JMP data table to an existing Google spreadsheet or create a new one.

To export a JMP data table as a Google spreadsheet, follow these steps:

1. If you are exporting to an existing spreadsheet, perform these steps first. Otherwise, go to step 2.
   a. From a browser, log in to https://sheets.google.com.
   b. Open the existing spreadsheet.
   c. Copy the URL from the browser's address bar.
2. In JMP, open the data table with the data you want to export.
3. Select **File > Export > Google Sheets**.
4. Under Spreadsheet Type, choose either **Existing** or **New Spreadsheet**.
5. Type your Google account name. (@gmail.com is unnecessary.)
6. (Existing spreadsheets only) Next to Spreadsheet Name, paste the URL of the Google spreadsheet you copied in step 1.
7. (New spreadsheets only) Enter a name for the new spreadsheet.
8. Enter the name of the sheet (the tab name).
9. Click **Save**.
10. If you have already exported a data table from JMP, the data will appear in Google Sheets. Go to your Google spreadsheet to see the data in the specified sheet.

    or

    If you are exporting data using the specified Google account for the first time, you are prompted to give JMP access to your Google account.
    a. Click **Allow**.
    b. Go to your Google spreadsheet to see the data in the specified sheet.

    Depending on the size of the data table, the save operation might take some time.

About Exporting Google Sheets

- Information that Google uses to authenticate your account is stored in C:/Users/<user name>/AppData/Roaming/SAS/JMP/OAuthTokens.jmp on Windows. On Mac, oAuthTokens.jmp is stored in ~/Library/Application Support/JMP. (Passwords are not stored on the computer, and access tokens are encrypted.) Storing tokens in a data table enables

you to import and export data using the same Google account without providing the authentication information again. Note that a user with access to your computer's user account has access to all of your Google spreadsheets. If the spreadsheets contain confidential information, reconsider sharing your user account.

- JMP features such as formulas and List Check column properties are not supported in Google Sheets.
- If the spreadsheet is empty, look in the JMP log for error messages. On Windows, select **View > Log**. On macOS, select **Window > Log**.
- If a Google Sheet has only one sheet, you cannot replace the sheet with a new sheet. Add a second sheet with a unique name and then delete the original sheet. In the Export to Google Sheets window, select **Existing Spreadsheet** and export the data table again. The sheet name that you specified appears in the sheet.
- See "About Importing Google Sheets" for more information about security, country restrictions, and more.
- If **Display intranet sites in Compatibility View** in Internet Explorer on Windows is selected, the Google Sheet cannot be exported. To avoid this issue, follow these steps:
  1. In Internet Explorer, select the Tools button and then select **Compatibility View settings**.
  2. Deselect **Display intranet sites in Compatibility View**.

Save JMP Data as a SAS Data Set

You can save data tables as SAS version 7 and higher data sets. JMP writes columns as SAS variables, writes rows as SAS observations, and makes other changes to follow the SAS data set specifications.

When you export data to a SAS data set, JMP date columns become SAS date values with the appropriate SAS format.

On Windows, JMP can save column properties and table properties as extended attributes in a SAS 9.4 .sas7bxat file. The option is available in the Save JMP As window and in the SAS Integration preferences.

Windows

1. Select **File > Export.**
2. Select **SAS Data Set** and click **Next**.
3. Enter the file name next to File name.
4. (Optional) To save SAS variable names or SAS formats, select the **Preserve SAS variable names** or **Preserve SAS formats** check boxes.

5. (Optional) To store JMP metadata as extended attributes for SAS 9.4, select **Store table and column properties in SAS 9.4 extended attributes**. JMP automatically assigns the .sas7bxat file extension.
6. Click **Save**.
7. Click **Yes**.
8. If other alerts appear, click **OK** on each alert.

macOS

1. Select **File > Export**.
2. Select **SAS** and click **Next**.
3. Enter the filename and select export options.
   - To avoid exporting rows that have the Excluded row state, select **Honor Excluded Rows**. The other options are described in the preceding section.
4. Click **Export**.

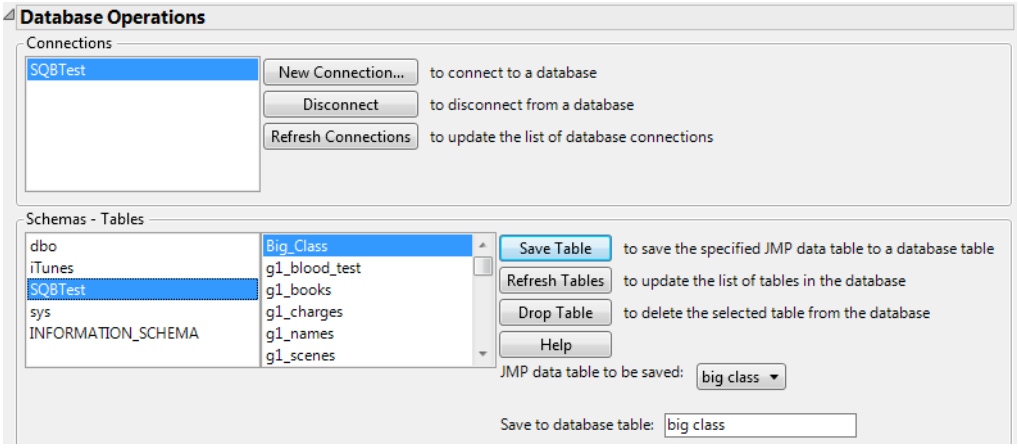
Save Data Tables to a Database

You can save a data table to any database on your system that has a compliant ODBC (Open Database Connectivity) driver.

When exporting a data table to a database using ODBC, JMP exports the data as either character (CHAR or NCHAR) or numeric (FLOAT or DOUBLE PRECISION) data types. If a column is defined as a JMP date or time, the column exports as character data. The receiving database handles changing the data type to a proper date format.

1. First, connect to the database. See "To add a new database connection".
2. Select **File > Database > Save Table**.

Figure 10.1 The Database Save Table Window



3. In the Connections box, highlight the name of the database to which you want to save the file (Figure 10.1). The Connections box contains a list of databases to which your system is connected.
4. From the **JMP data table to be saved menu**, select the open JMP data table that you want to save to the database.
5. In the **Save to database table** box, enter the name that you want the table to have when you save it in the database.
6. Click **Save Table**.

To add a new database connection

1. Click **Connect**.
2. Select the data source that you want and click **OK**. Or, to create a new source, click the **New** button (Windows) or **Choose DSN** button (macOS).

   Depending on which data source you select (and which database drivers you have installed on your computer), you might be presented with a variety of windows. Use them to create the database source.
3. Select the database to which you want to save the file.

Replace a Database with a Data Table

To replace a database with a data table, follow these steps:

1. First, connect to the database. See "To add a new database connection".
2. Select **File > Database > Save Table**.

3. In the Connections box, select the name of the database to which you want to replace the file. The Connections box contains a list of databases to which your system is connected.
4. In the Schemas - Tables box, select the database table that you want to replace.

   Note: Remember the name for the database table. You use this same name for the replacement database table.

5. Click **Drop Table**.

   The database table is deleted from the database.
6. From the **JMP data table to be saved** list, select the JMP data table that you want to save to the database.
7. In the **Save to database table** box, enter the name for the deleted database table.
8. Click **Save Table**.

   The data table is saved to the database.
9. Click **Disconnect**.
10. Close the Database Save Table window.

Save and Share Reports

To save a report, select **File > Export**.

- On Windows, you can save the report as any of the file types listed in Table 10.2.
- On macOS, the report is saved as a JMP report (.jrp). Note the following:
  - You can save the report as a journal by selecting **Edit > Journal** and then **File > Save**.
  - If you want to save the report as a text, image, HTML, PowerPoint, or RTF file, select **File > Export**. Supported image files include .png, .emf, .jpg, .gif, .tiff, .svg, and .eps.

Table 10.2 Windows Supported Report File Types

File Type	Description
Microsoft Word (.doc)	Word processing format; mixture of pictures, text, and tables.
Enhanced Metafile (.emf)	Can contain both vector and bitmap components.
Encapsulated PostScript File (.eps)	Line drawing (or *vector* image) that can show a bitmap preview of the image.

Table 10.2 Windows Supported Report File Types *(Continued)*

File Type	Description
CompuServe Graphics Interchange Format (.gif)	Compressed bitmap pictures.
Hypertext Markup Language (.htm, .html)	Browser format; marked up text and references to separate picture files. Save pictures within the HTML file by selecting one of these formats: PNG, JPEG, SVG, or GIF.
HTML or Interactive HTML with Data (.htm, .html)	Saves the data, reports, and graphs in an interactive web page, so non JMP users can explore the data. See "Share Reports as Interactive HTML".
Joint Photographics Expert Group (.jpg, .jpeg)	Compressed bitmap pictures; standard for photographs. See "Setting the Graphic DPI for Exported Graphics".
JMP Journal (.jrn)	Analysis report duplicated in a separate window titled Journal. You can edit it or append other reports to it. See "Create Journals". On macOS, select **Edit > Journal**, and then **File > Save**. On Windows, choose **File > Save**, and then choose the journal format.
JMP Report (.jrp)	Analysis report originally created in JMP. It can be reopened for continued analysis. Contains the JSL and embedded data or reference to the data to reproduce a live report. Choose this format if you want to work with the report again, perhaps with new data.
Portable Document Format (.pdf)	Format for sharing documents regardless of the operating system or application in which they were created.
Portable Network Graphics (.png)	Compressed bitmap pictures; successor to GIF. See "Setting the Graphic DPI for Exported Graphics".
PowerPoint Presentation (.pptx)	Microsoft PowerPoint format; mixture of pictures, text, and tables. See "Save Reports as PowerPoint Presentations".

Table 10.2 Windows Supported Report File Types *(Continued)*

File Type	Description
Rich Text Format (.rtf)	Word processing format; mixture of pictures, text, and tables. Save pictures within the RTF file by selecting one of these formats: PNG, JPEG, or EMF.
Scalable Vector Graphic (.svg)	Pictures stored as text; best used for two-dimensional graphics.
Tagged Image File Format (.tif, .tiff)	Raster file format. See "Setting the Graphic DPI for Exported Graphics".
Text Format (.txt)	Plain text format; no pictures.

Share Reports as Interactive HTML

Interactive HTML enables you to share JMP reports that contain dynamic graphs so that even non JMP users can explore the data. JMP reports are saved as an interactive web page that you can share with others (for example, on a shared network drive, by email, or on a website). Users then explore the data as they would in JMP.

Interactive HTML Contains Data

When you export or publish a report as interactive HTML, your data are embedded in the HTML. The content is unencrypted, because web browsers cannot read encrypted data. To avoid sharing sensitive data, export your results as a non-interactive web page instead by selecting **File > Export > HTML**.)

Which Features Are Supported Interactively

Interactive HTML provides a subset of features from JMP:

- If the features in your report are fully supported, the web page is created with no warnings.
- If your report contains unsupported features, you see a message in the export or publish window that interactive HTML is partially implemented. For details, see the JMP log (**View** > **Log**).
- Partially or unsupported features appear static in the web page. If you hover over an unsupported feature in the web page, a tooltip says that the feature is not yet interactive.

For information about exploring interactive HTML output, visit https://www.jmp.com/interactive.

Export a Single Report as Interactive HTML

To export a single JMP report as interactive HTML, use the Export as Interactive HTML with Data option to create a single web page.

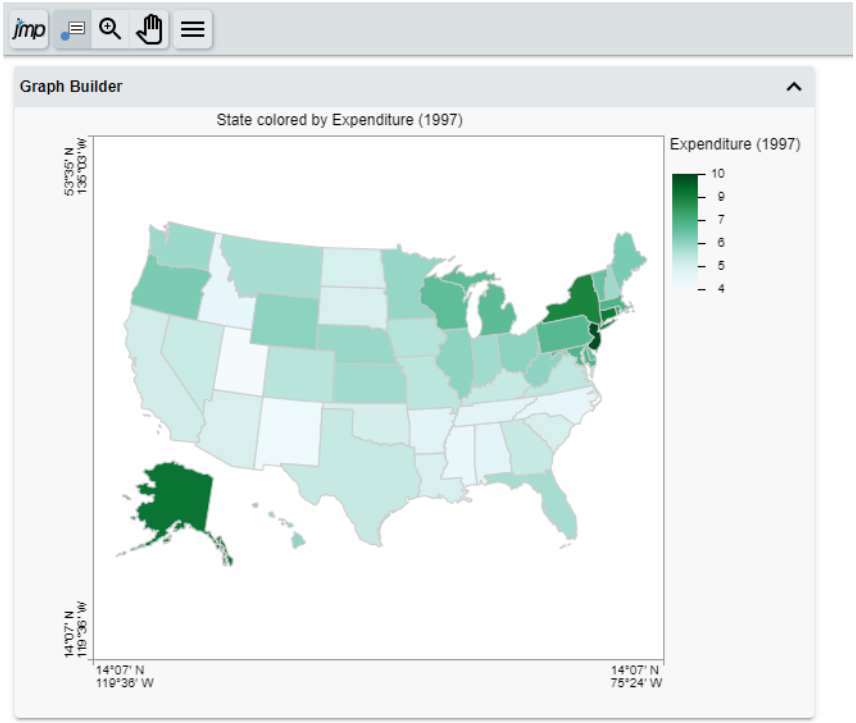
1. In JMP, create the report and make it the active window.

   Note: If a report contains a Local Data Filter, it is static in the web report and users cannot change the selections. To make the Local Data Filter interactive, deselect the **Include** mode. (In Graph Builder, to see the **Include** mode, select **Show Modes** from the Local Data Filter red triangle menu.)

2. Select **File > Export**, select **Interactive HTML with Data** and click **Next**.
3. Name the file.
4. (Optional) To open the HTML file in your default browser after exporting it, select **Open the file after saving**.
5. Click **Save**.

   The output is saved in the selected folder.

Figure 10.2 Web Page for Single Interactive HTML Report



Publish Multiple Reports as Interactive HTML

To publish multiple JMP reports as interactive HTML, use the Publish to File option, which creates an index page that contains the reports.

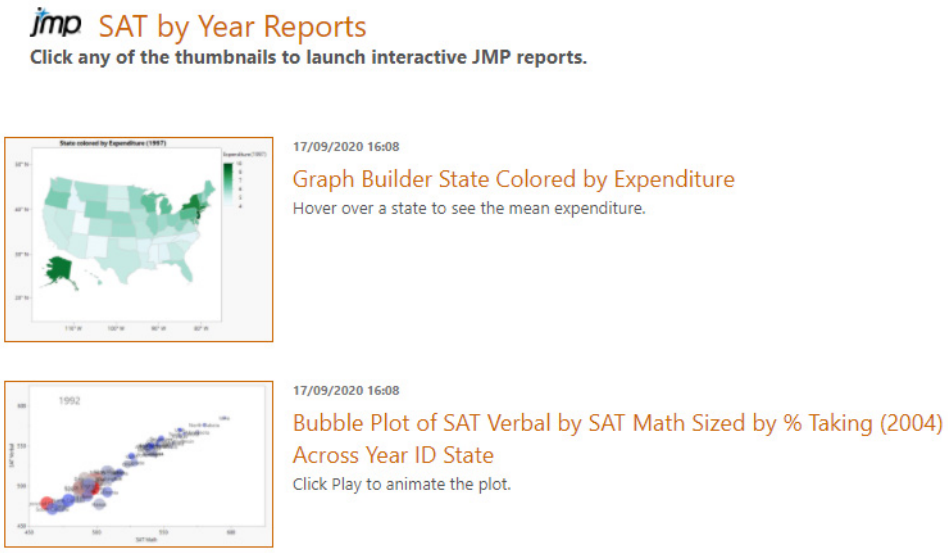
Tip: When you publish multiple JMP reports, you specify where to save the index page and report files. You can choose a shared network folder and provide the location to others, or choose a folder on your computer and zip the files before sharing. This is particularly helpful if you are sharing with non JMP users.

1. In JMP, create the reports.

   Note: If a report contains a Local Data Filter, it is static in the web report and users cannot change the selections. To make the Local Data Filter interactive, deselect the **Include** mode. (In Graph Builder, to see the **Include** mode, select **Show Modes** from the Local Data Filter red triangle menu.)

2. From a report window, select **File** > **Publish** > **Publish to File**.
3. Select the reports that you want to publish.
4. (Optional) Change where the parent folder resides or change the name of the subfolder that will contain the reports.
5. Click **Next**.
6. Enter a title for the index page. You can also update the report titles.
7. (Optional) Change additional options. See "Interactive HTML Report Options".
8. Click **Publish**.

Figure 10.3 Index Page for Multiple Interactive HTML Reports



9. Click a thumbnail to open a report.

For details about working with interactive reports, from an HTML report, click ☰ > **Help**. This opens the help at https://www.jmp.com/interactive.

Interactive HTML Report Options

Title Add a title for the index page (multiple reports only) or the reports.

Description (Optional) Add a description to the index page or the reports. The description will appear under the titles.

Customize (Appears for multiple reports only) Change the appearance of the web page. You can change the style, theme, font, logo, and whether the date or time appears. See "Customize Index Page Options".

Publish Data Select this for interactive reports. If you deselect this option, the reports are static.

Note: To avoid sharing sensitive data, save your results as a non-interactive web page. (Select **File > Export > HTML**.)

Add Image Adds an image to the bottom of the web page.

Open published web report Opens the web page in a browser once you click Publish.

Close reports after running Closes the reports in JMP once you publish the web report.

Delete icon (Appears for multiple reports only) Deletes a report.

Arrow icons (Appears for multiple reports only) Changes the order of reports.

Customize Index Page Options

Style Format Determines the layout of the reports.

Large List Shows the reports in a column with large thumbnail graphics.

Small List Shows the reports in a column with small thumbnail graphics.

Grid Shows the reports in rows.

Custom CSS Enables you to specify a CSS file to format the web page. The CSS file is copied into a subfolder called _css. The link to the CSS file is relative so that you can send the report and support files to another user and maintain the CSS formatting.

Color Theme Specifies the color of the web page, headings, and borders. The default web page has a white background, orange headings, and blue borders.

Change Font Change the font applied in the reports.

Change Logo Specifies an image to display along with the reports. Click the up or down arrow next to the image to move it above or below the reports.

Show date/time Shows or hides the date and time at which the web page was generated.

Publish Reports to JMP Live or JMP Public

JMP Live is a website purchased by your company for sharing JMP content privately. JMP Live can be hosted by SAS or your company. JMP Public is a website using the same technology as JMP Live, where anyone can share JMP content with the public.

JMP Live and JMP Public are platforms for sharing JMP data, visualizations, and dashboards. Prepare your reports in JMP and publish them as posts to collaborate with your colleagues, even those who might not have JMP.

JMP Live and JMP Public offer the visual interactivity of JMP to explore your data, and let you do the following:

- Store all of your reports in one location.
- Control sharing of individual reports with those who need to see them.
- Maintain up-to-date reports for your team in an easily accessible location.

- Extend the functionality of reports to other locations: embed and interact with your reports in web pages, blogs, social media posts and more.

Register and Sign Into a SAS Profile (JMP Public Only)

Before you can publish posts to JMP Public, you must register and sign in using a SAS profile.

Note: If you are not signed in to JMP Public, you see only featured posts.

1. In a browser, go to https://public.jmp.com.

   Supported browsers include Chrome, Firefox, Safari, and Edge (based on Chromium).
2. At the top right, click **Sign in**.
3. If you already have a SAS profile, log on using those credentials, then click **Submit**.
4. If you do not have a SAS profile:
   a. Click **Create profile**. Complete the form and click **Create profile**.
   b. Follow the instructions in the email to set your password and activate your profile.

Add a JMP Live Server

Before you can publish to JMP Live, you must add a JMP Live server.

1. In JMP, select **File** > **Publish** > **Manage Connections**.
2. Click **Add New**.
3. Enter an identifying name for the connection, such as JMP Live.
4. Enter the server URL, formatted like this: https://jmpliveserver.com:3501 (where 3501 is the default web server port).
5. (Optional) To use JSL to create JMP Live reports, enter an API key.
6. Click **Next**.
7. Sign in to the JMP Live server with your user name and password.
8. Once the connection is created successfully, click **Save**.

   The server name appears under Select a connection.
9. Click **Exit**.

Create and Publish JMP Reports

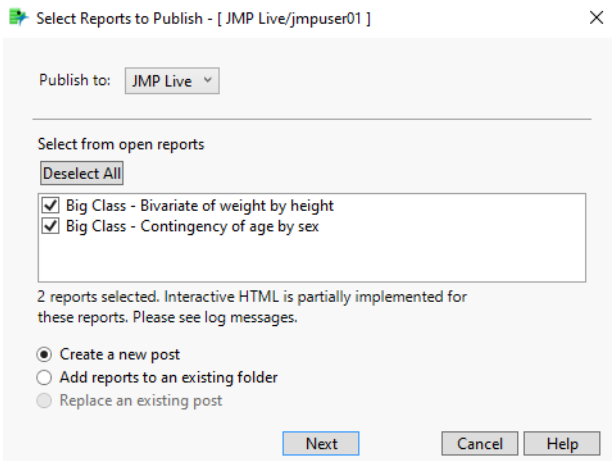
1. In JMP, create the reports.

   You can create a single report, which appears as its own post in JMP Live or JMP Public, or multiple reports which appear in a folder.

2. (Skip if your report does not contain a Local Data Filter.) If the report contains a Local Data Filter, add columns before you publish, otherwise the Local Data Filter will be static instead of active. To add columns, select them in the Local Data Filter and click the Add icon.
3. Select **File** > **Publish** > **Publish to JMP Live** or **Publish to JMP Public**.
   - (JMP Live only) If you have not added a JMP Live server, add one now. See "Add a JMP Live Server".
   - You might need to sign in to JMP Live or JMP Public.

   Note: To register for a SAS login, visit https://www.jmp.com and click **Sign In** near the search box at the top of the page.

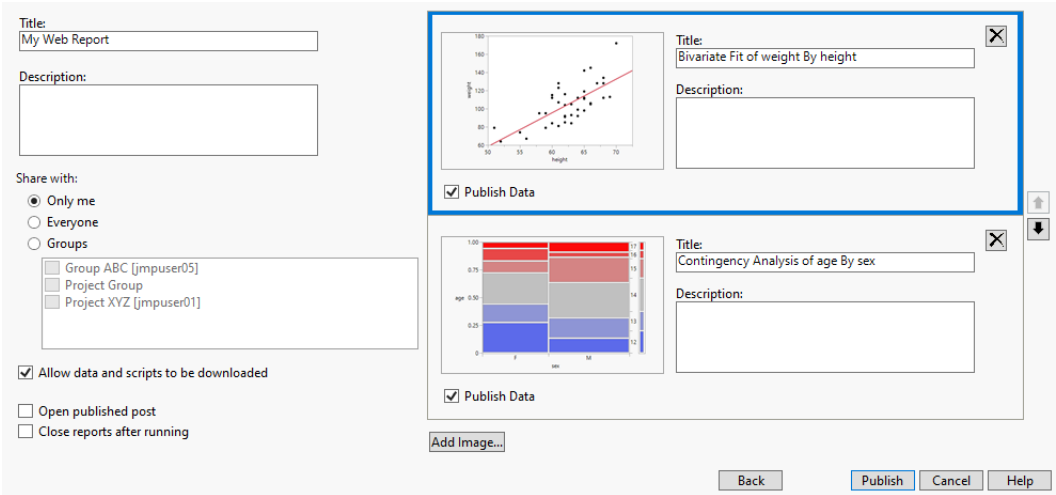
Figure 10.4 Select Reports Window



4. (JMP Live only) Next to **Publish to**, select a JMP Live server.
5. Select the open reports that you want to publish.
6. Choose your action and then click **Next**:
   - "Create a New Post"
   - "Add Reports to an Existing Folder"
   - "Replace an Existing Report in a Post" (if a post with the exact name exists)

Create a New Post

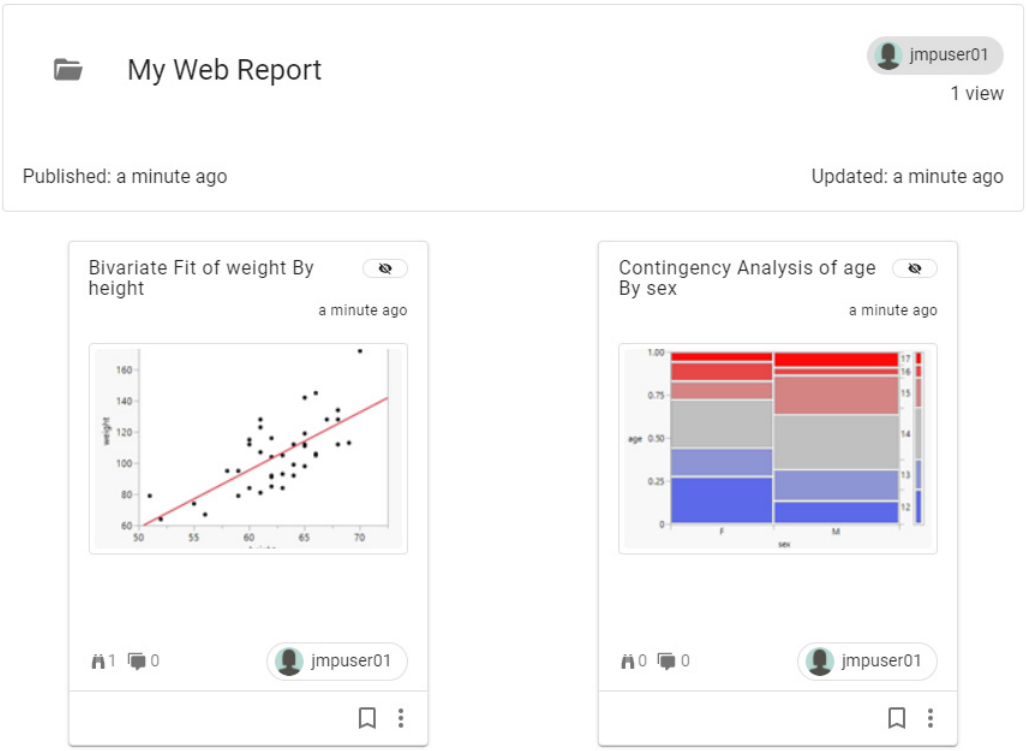
Figure 10.5 Create a New Post with Multiple Reports



Note: If you are publishing a single report, the window and options in Figure 10.5 are slightly different.

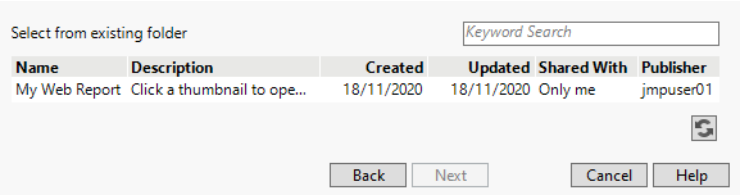
1. (Multiple posts only) Enter a title for the folder and an optional description.
2. (Optional) Update the title for each report and add descriptions.
3. Choose how to share the post:
   - **Only Me** keeps your post private and visible to you only.
   - **Everyone** shares your post with everyone on JMP Public or JMP Live.
   - **Groups** shares your post with groups that you select. Groups that you can publish to appear below this option.
4. Change additional options. See "Post Options".
5. Click **Publish**.
6. Click a link to open the folder or post or click **OK** to dismiss the window.

Figure 10.6 JMP Live Folder Containing Reports



Add Reports to an Existing Folder

Figure 10.7 Select an Existing Folder

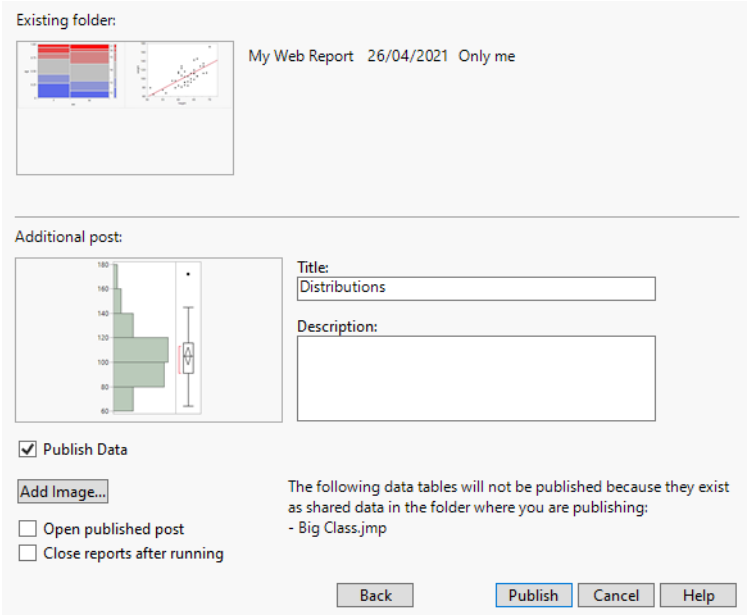


Note: You see only folders that you have permission to update. These are folders that you own or folders in a group that you are a member of.

1. Select a folder. Do one of the following:
   - Click a folder in the list and click **Next**.

– Search for a folder—In the Keyword Search box, enter a word that appears in the folder name or description and then click a folder. Click **Next**.

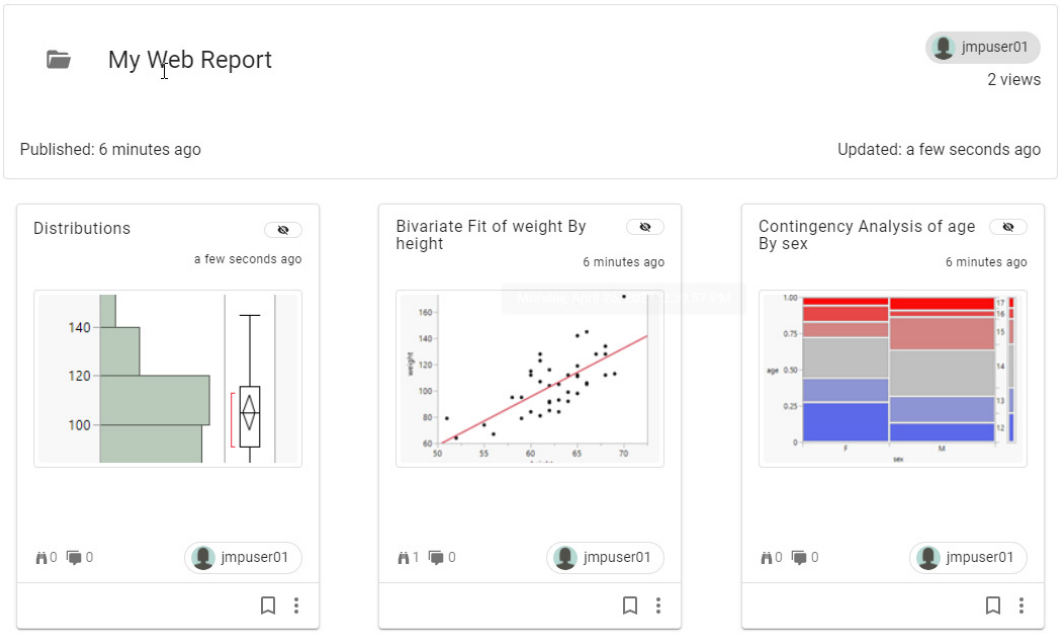
Figure 10.8 Add a Report to an Existing Folder



Note: If you are publishing multiple reports, the window and options in Figure 10.8 are slightly different.

2. (Optional) Update the title for each report and add descriptions.
3. Change additional options. See "Post Options".
4. Click **Publish**.
5. Click a link to open the folder or post or click **OK** to dismiss the window.

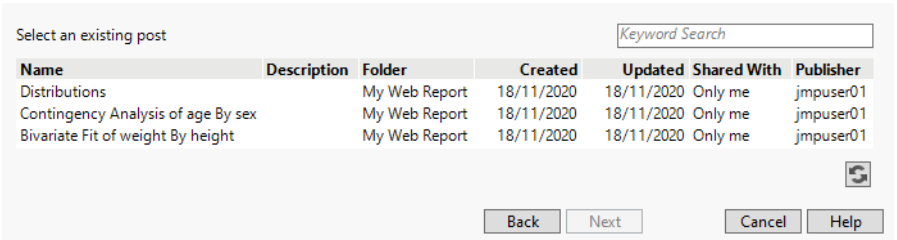
Figure 10.9 Report Added to Existing JMP Live Folder



Replace an Existing Report in a Post

Note: Republishing an entire folder is not possible in order to maintain existing comments and bookmarks on posts in the folder.

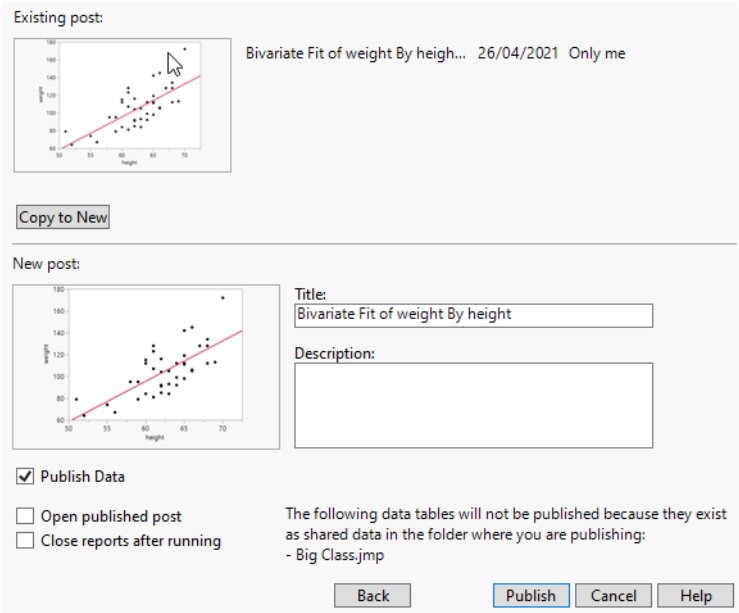
Figure 10.10 Select an Existing Post



Note: You see only posts that you have permission to update.

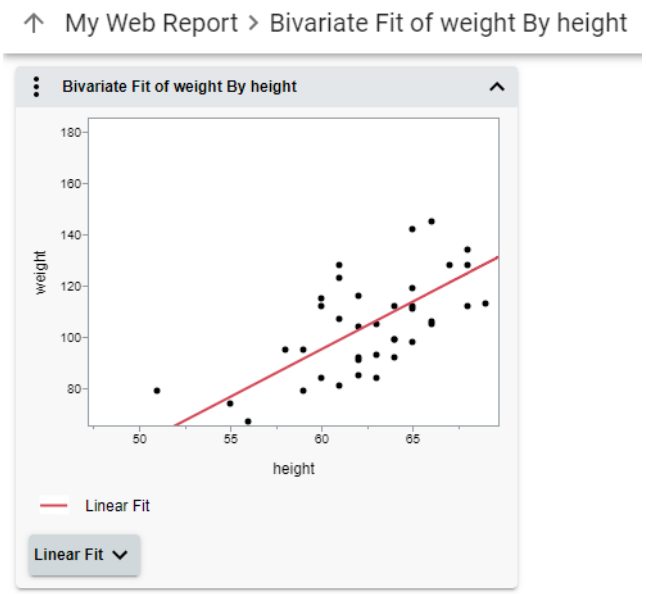
1. Select a post. Do one of the following:
   - Click a post in the list and click **Next**.
   - Search for a post—In the Keyword Search box, enter a word that appears in the post name or description and then click a post. Click **Next**.

Figure 10.11 Replace an Existing Post



2. Choose one of the following options:
   - To use the same title and description (if one exists) from the existing post, click **Copy to New**.
   - To use an updated title and description, enter a new title and description.
3. Change additional options. See "Post Options".
4. Click **Publish**.
5. Click a link to open the folder or post or click **OK** to dismiss the window.

Figure 10.12 Report Replaced in JMP Live



Post Options

Title Add a title for the web page. You can also change the report titles.

Description (Optional) Add a description to the web page that appears under the title. You can also add report descriptions that appear under the report titles.

Publish Data Select this for interactive reports. If you deselect this option, the reports are static. When selected, the data behind the report and a JSL script of the report are published.

Note: To avoid sharing sensitive data, save your results as a non-interactive web page. (For a single report, select **File > Export > HTML**. For multiple reports, select **File > Publish > Publish to File** and deselect **Publish Data**.)

Enable Warnings (Applicable for platforms that support control charts only.) Notifies interested parties when the post has control chart warnings. Interested parties include the publisher of the post and any groups the post is shared with that have warning notifications enabled.

Add Image Adds an image as an image-only report.

Allow data and scripts to be downloaded Allows data tables and scripts associated with the reports to be downloaded. If you do not select this option, in JMP Live there is no

download option next to the data table names and no JSL script to re-create the report. See "Note on Data Accessibility".

Open published web report Opens the web page in a browser once you click Publish.

Close reports after running Closes the reports in JMP once you publish the web report.

Delete icon Deletes a report.

Arrow icons Changes the order of reports.

JMP Live and JMP Public Help

For help with using JMP Live or JMP Public, click > **Help**.

Note on Data Accessibility

The data that is represented in your published reports will exist online in HTML5 files as well as in the JMP data table format.

Disabling the Allow data to be downloaded option prevents any user who has access to the JMP Live page from being able to download the data in the JMP data table format, but it does not prevent a user from being able to get the HTML file containing the data in XML format. Enabling the Allow data to be downloaded option allows a user who has access to the JMP Live page to get the data in either way. The JMP format is only useful to users who have access to the JMP Live page who also have a separate license of JMP software or a third-party software that knows how to read JMP files, but a user of the JMP Live page only needs a web browser to see the data in the HTML file.

Save Reports as PDF Files

You save reports as PDF files by selecting File > Save As > PDF File (*.pdf) (Windows) or File > Export > Image > PDF (macOS).

Windows

- You can control pagination features such as headers, footers, and margins when you export the file.
  - Select **File > Export**, select **PDF**, and click **Next**.
  - Name the file, click **Save**, and then change settings in the Page Setup window.
- You can also control pagination by selecting **File > Page Setup**. Note that you can click **Set as Default** to apply the settings to future page setups.
- You can also control pagination by selecting **File > Print Preview**.

macOS

- Select **File > Export > Image > PDF**. The resulting PDF file contains a single PDF image with no headers or footers.
- You can also select **File > Print** and then select **Save as PDF** from the PDF window. The resulting PDF file is paginated into multiple pages as needed, with headers and footers, and using the page size, orientation, scaling, and margins that you selected in the Print and Page Setup windows.

Windows and macOS

Through JSL, you can control pagination on both Windows and macOS. See *JSL Syntax Reference* for more information about `Set Page Setup`, `Set Print Headers`, `Set Print Footers`, and `Save PDF`.

Save Reports as PowerPoint Presentations

Create a presentation by saving JMP results as a Microsoft PowerPoint presentation (.pptx). Rearrange JMP content and edit text in PowerPoint after saving as a .pptx file. Sections of a JMP report are imported into PowerPoint differently. Report headings are imported as editable text boxes, and graphs are imported as images. Certain graphical elements, such as legends, can be imported as separate images. Images resize once opened in PowerPoint to fit the slide. Typically, each outline in the JMP report appears on separate PowerPoint slides. If an outline in the report is closed when saving as a .pptx file, it is excluded from the PowerPoint. Larger content can spread across multiple PowerPoint slides. Delete unwanted content from the PowerPoint file.

Note: On Windows, PowerPoint 2007 is the minimum version required to open .pptx files created in JMP. On macOS, at least PowerPoint 2011 is required.

1. In JMP, create the report.
2. Select **File > Export > Microsoft PowerPoint** and click **Next**.
3. Select a graphic file format.

   On Windows, EMF is the default format. On macOS, PDF is the default format.
4. (Optional) To avoid opening the file in the default presentation program after saving, deselect **Open the file after saving**.
5. Name and save the file (or export on macOS).

Tips:

- The native EMF graphics produced on Windows are not supported on macOS. The native PDF graphics produced on macOS are not supported on Windows. For cross-platform compatibility, change the default graphics file format by selecting **File > Preferences >**

Third Party Data (Windows) or **JMP > Preferences > Third Party Data** (macOS). Then, change the **Image Format for PowerPoint** to either PNG or JPEG.

- It is sometimes necessary to auto-resize column widths using PowerPoint. PowerPoint often sizes objects differently from JMP, which can lead to abnormally sized table columns.
- Drag the column separators in PowerPoint to adjust column widths for a nicer layout. You can also double-click a column border to decrease the amount of white space in the column.
- A default PowerPoint template is installed in the JMP directory's pptx folder. You can save a copy of this PowerPoint template and replace the default template with your own if you want to always use an alternate template when exporting. For example, you might want to create a German PowerPoint template that has German default fonts. See "Custom PowerPoint Templates".

Custom PowerPoint Templates

PowerPoint presentations use the default JMP template. You can use a custom template by overwriting the default template file. To do this, open JMPExportTemplate.pptx located in one of the following locations:

C:/Program Files/SAS/JMP/16/pptx/

C:/Program Files/SAS/JMPPro/16/pptx/

Then, choose your own PowerPoint template, and select **File > Save**. The next time you save JMP content as a PowerPoint, it opens with your custom template. Alternatively, create your custom template in PowerPoint first, and save it as JMPExportTemplate.pptx.

Notes:

- The custom PowerPoint template on macOS cannot be edited.
- To use the JMP default template later, make a duplicate copy on your machine before overwriting the file.

Also, you can use a custom template with a JSL script. For more information, see the *JSL Syntax Reference*.

Setting the Graphic DPI for Exported Graphics

On Windows, there are several ways to set the resolution (or DPI) for exported graphics:

- To change the default DPI for all graphics, set the DPI to in the Windows Specific preferences. The Default option uses the default setting of your operating system.
- The following script lets you set the default DPI to a different value. Change the number if you want a different DPI and then run the script.

```
Pref( Save Image DPI( 600 ) );
```

To verify the default DPI after you run the preceding script, run the following script:

```
Get Preferences( Save Image DPI );
```

The Windows Specific preferences and Save Report As window show Default as the DPI after you run this script. The setting applies until you change it. On the Save Report As window, select **Always use this setting** to save your choice across multiple JMP sessions.

- For a specific graphic, set the DPI in the Save Report As window.

On macOS, you cannot change the default DPI, which is 72.

Save a JMP Analysis as a Script

After completing an analysis and receiving a report, you might want to save the process that you used to arrive at the report as a JSL script. You can save the script inside the data table, inside the report, or in a separate script window. To do this, click the red triangle in the report title and select the Save Script option that you prefer. See "Save Script Menus in Reports".

Non-English versions of JMP can save scripts in either English or your local language. See "General JMP Preferences".

Save Parts of a Report in a Graphic Format

You can save part of an analysis report window as a graphic. You can save the selection in .png, .jpg, .gif, .eps, .emf, .pdf, .tiff, and .svg formats. On macOS, you can save the selection in .png, .jpg, .gif, .svg, .eps, .pdf, and .pptx formats.

To save a selection, follow these steps:

1. Click the selection tool ( ).
2. Highlight the area that you want to save.
3. On Windows, select **Edit > Save Selection As**. On macOS, press Control and select **File > Save Selection As**.

   The Save Selection As window appears (or the Save window on macOS).
4. Enter the filename for the graphic.

   On macOS, change the file extension to save the file in a format other than .png, the default format.

Note: If you use Save Selection As and create a PDF file from the selection, the PDF file has headers and footers. With JSL, you can control pagination on both Windows and macOS. See the *JSL Syntax Reference*.

5. Change options on Windows:
   - Select the graphics file format type to which you want to save the selection. See "Save and Share Reports".
   - Select the Image DPI Setting. 300 or higher is best for graphics that must be stretched or printed.
   - To keep your selected DPI Setting for the current session, select **Always use this setting**.
   - To avoid opening the file in the default graphics program after saving, deselect **Open the file after saving**.
   - Select **Select this filter the next time this window is invoked** to save your file type selection for the current session.

   Note: If you cannot see the Select this filter option, select a folder location for saving the graphic file.

6. Click **Save**.

Tips:

- You can change the default DPI setting selection in the Windows Specific preferences from 96 to 300 DPI. The following script lets you set the default DPI to a different value. Change the number if you want a different DPI and then run the script.

  ```
  Pref( Save Image DPI( 600 ) );
  ```

- On macOS, you cannot change the default DPI, which is 72.
- When you export a graph or report that contains a bitmap image to an .svg file, the image is embedded in the .svg file. The exported content is then easier to reuse in other documents.

Print Reports

To print the report in the active window, select **File > Print**. This command displays the standard window for printing. The appearance of the window depends on your operating system and printer driver.

To insert a page break for printing purposes, follow these steps:

1. Right-click the disclosure icon in the report window.
2. Select **Edit > Page Break**.

Copy and Paste Reports

When you need to use JMP reports or data tables in another program, you can paste or drag parts of a report or table into another program. For example, you might want to send a colleague a copy of a JMP report in a Microsoft Word file.

1. Click the selection tool .
2. Click and drag (or press Shift and click) to select items in a report window or data table. Clicking near the edge of the report window selects the entire report.
3. Click the selected items and drag them from JMP to the other program. Or, copy the selected items in JMP and paste them into the other program. When you paste an element into another application, the format used depends on the application into which you paste. If you use the applications' Paste Special command, you can select a format to use, such as text (.rtf), unformatted text (.txt), or bitmap (.bmp).

Notes:

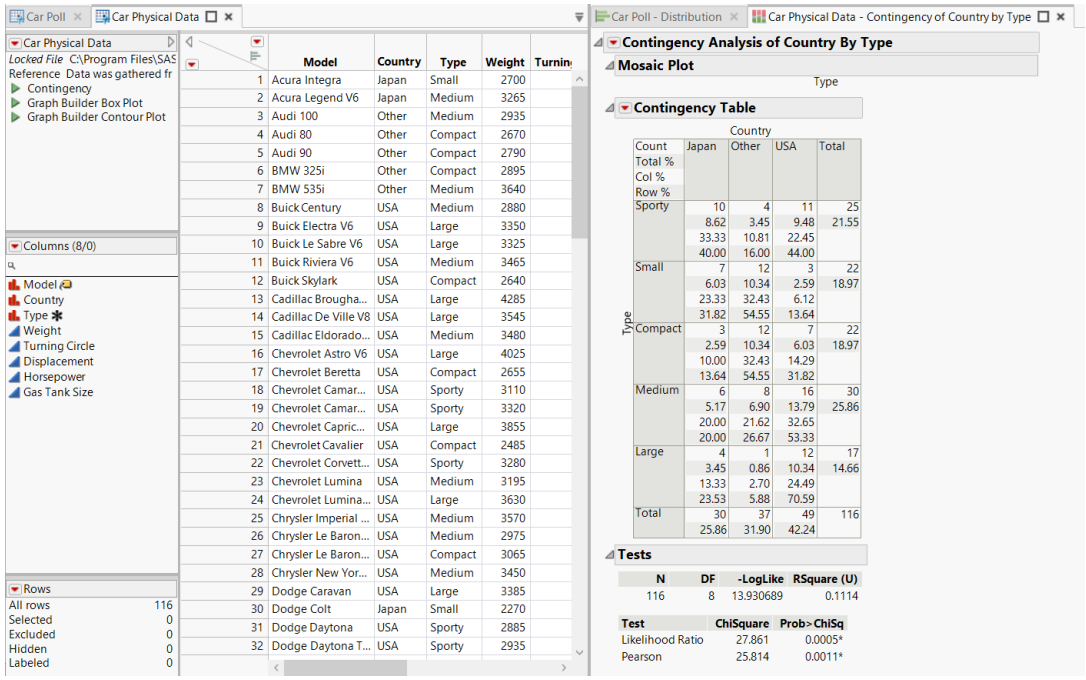
- To copy all text (no graphs) from the active report window as unformatted text, select **Edit > Copy As Text**. On macOS, select **Paste > Special** in Microsoft Word or **Paste and Match Style** in other macOS apps.
- To copy a graph, which includes labels and axes, right-click the graph and select **Edit > Copy Graph**.
- To copy anything other than a graph, right-click and select **Edit > Copy Picture**. For example, to copy a report, right-click the bar containing the report title and select the option.
- To copy text from a text box onto the clipboard, right-click the text and select **Copy Text**.
- To copy text from one column onto the clipboard, right-click the column and select **Copy Column**.
- To copy and entire table onto the clipboard, right-click the table and select **Copy Table**.
- The Reports preference **Transparent background in report PNG images** determines whether reports that are saved as PNG images have transparent backgrounds. The preference is deselected by default.

Work with Projects

With JMP projects, you can do the following:

- Explore your data more efficiently with a single, tabbed JMP window
- Quickly save and reopen a set of related files and reports
- Easily share your work by embedding your tables and scripts in a self-contained project file

Figure 10.13 Project File with Data Tables and Reports



This section contains the following information:

- "Create a New Project"
- "Open Files in a Project"
- "Rearrange Files in Projects"
- "Save a Project"
- "Project Workspace"
- "Project Bookmarks"
- "Project Contents"
- "Recent Files in Projects"
- "Project Log"
- "Move Files into Projects"
- "Write a Project On Open Script"
- "Example of Creating a New Project"

Create a New Project

To create a new, empty JMP project, select **File** > **New** > **Project** (Windows) or **File** > **New** > **New Project** (macOS).

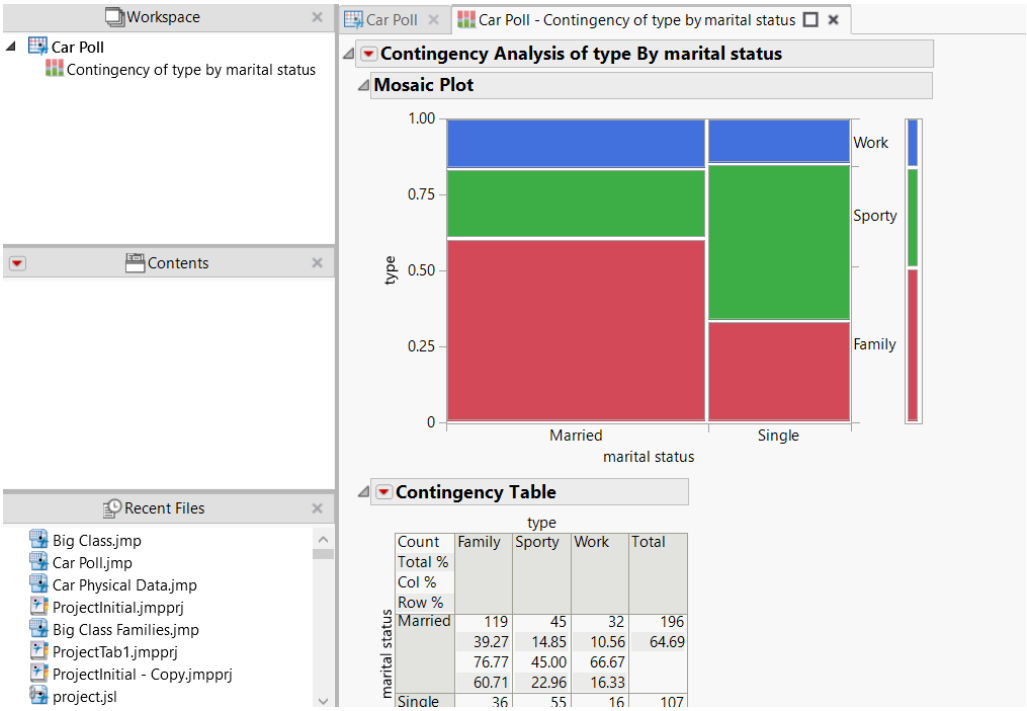
Open Files in a Project

In a JMP project, you can open data tables and then run analyses on the data. Each data table and analysis report opens in a new tab.

1. From a project window, select **File** > **Open** and navigate to the data tables that you want in the project.
2. From a data table, run an analysis.

If a data table is updated on your computer, any associated reports in your project update when you reopen the project.

Figure 10.14 Initial Project



Tip: You can use JMP keyboard shortcuts in projects, such as Ctrl+S to save a file, or Ctrl+W to close the active window pane. For a full list, select **Help** > **Quick Reference Card**.

Rearrange Files in Projects

In a JMP project, data tables and reports appear in individual tabs. You can rearrange tabs by dragging them into a dock zone.

Tip: To undo or redo a docked item, select **Project** > **Undo Layout** or **Project** > **Redo Layout**. To return all tables and reports to individual tabs, select **Project** > **Reset Layout**.

Example of Rearranging Project Files

1. Select **File** > **New** > **Project** (Windows) or **File** > **New** > **New Project** (macOS).
2. Select **Help** > **Sample Data Library** and open Car Physical Data.jmp and Car Poll.jmp.
3. In the Car Poll.jmp data table, run the Distribution script.
4. In the Car Physical Data.jmp data table, run the Contingency script.
5. Drag the **Car Poll - Distribution** report tab to the right and drop it into the *Dock right* zone.

The Distribution report pane appears at the right of the project window. The report is docked and stays visible when you switch between tabs.

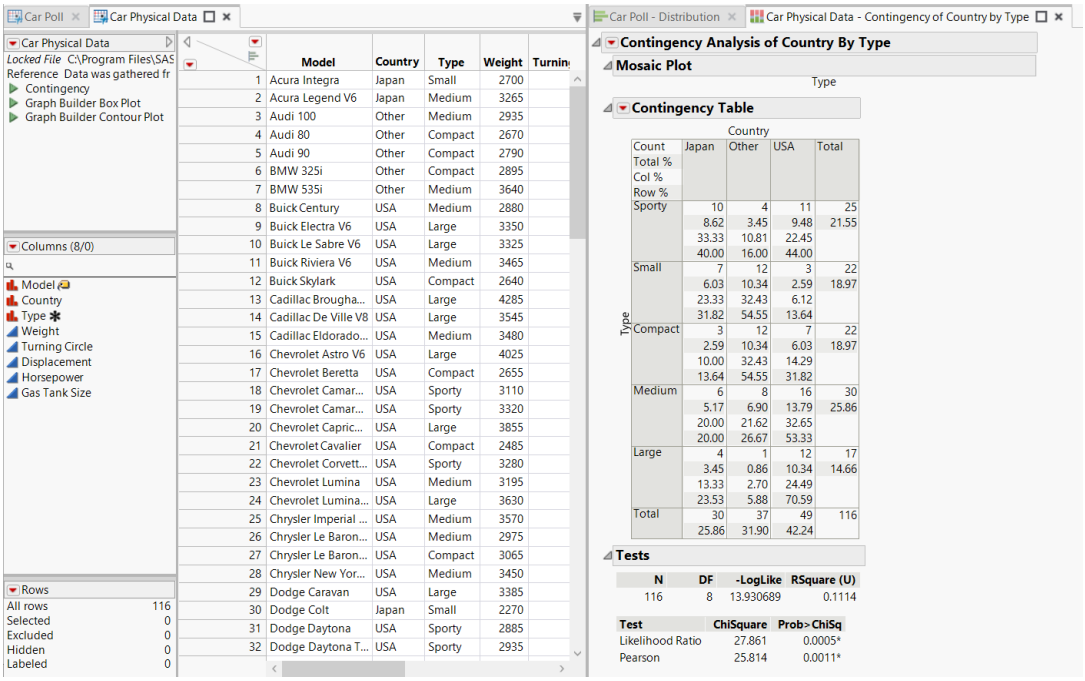
6. Drag the **Car Physical Data - Contingency** tab to the middle of the **Car Poll - Distribution** report and drop it into the *Dock tab* zone.

Tip: You can adjust the size of the data table window to fully show both reports.

7. Click the x to the right of Workspace, Contents, and Recent Files in the left column.

   You can show these panes later by selecting the options in the Project menu.

Figure 10.15 Reports and Data Tables Grouped into Tabs



The Distribution and Contingency reports are now tabbed together, as are the two data tables.

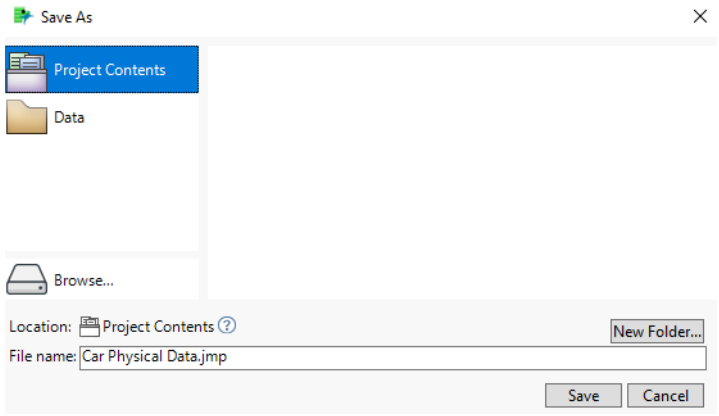
Save a Project

If you want to share, distribute, or archive your project, create a self-contained project by saving all your data tables and scripts to the project contents. This embeds all project files and folders into a single project file that you can share with other JMP users.

1. (Optional) To save the project as a self-contained project, save each data table and script in the project by selecting **File** > **Save As** and then select **Project Contents**. You can click **New Folder** to create a folder in the project contents.

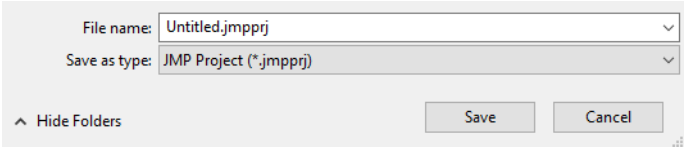
Note: You do not need to save reports as these are automatically saved to the project.

Figure 10.16 Save a Data Table to the Project Contents



2. Select **File** > **Save Project** to save the project file.

Figure 10.17 Save a Project File



Project Workspace

In a JMP project, you can see all currently open files in the Workspace tool pane. Reports appear under the corresponding data table. The active data table appears in bold.

Tip: To hide or show the Workspace tool pane, select **Project** > **Show Workspace**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

In the Workspace tool pane, you can do the following:

- Double-click a file to make it active.
- Right-click a file to close it, hide it (removes the tab and dims the file name), bookmark it, and more.

Project Bookmarks

In a JMP project, you can create a shortcut to a file or folder in the Bookmarks tool pane. You can also organize your bookmarked files and folders by creating a bookmark group.

Tip: To hide or show the Bookmarks tool pane, select **Project** > **Show Bookmarks**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

Bookmark an open file in the project

- Right-click a tab and select **Bookmark**.
- To bookmark all open files, right-click a tab and select **Bookmark All**.

Bookmark a file or folder on your computer

Drag a file or folder from your computer into the Bookmarks tool pane, or click the Bookmarks red triangle menu and select **Add Files** or **Add Folder**.

Note: If you add or remove files from a bookmarked folder on your computer, the folder contents automatically update in the project.

Open a bookmarked file

In the Bookmarks tool pane, double-click a file.

Remove a bookmark

In the Bookmarks tool pane, right-click a file and select **Remove Bookmark**.

Create a bookmark group

1. Click the Bookmarks red triangle menu and select **New Group**.
2. Name the group and click **OK**.
3. Drag new or existing bookmarks into the group, or in the Bookmarks tool pane, right-click the group and select **Add Files** or **Add Folder**.

Project Contents

In a self-contained JMP project, any files you save to the project contents appear in the Contents tool pane.

Tip: To hide or show the Contents tool pane, select **Project** > **Show Contents**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

Open a file from the project contents

In the Contents tool pane, double-click a file.

Create a folder in the project contents

1. Click the Contents red triangle menu and select **New Folder**.
2. Name the folder and click **OK**.

Move files into folders

In the Contents tool pane, drag a file into a folder.

Copy a file or folder from your computer into the project contents

1. Click the Contents red triangle menu and select **Copy Files into Project** or **Copy Folder into Project**.
2. Navigate to the file and click **Open**, or navigate to the folder and click **Select**.

Rename a file in the project contents

In the Contents tool pane, right-click a file and select **Rename**.

Delete a file in the project contents

In the Contents tool pane, right-click a file and select **Delete**.

Recent Files in Projects

In a JMP Project, you can open a file from the Recent Files tool pane.

Tip: To hide or show the Recent Files tool pane, select **Project** > **Show Recent Files**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.

Open a file

In the Recent Files tool pane, double-click on a file or drag it into a project.

Note: Open files that are saved to the project contents do not appear in the Recent Files tool pane.

Project Log

In a JMP Project, you can see log messages in the Log tool pane.

Tips:

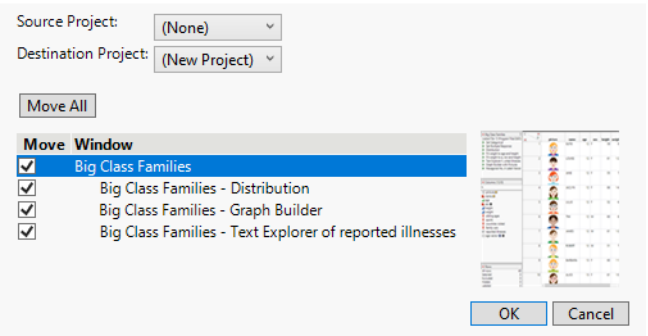
- To hide or show the Log tool pane, select **Project** > **Show Log**. To specify which tool panes appear by default when you create a project, select **File** > **Preferences** > **Projects** and choose the initial tool panes.
- By default, log messages that occur in projects appear only in the project log, not in the main JMP log. To change this setting, select **File** > **Preferences** > **Projects** and update **Use Project Log Pane** to **If Open** or **Never**.

Move Files into Projects

In JMP, you can move files into a project, out of a project, or between projects.

1. Select **Help** > **Sample Data Library** and open Big Class Families.jmp.
2. Run the Distribution, Text Explorer, and Graph Builder scripts.
3. From any window, select **Window** > **Move to/from Project**.
4. Leave the Source Project as **(None)** to see files that are not open in any project.
5. Leave the Destination Project as **(New Project)** to move the selected files into a newly created project.
6. Select the check box next to Big Class Families.jmp. The graphs associated with the data table are also selected.

Figure 10.18 Move Windows To/From Project



7. Click **OK**. A new project is created with the selected table and reports.

Write a Project On Open Script

In the Project On Open Script box, you can add a JSL script that will run each time this specific project is opened. For example, you can create a script that creates or modifies reports, prompts the user for information, or writes usage information to the log window.

1. Select **Project** > **Project Settings**.
2. Paste the JSL script and click **OK**.

Tips:

- See the *Scripting Guide*, which explains how to create a start-up script when *any* project is opened (instead of a specific project).
- To use an existing project as a template for new projects, specify the existing project as a new project template under **File** > **Preferences** > **Projects**.

Example of Creating a New Project

In this example, you create a project, import data, generate an analysis and dock the report in the project window, create a subset table, and then save, close, and reopen the project.

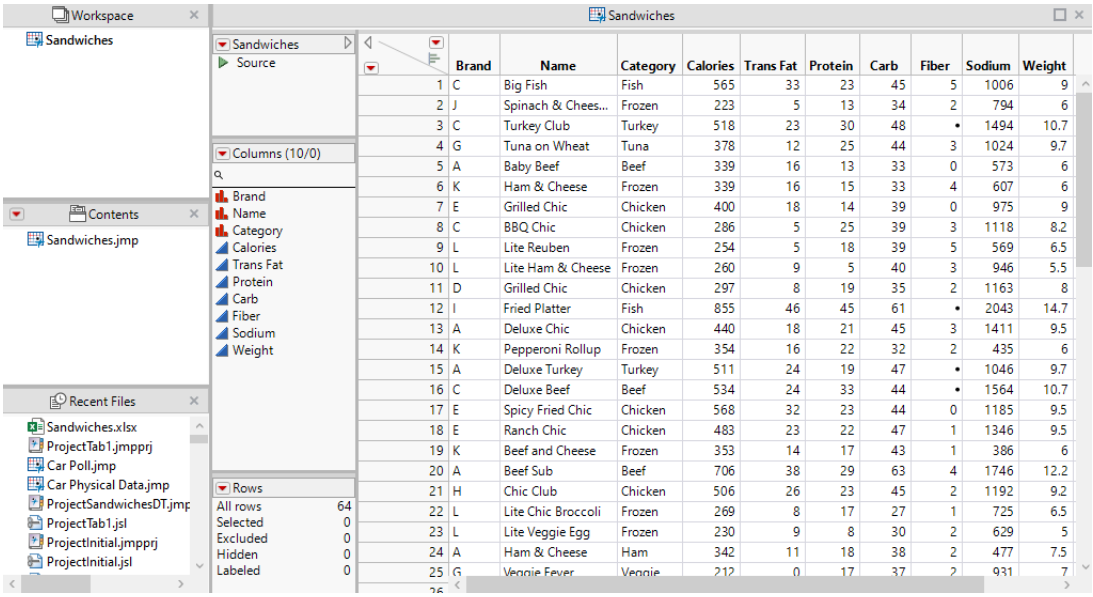
1. Select **File** > **New** > **Project** (Windows) or **File** > **New** > **New Project** (macOS).
2. From the project window, select **File** > **Open**.
3. Open the sandwiches.xlsx file, located here by default:

   C:/Program Files/SAS/JMP/16/Samples/Import Data

   Tip: At the bottom, you might need to change **All JMP Files** to **Excel Files**.

4. Click **Import**.
5. Select **File** > **Save**.
6. Make sure **Project Contents** is selected.
7. Change the file name to Sandwiches.jmp and click **Save**.

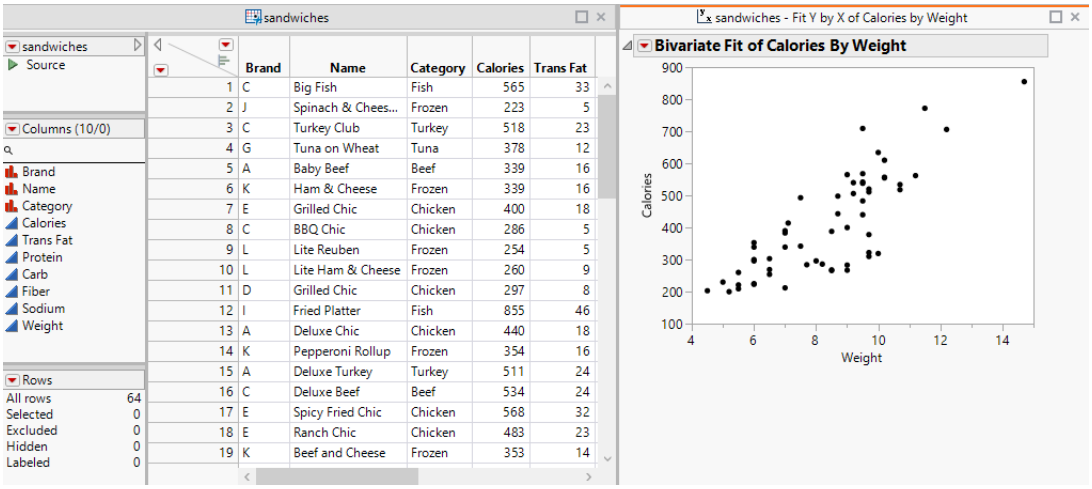
Figure 10.19 Project with Sandwiches Data Imported



8. Select **Analyze** > **Fit Y by X**.
9. Select Calories and click **Y, Response**.
10. Select Weight and click **X, Factor**.
11. Click **OK**.
12. Drag the **Sandwiches - Fit Y by X** tab to the right and drop it into the *Dock right* zone.

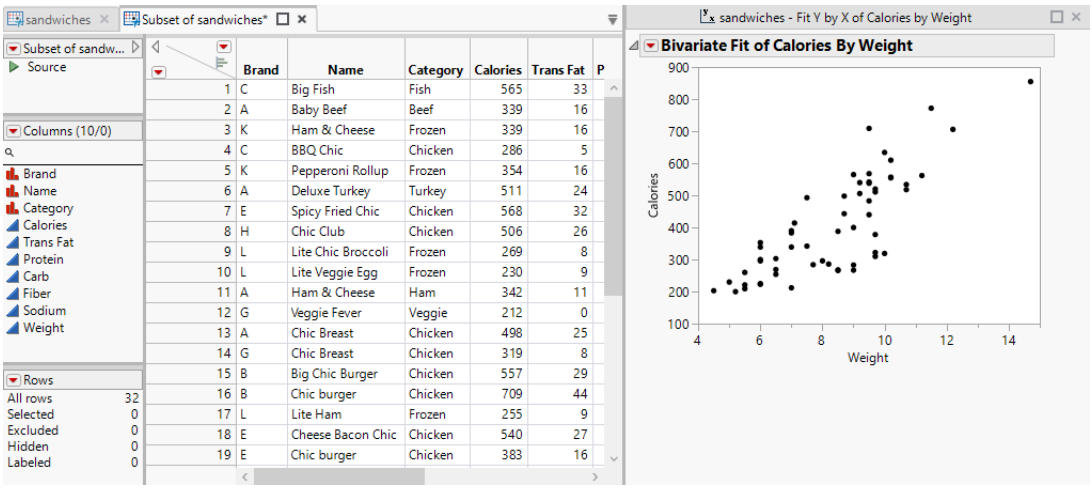
Tip: To show the entire report, you can drag the line between the data table and the report to the left.

Figure 10.20 Fit Y by X (Bivariate) Report Docked at Right



13. Select **Tables** > **Subset**.
14. Under Rows, select **Random: sampling rate 0.5**.
15. Click **OK**.

Figure 10.21 Project with Unsaved Subset Table



16. Select **File** > **Save Project**.
17. Navigate to the folder where you want to save your project, name the project file, and click **Save**.
18. Close the project.
19. Select **File** > **Open** and open your project file.

Notice that the subset table that you did not save has been saved automatically to the project file.

Create Journals

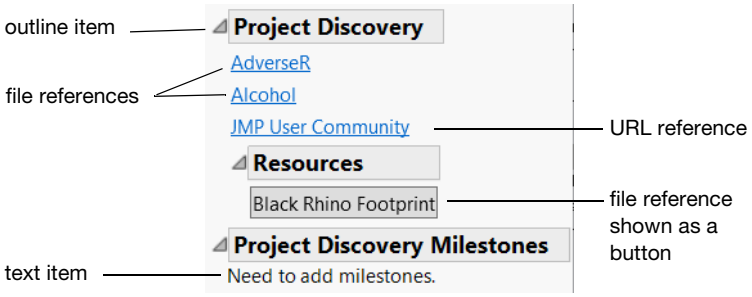
Create a JMP journal when you want to store results and present them. One of the benefits of journaling is that you can capture an analysis and not depend on external files. Another benefit is the ability to copy and paste specific reports or graphs into the journal. This enables you to create a custom report to present to other users.

There are two types of presentations that you can create using journals:

- A static presentation embeds the output of one or more JMP reports, fixed at a moment in time. The output is similar to what you get when you save output as a PDF file. There is no data table, points cannot be selected, colored, hidden, excluded, and so on. Only cosmetic changes (fonts, axes, sizes) can be made. The benefit is that there is no dependency on external files.
- A dynamic presentation is built from outlines containing text and buttons (or links) that help you organize, hide, or reveal a set of data tables and reports. This type of journal is dynamic because you can open reports and tables and interact with them, selecting points, subsetting data, making more graphs, or rerun an analysis.

Figure 10.22 shows some of the items that you can add to a journal.

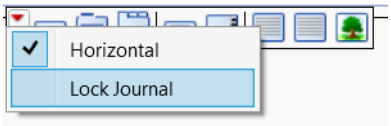
Figure 10.22 Example of a Journal



Notes:

- When you're finished creating the journal, select **Edit > Lock Journal**. To unlock the journal for editing, select **Edit** and deselect **Lock Journal**.

Figure 10.23 Lock a Journal



- You can save journals to other formats, such as Microsoft Word, HTML, PDF, and so on. See "Save Journals in Other Formats".
- Journals support GZ compression to reduce file size. See "General JMP Preferences". This can be particularly helpful if large data tables are embedded in the journal.
- Consider setting the *Autosave timeout* value in the General preferences to automatically save open journals at the specified number of minutes. This autosave value also applies to data tables, scripts, projects, and reports.
- A toolbar appears at the top of the journal that enables you to insert objects in the journal, lock the journal, and change its orientation (Figure 10.23). When you select or deselect Horizontal from the red triangle menu, this setting is not saved in the journal. The setting is only for the current session.

Example of Making a Journal for a Presentation

Most people use a presentation application like PowerPoint to give presentations. With JMP journals, you can avoid using a presentation application: all your bullet points can be combined with live links and buttons to help automate the analyses that you want to show. Interactivity is also maintained in reports.

Follow this example to create a presentation using a journal.

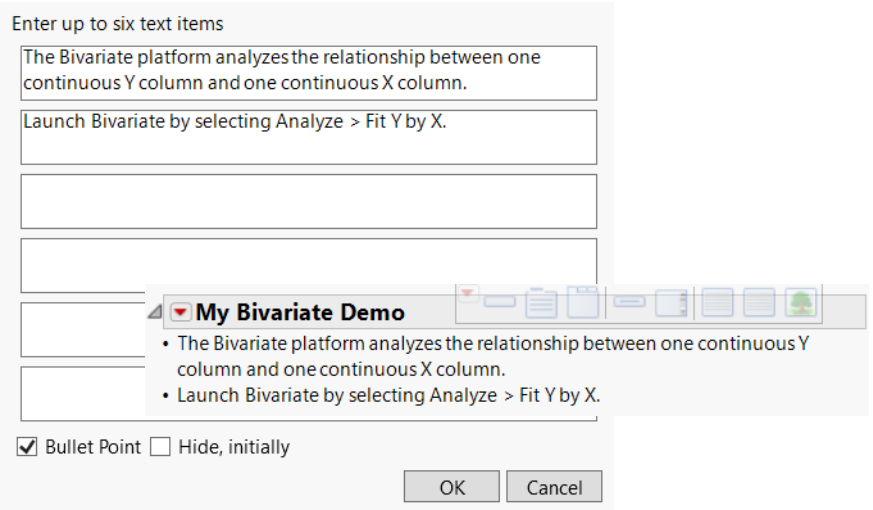
1. Select **File > New > Journal**.
2. Right-click in the journal. Start your outline by selecting **Add Outline Item**.
3. Enter the title of the presentation "My Bivariate Demo."

Figure 10.24 Add an Outline Item

A presentation should have outline nodes. The nodes are nested, opened, and closed in sequence, as you give the presentation.

4. To add bullet points into the outline, click the red triangle in the My Bivariate Demo title bar and select **Add Text Item**.
5. Enter the text shown in Figure 10.25.

Figure 10.25 Add Text Item



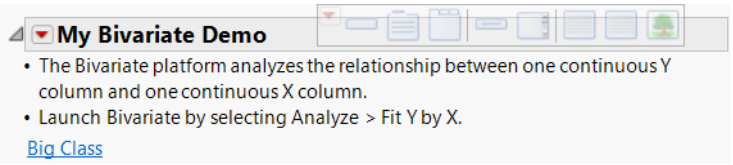
6. Select the **Bullet Point** check box and click **OK**.

> **Tip:** A horizontal flashing line appears at the top of the journal. Use the arrow keys to move the line to where you want to insert an object.

Now add a window reference as a link. These links lets you open a file with one click during the presentation.

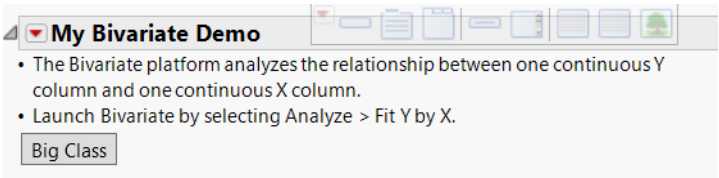
7. Select **Help > Sample Data Library** and open Big Class.jmp.
8. Click the red triangle in the My Bivariate Demo title bar and select **Add Window Reference**.
9. Select Big Class and click **OK**.

Figure 10.26 Add a Window Reference



10. Click the window reference to verify the link.
11. Display the window reference as a button by right-clicking the link and selecting **Style > Normal**.

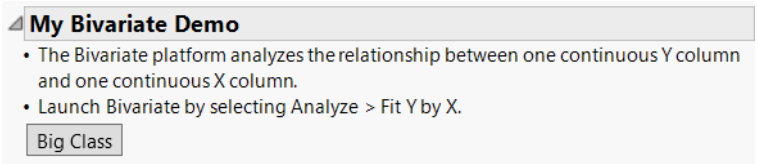
Figure 10.27 Change the Link to a Button



12. Select **Edit > Lock Journal** to see the final journal.

    When you lock the journal, the journal toolbar and insertion cursor no longer appear. And when you select Save Script > To Journal from a platform, the journal is not appended to the open locked journal.

Figure 10.28 Final Journal



Tip: You can also add objects to the journal by dragging them from the journal toolbar or putting your cursor where you want the object and double-clicking the object in the journal toolbar. The toolbar is available when you place your cursor over it.

Create New Journals

1. Close all open journals, and do one of the following:
   - To create an empty journal, select **File > New > Journal**. Or, from the JMP Starter window, select **New** in the Journal section.
   - To create a journal from an entire report or data table, select **Edit > Journal**.
   - To create a journal from a specific report or text in a report, click and drag the selection ( ) tool to select adjacent items in a report or data table. To select discontinuous items, press Shift and click the items with the selection ( ) tool. Then select **Edit > Journal**.
   - To create a journal from specific text in a data table, select the text, and then select **Edit > Journal**.
   - To create a journal from most graphs, right-click the graphic, and then select **Edit > Journal**. (Not available for surface plots and 3-D scatterplots.)

Tip: If you prefer keyboard shortcuts, press Ctrl+J (Windows) or Command+J (macOS) to create the new journal.

2. Select **File > Save** and save the journal in .jrn format.

Manage Links in Shared Journals

When you insert a link to a data table on your computer, the entire path to that data table is embedded in the journal. To share the journal or move it to another computer, store the linked files in the same directory as the journal or on a shared network drive. Because the files are in the same directory, the full pathname is unnecessary. Then the links are intact no matter where the journal is opened.

Tips:

- Zip the journal and supporting files so that another user can extract them to a directory. If the files remain in that directory, the links work.
- If you create a link to a sample data table, consider using the $SAMPLE_DATA variable in the path. Likewise, you can use $DESKTOP if the files are on your desktop or $DOCUMENTS for your personal documents folder.
- You can edit several links at once within an outline. There is also an option to replace strings in the links. Right-click the disclosure icon next to the outline, and then select **Append Item > Edit Links**.

To create portable links in a journal

1. Put the data tables and other files that you want to reference in the same folder.
2. Save the journal in that folder.
3. Right-click at the bottom of the journal, select **Add File Reference**, select the file, and then click **Open**.
4. Right-click the link in the journal and select **Edit > Set Script**.

   Notice the full file path.

   ```
   Open("C:\Users\shlori\journals\San Francisco Traffic.jmp")
   ```
5. Delete everything in the string except for the file name.

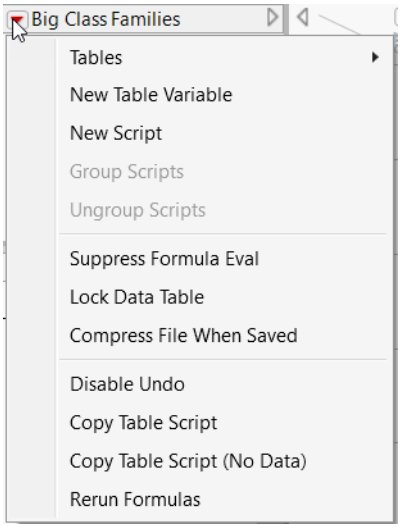
   ```
   Open("San Francisco Traffic.jmp")
   ```

To embed a data table in a journal

To avoid linking to an external data table, copy the data table into a script and place it in the journal. The advantage is that other users always have access to your data, and you do not need to edit links. The disadvantage is that, if you want to edit the data, you must edit the script.

1. In the data table, click the red triangle next to the data table name and select **Copy Table Script**.

Figure 10.29 Copying a Table Script



2. In the journal, right-click and select **Add Script Button**.
3. Enter a script name in the Name box.
4. Paste the table script into the Script box and click **OK**.
5. Right-click the link and deselect **Underline Style** display the link as a button.

Add and Edit Outlines to Journals

You can group text and links in an outline and then click the gray disclosure icon to expand and collapse the group.

- To add an outline, right-click in the journal, and then select **Append Item > Add Outline Item**. You can also add nested outline levels to outline levels that you have created. (When you click the red triangle menu of a graph that you have journaled, the **Add Outline Item** option is unavailable.) Click the outline's red triangle and select **Add Outline Item**.

- To edit the outline heading, double-click the heading, enter the new name, and press Enter.
- To add text or links within an outline, click the outline's red triangle and select an option described in "Right-Click Options in Journals". Note that these options are not available when you click a scripted item's red triangle.
- To add text or links outside an outline, right-click the journal and then select one of the options described in "Right-Click Options in Journals".
- To edit a link, right-click the link, select **Edit > Set Script**, edit the path, and then click **OK**.
- To edit the name of the link displayed in the journal, right-click the link, select **Set Button Name**, enter the new name, and then click **OK**.

Right-Click Options in Journals

Add Outline Item Groups text and links into collapsible and expandable lists.

Add Text Item Enter up to six paragraphs with the option to add a bullet or hide the text. To unhide, right-click the hidden text and deselect **Hide**.

Note: To add text outside an outline level, you can also double-click at the end of the journal and enter text in the box that appears. Click outside the box to exit the editing mode.

Add Window Reference Creates a link to a window already opened in JMP.

Add File Reference Creates a link to a file on your computer.

Add Directory of Files Creates links to all files (including non JMP files) in the selected directory.

Add Open Files Creates outline levels and links to open files in JMP (except for the log). For example, a link to the open Big Class.jmp data table and chart is displayed below a collapsible heading named Big Class. Files such as journals, scripts, and tutorials are grouped below a collapsible heading called *Other Files*.

Add URL Reference Creates a link to files that are delivered through an internet protocol such as HTTP, FTP, or FILE://.

Add Script Button Creates a link to a JSL script. Right-click the link and deselect **Underline Style** to display the link as a button.

Add Script Box Adds a box in which you type a script.

Lock Hides the journal toolbar and insertion cursor. If you select Save Script > To Journal from a platform, the results are not appended to a locked journal

Edit Links Enables you to rename file links (for example, to remove a hardcoded path). An option to test links is also provided. If the path is valid, nothing happens when you click Test Links. If it is not valid, an error message appears in the log. To update an invalid link, click **Replace**.

Size to Fit Resizes the window to ensure that button names are fully displayed.

Control the Display of Outline Levels in Journals

You can instantly control which outline levels are displayed (for example, closing all sublevels below the selected level). Click the red triangle for that level, select **OutlineBox**, and then select one of the Open or Close commands. For more information about the Open and Close commands, see "Display Options for JMP Reports".

Tip: On Windows, show the OutlineBox options by pressing Alt and right-clicking the outline level's gray disclosure icon.

Add a Graph or Graphic to Journals

PNG, GIF, JPG, BMP, and TIF files are supported in journals. Do one of the following to add a graphic to a journal:

- Copy a graph from within JMP or a graphic from another application and then select **Edit > Paste**. JMP places the graphic at the end of the journal.
- Drag and drop the graph or graphic from another window to the journal. (See "Paste a Background Image into a JMP Graph".) A blue line indicates where you can drop the graphic.

Tip: Consider resizing the graphic in an external program. Then you can insert the graphic at 100% resolution in the journal without dragging to resize the graphic.

Customize Items in Journals

- To click and drag journal items to different positions, select the selection tool ( ), select the item, and then drag the item to the new position. A line indicates where you can drag the item.
- To adjust text wrapping, right-click the text and select **Set Wrap**. Enter the number of desired pixels per line and click **OK**. To apply the wrap to all text items, select **Set Wrap,** select **Extend this to other text boxes**, and then click **OK**.
- To modify a plot axis, double-click or drag the axis. See "Customize Axes and Axis Labels in Graphs".

- To add text or shapes anywhere in the report, use the drawing tools (Annotate, Line, Polygon, and Simple Shape). For more information about the drawing tools, see "Add Annotations and Shapes to JMP Reports and Graphs".
- To resize plots and graphs, do one of the following:
  - Click and drag the edge.
  - Right-click the plot or graph, and then select **Size/Scale > Frame Size**. See "Resize Graphs".
- To modify the format of a numeric column, you edit the Properties pane for the display box. See "Modify the Format of Numeric Columns".

Prevent Modifications to Journals

When a journal is opened, and you then journal another report or data table, the second report or data table is added to the end of the open journal.

To prevent modifications to a journal, right-click the blank area at the bottom of the journal and select **Lock**. Additional reports are not added to the journal (as described in the following section). JMP instead finds an open unlocked journal or creates a new journal when you select the **Edit > Journal** command again.

To unlock the journal, right-click the blank area at the bottom of the journal and deselect **Lock**.

Append Reports to Journals

To append other reports to a report that you already journaled, open the journal and select **Edit > Journal**. If an area of an analysis window is selected, **Edit > Journal** saves only the selected area instead of the entire window.

Notes:

- The journal window has the same functions as the report window: you can click icons, click and drag, and right-click to access menus.
- When a report is journaled, the journaled copy is no longer connected to the data table. To maintain that connection, save the report as JMP report and insert a link to it in the journal.

Save Journals in Other Formats

You can save a journal in another format (such as HTML or PDF). When you save the journal as HTML, buttons and links are active when the file is viewed in a web browser.

Note: The header and footer defined in your printer setup appears in the PDF file. However, you can set the left, center, and right header and footer in a JSL script. Search for the `Set Print Footers` message in the JMP Scripting Index. (Select **Help > Scripting Index** in JMP.)

Windows

1. Select **File > Export**.
2. Select the format. See "Save and Share Reports".
3. Click **Next**.
4. Name and save the file.

macOS

1. Select **File > Export**.
2. Select the format. See "Save and Share Reports".
3. Click **Next**.
4. Name and export the file.

Delete Items from Journals

1. Click the selection tool ( ).
2. Select the item that you want to delete and press Delete.

Save JMP Sessions

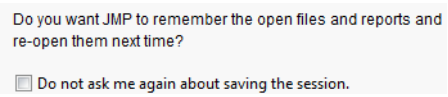
Each time you use JMP is called a *session*. A *saved session* is a JSL script that re-opens documents and re-runs analyses to restore JMP's state when the session script was saved. A saved session can help get you back to a previous state without having to manually re-open files and re-run analyses.

If you are an advanced user, it is important to understand what session information is preserved in a saved session. Any documents (such as data tables, scripts, and journals) that have been saved are re-opened. JMP windows that support script saving are re-run (equivalent to clicking the red triangle in a report and selecting **Redo > Redo Analysis**). Side effects of running scripts, such as global variable values or custom windows, are not saved. The state of highly interactive analyses is also not saved.

Save Sessions upon Exiting JMP

The most common use of saved sessions is to save the state when JMP exits so that it can be restored when JMP restarts. By default, JMP asks whether you would like to save the state of your session each time you exit the program (Figure 10.30). This enables you to quit JMP, and then return to it later without having to open the files with which you were previously working.

Figure 10.30 Saving Session upon Exiting



To always save the session upon exiting, select **Do not ask me again about saving the session** and click **Yes** or **No**. This option also prevents JMP from saving the session upon exiting,

You can change this option later in Preferences:

1. Choose **File > Preferences**.
2. Select the **General** icon.
3. Next to the **Save the session when exiting** option, choose when you want JMP to save the session. Always, never, and prompt whether you want to save are the options.

Save Sessions Manually

You can also save a session to a location of your choosing and continue working, so you can restore the saved state whenever you like. Manually saving sessions is useful when you want more control of session saving and restoring. This option is especially helpful when you want to maintain multiple independent session states. Each state has a different set of files and analyses.

To create a script of a JMP session

1. Select **File > Save Session Script**.
2. Enter the name of your script in the window and click **Save**.

Another way of manually saving a session is by creating a journal of each session. A journal can be a notebook-style or project-style file. With it, you can collect references to files in a project, develop presentation launch pads, document projects, and store many scripts in one place. See “Create Journals”.

To create a journal of a JMP session

1. Open the files that you would like to include in the journal.

2. Select **File > New > Journal**. Or, to append your open files to an existing journal, open that journal.
3. Right-click in the empty journal and select **Add All Open Files**.

Save a Log Window

Selecting **View > Log** displays a window that monitors JMP activities, such as JSL statements as they execute and script errors. You might also encounter instances where running a formula, matrix, or another operation writes information of interest to the log window.

To save a log, follow these steps:

1. Open the Log window.
2. Select **File > Save As**.
3. (Windows only) To open the log automatically when text is added to it, select **File > Preferences > Windows Specific**. In the **Open the JMP Log window** list, select **whenever text is added**.

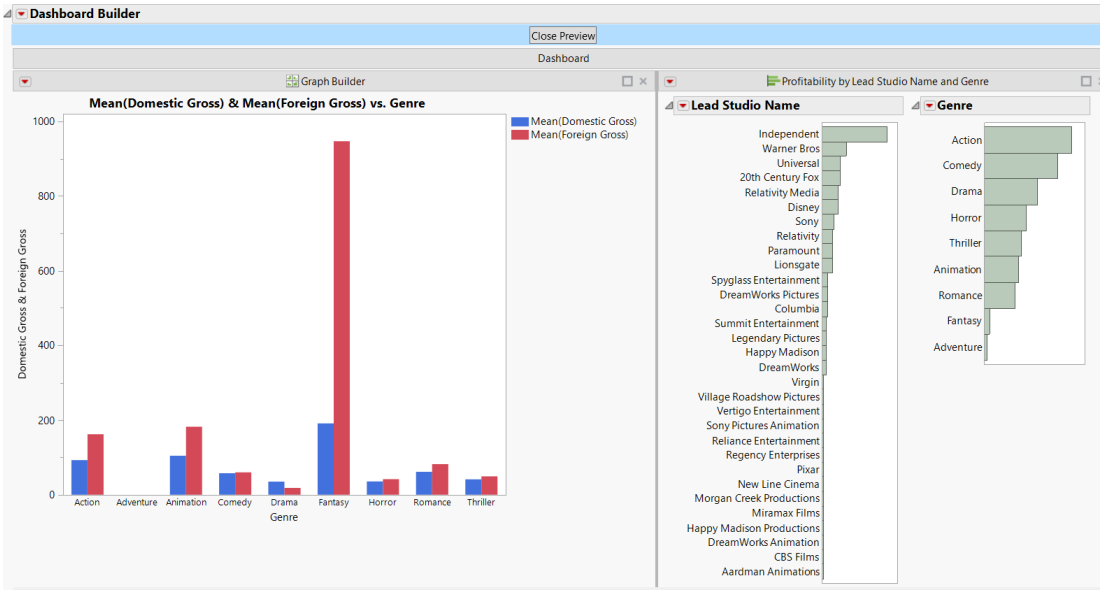
Chapter 11

Combine and Present Reports Using Dashboards

Extend JMP

JMP is an extensible program that lets you broaden its functionality to create your own JMP components. With Dashboard Builder, you combine reports into a dashboard, a visual tool that lets you run and present reports on a regular basis. No JSL knowledge is required. Start with a template, click report components, insert graphs and pictures, and the dashboard is done. You can also combine windows outside Dashboard Builder to create dashboards.

Figure 11.1 A Dashboard with Two Reports



Contents

Combine JMP Reports by Creating a Dashboard 623
Example of Creating a Dashboard with Two JMP Reports 623
Using the Dashboard Builder 624
Red Triangle Options in the Dashboard Builder 625
Start with a Sample Dashboard 626
Edit a Dashboard 627
View a Running Dashboard 630
Example of Combining Windows to Create a Dashboard 633
Example of Adding Selection Filters to a Dashboard 635
Example of Creating a JMP Query Dashboard and Add-In 637
Example of Creating a Dashboard from Two Data Tables 639

Combine JMP Reports by Creating a Dashboard

With Dashboard Builder, you combine reports to create your dashboards without writing JSL scripts. Similar to Application Builder, Dashboard Builder has a drag-and-drop interface but with pre-configured display boxes. In Dashboard Builder, the display boxes do not require customization as they do in Application Builder. You can put reports and graphs in tabs, embed database queries, save the dashboard as an add-in, and more.

You can also quickly create dashboards by combining several open windows using the Window > Combine Windows option. For example, you might combine two reports and a data table to arrange them on a dashboard. See "Example of Combining Windows to Create a Dashboard" for an example.

Example of Creating a Dashboard with Two JMP Reports

Suppose that you created two reports and want to run the reports again the next day against an updated set of data. This example shows how to create a dashboard from the reports.

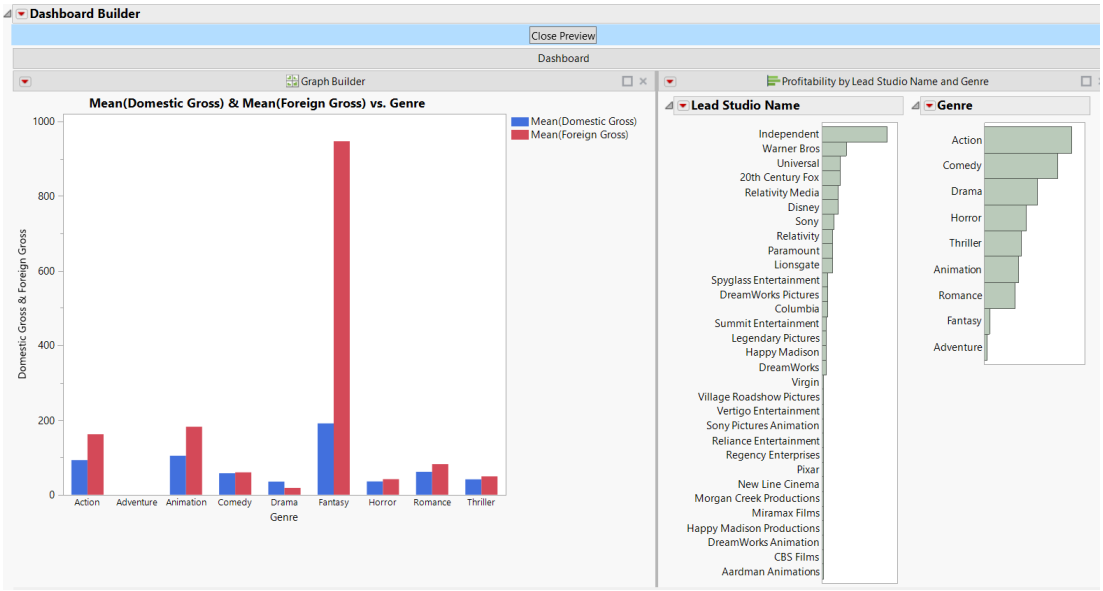
1. Select **Help > Sample Data Library** and open Hollywood Movies.jmp.
2. To create the reports, run the table scripts named "Distribution: Profitability by Lead Studio and Genre" and "Graph Builder: World and Domestic Gross by Genre".
3. From any window, select **File > New > Dashboard**.

   Templates for common layouts appear.
4. Select the **2x1 Dashboard** template.

   A box with room for two reports appears on the workspace.
5. In the Reports list, double-click the report thumbnails to put them on the dashboard (Figure 11.3).
6. Click the Dashboard Builder red triangle and select **Preview Mode**.

   A preview of the dashboard appears (Figure 11.2). Notice that the graphs are linked to each other and the data table. They also have the same red triangle options as the Distribution and Graph Builder platforms.

Figure 11.2 A Dashboard with Two JMP Reports



7. Click **Close Preview**.
8. In Dashboard Builder, select **File > Save**, and then save the file as a .jmpapp file.

   Users who have JMP can open the data table and double-click this file to display the dashboard. Note that the .jmpapp file cannot be edited.
9. Return to the Dashboard Builder window, select **File > Save As**, and then save the file as a .jmpappsource file.

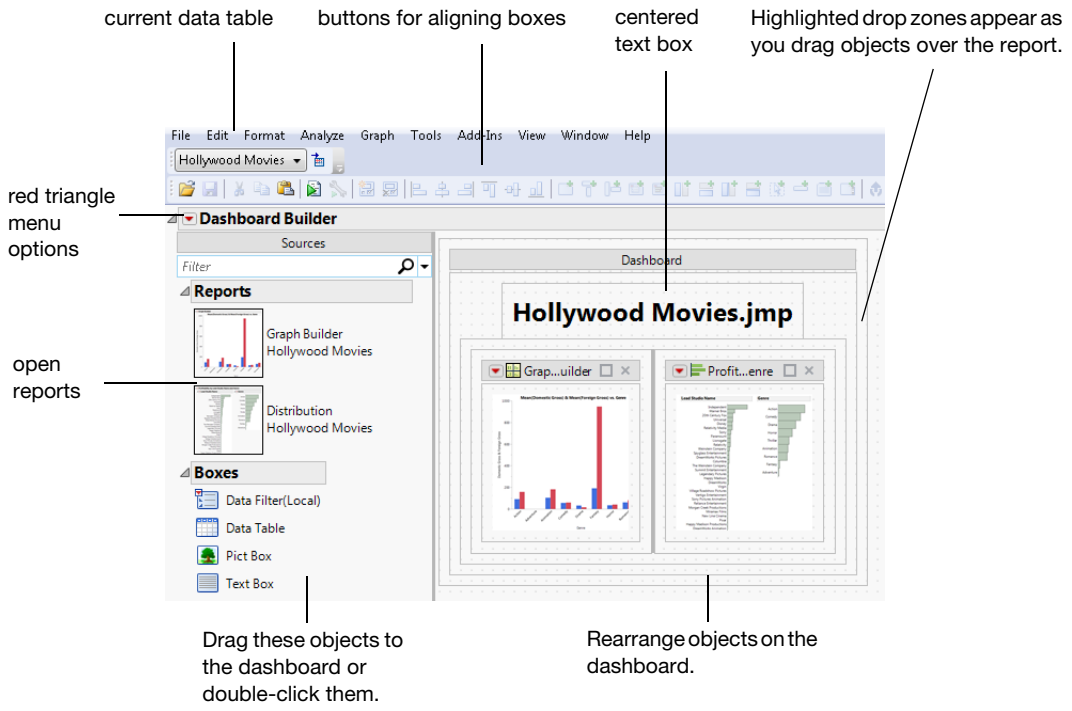
   The dashboard is saved in a file that you can edit in the future.

Note: You might also want to save the dashboard as an add-in. Users then view the dashboard by selecting a menu item in the main Add-Ins menu instead of opening the .jmpapp file. See "Example of Creating a JMP Query Dashboard and Add-In".

Using the Dashboard Builder

Figure 11.3 shows a dashboard in development. The two reports in the left pane were generated from the Hollywood Movies.jmp sample data table. Those reports were added to the right pane. "Hollywood Movies.jmp" is displayed in a text box that was added to the dashboard and centered using the toolbar buttons.

Figure 11.3 The Dashboard Builder Window



Red Triangle Options in the Dashboard Builder

The Dashboard Builder red triangle menu provides options such as running the dashboard, previewing the dashboard, and saving the dashboard as an add-in.

Run Dashboard Starts the dashboard.

Preview Mode Displays a preview of the dashboard. Click **Close Preview** on the dashboard to return to the dashboard mode.

Start Over Remove objects from the workspace and select a new template.

Show Grid Displays dotted grid lines on the workspace. Selected by default.

Show Properties Shows or hides the Objects and Properties panels for editing display boxes. See the *Scripting Guide* for more information about editing properties.

Auto Scroll Automatically scrolls horizontally or vertically as you drag an object near the edges of the workspace. Selected by default.

Dashboard Mode Deselect to display the dashboard in Application Builder, where you can fully customize the application and write scripts. See the *Scripting Guide*. Select **Dashboard Mode** to return to the dashboard.

Source Panel Shows or hides the Sources panel (the left column in which report components are displayed) and categorizes the icons. Show Sources is selected by default.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

To Data Table Saves the script to the data table that was used to produce the report. This enables you to run the script again from the data table to recreate the results.

To Journal Saves a link in a journal window that runs the script for the analysis. If a journal window is already open, the script is added to that journal window. The script contains the path to the data table. Note that if the data table cannot be found, the script does not run.

To Script Window Opens a script editor window and adds the script to it. If a script window is already open, the script is added to the bottom of that open window.

To Add-In Lets JMP users install the application and launch it from a JMP menu. See the *Scripting Guide*.

To Template Saves the application as a template from which you can create new dashboards. The templates are saved in the following locations based on your operating system and JMP version:

- Windows (JMP): "/C:/Users/<username>/AppData/Roaming/SAS/JMP/16/DashboardTemplates"
- Windows (JMP Pro): "/C:/Users/<username>/AppData/Roaming/SAS/JMPPro/16/DashboardTemplates"
- macOS: "/Users/<username>/Library/Application Support/JMP/16/DashboardTemplates"

Start with a Sample Dashboard

Examples of dashboards are shown when you open Dashboard Builder. You can modify these samples instead of starting with an empty dashboard. The dashboards are also installed in the JMP Samples/Dashboards folder.

The following sample dashboards are available:

Instant Dashboard.jmpappsource Combines two Multivariate reports (Principal Components/Factor Analysis and T Square with All Principal Components).

Six Quality Graphs Dashboard.jmpappsource Creates three Control Charts, a Distribution report, and a Capability Analysis report.

Edit a Dashboard

"Example of Creating a Dashboard with Two JMP Reports" shows how to create a dashboard from two reports. This section shows how to arrange those reports and preview the final dashboard.

Rearrange Reports in Dashboard Edit Mode

To help you rearrange reports while editing a dashboard, the outline around a report includes drop zones. When you drag the report into a different area of the box, a portion of the box is highlighted. The highlight shows where you can drop the box.

Figure 11.4 shows how to create tabbed reports. To put the Distribution report inside a tab, drag the report over the other report until the upper right corner of the bar chart report is highlighted. The Distribution report appears in a tab on top of the other report.

To remove a tabbed report, click the middle of the report and drag it outside of the tab.

Figure 11.4 Working with Tabbed Reports in Dashboard Edit Mode

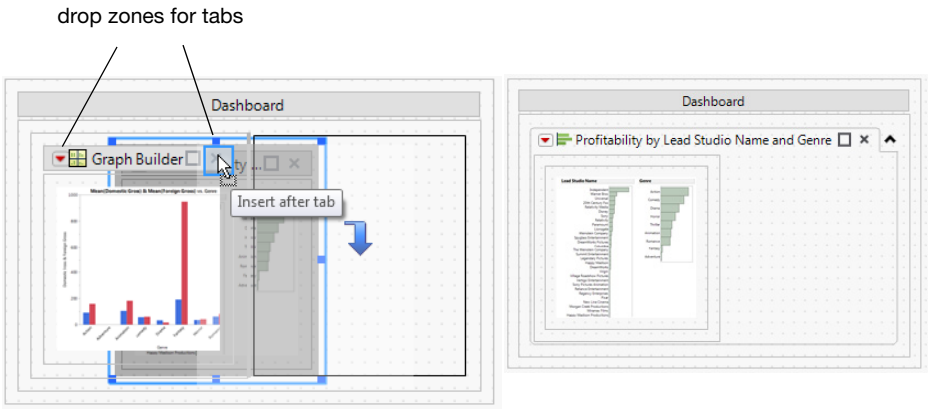


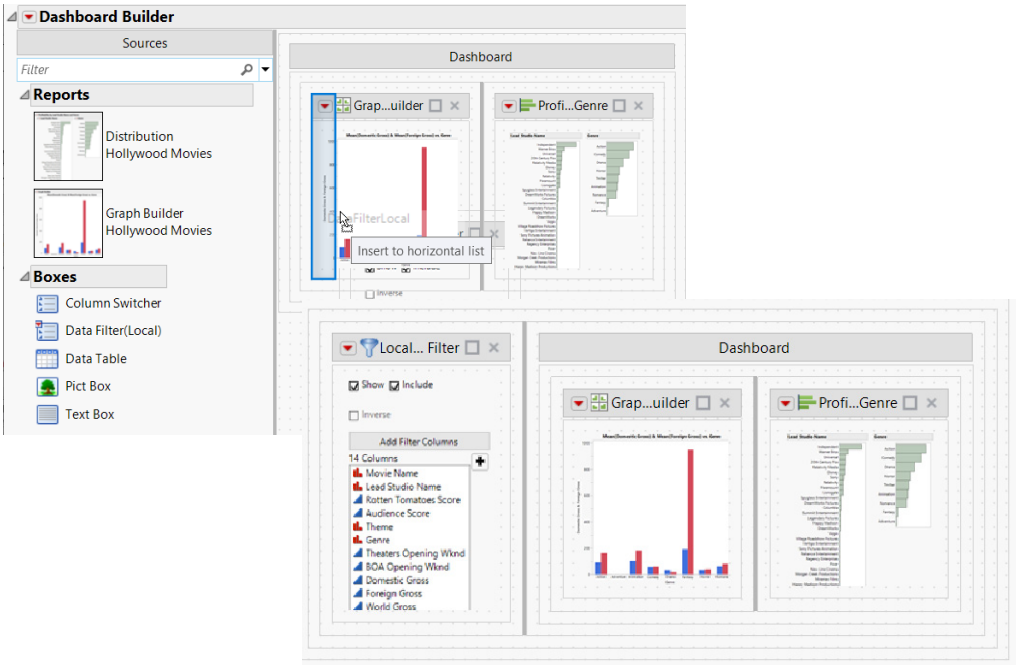
Figure 11.5 shows an example of placing a data table inside the bottom border of the report.

To display the data table *below* the report instead of inside it, drag the Data Table box onto the dashboard below the report. The box snaps inside the bottom border of the report.

Figure 11.5 Displaying a Data Table Inside a Report

Figure 11.6 shows a data filter that appears to the left of a report.

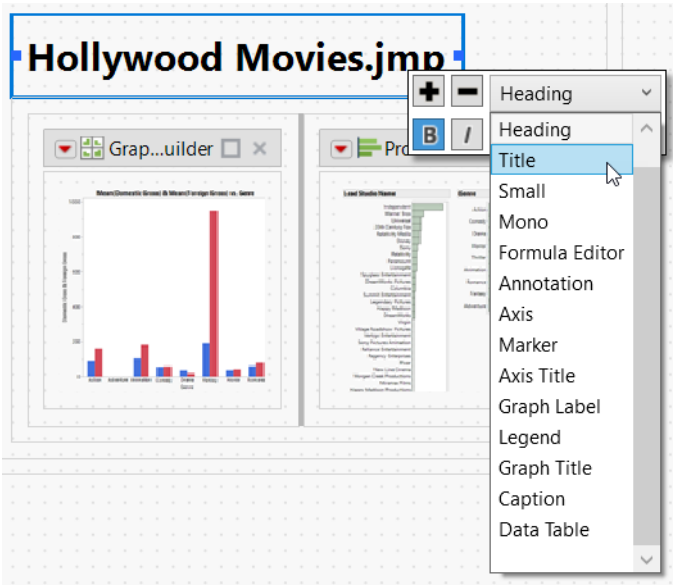
Figure 11.6 Adding a Data Filter to the Left of a Report



Format Text in a Dashboard

When you edit a dashboard, you can format text in a text box. Drag the text box to the dashboard and click the middle of the text box. Formatting options appear.

Figure 11.7 Formatting Options for a Text Box



Preview a Dashboard

When you edit a dashboard, you can see a preview of the final dashboard by selecting **Preview Mode** from the Dashboard Builder red triangle menu. Previewing a dashboard is particularly helpful when testing interactive elements such as column switchers, data filters, and tabbed reports.

Note: If you resize panes in preview mode, your changes are not saved.

Tips:

- The column switcher initially shows all columns in the table, but it does not actively make changes to reports in the dashboard or application.
- You should switch into preview mode to do the following:
  - Select the initial column to be switched in all reports.
  - Remove any columns that you do not want to appear in the column switcher, including any other columns that are used in the reports.
- When you run the dashboard or application, the column switcher automatically connects to all platforms in the dashboard that use the same data table.

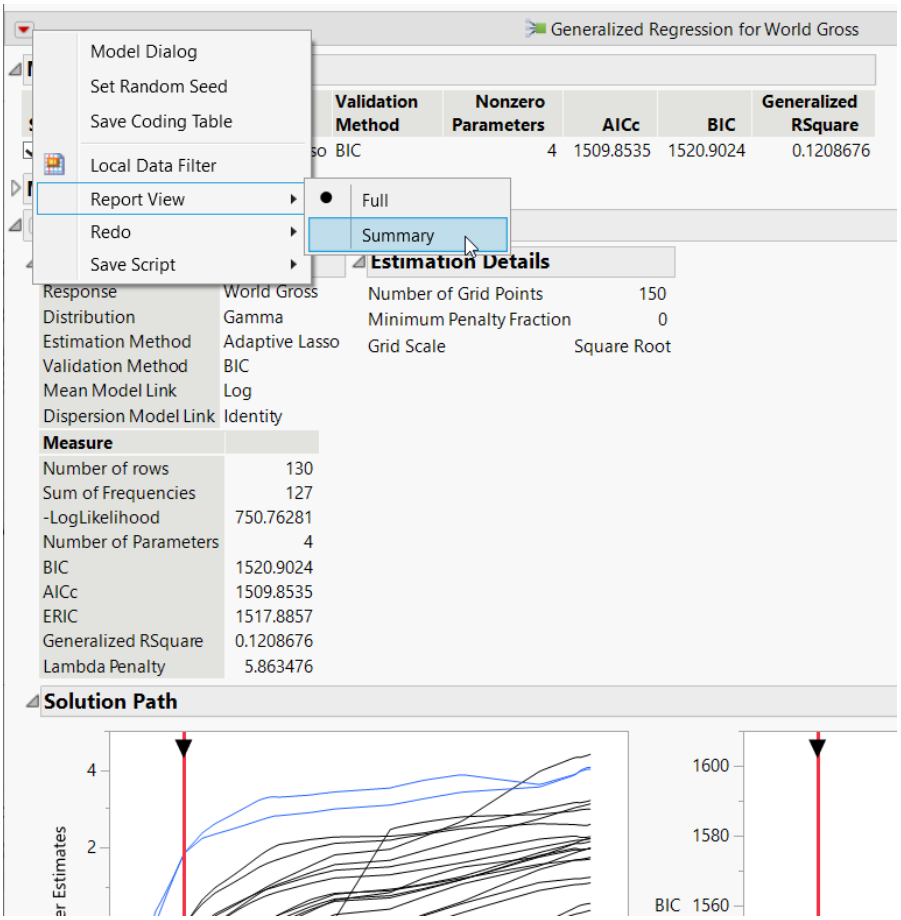
View a Running Dashboard

When you view a running dashboard, the same red triangle options that you find in JMP platforms are available in the reports. Dashboard Builder also provides several other options.

View a Report Summary

A report that contains graphs and statistics can grow fairly large. To view only the graphs while running the report, select the top red triangle in a report and then select **Report View > Summary**. The report shown in Figure 11.8 originally showed parameter estimates. In the Summary view, only the graphs are shown.

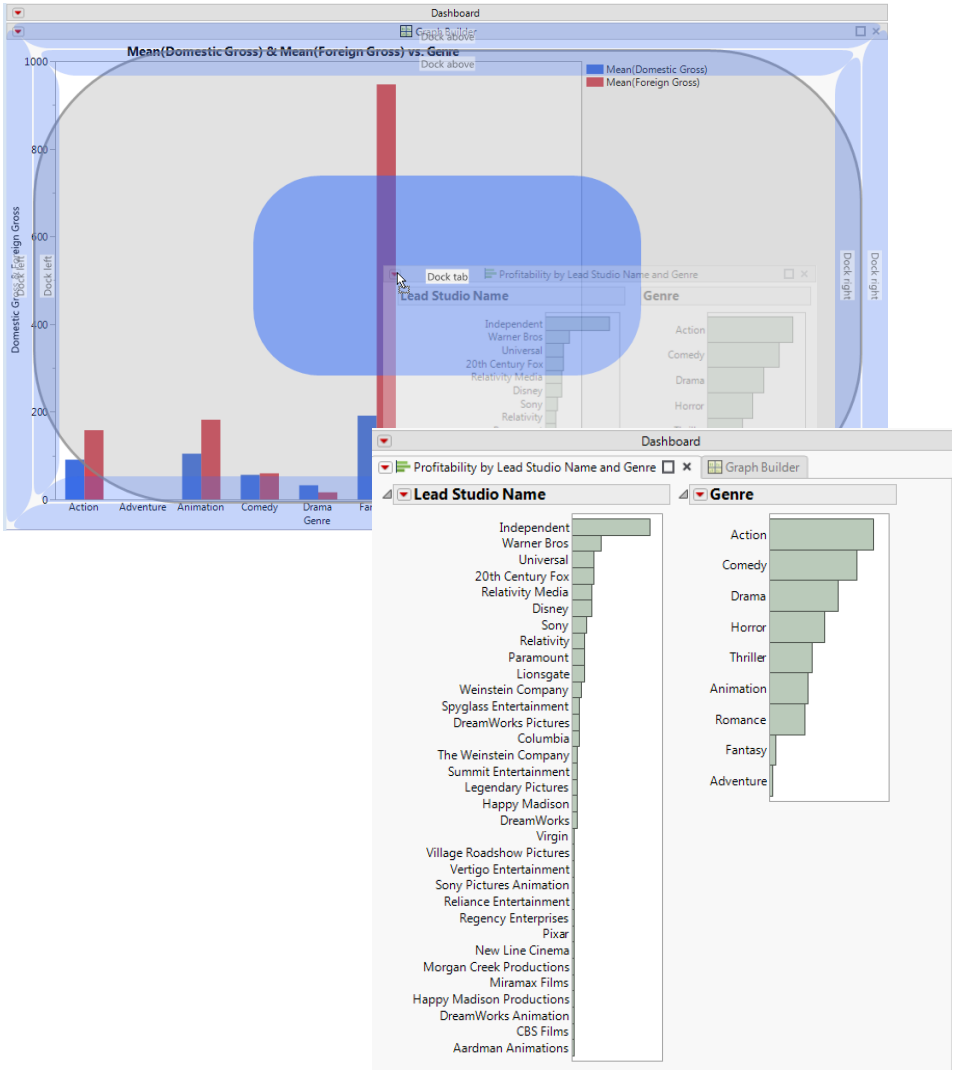
Figure 11.8 Viewing a Summary of the Report



Rearrange Reports in a Running Dashboard

You rearrange reports in a running dashboard by dragging the reports into highlighted drop zones. The drop zones indicate where you can place the report. Figure 11.9 shows an example of moving a report into a tab. The Distribution report is dropped inside the “Dock tab” drop zone to place it on the first tab.

Figure 11.9 Creating Tabbed Reports in a Running Dashboard

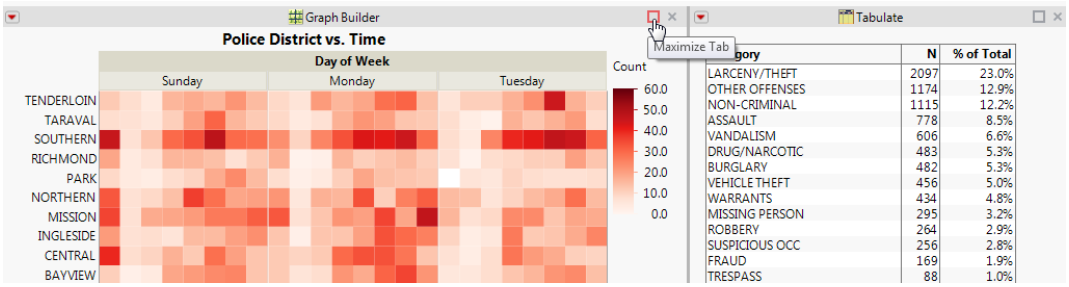


To remove a tabbed report, click the middle of the report and drag it outside of the tab.

Maximize a Report in a Running Dashboard

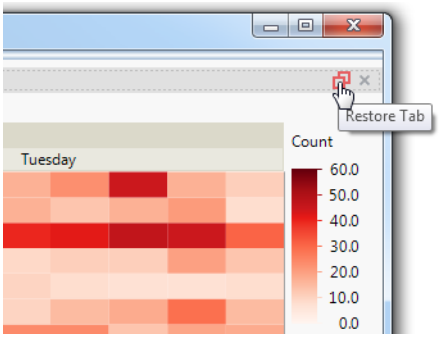
To see only a specific report while running a dashboard, select the report's **Maximize Tab** box.

Figure 11.10 Maximizing a Tab



To restore the report to its original size, select the **Restore Tab** box.

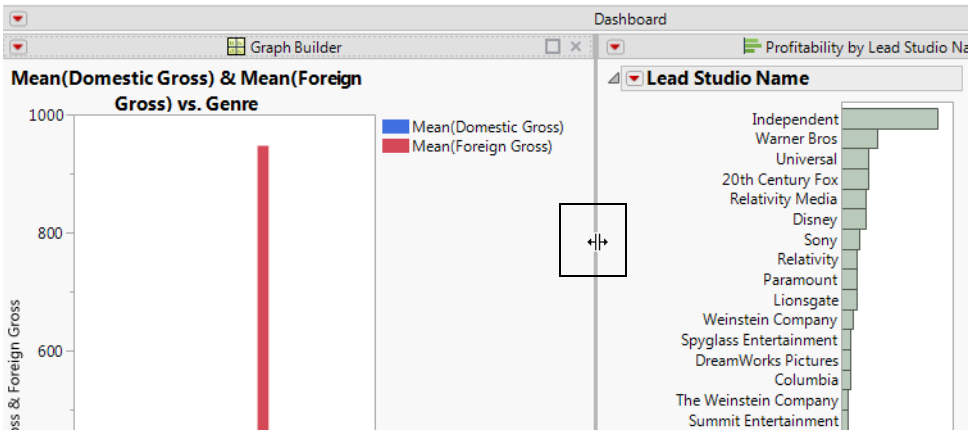
Figure 11.11 Restoring a Maximized Report



Resize Reports in a Running Dashboard

The thick gray line that appears between reports in a dashboard is called a *splitter*. Drag the splitter to resize the panes. When the pane is too wide to display the enclosed report, the pane closes. Drag the splitter from the outer edge of the dashboard to see all reports again. Figure 11.12 shows the collapsed and expanded reports.

Figure 11.12 Resizing Panes in a Running Dashboard



You can also drag the edge of a graph. The graph widens as far as the width of the window and the other reports allow.

Share Reports with Users Working Outside JMP

You can share reports with users working outside JMP by saving a dashboard as interactive HTML. First, click the Dashboard Builder red triangle and select **Run Dashboard** to run the dashboard. In the running dashboard, select **File > Save As** and then select **Interactive HTML with Data** from the list. The reports are saved in a web page that also contains the data. Email the web page to users or publish the page on a website. Users then explore the data as they would in JMP. See "Share Reports as Interactive HTML".

Edit a JMP Platform Script

A dashboard consists of a JSL script, which you can edit in Dashboard Builder. For example, you might edit the script for the Multivariate platform to show the scatterplot matrix or to use different columns.

Right-click the object for a platform in the dashboard and select **Edit Platform Script**. The script opens in a script editor window.

After you edit the script and click OK, the object on the workspace is updated.

Example of Combining Windows to Create a Dashboard

You can quickly create dashboards by merging several open windows in JMP. Combining windows provides options to view a summary of statistics and include a selection filter.

1. Select **Help > Sample Data Library** and open Birth Death.jmp.

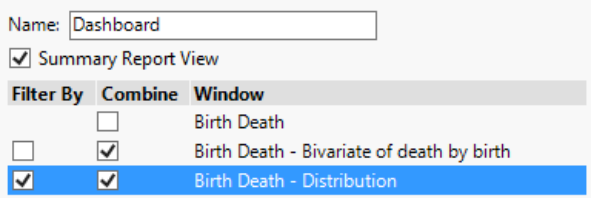
2. Run the Distribution and Bivariate table scripts.
3. From one of the report windows, select **Window > Combine Windows**.

   The Combine Windows window appears.

   Tip: On Windows, you can also select Combine Windows from the Arrange Menu option in the lower right corner of JMP windows.

4. Select **Summary Report View** to display the graphs and omit the statistical reports
5. In the Combine column, select **Birth Death - Bivariate of death by birth** and **Birth Death - Distribution**.
6. In the Filter By column, select **Birth Death - Distribution**.

Figure 11.13 Combine Windows Options



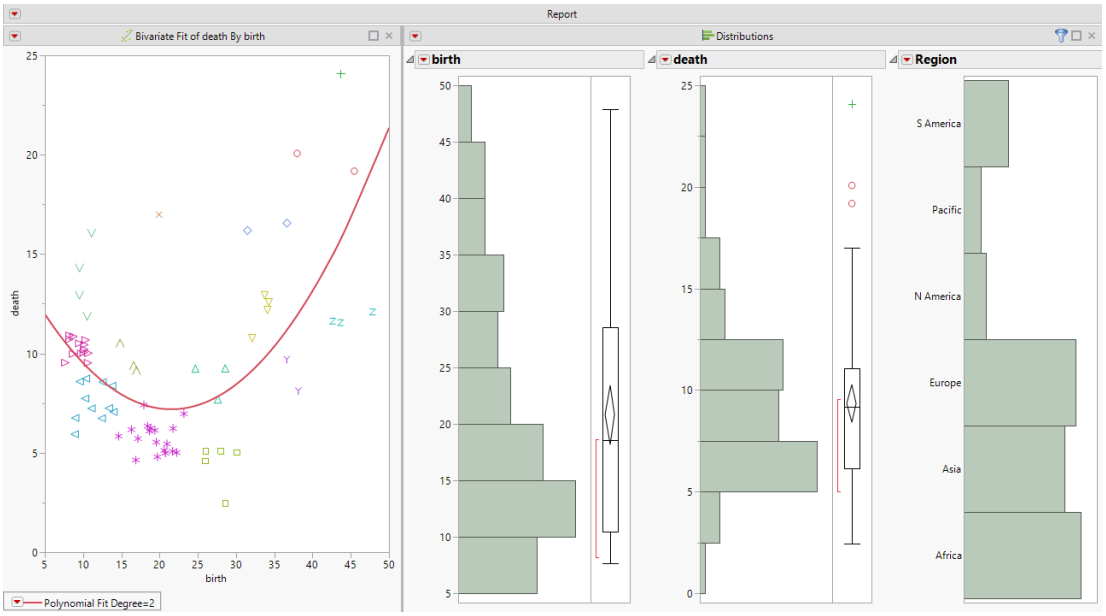
7. Click **OK**.

   The two reports are combined into one window. Notice the filter icon at the top of the Distribution report. When you select a bar in one of the histograms, the corresponding data in the Bivariate graph are selected.

Notes:

- To combine reports on Windows, you can also select Combine Windows from the Arrange Menu option in the lower right corner of JMP windows.
- In the Combine Windows window, select **Summary Report View** to see only the graphs in a report and omit the statistics.

Figure 11.14 Combined Windows



Example of Adding Selection Filters to a Dashboard

In a dashboard that contains several reports, you can select data in a report and then view only that data in other reports contained in the same window. This example shows how to add selection filters to the primary report when you edit a dashboard.

Open Reports and Create a Dashboard

1. Select **Help > Sample Data Library** and open Hollywood Movies.jmp.
2. To create the reports, run the table scripts named "Distribution: Profitability by Lead Studio and Genre", "Graph Builder: World and Domestic Gross by Genre", and "Bubble Plot: Theaters Open Week by Domestic Gross".
3. From any window, select **File > New > Dashboard**.

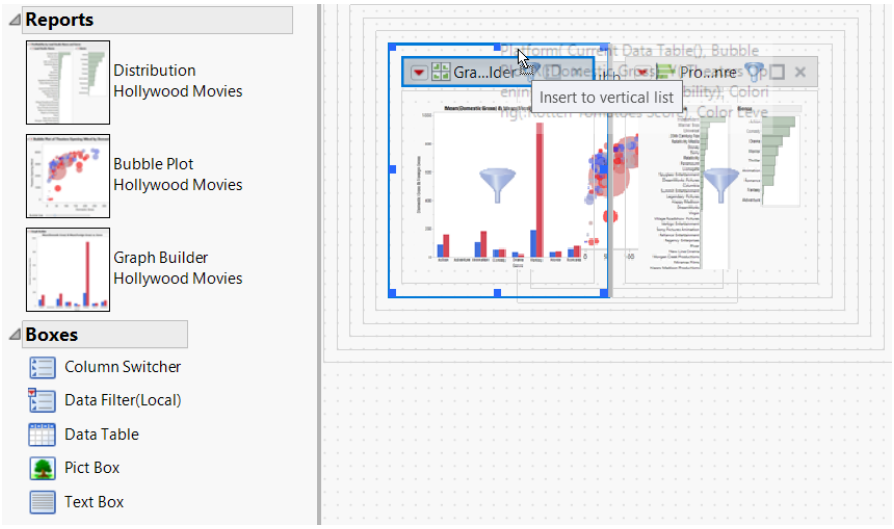
   Templates for common layouts appear.
4. Select the **2x1 Dashboard** template.

   A box with room for two reports appears on the workspace.
5. In the Reports list, double-click the Graph Builder and Distribution report thumbnails to put them on the dashboard.

Add and Use Selection Filters

1. Right-click the center of the left report (Graph Builder) and select **Use As Selection Filter**.
2. To add another report as a second selection filter, click the Bubble Plot thumbnail and drag it into the blue line around the Graph Builder graph.

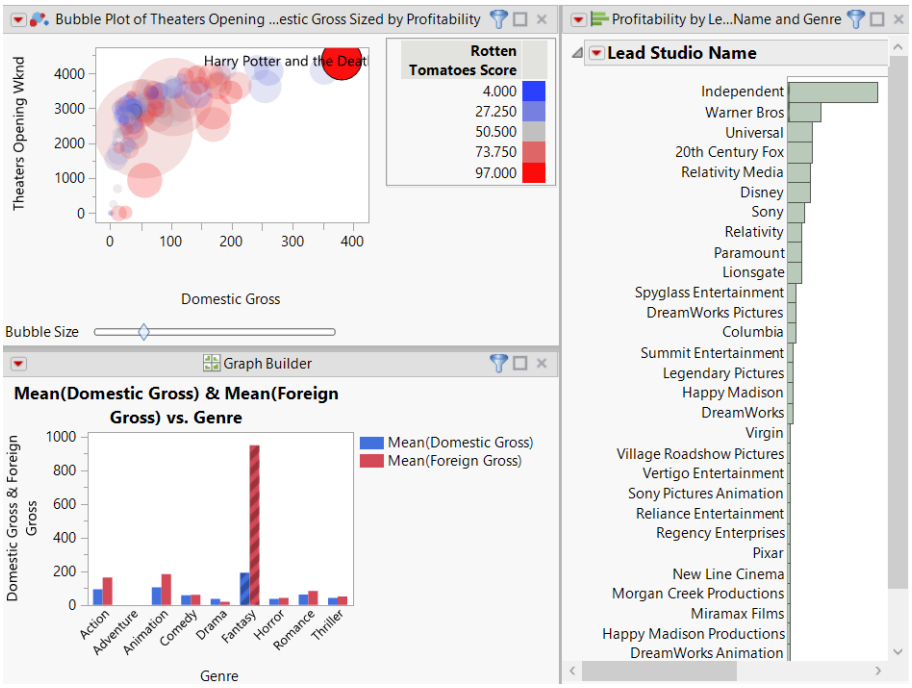
Figure 11.15 Dragging the Bubble Plot into the Graph Builder Graph



A filter icon appears on both thumbnails.

3. Click the Dashboard Builder red triangle and select **Preview Mode**.
4. Select one of the bars for the Fantasy genre in the left report.

Figure 11.16 Filtering Data



In the right report, you can see that Warner Bros and Independent studios produced the most fantasy movies. In the bottom report, you can see that Harry Potter was the most profitable fantasy movie.

Example of Creating a JMP Query Dashboard and Add-In

Suppose that you used JMP Query Builder to import portions of two data tables into JMP. A dashboard displays reports of the data. You want to create an add-in that queries the data and shows the updated reports on a dashboard.

Build the Query

1. Select **Help > Sample Data Library** and open Hollywood Movies.jmp.
2. Select **Tables > JMP Query Builder**.
3. Click **Build Query**.
4. In the Available Columns list, double-click t1.Movie Name, t1.Rotten Tomatoes Score, t1.Audience Score, and t1.Theaters Opening Wknd to add them to the Included Columns tab.
5. Click **Run Query**.

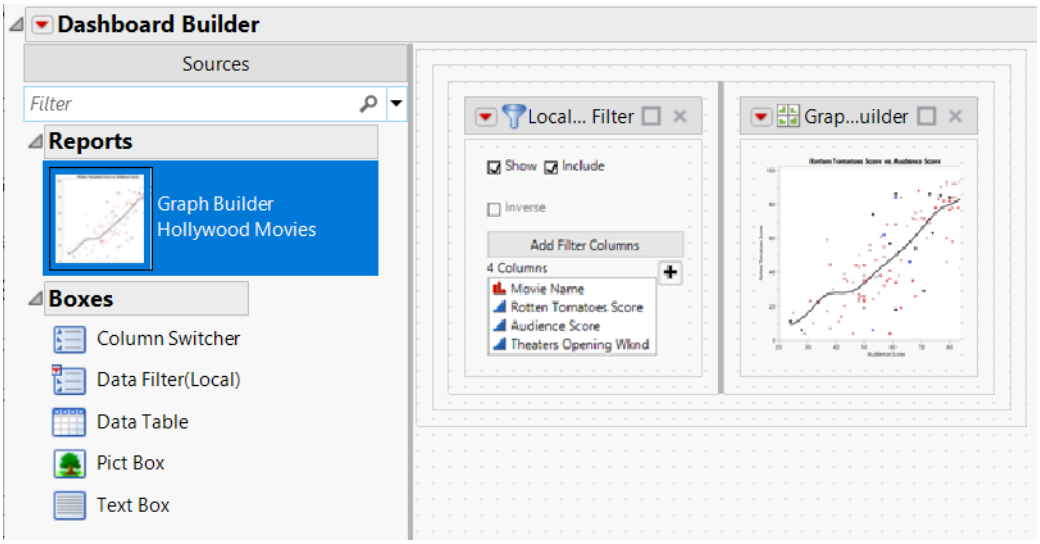
The queried data appears in a new data table. Note the Source table script. When you open the dashboard, JMP runs this Source script to query the data in Hollywood Movies.jmp.

6. In the new data table, run the **Graph Builder: Scores by Opening Weekend** table script to create a report.

Create the Dashboard

1. From the report window, select **File > New > Dashboard**.
2. Select the **Blank Dashboard** template.
3. Double-click the **Data Filter(Local)** box in the left pane to add it to the dashboard.
4. Double-click the Graph Builder report in the left pane to place it next to the local data filter.

Figure 11.17 Dashboard for Hollywood Movies



Create and Run the Add-In

1. Click the Dashboard Builder red triangle and select **Save Script > To Add-In**.

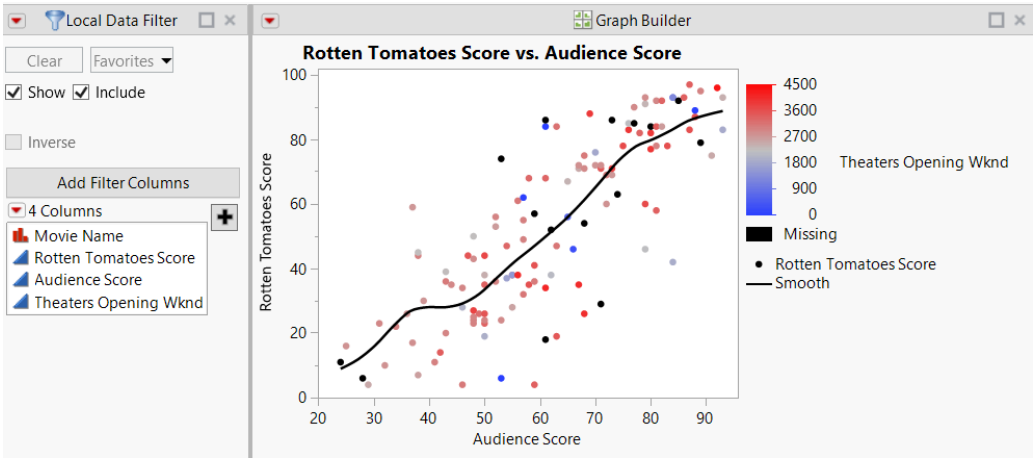
   Add-In Builder opens.
2. Next to Add-In name, type Hollywood Movies Dashboard (the add-in filename).
3. Click the **Menu Items** tab and type Hollywood Movies Dashboard next to **Menu item name** (the name of the add-in menu item).
4. Click **Save** and save the add-in to your desktop.

   The add-in is saved and installed in your Add-Ins menu.
5. Click **Close** on the Add-In Builder window.

6. From the JMP main menu, select **Add-Ins** and select **Hollywood Movies Dashboard**.

   The Hollywood Movies.jmp data table is queried, and the dashboard opens to show the local data filter and updated report.

Figure 11.18 Hollywood Movies Add-In



Example of Creating a Dashboard from Two Data Tables

The graphs or reports in a dashboard can be created from multiple data tables. The data table used for the first graph on the dashboard is considered the current data table by default. This gives the user the flexibility to use any current data table to create the graph or report. The dashboard author can also hard-code the path. Specifying the name gives the dashboard author control over which data table is used.

1. Select **Help > Sample Data Library**, open Big Class.jmp, and run the Bivariate script.
2. Select **Help > Sample Data Library**, open Hollywood Movies.jmp, and run the Distribution script.
3. From any JMP window, select **File > New > Dashboard**.
4. In Dashboard Builder, select the **2 x 1 Dashboard** template.
5. Double-click the Distribution graph to add it to the dashboard first.

   By default, this graph uses whatever data table is the current data table. In this example, Hollywood Movies.jmp stored in the Samples/Data folder is the current data table.
6. Double-click the Bivariate graph to add it to the dashboard.

   You are prompted to determine whether the first graph uses the current data table or whether you want to hard-code the path to the data table (reference the table by name) in the dashboard.

7. Select **Reference table by name** to make sure the specified data table is always used to create the graph.
8. Click **OK**.

   In this example, Hollywood Movies.jmp in the Samples/Data folder will always be used as the current data table for the first graph, despite the current data table that is open.

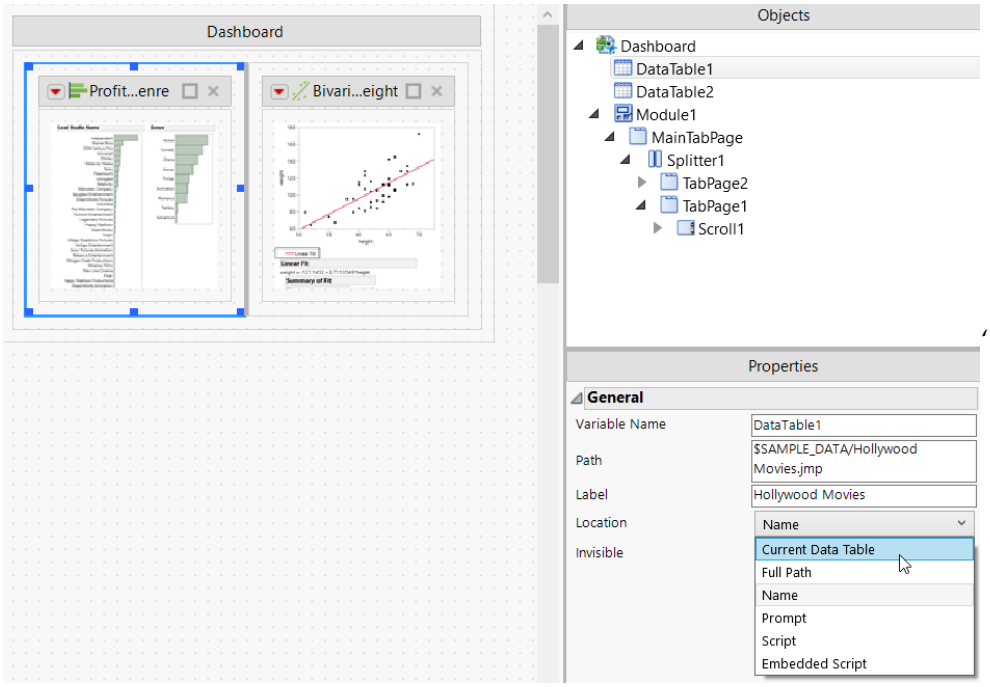
Tip: Sometimes a named data table cannot be found when the dashboard is run. This is particularly common when the dashboard is moved to a different computer. You can either open the data table before running the dashboard or run the dashboard and select the data table when prompted.

To Change the Location of the Data Table

After you assign the location of the data table (current data table or name), you can change the setting in the dashboard Objects and Properties panes.

1. Click the Dashboard Builder red triangle and select **Show Properties**.

Figure 11.19 The Objects and Properties Panes



2. In the Objects pane, select **DataTable1**.
3. In the Properties pane, change the Location as shown in Figure 11.19.

For example, if you change the Name location to Current Data Table, the selected graph on the dashboard is based on the current data table.

Note: The Location value of Current Data Table applies only when you are running the dashboard. When you are editing the dashboard, Dashboard Builder uses the full path to locate the data table.

Chapter 12

Customize Menus and Toolbars

Personalize JMP

This chapter describes how to customize JMP menus and toolbars to show only the commands that you need. For example, you might remove the SAS option in the File menu if you do not use SAS. Or you might assign a shortcut key to the Run Script command.

Customizing JMP also lets you set up JMP for groups with special interests. If one group does not design experiments, you could remove DOE from the JMP menu.

Note: When you upgrade to JMP 16 Subscription, changes to preferences are maintained.

Contents

Customize JMP Toolbars and Menus on Windows 645
Customize Toolbars 646
Customize the Main Menu Bar 649
Create Menu Items and Toolbar Buttons 650
Set User Permissions for Customized Toolbars and Buttons 654
Rearrange Toolbars 655
Copy and Paste Menus, Menu Items, Toolbars, and Buttons 655
Rearrange Custom Menus, Menu Items, and Buttons 657
Delete Custom Items 658
Show or Hide Custom Items 659
Import Customizations 660
Remove Customizations from JMP 660
Customize Toolbars on macOS 661
Customize Menus on macOS 663
Customize Menus and Toolbars for Add-Ins 664

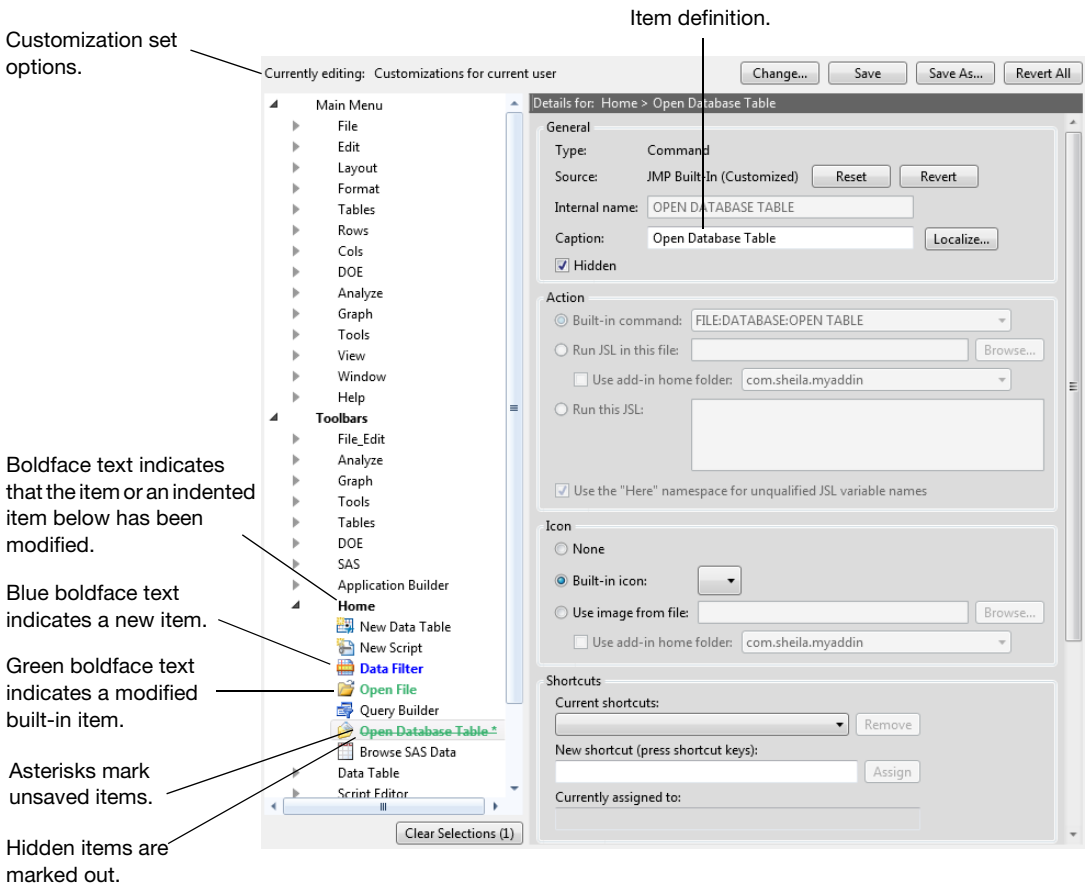
Customize JMP Toolbars and Menus on Windows

Toolbars consist of buttons that execute commands and dividers that help you organize the buttons. In Windows, there are many ways to customize toolbars. For example, you can create, rename, hide, and modify toolbars or buttons. You can also rearrange or delete toolbars and buttons that you create.

Menus consist of commands, submenus, and separators. As with toolbars and buttons, you can create, rename, hide, and modify menus. You can also rearrange or delete menus and menu items that you create.

You customize toolbars and menus in the Menu Editor. The names of modified items are formatted to indicate which items were created, modified, or hidden, and which items are unsaved.

Figure 12.1 Menu Editor Options



Customization Set Options

Customization Set to Modify

Current User Only you see the changes. The customization file is called usercust.jmpcust. Modifying this file manually might cause unexpected results.

All users All users who open JMP on the computer see the changes. The customization file is called admincust.jmpcust.

JMP Add-In Those who use the selected JMP Add-in, and select **JMP Add-In customizations** as the Starting Set, see the changes. When you modify the customization set for a disabled add-in, those changes do not appear until you enable the add-in.

Other file Changes appear when the selected .jmpcust file is chosen as the customization set. You create this file in a text editor.

New Changes appear when this new customization is selected. (After you make your changes, click **Save** to name the new file in the selected folder.)

Starting Set

JMP built-in items The selected customization set is based on the default JMP toolbars and menus. (Applies to all customization sets and cannot be deselected.)

All Users customizations The selected customization set is based on the All users customization set. (Available for the Other file or New customization sets. Cannot be deselected from the Current user customization set.)

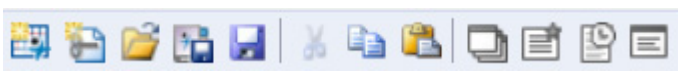
JMP Add-In customizations The selected customization set is based on the JMP Add-In customizations. The add-in customizations are typically installed with the add-in. (Available for all customization sets except for All users.)

Current User customizations The selected customization set is based on the Current user customization set. (Available only for Other file and New customization sets.)

Customize Toolbars

JMP includes toolbars that you can add new buttons to, or you can create your own toolbars. By default, the JMP window type determines which bars appear. For example, the File_Edit toolbar is in all windows. In data tables, the Data Tables toolbar also appears.

Project Shows icons for projects.



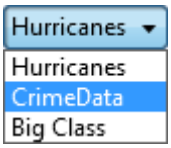
File_Edit Shows icons for commands found in the File and Edit menus.

Analyze Shows icons for common commands found in the Analyze menu.

Graph Shows icons for commands found in the Graph menu.

Tools Shows icons of tools that you can click and use as your cursor. In some windows, the toolbar is hidden. A blue line appears instead. Hover over the blue line to show the toolbar.

Data Table List Shows a list of open data tables. You select a data table in this list to make it the current table. Note that the current table is not necessarily the front window. To bring a table (or any window) to the front, select its name from the list in the Window menu.

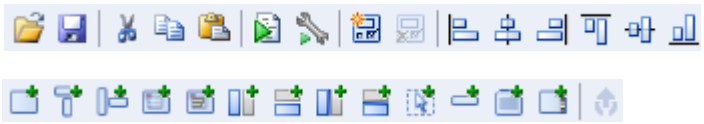


Tables Shows icons for commands in the Tables menu.

DOE Shows icons for commands in the DOE menu.

SAS Shows icons for accessing and browsing SAS data and folders, opening SAS Query Builder, creating a new SAS program, submitting data to SAS, opening the SAS log window and output window, and viewing server connections.

Application Builder Shows icons for commands in Application Builder.



Home Shows frequently used icons from the File menu.

Data Table Shows icons for performing common tasks in a data table, such as opening the Data Filter, Fit Model, and Recode windows.

Script Editor Shows icons for running, debugging, and reformatting scripts.

Report Shows icons for opening the Local Data Filter, Column Switcher, and saving a script to the data table.

Create your own toolbars for frequently used commands that are not included in the default toolbars. You assign a command to the button and can add other properties such as shortcut keys, icons, and JMP Scripting Language (JSL) scripts.

When you want to create a toolbar based on an existing toolbar, make a copy of the existing toolbar and then change settings as necessary. See "Copy and Paste Menus, Menu Items, Toolbars, and Buttons".

Step 1: Create the Toolbar

1. Select **View > Customize > Menus and Toolbars**.

   The Toolbars list appears on the left.
2. (Optional) Change the customization set to control which users see your customizations. See "Customization Set Options".
3. With the Menu Editor still opened, right-click one of the toolbars, such as **File_Edit**.

   Select any toolbar, because all new toolbars appear at the end of the Toolbars list.
4. Select **New Toolbar**.

   The toolbar is inserted at the end of the list. The toolbar includes an untitled button, because all toolbars must have at least one button.

Step 2: Specify the Caption and Internal Name

Toolbars have several basic properties:

- The caption appears in the **View > Toolbars** list, which lets you show or hide the toolbar. Give each toolbar a unique name. JMP merges toolbars with the same name after you close and reopen the Menu Editor.
- The internal name, which is case insensitive. JMP identifies the location of an item by its internal name rather than its caption.

To specify these toolbar properties, follow these steps:

1. With the Menu Editor still opened, select the toolbar.

   The General properties appear on the right.
2. Enter the internal name next to **Internal name**.
3. Enter a unique name for the toolbar or button in the **Caption** box.
4. (Optional) To specify the translation for the caption, do the following:
   - Click the **Localize** button for the item that you want to customize.
   - Select the language.
   - Enter the translation next to **Text**.
   - Click **OK**.
5. (Optional) Click **Hidden** to keep the toolbar hidden after creation.
6. Customize the untitled button as described in "Create Menu Items and Toolbar Buttons".
7. Click **Save** to save your changes.

Customize the Main Menu Bar

Add your own menu items to the main menu bar in JMP. These menus appear before or after menus such as **File**, **Edit**, and **Tables**. An untitled command is inserted automatically in the menu, because all menus must have at least one command.

When you want to create a main menu based on an existing menu, make a copy of the existing menu and then change settings as necessary. See "Copy and Paste Menus, Menu Items, Toolbars, and Buttons".

To create a main menu

1. Select **View > Customize > Menus and Toolbars**.

   The Main Menu list appears on the left.
2. (Optional) Change the customization set to control which users see your customizations. See "Customization Set Options".

3. Right-click the menu next to which you want to add a new menu.
4. Select **Insert Before** or **Insert After**.

   An untitled menu and a menu item are added to the list.
5. Complete "Step 2: Specify the Caption and Internal Name" to rename the main menu.
6. Customize the menu item as described in "Create Menu Items and Toolbar Buttons".
7. Click **Save** to save your changes.

Create Menu Items and Toolbar Buttons

Menus consist of the following menu items:

- Commands are the items that you click to execute a command (such as Data Table and Script).
- Submenus are menu items that you click to reveal more menu options. An example of submenu is New inside the top-level File menu. New is also a submenu because it contains commands (such as Data Table and Script).
- Separators ( ) are lines that divide or group commands and submenus.

The button on a toolbar is considered a type of command, so you also complete the following steps to create new toolbar buttons.

When you want to create a menu item or button based on an existing item, make a copy of the existing item and then change settings as necessary. See "Copy and Paste Menus, Menu Items, Toolbars, and Buttons".

Step 1: Create a New Menu Item or Button

1. Select **View > Customize > Menus and Toolbars**.

   The Menu Editor appears.
2. (Optional) Change the customization set to control which users see your customizations. See "Customization Set Options".
3. Right-click where you want to add the menu item or button.

   A list of possible locations appears.
4. Select **Insert Before** or **Insert After**.

   The Specify Type window appears.

Figure 12.2 Specify Type Window

5. Do one of the following:
   - To create a button, select **Command**.
   - To create a menu within a menu, select **Submenu**. (Not available for buttons.)
   - To create divider between menu items or toolbar buttons, select **Separator**.
6. Click **OK**.

   The new untitled button, submenu, or separator is added.

Step 2: Specify the Caption, Tooltip, and Internal Name

Buttons have several basic properties:

- The caption appears in the View > Toolbars list, which lets you show or hide the toolbar. Give each toolbar a unique name. JMP merges toolbars with the same name after you close and reopen the Menu Editor.
- The tooltip appears when you place the cursor over a menu item or button.
- The internal name, which is case insensitive. JMP identifies the location of an item by its internal name rather than its caption.

Submenu items include many of the same General properties as buttons, except that submenu items do not have tooltips.

To specify these properties, follow these steps:

1. With the Menu Editor still opened, select the menu item or button.

   The General properties appear on the right.
2. Enter the internal name next to **Internal name**.
3. Enter a unique name for the menu item or button in the Caption box.
4. Enter a description for button in the Tip box.
5. (Optional) To specify the translation for the caption or tip, do the following:
   - Click the **Localize** button for the item that you want to customize.
   - Select the language.
   - Enter the translation next to Text.
   - Click **OK**.
6. Customize the untitled button or menu item as described in the following procedure.

Step 3: Assign Functionality

Clicking a menu item or button either executes a predefined command or runs a JSL script. The script can be stored in a separate file, or you can enter the JSL in the Run this JSL area of the Menu Editor. If users have access to a central location, such as a network, you typically want to run the script from that location. This also simplifies giving users access to updated scripts. Otherwise, store the JSL in the menu item or button definition.

To associate a menu item or button with an add-in JSL script, you have two options:

- Select **Run JSL in this file**, and then browse to find the external file.
- Type the relative path to the JSL script, and select the add-in from the **Use add-in home folder** list. For example, the following command runs the mds_application.jsl script from the selected add-in:

  ```
  $ADDIN_HOME(com.jmp.mdswithr)/mds_application.jsl
  ```

In addition, you can select **Use the "Here" namespace for unqualified JSL variable names** if you include namespaces in the internal or external JSL script. See the *Scripting Guide* for more information about namespaces.

To assign this functionality, follow these steps:

1. With the Menu Editor still opened, select the menu item or button that you want to modify.

   The Action properties appear on the right.
2. Select the action that you want to execute.
3. To run an internal JSL script, delete the placeholder text *print("Not implemented.");* in the **Run this JSL** area and enter the JSL.
4. (Optional) If your internal or external JSL script includes namespaces, select **Use the "Here" namespace for unqualified JSL variable names**.
5. Click **Save** to save your changes.

Step 4: Show an Icon on the Menu Item or Button (Optional)

New menu items have no icon next to the item. New buttons show up as blue squares on the toolbars. You can assign a descriptive icon to the menu item or button. The graphic can reside on your computer or in an add-in folder. .ico, .png, .jpg, and .bmp graphics are supported.

Assigning an icon to a button is particularly important. Otherwise, the user must view the button's tooltip to see what the button does.

To show an icon on the menu item or button, follow these steps:

1. With the Menu Editor still opened, select the menu item or button.

   The Icon properties appear on the right.

2. Select one of the following options:

   None Select this option to show a blue square instead of an icon.

   Built-in icon Select this option to show an icon that JMP provides, and then select the icon from the list.

   Use image from file Select this option to show an image that you created. Click **Browse**, select the graphic, and then click **OK**.

   Use add-in home folder Select this option to display an icon from an add-in to the left of the menu item. After you select the checkbox, select the add-in, click **Browse**, and then select the icon. The path to the image is displayed in the **Use image from file** field. The image can be an ICO, PNG, JPG, or BMP graphic.

3. Click **Save** to save your changes.

Step 5: Assign a Shortcut Key (Optional)

A shortcut key executes an action so that you do not have to select the menu item or click the button on the toolbar. This option is also helpful when the button's toolbar is not shown, but you still want to execute the action.

Shortcuts begin with Ctrl, Ctrl+Shift Ctrl+Alt and end with a number, letter, or symbol that appears on the keyboard. The Function keys (such as F1) are also supported. Many Ctrl + *letter* and Function shortcuts are already assigned in JMP, but you can reassign them to your buttons if you want.

Shortcut keys appear in menus next to each menu item.

To assign a shortcut key, follow these steps:

1. With the Menu Editor still opened, select the menu item or button.

   The Shortcuts properties appear on the right.

2. Do one of the following:
   - To assign a new shortcut, click in the **New shortcut** area and press the shortcut keys. If the keystrokes then appear in the **Currently assign to** box, press another combination of shortcut keys. (You do not have to delete the keystrokes before pressing the shortcut keys.)
   - To change a shortcut that you previously specified, click **Remove**, click in the **New shortcut** area and press the shortcut keys.

3. Click **Assign**.

   The shortcut appears in the **Current shortcuts** list.

4. Click **Save** to save your changes.

To remove a shortcut, select the shortcut and click **Remove**.

Set User Permissions for Customized Toolbars and Buttons

Use the JMP Menu Editor to specify which users can see your customized toolbars and buttons. Your changes are saved in a text file with the .jmpcust extension called a *customization set*. By default, your customizations are shown only to the current user; other users who log on to the computer and open JMP do not see your personal toolbars and menus.

All customization sets include the built-in JMP buttons and menus. You can also base customization sets on other customization sets. For example, you might create a new customization set that shows the JMP add-in menus and toolbars along with your modifications.

When you modify the **Current user** customization set, JMP creates a backup file of the set. The file is located in your Windows Users folder within the JMP or JMPPro folder.

C:/Users/<user name>/AppData/Roaming/SAS/JMP/16/

When you modify the **All users** customization set, JMP creates a backup file of the set. The file is located in the Windows All Users folder within the JMP or JMPPro folder.

C:/Users/All Users/SAS/JMP/16/

The All Users folder does not appear when you try to browse for it, so you must type the path into the Windows Explorer address field.

Note: To find the customization set files in Windows Explorer, show all hidden files in the Windows Explorer Folder Options. See your operating system documentation for details.

To change the customization set, follow these steps:

1. Select **View > Customize > Menus and Toolbars**.

   The Menu Editor appears.
2. Click **Change**.

   When you try to change the customization set, and changes to the selected set are not saved, you are prompted to save the changes. Click **Save Changes**.
3. Select the customization set in the Customization Set to Modify area. (By default, changes apply only to the **Current user** customization set.) "Customization Set Options" describes the options.
4. (Optional) To change the set on which the selected customization set is based, select the set in the Starting Set area. (Not available for the *All users* customization set.) "Customization Set Options" describes the options.
5. Click **OK**.

Rearrange Toolbars

On Windows, there are two ways to rearrange toolbars:

- To rearrange toolbars quickly for the current user, drag the toolbar above, below, to the left, or to the right of the adjacent toolbar.
- To specify the new location rather than drag the toolbar, right-click the toolbar and select a position from the Location menu. This method lets you move toolbars to the top or bottom of the window. You can also position the toolbars vertically on the left or right side of the window.

  In some windows, the toolbar is hidden by default. Specifying a new location shows the toolbar in all windows of that type. On Windows, you can change the toolbar preferences to always show toolbars. See "Windows Preferences"

To drag a toolbar

1. Point to the left corner of the toolbar until the Move cursor appears.
2. Drag the toolbar to the new location. When a toolbar is too wide to show completely, click the arrow to show all of the buttons, as shown in Figure 12.3.

Figure 12.3 Expanding a Toolbar

To position the toolbar

1. Right-click the toolbar that you want to move.
2. Select **Location**, and then select the new position.

Note: After you move a toolbar to the left, right, or bottom positions, you can add other toolbars to the same position. For example, you might relocate the **File_Edit** toolbar to the right side of the JMP window. To add other toolbars on the right side, right-click the bar and select the toolbar that you want to add.

Copy and Paste Menus, Menu Items, Toolbars, and Buttons

Copying existing items is a shortcut to creating new items in a menu or toolbar. This option lets you change only a few settings in the new item when possible.

To copy and paste menus and menu items

1. Select **View > Customize > Menus and Toolbars**.

The Menu Editor appears.

2. (Optional) Change the customization set to control which users see your customizations. See "Customization Set Options".
3. Right-click the menu or menu item that you want to copy and select **Copy**.
4. Right-click the menu or menu item before, after, or into which the item will appear and select **Paste**.

   One of the following occurs:

   – A list of possible locations appears.
   – The menu or menu item appears below the selected item. Skip to step 7.
5. Select the location of the menu by doing one of the following:
   – To paste the item *before* the selected menu, select **Paste before**.
   – To paste the menu *after* the selected menu, select **Paste after**.
   – To paste the menu *inside* the selected menu, select **Paste into**. (Available only when you select a menu as the new location.)
   – To cancel the action, select **Cancel**.

   The item appears as you indicated.
6. Modify the item.
7. Click **Save** to save your changes.

See "Customize the Main Menu Bar" and "Create Menu Items and Toolbar Buttons" for more information about modifying the items.

To copy and paste toolbars and buttons

1. Select **View > Customize > Menus and Toolbars**.

   The Menu Editor appears.
2. (Optional) Change the customization set to control which users see your customizations. See "Customization Set Options".
3. Right-click the toolbar or button that you want to copy and select **Copy**.
4. Do one of the following:
   – To paste a toolbar, right-click **Toolbar** and select **Paste**. The toolbar appears at the end of the list of toolbars.
   – To paste a button, right-click the button before or after which you want the new button, and select **Paste**. If you selected the first button on the toolbar, select the location of the button. Otherwise, the button is pasted below the selected button.
5. Modify the item.
6. Click **Save** to save your changes.

See "Customize Toolbars" and "Create Menu Items and Toolbar Buttons" for more information about modifying the items.

Rearrange Custom Menus, Menu Items, and Buttons

On Windows, you can rearrange the order of menus, menu items, and buttons that you create. For example, under the **File > New** menu, you could move your custom menu before the **Data Table** command.

In the item's General properties, the Source determines whether you can move the item. You can move items if the Source is *Custom Item*. Built-in items cannot be moved. Items that are defined in other customization sets can be moved only in that customization set. See "Customization Set Options".

Note: The right-click menu has options for cutting and pasting items. To cut an item, you can also press Ctrl+X, or press Shift+Delete. To paste, press Ctrl+V, or press Shift+Insert.

To rearrange custom menus and menu items

1. Select **View > Customize > Menus and Toolbars**.

   The Main Menu list appears on the left.
2. Do the following:
   - Select the item that you want to move.
   - Verify that the Source is *Custom Item*.
   - If the Source is *All Users* or *JMP Add-In*, select the specified customization set and verify that the Source is *Custom Item*.
3. Right-click the custom menu or menu item that you want to move and select **Cut**.
4. Right-click the new location of the cut item and select **Paste**.

   One of the following occurs:
   - A list appears with the possible locations of the cut item. This occurs when you select a menu (such as **File** or **New**) or the first item in a menu (such as **Data Table** in the **File > New** menu).
   - The cut menu item appears below the selected menu item. Skip to step 6.
5. Select the location of the cut item by doing one of the following:
   - To paste the cut item *before* the selected item, select **Paste before**.
   - To paste the cut item *after* the selected item, select **Paste after**.
   - To paste the cut item *inside* the selected menu, select **Paste into**. (Available only when you select a menu as the new location.)

– To cancel the move, select **Cancel**.

The item appears as you indicated.

6. Click **Save** to save your changes.

To rearrange custom buttons

1. Select **View > Customize > Menus and Toolbars**.

   The Toolbars list appears on the left.

2. Do the following:
   – Select the custom button that you want to move.
   – Verify that the Source is *Custom Item*.
   – If the Source is *All Users* or *JMP Add-In*, select the specified customization set and verify that the Source is *Custom Item*.
3. Right-click the custom button that you want to move and select **Cut**.
4. Right-click the new location of the cut button.

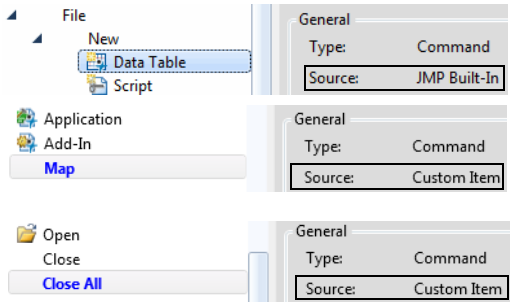
   One of the following occurs:

   – A list appears with the possible locations of the cut button. This occurs when you select the first button on the toolbar (such as **New Data Table** in the **File_Edit** toolbar).
   – The button is pasted at the end of the toolbar. This occurs when you select the name of the toolbar. For example, when you select **File_Edit** and paste, the button appears after the last button, **Run Script**. Skip to step 6.
   – The cut button appears after the selected button. Skip to step 6.
5. Select the location of the cut button by doing one of the following:
   – To paste the cut button *before* the selected button, select **Paste before**.
   – To paste the cut button *after* the selected button, select **Paste after**.
6. Click **Save** to save your changes.

Delete Custom Items

Rather than temporarily showing or hiding toolbars, buttons, menus, and menu items, you can delete items that you created to remove them permanently from JMP. The Source determines whether you can delete the item and which customization set contains the item. Figure 12.4 shows examples of customization Source types.

Figure 12.4 Examples of Customization Sources



The item cannot be deleted.

Delete the item in the current customization set.

Delete the item from the *All Users* customization set.

Empty menus are not supported. When you delete the only item in a menu, the entire menu is deleted, not just the selected item.

Note: Make sure that you really want to delete custom items. They are removed immediately without confirmation.

To delete a custom item, follow these steps:

1. Select **View > Customize > Menus and Toolbars**.

   The Menu Editor appears.
2. Select the item that you want to delete.
3. Verify whether the item can be deleted, and change the customization set, if necessary.
4. Right-click the selected item and select **Delete**.

   The item is immediately deleted.
5. Click **Save**.

   Your changes are saved.

Show or Hide Custom Items

In the Menu Editor, you show or hide toolbars, buttons, menus, and menu items in specific customization sets. The item's Source determines where you show or hide the item. For example:

- When the Source is *All Users*, change the customization set to **All users** to hide the item from all users.
- When the Source is *JMP Add-In*, change the customization set to **JMP Add-In** followed by the name of the add-in. The item is shown or hidden in all sets that include the selected JMP add-in.

If you do not change the customization set, the item is shown or hidden only in the currently selected set. See "Customization Set Options" for more information about customization sets.

Note: To quickly hide or show toolbars for the current user, select or deselect them from the **View > Toolbars** list.

To show and hide items in a customization set, follow these steps:

1. Select **View > Customize > Menus and Toolbars**.

   The Menu Editor appears.
2. Select the customization set that you want to modify.
3. Select the item that you want to show or hide.
4. Select or deselect **Hidden** in the General properties. See "Step 2: Specify the Caption and Internal Name" for more information about the General properties.
5. Click **Save** to save your changes.

Import Customizations

Changes to toolbars and menus are stored in customization sets, or plain text files. You open a customization set in JMP to import your modified toolbars and menus. The customizations are then shown in the Menu Editor. See "Customization Set Options" for more information about the menu customization files.

To import customizations, follow these steps:

1. Select **File > Open**.
2. In the File name list, select **All JMP Files** (or **JMP Menu Files**) and then select the customization set.
3. Click **Open**.

   A confirmation is displayed, stating that the customizations have been imported. The Menu Editor is also opened.
4. Display the Menu Editor and click **OK** to save the changes, or click **Cancel** to discard them.

Remove Customizations from JMP

As you modify items, you discard unsaved changes by clicking the Menu Editor's **Cancel** button and clicking **Yes** to confirm. After saving customizations, you can also remove all customizations and revert to the original menus and toolbars.

To remove all of the current user's customizations

1. Select **View > Customize > Revert to Factory Defaults**.
2. Click **Yes** to remove the current user's customizations.

When JMP finds customizations from a previous version of JMP, a confirmation appears. Do one of the following:

- Select **No** to avoid adding those customizations to the current installation of JMP.
- Select **Yes** to add those customizations to the current installation of JMP.

To remove customizations from another customization set

1. Select **View > Customize > Menus and Toolbars**.

   The Menu Editor appears.
2. Click **Change**.
3. Select the customization set, and click **OK**.
4. Click **Revert All**.

   A confirmation window appears.
5. Click **Yes** to remove your customizations.

To remove customizations from the selected item

1. Select **View > Customize > Menus and Toolbars**.

   The Menu Editor appears.
2. (Optional) Select the customization set that contains the item.
3. Select the item that you want to modify.
4. Do one of the following:
   - To restore the *previous* properties of an *unsaved* item, click **Reset**.
   - To restore the *original* properties of a *built-in* item, click **Revert All**, and then click **Yes** to confirm.
5. Click **Save** to save your changes.

Customize Toolbars on macOS

On macOS, you can set up toolbars to display only the icons that you need. Icons are available based on the type of window. For example, in a script window, icons for options such as reformatting, encrypting, and running the script are available. Table icons are available only for data table toolbars.

You can specify whether the icon, icon and icon name, or just the name appear on the toolbar. Displaying small icons is also an option.

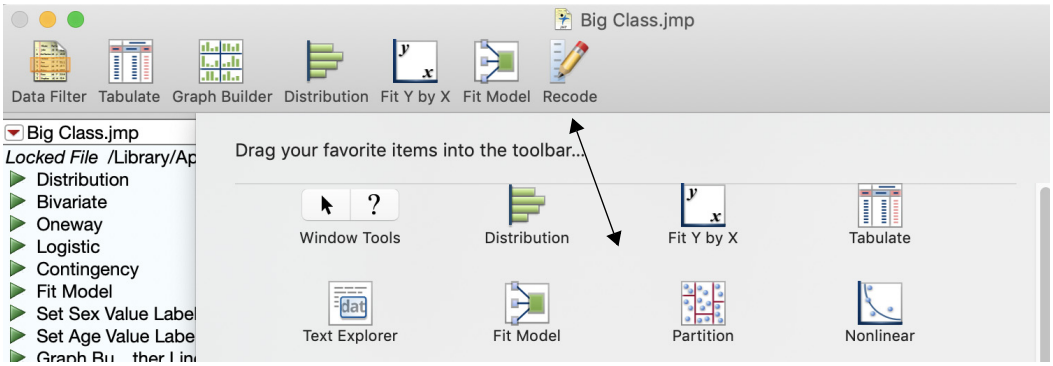
To add, remove, or rearrange toolbar icons

1. Open the type of window whose toolbar you want to customize.
2. Select **View > Customize Toolbar**.

   A window appears that shows icons relevant to the current type of window.
3. Do any of the following:
   - To add an icon to the toolbar, drag the icon from the window onto the toolbar.
   - To move an icon on the toolbar, drag the icon to its new location onto the toolbar.
   - To remove an icon from the toolbar, drag the icon from the toolbar onto the window.

   Your changes are applied to the current window and all open windows of the same type.

Figure 12.5 Add, Rearrange, and Remove Toolbar Buttons on macOS



4. Click **Done**.

   When a toolbar is too wide to display completely, click the right arrows on the toolbar to show and then select other icons. Alternatively, you can click and drag the window until the entire toolbar appears.

Note: To display the original set of toolbar icons, drag the default set from the bottom of the toolbar customization window to the toolbar.

To change the appearance of toolbar icons

1. Open the type of window whose toolbar you want to customize.
2. Select **View > Customize Toolbar**.

   A window containing toolbar icons appears.
3. At the bottom of the window, select one of the following options from the Show list:
   - To display only an icon, select **Icon Only**.
   - To display an icon and its name, select **Icon and Text**.

– To display only the icon name, select **Text Only**.

4. To display small or standard icons, select or deselect **Use small size**.

   Your changes are applied to the current window and all open windows of the same type.

5. Click **Done**.

Customize Menus on macOS

Customize JMP menus to quickly access files or modify existing menu items. Selecting a menu item runs a JSL script from your computer or executes the JSL commands that you specify. For example, you can create a menu item to open a frequently used script. This option is an alternative to adding the script as a favorite on the Home Window.

Features include adding and renaming menu items, hiding menu items, inserting menu item separators, and associating a menu item with JSL.

JMP customizations are stored in <username>/Library/Application Support/JMP/16/usercust.jmpcust.

Note: You cannot delete default JMP menus or delete submenus (which open to reveal other menu items). Default and custom submenus also cannot be renamed.

Example of Adding a Menu Item

In this example, create a menu item that runs a script.

1. Select **View > Customize Menus.**
2. In the JMP Menu Editor, select **File** and then **Open**.
3. Press Control and then select **Add New Menu Item after**.
4. Type *What is the Current Time?* in the Menu Label box.
5. Select **Run JSL in this file:** and select Library/Application Support/JMP/16/Samples/Scripts/timeAnnouncer.jsl.
6. Click **Add** and close the Menu Editor window.
7. Select **File > What is the Current Time?**.

   The Scheduler window appears, and the current time is spoken.

Example of Adding a Submenu Item

A submenu opens to reveal submenu items. In this example, create a submenu item that opens a data table.

1. Select **View > Customize Menus.**

2. In the JMP Menu Editor, select **File** and then **Open**.
3. Press Control and select **Add New Submenu after**.
4. Type Open My Data Tables and select **JSL Commands**.

Note: Selecting **JSL Script** lets you select a JSL script from your computer.

5. Click **Add**.
6. Select **Open My Data Tables**, delete **Subitem placeholder** in the Menu Label box, and then type Open Napoleons March.jmp.
7. In the Run this JSL box, type `Open("$SAMPLE_DATA/Napoleons March.jmp");`.
8. Click **Save Changes**.
9. Select the **File** menu to see the new menu item.

Tip: If the new menu item appears in the menu but not the menu editor, close the menu editor window and reopen it.

To delete a custom menu or submenu item

- Delete one custom menu item by selecting the item and selecting **Delete Selection** from the options list.
- Delete all custom menu items by selecting **Revert All** from the options list.
- To delete items within a submenu, you must delete the entire submenu. Empty menus are not permitted. Select the submenu and select **Delete Selection** from the options list.

Customize Menus and Toolbars for Add-Ins

To add new features to JMP, you can write the scripts and compile files to create an add-in. One way to compile the scripts is to use Add-In Builder as described in the *Scripting Guide*.

Add-In Builder provides a menu editor that enables you to create menu items in the Add-Ins menu. You can also use the JMP Menu Editor to create menu items and add toolbar buttons for an add-in.

Chapter 13

JMP Preferences

The Preferences Window

JMP preferences enable you to specify general and specific settings and save the settings.

To change preferences

1. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS). The window in Figure 13.1 appears. The previous preferences page that you selected appears.
2. Click a category and make selections. Click **Apply** to see the results, and then click **OK**.

Note: When you upgrade to JMP 16 Subscription, changes to preferences are maintained. In other JMP versions, you must customize the preferences after each upgrade.

To search preferences

1. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS).
2. In the Filter box at the top, enter your text.

   Text matches are highlighted.
3. Under Preference Group, click a category to see its results.

A search term can show up in these places:

- the preference itself
- a tooltip for the preference
- a list of options for a preference

To reset preferences to the default settings

1. Select **File > Preferences** (Windows) or **JMP > Preferences** (macOS).
2. Click **Reset All**, and then click **OK**.

Contents

Overview of JMP Preferences 667
General JMP Preferences 667
Preferences for JMP Projects 671
Preferences for JMP Reports 672
Preferences for JMP Graphs 675
Style Preferences for JMP Graphs and Reports 679
Preferences for Data Tables 682
Preferences for JMP Data Filters 687
Preferences by JMP Platform 688
Print Preferences 690
Preferences for Importing and Exporting Text Files 691
Preferences for Third-Party Data 696
Windows Preferences 698
Mac OS Preferences 702
Fonts Preferences 704
Communications Preferences 707
Preferences for Internet Options 707
Preferences for File Locations 708
Preferences for the JSL Script Editor 710
Preferences for SAS Integration 713
Preferences for Updating JMP 716
Preferences for the JSL Debugger 718
Menu Preferences 719
Preferences for the Query Builder 720
Preferences for Recoding Columns 721
Preferences for the Column Switcher 723
Preferences for the Log 725

Overview of JMP Preferences

The Preferences command on the File menu (on the JMP menu on macOS) displays the Preferences window. See Figure 13.1. Each category is described in this chapter.

On Windows, changes that you make to preferences are stored in a version-specific JMP.PFS file located in one of the following folders:

C:/Users/<user name>/AppData/Roaming/SAS/JMP/16/

C:/Users/<user name>/AppData/Roaming/SAS/JMPPro/16/

On macOS, the preferences are stored in com.sas.jmp.plist, which is located in ~/Library/Preferences/.

JMP looks for this file when you open the program and considers your changes to be the factory defaults. And all subsequent changes are stored in this file.

General JMP Preferences

Most General preferences customize your JMP session at start-up. Others set preferences for file handling and the general appearance of the JMP workspace.

Figure 13.1 General Preferences

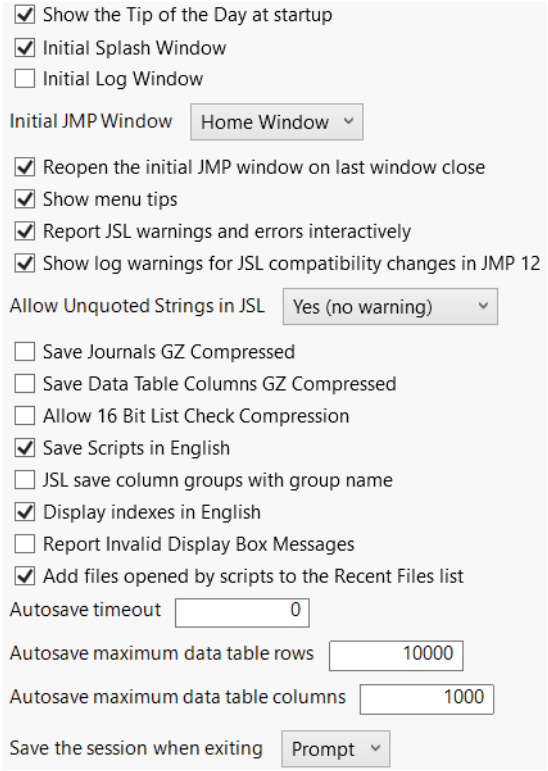


Table 13.1 Preferences on the General Page

Preference	Description
Show Tip of the Day at startup	Select this option to show the Tip of the Day window at start-up. This option is selected by default.
Initial Splash Window	Select this option to show the initial splash window at start-up. This option is selected by default.
Initial Log Window	(Windows only) Select this option to open the log when JMP opens.
Initial JMP Starter Window	(macOS only) Select this option to show the JMP Starter window every time you start JMP. This option is selected by default.
Initial JMP Window	(Windows only) Select one of the listed windows as the default window. When JMP starts, this window appears. The Home Window is the default window.

Table 13.1 Preferences on the General Page *(Continued)*

Preference	Description
Home Window	(macOS only) Select this option to show the JMP Home window when you start JMP on macOS. This option is selected by default.
Reopen the initial JMP window on last window close	(Windows only) Select this option to open whichever window you have set as the Initial JMP Window when you close the last JMP window. If this option is turned off, closing the last JMP window also quits JMP. This option is selected by default.
Show menu tips	(Windows only) Select this option to see tooltips on main menu options and options in red triangle menus. This option is selected by default.
Report JSL warnings and errors interactively	Shows script errors in a new window rather than only in the log. This option is selected by default.
Show log warnings for JSL compatibility changes in JMP 12	Select this option to print warnings about JSL compatibility to the log. This option is selected by default.
Allow Unquoted Strings in JSL	Select this option to print a warning to the log when a string value in a platform message is unquoted. Quoted string values are preferred over literal names. **Yes (no warning)** Allows unquoted string values and does not print a warning to the log. This option is selected by default. **Yes (warning)** Allows unquoted string values and prints a warning to the log. **No** Ignores the unquoted string value and does not print a warning to the log.
Save Journals GZ Compressed	Select this option to save JMP journals in a compressed format to save disk space. This option is deselected by default.
Save Data Table Columns GZ Compressed	Select this option to allow JMP to save data tables using GZ compression. For backward compatibility, JMP 10 can read the compressed files but not save them. This option is deselected by default.
Allow 16 Bit List Check Compression	Compresses a column that has less than or equal to 65,535 unique values. Adds the List Check property to the column. This option is deselected by default.

Table 13.1 Preferences on the General Page *(Continued)*

Preference	Description
Save Scripts in English	Non-English versions of JMP can save scripts in either English or your local language. User-specified values (such as column names and text strings) in these scripts appear as they do in the data table. However, command words (such as `Distribution` and `Set Value`) appear in English rather than the local language in order for the script to run on JMP in English. JMP properly displays non-Roman characters (such as Japanese variable names) in JMP in English when the fonts support the necessary characters. The option is selected by default. Deselect this option to save scripts in your local language. Note that these scripts run correctly *only* if JMP is running in the same language.
JSL save column groups with group name	Select this option to specify the column group name in a JSL script that you save from a platform. By default, the script contains the column names, not the column group name.
Display indexes in English	Shows sections of the JMP Indexes in English or the current locale's language. The option is selected by default. When you deselect it, the following items are displayed in the current locale's language: • The message list for objects • The message list for display boxes. • The category list for JSL functions. **Note:** The description of each index item is always displayed in the current locale's language. Examples are always in English.
Report Invalid Display Box Messages	Sends information about invalid display box messages to the log. This option is selected by default. This option can be useful during script development, but can cause unwanted log messages for existing scripts.
Add files opened by scripts to the Recent Files list	(Windows only) Select this option to include files opened by the JSL `Open()` command in the Home Window Recent Files pane and the File > Recent Files list. This option is selected by default.
Autosave timeout	Automatically saves the contents of opened data tables, journals, reports, projects, and scripts except for untitled files at the specified interval. The default setting is 0 minutes, which indicates no autosave.

Table 13.1 Preferences on the General Page *(Continued)*

Preference	Description
Autosave maximum data table rows	Autosaves data tables if the number of rows does not exceed the specified value. 10,000 is the default setting.
Autosave maximum data table columns	Autosaves data tables if the number of columns does not exceed the specified value. 1,000 is the default setting.
Save the session when exiting	This option enables you to save the state of the JMP window when exiting JMP. When opening JMP, the saved state is restored, including any open files and windows. The default setting is Prompt.

Preferences for JMP Projects

Projects preferences enable you to control how new empty projects are created, which tool windows appear, and more.

Figure 13.2 Projects Preferences

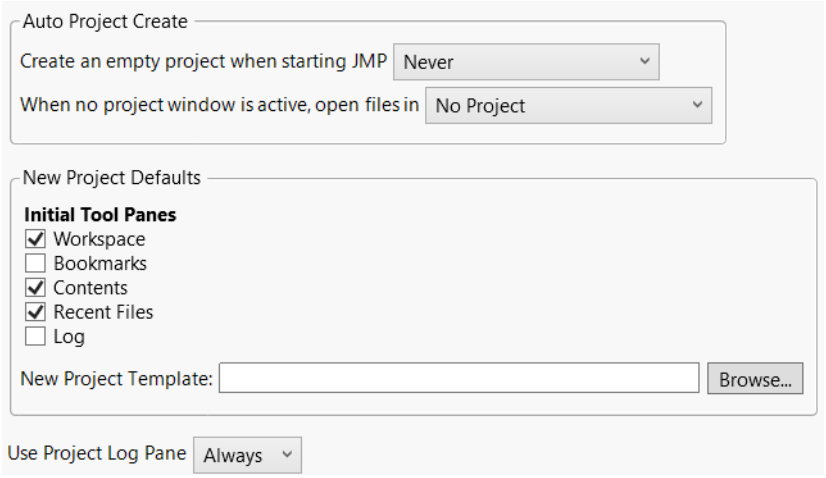


Table 13.2 Preferences on the Projects Page

Preference	Description
Create an empty project when starting JMP	Creates an empty project when you start JMP: Never, Always, or If no other project is open. Never is the default setting.
When no project window is active, open files in	Specifies how a new project is automatically created. **No Project** Doesn't put the files in a project. This is the default setting. **Open Project or No Project** Puts the files in an open project if found or no project. **Open Project or New Project** Puts the files in an open project if found or a new project. **New Project** Puts the files in a new project.
Initial Tool Panes	Specifies which tool panes initially appear in a project. The Workspace, Contents, and Recent Files tools appear by default.
New Project Template	Specifies the project that you want to use as a template for new projects. The new project has the same window layout, linked documents, graphs, tool windows, and other project elements as the template.
Use Project Log Pane	Specifies when you want to view the project log rather than use the JMP log. Always is the default setting.

Preferences for JMP Reports

Report preferences customize the appearance of reports.

Figure 13.3 Reports Preferences

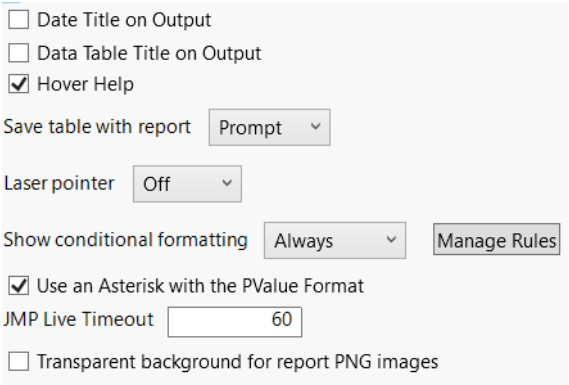


Table 13.3 Preferences on the Reports Page

Preference	Description
Date Title on Output	Select this option to display the date and time the analysis occurred in your report windows. This option is deselected by default.
Data Table Title on Output	Select this option to display the name of the data table and notes, if there are any, at the top of the report. This option is deselected by default.
Hover Help	On some numeric output, JMP provides tooltip-style help when you circle the mouse over a result. Select this option to see a tooltip where it is available. This option is selected by default. **Note:** If you deselect this option, hover label graphs no longer appear.

Table 13.3 Preferences on the Reports Page *(Continued)*

Preference	Description
Save table with report	Select one of the options from the menu to determine how a table is saved to a report: **Embed** When you save a report, the table is embedded into the report. Choose this option if you want to share the report with others. The data table that is stored in the report is reopened (unchanged) each time the report is opened. If you make changes to the table, they are not saved into the report unless you re-save the report. **Separate** When you save a report, the table is referenced in the report. **Prompt** When you save a report, you are prompted to specify how to save the report: either to embed the table within the report, or to add a reference to the table in the report. This is the default setting.
Laser pointer	JMP has a built-in laser pointer that enables you to visually emphasize parts of a report. The option is deselected by default. To turn it on, select a color for the laser pointer from the list.
Show conditional formatting	Conditionally formats the color of the text that represents the values. Available for correlation values, *p*-values, and factor pattern values. Select one of the options from the menu: **Always** Conditional formatting is always applied to correlation values, *p*-values, and factor pattern values. This is the default setting. **Screen Only** Conditional formatting is always applied to correlation values, *p*-values, and factor pattern values. However, the conditional formatting does not show when the report is printed. **Never** Conditional formatting is never applied to correlation values, *p*-values, and factor pattern values.

Table 13.3 Preferences on the Reports Page *(Continued)*

Preference	Description
Manage Rules	Creates custom conditional formatting rules that can be applied to numeric columns or matrices. See "Use Conditional Formatting in Reports". **Note:** Verify the **Use an Asterisk with the PValue Format** preference to ensure that the asterisk (*) is or is not used for any *p*-value conditions.
Use an Asterisk with the PValue Format	Select this option to display an asterisk (*) next to significant *p*-values. This option is selected by default. **Note:** Verify any conditional formatting rules for PValue to ensure asterisk (*) is not used for any defined conditions.
JMP Live Timeout	Species the timeout value for publishing to JMP Live. The default time is 60 seconds.
Transparent background for report PNG images	Select this option to display a transparent background when a report is saved as a PNG file. This option is deselected by default.

Preferences for JMP Graphs

Graph preferences customize the appearance of graphs. See "Style Preferences for JMP Graphs and Reports" for more information about configuring the appearance of graphs.

Figure 13.4 Graphs Preferences

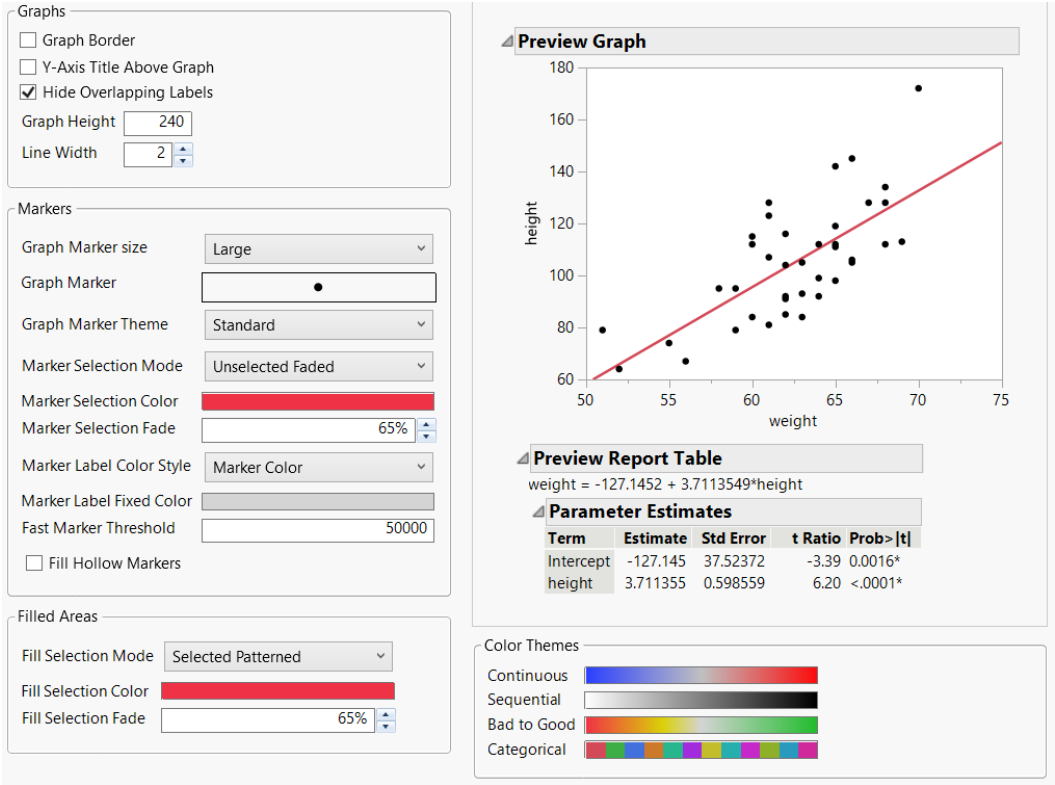


Table 13.4 Preferences on the Graphs Page

Preference	Description
Graph Border	Draws a border line around the area of the graph. This option is deselected by default.
Y-Axis Title Above Graph	Shows the Title for the Y-Axis above the line rather than beside the line. This option is deselected by default.
Hide Overlapping Labels	Select this option to hide overlapping chart labels. This option is selected by default.

Table 13.4 Preferences on the Graphs Page *(Continued)*

Preference	Description
Graph Height	Sets the default height (in pixels) for a graph. The default setting is 240 pixels. Some platforms take the value and multiply it by a certain amount. **Note:** With Graph Builder windows, specifying the Graph Height enlarges the window proportionally, but the exact value is not applied.
Line Width	Sets the default width (in pixels) for lines that pertain to content. The default setting is two pixels. **Note:** This preference does not apply to grid lines.
Graph Marker Size	Select a default size for the markers in graphs: Dot, Small, Medium, Large, XL, XXL, or XXXL. This preference is applied when the marker size is set to Preferred on a graph. The default setting is Large. When the number of markers in a graph exceeds the Fast Marker Threshold, the graph's Marker Drawing Mode is set to Fast. which draws dot-sized markers.
Graph Marker	Select a default marker shape for the markers in graphs. The default setting is a small dot.
Graph Marker Theme	Select a default theme for the markers in graphs when you mark by row or column. The default setting is Standard.
Marker Selection Mode	Select the default formatting for selected markers. The default setting is Unselected Faded. See "Change the Selection Mode of Markers".
Marker Selection Color	When the Marker Selection Mode is Selected Same Color, this setting applies the specified color to selected markers. The default color is Red.
Marker Selection Fade	When the Marker Selection Mode is Unselected Faded, this setting fades the unselected markers by the specified amount (in percent). The default setting is 65%.

Table 13.4 Preferences on the Graphs Page *(Continued)*

Preference	Description
Marker Label Color Style	Select the color of the marker label that appears on a graph when you label a column in the data table. **Marker Color** The marker label is the same color as the marker. This option is selected by default. **Marker Color Faded** The marker label is faded according to the Marker Selection Fade preference. **Fixed Color** The marker label color is as specified. The Marker Label Fixed Color preference determines the color.
Marker Label Fixed Color	Select the color of fixed-color marker labels, an option that you select in the Marker Label Color Style preference. The default color is gray.
Fast Marker Threshold	When JMP refreshes a report window, it can draw markers or points on a plot at two different speeds: normal and fast. If JMP is in normal drawing mode, and the number of markers on a graph are more than the specified threshold number, JMP automatically switches to fast mode. Enter the number of markers that separate normal and fast mode. When the number of markers in a graph exceeds the Fast Marker Threshold, the graph's Marker Drawing Mode is set to Fast. which draws dot-sized markers. The default setting is 50,000.
Fill Hollow Markers	Select this option to apply an opaque background to markers. This option is deselected by default.

Table 13.4 Preferences on the Graphs Page *(Continued)*

Preference	Description
Fill Selection Mode	Select the default highlight for selected rows in graphic objects: **Selected Patterned** A striped pattern (default) **Selected Darkened** A solid darker color **Selected Outlined** An outline **Selected Same Color** The Fill Selection Color in the preferences formats the selected object. **Unselected Faded** Unselected rows are faded, and selected rows are the original color.
Fill Selection Color	Select the default color of selected rows in graphic objects when the Fill Selection Mode is "Selected Same Color". The default color is red.
Continuous Color Theme	Select a default color theme for continuous data. The default theme is Blue to Gray to Red. See "Create Color Themes in Graphs".
Sequential Color Theme	Select the default color theme for sequential data that do not have a meaningful midpoint. The default theme is White to Black.
Bad to Good Color Theme	Select the default color theme for diverging data that ranges from positive to negative data. The default theme is Stoplight Bad to Good.
Categorical Color Theme	Select a default color theme for categorical data. The default theme is JMP Default. See "Create Color Themes in Graphs".

Style Preferences for JMP Graphs and Reports

The Styles page enables you to customize colors, tick marks, grid lines, and report tables.

See "Preferences for JMP Graphs" for more information about configuring the appearance of graphs.

Figure 13.5 Styles Preferences

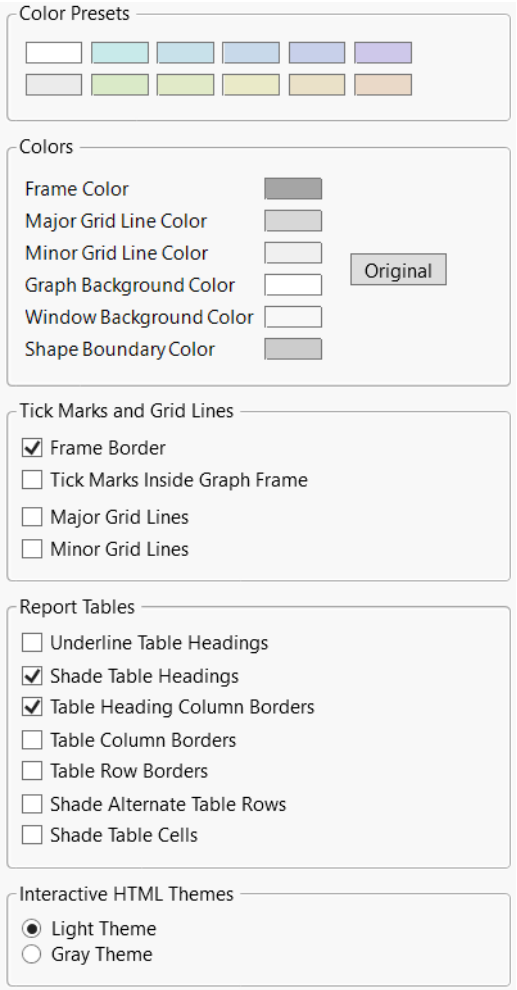


Table 13.5 Preferences on the Styles Page

Preference	Description
Color Presets	The Color Presets provides a set of predefined colors for use in customizing the appearance of reports, graphs, charts, and so on Tints of the selected color are applied to the various elements. **Note:** The selected color is applied also to windows.
Frame Color	Defines the color applied to a graph frame and tick marks. Click to change the default color.

Table 13.5 Preferences on the Styles Page *(Continued)*

Preference	Description
Major Grid Line Color	Defines the major grid line color. Click to change the default color.
Minor Grid Line Color	Defines the minor grid line color. Click to change the default color.
Graph Background Color	Click the color box to select a background color for all graphs. Click to change the default color.
Window Background Color	Click the color box to select a background color for all reports and data tables. Click to change the default color.
Shape Boundary Color	Click the color box to select the color for shape files boundaries.
Original	Click to restore default color settings.
Frame Border	Shows borders only on the axes. This option is selected by default.
Tick Marks Inside Graph Frame	Shows axis tick marks inside the graph frame. This option is deselected by default.
Major Grid Lines	Shows major grid lines on graphs. This option is deselected by default.
Minor Grid Lines	Shows minor grid lines on graphs. This option is deselected by default.
Underline Table Headings	Shows border under table headings. This option is deselected by default.
Shade Table Headings	Shows table headings with shading. This option is selected by default.
Table Heading Column Borders	Shows divider lines between columns in table headings. This option is selected by default.
Table Column Borders	Shows borders between table columns. This option is deselected by default.
Table Row Borders	Shows borders between table rows. This option is deselected by default.
Shade Alternate Table Rows	Shows alternate table rows with shading. This option is deselected by default.
Shade Table Cells	Shows table cells with shading. This option is deselected by default.

Table 13.5 Preferences on the Styles Page *(Continued)*

Preference	Description
Interactive HTML Themes	Specifies the color theme for interactive HTML pages. The default theme is Light Theme.

Preferences for Data Tables

Table preferences customize JMP data tables, including formula handling, appearance, and compatibility with SAS data sets.

Figure 13.6 Tables Preferences

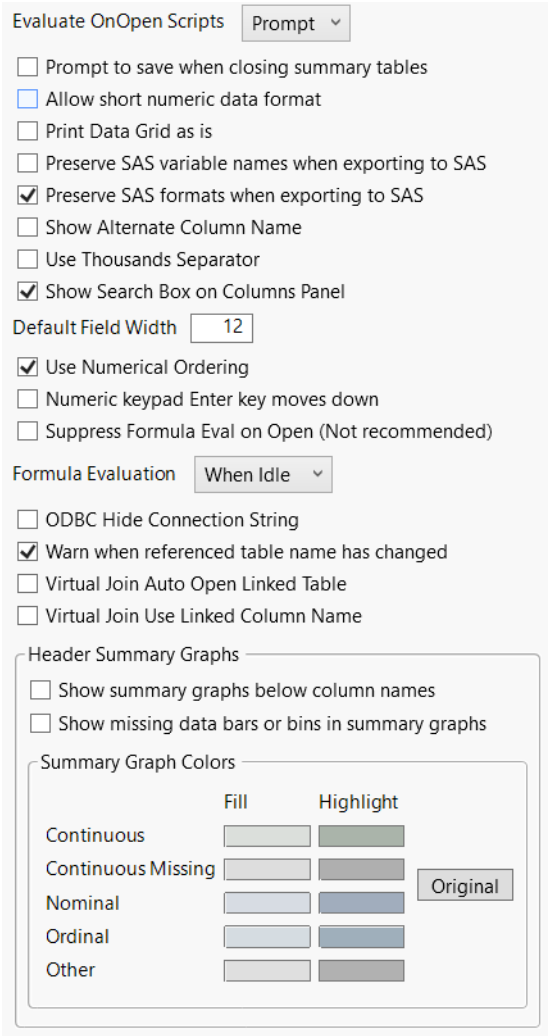


Table 13.6 Preferences on the Tables Page

Preference	Description
Evaluate OnOpen Scripts	If you save a script to a data table and name it `OnOpen` or `On Open`, JMP can automatically run the script whenever you open the data table. `On Open` scripts that execute other programs are never run. Prompt is selected by default. Your choice is remembered each time you open the data table in the current JMP session. **Always** Allows `On Open` scripts to run without prompting. **Never** Prevents `On Open` scripts from automatically running when a data table is opened.
Prompt to save when closing summary tables	Prompts you to save an unsaved linked summary table when you close it. Note that, when you select this option in the Summary window, it is selected in the Tables preferences. If you do not want to be prompted to save all linked summary tables in the future, deselect the preference. This option is deselected by default.
Allow short numeric data format	JMP has the ability to store numeric data in as few as 8 bits (one byte). This option makes short-integer formats available to you when you select Cols > Column Info and assign a column a data type. When you use the correct short-integer format for your data, the numbers are not displayed differently, but the data table uses less disk space. See "The Short-Integer Format".
Print Data Grid as is	Select this option to print the JMP data table as it appears on the screen. Clear this option to resize column widths to accommodate the content width. This option is deselected by default.
Preserve SAS variable names when exporting to SAS	Select this option to use variable names that are compliant with SAS when you export a JMP data table to a SAS data set. This option is deselected by default.
Preserve SAS formats when exporting to SAS	Select this option to use formats that are compliant with SAS when you export a JMP data table to a SAS data set. This option is selected by default.

Table 13.6 Preferences on the Tables Page *(Continued)*

Preference	Description
Show Alternate Column Name	Select this option to show the name-label pair of the column. A column has name-label pair if the two column properties are defined and the Column Name equals one of the two properties. JMP recognizes the following name-label pairs: • SAS name-label pair, where the properties are "SAS Name" and "SAS Label" • SPSS name-label pair, where the properties are "SPSS Name" and "SPSS Label" • Short-long name-label pair, where the properties are "Short Name" and "Long Name" If the option is selected, then both the name and label appear in the dialog.
Use Thousands Separator	Select this option to display numbers using the locale-appropriate thousands separator. This option is deselected by default.
Show Search Box on Columns Panel	Shows a search box on the Columns panel above the column names. This option is selected by default.
Default Field Width	Changes the number of digits that appear in numeric columns. The default setting is 12.
Use Numerical Ordering	Specifies that text containing numbers be sorted in numerical order. This option is selected by default.
Numeric keypad Enter key moves down	(Windows only) When a data table cell is selected and you press Enter on the keyboard, the next table cell down is selected. Pressing either Tab or Enter on the numeric keypad located to the right of the keyboard selects the next table cell to the right. Select this option to change the behavior of the Enter key on the numeric keypad to select the next table cell down instead of to the right. This option is deselected by default.
Suppress Formula Eval on Open	Select this option to prevent columns formulas from being evaluated when you open data tables. Deselect this option to allow formulas to be evaluated when you open data tables. This option is deselected by default. This option is not recommended because typically you want to know whether a formula evaluates.

Table 13.6 Preferences on the Tables Page *(Continued)*

Preference	Description
Formula Evaluation	Specifies when formulas are evaluated. **When Idle** Evaluates the formula when JMP is not performing actions. A progress bar does not appear. This option is selected by default. **Immediate** Evaluates the formula immediately and shows a progress bar.
ODBC Hide Connection String	Select this option to have the Open Database command hide the ODBC connection settings (that is, user ID and password). See the *Scripting Guide* for additional information. This option is deselected by default.
Warn when referenced table name has changed	Shows a warning when a virtually linked (*referenced*) table name has changed. This option is selected by default.
Virtual Join Auto Open Linked Table	Automatically opens a virtually joined data table. A link to the data table is in the JMP Home window. This option is deselected by default.
Virtual Join Use Linked Column Name	Specifies that the linked column name be used in virtually joined columns that are shown in the Columns list. This option lets you specify a shorter column name (for example, Cheese instead of Cheese[Choice]). This option is deselected by default.
Header Summary Graphs	**Show summary graphs below column names** Shows a summary graph (a distribution) in the column header. Applies when the number of rows is below a threshold such as 3 million rows. This option is deselected by default. **Show missing data bars or bins in summary graphs** Specifies whether missing data bars or bins initially appear in summary graphs. To show missing value bars in an individual graph, right-click the graph and select **Missing values bars**. Missing value bars are not shown by default. **Summary Graph Colors** Specifies the color of the fill and the highlight (the color that appears when you select a bar), including by data type.

Preferences for JMP Data Filters

Data Filter preferences customize settings for data filters.

Figure 13.7 Data Filter Preferences

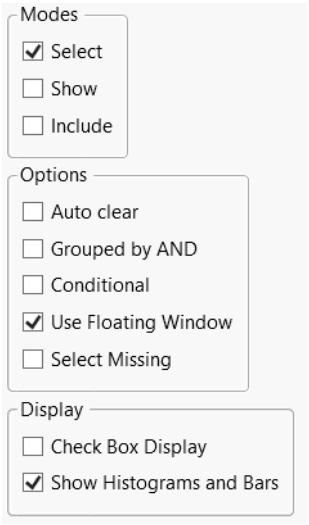


Table 13.7 Preferences on the Data Filter Page

Preference	Description
Modes	Specifies the filtering mode. **Select** Shows the selected rows in the data table in a highlighted state. This option is selected by default. **Show** Shows the unselected rows with the Hide icon. This option is deselected by default. **Include** Shows the unselected rows with the Exclude icon. This option is deselected by default.

Table 13.7 Preferences on the Data Filter Page *(Continued)*

Preference	Description
Options	Specifies options for the data filter. **Auto clear** If you have more than one nominal or ordinal column selected in the Data Filter, this option clears any other selections before making a new selection. This option is deselected by default. **Grouped by AND** Enables you to control the AND and OR behavior of multiple groups of column filters. This option is deselected by default. **Conditional** Limits the categories displayed for the selected filter column. This option is deselected by default. **Use Floating Window** Keeps the Data Filter window on top of its associated data table. This option is selected by default. **Select Missing** Highlights missing values in the data table. This option is deselected by default.
Display	Specifies display options for categorical columns. **Check Box Display** Shows a check box next to each level. This option is deselected by default. **Show Histograms and Bars** Shows the histogram and bars in the data filter. This option is selected by default.

Preferences by JMP Platform

Each analysis report has a variety of plot and table options that are shown by default. However, there might be additional options that you want to see each time you run a particular analysis. For example, a bivariate analysis shows a scatterplot by default, but you might also always want to see a linear fit each time. By selecting the Platforms option in the left panel of the Preferences window, you can set the default options for analyses. (Analyses are run by using platforms, such as the Bivariate platform. Thus, the name of this category is Platforms.)

Highlight an analysis name in the Platforms list. Its available options appear in the Options box with the defaults selected.

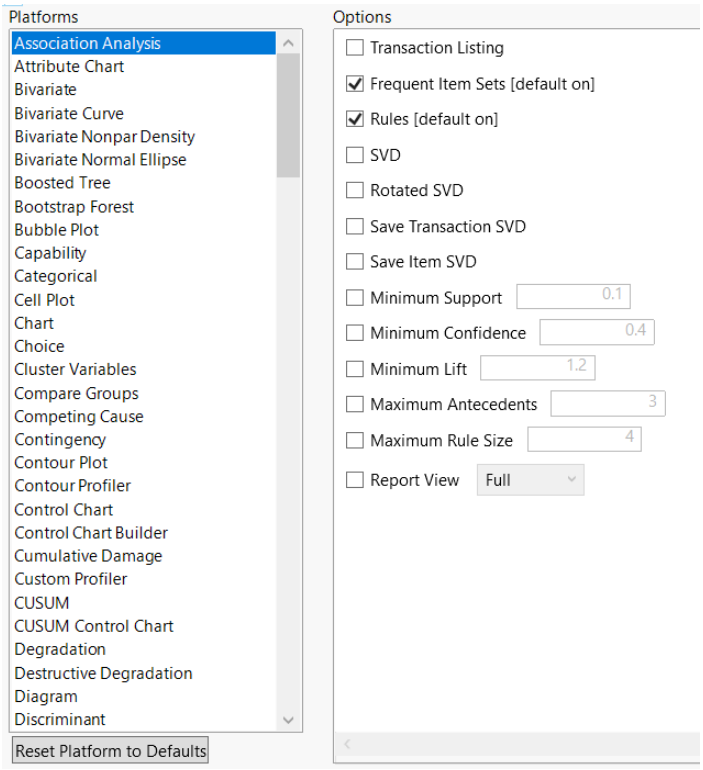
Some platform preferences include a Set check box to control whether an option appears in a report.

- To prevent an option from appearing in the report, next to an option, select **Set** but do not select the option.
- To ensure an option appears in the report, select **Set** and select the option.

Click **Reset Platform to Defaults** to return the selected platform options to the default settings.

Click **Reset to Defaults** to return all platforms to their default settings.

Figure 13.8 Platforms Preferences



Print Preferences

The Print pages enables you to configure default print settings such as, margins, header, and footers.

Figure 13.9 Print Preferences

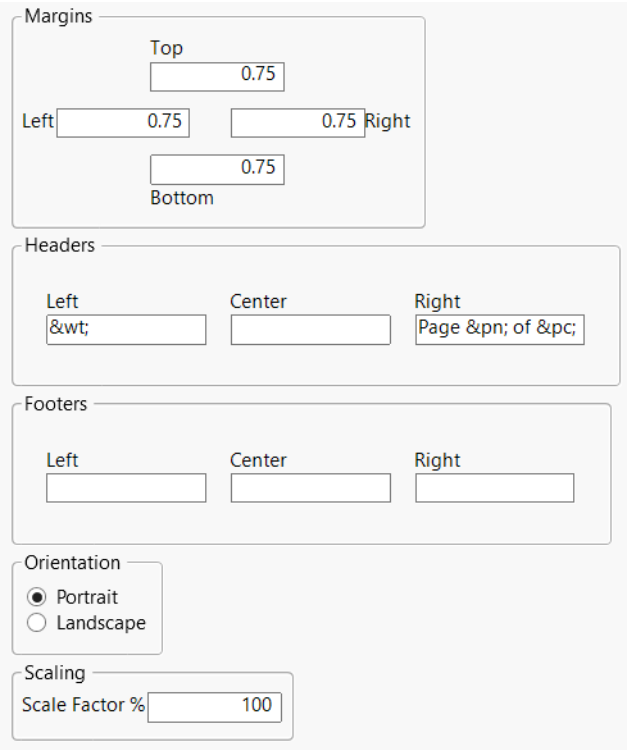


Table 13.8 Preferences on the Print Page

Preference	Description
Top (margin)	Sets the print page's top margin. The default setting is 0.75 inches.
Left (margin)	Sets the print page's left margin. The default setting is 0.75 inches.
Right (margin)	Sets the print page's right margin. The default setting is 0.75 inches.
Bottom (margin)	Sets the print page's bottom margin. The default setting is 0.75 inches.

Table 13.8 Preferences on the Print Page *(Continued)*

Preference	Description
Left (header)	Sets the page's left-side header information. The default setting is **&wt**; (that is, the title).
Center (header)	Sets the page's center header information. The default setting is blank.
Right (header)	Sets the page's right-side header information. The default setting is **Page &pn; of &pc;** (that is, Page # of count).
Left (footer)	Sets the page's left-side footer information. The default setting is blank.
Center (footer)	Sets the page's center footer information. The default setting is blank.
Right (footer)	Sets the page's right-side footer information. The default setting is blank.
Orientation	Sets the page's print orientation as either Portrait or Landscape. The default setting is Portrait.
Scale Factor	Sets the page's print scaling factor (in percent). The default setting is 100%.

Note: The code for printing the current date in the header or footer is **&d**.

Preferences for Importing and Exporting Text Files

Text Data File preferences customize importing and exporting text files.

Figure 13.10 Text Data Files Preferences

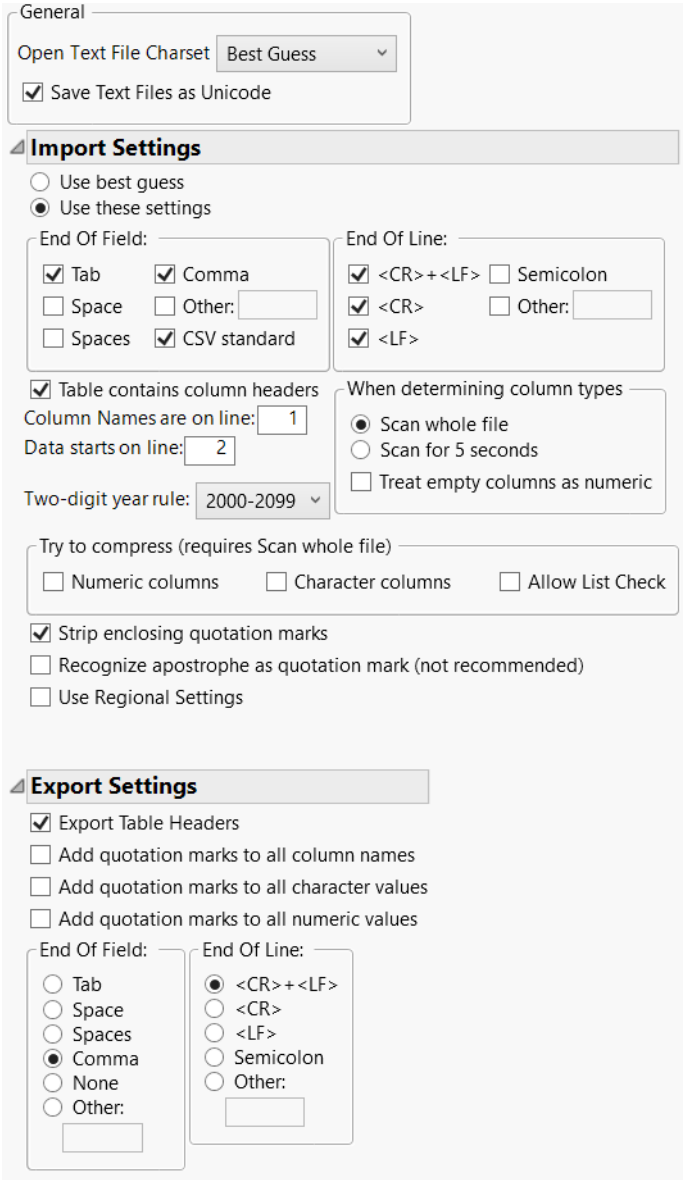


Table 13.9 Preferences for Import Settings for Text Files

Preference	Description
Open Text File Charset	Select one of the options from the menu to determine what character encoding JMP uses to open files. The default setting is Best Guess. Note that Windows-1252 is considered ANSII on some systems, and UTF-8-BOM is not supported.
Save Text Files as Unicode	JMP uses the Unicode character set, which supports special characters such as é and ½. It saves files without special Unicode characters as plain text automatically. This option is selected by default. Deselect this check box to save all your files as plain text.
Import Settings	Select the strategy JMP uses to open text files. The default selection is **Use these settings**. In that case, you need to ensure that the settings reflect your text files. If you select **Use best guess**, JMP collects statistics in the text file on tabs, commas, blanks, and a few other characters and uses a rule-based system to decide what the file format might be. The rules try to make reasonable field widths and a reasonable number of fields per line. If your data format is too different from what the rules are designed to guess, JMP guesses incorrectly. In that case, either use the wizard or explicitly describe your data in these preference settings.
End Of Field	Select one or more characters to use as the delimiter that signifies the end of a field when importing text data. Tab, comma, and CSV standard are selected by default. Select the **Other** option and enter a character to specify a delimiter that is not listed.

Table 13.9 Preferences for Import Settings for Text Files *(Continued)*

Preference	Description
End Of Line	Select one or more characters to use as the delimiter that signifies the end of a line (row). <CR>+<LF>, <CR>, and <LF> are selected by default. Select the **Other** option and enter a character to specify a delimiter that is not listed. Note that if double-quotes are encountered when importing text data, the delimiter rules change to look for an end double-quote. Other text delimiters, including spaces, that are embedded within the quotes are ignored and treated as part of the text string.
Table contains column headers	Select this option if your text file contains columns names. If you select this option, enter the line number where the column names are located in the field next to **Column Names are on line**. This option is selected by default.
Column Names are on line	If you select the **Table contains column headers** option, enter the line number where the column names are located in this field. Line one is the default setting.
Data starts on line	Enter the line number where the data starts in your text file. Line two is the default setting.
When determining column types	Set how long JMP scans a text file to determine data types for the columns. **Scan whole file** is selected by default. Note that the **Scan whole file** option can cause importing a text file to be slow for large files. Consider selecting **Scan for 5 seconds** instead. When your text file contains columns of missing data, select **Treat empty columns as numeric** to import the columns as numeric rather than character. A period, Unicode dot, *NaN*, or a blank string are possible missing value indicators. This option is deselected by default.
Two-digit year rule	Select the rule that you want to use to import dates that have two-digit years instead of four. 2000-2099 is the default setting. For more information about these rules, see "Two-digit year rule".

Table 13.9 Preferences for Import Settings for Text Files *(Continued)*

Preference	Description
Try to compress	Select the options used for compressing text files. The following options are available, and all are deselected by default: • Numeric columns • Character columns • Allow List Check **Note:** This feature requires a scan of the entire file.
Strip enclosing quotation marks	Select this option to remove quotation marks that enclose data in the text file. This option is selected by default.
Recognize apostrophe as quotation mark	Select this option to treat apostrophes as quotation marks and omit them. This option is deselected by default. **Note:** This option is not recommended unless your data comes from a nonstandard source that places apostrophes around data fields rather than quotation marks.
Use Regional Settings	Select this option to use the operating system's regional settings when importing a text file. • If the option is deselected (the default setting), files that use a period for a decimal point and a comma for the value separator import correctly. • If the file uses a comma for a decimal point and some other value separator (and the regional settings use a comma for a decimal point), selecting this option imports the text correctly. You must specify the value separator in the Text Data Files import preferences.

Table 13.10 Preferences for Export Settings for Text Files

Preference	Description
Export Table Headers	Select this option to include column names when you save data tables as text files. This option is selected by default.
Add quotation marks to all column names	Select this option to insert quotation marks around column names. Used to export data to a program that has more stringent requirements than CSV. This option is deselected by default.

Table 13.10 Preferences for Export Settings for Text Files *(Continued)*

Preference	Description
Add quotation marks to all character values	Select this option to insert quotation marks around character values. Used to export data to a program that has more stringent requirements than CSV. This option is deselected by default.
Add quotation marks to all numeric values	Select this option to insert quotation marks around numeric values. Used to export data to a program that has more stringent requirements than CSV. This option is deselected by default.
End Of Field	Select one or more characters to use as the delimiter signifying the end of a field when exporting text data. The comma is the default setting. Select the **Other** option and enter a character to specify a delimiter that is not listed.
End Of Line	Select one or more characters to use as the delimiter that signifies the end of a line (row). <CR>+<LF> is the default setting. Select the **Other** option and enter a character to specify a delimiter that is not listed.

Preferences for Third-Party Data

These preferences specify settings for third-party data, such as Microsoft Excel and SPSS files.

Figure 13.11 Third Party Data Preferences

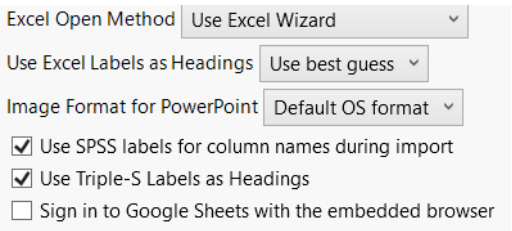


Table 13.11 Preferences on the Third Party Data Page

Preference	Description
Excel Open Method	Select one of the options to determine how JMP imports Microsoft Excel worksheets. **Excel Wizard** Lets you preview a Microsoft Excel worksheet and modify the settings before importing the data. This is the default setting on Windows. On macOS, all .xlsx files open in the Excel Wizard. **Open All Sheets** Opens all worksheets in the workbook. On macOS, this preference is only for .xls files. **Select Individual Worksheets** Lets you select the worksheets that you want to open from a workbook. On macOS, this preference is only for .xls files. The option is also provided on macOS when you open an Excel (.xls) file. On Windows, you can also select **File > Open**, select the workbook, click the **Open** button arrow, and select **Open Selected Worksheets** or **Use Excel Wizard**.
Use Excel Labels as Headings	Select this option to allow JMP to use Excel label names as column headings when importing Excel files. Available settings include: • Use best guess (default) • Always • Never
Image Format for PowerPoint	Select the default file format for images exported to Microsoft PowerPoint. Windows does not support the native PDF graphics produced on macOS. macOS does not support the native EMF graphics produced on Windows. For cross-platform compatibility, specify the PNG or JPEG graphic format. Additional file formats are available through scripting. See the *JSL Syntax Reference*.
Use SPSS Labels for column names during import	Select this option to allow JMP to use SPSS label names as column headings when importing SPSS files.
Use Triple-S Labels as Headings	Select this option to allow JMP to use Triple-S label names as column headings.

Table 13.11 Preferences on the Third Party Data Page *(Continued)*

Preference	Description
Sign in to Google Sheets with the embedded browser	Select this option to use the embedded JMP browser for signing in to Google Sheets instead of the default system browser.

Windows Preferences

The Windows Specific preferences customize settings for Windows computers, including auto-hiding menus and toolbars and selecting the default graphics format.

Figure 13.12 Windows Specific Preferences

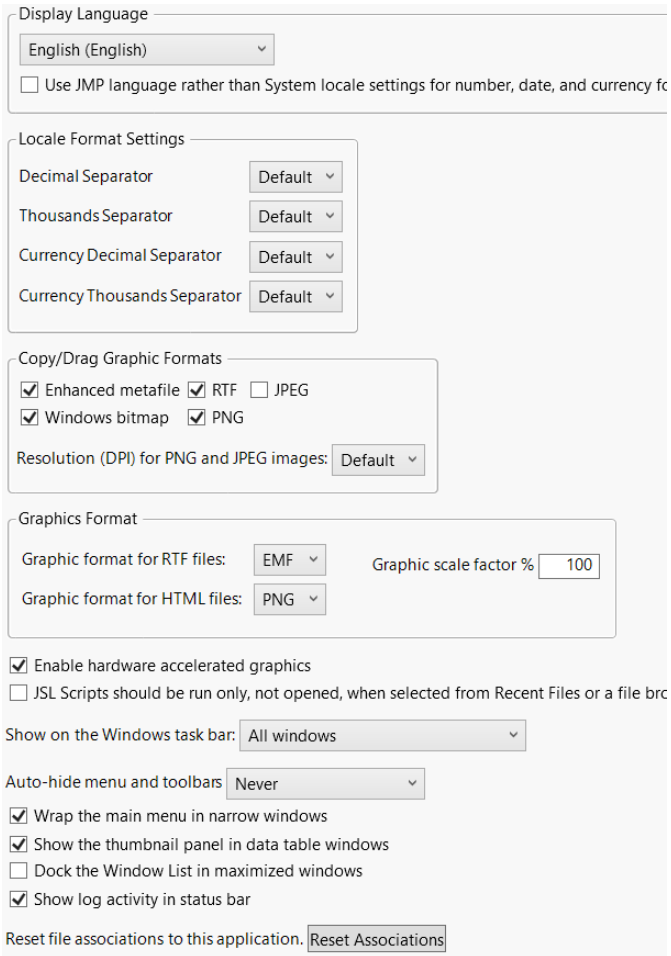


Table 13.12 Preferences on the Windows Specific Page

Preference	Description
Display Language	Select the language in which you want to run JMP. The locale settings for your operating system normally determine settings for number, date, and currency formats. Select the option below the language menu to use that language to determine these formats instead. Text that appears in windows provided by the operating system (for example, **File > Open**), do not reflect changes in the language setting. **Notes:** • After changing the display language, close and restart JMP to have the language settings fully take effect. • The languages that you selected in the JMP installation program determine which languages are available. To make a language available, rerun the JMP installation program, select Modify, and then select the language.
Use JMP language rather than System locale settings for number, date, and currency formats	Uses the JMP display language to specify number, date, and currency formats.
Locale Format Settings	Specify the decimal and thousands separators used for numbers and currency values. The Default setting uses the system locale and the settings in the Display Language preferences. If a non-default setting is specified, that setting overrides the Display Language preferences and the system locale.
Copy/Drag Graphic Formats	Select one or more graphic formats to use when copying and pasting (or dragging and dropping) graphics from JMP into other applications. Enhanced Metafile, Windows bitmap, RTF, and PNG are selected by default. The resolution for PNG and JPEG images is Default.

Table 13.12 Preferences on the Windows Specific Page *(Continued)*

Preference	Description

Resolution (DPI) for PNG and JPEG Images	Specify the DPI to be used when you copy a graphic format. Choose from the Default (96) or 300. 300 is better for images that must be stretched, embedded in trade publications, or printed. However, this setting uses more memory and is slower to generate for large images.
	
	The following script lets you set the default DPI. Change the number to a different DPI if you want and then run the script.
	
	`Pref( Save Image DPI( 600 ) );`
	
	To verify the default DPI after you run the preceding script, run the following script:
	
	`Get Preferences( Save Image DPI );`
	
	For more information about specifying a higher DPI, see "Setting the Graphic DPI for Exported Graphics".

Graphics Formats	Select the format to use for graphics when you save a JMP report as an RTF file (EMF, PNG, or JPEG) or an HTML file (PNG, JPEG, GIF, and SVG). In the Graphic Scale Factor % box, enter the percentage at which you want graphics to appear in other applications. This feature might not work with all versions of your chosen application. 100% is the default setting.
Enable hardware accelerated graphics	Takes better advantage of the computer's graphics processing unit (GPU) for better video performance. Consider selecting this option if your computer is sluggish when drawing JMP windows. This is an experimental feature.
JSL Scripts should be run only, not opened, when selected from Recent Files or a file browser	Select this option to force all scripts to run when opened. If this is selected, the script window for the script is not opened. This option is deselected by default.
Show on the Windows task bar	Select which JMP windows you would like displayed on the Windows task bar. The default selection is All Windows. You can also choose to show the main JMP window and data table windows.

Table 13.12 Preferences on the Windows Specific Page *(Continued)*

Preference	Description
Auto-hide menu and toolbars	Select the rule that determines when menus and toolbars are hidden. Menus and toolbars are always hidden in reports and layout windows. For other windows, the menus and toolbars are shown based on the height of the window. The default setting is Never.
Wrap the main menu in narrow windows	Select this option to wrap the menu to additional lines when the window is narrower is than the menu. This option is selected by default.
Show the thumbnail panel in data table windows	Select this option to show the thumbnail panel of reports at the bottom of a data table. This option is selected by default.
Dock the Window List in maximized windows	Select this option to automatically dock the Window List if you maximize your JMP windows. This option is deselected by default.
Show log activity in status bar	Select this option to show an icon in the status bar that appears when activity has been detected in the log window. You can then click the icon to bring the log window to the front. This option is selected by default.
Reset file associations to this application	Click this button to associate all JMP file types with the JMP version that you are running.

Mac OS Preferences

Mac OS Settings preferences customize settings for macOS machines, including the graphics formats and file associations.

Figure 13.13 Mac OS Settings Preferences

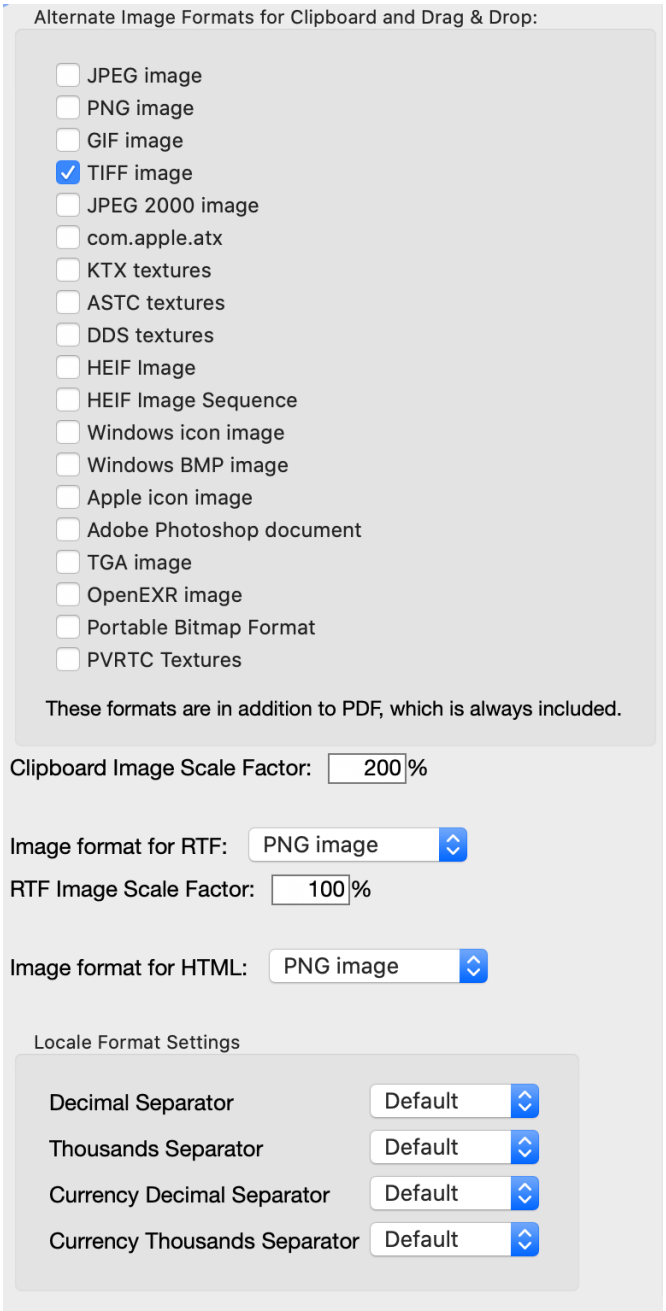


Table 13.13 Preferences on the Mac OS Settings Page

Preference	Description
Alternate Image Formats for Clipboard and Drag & Drop	Select one or more graphic formats to use when copying and pasting (or dragging and dropping) graphics from JMP into other applications. The default setting is TIFF.
Clipboard Image Scale Factor	Enter the percentage at which you want graphics to appear in other applications. The default setting is 200%.
Image format for RTF	Select the format to use for graphics when you save a JMP report as an RTF file. The default setting is PNG.
RTF Image Scale Factor	Enter the percentage at which you want graphics to appear in RTF documents. The default setting is 100%.
Image format for HTML	Select the format to use for graphics when you save a JMP report as an HTML file. The default setting is PNG.
Locale Format Settings	Specify the decimal and thousands separators used for numbers and currency values. The Default setting uses the system locale. If a non-default setting is specified, that setting overrides the system locale.

Fonts Preferences

Font preferences customize the appearance of reports, data tables, and scripts, including fonts, text size, and font style.

Figure 13.14 Fonts Preferences

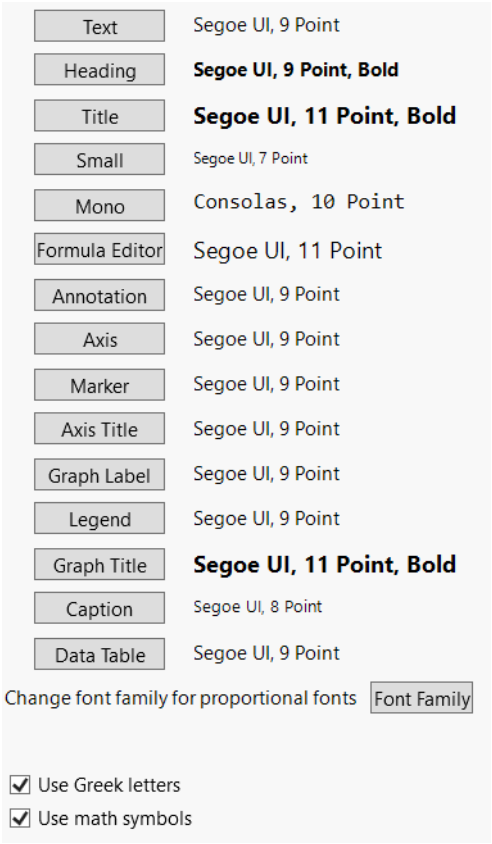


Table 13.14 Preferences on the Fonts Page

Preference	Description
Text	Sets the font for the text portion of a JMP analysis report. The default setting is Segoe UI, 9 Point.
Heading	Sets the font for the heading of columns in an analysis report and a data table. The default setting is Segoe UI, 9 Point Bold.
Title	Sets the font for the title shown in all title bars. The default setting is Segoe UI, 11 Point, Bold.
Small	Sets the font for small text, which is used in the upper left corner of the data grid to show the number of columns and rows. The default setting is Segoe UI, 7 Point.

Table 13.14 Preferences on the Fonts Page *(Continued)*

Preference	Description
Mono	Sets the font used in the JMP Scripting Language (JSL) editor and editor boxes for script commands. The default setting is Consolas, 10 Point.
Formula Editor	Sets the font for the expressions entered into the Formula Editor. The default setting is Segoe UI, 11 Point.
Annotation	Sets the default font used in annotations. The default setting is Segoe UI, 9 Point.
Axis	Sets the font used for the axis tick labels. The default setting is Segoe UI, 9 Point.
Marker	Sets the font used for alphanumeric markers. The default setting is Segoe UI, 9 Point.
Axis Title	Sets the font for axis titles. The default setting is Segoe UI, 9 Point.
Graphic Label	Sets the font for data labels in graphs such as marker labels or bar labels. The default setting is Segoe UI, 9 Point.
Legend	Sets the font for legend titles, values, and other legend text. The default setting is Segoe UI, 9 Point.
Graph Title	Sets the font for the main graph title but not the secondary title or footnote. The default setting is Segoe UI, 11 Point, Bold.
Caption	Sets the font for the caption element and also any caption-like text (for example, line of fit statistics and the box plot 5-number summary). The default setting is Segoe UI, 8 Point.
Data Table	Sets the font for displaying data in the data table. The default setting is Segoe UI, 9 Point.
Font Family	Select a proportional font. The font that you select is immediately applied to all settings except for Mono, which is used for scripts.
Use Greek letters	Select this option to use Greek letters instead of spelling out Greek letters (for example, π instead of pi.) This option is selected by default.
Use math symbols	Select this option to use math symbols instead of simple text representations of math symbols (for example, ± instead of +/-) This option is selected by default.

Communications Preferences

(Windows only) Communications preferences customize settings for reading data from an external source. These settings need to be specified only if you are using an instrument to do so.

Figure 13.15 Communications Preferences

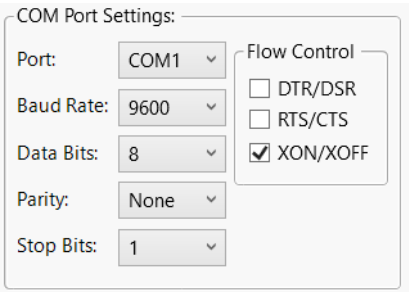


Table 13.15 Preferences on the Communications Page

Preference	Description
Port	Select the port that your data source uses.
Baud Rate	Set the baud rate for your data source.
Data Bits	Set the number of data bits (7 or 8).
Parity	Set the parity bit. None is the default setting.
Stop Bits	Set the stop bits (1 or 2).
Flow Control	Set the flow control. XON/XOFF is the default setting.

Preferences for Internet Options

Internet Options specify settings for opening files from the Internet.

Figure 13.16 Internet Options

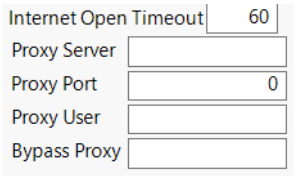


Table 13.16 Preferences on the Internet Options Page

Preference	Description
Internet Open Timeout	When you use Internet Open to open a web page, JMP waits the specified number of seconds before stopping the import due to an error. The default setting is 60 seconds.
Proxy Server	Specifies the proxy host. `http` (or `https`) needs to be included with the specification. `http` is assumed if no protocol is used.
Proxy Port	Specifies the port used to connect to the proxy host. If none is specified, the port is 1080 (for `http`) and 443 for all others (`https`). The default setting is 0.
Proxy User	Specifies an optional user for the proxy. Not all proxy settings require a user.
Bypass Proxy	Specifies an optional comma-separated list of hosts to bypass (that is, not going through a proxy). A user might want to filter out hosts that are behind the corporate firewall (which don't require a proxy).

Preferences for File Locations

(Windows only) File Locations preferences set the default locations of JMP system files.

Figure 13.17 File Locations Preferences

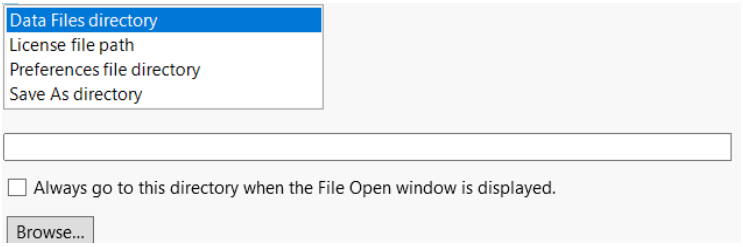


Table 13.17 Preferences on the File Locations Page

Preference	Description
Data Files directory	Change the folder in which JMP looks for data files (for example, data tables).
License file path	Change the folder in which JMP looks for your JMP License file.
Preferences file directory	Change the folder in which JMP looks for preference information.
Save As directory	Change the folder in which JMP saves data files when you select File > Save As.
Always go to this directory when the File Open window is displayed	Select to have JMP always use the specified path when you select File > Open.

Preferences for the JSL Script Editor

Script Editor preferences customize the appearance of the JSL Script Editor and script editor boxes, such as tab width, syntax coloring, and tooltips.

Figure 13.18 Script Editor Preferences

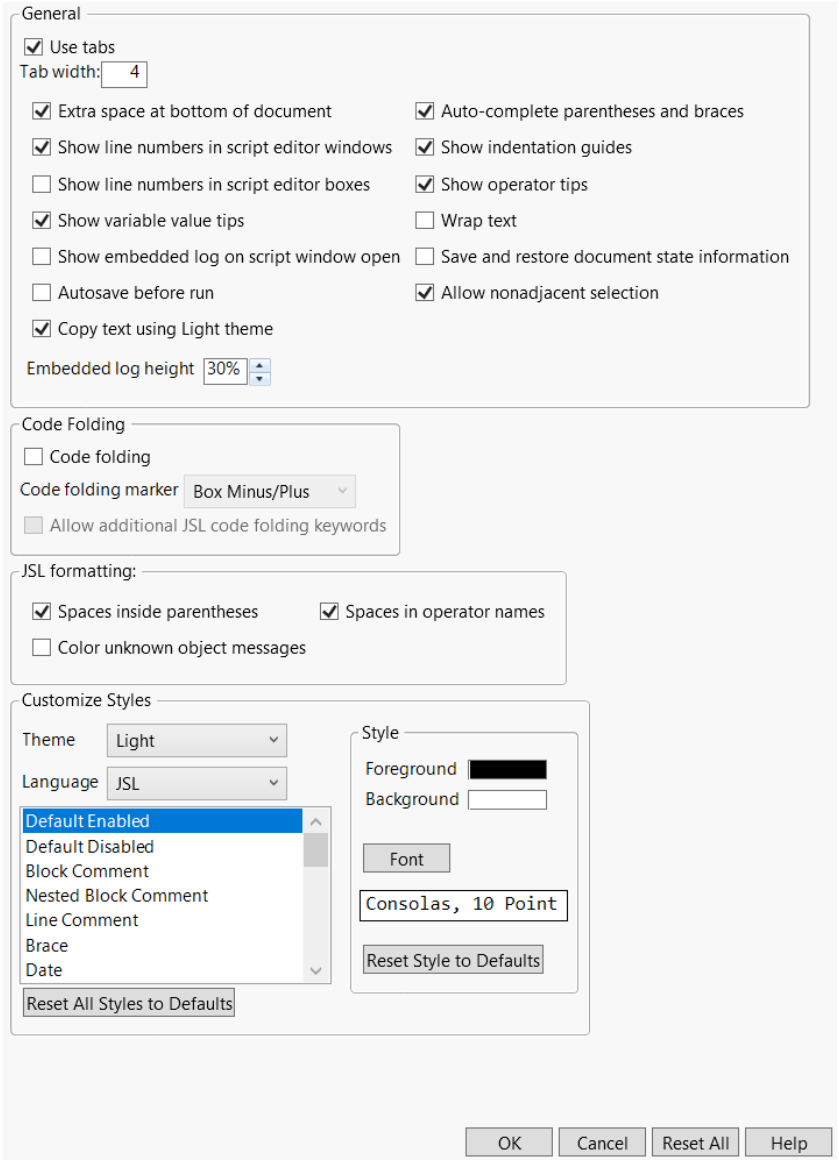


Table 13.18 Preferences on the Script Editor Page

Preference	Description
Use tabs	Select this option to enable tabs in your scripts. This option is selected by default. Clear this option to replace any tab that you type with spaces.
Tab width	Enter how many spaces a tab should indent. If you have disabled tabs, any tab you type is replaced with the number of spaces specified. The default setting is 4.
Extra space at bottom of document	Select this option to enable scrolling up from the last blank lines of a script. This option is selected by default on Windows and deselected on macOS.
Auto-complete parentheses and braces	Select this option to enable the script editor to automatically add closing parentheses, square brackets, and curly braces when you type an opening one. This option is selected by default.
Show line numbers in script editor windows	Select this option to show the line numbers on the left side of the script editor. This option is selected by default.
Show indentation guides	Select this option to see faint vertical lines that mark indention. This option is selected by default.
Show line numbers in script editor boxes	Select this option to show the line numbers on the left side of script editor boxes (such as in the Scripting Index). This option is off by default.
Show operator tips	Select this option to see tooltips for JSL operators. This option is selected by default.
Show variable value tips	Select this option to see tooltips for variable values. This option is selected by default.
Wrap text	Select this option to always wrap text in the script editor. This option is off by default.
Show embedded log on script window open	Select this option to have an embedded log window appear in the scripting window when editing or running scripts. This option is off by default.
Save and restore document state information	Saves the state of collapsed and expanded code, and restores that state when the script is reopened.

Table 13.18 Preferences on the Script Editor Page *(Continued)*

Preference	Description
Autosave before run	Automatically saves the script before you run it. This option is off by default.
Allow nonadjacent selection	Enables you to select text that is not adjacent. Press Ctrl on Windows or Command on macOS and drag your cursor over the text. Continue this action for any other text that you want to select. This option is selected by default.
Copy text using Light theme	Determines whether the background color of the code is pasted to another document. This option is selected by default. This means that, when the Light theme is selected in the Script Editor preferences, the code will be pasted with a light background into another document.
Embedded log height	Specifies the height of the embedded log in a script window. The default setting is 30%.
Code folding	Select this option to use code folding markers in the script editor, which mark the opening and closing of user defined functions and `Expr()` blocks. You can expand and collapse these marked blocks of code. This option is off by default. You can also choose the appearance of the marker using the JSL code folding marker menu. See the *Scripting Guide*.
Allow additional JSL code folding keywords	Select this option to enable using additional keywords for folding markers in the script editor. See the *Scripting Guide*.
Spaces inside parentheses	Select this option to cause the script editor to add spaces between parentheses, brackets, and braces and their contents for automatically formatted scripts. This is on by default.
Spaces in operator names	Select this option to cause the script editor to add spaces between words within operator names. For example, turning on this option results in `New Window` instead of `NewWindow`. This option is selected by default.
Color unknown object messages	Select this option to always show unknown object messages in color. Unknown object messages will appear in the color specified by “Message unknown color”. This option is off by default.

Table 13.18 Preferences on the Script Editor Page *(Continued)*

Preference	Description
Customize Styles	Select colors and fonts for code elements shown in JSL scripts, SAS scripts, the Text Explorer customization window, text files, and so. Reset All Styles to Defaults restores the original colors for all items in the list. Reset Style to Defaults restores the original colors and font for the selected item on the right. Note that resetting the Fonts preferences does not affect the fonts selected on this page.

Preferences for SAS Integration

SAS Integration preferences customize the default settings for working with SAS servers. For more information about using the SAS Integration capabilities, see "Import Data from SAS".

Note: The SAS Environment options appear after you select SAS 9.3 or 9.4 from the SAS Server Version list.

Figure 13.19 SAS Integration Preferences

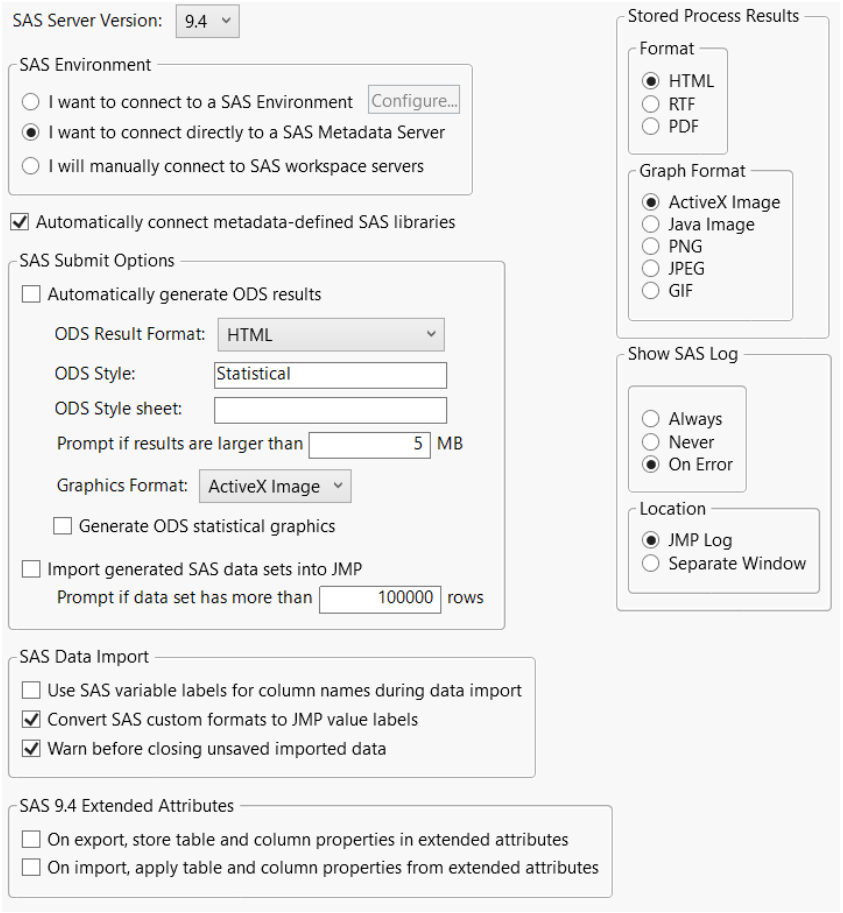


Table 13.19 Preferences on the SAS Integration Page

Preference	Description
SAS Server Version	Select the default version for the SAS Server. This can also be changed in the Server Connections window. (See "Connect to a SAS Metadata Server from JMP".) This option does not apply when connecting to local SAS on Windows. The default setting is 9.4.
I want to connect to a SAS Environment	Select this option to always connect to a SAS Environment and click **Configure** to configure the URL of the location. This option is deselected by default.

Table 13.19 Preferences on the SAS Integration Page *(Continued)*

Preference	Description
I want to connect to a SAS Metadata Server	Select this option to always connect to a SAS Metadata Server. This option is selected by default. Clear this option if you do not have a SAS Metadata Server available, and you connect directly to SAS Workspace Servers instead.
I will manually connect to SAS workspace servers	Select this option to manually connect to a SAS workspace server. This option is deselected by default.
Automatically connect metadata-defined SAS libraries	Select this option to connect to metadata-defined SAS libraries automatically across all JMP sessions when you connect to a SAS Workspace Server. his option is selected by default. When the Workspace Server contains a large number of metadata-defined SAS libraries, deselect this option to speed up your connection to the server.
Automatically generate ODS results	Select this option to generate output delivery system (ODS) results. This option is deselected by default.
ODS Result Format	Select the format for ODS reports. The default setting is HTML.
ODS Style	Enter the style name for ODS reports. The default setting is Statistical.
ODS Style sheet	Enter the style sheet name for ODS reports.
Prompt if results are larger than __ MB	Enter the number of MB that triggers a prompt for you to continue or cancel the operation. Five is the default setting.
Graphics Format	Select the format for graphics for ODS reports. ActiveX Image is the default setting.
Generate ODS statistical graphics	Select this option to include statistical graphics in the ODS reports.
Import generated SAS data sets into JMP	Select this option to import any generated data sets into JMP automatically.
Prompt if data set has more than __ rows	Enter the number of rows that triggers a prompt for you to continue or cancel the operation. The default setting is 100,000.

Table 13.19 Preferences on the SAS Integration Page *(Continued)*

Preference	Description
Use SAS variable labels for column names during data import	Select this option to use the column labels in the SAS data set as the JMP data table column names when importing a SAS data set into a JMP data table. This option is deselected by default.
Convert SAS custom formats to JMP value labels	Select this option to use the information in the SAS column formats to set JMP value labels when importing a SAS data set into a JMP data table. This option is selected by default.
Warn before closing unsaved imported data	Select this option to trigger a prompt to save any SAS data sets that you imported into JMP and have not saved before closing. This option is selected by default.
On export, store table and column properties in extended attributes	Select to allow extended attribute support during SAS 9.4 data export. This option is deselected by default.
On import, apply table and column properties from extended attributes	Select to allow extended attribute support during SAS 9.4 data import. This option is deselected by default.
Stored Process Results: Format	Select a report format: HTML (default), RTF, or PDF.
Stored Process Results: Graph Format	Select a graph format: ActiveX Image (default, available only on a SAS server that runs on Windows), Java Image, PNG, JPEG, or GIF.
Show SAS Log	Select Always, Never, or On Error (default) to set when the SAS log is displayed
Location	Select the location for SAS log information: within the JMP log window, or in a separate SAS log window. The default setting is the JMP log window.

Preferences for Updating JMP

The JMP Update preferences show you what version of JMP you currently have, the last time it was updated, and whether and how often JMP checks automatically for updates.

Figure 13.20 JMP Updates Preferences

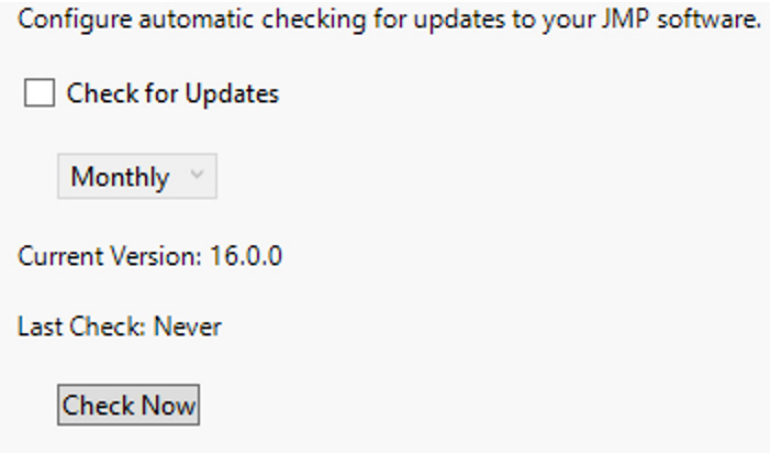


Table 13.20 Preferences on the JMP Updates Page

Preference	Description
Check for Updates	Select this option to have JMP automatically check for updates. If this option is selected, specify how often the check is performed: Daily, Weekly, or Monthly.
Current Version	What version of JMP you are currently running.
Last Check	The last time you checked for an update.
Check Now	Click this button to perform an update check immediately.

Preferences for the JSL Debugger

The JSL Debugger preferences let you show or hide line numbers, create breaks in scripts, and show warnings.

Figure 13.21 JSL Debugger Preferences

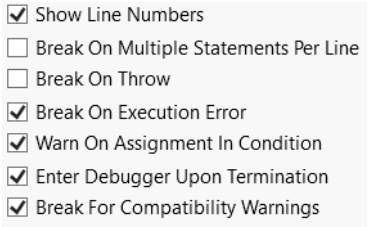


Table 13.21 Preferences on the JSL Debugger Page

Preference	Description
Show Line Numbers	Shows or hides the line numbers for the script in the debugger. This option is selected by default.
Break on Multiple Statements Per Line	Stops executing the script between each expression in a single line. This option is deselected by default.
Break On Throw	Breaks when an exception error occurs in a `Try()` function. For example, the exception error in the `Try()` function is thrown by a `Throw()` function. The Debugger breaks on the`Throw()` instead of continuing through the rest of the codes. This lets you identify where the problem occurred in the script and then return to debugging. This option is deselected by default.
Break On Execution Error	Stops executing the script when the error occurs rather than closing the Debugger. The option is selected by default
Warn On Assignment In Condition	Select to show a warning when you enter an assignment expression for the breakpoint condition. For example, if you specify `x = 1` instead of `x == 1` in the breakpoint condition, you are prompted to verify the specification. The option is selected by default

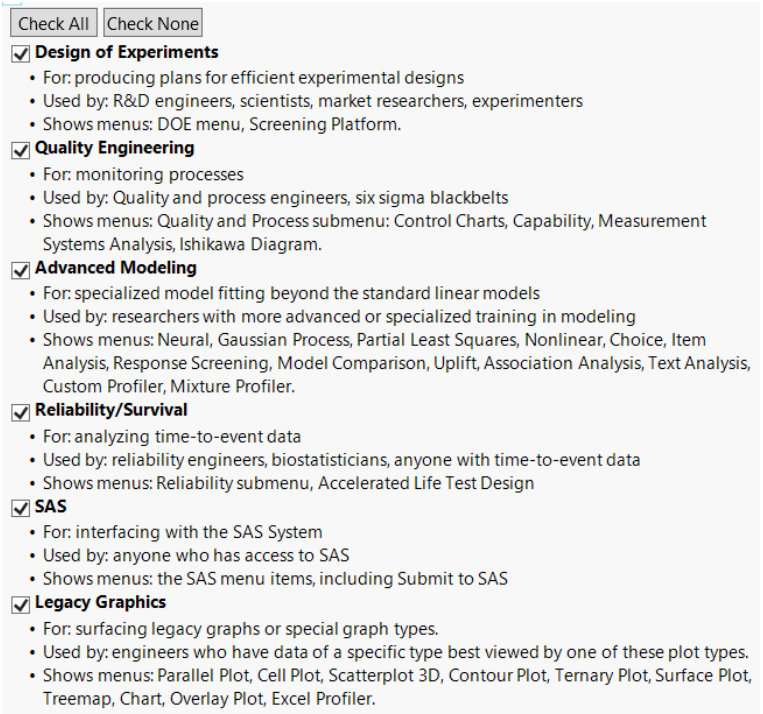
Table 13.21 Preferences on the JSL Debugger Page *(Continued)*

Preference	Description
Enter Debugger Upon Termination	Select to enter the Debugger when JMP terminates. The option is selected by default
Break for Compatibility Warnings	Stops executing the script when a JSL compatibility issue is encountered. The option is selected by default

Menu Preferences

The Menu preferences show and hide menus based on how you use JMP. This gives you fewer menu items to browse through and streamlines the JMP interface. For example, if you never design experiments, deselect **Design of Experiments**. Other menus are grouped by area of interest, such as quality engineering, advanced modeling, reliability and survival, and SAS options.

Figure 13.22 Menu Preferences



Preferences for the Query Builder

The Query Builder preferences customize SQL queries that you perform in Query Builder.

Figure 13.23 Query Builder Preferences

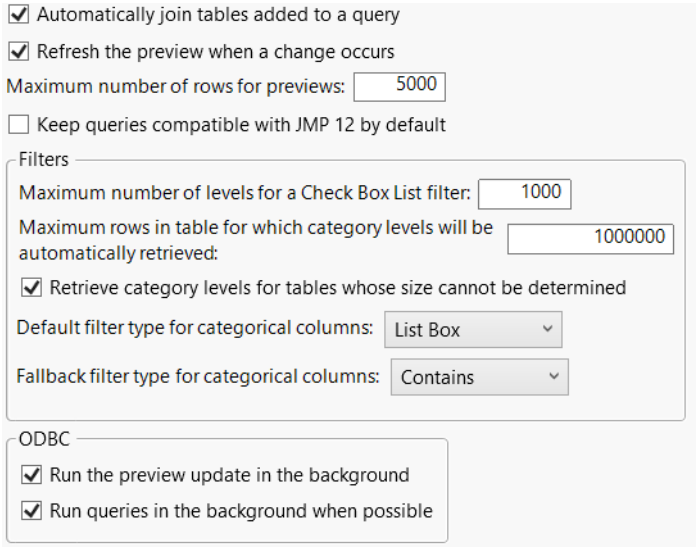


Table 13.22 Preferences on the Query Builder Page

Preference	Description
Automatically join tables added to a query	Automatically joins database tables in Query Builder based on keys and common column names. This option is selected by default. Deselect this option to prevent memory issues with large databases.
Refresh the preview when a change occurs	Updates the data view on the Query Preview tab when a change occurs. This option is selected by default.
Maximum number of rows for previews	Limits the number of rows that appear on the Query Preview tab. The default number is 5000.
Keep queries compatible with JMP 12 by default	Maintains query compatibility with JMP 12. When you open a query that is marked JMP 12 compatible in JMP 13 or later, features that create compatibility problems are hidden regardless of how the preference is set.

Table 13.22 Preferences on the Query Builder Page *(Continued)*

Preference	Description
Maximum number of levels for a Check Box List filter	Determines whether the filter for a categorical variable is added as a Check Box List filter. By default, when you add a categorical variable that has more than 1,000 levels, the filter is added as a Contains filter.
Maximum rows in table for which category levels will be automatically retrieved	Specifies the maximum numbers of rows for which unique category levels in the filtered column are retrieved. The minimum value is -1 (no limit). The maximum value is 1 billion. By default, if the categorical column has more than 1 million rows, JMP does not automatically retrieve the unique category levels.
Retrieve category levels for tables whose size cannot be determined	Determines whether category levels are retrieved for a filtered column when the number of rows cannot be determined. Selected by default. If this option is deselected, the filter is added as a Contains filter.
Default filter type for categorical columns	Specifies a List Box as the default filter type for categorical columns.
Fallback filter type for categorical columns	Specifies a Contains filter as the default filter type for categorical columns when the number of rows cannot be determined or the query is canceled.
Run the preview update in the background	Updates the data view Query Preview tab without displaying the status so that you can continue to work while the queries run. This option is selected by default.
Run queries in the background when possible	Runs database queries without displaying the status so that you that can continue to work Query Builder while the queries run. This option is selected by default.

Preferences for Recoding Columns

The Recode preferences customize the recoding of columns.

Figure 13.24 Recode Preferences

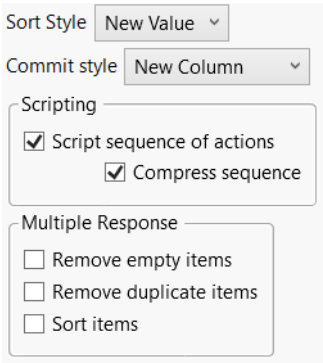


Table 13.23 Preferences on the Recode Page

Preference	Description
Sort style	Specifies how values are sorted by default. **New Value** Sorts in ascending order by the new value. **Old Value** Sorts in ascending order by the old value. **Count** Sorts in descending order with the most common values at the top.
Commit style	Defines the default behavior for recoding columns. **New Column** Puts the values in a new column is the default setting. **Formula Column** Put the values in a formula column. Select this option if you'd like to see how the values are being replaced. **In Place** Replaces the recoded column "in place".
Script sequence of actions	Saves all of your actions in the script. This option is selected by default. If the option is deselected, only the original and replacement values are included in the script.
Compress sequence	Combines expressions to simplify the saved script. Selected by default when you select **Script sequence of actions**.

Table 13.23 Preferences on the Recode Page *(Continued)*

Preference	Description
Remove Empty Items	In a Multiple Response column, excludes empty items in the column when recoding the column in the data table. This option is deselected by default.
Remove Duplicate Items	In a Multiple Response column, excludes duplicate items. For example, if the original data were a, b, b, c, and no changes to those particular items were made, the recoded column in the data table would contain a, b, c. This option is deselected by default.
Sort Items	In a Multiple Response column, sorts items alphabetically. For example, if the original data were b, a, c, and no changes to those particular items were made, the recoded column in the data table would contain a, b, c. This option is deselected by default.

Preferences for the Column Switcher

The Column Switcher preferences customize the settings for the column switcher utility.

Figure 13.25 Column Switcher Preferences

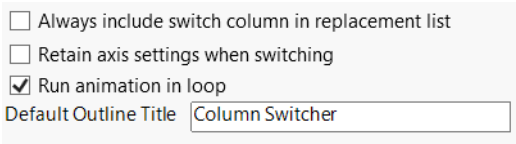


Table 13.24 Preferences on the Column Switcher Page

Preference	Description
Always include switch column in replacement list	If this option is selected, the current column is automatically included in the list of columns in the Column Switcher. If this option is deselected (the default setting), a warning appears if a specified Column Switcher does not include the current column.

Table 13.24 Preferences on the Column Switcher Page *(Continued)*

Preference	Description
Retain axis settings when switching	Specifies that the axis settings should be held constant when switching columns in the Column Switcher. This option is deselected by default.
Run animation in a loop	Specifies the default behavior for the end of the animation loop. If this option is selected (the default setting) the animation starts back at the beginning when it reaches the end of the loop. If this option is deselected, the animation stops at the end of the loop.
Default Outline Title	Specifies the title of the Column Switcher. The default setting is “Column Switcher”.

Preferences for the Log

The Log preferences specify which interactive JMP actions are recorded in the log, the log mode, and customizations of the appearance and behavior of the Enhanced Log. A preview of the log appears to the right of the preference options.

Figure 13.26 Log Preferences

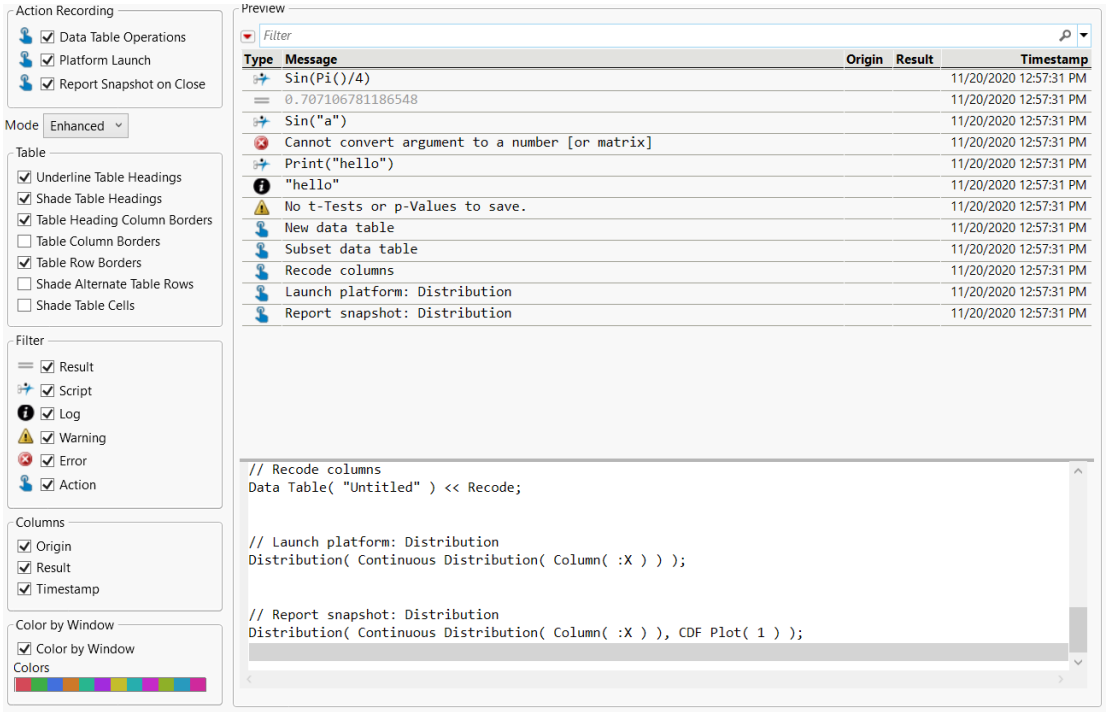


Table 13.25 Preferences on the Log Page

Preference Group	Description
Action Recording	Contains options to control what types of actions are recorded in the log. You can choose to record operations on data tables, launches of platforms, as well as a script of each report when the report is closed. The Action Recording preferences are applicable to both modes of the log. All options are selected by default.

Table 13.25 Preferences on the Log Page *(Continued)*

Preference Group	Description
Mode	Enables you to switch between the Enhanced Log and the Text Log (the default setting).
Table	Contains options to control the appearance of the contents of the Enhanced Log. Underline Table Headings, Shade Table Headings, Table Heading Column Borders, and Table Row Borders are selected by default.
Filter	Contains options to control what types of output appear in the Enhanced Log. All options are selected by default.
Columns	Contains options to control which columns of information appear in the Enhanced Log. You can show or hide the following columns of information: the origin type of the message, the result type of the message, and the time stamp of the message. All options are selected by default.
Color by Window	Specifies whether the rows in the Enhanced Log are colored by the JMP window that created the row of the log output. This option is selected by default. "JMP Default" is the default color theme.

Appendix A

Reference for JMP Functions in Formulas

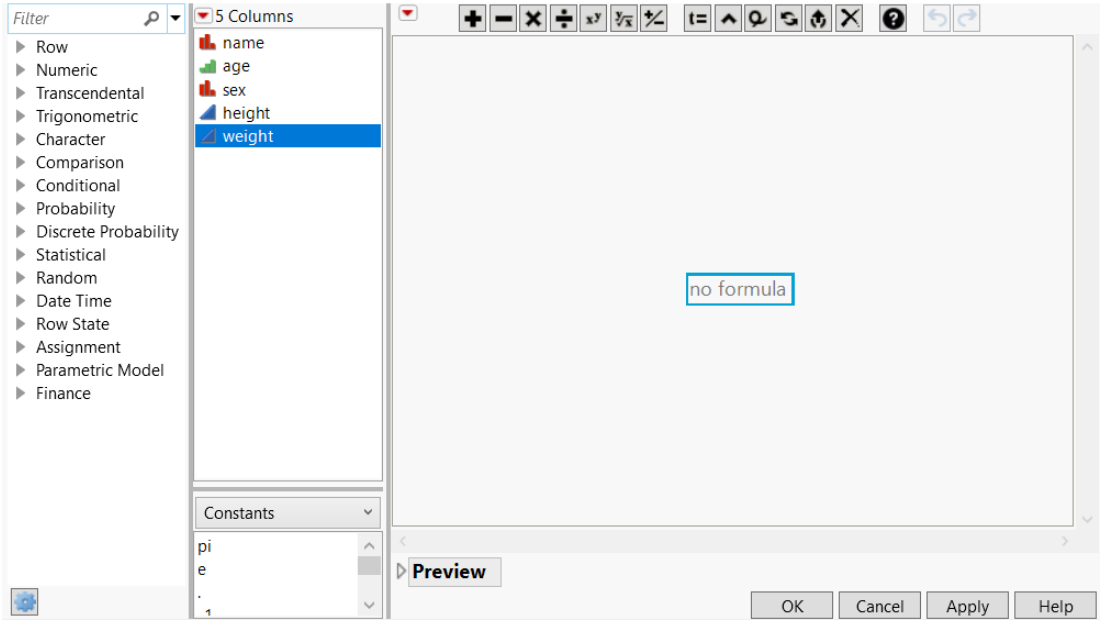
Descriptions of Functions in the Formula Editor

You can add functions to a formula. All of these functions are available in the Functions list of the Formula Editor. This chapter gives a description of functions in the Formula Editor.

More information about functions is available in the following resources:

- The Scripting Index describes all functions and their arguments, demonstrates how the functions work, and links to online Help. In JMP, select **Help > Scripting Index** to view this interactive resource.
- The *JSL Syntax Reference* also provides the arguments for JMP functions, not just those available in the Formula Editor. In JMP, select **Help > JMP Help** and find *JSL Syntax Reference* in the left column.

Figure A.1 Functions in the Formula Editor



For instructions on how to create a formula that contains a function, see "Build Formulas".

Contents

Row Functions 729
Numeric Functions 731
Transcendental Functions 732
Trigonometric Functions 736
Character Functions 737
Comparison Functions 743
Conditional Functions 744
Probability Functions 749
Discrete Probability Functions 760
Statistical Functions 762
Random Functions 766
Date Time Functions 773
Row State Functions 777
Assignment Functions 783
Parametric Model Functions 783
Finance Functions 784

Row Functions

Adding a row function to a formula lets you reference specific rows or cells within specific rows. You can also insert values based on an arithmetic sequence. See the *Scripting Guide* for more information about syntax.

Col Stored Value

Returns the column value that does not have column properties applied to it. If the row is not specified, the current row is used.

Suppose that the Missing Value Codes column property is assigned to the x1 column to treat "999" as a missing value. Another column includes a formula that calculates the mean. To use the value "999" instead of a missing value to calculate the mean, use Col Stored Value in the formula:

```
Mean( Col Stored Value( :x1 ), :x2, :x3 )
```

Count

Creates a list of values beginning with the **from** value and ending with the **to** value. The number of **steps** specifies the number of values in the list between and including the **from** and **to** values. Each value determined by the first three arguments of the count function occurs consecutively the number of times that you specify with the **times** argument. When the **to** value is reached, **Count** starts over at the **from** value.

Also, you can add the **times** argument with the insert button ⌃ on the keyboard. This argument is one by default, but repeats the count process as many times as you specify, as illustrated by the Count4 column in the data table in Figure A.2. To add any argument to the **Count** function, highlight the argument preceding the one that you want to enter. Either type a comma or use the insert button ⌃ on the Formula Editor keypad.

The columns in the data table below result from the following formulas:

- **Count (1, 9, 2)** gives Count 1
- **Count (1, 9, 3)** gives Count 2
- **Count (1, 9, 9)** gives Count 3
- **Count (1, 9, 3, 3)** gives Count 4

Figure A.2 Example of the Count Function

	count1	count2	count3	count4
1	1	1	1	1
2	9	5	2	1
3	1	9	3	1
4	9	1	4	5
5	1	5	5	5

The **Count** function is useful for generating a column of grid values. For example, the following formulas create a square grid of increment **NRow()**. **NRow()** is the **Row** function that gives the total number of rows in the data table. **NRow()** also creates axes that range from –5 to 5:

```
Count(-5, 5, Root( NRow() ) );
Count(-5, 5, Root( NRow() ), Root( NRow() ) );
```

Dif

Returns the difference between the value of the first argument in the current row and its value in the row defined by the current row less the second argument. The default **Dif** is one, which you can change to any number. Note that `Dif(X, n)` gives the same result as $X_{Row()}-X_{Row()-n}$, or as $X_{Row()}-$`Lag(X, n)`.

Dim

Returns a row vector with the dimensions of the current data table, a specified data table, or a matrix. The dimensions are the number of rows and the number of columns and are listed in that order.

Lag

Returns the value of the first argument in the row defined by the current row less the second argument. The default **Lag** is one, which you can change to any number. The value returned for any lag that identifies a row number less than one is missing. Note that `Lag(X, n)` gives the same result as the subscripted notation, $X_{Row()-n}$.

NRow

Returns the total number of rows in the active data table.

Row

Returns the current row number when an expression is evaluated for that row. You can use **Row()** in any expression, including column name subscripts. The default subscript of a column name is **Row()** unless otherwise specified.

Subscript

Enables you to use a column's value from a row other than the current row. After choosing **Subscript** from the list, enter a numeric expression into the subscript argument. Subscripts that evaluate to nonexistent row numbers produce missing values. Column names with no subscript refers to the current row. To remove a subscript, select the subscript and delete it. Then delete the missing box.

The formula $\texttt{Count}_{\texttt{Row()}} - \texttt{Count}_{\texttt{Row()}-1}$, where **Row()** is the row number as described below, uses subscripts to calculate the difference between each pair of values from the column named Count. This result is the same as that given by the **Dif()** function. When **Row()** is 1, the computation produces a missing value.

The formula below calculates a column called Fib, which contains the terms of the *Fibonacci* series (each value is the sum of the two preceding values in the calculated column).

$$\text{If}\begin{bmatrix}\text{Row()}<=2 \Rightarrow 1\\ \text{else} \quad\quad \Rightarrow Fib_{\text{Row()-1}}+Fib_{\text{Row()-2}}\end{bmatrix}$$

It shows the use of subscripts to do recursive calculations. A recursive formula includes the name of the calculated column, subscripted such that it references only previously evaluated rows (rows 1 through (*i*–1)). The calculation of the Fibonacci series shown includes a conditional expression and a comparison. See the sections "Conditional Functions" and "Comparison Functions".

Sequence

Produces an arithmetic sequence of numbers across the rows in a data table, where the start value, ending limit, and increment are specified as arguments.

Numeric Functions

You can create a formula that contains arithmetic operators that are commonly used in formulas. See the *Scripting Guide* for more information about syntax.

Abs

Returns a positive number of the same magnitude as the value of its argument. For example, |5| and |–5| both result in 5.

Ceiling

Returns the smallest integer greater than or equal to its argument. For example, **Ceiling(2.3)** results in 3, while **Ceiling(–2.3)** results in –2.

Floor

Returns the largest integer less than or equal to its argument. For example, **Floor(2.7)** results in 2, but **Floor(–0.5)** results in –1.

Integrate

Integrates an expression with respect to a scalar value, using the adaptive quadrature method from Gander and Gautschi (2000). For example, **Integrate(Exp(-x), x, 0, .)** results in 0.99999999986846.

Modulo

Returns the remainder when the second argument is divided into the first. For example, **Modulo(6, 5)** results in 1.

Round

Rounds the first argument to the number of decimal places given by the second argument. For example, **Round(3.554, 2)** rounds to 3.55 and **Round(3.555, 2)** rounds to 3.56.

Transcendental Functions

You can create a formula that supports transcendental functions, such as logarithmic functions for any base, functions for combinatorial calculations, the **Beta** function, and several gamma functions. See the *Scripting Guide* for more information about syntax.

Exp

Raises *e* to the power that you specify. Thus, **Exp(1) = e**.

ExpM1

Returns a more accurate calculation of **Exp(x) - 1** when *x* is very small.

Ln

Calculates the natural logarithm of *x*.

Log and Log10

Calculates the natural logarithm (base *e*). To change the default base, highlight the argument and type a comma or click the Insert key on the Formula Editor keypad. The base appears and is editable. The **Log** argument can be any numeric expressions. The expression **Log(e)** evaluates as 1, and **Log(32,2)** is 5. The **Log10** function calculates the logarithm of base 10 only.

Log1P

Returns a more accurate calculation of **Log(1+x)** when x is very small.

Squash

Computes the function **1 / (1 + e^x)**, where x is any numeric column, variable, or expression.

Logist

Also known as Squish or Logistic, is an efficient computation of the function **1 / (1+e^-x)**, where x is any numeric column, variable, or expression.

Root (Square Root)

Calculates the root of its argument as specified by the index. **Root** initially shows with an index of 2. To change the index, highlight the index argument and enter the value that you want.

Factorial

Returns the product of all numbers 1 through the argument that you specify. For example, **Factorial(5)** evaluates as 120.

NChooseK

Returns the number of n things taken k at a time (n select k) and is computed in the standard way using factorials, as $n! / (k!(n - k)!)$. For example, **NChooseK(5,2)** evaluates as 10.

Beta

Adds the two parameter **Beta** function and is written terms of the Gamma function as:

$$B(m, n) = \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n)}$$

Gamma

Adds the **Gamma** function, denoted $\Gamma(i)$, and is defined as:

$$\Gamma(i) = \int_0^\infty (x^{i-1})(e^{-x})dx$$

Gamma with a single argument is the same as **Gamma(x, infinity)**. The optional second argument changes the upper integer from infinity to the value that you enter. Other interesting gamma function relationships are

- for any $\alpha > 1$, $\Gamma(\alpha) = (\alpha-1) \bullet \Gamma(\alpha-1)$
- for any positive integer, n, $\Gamma(n) = (n-1)!$
- $\Gamma(0.5)$ = the square root of π

LGamma

Is the natural log of the result of the gamma function evaluation. You get the same result using the **Log** (natural log) function with the **Gamma** function. However, the **LGamma** function computes more efficiently than do the **Log** (natural log) and the **Gamma** functions together. **NChooseK** is implemented using **LGamma** functions. The result is not always an exact integer. If the result is close to an integer, it is rounded up using the **Floor** function.

Digamma

The logarithmic derivative of the **Gamma** function.

Trigamma

The derivative of the **Digamma** function, or the logarithmic second derivative of the **Gamma** function.

Arrhenius

Calculates the non-specific component of the Arrhenius relationship that is then multiplied by the activation energy in the Arrhenius equation.

$$\frac{11604.5181215503}{T + 273.15}$$

Arrhenius Inv

The inverse of the Arrhenius function:

$$\left(\frac{11604.5181215503}{y}\right) - 273.15$$

Logit

Applies the logit transformation to the argument using:

$$\text{logit}(p) = \log\left(\frac{p}{(1-p)}\right)$$

Logit Percent

Calculates the logit as a percent for the argument.

$$\text{LogitPct}(pct) = \log\left(\frac{\left(\frac{pct}{100}\right)}{1-\left(\frac{pct}{100}\right)}\right)$$

Logist Percent

Calculates the logistic as a percent for the argument.

$$\text{LogisticPct}(x) = \frac{100}{(1+e^{-x})}$$

Scheffe Cubic

Is used in fitting certain models. **Scheffe Cubic (X1, X2)** is equivalent to **X1*X2*(X1-X2)**.

SbInv

Johnson Sb inverse transformation. If argument is normal, the result is Johnson Sb.

SbTrans

Johnson Sb transformation from a doubly bound variable to a standard normal (0, 1) distribution.

SHASHInv

Returns a transformation of a standard normal variable into a sinh-arcsinh (SHASH) distributed variable.

SHASHTrans

Returns a transformation of a sinh-arcsinh (SHASH) distributed variable into a standard normal distributed variable.

SlInv

Johnson Sl inverse transformation. If argument is normal, the result is Johnson Sl.

SlTrans

Johnson Sl transformation from a doubly bound variable to a standard normal (0, 1) distribution.

SuInv

Johnson Su inverse transformation. If argument is normal, the result is Johnson Su.

SuTrans

Johnson Su transformation from a doubly bound variable to a standard normal (0, 1) distribution.

Trigonometric Functions

You can create a formula that supports trigonometric functions, such as sine and cosine. JMP's trigonometric functions expect all angle arguments in radians. See the *Scripting Guide* for more information about syntax.

Sine, Cosine, Tangent

The **Sine** and **Cosine** functions calculate the sine and cosine of their respective arguments given in radians. For example, the expression **Sine(0)** evaluates as 0, and **Cosine(0)** evaluates as 1. The tangent function calculates the tangent of an argument given in radians. The expression **Tan(Pi()/4)** evaluates as 1.

ArcSine, ArcCosine, ArcTangent

The **ArcSine** and **ArcCosine** functions return the inverse sine and inverse cosine of their respective arguments. The returned value is measured in radians. For example, both expressions **ArcSine(1)** and **ArcCosine(0)** evaluate as 1.57080. The **ArcTangent** function returns the inverse tangent of its argument. The returned value is measured in radians. The expression **ArcTangent(1)** evaluates as 0.78540 (=3.14159/4).

SinH, CosH, TanH

The **SinH** and **CosH** functions return the hyperbolic sine and hyperbolic cosine of their respective arguments. The expression **SinH(1)** evaluates as 1.175201, and **CosH(0)** evaluates as 1.0. The **TanH** function returns the hyperbolic tangent of its argument. The expression **TanH(1)** evaluates as 0.761594.

ArcSinH, ArcCosH, ArcTanH

The **ArcSinH** and **ArcCosH** functions return the inverse hyperbolic sine and inverse hyperbolic cosine of their respective arguments. The expression **ArcSinH(1)** evaluates as 0.881374, and **ArcCosH(1)** is 0. The **ArcTanH** function returns the inverse hyperbolic tangent of its argument. The expression **ArcTanH(0.5)** evaluates as 0.549306.

Character Functions

You can create a formula that accepts character arguments or returns character strings and converts the data type of a value from numeric to character, or character to numeric. When you create these formulas, note that:

- Character functions can result in either character or numeric data. If you calculate a data type different from the one specified, the data type of the computed column is automatically changed to match the result.
- Arguments that are literal character strings must be enclosed in quotation marks.

See the *Scripting Guide* for more information about syntax.

Char

Produces a character string that corresponds to the digits in its numeric argument. For example, **Char(1.123)** evaluates as 1.123. See the *Scripting Guide*.

Collapse Whitespace

Trims leading and trailing whitespace and replaces interior whitespace with single space. That is, if more than one white space character is present, the **Collapse Whitespace** command replaces the two spaces with one space.

Concat ||

Concatenates character strings to produce a new string with the function's second character argument appended to the first. For example, **"Dr." || " " || name** produces a new string consisting of the title **Dr.** followed by a space and the contents of the *name* string.

Contains

Returns the numeric position within the first argument of the first instance of the second argument, if it exists. The second argument can contain one or more characters. If the second argument does not exist, **Contains** returns a zero. For example, **Contains("Veronica Layman", "ron")** evaluates as 3. **Contains("Lillie Layman", "L")** evaluates as 1. **Contains("Lillie Layman", "Veronica")** evaluates as **0.** The third argument, *offset*, is optional. Offset is a numeric value that specifies the search starting position. **Contains("Lillie Layman", "L", 5)** evaluates as 8 as the search starts at the 5th position, skipping the first "L". If *offset* is negative, **Contains** searches backward from *offset* from the end of the string.

Munger

Computes new character strings from existing strings by inserting or deleting characters. It can also produce substrings, calculate indices, and perform other tasks depending on how you specify its arguments. The **Munger** function treats uppercase and lowercase letters as different characters.

Text is a character expression. **Munger** applies the other three arguments to this string to compute a result.

Offset is a numeric expression indicating the starting position to search in the string. If **Offset** is greater than the position of the first instance of the find argument, the first instance is disregarded.

Find/Length is a character or numeric expression. Use a character string as search criterion, or use a positive integer to return that number of consecutive characters starting from the **Offset** position. If you specify a negative integer as the **Length** value, **Munger** returns all characters from the **Offset** through to the end of the string.

Replace (optional argument) can be a string or unspecified. If it is a string and the **Find/Offset** value is numeric, **Munger** replaces the search criterion with the **Replace string** to form the result. If the Find/Offset value is numeric and no string is specified, **Munger** calculates a substring. If the **Find/Length** value is a character string, **Munger** always returns the numeric offset, disregarding the **Replace** value if it exists. To insert the **Replace** argument, click any argument in the **Munger** function and then click the insert button. Press Delete or click the Delete button ( ) on the Formula Editor keypad to remove the **Replace** argument.

Lowercase, Uppercase

The **Lowercase** function converts any uppercase character found in its argument to the equivalent lowercase character. For example, **Lowercase("VERONICA LAYMAN")** evaluates as veronica layman. The **Uppercase** function converts any lowercase character found in its argument to the equivalent uppercase character. For example, **Uppercase("Veronica Layman")** evaluates as VERONICA LAYMAN.

Length

Calculates the length of its argument. For example, **Length("Veronica")** evaluates as 8. If the argument is

- a string, length returns the number of characters;
- a list, length returns the number of items in the list;
- a blob (binary object), the number of bytes.

Num

Produces a numeric value that corresponds to its character string argument when the character string consists of numbers only. If a character string contains a non-numeric value, the result is a missing value. For example, **Num("1.123")** evaluates as 1.123.

Substr

Extracts the characters that are the portion of the first argument. Begins at the position given by the second argument, and ends based on the number of characters specified in the third argument. The first argument can be either a character column or a literal value. The starting argument and the length argument can be numbers of expressions that evaluate to numbers. For example, to show the first name only, **Substr("Veronica Layman", 10, 6)** starts at position 10 and reads through position 15, which yields **Layman**.

If *start* is negative, **Substr** searches backward from *start* from the end of the string. If *length* is negative or absent, **Substr** returns a string that begins with *start* and continues to the end of the text string.

Substr can also be used with lists.

Titlecase

Converts the string to title case, that is, an initial uppercase character and subsequent lowercase characters. For example, **Titlecase("Veronica layman")** results in **Veronica Layman**.

Trim

Produces a new character string from its argument, removing any leading and trailing whitespace. The second argument determines if whitespace is removed from the **left**, the **right**, or **both** ends of the string. If no second argument is used, whitespace is removed from both ends. For example, **Trim("john ")** evaluates as **john**. **Trim(" john ", both)** also evaluates as **john**.

Word

Extracts the n^{th} word from a character string. One or more spaces define where each word begins and ends unless the optional **delimiters** argument is specified. For example, **Word(2, "Veronica Layman")** returns the word **Layman**.

To insert the **delimiters** argument, click any argument in the **Word** function and then click the insert button on the Formula Editor keypad. Press Delete or click the Delete button on the Formula Editor keypad to remove the **delimiters** argument. If you do not specify a delimiter, space is used as the delimiter. If you define the delimiter as an empty string, each character is treated as a separate word.

Most special characters act as single delimiters. You can enter any character or set of characters to act as a word delimiter. For example, to extract the last name in the following example, use a comma and blank together as the delimiting characters and ask for the first word. **Word(1, "Layman, Veronica", ", ")** returns the word **Layman**.

Words

Extracts the words from *text* according to the delimiters listed in the optional second argument. The default delimiter is space. For example, **Words("the quick brown fox")** returns **{"the","quick","brown","fox"}**.

If you include a second argument, any and all characters in that argument are taken to be delimiters. For example, **Words("Doe, Jane P.",", .")** returns **{"Doe","Jane","P"}.**

To insert the **delimiters** argument, click any argument in the **Words** function and then click the insert button on the Formula Editor keypad. Press Delete or click the Delete button on the Formula Editor keypad to remove the **delimiters** argument. If you do not specify a delimiter, white space is used as the delimiter. If you define the delimiter as an empty string, each character is treated as a separate word.

Left, Right

Returns a substring of the left-most or right-most *n* characters of the string *text*, respectively. Both functions also work with lists.

Starts With, Ends With

Returns 1 if *whole* begins or ends with *part*, respectively. Returns 0 otherwise. Both functions also work with lists.

Item

Is different than the **Word** function because of the way it treats word delimiters. If a delimiter is found multiple times, or you enter a delimiter with multiple characters, the **Word** function treats them as a single delimiter. The **Item** function uses each delimiter to define a new word position. To compare, suppose a name is of the form lastname, firstname. The delimiter is a comma followed by a blank, such as:

Item(2, "Layman, Veronica", ", ")

Word(2, "Layman, Veronica", ", ")

The **Item** function returns a missing value because it treats the comma and blank separately and finds nothing between them. The **Word** function treats the comma and blank as a single delimiter and finds **Veronica** as the second word.

If you do not specify a delimiter, white space (blank space) is used as the delimiter. If you define the delimiter as an empty string, each character is treated as a separate item.

Hex and Hex to Number

Converts between hex and hex and numbers.

Hex returns the hexadecimal representation of its argument. If the argument is character (in quotes), then the result is a character string twice as long containing the hexadecimal codes for the character values. For example, **Hex("A")** returns the string 41.

If the argument is numeric and "integer" is specified, the Hex function returns an 8-hexadecimal-character representation of the integer returned. For example, **Hex(12, "integer")** returns the string **0000000C**.

Hex to Number converts hexadecimals to numbers.

See the *Scripting Guide*.

Repeat

Creates a string that is the first argument repeated the number of times specified by the second argument. The first argument can be either a character literal, a character variable, or a character expression. For example, **Repeat("Katie", 3)** creates **KatieKatieKatie**.

A third argument applies when **Repeat** is used in a JSL script to repeat a matrix. When the first argument is a matrix, the second argument is the rowwise repeat and the third argument is the columnwise repeat.

Insert

Insert inserts a new item into the *list* or *expression* at the given position. If position is not given, it is inserted at the end.

Reverse

Reverse reverses the characters in the string.

Substitute

The first argument is a string, the second is a pattern, and the third is a replacement string. **Substitute** finds all matches to the pattern in the string, and replaces them with the replacement string.

Regex

The first argument is the source string that Regex searches for a match to the pattern. The second argument is the pattern, in the form of a regular expression. The Formula Editor prompts you for these two required arguments.

Tip: For more information about using regular expressions, search the Internet for *regular expression tutorial*.

By default, Regex performs a case-sensitive search and returns the parts of the source string that match the pattern that you specified (or returns MISSING if the match fails). There are two optional arguments that you can add. You can type a third argument—the *format*—that specifies the string to return. If you choose, you can use regular expressions to specify replacement text in the returned string. If you specify the third argument, you can also specify **IGNORECASE** so that Regex ignores capitalization when searching the source string for a match.

Table A.1 Regex Examples

Sample Regex function	String that is returned
`Regex( "@ q3 #", "([a-z])([0-9])" )`	q3 The function is case sensitive, so q3 matches but Q3 would not.
`Regex( "@ Q3 #", "([a-z])([0-9])", "\0",IGNORECASE)`	Q3 Although `\0` is the default argument, it is required in this example so that IGNORECASE can be specified.
`Regex( "@ Q3 #", "([a-z])([0-9])", "\2\1",IGNORECASE)`	3Q

For more information and an example that you can run, select **Help > Scripting Index** and do a search for **Regex**.

Comparison Functions

You can create a formula that compare the values of two arguments by using the comparison function. Each comparison relationship evaluates as *true* or *false* based on numeric magnitudes or character rankings. A true relationship evaluates as one, and false evaluates as zero.

Comparisons are useful when you include them in conditional expressions, but they can also stand alone as numeric expressions if neither term in comparison is missing. A relational symbol's arguments can be any two expressions. However, both arguments in a comparison function must be of the same data type. Also note that:

- JMP displays an error if you use a single "=" in a conditional where "==" is expected.
- The Formula Editor uses the International Utilities package when comparing character strings. This package contains different rankings for each international character set and takes diacritical marks into consideration.
- You should not use comparison operators to specifically compare to a missing value. Instead, use the **Is Missing** function to detect a missing value.

See the *JSL Syntax Reference* for more information about syntax.

< Less than

> Greater than

<= Less than or equal to

>= Greater than or equal to

== Equal to

!= Not equal to

a<b<=c *b* is greater than *a* and less than or equal to *c*

a<=b<c *b* is greater than or equal to *a* and less than *c*

Is Missing Returns a one (1) if the value of the argument for the current row is missing, and a zero if the value is not missing.

Conditional Functions

You can include conditional expressions (called *conditionals* for short) in your formulas. These expressions let you build a sequence of clauses paired with result expressions. Constructing a sequence of clauses is the way you conditionally assign values to cells in a calculated column. Conditionals follow these rules:

- When no clause is true, the Formula Editor evaluates the result expression that accompanies the else clause.
- All result expressions in a conditional expression must evaluate to the same data type.
- A missing term matches any data type.
- By definition, expressions that evaluate as zero are false.
- If an expression evaluates as missing, no clauses are executed and missing is returned. All other numeric expressions are true.

See the *Scripting Guide* and the *JSL Syntax Reference* for more information about syntax.

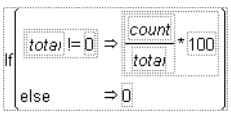
Use the insert and delete clause buttons on the Formula Editor panel to expand the expression. For maximum efficiency, list the most frequently evaluated clause and result pairs first in the sequence.

Note: Interpolate, **Step**, **For**, and **While** are most often used in conjunction with other commands to build a JSL script. You can use the Formula Editor to create and execute a script in that column, but this is not recommended because of dependencies and ambiguities that can result. Most often, scripts are stored as .jsl files, and can be saved with a data table as a table property. For more information about table properties, see "Table Panel in Data Tables". For documentation of scripting commands, see the *JSL Syntax Reference*.

If

Shows a single **If** condition with a missing expression and a missing **then clause**. Highlight either **expr** or **then clause** and enter a value. For example, to calculate count as a percentage of total when total is not 0, enter the conditional expression (using columns called **count** and **total**) in Figure A.3.

Figure A.3 A Conditional Expression



To add a new condition to the **If** conditional, highlight **then clause** and click the insert button ( ) on the Formula Editor keypad. Initially, this changes the existing **else** condition to an **expr** clause. Click the insert button again to add an **else clause**. Highlighting **then** or **else** and repetitively clicking the insert button changes the **else** to **expr** or adds a new **expr** clause.

To delete a clause, select the **then clause** above it and press Delete or click the Delete button ( ) on the Formula Editor keypad.

By definition, expressions that evaluate as zero are false. If an expression evaluates as missing, no clauses are executed and missing is returned. All other numeric expressions are true.

Match

Compares an expression to a list of clauses and returns the value of the resulting expression for the first matching clause encountered. You provide the matching expression only once and then give a match value for each clause.

After you select **Match** in the Formula Editor, a list appears with two options:

- Select **Add Match Arguments from Data**, and clauses that correspond to all of the levels in your data are added automatically. Alternatively, press Shift, select **Conditional**, and then select **Match**. In Figure A.4, the example on the left shows clauses that were added automatically.
- Select **Don't Add** so that you can add each clause individually. In Figure A.4, the example on the right shows an empty clause, which you fill with the missing expressions.

Figure A.4 Examples of Using the Match Function

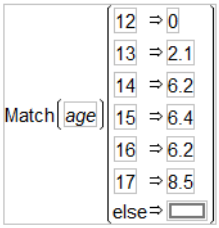
In an automatically filled argument, you should highlight **then clause**, and then enter an expression. In an empty argument, you highlight either **expr**, **value**, or **then clause**, and then enter an expression. (Or, if you highlight an expression and click **Match**, the Formula Editor creates a new **Match** conditional, with the original highlighted expression as **expr** and nothing for the **value** and **else clause**.) Also, keep in mind that:

- **Match** evaluates faster and uses less memory than an equivalent **If** because the variable is evaluated only once for each row in the data table. The **If** condition must evaluate the variable at each **If** clause for each row until a clause evaluates as true. See "Comparison Functions", for a comparison of **Match** and **If** conditionals.

- With **If** and **Match**, the Formula Editor searches down from the top of the sequence for the first true clause and evaluates the corresponding result expression. Subsequent true clauses are ignored.

In the following example, each value is assigned depending on the value of the *age* variable.

Figure A.5 An Example of Using the Match Function



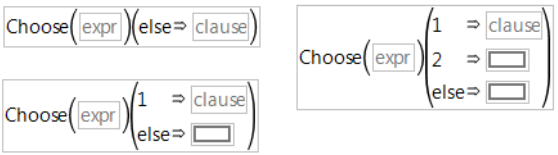
Note: Match ignores trailing spaces and **If** does not.

Although **Match** returns missing for any missing values, you can also specifically match missing values.

Choose

Choose is a special case of **Match** in which the arguments of the condition are a sequence of integers starting at one. The value of **clause** replaces the match condition. An example of a **Choose** condition is shown in Figure A.6. With **Choose**, the Formula Editor goes directly to the correct choice clause and evaluates the result expression.

Figure A.6 Example of a **Choose** Condition



When you highlight an expression and click **Choose**, the Formula Editor creates a new conditional expression with one **clause**. Use the insert ( ) and delete ( ) buttons on the keypad to add new clauses or remove unwanted clauses, as described previously for the **If** conditional.

Choose evaluates the **Choose** expression and goes immediately to the corresponding result expression to generate the returned value. With **Choose**, you provide a choosing expression that yields sequential integers starting at 1 only once, and then you give a choice for each integer in the sequence.

IfMax

Evaluates the first of each pair of arguments and returns the evaluation of the result expression (the second of each pair) associated with the maximum of the expressions. If more than one expression is the maximum, the first maximum is returned. If all expressions are missing and a final result is not specified, missing is returned. If all expressions are missing and a final result is specified, that final result is returned. The test expressions must evaluate to numeric values, but the result expressions can be anything.

IfMin

Evaluates the first of each pair of arguments and returns the evaluation of the result expression (the second of each pair) associated with the minimum of the expressions. If more than one expression is the minimum, the first minimum is returned. If all expressions are missing and a final result is not specified, missing is returned. If all expressions are missing and a final result is specified, that final result is returned. The test expressions must evaluate to numeric values, but the result expressions can be anything.

And &

Evaluates as 1 when both of its arguments are true. Otherwise, it evaluates as 0 (Figure A.9.) The formula in Figure A.7 labels Group 1 as drivers only if both comparisons are true.

Figure A.7 Creating an **And** Function

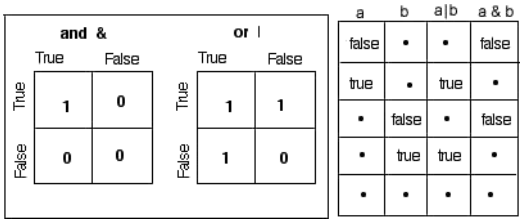
Or |

Evaluates as 1 when either of its arguments is true. If both of its arguments are false, then the `Or` expression evaluates as 0 (Figure A.9.) The formula in Figure A.8 assigns males and all participants who are more than 13 years old to Group 1.

Figure A.8 Creating an **Or** Function

The truth tables on the left in Figure A.9 illustrate the results of the **And** (&) and **Or** (|) functions when both arguments have nonmissing values that evaluate to true or false. The table on the right illustrates the result when either the left or right expression (call them **a** and **b**) or both have missing values.

Figure A.9 Evaluations of And and Or Expressions



a	b	a|b	a & b
false	•	•	false
true	•	true	•
•	false	•	false
•	true	true	•
•	•	•	•

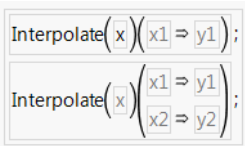
Not !

Evaluates as 1 when its argument is false. Otherwise, **Not** evaluates as **0**. When you apply the **Not** function, use parentheses where necessary to avoid ambiguity. For example, `!(weight==64)` can be either true or false (either 1 or 0), but `(!weight)==64` is always false (0) because `Not` can return only 0 or 1. Expressions such as `!(weight==64)` can also be entered as `weight != 64`.

Interpolate

Linearly interpolates the *y*-value between two points, *x1*, *y1* and *x2*, *y2* that corresponds to the arguments that you give. You can insert additional pairs of *x*, *y* arguments with the Insert key. **Interpolate** finds the pair of *x*, *y* points that correspond to the *x*-value and completes the interpolation.

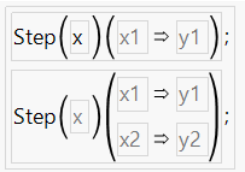
Figure A.10 Examples of Interpolate



Step

Step is like **Interpolate** except that it returns the *y*-value corresponding to the greatest *x*-value less than or equal to the *x* and *y* arguments. That is, it finds the corresponding *y* for a given *x* from a step function rather than a linear fit between points. Like **Interpolate**, you can have as many *x* and *y* argument pairs as you want.

Figure A.11 Examples of Step



For

Repeats the statements in the body argument as long as the **while** condition is true. The **init** and **next** control the iterations.

While

Repeatedly tests the **expr** condition and executes the **body** until **expr** is no longer true.

Break, Continue

Break stops execution of a loop completely and continues to the statement following the loop. **Continue** ends the current iteration of a loop and begins the loop at the next iteration.

Both are used in **For**, **While**, and **For Each Row** loops.

Stop

Immediately stops a script that is running.

Return

Returns an expression value from a user-defined function.

Probability Functions

You can build formulas that calculate probabilities and quantiles for statistical distributions including the beta, Chi-square, F, gamma, normal, Student's t, and Weibull distributions. See the *JSL Syntax Reference* for more information about distribution parameterizations and function arguments.

Beta Density

For information about the syntax for the Beta Density function, see the *JSL Syntax Reference*.

Beta Distribution

For information about the syntax for the Beta Distribution function, see the *JSL Syntax Reference*.

Beta Quantile

For information about the syntax for the Beta Quantile function, see the *JSL Syntax Reference*.

Cauchy Density

For information about the syntax for the Cauchy Density function, see the *JSL Syntax Reference*.

Cauchy Distribution

For information about the syntax for the Cauchy Distribution function, see the *JSL Syntax Reference*.

Cauchy Quantile

For information about the syntax for the Cauchy Quantile function, see the *JSL Syntax Reference*.

ChiSquare Density

For information about the syntax for the ChiSquare Density function, see the *JSL Syntax Reference*.

ChiSquare Distribution

For information about the syntax for the ChiSquare Distribution function, see the *JSL Syntax Reference*.

ChiSquare Log CDistribution

For information about the syntax for the ChiSquare Log CDistribution function, see the *JSL Syntax Reference*.

ChiSquare Log Density

For information about the syntax for the ChiSquare Log Density function, see the *JSL Syntax Reference*.

ChiSquare Log Distribution

For information about the syntax for the ChiSquare Log Distribution function, see the *JSL Syntax Reference*.

ChiSquare Noncentrality

For information about the syntax for the ChiSquare Noncentrality function, see the *JSL Syntax Reference*.

ChiSquare Quantile

For information about the syntax for the ChiSquare Quantile function, see the *JSL Syntax Reference*.

Dunnett P Value

For information about the syntax for the Dunnett P Value function, see the *JSL Syntax Reference*.

Dunnett Quantile

For information about the syntax for the Dunnett Quantile function, see the *JSL Syntax Reference*.

Exp Density

For information about the syntax for the Exp Density function, see the *JSL Syntax Reference*.

Exp Distribution

For information about the syntax for the Exp Distribution function, see the *JSL Syntax Reference*.

Exp Quantile

For information about the syntax for the Exp Quantile function, see the *JSL Syntax Reference*.

F Density

For information about the syntax for the F Density function, see the *JSL Syntax Reference*.

F Distribution

For information about the syntax for the F Distribution function, see the *JSL Syntax Reference*.

F Log CDistribution

For information about the syntax for the F Log CDistribution function, see the *JSL Syntax Reference*.

F Log Density

For information about the syntax for the F Log Density function, see the *JSL Syntax Reference*.

F Log Distribution

For information about the syntax for the F Log Distribution function, see the *JSL Syntax Reference*.

F Noncentrality

For information about the syntax for the F Noncentrality function, see the *JSL Syntax Reference*.

F Power

For information about the syntax for the F Power function, see the *JSL Syntax Reference*.

F Quantile

For information about the syntax for the F Quantile function, see the *JSL Syntax Reference*.

F Sample Size

For information about the syntax for the F Sample Size function, see the *JSL Syntax Reference*.

Frechet Density

For information about the syntax for the Frechet Density function, see the *JSL Syntax Reference*.

Frechet Distribution

For information about the syntax for the Frechet Distribution function, see the *JSL Syntax Reference*.

Frechet Quantile

For information about the syntax for the Frechet Quantile function, see the *JSL Syntax Reference*.

Gamma Density

For information about the syntax for the Gamma Density function, see the *JSL Syntax Reference*.

Gamma Distribution

For information about the syntax for the Gamma Distribution function, see the *JSL Syntax Reference*.

Gamma Log CDistribution

For information about the syntax for the Gamma Log CDistribution function, see the *JSL Syntax Reference*.

Gamma Log Density

For information about the syntax for the Gamma Log Density function, see the *JSL Syntax Reference*.

Gamma Log Distribution

For information about the syntax for the Gamma Log Distribution function, see the *JSL Syntax Reference*.

Gamma Quantile

For information about the syntax for the Gamma Quantile function, see the *JSL Syntax Reference*.

GenGamma Density

For information about the syntax for the GenGamma Density function, see the *JSL Syntax Reference*.

GenGamma Distribution

For information about the syntax for the GenGamma Distribution function, see the *JSL Syntax Reference*.

GenGamma Quantile

For information about the syntax for the GenGamma Quantile function, see the *JSL Syntax Reference*.

GLog Density

For information about the syntax for the GLog Density function, see the *JSL Syntax Reference*.

GLog Distribution

For information about the syntax for the GLog Distribution function, see the *JSL Syntax Reference*.

GLog Quantile

For information about the syntax for the GLog Quantile function, see the *JSL Syntax Reference*.

Johnson Sb Density

For information about the syntax for the Johnson Sb Density function, see the *JSL Syntax Reference*.

Johnson Sb Distribution

For information about the syntax for the Johnson Sb Distribution function, see the *JSL Syntax Reference*.

Johnson Sb Quantile

For information about the syntax for the Johnson Sb Quantile function, see the *JSL Syntax Reference*.

Johnson Sl Density

For information about the syntax for the Johnson Sl Density function, see the *JSL Syntax Reference*.

Johnson Sl Distribution

For information about the syntax for the Johnson Sl Distribution function, see the *JSL Syntax Reference*.

Johnson Sl Quantile

For information about the syntax for the Johnson Sl Quantile function, see the *JSL Syntax Reference*.

Johnson Su Density

For information about the syntax for the Johnson Su Density function, see the *JSL Syntax Reference*.

Johnson Su Distribution

For information about the syntax for the Johnson Su Distribution function, see the *JSL Syntax Reference*.

Johnson Su Quantile

For information about the syntax for the Johnson Su Quantile function, see the *JSL Syntax Reference*.

LEV Density

For information about the syntax for the Largest Extreme Value (LEV) Density function, see the *JSL Syntax Reference*.

LEV Distribution

For information about the syntax for the Largest Extreme Value (LEV) Distribution function, see the *JSL Syntax Reference*.

LEV Quantile

For information about the syntax for the Largest Extreme Value (LEV) Quantile function, see the *JSL Syntax Reference*.

LogGenGamma Density

For information about the syntax for the LogGenGamma Density function, see the *JSL Syntax Reference*.

LogGenGamma Distribution

For information about the syntax for the LogGenGamma Distribution function, see the *JSL Syntax Reference*.

LogGenGammaQuantile

For information about the syntax for the LogGenGamma Quantile function, see the *JSL Syntax Reference*.

Logistic Density

For information about the syntax for the Logistic Density function, see the *JSL Syntax Reference*.

Logistic Distribution

For information about the syntax for the Logistic Distribution function, see the *JSL Syntax Reference*.

Logistic Quantile

For information about the syntax for the Logistic Quantile function, see the *JSL Syntax Reference*.

Loglogistic Density

For information about the syntax for the Loglogistic Density function, see the *JSL Syntax Reference*.

Loglogistic Distribution

For information about the syntax for the Loglogistic Distribution function, see the *JSL Syntax Reference*.

Loglogistic Quantile

For information about the syntax for the Loglogistic Quantile function, see the *JSL Syntax Reference*.

Lognormal Density

For information about the syntax for the Lognormal Density function, see the *JSL Syntax Reference*.

Lognormal Distribution

For information about the syntax for the Lognormal Distribution function, see the *JSL Syntax Reference*.

Lognormal Quantile

For information about the syntax for the Lognormal Quantile function, see the *JSL Syntax Reference*.

Normal Biv Distribution

For information about the syntax for the Normal Biv Distribution function, see the *JSL Syntax Reference*.

Normal Density

For information about the syntax for the Normal Density function, see the *JSL Syntax Reference*.

Normal Distribution

For information about the syntax for the Normal Distribution function, see the *JSL Syntax Reference*.

Normal Log CDistribution

For information about the syntax for the Normal Log CDistribution function, see the *JSL Syntax Reference*.

Normal Log Density

For information about the syntax for the Normal Log Density function, see the *JSL Syntax Reference*.

Normal Log Distribution

For information about the syntax for the Normal Log Distribution function, see the *JSL Syntax Reference*.

Normal Mixture Density

For information about the syntax for the Normal Mixture Density function, see the *JSL Syntax Reference*.

Normal Mixture Distribution

For information about the syntax for the Normal Mixture Distribution function, see the *JSL Syntax Reference*.

Normal Mixture Quantile

For information about the syntax for the Normal Mixture Quantile function, see the *JSL Syntax Reference*.

Normal Quantile

For information about the syntax for the Normal Quantile function, see the *JSL Syntax Reference*.

SEV Density

For information about the syntax for the Smallest Extreme Value (SEV) Density function, see the *JSL Syntax Reference*.

SEV Distribution

For information about the syntax for the Smallest Extreme Value (SEV) Distribution function, see the *JSL Syntax Reference*.

SEV Quantile

For information about the syntax for the Smallest Extreme Value (SEV) Quantile function, see the *JSL Syntax Reference*.

SHASH Density

For information about the syntax for the SHASH Density function, see the *JSL Syntax Reference*.

SHASH Distribution

For information about the syntax for the SHASH Distribution function, see the *JSL Syntax Reference*.

SHASH Quantile

For information about the syntax for the SHASH Quantile function, see the *JSL Syntax Reference*.

t Density

For information about the syntax for the t Density function, see the *JSL Syntax Reference*.

t Distribution

For information about the syntax for the t Distribution function, see the *JSL Syntax Reference*.

t Log CDistribution

For information about the syntax for the t Log CDistribution function, see the *JSL Syntax Reference*.

t Log Density

For information about the syntax for the t Log Density function, see the *JSL Syntax Reference*.

t Log Distribution

For information about the syntax for the t Log Distribution function, see the *JSL Syntax Reference*.

t Noncentrality

For information about the syntax for the t Noncentrality function, see the *JSL Syntax Reference*.

t Quantile

For information about the syntax for the t Quantile function, see the *JSL Syntax Reference*.

Tukey HSD P Value

For information about the syntax for the Tukey HSD P Value function, see the *JSL Syntax Reference*.

Tukey HSD Quantile

For information about the syntax for the Tukey HSD Quantile function, see the *JSL Syntax Reference*.

Weibull Density

For information about the syntax for the Weibull Density function, see the *JSL Syntax Reference*.

Weibull Distribution

For information about the syntax for the Weibull Distribution function, see the *JSL Syntax Reference*.

Weibull Quantile

For information about the syntax for the Weibull Quantile function, see the *JSL Syntax Reference*.

Discrete Probability Functions

Beta Binomial Distribution

For information about the syntax for the Beta Binomial Distribution function, see the *JSL Syntax Reference*.

Beta Binomial Probability

For information about the syntax for the Beta Binomial Probability function, see the *JSL Syntax Reference*.

Beta Binomial Quantile

For information about the syntax for the Beta Binomial Quantile function, see the *JSL Syntax Reference*.

Binomial Distribution

For information about the syntax for the Binomial Distribution function, see the *JSL Syntax Reference*.

Binomial Probability

For information about the syntax for the Binomial Probability function, see the *JSL Syntax Reference*.

Binomial Quantile

For information about the syntax for the Binomial Quantile function, see the *JSL Syntax Reference*.

Gamma Poisson Distribution

For information about the syntax for the Gamma Poisson Distribution function, see the *JSL Syntax Reference*.

Gamma Poisson Probability

For information about the syntax for the Gamma Poisson Probability function, see the *JSL Syntax Reference*.

Gamma Poisson Quantile

For information about the syntax for the Gamma Poisson Quantile function, see the *JSL Syntax Reference*.

Hypergeometric Distribution

For information about the syntax for the Hypergeometric Distribution function, see the *JSL Syntax Reference*.

Hypergeometric Probability

For information about the syntax for the Hypergeometric Probability function, see the *JSL Syntax Reference*.

Neg Binomial Distribution

For information about the syntax for the Neg Binomial Distribution function, see the *JSL Syntax Reference*.

Neg Binomial Probability

For information about the syntax for the Neg Binomial Probability function, see the *JSL Syntax Reference*.

Poisson Distribution

For information about the syntax for the Poisson Distribution function, see the *JSL Syntax Reference*.

Poisson Probability

For information about the syntax for the Poisson Probability function, see the *JSL Syntax Reference*.

Poisson Quantile

For information about the syntax for the Poisson Quantile function, see the *JSL Syntax Reference*.

Statistical Functions

There are two types of **Statistical** functions that you can use in a formula:

- The functions with names that have the prefix **Col**. These functions compute statistics for a column of numbers or expressions involving columns.
- The **Mean**, **Std Dev**, **Number**, **Sum**, **Quantile**, **Maximum**, **Minimum**, and **N Missing** functions. These functions evaluate across columns or arguments. The statistic is computed for each row across the series of arguments. You can click the Insert key ( ) on the Formula Editor keypad, or type a comma to add arguments to the functions that accept multiple arguments. When there are multiple contiguous arguments, select the function and the first argument, and then press Shift and click the last argument in the group. These functions then automatically show with the complete list.

Note: Missing values are ignored in statistical functions.

See the *JSL Syntax Reference* for more information about syntax.

Col Cumulative Sum

Returns the cumulative sum for the current row. Col Cumulative Sum supports By columns, which do not need to be sorted.

Col Mean

Calculates the mean (or arithmetic average) of the numeric values identified by its argument. The formula **Col Mean(age)** calculates the average of all nonmissing values in the **age** column.

Col Moving Average

Returns the moving average over a given interval based at the current row. Missing values are ignored

Col Std Dev

Measures the spread around the mean of the distribution identified by its argument. In the normal distribution, about 68% of the distribution is within one standard deviation of the mean. 95% of the distribution is within two standard deviations of the mean. 99% of the distribution is within three standard deviations of the mean.

Col Number

Counts the number of nonmissing values in the column that you specify. A missing numeric value occurs when a cell has no assigned value or is the result of an invalid operation (such as division by zero). Missing values show on the spreadsheet as a missing value mark (•). Missing character values are null character strings. In formulas for row state columns, an excluded row state characteristic is treated as a missing value.

Col N Missing

Counts the number of missing values in the column that you specify. A missing numeric value occurs when a cell has no assigned value or is the result of an invalid operation (such as division by zero). Missing values show in the data grid with a missing value character (•). Missing character values are null character strings.

Col Sum

Computes the sum of the values in its numeric argument. Missing values are ignored.

Col Minimum and Col Maximum

Takes the minimum of its numeric arguments. **Col Minimum** ignores missing values. **Col Maximum** takes the maximum of a numeric column argument and ignores missing values.

Col Quantile

Computes the value at which a specific percentage of the values is less than or equal to that value. For example, the value calculated as the 50% quantile, also called the median, is greater than or equal to 50% of the data. Half of the data values are less than the 50^{th} quantile.

The **Col Quantile** function's quantile argument represents the quantile percentage divided by 100. The 25% quantile, also called the lower quartile, corresponds to $p = 0.25$, and the 75% quantile, called the upper quartile, corresponds to $p = 0.75$.

The Formula Editor computes a quantile for a column of n nonmissing values by arranging the values in ascending order. The subscripts of the sorted column values, $y_1, y_2, \ldots, y_n$, represent the ranks in ascending order.

The pth quantile value is calculated using the formula $p(n + 1)$, where p is the percent value and n is the total number of nonmissing values. If $p(n+1)$ is an integer, then the quantile value is $y_{p(n+1)}$. If $p(n + 1)$ is not an integer, then the value is interpolated by assigning the integer part of the result to i, assigning the fractional part to f, and applying the formula $(1 - f)y_i + (f)y_{i+1}$.

For example, suppose a column has values 2, 4, 6, 8, 10, 12, 14, 16, 18, and 20. The 50% quantile is calculated as `0.5(10 + 1) = 5.5`.

Because the result is fractional, the 50% quantile value is interpolated as

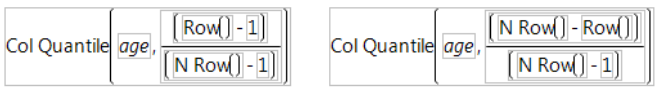
$$(1 - 0.5) \times 10 + (0.5) \times 12 = (0.5)10 + (0.5)12 = 6 + 5 = 11$$

The following are example **ColQuantile** formulas:

- **ColQuantile(age, 1)** Calculates the maximum age.
- **ColQuantile(age, 0.75)** Calculates the upper quartile age.
- **ColQuantile(age, 0.5)** Calculates the median age.
- **ColQuantile(age, 0.25)** Calculates the lower quartile age.
- **ColQuantile(age, 0)** Calculates the minimum age.

The **ColQuantile** argument can be any expression that evaluates to a value between (and including) 0 and 1. For example, the first formula in Figure A.12 calculates quantile values of age in ascending order for each row. The column then contains the interpolated values of **age** in ascending order in the calculated column. The second formula lists the interpolated values of age in descending order.

Figure A.12 Examples of the Quantile Function



Col Rank

Ranks each row's value, from 1 for the lowest value to the number of non-missing columns for the highest value. Ties can be broken by reporting the average of the possible rankings, assigning the ranks in the order that they originally appear, giving both values the lowest possible rank, or arbitrarily. (JMP 12 used arbitrary tie breaking.) Distribution's Normal Quantile output uses average tie breaking.

Col Standardize

Performs the usual standardization on its numeric expression. For each row *i*, **Col Standardize(height)** is **(Height$_{Row()}$–Col Mean(Height))/Col Std Dev(Height)**.

Mean

Calculates the arithmetic average of the nonmissing values in the list of arguments that you specify. The arguments can be constants, numbers, or expressions. The **Mean** function initially shows with a single argument. You add arguments with the insert button ( ) on the Formula Editor keypad or by typing a comma.

Std Dev

Computes standard deviation of the nonmissing values in the list of arguments that you specify. The arguments can be constants, numbers, or expressions. The **Std Dev** function initially shows with a single argument. You add arguments by clicking the insert button ( ) on the Formula Editor keypad or by typing a comma.

Number

Counts the number of nonmissing values in the list of arguments that you specify.

Sum

Returns the sum of the nonmissing values in the list of arguments that you specify.

Quantile

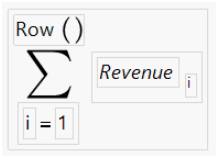
Calculates the quantile specified by the first argument for all the nonmissing values in the remaining list of arguments that you specify.

Summation (Σ)

Evaluates for an explicit range of values in a column, as given by the summation indices. This behavior is different from all other statistical functions (except **Product**), which *always* evaluate *on every row*. The **Summation** function uses the summation notation shown in Figure A.13. To calculate a sum, replace the missing *body* term with an expression containing the index variable *i*, or an index variable that you assign. **Summation** repeatedly evaluates the expression for *i* = 1, *i* = 2, through *i* = **NRow()** and then adds the nonmissing results together to determine the final result.

You can replace **NRow()**, the number of rows in the active spreadsheet, and the index constant, *i*, with any expression appropriate for your formula. For example, the summation formula in Figure A.13 computes the total for each row of all revenue values for rows 1 through the current row number, filling the calculated column with the cumulative totals of the revenue column.

Figure A.13 Example of the **Summation** function



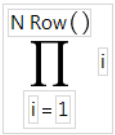
Product (Π)

Evaluates for an explicit range of values in a column, as given by the summation indices, as opposed to all other statistical functions (except **Summation**), which *always* evaluate *on every row*. **Product** uses the notation shown in the formula in Figure A.14. To calculate a product, replace the missing body term with an expression containing the index variable *j*. **Product** repeatedly evaluates the expression for $i = 1$, $i = 2$, through $i = n$ and multiplies the nonmissing results together to determine the final result.

You can replace **NRow()**, the number of rows in the active spreadsheet and the index constant, *i*, with any expression appropriate for your formula.

For example, the product example in Figure A.14 calculates $i!$ (each row number's factorial).

Figure A.14 Examples of the **Product** Function



Minimum and Maximum

Return the minimum and maximum value, respectively, from the list of nonmissing arguments that you specify.

N Missing

Counts the number of missing values in the list of arguments that you specify.

SSQ

Returns the sum of squares of all elements.

Random Functions

You can create formulas that generate random numbers by effectively "rolling the dice" within the constraints of the specified distribution. Each time you click **Apply** in the Formula Editor window, these functions produce a new set of random numbers.

You can use the **Random Reset** function to set a random seed that enables you to reproduce a set of random numbers. The random number algorithms in JMP are sometimes refined, and therefore the results might not be consistent across different versions of JMP.

Note: Random numbers are generated using the Mersenne-Twister technique. This technique has a period length of 2^{19937}-1. For more information about the generators, see Matsumoto and Nishimura (1998). The new generators are verified to pass all the DIEHARD tests as documented in Marsaglia (1995).

See the *JSL Syntax Reference* for more information about function arguments. See the *JSL Syntax Reference* for more information about distribution parameterizations.

Random ChiSquare

Generates a column of random numbers from a Chi-Square distribution with specified degrees of freedom. The optional noncentrality parameter must be greater than or equal to 0. The noncentrality parameter is 0 by default. For more information about the parameterization of the Chi-Square distribution, see the *JSL Syntax Reference*.

Random F

Generates a column of random numbers from an F distribution with specified numerator degrees of freedom and denominator degrees of freedom. The optional noncentrality parameter must be greater than or equal to 0. The noncentrality parameter is 0 by default. For more information about the parameterization of the F distribution, see the *JSL Syntax Reference*.

Random Index

Returns a k by 1 matrix of random integers between 1 and n with no duplicates.

Random Uniform

Generates a column of random numbers from a uniform distribution. If no arguments are specified, the minimum of the uniform distribution is 0 and the maximum is 1. This means that any number between 0 and 1 is as likely to be generated as any other. The result is an approximately even distribution. You can use the optional arguments to specify different values for the minimum and maximum numbers of the uniform distribution.

Random Normal

Generates a column of random numbers from a normal distribution with specified mean and standard deviation. If no arguments are specified, the normal distribution has mean of 0 and standard deviation of 1. For more information about the parameterization of the normal distribution, see the *JSL Syntax Reference*.

Random Normal Mixture

Returns a random number from a normal mixture distribution with the specified arguments. For information about the syntax for the Random Normal Mixture function, see the *JSL Syntax Reference*.

Random Exp

Generates a column of random numbers from an exponential distribution with lambda=1. You can scale the exponential function to use a different lambda. For example, `Random Exp()*.1` generates an exponential distribution for lambda=0.1. For more information about the parameterization of the exponential distribution, see the *JSL Syntax Reference*.

Random Gamma

Generates a column of random numbers from a gamma distribution with specified shape and scale parameters. The default value of the scale parameter is 1. When the scale parameter is 1, the gamma distribution is equivalent to an exponential distribution. For more information about the parameterization of the gamma distribution, see the *JSL Syntax Reference*.

Random GenGamma

Generates a column of random numbers from an extended generalized gamma distribution with specified location, scale, and shape parameters. If no arguments are specified, the generalized gamma distribution has location of 0, scale of 1, and shape of 0. For more information about the parameterization of the generalized gamma distribution, see the *JSL Syntax Reference*.

Random LogGenGamma

Generates a column of random numbers from a log generalized gamma distribution with specified location, scale, and shape parameters. If no arguments are specified, the extended generalized gamma distribution has location of 0, scale of 1, and shape of 0. For more information about the parameterization of the log generalized gamma distribution, see the *JSL Syntax Reference*.

Random Beta

Generates a column of random numbers from a beta distribution with two specified shape parameters and optional threshold and scale parameters. The default threshold is 0, and the default scale is 1. For more information about the parameterization of the beta distribution, see the *JSL Syntax Reference*.

Random Cauchy

Generates a column of random numbers from a Cauchy distribution with location parameter 0 and scale parameter 1. A Cauchy variate with location parameter alpha and scale parameter beta can be generated with the formula `alpha+beta*Random Cauchy()`. For more information about the parameterization of the Cauchy distribution, see the *JSL Syntax Reference*.

Random Category

Generates a column of random category assignments based on a set of probability and result expressions (for example, `Random Category(.2, "A", .3, "B", .4, "C", "D");`). A random uniform number is generated and compared to the probabilities to determine which result expression is returned.

Random Johnson Su

Generates a column of random numbers from a Johnson Su distribution with two specified shape parameters, a location parameter, and a scale parameter. For more information about the parameterization of the Johnson Su distribution, see the *JSL Syntax Reference*.

Random Johnson Sb

Generates a column of random numbers from a Johnson Sb distribution with two specified shape parameters, a location parameter, and a scale parameter. For more information about the parameterization of the Johnson Sb distribution, see the *JSL Syntax Reference*.

Random Johnson Sl

Generates a column of random numbers from a Johnson Sl distribution with two specified shape parameters, a location parameter, and an optional skewness parameter. The skewness parameter can be +1 for positive skew or -1 for negative skew. The default is +1. For more information about the parameterization of the Johnson Sl distribution, see the *JSL Syntax Reference*.

Random Seed State

Retrieves or restores the random seed state to or from a BLOB object.

Random Triangular

Generates a column of random numbers from a triangular distribution with specified minimum, maximum, and mode parameters. If you specify only one argument, it represents the mode and the triangular distribution is between 0 and 1. If you specify two arguments, they represent the mode and the maximum, respectively, and the triangular distribution is between 0 and the maximum. If you specify three arguments, they represent the minimum, the mode, and the maximum, respectively, and the triangular distribution is between the minimum and the maximum.

Random Integer

Generates a column of random numbers from a uniform distribution of integers. If you specify one argument, the integers are between 1 and the argument. If you specify two arguments, the integers are between and include the values of the two arguments.

Random Binomial

Generates a column of random numbers from a binomial distribution with specified parameters. The first argument is *n*, the number of trials in a binomial experiment. The second argument is *p*, the probability that the event of interest occurs. When *n* is 1, the binomial function generates a distribution of Bernoulli trials. For more information about the parameterization of the binomial distribution, see the *JSL Syntax Reference*.

Random Negative Binomial

Generates a column of random numbers from a negative binomial distribution with specified parameters. The first argument is *r*, the number of successes of interest. The second argument is *p*, the probability of success. The random variable of interest is the number of failures that precede the r^{th} success. In contrast to the binomial variate, where the number of trials is fixed and the number of successes is variable, the negative binomial variate is for a fixed number of successes and a random number of trials. For more information about the parameterization of the negative binomial distribution, see the *JSL Syntax Reference*.

Random Beta Binomial

Generates a column of random numbers from a beta binomial distribution with specified parameters. The first argument is *n*, the number of trials. The second argument is *p*, the probability of success for each trial. The third argument is optional and represents the overdispersion parameter. The default overdispersion parameter is 0. For more information about the parameterization of the beta binomial distribution, see the *JSL Syntax Reference*.

Random Frechet

Generates a column of random numbers from a Fréchet distribution with specified location and scale parameters. If no arguments are specified, the Fréchet distribution has location of 0 and sigma of 1. For more information about the parameterization of the Fréchet distribution, see the *JSL Syntax Reference*.

Random Geometric

Generates a column of random numbers from a geometric distribution with specified probability parameter. The argument is the probability that a specific event occurs at any one trial. The random variable of interest is the number of trials until a specific event occurs for the first time.

Random Poisson

Generates a column of random numbers from a Poisson distribution with specified shape parameter. For more information about the parameterization of the Poisson distribution, see the *JSL Syntax Reference*.

Random Gamma Poisson

Generates a column of random numbers from a gamma Poisson distribution with specified shape and overdispersion parameters. If the second argument is not specified, the gamma Poisson distribution has an overdispersion of 1. For more information about the parameterization of the gamma Poisson distribution, see the *JSL Syntax Reference*.

Random Weibull

Generates a column of random numbers from a Weibull distribution with specified shape and scale parameters. If the second argument is not specified, the Weibull distribution has scale of 1. For more information about the parameterization of the Weibull distribution, see the *JSL Syntax Reference*.

Random Logistic

Generates a column of random numbers from a logistic distribution with specified location and scale parameters. If no arguments are specified, the logistic distribution has location of 0 and scale of 1. For more information about the parameterization of the logistic distribution, see the *JSL Syntax Reference*.

Random Loglogistic

Generates a column of random numbers from a loglogistic distribution with specified location and scale parameters. If no arguments are specified, the loglogistic distribution has location of 0 and scale of 1. For more information about the parameterization of the loglogistic distribution, see the *JSL Syntax Reference*.

Random Lognormal

Generates a column of random numbers from a lognormal distribution with specified location and scale parameters. If no arguments are specified, the lognormal distribution has location of 0 and scale of 1. For more information about the parameterization of the lognormal distribution, see the *JSL Syntax Reference*.

Random GLog

Generates a column of random numbers from a generalized logarithmic distribution with specified location, scale, and shape parameters. When the shape parameter is 0, the generalized logarithmic distribution is equivalent to a lognormal distribution. For more information about the parameterization of the generalized logarithmic distribution, see the *JSL Syntax Reference*.

Random Reset

Restarts the random number sequences with a seed that you specify.

Random LEV

Generates a column of random numbers from a largest extreme value (LEV) distribution with specified location and scale parameters. If no arguments are specified, the LEV distribution has location of 0 and scale of 1. For more information about the parameterization of the LEV distribution, see the *JSL Syntax Reference*.

Random SEV

Generates a column of random numbers from a smallest extreme value (SEV) distribution with specified location and scale parameters. If no arguments are specified, the SEV distribution has location of 0 and scale of 1. For more information about the parameterization of the SEV distribution, see the *JSL Syntax Reference*.

Random SHASH

Generates a column of random numbers from a sinh-arcsinh (SHASH) distribution with two specified shape parameters, a location parameter, and a scale parameter. For more information about the parameterization of the SHASH distribution, see the *JSL Syntax Reference*.

Random Shuffle

Generates a randomly ordered column of numbers from the numbers in the matrix argument.

Random t

Generates a column of random numbers from a t distribution with specified degrees of freedom. The optional noncentrality argument may be negative or positive. The noncentrality parameter is 0 by default. For more information about the parameterization of the t distribution, see the *JSL Syntax Reference*.

Col Shuffle

Selects a row number at random from the current data table. Each row number is selected only once. When **Col Shuffle** is used as a subscript, it returns a value selected at random from the column that serves as its argument. Each value from the original column is assigned only once as **Col Shuffle**'s result. For more information about the syntax for the Col Shuffle function, see the *JSL Syntax Reference*.

Resample Freq

Generates a random selection with replacement frequency counts, suitable for use in bootstrapping. For example, it supports a second **Freq Column** argument, enabling it to do bootstrap samples relating to a pre-existing frequency column specified in the second argument. `Resample Freq()` generates a 100% resample. `ResampleFreq(rate)` generates a `rate` frequency sample. `Resample(rate, column)` generates a sample that is calculated by the rate multiplied by the sum of the specified column. For more information about the syntax for the Resample Freq function, see the *JSL Syntax Reference*.

Date Time Functions

JMP stores dates and times in numeric columns using macOS standard of the *number of seconds since January 1, 1904*. When a column has date values, you can assign a date format to that column by double-clicking a column name and selecting **Date** or **Time** from the **Format** menu. See "Numeric Format Options".

See the *JSL Syntax Reference* for more information about syntax.

In Minutes, In Hours, In Days, In Weeks, In Years

Converts from the units of the function name to the equivalent number of seconds for the argument. The argument must be a number or numeric expression. For example, **In Minutes(2)** yields **120**, and **In Years(1)** yields **31,557,600** (60 seconds * 60 minutes * 24 hours * 365.25 days).

Date DMY, Date MDY

Accepts numeric expressions for day, month, and year and return the associated JMP date. For example, **Date DMY (20, 3, 1991)** and **Date MDY(3, 20, 1991)** evaluate to **2,752,272,000**.

Today

Returns the number of seconds between January 1, 1904 and the current date. For example, at midnight on March 20, 1991 (a Wednesday), the **Today** function returns **2752272000** (2,752,272,000 seconds) and continues counting. If you evaluate the **Today** function later in the day, it reflects the additional seconds.

Day, Month, Year

Returns the day of the month, the month (as a number from 1 to 12), a four-digit year, respectively. The argument for these functions is interpreted as a JMP date. For example, on March 20, 1991:

- **Day(2752272000)** returns the number **20**.
- **Month(2752272000)** returns the number **3**.
- **Year(2752272000)** returns the number **1991**.

Quarter

Returns the annual quarter of a *datetime* value as an integer 1-4.

Hour, Minute, Second

Returns the hour, the minute, and the seconds of a date-time value, respectively. The argument for these functions is interpreted as a JMP date. For example, on March 20, 1991:

- **Hour(2752572649)** returns the number **11**.
- **Minute(2752572649)** returns the number **30**.
- **Second(2752572649)** returns the number **49**.

Day of Week, Day of Year, Week of Year, Time of Day

The argument for these functions is a JMP date. **Day Of Week** returns a number from 1 to 7, where 1 represents Sunday. **Day Of Year** returns the number of days from the beginning of the year. **Week Of Year** returns a number from 1 to 52 based on the rule specified. Rule 1 (default) has weeks start on Sunday with the first Sunday being week 2 and week 1 is a partial week or empty; rule 2 has the first Sunday begins week 1 with any previous days being week 0; rule 3 returns the ISO week number where the week starts on Monday and week 1 is the first week of the year with four days in that year. With ISO weeks, it is possible for the first or last three days of the year to belong to the neighboring year's week number. **Time Of Day** returns a number from 0 to 86399 (time of day in seconds). For example, on Wednesday, March 20, 1991:

- **Day Of Week(2752272000)** returns the number **4**.
- **Day Of Year(2752272000)** returns the number **79**.
- **Week Of Year(2752272000)** returns the number **12**.
- **Time Of Day(2752272000)** returns the number **0**.

Informat

The argument for the **Informat** function is a date character string. For example, **Informat("03/20/1991")** returns the appropriate JMP date value, **2752272000**. JMP can read all the date formats except for **Abbrev Date** and **Long Date**.

Abbrev Date, Long Date, Short Date

The argument for these date functions is a JMP date. They return character strings that are the formatted representation of the argument. For example:

- **Abbrev Date(2752272000)** returns **Wed, Mar 20,1991**.
- **Long Date(2752272000)** returns **Wednesday, March 20, 1991**.
- **Short Date(2752272000)** returns **3/20/91**.

Format

The first argument in the **Format** function is a JMP date. This function returns the character string representation of the date by the date format that you specify in the second argument, which is a quoted string. If you apply this formula to a numeric column, JMP automatically changes the column's data type to character.

You can also supply a column for the first argument and leave the rest blank. The result is the formatted value of the column reference. This can be used to extract value labels of a column when the value labels are turned off.

MDYHMS

The argument of **MDYHMS** is a JMP date. This function shows all date and time fields, appending zeros as time fields if no time information is present. This is useful if a date column is formulated such that not all date information is displayed. The **MDYHMS** function can be used to see all available date and time information.

Date Increment

Adds 1 or more intervals to a starting datetime value. For example, **Date Increment(Today(), "Day", 3)** adds three days to the current date. **Date Increment(Today(), "Year", 3)** adds 3 years to the current date.

Date Difference

Returns the difference of two datetime values. The interval argument can be Second, Minute, Hour, Day, Week, Month, Quarter, Year. The alignment arguments are described here:

Start Used to count the number of times an interval starts.

Actual Used to count whole intervals.

Fractional Used to count fractional intervals.

For example, the following formula returns **207.890243055556**, the number of days between the dates:

```
Date Difference(
   01Jan2010:00:00:00,
   27Jul2010:21:21:57,
   "Day",
   "fractional"
);
```

The following formula returns **207**, the number of completed days between the dates:

```
Date Difference(
   01Jan2010:00:00:00,
   27Jul2010:21:21:57,
   "Day",
   "actual"
);
```

The following formula returns 9, the number of completed hours between the times:

```
Date Difference(
   01Jan2010:00:00:00,
   01Jan2010:09:22:57,
   "Hour",
   "actual"
```

```
);
```

The following formula returns 1, the number of times a new hour started between the times:

```
Date Difference(
   31Dec2010:23:59:59,
   01Jan2011:00:59:59,
   "Hour",
   "start"
);
```

Row State Functions

There are six characteristics that rows in a data table can have: selected, hidden, excluded, labeled, colored, and marked. If you give rows one or more of these characteristics and then create row state data table columns, you can then create a formula that computes and saves row state conditions. (See "Column Properties in JMP", and "Store Information in Row State Columns".)This formula processes row state data just as it would process character and numeric data.

See the *JSL Syntax Reference* for more information about syntax.

Note: A row can be assigned any combination of row states; a row state column can have multiple row states as a value.

Table A.2 describes the type of argument each Row State function requires and what each returns.

Table A.2 Row State Functions

Function Name	Argument Type Required	What the Function Returns (Your Column Data Type Should be This Type)
Row State	none	row state of current row
As Row State	numeric	all row states of current row
Combine States	multiple row state arguments	multiple row state assignments
Excluded State	positive integer or zero	row state-excluded or not excluded
Hidden State	positive integer or zero	row state-hidden or not hidden
Labeled State	positive integer or zero	row state-labeled or not labeled

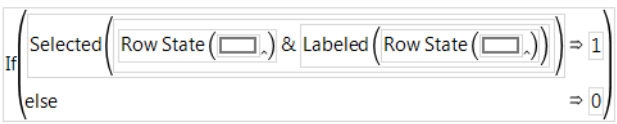
Table A.2 Row State Functions *(Continued)*

Function Name	Argument Type Required	What the Function Returns (Your Column Data Type Should be This Type)
Color State	integer or color name or {*red*, *green*, *blue*}	row state color
Marker State	integer or character	row state marker
Selected State	positive integer or zero	row state-selected or not selected
Hue State	integer	row state hue
Shade State	integer 1-5	row state intensity
Excluded	**Row State()** or row state column	integer 0 (not excluded) or 1 (excluded)
Hidden	**Row State()** or row state column	integer 0 (not hidden) or 1 (hidden)
Labeled	**Row State()** or row state column	integer 0 (not labeled) or 1 (labeled)
Color Of	**Row State()** or row state column	color map integer
Marker Of	**Row State()** or row state column	marker map integer
Selected	**Row State()** or row state column	integer 0 (not selected) or 1 (selected)

Row State

Returns the active row state condition of the current row as true or false. You can use this function to conveniently write conditional clauses that depend on the status of the current row. For example, Figure A.15 assigns a 1 to rows that are currently selected and labeled and 0 otherwise.

Figure A.15 Row State



As Row State

Converts a numeric argument to a row state or set of row state conditions. Row states are stored internally in JMP as a 16-bit number, with each bit assigned to represent one of the possible row states as illustrated in Figure A.3. For example, the binary representation of **1327** is **0000010100101111**. **As Row State(1327)** would therefore set the row state as selected, excluded, hidden, labeled, with marker 2 and color 10.

Table A.3 Row States Stored as 16-Bit Numbers: Each Bit Represents a Row State

Bit	Row State
0	Not selected (0) or Selected (1)
1	Unexcluded (0) or Excluded (1)
2	Unhidden (0) or Hidden (1)
3	Unlabeled (0) or Labeled (1)
4-7	Marker
8-14	Color

Combine States

Generates a row state combination with two or more arguments. Use the insert button ( ) on the Formula Editor keypad or type a comma to add arguments to the **Combine States** function. The currently selected expression becomes the first argument when you select **Combine States**. **Replace** each argument with an expression that evaluates to a row state. This formula:

```
Combine States(
  Selected State( Modulo( Row(), 2 ) ),
  Labeled State( Modulo( Row() + 1, 2 ) ) );
```

alternately labels or selects each row in the calculated row state column. The **Selected State** and **Labeled State** functions are defined later in this section. Use the insert ( ) and delete ( )buttons on the Formula Editor keypad to add more arguments or remove unwanted arguments.

If you include conflicting row states in a combination, the results are unpredictable.

Excluded State

Interprets a numeric argument as true or false. When an argument evaluates as true, the **Excluded State** function assigns the excluded condition as the value of the column for that row. For example, **Excluded State(Modulo(Row(),2))** assigns the excluded row state as the value of the row state column for each odd numbered row.

Hidden State

Assigns the hidden row state condition when its argument is greater than zero. If the argument is zero, the value in the column for that row is not hidden.

Labeled State

Gives the labeled row state condition when its argument is greater than zero. If the argument is zero the row value in the column for that row is not labeled.

Color State

Returns the color from the JMP color map that corresponds to its integer argument. JMP colors are numbered 0 through 84. Zero maps to black.

Marker State

Returns markers from the JMP marker map that correspond to its integer argument. JMP markers are numbered 0 through 16. The formula **Marker State(Row())** assigns all the row state markers in a repeating sequence determined by the current row number to the calculated row state column. A row state column can have multiple row states as a value.

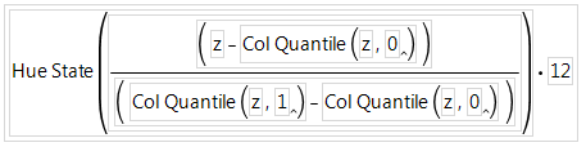
Selected State

Gives the selected row state condition when its argument is greater than zero. If the argument is zero, the value in the column for that row is not selected.

Hue State

Returns the color from the JMP hue map that corresponds to its integer argument. JMP hues are numbered 0 through 11 but larger integers are treated as *modulo* 12. The **Hue State** function does not map to black, gray, or white. A hue of zero maps to red, and a hue of 11 maps to magenta. The formula in Figure A.16 assigns row state colors in a chromatic spread based on the value of *z*. The **Hue State** function is used with a row state data type column.

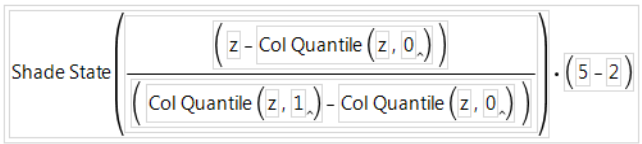
Figure A.16 Example of Hue State Function



Shade State

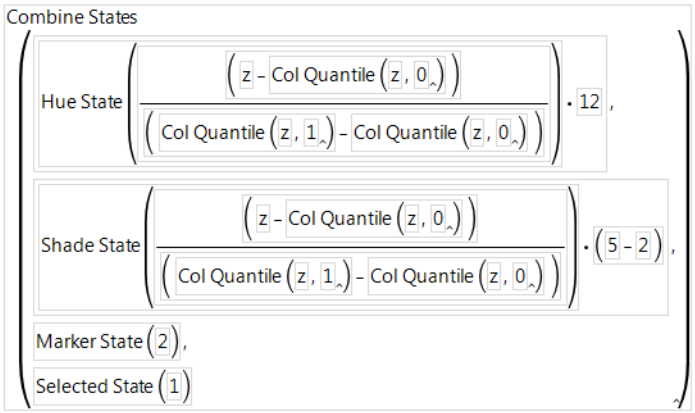
Assigns five shade levels to a color or hue. A shade of –2 is darkest and shade of +2 is lightest. A shade of zero is a pure color. The formula in Figure A.17 assigns shade values based on the value of *z*.

Figure A.17 Example of Shade State Function



To assign all shades of all the colors in the colors palette, you need to use the **Hue State** and **Shade State** assignments together. The formula in Figure A.18 uses the **Combine States** function described at the beginning of this section. The first argument in the **Combine States** function is the **Hue State** formula shown previously, and the second argument is the **Shade State** formula. In addition, the **Marker State** function with an argument of 2 assigns the **X** marker to each row, and the **Selected State** function with an argument of 1 selects each row.

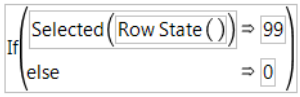
Figure A.18 Combine States Example For Using Both Hue State and Row State



Excluded, Hidden, Labeled, and Selected

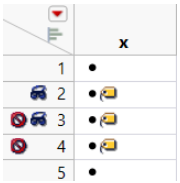
Accepts a row state expression argument (row state column or row state constant) that evaluates as either 1 or 0 (true or false). These characteristics are inactive by default. Often, the **Row()** function is the argument, which detects the active row state condition of each row. For example, in Figure A.19, the formula assigns 99 whenever a row is actively selected, and 0 otherwise. Note that this formula is used in a column that has a numeric data type.

Figure A.19 Example of a Formula Using the **Selected** Function



The example in Figure A.20 assigns row state conditions to a row state column. The formula for the row state column (in the column called x) checks to determine whether the active row state is either **Hidden** or **Excluded**, and if so, assigns the **Labeled** row state.

Figure A.20 Calculate Row State Information in a Row State Column



Color Of

Accepts any row state expression or column, or the **Row State()** function as its argument. Returns a number from the JMP color map that corresponds to the active color state, or zero if there is no assigned color.

Marker Of

Accepts any row state expression or column, or the **Row State()** function as its argument. Returns a number from the JMP marker map that corresponds to the active marker or zero if there is no assigned marker.

Assignment Functions

Assignment functions work in place. That is, the result returned by the operation (on the right of the operator) is stored in the argument on the left of the operator and replaces its current value.

Assignment statements are most often used in conjunction with other commands to build a JSL script. You can use the Formula Editor to create and execute a script in that column, but this is not recommended because of dependencies and ambiguities that can result. Most often, scripts are stored as .jsl files, and can be saved with a data table. See "Create and Save Scripts in Data Tables". For more information about syntax, see the *JSL Syntax Reference*.

Note: The first argument of an **assignment** function must be capable of being assigned. This means you cannot have an assignment such as 3+=4, because 3 is a constant value that cannot be reassigned. You must first create a variable (a table variable or local variable) whose value is 3. (For more information about table variables, see "Use Data Table Variables". For more information about local variables, see "Refer to Data Table Values in Formulas"). Then use that variable as the left-hand argument of the assignment function.

= (assign) Puts the value of *b* into *a*. For example (*a*=*b*).

+= (add to) Adds the value of *b* to *a* and puts the result back into *a*. For example, *a*+=*b*.

-= (subtract to) Subtracts the value of *b* and puts the result back into *a*. For example, *a*–=*b*.

***= (multiply to)** Multiplies *b* with *a* and puts the result back into *a*. For example, *a**=*b*.

/= (divide to) Divides *b* into *a* and puts the result back into *a*. For example, *a*/=*b*.

++ (post increment) Adds one (1) to *a*, in place, so that *a*++. For example, if the initial value of *a* is 4, the expression *a*++ changes a to 5.

-- (post decrement) Subtracts one (1) from *a*, in place, so that *a*– – . For example, if the initial value of *a* is 4, the expression *a*– – changes a to 3.

Parametric Model Functions

This category is a short cut to create three parametric models that are linear functions of set of window-selected columns.

Linear Model, Interactions Model, Full Quadratic Model

Selecting each of these opens a column selection box that lets you select one or more columns to be included in the model. The function then creates and populates the chosen model.

Finance Functions

Lets you create formulas to calculate principal payments, interest rate, rate of return, and so on.

Double Declining Balance

Returns the depreciation of an asset for a specified period of time. The function uses the double-declining balance method or some other depreciation factor.

Future Value

Returns the future value of an investment that is based on periodic, constant payments and a constant interest rate.

Interest Payment

Returns the interest payment for a given period for an investment that is based on periodic, constant payments and a constant interest rate.

Interest Rate

Returns the interest rate per period of an annuity.

Internal Rate of Return

Returns the internal rate of return for a series of cash flows in the *values* argument.

Modified Internal Rate of Return

Returns the modified internal rate of return for a series of periodic cash flows. The cost of investment and the interest received on reinvested cash is included.

Net Present Value

Returns the net present value of an investment by using a discount rate and a series of future payments (negative values) and income (positive values).

Number of Periods

Returns the number of periods for an investment that is based on periodic, constant payments and a constant interest rate.

Payment

Returns the payment for a loan that is based on constant payments and a constant interest rate.

Present Value

Returns the present value of an investment.

Principal Payment

Returns the payment on the principal for a given period for an investment that is based on periodic, constant payments and a constant interest rate.

Straight Line Depreciation

Returns the straight-line depreciation of an asset for one period.

Sum Of Years Digits Depreciation

Returns the sum-of-years' digits depreciation of an asset for a specified period.

Appendix B

JMP Technical Details

This chapter contains technical details that apply to multiple platforms in JMP and to SAS integration.

Contents

JMP Features That Support Multithreading 789
Conventions for Mapping JMP Attributes to SAS Extended Attributes 790

JMP Features That Support Multithreading

Some features in JMP are coded to take advantage of multiple central processing units (CPUs) on a machine, allowing these features to run significantly faster. This process is called multithreading.

The following features support multithreading:

- JMP PRO Boosted Trees
- JMP PRO Bootstrap Forest
- Choice
- JMP PRO Covering Arrays
- Capability Analysis in Distribution
- Explore Outliers
- Explore Patterns
- Factor Analysis
- Fit Life by X
- Fit Model: Parametric Survival, JMP PRO Mixed Model, JMP PRO Generalized Regression, and JMP PRO Response Screening
- Functional Data Explorer
- Gaussian (JMP PRO Fast GASP and the use of Categorical variables are only in JMP Pro.)
- JMP PRO K Nearest Neighbors
- Latent Class Analysis
- Life Distribution
- Multivariate
- Neural (JMP PRO Some features are only in JMP Pro.)
- Nominal Logistic
- Nonlinear and Nonlinear Curve
- Normal Mixtures in Cluster Analysis
- Partial Least Squares (JMP PRO Some features are only in JMP Pro.)
- Partition
- Predictor Screening
- Principal Components
- Process Screening
- Profiler (Does not apply to Profilers launched from within other platforms.)

- Reliability Forecast
- JMP PRO Repairable Systems Simulation
- Response Screening
- JMP PRO Sparse Principal Components
- JMP PRO Structural Equation Models
- JMP PRO Support Vector Machines
- SVD Imputation in the Missing Values Screening
- Text Explorer (JMP PRO Some features are only in JMP Pro.)
- Time Series Forecast
- JMP PRO Uplift
- Variable Clustering

Conventions for Mapping JMP Attributes to SAS Extended Attributes

SAS extended attributes are metadata that you define in SAS code to import information such as table scripts, labels, length, and type. You associate the extended attributes with a data set or variable and define them in name-value pairs, such as `_JMP_TABLESCRIPTNAME_2="OnOpen"`.

This section provides information about SAS extended attributes and their corresponding JMP attributes.

Table B.1 Table Attributes

JMP Table Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Table Variable Count	_JMP_TABLEVARCOUNT	Numeric, 0	Count of table variables
Table Variable Name 1	_JMP_TABLEVARNAME_1	String	Name of the first table variable
Table Variable Value 1	_JMP_TABLEVARVALUE_1	String	Value of the first table variable

Table B.1 Table Attributes *(Continued)*

JMP Table Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Table Variable Name 2	_JMP_TABLEVARNAME_2	String	Name of the second table variable
Table Variable Value 2	_JMP_TABLEVARVALUE_2	String	Value of the second table variable
Table Variable Name n	_JMP_TABLEVARNAME_*n*	String	Name of the *n*th table variable
Table Variable Value n	_JMP_TABLEVARVALUE_*n*	String	Value of the *n*th table variable
Table Script Count	_JMP_TABLESCRIPTCOUNT	Numeric	Count of table scripts
Table Script Name 1	_JMP_TABLESCRIPTNAME_1	String	Name of the first table scripts
Table Script Value 1	_JMP_TABLESCRIPTVALUE_1	String	Value of the first table scripts
Table Script Name 2	_JMP_TABLESCRIPTNAME_2	String	Name of the second table script
Table Script Value 2	_JMP_TABLESCRIPTVALUE_2	String	Value of the second table script
Table Script Name n	_JMP_TABLESCRIPTNAME_*n*	String	Name of the table script *n*
Table Script Value n	_JMP_TABLESCRIPTVALUE_*n*	String	Value of table script *n*
Lock Data Table	_JMP_ISLOCKED	Boolean, 0	Locked table
Suppress Formula Eval	_JMP_SUPPRESSEVAL	Boolean, 0	Suppressed formula evaluation
Column Group Count	_JMP_COLGRPCOUNT	Numeric, 0	Number of column groups

Table B.1 Table Attributes *(Continued)*

JMP Table Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Column Group Name 1	_JMP_COLGRPNAME_1	String	Name of column group 1
Column Group Start Col 1	_JMP_COLGRPSTARTCOL_1	String	Name of first column in group 1
Column Group NCols 1	_JMP_COLGRPNCOLS_1	Numeric	Number of columns in group 1
Column Group Name 2	_JMP_COLGRPNAME_2	String	Name of column group 2
Column Group Start Col 2	_JMP_COLGRPSTARTCOL_2	String	Name of first column in group 2
Column Group NCols 2	_JMP_COLGRPNCOLS_2	Numeric	Number of columns in group 2
Column Group Name n	_JMP_COLGRPNAME_*n*	String	Name of column group *n*
Column Group Start Col n	_JMP_COLGRPSTARTCOL_*n*	String	Name of first column in group *n*
Column Group NCols n	_JMP_COLGRPNCOLS_*n*	Numeric	Number of columns in group *n*

Table B.2 Column Attributes

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Column Name	_JMP_COLNAME	String	Original column name in JMP

Table B.2 Column Attributes *(Continued)*

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Data Length	_JMP_DATALENGTH	Numeric, 0	Length of the data. For numeric types, the value can be 1, 2, 4, or 8. For character types, 0 (or negative) means variable length, positive values mean fixed-length.
Row State column	_JMP_ISROWSTATE	Numeric, 0	0 = not row state, 1 = default row state for table, 2 = row state column
Label	_JMP_ISLABEL	Boolean, 0	Label column. 1 for yes. 0 for no.
Scroll Lock	_JMP_ISSCROLLLOCK	Boolean, 0	Scroll locked column. 1 for yes. 0 for no.
Hidden	_JMP_ISHIDDEN	Boolean, 0	Hidden column. 1 for yes. 0 for no
Excluded	_JMP_ISEXCLUDED	Boolean, 0	Excluded column. 1 for yes. 0 for no
Notes	_SAS_NOTES	String	Notes about the column
Modeling Type	_SAS_LEVEL	String	Valid values are Binary, Interval, Ordinal, Nominal, Unary. JMP maps Binary and Unary to Nominal and Interval to Continuous.

Table B.2 Column Attributes *(Continued)*

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Preselect Role	_SAS_MININGROLE	String	Valid values are Assessment, Censor, Classification, Cost, Cross ID, Decision, Frequency, ID, Input, Key, Label, Prediction, Referrer, Rejected, Residual, Segment, Sequence, Target, Text, Text Location, Time ID, Treatment, Web Address, Weight. JMP maps Y to Target, X to Input, Freq to Frequency, and Weight to Weight.
Distribution	_SAS_DISTRIBUTION	String	Distribution type to fit for this column
Format Name	_JMP_FORMATNAME	String	Format of the column (such as Fixed Dec or Percent)
Format Width	_JMP_FORMATWIDTH	Numeric	Width of the column
Format Decimals	_JMP_FORMATDECIMALS	Numeric	Number of decimal places
Format Code	_JMP_FORMATCODE	String	Typically the currency code for currency formats
Use Thousands Separator	_JMP_USETHOUSANDSSEP	Boolean, 0	Include thousands separator. 1 for yes. 0 for no.
Input Format	_JMP_INFORMAT	String	Input format (for date, time, or duration-formatted columns)
Formula	_JMP_FORMULA	String	Column formula

Table B.2 Column Attributes *(Continued)*

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Suppress Formula Eval	_JMP_SUPPRESSEVAL	Boolean, 0	Suppressed formula evaluation. 1 for yes. 0 for no.
Lock	_JMP_ISLOCKED	Boolean, 0	Locked column. 1 for yes. 0 for no.
Range Check Code	_JMP_RANGECHECK_CODE	String	A code that identifies the type of range check. Examples are LE, LT, LTLT, !LT, and !LELE.
Range Check Low	_JMP_RANGECHECK_LOW	Numeric	First argument to range check
Range Check High	_JMP_RANGECHECK_HIGH	Numeric	Second optional argument to range check
List Check	_JMP_LISTCHECK	String	List check
Missing Value Codes	_JMP_MISSINGCODES	String	Values to be treated as missing
Value Label Count	_JMP_VALUELABEL_COUNT	Numeric	Count of value labels
Value Label Code	_JMP_VALUELABELCODE_1	String	Code for the value label (such as EQ, LT, LE, LTLT, and LELT.). Always EQ for character variables.
Value Label Lower Bound	_JMP_VALUELABELLOW_1	Varies	The lower bound of a numeric range, or the only bound if the ranges is unbounded on one side. A string for character columns. Numeric for numeric columns.

Table B.2 Column Attributes *(Continued)*

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Value Label Upper Bound	_JMP_VALUELABELHIGH_1	Numeric	The upper bound of a numeric range. Not used for character value labels.
Value Label Display	_JMP_VALUELABELDISPLAY_1	String	The display string for this value label
Use Value Labels	_JMP_USEVALUELABELS	Boolean, 0	Uses value labels when displaying this column. 1 for yes. 0 for no.
Value Scores	_JMP_VALUESCORES	String	Associates data values with numerical scores
Value Order	_JMP_VALUEORDER	String	The order of values for the column
Value Colors	_JMP_VALUECOLORS	String	Maps values to colors.
Color Gradient	_JMP_COLORGRADIENT	String	Maps values to a color gradient.
Color Cells	_JMP_COLORCELLS	Boolean, 0	Color cells of this column using the specified value colors or color gradient. 1 for yes. 0 for no.
Axis	_JMP_AXIS	String	Axis definition
Coding	_JMP_CODING	String	Code a range of values to -1, 1 for modeling purposes
Mixture	_JMP_MIXTURE	String	Participation in a mixture of columns adding up to a value. The Mixture column property has numeric values (limits) and string values (L Pseudo Component and U Pseudo Component).

Table B.2 Column Attributes *(Continued)*

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Row Order Levels	_JMP_ROWORDER	Boolean, 0	Sort the column by occurrence rather than value. 1 for yes. 0 for no.
Spec Limits	_JMP_SPECLIMITS	String	Target value and upper and lower spec limits
Control Limits	_JMP_CONTROLLIMITS	String	Control limits for control chart type
Response Limits	_JMP_RESPONSELIMITS	String	Bounds on response's acceptable range, used by profilers
Design Role	_JMP_DESIGNROLE	String	Role in a designed experiment
Factor Changes	_JMP_FACTORCHANGES	String	Difficulty of changing a factor, for DOE. Value values: Easy, Hard, Very Hard
Sigma	_JMP_SIGMA	String	Sigma for Control Charts
Units	_JMP_UNITS	String	Units for this variable; can be any string
Time Frequency	_JMP_TIMEFREQ	String	Numeric, Annual, Quarterly, Monthly, Weekly, Daily, Hourly, By Minutes, By Seconds
Map Role	_JMP_MAPROLE	String	Role in a map data set
Multiple Response Separator	_JMP_MRSEP	String, ","	Separator for a multiple-response column.
Label Column	_JMP_LABELCOLUMN	Boolean, 0	Is this column a label column? 1 for yes. 0 for no.

Table B.2 Column Attributes *(Continued)*

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Expression Column	_JMP_ISEXPRESSIONCOLUMN	Boolean, 0	Is this column an expression column? 1 for yes. 0 for no.

Table B.3 Custom Column Properties

JMP Column Attribute	SAS Attribute Name	SAS Attribute Type and Default Value	Description
Custom Property Count	_JMP_CUSTCOLPROPCOUNT	Numeric, 0	Count of custom column property
Custom Property Name 1	_JMP_CUSTCOLPROPNAME_1	String	Name of the custom column property 1
Custom Property Value 1	_JMP_CUSTCOLPROPVALUE_1	String	Value of the custom column property 1
Custom Property Name 2	_JMP_CUSTCOLPROPNAME_2	String	Name of the custom column property 2
Custom Property Value 2	_JMP_CUSTCOLPROPVALUE_2	String	Value of the custom column property 2
Custom Property Name n	_JMP_CUSTCOLPROPNAME_*n*	String	Name of the custom column property *n*
Custom Property Value n	_JMP_CUSTCOLPROPVALUE_*n*	String	Value of the custom column property *n*

Appendix C

References

Gander, W., and Gautschi, W. (2000). "Adaptive Quadrature–Revisited." *BIT Numerical Mathematics* 40:84–101.

Marsaglia, G. (1995). *DIEHARD: A Battery of Tests of Randomness*.

Matsumoto, M., and Nishimura, T. (1998). "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-random Number Generator." *ACM Transactions on Modeling and Computer Simulation* 8:3–30.

Appendix D

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
- You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS”, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Basic Analysis

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Basic Analysis*. Cary, NC: SAS Institute Inc.

JMP® 16 Basic Analysis

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Basic Analysis

1 Learn about JMP 15
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 17
JMP Help 18
JMP Documentation Library 18
Additional Resources for Learning JMP 24
JMP Tutorials 24
Sample Data Tables 24
Learn about Statistical and JSL Terms 25
Learn JMP Tips and Tricks 25
JMP Tooltips 25
JMP User Community 26
Free Online Statistical Thinking Course 26
JMP New User Welcome Kit 26
Statistics Knowledge Portal 26
JMP Training 26
JMP Books by Users 27
The JMP Starter Window 27
JMP Technical Support 27

2 Introduction to Basic Analysis 29
Overview of Fundamental Analysis Methods

3 Distributions 31
Using the Distribution Platform
Overview of the Distribution Platform 34
Example of the Distribution Platform 35
Launch the Distribution Platform 36
The Distribution Report 38
Histograms 39
The Frequencies Report 42
The Quantiles Report 43
The Summary Statistics Report 43

Distribution Platform Options 46
Options for Categorical Variables 47
Display Options for Categorical Variables 47
Histogram Options for Categorical Variables 47
Save Options for Categorical Variables 48
Options for Continuous Variables 48
Display Options for Continuous Variables 49
Histogram Options for Continuous Variables 50
Normal Quantile Plot 51
Outlier Box Plot 52
Quantile Box Plot 53
Stem and Leaf 54
CDF Plot 55
Test Mean 56
Test Std Dev 57
Test Equivalence 57
Confidence Intervals 58
Prediction Intervals 59
Tolerance Intervals 59
Process Capability 60
Fit Distributions 62
Save Options for Continuous Variables 67
Additional Examples of the Distribution Platform 69
Example of Selecting Data in Multiple Histograms 69
Example Using a By Variable 70
Examples of the Test Probabilities Option 71
Example of Prediction Intervals 73
Example of Tolerance Intervals 74
Example of Process Capability 76
Statistical Details for the Distribution Platform 77
Standard Error Bars 77
Statistical Details for Quantiles 77
Summary Statistics 78
Normal Quantile Plot 79
Wilcoxon Signed Rank Test 80
Standard Deviation Test 82
Normal Quantiles 82
Saving Standardized Data 82
Prediction Intervals 83
Tolerance Intervals 83

Continuous Fit Distributions ... 86
Discrete Fit Distributions ... 92
Details for the Legacy Distribution Fitters ... 96
Fit Distributions Options (Legacy) ... 96
Statistical Details for Continuous Fit Distributions (Legacy) ... 101
Statistical Details for Discrete Fit Distributions (Legacy) ... 106
Statistical Details for Fitted Quantiles (Legacy) ... 108
Statistical Details for Fit Distribution Options (Legacy) ... 108

4 Introduction to Fit Y by X ... 111
Examine Relationships between Two Variables

Overview of the Fit Y by X Platform ... 113
Launch the Fit Y by X Platform ... 113
Launch Specific Analyses from the JMP Starter Window ... 114

5 Bivariate Analysis ... 115
Examine Relationships between Two Continuous Variables

Example of Bivariate Analysis ... 118
Launch the Bivariate Platform ... 118
The Bivariate Plot ... 119
Fitting Options ... 120
Fitting Options ... 121
Fitting Option Categories ... 123
Fit the Same Option Multiple Times ... 123
Histogram Borders ... 124
Fit Mean ... 125
Fit Mean Report ... 125
Fit Line and Fit Polynomial ... 126
Linear Fit and Polynomial Fit Reports ... 126
Fit Special ... 132
Fit Special Reports and Menus ... 133
Flexible ... 134
Fit Spline ... 134
Kernel Smoother ... 135
Fit Each Value ... 136
Fit Orthogonal ... 137
Orthogonal Fit Ratio Report ... 137
Robust ... 138
Fit Robust ... 138
Fit Cauchy ... 139
Density Ellipse ... 139

Bivariate Normal Ellipse Report . . . 140
Nonpar Density . . . 141
Quantile Density Contours Report . . . 141
Group By . . . 142
Fitting Menus . . . 142
Fitting Menu Options . . . 143
Diagnostics Plots . . . 146
Additional Examples of the Bivariate Platform . . . 147
Example of the Fit Special Option . . . 147
Example of the Fit Orthogonal Option . . . 150
Example of the Fit Robust Command . . . 151
Example of Group By Using Density Ellipses . . . 153
Example of Group By Using Regression Lines . . . 154
Example of Grouping Using a By Variable . . . 155
Statistical Details for the Bivariate Platform . . . 157
Fit Line . . . 157
Fit Spline . . . 157
Fit Orthogonal . . . 157
Summary of Fit Report . . . 158
Lack of Fit Report . . . 159
Parameter Estimates Report . . . 160
Smoothing Fit Reports . . . 160
Bivariate Normal Ellipse Report . . . 160

6 Oneway Analysis . . . 161
Examine Relationships between a Continuous Y and a Categorical X Variable

Overview of Oneway Analysis . . . 164
Example of Oneway Analysis . . . 164
Launch the Oneway Platform . . . 166
Data Format . . . 167
The Oneway Plot . . . 167
Oneway Platform Options . . . 168
Display Options . . . 172
Quantiles . . . 173
Outlier Box Plots . . . 173
Means/Anova and Means/Anova/Pooled t . . . 174
The Summary of Fit Report . . . 175
The t Test Report . . . 175
The Analysis of Variance Report . . . 176
The Means for Oneway Anova Report . . . 177

The Block Means Report 178
Mean Diamonds and X-Axis Proportional 178
Mean Lines, Error Bars, and Standard Deviation Lines 179
Analysis of Means Methods 179
Analysis of Means for Location 180
Analysis of Means for Scale 181
Analysis of Means Charts 182
Analysis of Means Options 182
Compare Means 183
Using Comparison Circles 185
Each Pair, Student's t 187
All Pairs, Tukey HSD 187
With Best, Hsu MCB 187
With Control, Dunnett's 188
Each Pair Stepwise, Newman-Keuls 189
Compare Means Options 190
Nonparametric Tests 191
The Wilcoxon, Median, Van der Waerden, and Friedman Rank Test Reports 192
Kolmogorov-Smirnov Two-Sample Test Report 193
Nonparametric Multiple Comparisons 195
Unequal Variances 197
Tests That the Variances Are Equal Report 198
Equivalence Tests 200
Equivalence Tests Report 200
Robust 202
Robust Fit 202
Cauchy Fit 202
Power 203
Power Details Window and Reports 203
Normal Quantile Plot 204
CDF Plot 205
Densities 205
Matching Column 205
Additional Examples of the Oneway Platform 207
Example of an Analysis of Means Chart 207
Example of an Analysis of Means for Variances Chart 208
Example of the Each Pair, Student's t Test 209
Example of the All Pairs, Tukey HSD Test 211
Example of the With Best, Hsu MCB Test 213
Example of the With Control, Dunnett's Test 214

Example of the Each Pair Stepwise, Newman-Keuls Test ... 216
Example Contrasting Four Compare Means Tests ... 216
Example of the Nonparametric Wilcoxon Test ... 217
Example of the Unequal Variances Option ... 220
Example of an Equivalence Test ... 221
Example of the Robust Fit Option ... 223
Example of the Power Option ... 224
Example of a Normal Quantile Plot ... 226
Example of a CDF Plot ... 227
Example of the Densities Options ... 228
Example of the Matching Column Option ... 229
Example of Stacking Data for a Oneway Analysis ... 231
Statistical Details for the Oneway Platform ... 237
Comparison Circles ... 237
Power ... 239
Summary of Fit Report ... 239
Tests That the Variances Are Equal ... 240
Nonparametric Test Statistics ... 241

7 Contingency Analysis ... 245
Examine Relationships between Two Categorical Variables

Example of Contingency Analysis ... 247
Launch the Contingency Platform ... 248
Data Format ... 249
The Contingency Report ... 250
Contingency Platform Options ... 251
Mosaic Plot ... 253
Pop-Up Menu ... 254
Contingency Table ... 255
Description of the Contingency Table ... 256
Tests ... 257
Description of the Tests Report ... 257
Fisher's Exact Test ... 258
Analysis of Means for Proportions ... 258
Correspondence Analysis ... 259
Understanding Correspondence Analysis Plots ... 259
Correspondence Analysis Options ... 259
The Details Report ... 259
Cochran-Mantel-Haenszel Test ... 260
Agreement Statistic ... 261

Relative Risk 261
Two Sample Test for Proportions 262
Measures of Association 262
Cochran Armitage Trend Test 264
Exact Test 264
Additional Examples of the Contingency Platform 265
Example of Analysis of Means for Proportions 265
Example of Correspondence Analysis 266
Example of a Cochran Mantel Haenszel Test 268
Example of the Agreement Statistic Option 269
Example of the Relative Risk Option 270
Example of a Two Sample Test for Proportions 272
Example of the Measures of Association Option 273
Example of the Cochran Armitage Trend Test 274
Statistical Details for the Contingency Platform 275
Agreement Statistic Option 275
Odds Ratio Option 276
Tests Report 276
Details Report in Correspondence Analysis 277

8 Logistic Analysis 279
Examine Relationships between a Categorical Y and a Continuous X Variable

Overview of Logistic Regression 281
Nominal Logistic Regression 281
Ordinal Logistic Regression 281
Example of Nominal Logistic Regression 282
Launch the Logistic Platform 283
Data Structure 284
The Logistic Report 285
Logistic Plot 286
Iterations 286
Whole Model Test 286
Fit Details 288
Parameter Estimates 289
Logistic Platform Options 289
ROC Curves 290
Save Probability Formula 291
Inverse Prediction 292
Additional Examples of Logistic Regression 292
Example of Ordinal Logistic Regression 292

Additional Example of a Logistic Plot 294
Example of ROC Curves 296
Example of Inverse Prediction Using the Crosshair Tool 297
Example of Inverse Prediction Using the Inverse Prediction Option 298
Statistical Details for the Logistic Platform 300

9 Tabulate 301
Create Summary Tables Interactively

Example of the Tabulate Platform 303
Launch the Tabulate Platform 308
Use the Dialog 310
Add Statistics 311
The Tabulate Output 314
Analysis Columns 315
Grouping Columns 315
Column and Row Tables 316
Edit Tables 317
Tabulate Platform Options 318
Show Test Build Panel 319
Right-Click Menu for Columns 319
Additional Examples of the Tabulate Platform 320
Example of Creating Different Tables and Rearranging Contents 320
Example of Combining Columns into a Single Table 324
Example Using a Page Column 326

10 Simulate 329
Answer Challenging Questions with Parametric Resampling

Overview of the Simulate Platform 331
Examples That Use Simulate 332
Construct Semiparametric Confidence Intervals for Variance Components 332
Conduct a Permutation Test 338
Explore Retaining a Factor in Generalized Regression 340
Conduct Prospective Power Analysis for a Nonlinear Model 345
Launch the Simulate Window 354
The Simulate Window 354
The Simulate Results Table 355
Simulation Results Report 356
Simulated Power Report 356

11 Bootstrapping 357
Approximate the Distribution of a Statistic through Resampling

Overview of Bootstrapping 359
Platforms that Support Bootstrapping 360
Example of Bootstrapping 361
Bootstrapping Window Options 363
Stacked Bootstrap Results Table 365
Unstacked Bootstrap Results Table 366
Analysis of Bootstrap Results 367
Additional Example of Bootstrapping 368
Statistical Details for Bootstrapping 373
Calculation of Fractional Weights 373
Bias-Corrected Percentile Intervals 373

12 Text Explorer 375
Explore Unstructured Text in Your Data

Overview of the Text Explorer Platform 377
Text Processing Steps 379
Example of the Text Explorer Platform 380
Launch the Text Explorer Platform 383
Customize Regex in the Regular Expression Editor 385
The Text Explorer Report 391
Summary Counts Report 391
Term and Phrase Lists 392
Text Explorer Platform Options 396
Text Preparation Options 396
Text Analysis Options 403
Save Options 404
Report Options 406
Latent Class Analysis 407
Latent Semantic Analysis (SVD) 408
SVD Report 409
SVD Report Options 410
Topic Analysis 413
Topic Analysis Report 413
Topic Analysis Report Options 414
Discriminant Analysis 415
Discriminant Analysis Report 416
Discriminant Analysis Report Options 416
Term Selection 417

Term Selection Settings 417
Term Selection Report 418
Term Selection Report Options 420
Sentiment Analysis 421
Sentiment Analysis Report 422
Sentiment Analysis Report Options 424
Additional Example of the Text Explorer Platform 426

A References 431

B Technology License Notices 435

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 17
JMP Help . . . 18
JMP Documentation Library . . . 18
Additional Resources for Learning JMP . . . 24
JMP Tutorials . . . 24
Sample Data Tables . . . 24
Learn about Statistical and JSL Terms . . . 25
Learn JMP Tips and Tricks . . . 25
JMP Tooltips . . . 25
JMP User Community . . . 26
Free Online Statistical Thinking Course . . . 26
JMP New User Welcome Kit . . . 26
Statistics Knowledge Portal . . . 26
JMP Training . . . 26
JMP Books by Users . . . 27
The JMP Starter Window . . . 27
JMP Technical Support . . . 27

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction to Basic Analysis

Overview of Fundamental Analysis Methods

Basic Analysis describes the initial types of analyses that you often perform in JMP:

- The Distribution platform illustrates the distribution of a single variable using histograms, additional graphs, and reports. Once you know how your data is distributed, you can plan the appropriate type of analysis going forward. See "Distributions".
- The Fit Y by X platform analyzes the pair of X and Y variables that you specify, by context, based on modeling type. See "Introduction to Fit Y by X". The four types of analyses include:
  - The Bivariate platform, which analyzes the relationship between two continuous X variables. See "Bivariate Analysis".
  - The Oneway platform, which analyzes how the distribution of a continuous Y variable differs across groups defined by a categorical X variable. See "Oneway Analysis".
  - The Contingency platform, which analyzes the distribution of a categorical response variable Y as conditioned by the values of a categorical X factor. See "Contingency Analysis".
  - The Logistic platform, which fits the probabilities for response categories (Y) to a continuous X predictor. See "Logistic Analysis".
- The Tabulate platform interactively constructs tables of descriptive statistics. See "Tabulate".
- The Simulate feature provides parametric and nonparametric simulation capability. See "Simulate".
- Bootstrap analysis approximates the sampling distribution of a statistic. The data is re-sampled with replacement and the statistic is computed. This process is repeated to produce a distribution of values for the statistic. See "Bootstrapping".
- The Text Explorer platform enables you to categorize and analyze unformatted text data. You can use regular expressions to clean up the data before you proceed to analysis. See "Text Explorer".

Chapter 3

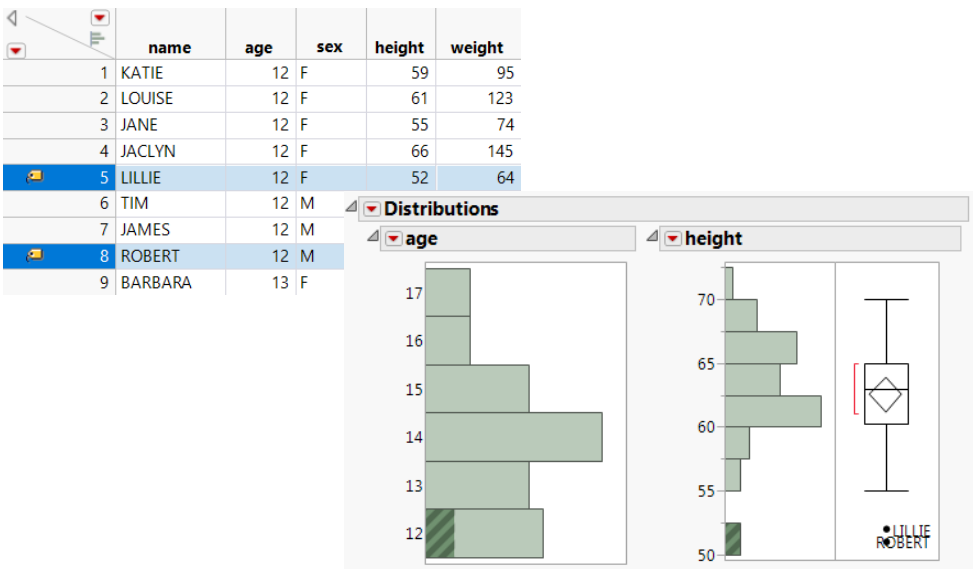
Distributions

Using the Distribution Platform

Use the Distribution platform to explore the distribution of a single variable using histograms, box plots, and summary statistics. You can perform several types of hypothesis tests, including *t* tests, *z* tests, Chi-squared tests, and equivalence tests. You can also construct confidence, tolerance, and prediction intervals, and estimate process capability. The word *univariate* simply means involving one variable instead of two (bivariate) or many (multivariate). However, you can examine the distribution of several individual variables within a report. The report content for each variable depends on whether the variable is categorical (nominal or ordinal) or continuous.

The Distribution report window is interactive. Clicking on a histogram bar highlights the corresponding data in any other histograms and in the data table.

Figure 3.1 Example of the Distribution Platform



Contents

Overview of the Distribution Platform 34
Example of the Distribution Platform 35
Launch the Distribution Platform 36
The Distribution Report 38
Histograms 39
The Frequencies Report 42
The Quantiles Report 43
The Summary Statistics Report 43
Distribution Platform Options 46
Options for Categorical Variables 47
Display Options for Categorical Variables 47
Histogram Options for Categorical Variables 47
Save Options for Categorical Variables 48
Options for Continuous Variables 48
Display Options for Continuous Variables 49
Histogram Options for Continuous Variables 50
Normal Quantile Plot 51
Outlier Box Plot 52
Quantile Box Plot 53
Stem and Leaf 54
CDF Plot 55
Test Mean 56
Test Std Dev 57
Test Equivalence 57
Confidence Intervals 58
Prediction Intervals 59
Tolerance Intervals 59
Process Capability 60
Fit Distributions 62
Save Options for Continuous Variables 67
Additional Examples of the Distribution Platform 69
Example of Selecting Data in Multiple Histograms 69
Example Using a By Variable 70
Examples of the Test Probabilities Option 71
Example of Prediction Intervals 73
Example of Tolerance Intervals 74
Example of Process Capability 76
Statistical Details for the Distribution Platform 77
Standard Error Bars 77

Statistical Details for Quantiles 77
Summary Statistics 78
Normal Quantile Plot 79
Wilcoxon Signed Rank Test 80
Standard Deviation Test 82
Normal Quantiles 82
Saving Standardized Data 82
Prediction Intervals 83
Tolerance Intervals 83
Continuous Fit Distributions 86
Discrete Fit Distributions 93
Details for the Legacy Distribution Fitters 96
Fit Distributions Options (Legacy) 96
Statistical Details for Continuous Fit Distributions (Legacy) 101
Statistical Details for Discrete Fit Distributions (Legacy) 107
Statistical Details for Fitted Quantiles (Legacy) 108
Statistical Details for Fit Distribution Options (Legacy) 108

Overview of the Distribution Platform

The treatment of variables in the Distribution platform is different, depending on the modeling type of the variable, which can be categorical (nominal or ordinal) or continuous.

Categorical Variables

For categorical variables, the initial graph that appears is a histogram. The histogram shows a bar for each level of the ordinal or nominal variable. You can also add a divided (mosaic) bar chart.

The Frequencies report show counts and proportions. You can add confidence intervals and test the probabilities from the options in the red triangle menu.

Continuous Variables

For numeric continuous variables, the initial graphs show a histogram and an outlier box plot. The histogram shows a bar for grouped values of the continuous variable. The following options are also available:

- normal quantile plot
- quantile box plot
- stem and leaf plot
- CDF plot

The reports show selected quantiles and summary statistics. Additional report options are available in the red triangle menu for the following:

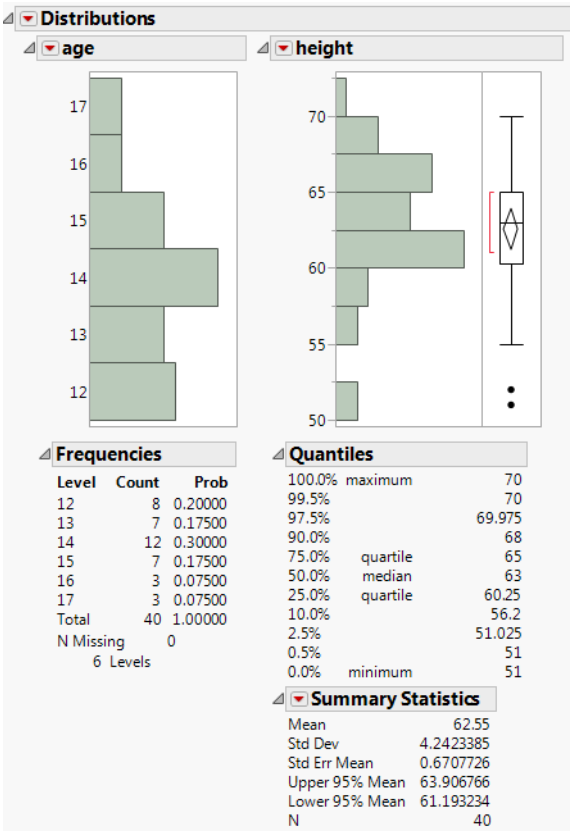
- saving ranks, probability scores, normal quantile values, and so on, as new columns in the data table (Save options)
- testing the mean and standard deviation of the column against a constant you specify (Test Mean and Test Std Dev options)
- fitting various distributions and nonparametric smoothing curves (Continuous Fit and Discrete Fit options)
- performing a process capability analysis for a quality control application
- confidence intervals, prediction intervals, and tolerance intervals

Example of the Distribution Platform

Learn how to create a distribution in JMP. Suppose that you have data on 40 students, and you want to see the distribution of age and height among the students.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Distribution**.
3. Select age and height and click **Y, Columns**.
4. Click **OK**.

Figure 3.2 Example of the Distribution Platform



From the histograms, you notice the following:

- The ages are not uniformly distributed.
- For height, there are two points with extreme values (that might be outliers).

Click the bar for 50 in the height histogram to take a closer look at the potential outliers.

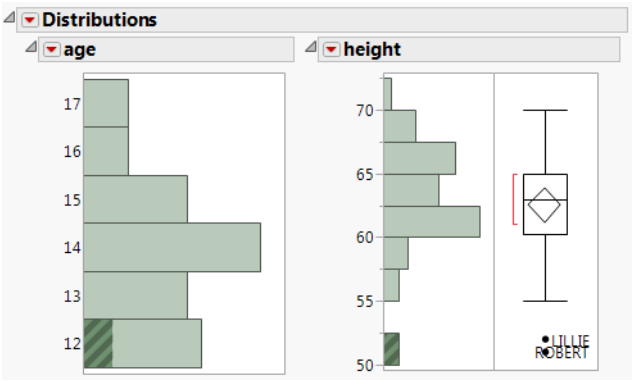
- The corresponding ages are highlighted in the age histogram. The potential outliers are age 12.
- The corresponding rows are highlighted in the data table. The names of the potential outliers are Lillie and Robert.

Add labels to the potential outliers in the height histogram.

1. Select both outliers.
2. Right-click one of the outliers and select **Row Label**.

   Label icons are added to these rows in the data table.
3. Resize the box plot wider to see the full labels.

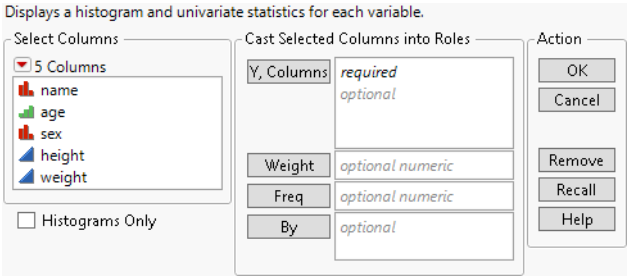
Figure 3.3 Potential Outliers Labeled



Launch the Distribution Platform

Launch the Distribution platform by selecting **Analyze > Distribution**.

Figure 3.4 The Distribution Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns Assigns the variables that you want to analyze. A histogram and associated reports appear for each variable.

Weight Assigns a variable that specifies weights for observations on continuous Ys. For categorical Ys, the Weight column is ignored. Any statistic that is based on the sum of the weights is affected by weights.

Freq Assigns a frequency variable to this role. This is useful if your data are summarized. In this instance, you have one column for the Y values and another column for the frequency of occurrence of the Y values. The sum of this variable is included in the overall count appearing in the Summary Statistics report (represented by N). All other moment statistics (mean, standard deviation, and so on) are also affected by the **Freq** variable.

By Produces a separate report for each level of the **By** variable. If more than one **By** variable is assigned, a separate report is produced for each possible combination of the levels of the **By** variables.

Create Process Capability (Appears only if a column contains a Spec Limits column property.) Adds a Process Capability report for the analysis columns that contain a Spec Limits column property.

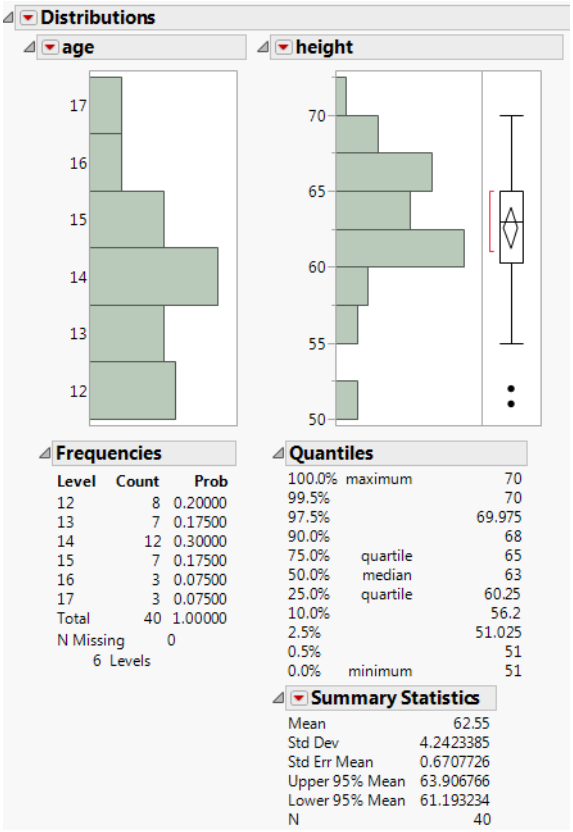
Histograms Only Removes everything except the histograms from the report window.

For more information about launch windows, see *Using JMP*.

The Distribution Report

The initial Distribution report contains a histogram and reports for each variable.

Figure 3.5 The Initial Distribution Report Window



Note: Follow the instructions in "Example of the Distribution Platform" to produce the report shown in Figure 3.5.

Red Triangle Menus

The red triangle menu next to Distributions contains options that affect all of the variables. See "Distribution Platform Options".

The red triangle menu next to each variable contains options that affect only that variable. See "Options for Categorical Variables" or "Options for Continuous Variables".

Tip: If you hold down the Ctrl key and select a variable option, the option applies to all of the variables in the report that have the same modeling type.

Histograms and Reports

Histograms visually display your data. See "Histograms".

The initial report for a categorical variable contains a Frequencies report. See "The Frequencies Report".

The initial report for a continuous variable contains a Quantiles and a Summary Statistics report. See "The Quantiles Report" and "The Summary Statistics Report".

Replace, Add, or Remove Variables in Histograms

To replace a variable in a histogram, from the Columns panel of the associated data table, drag and drop the variable into the axis of the histogram.

To create a new histogram in a report using a new variable, drag and drop the variable outside of an existing histogram. The new variable and histogram can be added before, between, or after the existing histograms.

To remove a variable from a histogram, select **Remove** from the red triangle menu.

Hidden and Excluded Rows

If you apply the Hidden row state to rows in the data table, the corresponding points do not appear in plots that show points. However, histograms are still constructed using the hidden rows.

If you want to exclude rows from the construction of the histograms and from analysis results, apply the Exclude row state. Then select **Redo > Redo Analysis** from the red triangle menu next to Distributions. Any rows that are excluded in the data table are also hidden in plots that show points.

Histograms

Histograms visually display your data. For categorical (nominal or ordinal) variables, the histogram shows a bar for each level of the ordinal or nominal variable. For continuous variables, the histogram shows a bar for grouped values of the continuous variable.

Highlighting data Click a histogram bar or an outlying point in the graph. The corresponding rows are highlighted in the data table, and corresponding sections of other histograms are also highlighted, if applicable. See "Highlight Bars and Select Rows".

Creating a subset Double-click a histogram bar, or right-click a histogram bar and select **Subset**. A new data table that contains only the selected data is created.

Resizing the entire histogram Hover over the histogram borders until you see a double-sided arrow. Then click and drag the borders.

Rescaling the axis Click and drag on an axis to rescale it.

Alternatively, hover over the axis until you see a hand. Then, double-click the axis and set the parameters in the Axis Settings window.

Resizing histogram bars (Available only for continuous variables.) There are multiple options to resize histogram bars. See "Resize Histogram Bars for Continuous Variables".

Specifying your selection Specify the data that you select in multiple histograms. See "Specify Your Selection in Multiple Histograms".

To see additional options for the histogram or the associated data table:

- Right-click a histogram. See *Using JMP*.
- Right-click an axis. You can add a label or modify the axis. See *Using JMP*.
- Click the red triangle next to the variable, and select **Histogram Options**. Options are slightly different depending on the variable modeling type. See "Options for Categorical Variables" or "Options for Continuous Variables".

Resize Histogram Bars for Continuous Variables

Resize histogram bars for continuous variables by using the following:

- the Grabber (hand) tool
- the **Set Bin Width** option
- the **Increment** option

Use the Grabber Tool

The Grabber (hand) tool is a quick way to explore your data.

1. Select **Tools > Grabber**.
2. Place the grabber tool anywhere in the histogram.
3. Click and drag the histogram bars.

Recall that histograms are based on binning the data. The height of each histogram bar is proportional to the number of observations that fall within the bin represented by the bar. For histograms in the default vertical orientation:

- Moving the hand to the left increases the bin width, which decreases the number of bins. The number of bars decreases as the bin width increases.

- Moving the hand to the right decreases the bin width, which increases the number of bins. The number of bars increases as the bin width decreases.
- Moving the hand up or down shifts the starting value for each bin, which shifts the data points that fall within each bin.

Note: If you have changed the histogram orientation to horizontal, reverse these directions. Move the hand down to increase bin width, up to decrease bin width, and left or right to shift bin starting values.

Use the Set Bin Width Option

The **Set Bin Width** option is a more precise way to set the width for all bars in a histogram. To use the Set Bin Width option, from the red triangle menu for the variable, select **Histogram Options > Set Bin Width**. Change the bin width value.

Use the Increment Option

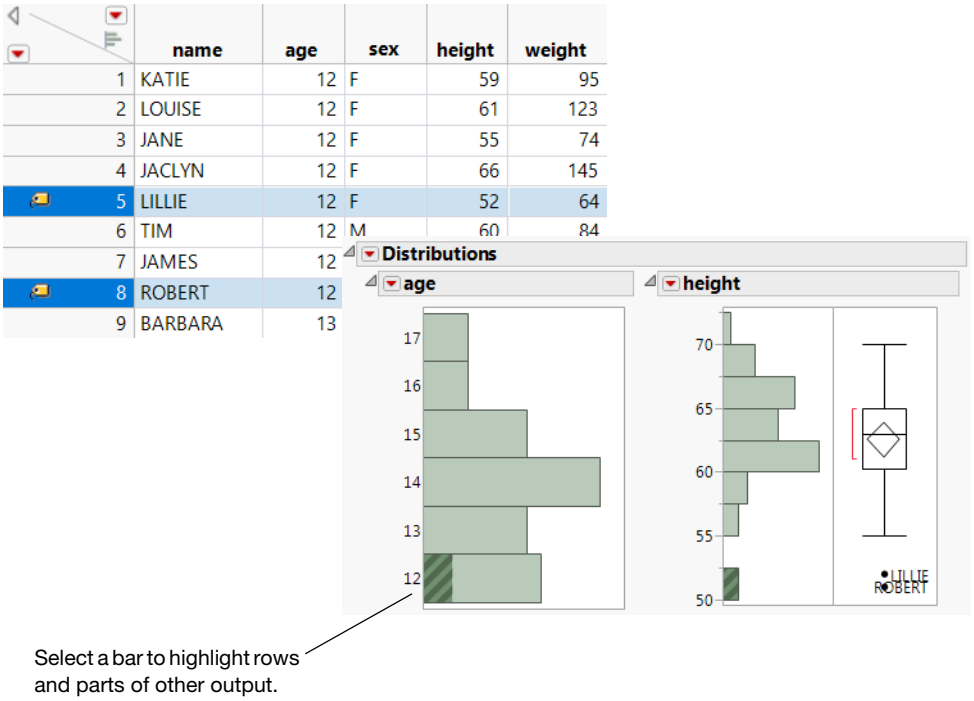
The **Increment** option is another precise way to set the bar width. To use the **Increment** option, double-click the axis, and change the Increment value.

Highlight Bars and Select Rows

Clicking on a histogram bar highlights the bar and selects the corresponding rows in the data table. The appropriate portions of all other graphical displays also highlight the selection. Figure 3.6 shows the results of highlighting a bar in the height histogram. The corresponding rows are selected in the data table.

Tip: To deselect specific histogram bars, press Ctrl and click the highlighted bars.

Figure 3.6 Highlighting Bars and Rows



Specify Your Selection in Multiple Histograms

Extend or narrow your selection in histograms:

- To extend your selection, hold down the Shift key and select another bar. This is the equivalent of using an *or* operator.
- To narrow your selection, hold down the Ctrl and Alt keys (Windows) or Command and Option keys (macOS) and select another bar. This is the equivalent of using an *and* operator.

For an example, see "Example of Selecting Data in Multiple Histograms".

The Frequencies Report

For nominal and ordinal variables, the Frequencies report lists the levels of the variables, along with the associated frequency of occurrence and probabilities.

For each level of a categorical (nominal or ordinal) variable, the Frequencies report contains the information described in the following list. Missing values are omitted from the analysis.

Tip: Click a value in the Frequencies report to select the corresponding data in the histogram and data table.

Level Lists each value found for a response variable.

Count Lists the number of rows found for each level of a response variable. If you use a Freq variable, the Count is the sum of the Freq variables for each level of the response variable.

Prob Lists the probability (or proportion) of occurrence for each level of a response variable. The probability is computed as the count divided by the total frequency of the variable, shown at the bottom of the table.

StdErr Prob Lists the standard error of the probabilities. This column might be hidden. To show the column, right-click in the table and select **Columns > StdErr Prob**.

Cum Prob Contains the cumulative sum of the column of probabilities. This column might be hidden. To show the column, right-click in the table and select **Columns > Cum Prob**.

The Quantiles Report

For continuous variables, the Quantiles report lists the values of selected quantiles (sometimes called *percentiles*). For statistical details, see "Statistical Details for Quantiles".

The Summary Statistics Report

For continuous variables, the Summary Statistics report displays the mean, standard deviation, and other summary statistics. You can control which statistics appear in this report by selecting **Customize Summary Statistics** from the red triangle menu next to Summary Statistics.

Tip: You can specify which summary statistics show in the report each time you run a Distribution analysis for a continuous variable. Select **File > Preferences > Platforms > Distribution Summary Statistics**, and select the ones that you want to appear.

- "Description of the Summary Statistics Report" describes the statistics that appear by default.
- "Additional Summary Statistics" describes additional statistics that you can add to the report using the **Customize Summary Statistics** window.

Description of the Summary Statistics Report

Mean Estimates the expected value of the underlying distribution for the response variable, which is the arithmetic average of the column's values. It is the sum of the nonmissing values divided by the number of nonmissing values.

Std Dev The normal distribution is mainly defined by the mean and standard deviation. These parameters provide an easy way to summarize data as the sample becomes large:

- 68% of the values are within one standard deviation of the mean
- 95% of the values are within two standard deviations of the mean
- 99.7% of the values are within three standard deviations of the mean

Std Err Mean The standard error of the mean, which estimates the standard deviation of the distribution of the mean.

Upper and Lower Mean Confidence Limits The 95% confidence limits about the mean. They define an interval that is very likely to contain the true population mean.

N The total number of nonmissing values.

Additional Summary Statistics

Sum Weight The sum of a column assigned to the role of Weight (in the launch window). Sum Wgt is used in the denominator for computations of the mean instead of *N*.

Sum The sum of the response values.

Variance The sample variance, and the square of the sample standard deviation.

Skewness Measures sidedness or symmetry.

Kurtosis Measures peakedness or heaviness of tails. See "Kurtosis" for formula details.

CV The percent coefficient of variation. It is computed as the standard deviation divided by the mean and multiplied by 100. The coefficient of variation can be used to assess relative variation. For example, it can be used when comparing the variation in data measured in different units or with different magnitudes.

N Missing The number of missing observations.

N Zero The number of zero values.

N Unique The number of unique values.

Uncorrected SS The uncorrected sum of squares or sum of values squared.

Corrected SS The corrected sum of squares or sum of squares of deviations from the mean.

Autocorrelation (Appears only if you have not specified a Frequency variable.) First autocorrelation that tests if the residuals are correlated across the rows. This test helps detect non-randomness in the data.

Minimum Represents the 0 percentile of the data.

Maximum Represents the 100 percentile of the data.

Median Represents the 50th percentile of the data.

Mode The value that occurs most often in the data. If there are multiple modes, the smallest mode appears.

Trimmed Mean The mean calculated after removing the smallest p% and the largest p% of the data. The value of p is entered in the **Enter trimmed mean percent** text box at the bottom of the window. The Trimmed Mean option is not available if you have specified a Weight variable.

Geometric Mean The *n*th root of the product of the data. For example, geometric means are often used to calculate interest rates. The statistic is also helpful when the data contains a large value in a skewed distribution.

Note: Negative values result in missing numbers, and zero values (with no negative values) result in zero.

Range The difference between the maximum and minimum of the data.

Interquartile Range The difference between the 3rd and 1st quartiles.

Median Absolute Deviation (Does not appear if you have specified a Weight variable.) The median of the absolute deviations from the median.

Proportion Zero The proportion of nonmissing values that are equal to zero.

Proportion Nonzero The proportion of nonmissing values that are not equal to zero.

Robust Mean The robust mean, calculated in a way that is resistant to outliers, using Huber's M-estimation. See Huber and Ronchetti (2009).

Robust Std Dev The robust standard deviation, calculated in a way that is resistant to outliers, using Huber's M-estimation. See Huber and Ronchetti (2009).

Enter (1-alpha) for mean confidence interval Specify the alpha level for the mean confidence interval.

Enter trimmed mean percent Specify the trimmed mean percentage. The percentage is trimmed off each side of the data.

Summary Statistics Options

The red triangle menu next to Summary Statistics contains these options:

Customize Summary Statistics Select which statistics you want to appear from the list. You can select or deselect all summary statistics.

Show All Modes Shows all of the modes if there are multiple modes.

For statistical details, see "Summary Statistics".

Distribution Platform Options

The red triangle menu next to Distributions contains options that affect all of the reports and graphs in the Distribution platform.

Uniform Scaling Scales all axes with the same minimum, maximum, and intervals so that the distributions can be easily compared.

Stack Changes the orientation of the histogram and the reports to horizontal and stacks the individual distribution reports vertically. Deselect this option to return the report window to its original layout.

Arrange in Rows Enter the number of plots that appear in a row. This option helps you view plots vertically rather than in one wide row.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Options for Categorical Variables

The red triangle menus next to each variable in the report window contain additional options that apply to the variable. This section describes the options that are available for categorical (nominal or ordinal) variables.

Display Options See "Display Options for Categorical Variables".

Histogram Options See "Histogram Options for Categorical Variables".

Mosaic Plot Displays a mosaic bar chart for each nominal or ordinal response variable. A mosaic plot is a stacked bar chart where each segment is proportional to its group's frequency count.

Order By Reorders the histogram, mosaic plot, and Frequencies report in ascending or descending order, by count. To save the new order as a column property, use the **Save > Value Ordering** option.

Test Probabilities Displays a report that tests hypothesized probabilities. See "Examples of the Test Probabilities Option".

Confidence Interval This menu contains confidence levels. Select a value that is listed, or select **Other** to enter your own. JMP computes score confidence intervals.

Save See "Save Options for Categorical Variables".

Remove Permanently removes the variable and all its reports from the Distribution report.

Display Options for Categorical Variables

Frequencies Shows or hides the Frequencies report. See "The Frequencies Report".

Horizontal Layout Changes the orientation of the histogram and the reports to vertical or horizontal.

Axes on Left Moves the **Count**, **Prob**, and **Density** axes to the left instead of the right.

This option is applicable only if **Horizontal Layout** is selected.

Histogram Options for Categorical Variables

Histogram Shows or hides the histogram. See "Histograms".

Vertical Changes the orientation of the histogram from a vertical to a horizontal orientation.

Std Error Bars Draws the standard error bar on each level of the histogram.

Separate Bars Separates the histogram bars.

Histogram Color Changes the color of the histogram bars.

Count Axis Adds an axis that shows the frequency of column values represented by the histogram bars.

Prob Axis Adds an axis that shows the proportion of column values represented by histogram bars.

Density Axis Adds an axis that shows the length of the bars in the histogram.

The count and probability axes are based on the following calculations:

prob = (bar width)*density

count = (bar width)*density*(total count)

Show Percents Labels the percent of column values represented by each histogram bar.

Note: To specify the number of decimal places, right-click the histogram and select **Customize > Histogram**.

Show Counts Labels the frequency of column values represented by each histogram bar.

Save Options for Categorical Variables

Level Numbers Creates a new column in the data table called Level <colname>. The level number of each observation corresponds to the histogram bar that contains the observation.

Value Ordering (Use with the **Order By** option) Creates a new value ordering column property in the data table, reflecting the new order.

Script to Log Displays the script commands to generate the current report in the log window. Select **View > Log** to see the log window.

Options for Continuous Variables

The red triangle menus next to each variable in the report window contain additional options that apply to the variable. This section describes the options that are available for continuous variables.

Display Options See "Display Options for Continuous Variables".

Histogram Options See "Histogram Options for Continuous Variables".

Normal Quantile Plot Helps you visualize the extent to which the variable is normally distributed. See "Normal Quantile Plot".

Outlier Box Plot Shows the distribution and helps you identify possible outliers. See "Outlier Box Plot".

Quantile Box Plot Shows specific quantiles from the Quantiles report. See "Quantile Box Plot".

Stem and Leaf See "Stem and Leaf".

CDF Plot Creates a plot of the empirical cumulative distribution function. See "CDF Plot".

Test Mean Perform a one-sample test for the mean. See "Test Mean".

Test Std Dev Perform a one-sample test for the standard deviation. See "Test Std Dev".

Test Equivalence Assesses whether a population mean is equivalent to a hypothesized value. See "Test Equivalence".

Confidence Interval Choose confidence intervals for the mean and standard deviation. See "Confidence Intervals".

Prediction Interval Choose prediction intervals for a single observation, or for the mean and standard deviation of the next randomly selected sample. See "Prediction Intervals".

Tolerance Interval Computes an interval to contain at least a specified proportion of the population. See "Tolerance Intervals".

Process Capability Measures the conformance of a process to given specification limits. See "Process Capability".

Continuous Fit Fits distributions to continuous variables. See "Fit Distributions".

Discrete Fit (Available when all data values are integers.) Fits distributions to discrete variables. See "Fit Distributions".

Save Saves information about continuous or categorical variables. See "Prediction Intervals".

Remove Permanently removes the variable and all its reports from the Distribution report.

Display Options for Continuous Variables

Quantiles Shows or hides the Quantiles report. See "The Quantiles Report".

Set Quantile Increment Changes the quantile increment or revert to the default quantile increment.

Custom Quantiles Sets custom quantiles by values or by increments. You can specify the confidence level and choose whether to compute smoothed empirical likelihood quantiles (for large data sets, this can take some time).

- For more information about how the weighted average quantiles are estimated, see "Statistical Details for Quantiles".
- For more information about distribution-free confidence limits for the weighted average quantiles, see section 5.2 in Meeker et al. (2017).
- Smoothed empirical likelihood quantiles are based on a kernel density estimate. For more information about how these quantiles and their confidence limits are estimated, see Chen and Hall (1993).
- Confidence intervals and smoothed empirical likelihood quantiles are not available when fractional frequencies are used.

Summary Statistics Shows or hides the Summary Statistics report. See "The Summary Statistics Report".

Customize Summary Statistics Adds or removes statistics from the Summary Statistics report. See "The Summary Statistics Report".

Horizontal Layout Changes the orientation of the histogram and the reports to vertical or horizontal.

Axes on Left Moves the **Count**, **Prob**, **Density**, and **Normal Quantile Plot** axes to the left instead of the right.

This option is applicable only if **Horizontal Layout** is selected.

Histogram Options for Continuous Variables

Histogram Shows or hides the histogram. See "Histograms".

Shadowgram Replaces the histogram with a shadowgram. To understand a shadowgram, consider that if the bin width of a histogram is changed, the appearance of the histogram changes. A shadowgram overlays histograms with different bin widths. Dominant features of a distribution are less transparent on the shadowgram.

Note that the following options are not available for shadowgrams:

- Std Error Bars
- Show Counts
- Show Percents

Vertical Changes the orientation of the histogram from a vertical to a horizontal orientation.

Std Error Bars Draws the standard error bar on each level of the histogram using the standard error. The standard error bar adjusts automatically when you adjust the number of bars with the hand tool. See "Resize Histogram Bars for Continuous Variables" and "Standard Error Bars".

Set Bin Width Changes the bin width of the histogram bars. See "Resize Histogram Bars for Continuous Variables".

Histogram Color Changes the color of the histogram bars.

Count Axis Adds an axis that shows the frequency of column values represented by the histogram bars.

Note: If you resize the histogram bars, the count axis also resizes.

Prob Axis Adds an axis that shows the proportion of column values represented by histogram bars.

Note: If you resize the histogram bars, the probability axis also resizes.

Density Axis The density is the length of the bars in the histogram. Both the count and probability are based on the following calculations:

prob = (bar width)*density

count = (bar width)*density*(total count)

When looking at density curves that are added by the Fit Distribution option, the density axis shows the point estimates of the curves.

Note: If you resize the histogram bars, the density axis also resizes.

Show Percents Labels the proportion of column values represented by each histogram bar.

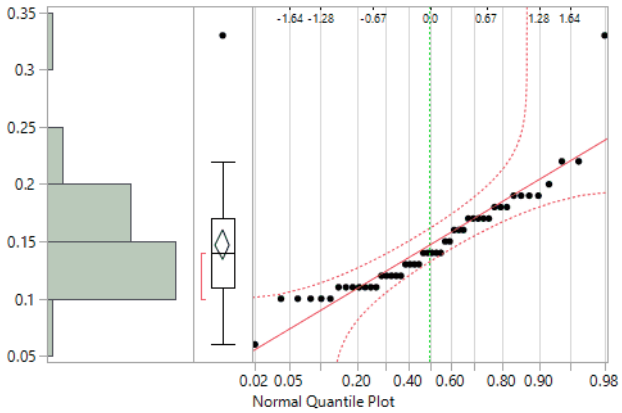
Show Counts Labels the frequency of column values represented by each histogram bar.

Normal Quantile Plot

Use the **Normal Quantile Plot** option to visualize the extent to which the variable is normally distributed. If a variable is normally distributed, the normal quantile plot approximates a diagonal straight line. This type of plot is also called a quantile-quantile plot, or Q-Q plot.

The normal quantile plot also shows Lilliefors confidence bounds (Conover 1980) and probability and normal quantile scales.

Figure 3.7 Normal Quantile Plot



Note the following information:

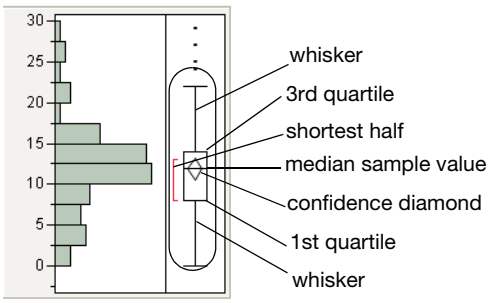
- The vertical axis shows the column values.
- The upper horizontal axis shows the normal quantile scale.
- The lower horizontal axis shows the empirical cumulative probability for each value.
- The dashed red line shows the Lilliefors confidence bounds.

For statistical details, see "Normal Quantile Plot".

Outlier Box Plot

Use the outlier box plot (also called a Tukey outlier box plot) to see the distribution and identify possible outliers. Generally, box plots show selected quantiles of continuous distributions.

Figure 3.8 Outlier Box Plot



Note the following aspects about outlier box plots:

- The horizontal line within the box represents the median sample value.

- The confidence diamond contains the mean and the upper and lower 95% of the mean. If you drew a line through the middle of the diamond, you would have the mean. The top and bottom points of the diamond represent the upper and lower 95% of the mean.
- The ends of the box represent the 25th and 75th quantiles, also expressed as the 1st and 3rd *quartile*, respectively.
- The difference between the 1st and 3rd quartiles is called the *interquartile range*.
- The box has lines that extend from each end, sometimes called *whiskers*. The whiskers extend from the ends of the box to the outermost data point that falls within these distances:

  1st quartile - 1.5*(interquartile range)

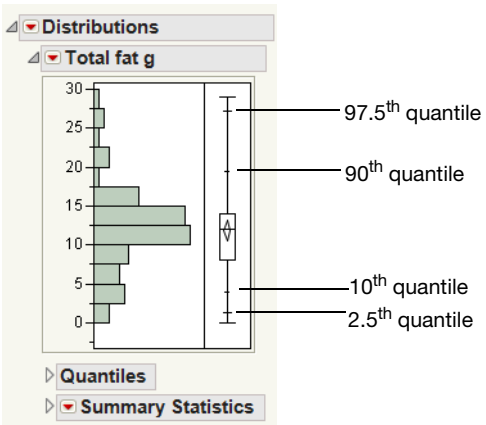
  3rd quartile + 1.5*(interquartile range)

  If the data points do not reach the computed ranges, then the whiskers are determined by the upper and lower data point values (not including outliers).
- The bracket outside of the box identifies the *shortest half*, which is the most dense 50% of the observations (Rousseeuw and Leroy 1987).
- To remove objects from outlier box plots, see "Remove Objects from the Outlier or Quantile Box Plot".

Quantile Box Plot

The Quantile Box Plot displays specific quantiles from the Quantiles report. If the distribution is symmetric, the quantiles in the box plot are approximately equidistant from each other. At a glance, you can see whether the distribution is symmetric. For example, if the quantile marks are grouped closely at one end, but have greater spacing at the other end, the distribution is skewed toward the end with more spacing.

Figure 3.9 Quantile Box Plot



Quantiles are values where the p^{th} quantile is larger than p% of the values. For example, 10% of the data lies below the 10^{th} quantile, and 90% of the data lies below the 90^{th} quantile.

Remove Objects from the Outlier or Quantile Box Plot

You can remove the confidence diamond and the shortest half from outlier or quantile box plots. You can remove them for a single graph, or remove them for all future graphs.

To remove them from an individual graph

1. Right-click the outlier box plot and select **Customize**.
2. Click **Box Plot**.
3. Deselect the check box next to **Confidence Diamond** or **Shortest Half**.

For more information about the Customize Graph window, see *Using JMP*.

To remove them for all future graphs

1. Select **File** > **Preferences** > **Platforms** > **Distribution**.
2. Deselect these options:
   - **Show Box Plot Confidence Diamond**
   - **Show Outlier Box Plot Shortest Half**
3. Click **OK**.

   Any box plots you now add in Distribution will not have the confidence diamond or shortest half.

Stem and Leaf

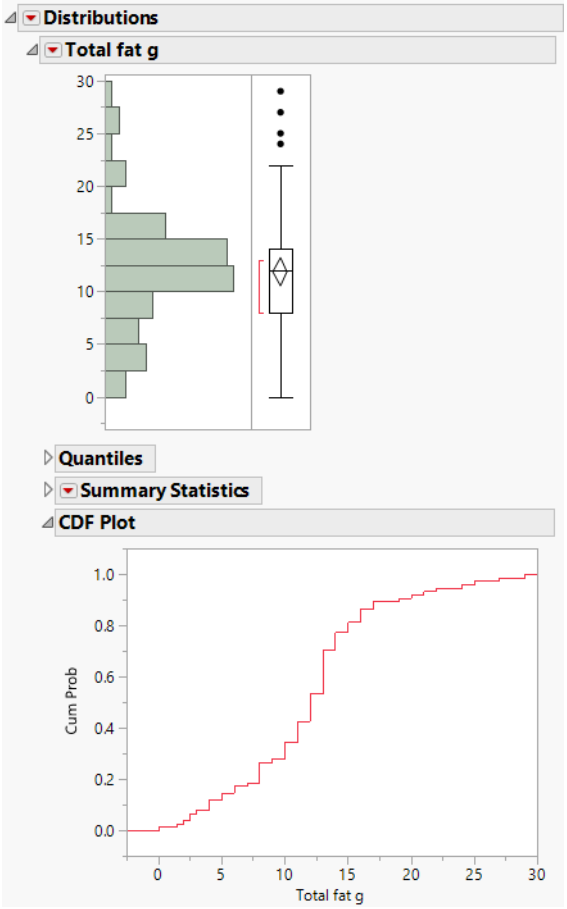
Each line of the plot has a Stem value that is the leading digit of a range of column values. The Leaf values are made from the next-in-line digits of the values. You can see the data point by joining the stem and leaf. In some cases, the numbers on the stem and leaf plot are rounded versions of the actual data in the table. The stem-and-leaf plot actively responds to clicking and the brush tool.

Note: The stem-and-leaf plot converts fractional frequencies to the smallest integer greater than or equal to the specified frequency.

CDF Plot

The CDF plot in Distribution creates a plot of the empirical cumulative distribution function. Use the CDF plot to determine the percent of data that is at or below a given value on the horizontal axis. When a distribution is fit to the data, the cumulative distribution function for the fitted distribution is overlaid on the CDF plot.

Figure 3.10 CDF Plot



For example, in this CDF plot, approximately 34% of the data are less than a total fat value of 10 grams.

Test Mean

Use the **Test Mean** window to specify options for and perform a one-sample test for the mean. If you specify a value for the standard deviation, a *z* test is performed. Otherwise, the sample standard deviation is used to perform a *t* test. You can also request the nonparametric Wilcoxon Signed-Rank test.

Use the **Test Mean** option repeatedly to test different values. Each time you test the mean, a new Test Mean report appears.

Description of the Test Mean Report

Statistics That Are Calculated for Test Mean

t Test (or z Test) Lists the value of the test statistic and the *p*-values for the two-sided and one-sided alternatives.

Signed-Rank (Appears only if the Wilcoxon Signed-Rank test is selected.) Lists the value of the Wilcoxon signed-rank statistic followed by *p*-values for the two-sided and one-sided alternatives. The test uses the Pratt method to address zero values. This is a nonparametric test whose null hypothesis is that the median equals the postulated value. It assumes that the distribution is symmetric. See "Wilcoxon Signed Rank Test".

Probability Values

Prob > |t| The probability of obtaining an absolute *t* value that is greater than the observed *t* value when the population mean is equal to the hypothesized value. This is the *p*-value for observed significance of the two-tailed *t* test.

Prob > t The probability of obtaining a *t* value greater than the computed sample *t* ratio when the population mean is not different from the hypothesized value. This is the *p*-value for an upper-tailed test.

Prob < t The probability of obtaining a *t* value less than the computed sample *t* ratio when the population mean is not different from the hypothesized value. This is the *p*-value for a lower-tailed test.

Descriptions of the Test Mean Options

PValue animation Starts an interactive visual representation of the *p*-value. Enables you to change the hypothesized mean value while watching how the change affects the *p*-value.

Power animation Starts an interactive visual representation of power and beta. You can change the hypothesized mean and sample mean while watching how the changes affect power and beta.

Remove Test Removes the mean test.

Test Std Dev

Use the **Test Std Dev** option to perform a one-sample test for the standard deviation. Use the **Test Std Dev** option repeatedly to test different values. Each time you test the standard deviation, a new Test Standard Deviation report appears.

Test Statistic Provides the value of the Chi-square test statistic. See "Standard Deviation Test".

Min PValue The probability of obtaining a more extreme Chi-square value when the population standard deviation does not differ from the hypothesized value. See "Standard Deviation Test".

Prob>ChiSq The probability of obtaining a Chi-square value greater than the computed sample Chi-square when the population standard deviation is not different from the hypothesized value. This is the *p*-value for observed significance of a one-tailed *t* test.

Prob<ChiSq The probability of obtaining a Chi-square value less than the computed sample Chi-square when the population standard deviation is not different from the hypothesized value. This is the *p*-value for observed significance of a one-tailed *t* test.

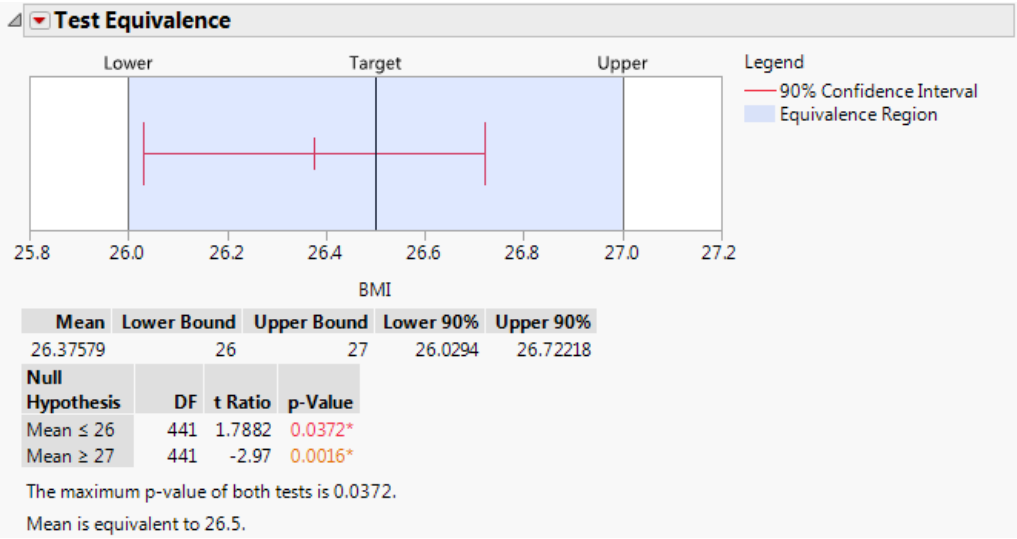
Test Equivalence

Equivalence tests assess whether a population mean is equivalent to a hypothesized value. You must define a threshold difference that is considered equivalent to no difference. The Test Equivalence option uses the Two One-Sided Tests (TOST) approach. Two one-sided *t* tests are constructed for the null hypotheses that the difference between the true mean and the hypothesized value exceeds the threshold. If both null hypotheses are rejected, this implies that the true difference does not exceed the threshold. You conclude that the mean can be considered practically equivalent to the hypothesized value.

When you select the Test Equivalence option, you specify the Hypothesized Mean, the threshold difference (Difference Considered Practically Zero), and the Confidence Level. The Confidence Level is 1 - alpha, where alpha is the significance level for each one-sided test.

The Test Equivalence report in Figure 3.11 is for the variable BMI in the Diabetes.jmp sample data table. The Hypothesized Mean is 26.5 and the Difference Considered Practically Zero is specified as 0.5.

Figure 3.11 Test Equivalence Report



Mean	Lower Bound	Upper Bound	Lower 90%	Upper 90%
26.37579	26	27	26.0294	26.72218

Null Hypothesis	DF	t Ratio	p-Value
Mean ≤ 26	441	1.7882	0.0372*
Mean ≥ 27	441	-2.97	0.0016*

The maximum p-value of both tests is 0.0372.

Mean is equivalent to 26.5.

The report shows the following:

- A plot of your defined equivalence region that shows the Target and boundaries, defined by vertical lines labeled Lower and Upper.
- A confidence interval for the calculated mean. This confidence interval is a 1 - 2*alpha level interval.
- A table that shows the calculated mean, the specified lower and upper bounds, and a (1 - 2*alpha) level confidence interval for the mean.
- A table that shows the results of the two one-sided tests.
- A note that summarizes the results, and states whether the mean can be considered equivalent to the Target value.

Confidence Intervals

The **Confidence Interval** options for continuous variables display confidence intervals for the mean and standard deviation. The **0.90**, **0.95**, and **0.99** options compute two-sided confidence intervals for the mean and standard deviation. Use the **Confidence Interval > Other** option to select a confidence level, and select one-sided or two-sided confidence intervals. You can also enter a known sigma. If you use a known sigma, the confidence interval for the mean is based on *z*-values rather than *t*-values.

The Confidence Intervals report shows the mean and standard deviation parameter estimates with upper and lower confidence limits for 1 - α.

Prediction Intervals

Prediction intervals concern a single observation, or the mean and standard deviation of the next randomly selected sample. The calculations assume that the given sample is selected randomly from a normal distribution. Select one-sided or two-sided prediction intervals.

When you select the **Prediction Interval** option for a variable, the Prediction Intervals window appears. Use the window to specify the confidence level, the number of future samples, and either a one-sided or two-sided limit.

Related Information

- For statistical details, see "Prediction Intervals".
- For an example, see "Example of Prediction Intervals".

Tolerance Intervals

A tolerance interval contains at least a specified proportion of the population. It is an interval estimate for a specified proportion of the population. Complete discussions of tolerance intervals are found in Meeker et al. (2017) and in Tamhane and Dunlop (2000).

When you select the **Tolerance Interval** option for a variable, the Tolerance Intervals window appears. Use the window to specify the confidence level, the proportion to cover, a one-sided or two-sided limit, and the method. The two available methods are Assume Normal Distribution and Nonparametric. The Assume Normal Distribution option computes tolerance intervals that are based on the assumption that the sample was randomly selected from a normal distribution. The Nonparametric option computes distribution-free tolerance intervals.

Note: When you specify the Assume Normal Distribution option, the value of k used for the intervals is contained in the report table. To view the value, in the Tolerance Intervals report, right-click the table and select Columns > K-factor.

Related Information

- For statistical details, see "Tolerance Intervals".
- For an example, see "Example of Tolerance Intervals".

Process Capability

Process capability analysis measures how well a process is performing compared to given specification limits. A good process is one that is stable and consistently produces product that is well within specification limits. A capability index is a measure that relates process performance, summarized by process centering and variability, to specification limits.

Specification Limits

If a column contains a Spec Limits column property and the Create Process Capability option on the launch window is selected, a Process Capability report is automatically created. This report is based on the normal distribution, unless the column also contains a Distribution column property. If the column contains a Distribution column property, the Process Capability report is based on the distribution specified in the column property.

Tip: To add specification limits to several columns at once, see *Quality and Process Methods*.

If a column does not contain specification limits, select **Process Capability** from the red triangle next to the name of the analysis variable and set specification limits in the Process Capability Analysis window.

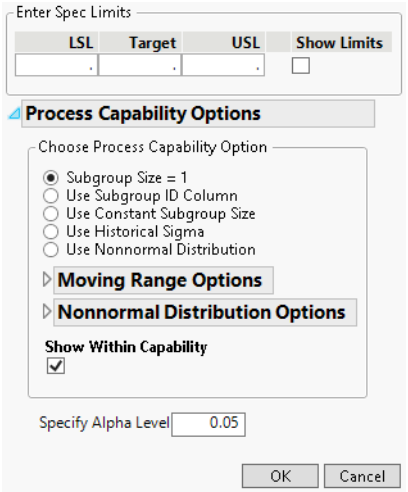
To save specification limits from a report to the data table as a column property, select **Save Spec Limits as Column Properties** from the Process Capability red triangle. When you repeat the process capability analysis, the saved specification limits are automatically retrieved.

Process Capability Analysis Window

The Process Capability Analysis window appears when you select the Process Capability option from the red triangle next to the name of the analysis variable.

Use the Process Capability Analysis window to specify options for the capability analysis, including specification limits, the underlying distribution for the analysis, and the estimation method for sigma. Process capability requires you to choose how to estimate sigma, the within-group (short-term) variation. Different suboptions appear depending on which process capability option you choose.

Figure 3.12 Process Capability Analysis Window



Enter Spec Limits Specifies the Lower Spec Limit, the Target, and the Upper Spec Limit for the process capability analysis. At least one of these must be a nonmissing value. If you select the Show Limits option, the specification limits appear on the histogram in the Distribution platform report.

Process Capability Options Depending on which option you choose, different additional options appear. Choose one of the following options:

Subgroup Size = 1 Sets the subgroup size to 1 and provides additional Moving Range options. See *Quality and Process Methods*.

Use Subgroup ID Column Enables you to select a subgroup ID column and provides additional Subgrouping and Moving Range options. See *Quality and Process Methods*.

Use Constant Subgroup Size Enables you to set a constant subgroup size and provides additional Subgrouping and Moving Range options. See *Quality and Process Methods*.

Use Historical Sigma Assigns a historically accepted value for sigma. See *Quality and Process Methods*.

Use Nonnormal Distribution Enables you to select a nonnormal distribution and provides additional Nonnormal Distribution Options. See *Quality and Process Methods*.

Show Within Capability (Available only when Subgroup Size = 1 or Use Nonnormal Distribution is selected in Process Capability Options.) Specifies if estimates of within sigma are shown in the report.

Specify Alpha Level Specifies the significance level for confidence limits.

Process Capability Analysis Report

After you click OK in the Process Capability Analysis window, a Process Capability report appears that contains a capability report for the selected variable. For more information about this report, see *Quality and Process Methods*.

- For statistical details, see *Quality and Process Methods*.
- For an example, see "Example of Process Capability".
- For the Process Capability platform, see *Quality and Process Methods*.

Note: You can set preferences for many of the options in the Process Capability report in Distribution at **File > Preferences > Platforms > Process Capability**.

Fit Distributions

You can use the options in the Continuous Fit or Discrete Fit submenus to fit a distribution to a continuous variable. When you fit a distribution to a continuous variable, a curve is overlaid on the histogram and a Compare Distributions report and a Fitted Distribution report are added to the report window. A red triangle menu in the Fitted Distribution report contains additional options. See "Fit Distribution Options". If a column contains a Distribution column property, the distribution in that column property is fit by default in the Distribution report.

Note: The Life Distribution platform also contains options for distribution fitting that might use different parameterizations and allow for censored observations. See *Reliability and Survival Methods*.

Continuous Fit

The Continuous Fit submenu contains options for fitting continuous distributions. For more information about the parameterization of these distributions, see "Continuous Fit Distributions".

Fit Normal Fits a normal distribution to the data. The normal distribution is often used to model symmetric data with most of the values falling in the middle of the curve.

Fit Cauchy Fits a Cauchy distribution to the data. The Cauchy distribution has an undefined mean and standard deviation. Although most data do not inherently follow a Cauchy distribution, it can be useful for estimating a robust location and scale for data that contain a large proportion of outliers (up to 50%).

Fit Student's t Fits a Student's *t* distribution to the data. The Student's *t* distribution is a robust option that spans the space between a normal distribution and a Cauchy distribution. As the degrees of freedom in the Student's *t* distribution approach infinity,

the distribution is equivalent to the normal. When the degrees of freedom in the Student's *t* distribution equals 1, the distribution is equivalent to the Cauchy. The Distribution platform estimates the degrees of freedom value.

Fit SHASH Fits a sinh-arcsinh (SHASH) distribution to the data. The SHASH distribution is similar to Johnson distributions in that it is a transformation to normality, but the SHASH distribution includes the normal distribution as a special case. This distribution can be symmetric or asymmetric.

Fit Exponential (Available only when all observations are nonnegative.) Fits an exponential distribution to the data. The exponential distribution is right-skewed and is often used to model lifetimes or the time between successive events.

Fit Gamma (Available only when all observations are positive.) Fits a gamma distribution to the data. The gamma distribution is a flexible distribution for modeling positive values.

Fit Lognormal (Available only when all observations are positive.) Fits a lognormal distribution to the data. The lognormal distribution is right-skewed and is often used to model lifetimes or the time until an event.

Fit Weibull (Available only when all observations are positive.) Fits a Weibull distribution to the data. The Weibull distribution is a flexible distribution and is often used to model lifetimes or the time until an event.

Fit Normal 2 Mixture Fits a mixture of two normal distributions. This flexible distribution is capable of fitting bimodal data.

Fit Normal 3 Mixture Fits a mixture of three normal distributions. This flexible distribution is capable of fitting multi-modal data.

Fit Smooth Curve Fits a smooth curve using nonparametric density estimation ("Kernel Smoother"). Control the amount of smoothing by changing the bandwidth with the slider that appears in the Nonparametric Density report.

Fit Johnson Fits a Johnson distribution to the data. The most appropriate of the three types of Johnson distribution (Su, Sb, and Sl) is fit and reported. The Johnson family of distributions is useful for its data-fitting capabilities because it supports every possible combination of skewness and kurtosis. Information about selection procedures and parameter estimation for the Johnson distributions can be found in Slifker and Shapiro (1980).

Fit Beta (Available only when all observations are between 0 and 1.) Fits a beta distribution to the data. The beta distribution is useful for modeling data that are between 0 and 1 (not inclusive) and is often used to model proportions or rates.

Fit All Fits all available continuous distributions to a variable. The Compare Distributions report contains statistics about each fitted distribution. By default, the best fit distribution

is selected and displayed on the histogram. Use the check boxes to show or hide a fit report and overlay curve for the selected distribution. Initially, the Compare Distributions list is sorted by AICc in ascending order.

Tip: You can quickly remove distributions from the Compare Distributions list by double-clicking the name of the distribution in the Distribution column. This action also removes the corresponding Fitted Distribution report.

Enable Legacy Fitters Shows or hides the Legacy Fitters submenu. Some features of distribution fitting have been updated in JMP 15. This option enables you to use the older features from previous JMP releases that have been retained for compatibility purposes. See "Details for the Legacy Distribution Fitters".

Discrete Fit

The Discrete Fit submenu is available when all of the data values are integers. The Discrete Fit submenu contains options for fitting discrete distributions. For more information about the parameterization of these distributions, see "Discrete Fit Distributions".

Fit Poisson Fits a Poisson distribution to the data. The Poisson distribution is useful for modeling the number of events in a given interval and is often expressed as count data.

Fit Negative Binomial Fits a negative binomial distribution to the data. The negative binomial distribution is useful for modeling the number of successes before a specified number of failures. The negative binomial distribution is also equivalent to the Gamma Poisson distribution.

Fit ZI Poisson (Available only when there are values of zero in the data.) Fits a zero-inflated Poisson distribution to the data. The zero-inflated Poisson assumes a greater proportion of the data are zero values than would occur in a standard Poisson distribution.

Fit ZI Negative Binomial (Available only when there are values of zero in the data.) Fits a zero-inflated negative binomial distribution to the data. The zero-inflated negative binomial assumes a greater proportion of the data are zero values than would occur in a standard negative binomial distribution.

Fit Binomial Fits a binomial distribution to the data. The binomial distribution is useful for modeling the total number of successes in n independent trials that all have a fixed probability, p, of success. The sample size can be specified as a fixed sample size for all observations, or it can be specified as another column in the data table that contains sample sizes for each row.

Note: When a non-constant sample size is specified, density curves, diagnostic plots, and profilers are not available.

Fit Beta Binomial Fits a beta binomial distribution to the data. The beta binomial distribution is an overdispersed version of the binomial distribution. It requires a sample size greater than one for each observation. The sample size can be specified as a fixed sample size for all observations, or it can be specified as another column in the data table that contains sample sizes for each row.

Note: When a non-constant sample size is specified, density curves, diagnostic plots, and profilers are not available.

Fit Distribution Options

Each fitted distribution report has a red triangle menu that contains additional options.

Density Curve Uses the estimated parameters of the distribution to overlay a density curve on the histogram.

Diagnostic Plots Contains the following options:

QQ Plot Shows or hides a quantile-quantile (QQ) plot. This plot shows the relationship between the observations and the quantiles obtained using the estimated parameters.

PP Plot Shows or hides a percentile-percentile (PP) plot. This plot shows the relationship between the empirical cumulative distribution function (CDF) and the fitted CDF obtained using the estimated parameters.

Profilers Contains the following options:

Distribution Profiler Shows or hides a prediction profiler of the cumulative distribution function (CDF).

Quantile Profiler Shows or hides a prediction profiler of the quantile function.

Save Columns Contains the following options:

Save Density Formula Saves a column to the data table that contains the density formula computed using the estimated parameter values.

Save Distribution Formula Saves a column to the data table that contains the cumulative distribution function (CDF) formula computed using the estimated parameter values.

Save Simulation Formula Saves a column to the data table that contains a formula that generates simulated values using the estimated parameters. This column can be used in the Simulate utility as a Column to Switch In. See "Simulate".

Save Transformed (Available only for Johnson and SHASH distribution fits.) Saves a column to the data table that contains a transform formula. The formula can be used to transform the analysis column to normality using the fitted distribution.

Goodness of Fit (Not available for Johnson, Smooth Curve, Normal Mixture, Binomial, or Beta Binomial distributions.) Shows or hides a Goodness-of-Fit Test report that contains a goodness-of-fit test for the fitted distribution.

For continuous fits, the goodness-of-fit test is the Anderson-Darling test. The *p*-value for the test is simulated using a parametric bootstrap, similar to the procedure described in Section 4.1 of Stephens (1974). For Normal distributions, the Shapiro-Wilk test for normality is also reported when the sample size is less than or equal to 2000 and there are no fixed parameters.

For discrete fits, the goodness-of-fit test is a Pearson chi-squared test.

Fix Parameters (Not available for Johnson distribution or smooth curve fits.) Enables you to fix parameters and re-estimate the non-fixed parameters. An Adequacy LR (likelihood ratio) Test report also appears, which tests your new parameters to determine whether they fit the data.

Process Capability (Not available for Cauchy, Student's *t*, or discrete distribution fits.) Enables you to create a Process Capability analysis using the fitted distribution, which is a measure of how well process performs with respect to the specification limits. When you select the Process Capability option from a Fitted Distribution red triangle menu, a window appears with the following options:

Enter Spec Limits Enables you to manually enter specification limits. To use the fitted distribution to calculate specification limits, leave this section blank and use the options under Calculate Quantile Spec Limits Options.

Calculate Quantile Spec Limits Options Enables you to calculate specification limits based on the fitted distribution. There are two methods available.

In the first method, you enter probabilities associated with the quantiles of the fitted distribution to calculate specification limits.

In the second method, you enter a K-Sigma Multiplier value that is used to calculate specification limits. This method has options for creating two-sided or one-sided limits.

After entering probabilities or a value for sigma multiplier, click **Calculate Spec Limits** to calculate the specification limits. These limits are entered into the Enter Spec Limits panel. Click **OK** to accept these limits and generate the Process Capability report.

Process Capability Options Contains the following options:

The Moving Range Options outline contains options that enable you to select the type of moving range statistic. See *Quality and Process Methods*.

The Nonnormal Distribution Options outline contains options that enable you to select methods used for nonnormal process capability calculations. See *Quality and Process Methods*.

For more information about the Process Capability options and report, see *Quality and Process Methods*.

Note: You can set preferences for many of the options in the Process Capability report in Distribution at **File > Preferences > Platforms > Process Capability**.

Remove Fit Removes the distribution fit from the report window.

Save Options for Continuous Variables

Use the Save menu options to save information about continuous variables. Each Save option generates a new column in the current data table. The new column is named by appending the variable name (denoted <colname> in the following definitions) to the Save command name (Table 3.1).

Select the Save options repeatedly to save the same information multiple times under different circumstances, such as before and after combining histogram bars. If you use a Save option multiple times, the column name is numbered (name1, name2, and so on) to ensure unique column names.

Table 3.1 Descriptions of Save Options

Option	Column Added to Data Table	Description
Level Numbers	Level <colname>	The level number of each observation corresponds to the histogram bar that contains the observation. The histogram bars are numbered from low to high, beginning with 1. **Note:** To maintain source information, value labels are added to the new column, but they are turned off by default.

Table 3.1 Descriptions of Save Options *(Continued)*

Option	Column Added to Data Table	Description
Level Midpoints	Midpoint <colname>	The midpoint value for each observation is computed by adding half the level width to the lower level bound. **Note:** To maintain source information, value labels are added to the new column, but they are turned off by default.
Ranks	Ranked <colname>	Provides a ranking for each of the corresponding column's values starting at 1. Duplicate response values are assigned consecutive ranks in order of their occurrence in the data table.
Ranks averaged	RankAvgd <colname>	If a value is unique, then the averaged rank is the same as the rank. If a value occurs *k* times, the average rank is computed as the sum of the value's ranks divided by *k*.
Prob Scores	Prob <colname>	For *N* nonmissing scores, the probability score of a value is computed as the averaged rank of that value divided by $N + 1$. This column is similar to the empirical cumulative distribution function.
Normal Quantiles	N-Quantile <colname>	Saves the Normal quantiles. See "Normal Quantile Plot".
Standardized	Std <colname>	Saves standardized values. See "Saving Standardized Data".
Centered	Centered <colname>	Saves values for centering on zero.
Robust Standardized	Robust Std <colname>	Saves a column that contains the response value centered around the robust mean and standardized using the robust standard deviation.
Robust Centered	Robust Centered <colname>	Saves a column that contains the response value centered around the robust mean.
Script to Log	(none)	Prints the script to the log window. Run the script to re-create the analysis.

Additional Examples of the Distribution Platform

- "Example of Selecting Data in Multiple Histograms"
- "Example Using a By Variable"
- "Examples of the Test Probabilities Option"
- "Example of Prediction Intervals"
- "Example of Tolerance Intervals"
- "Example of Process Capability"

Example of Selecting Data in Multiple Histograms

This example shows you how to select and highlight data in several histograms at once.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze > Distribution**.
3. Select Type and Size Co and click **Y, Columns.**
4. Click **OK**.

   You want to see the type distribution of companies that are small.
5. Click the bar next to small.

   You can see that there are more small computer companies than there are pharmaceutical companies. To broaden your selection, add medium companies.
6. Hold down the Shift key. In the Size Co histogram, click the bar next to medium.

   You can see the type distribution of small and medium sized companies. See Figure 3.13 at left. To narrow your selection, you want to see the small and medium pharmaceutical companies only.
7. Hold down the Ctrl and Shift keys (on Windows) or the Command and Shift keys (on macOS). In the Type histogram, click in the Computer bar to deselect it.

   You can see how many of the small and medium companies are pharmaceutical companies. See Figure 3.13 at right.

Figure 3.13 Selecting Data in Multiple Histograms

Broaden the selection using the Shift key.

Narrow the selection using Ctrl + Shift (Windows) or Command + Shift (macOS).

Example Using a By Variable

This example shows you how to use a By variable to create a separate analysis for each level of a categorical variable.

1. Select **Help > Sample Data Library** and open Lipid Data.jmp.
2. Select **Analyze > Distribution**.
3. Select Cholesterol and click **Y, Columns**.
4. Select Gender and click **By**.

   This results in a separate analysis for each level of Gender (female and male).
5. Click **OK**.

Change the orientation of the histograms and the reports.

6. Click the Distributions red triangle and select **Stack**.

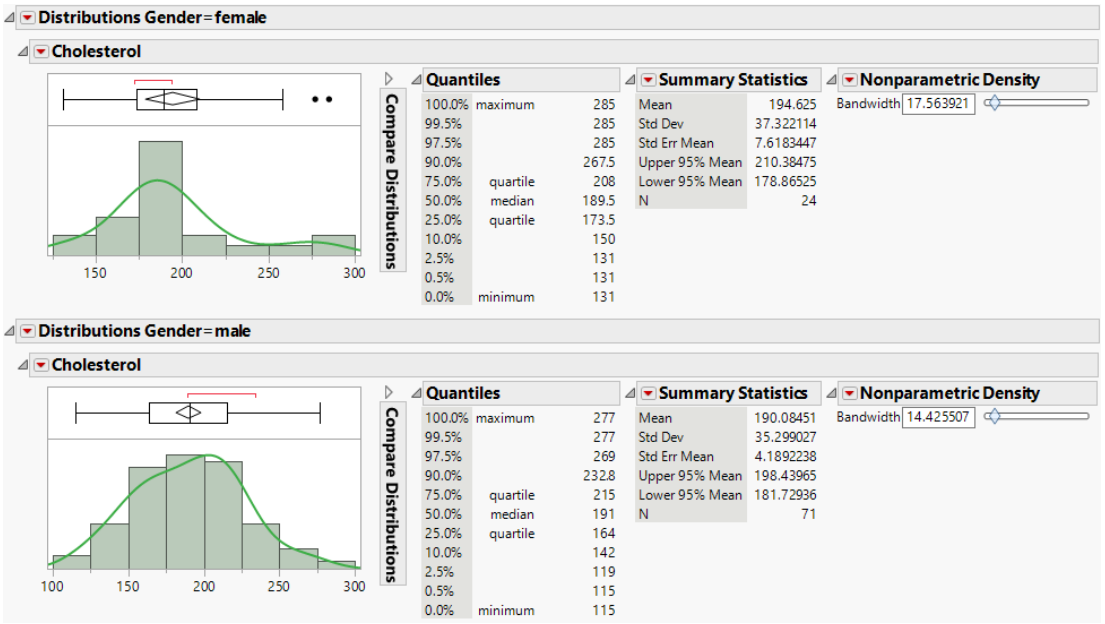
Add a smooth curve to both histograms.

7. Hold down the Ctrl key. Click the Cholesterol red triangle and select **Continuous Fit > Fit Smooth Curve**.

Hide the Compare Distributions report.

8. Hold down the Ctrl key. Click the gray disclosure icon next to **Compare Distributions**.

Figure 3.14 Separate Distributions by Gender



Examples of the Test Probabilities Option

These examples show you how to create a Test Probabilities report for variables with two levels or more.

Initiate a test probability report for a variable with more than two levels:

1. Select **Help > Sample Data Library** and open VA Lung Cancer.jmp.
2. Select **Analyze > Distribution**.
3. Select Cell Type and click **Y, Columns**.
4. Click **OK**.
5. Click the Cell Type red triangle and select **Test Probabilities**.

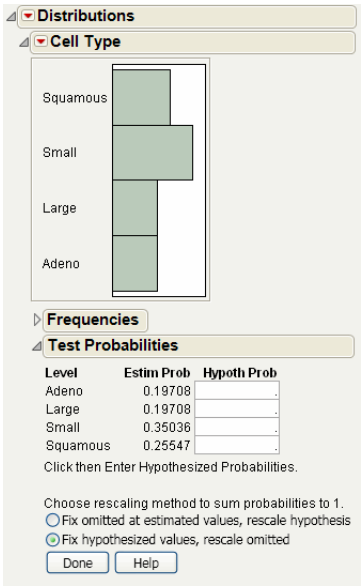
   See Figure 3.15 at left.

Initiate a test probability report for a variable with exactly two levels:

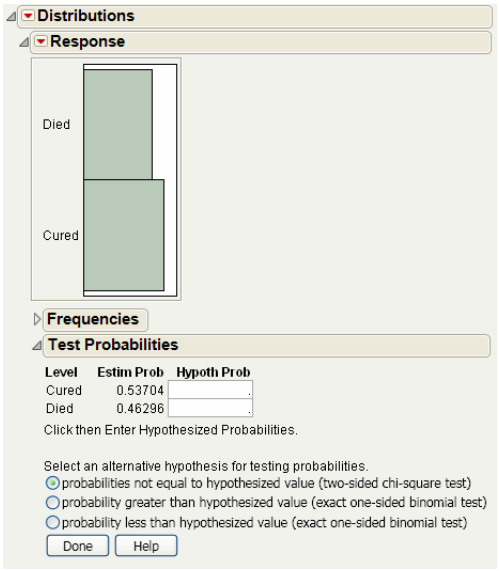
1. Select **Help > Sample Data Library** and open Penicillin.jmp.
2. Select **Analyze > Distribution**.
3. Select Response and click **Y, Columns**.
4. Click **OK**.
5. Click the Response red triangle and select **Test Probabilities**.

See Figure 3.15 at right.

Figure 3.15 Examples of Test Probabilities Options



report options for a variable with more than two levels



report options for a variable with exactly two levels

Example of Generating the Test Probabilities Report

To generate a test probabilities report for a variable with more than two levels:

1. Refer to Figure 3.15 at left. Type 0.25 in all four Hypoth Prob fields.
2. Click the **Fix hypothesized values, rescale omitted** button.
3. Click **Done**.

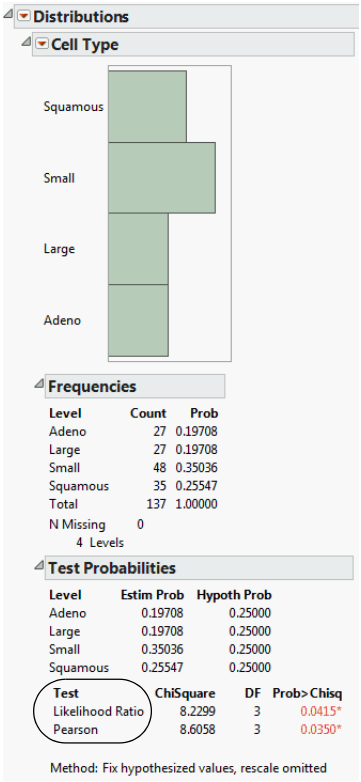
Likelihood Ratio and Pearson Chi-square tests are calculated. See Figure 3.16 at left.

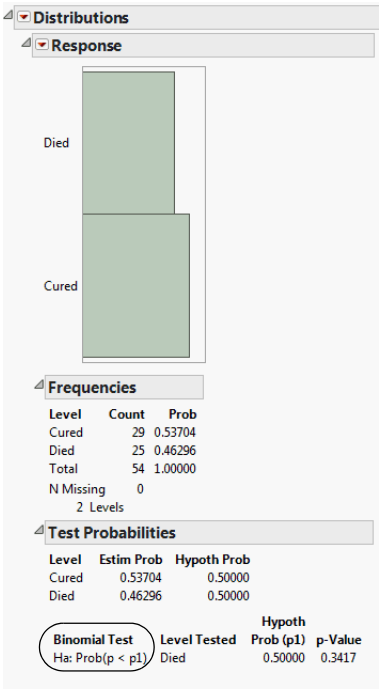
To generate a test probabilities report for a variable with exactly two levels:

1. Refer to Figure 3.15 at right. Type 0.5 in both Hypoth Prob fields.
2. Click the **probability less than hypothesized value** button.
3. Click **Done**.

Exact probabilities are calculated for the binomial test. See Figure 3.16 at right.

Figure 3.16 Examples of Test Probabilities Reports



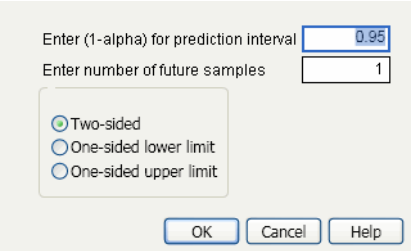


Example of Prediction Intervals

This example shows you how to add prediction intervals to your distribution report. Suppose you are interested in computing prediction intervals for the next 10 observations of ozone level.

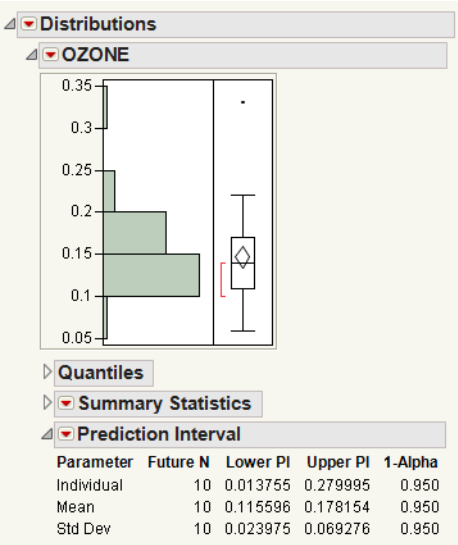
1. Select **Help > Sample Data Library** and open Cities.jmp.
2. Select **Analyze > Distribution**.
3. Select OZONE and click **Y, Columns**.
4. Click **OK**.
5. Click the OZONE red triangle and select **Prediction Interval**.

Figure 3.17 The Prediction Intervals Window



6. In the Prediction Intervals window, type 10 next to **Enter number of future samples**.
7. Click **OK**.

Figure 3.18 Example of a Prediction Interval Report



In this example, you can be 95% confident about the following:

- Each of the next 10 observations will be between 0.013755 and 0.279995.
- The mean of the next 10 observations will be between 0.115596 and 0.178154.
- The standard deviation of the next 10 observations will be between 0.023975 and 0.069276.

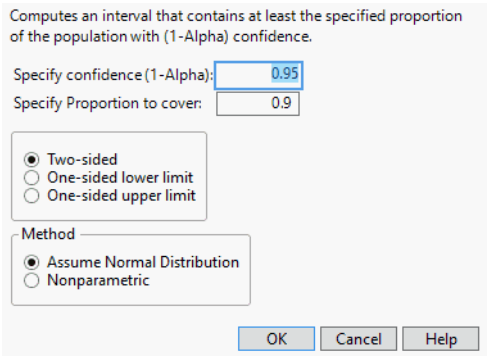
Example of Tolerance Intervals

This example shows you how to add tolerance intervals to your distribution report. Suppose you want to estimate an interval that contains 90% of ozone level measurements.

1. Select **Help > Sample Data Library** and open Cities.jmp.

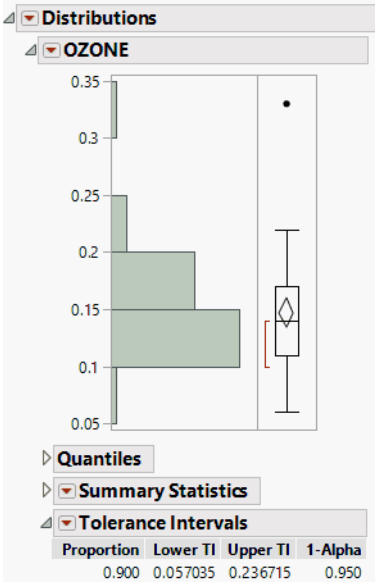
2. Select **Analyze > Distribution**.
3. Select OZONE and click **Y, Columns**.
4. Click **OK**.
5. Click the OZONE red triangle and select **Tolerance Interval**.

Figure 3.19 The Tolerance Intervals Window



6. Keep the default selections, and click **OK**.

Figure 3.20 Example of a Tolerance Interval Report



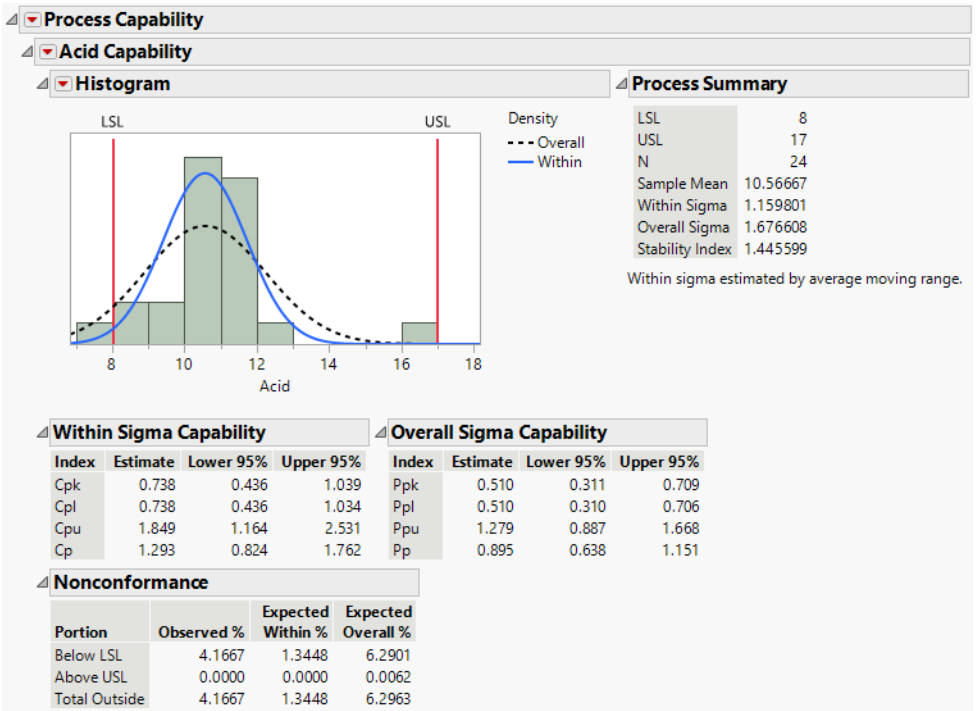
In this example, you can be 95% confident that at least 90% of the population lie between 0.057035 and 0.236715, based on the Lower TI (tolerance interval) and Upper TI values.

Example of Process Capability

This example shows you how to add process capability results to your distribution report. Suppose you want to characterize the acidity of pickles. The lower and upper specification limits are 8 and 17, respectively.

1. Select **Help > Sample Data Library** and open Quality Control/Pickles.jmp.
2. Select **Analyze > Distribution**.
3. Select Acid and click **Y, Columns**.
4. Click **OK**.
5. Click the Acid red triangle and select **Process Capability**.
6. Type 8 for the **LSL** (lower specification limit).
7. Type 17 for the **USL** (upper specification limit).
8. Click **OK**.

Figure 3.21 Example of the Process Capability Report



The Process Capability results are added to the report. The specification limits appear on the histogram in the Process Capability report so that the data can be visually compared to the limits. As you can see, some of the acidity levels are below the lower specification limit, and some are very close to the upper specification limit. The Ppk value is 0.510, indicating a process that is not capable, relative to the given specification limits.

Statistical Details for the Distribution Platform

- "Standard Error Bars"
- "Statistical Details for Quantiles"
- "Summary Statistics"
- "Normal Quantile Plot"
- "Wilcoxon Signed Rank Test"
- "Standard Deviation Test"
- "Normal Quantiles"
- "Saving Standardized Data"
- "Prediction Intervals"
- "Tolerance Intervals"
- "Continuous Fit Distributions"
- "Discrete Fit Distributions"

Standard Error Bars

Standard error bars are calculated using the standard error $\sqrt{np_i(1-p_i)}$ where $p_i=n_i / n$.

Statistical Details for Quantiles

This section describes how quantiles are computed.

To compute the pth quantile of n nonmissing values in a column, arrange the n values in ascending order and call these column values y_1, y_2, ..., y_n. Compute the rank number for the pth quantile as $p / 100(n + 1)$.

- If the result is an integer, the pth quantile is that rank's corresponding value.
- If the result is not an integer, the pth quantile is found by interpolation. The pth quantile, denoted q_p, is defined as follows:

$$q_p = (1-f)y_i + (f)y_{i+1}$$

where:

- n is the number of nonmissing values for a variable
- $y_1, y_2, \ldots, y_n$ represents the ordered values of the variable
- y_{n+1} is taken to be y_n
- i is the integer part and f is the fractional part of $(n+1)p$.
- $(n + 1)p = i + f$

For example, suppose a data table has 15 rows and you want to find the 75th and 90th quantile values of a continuous column. After the column is arranged in ascending order, the ranks that contain these quantiles are computed as follows:

$$\frac{75}{100}(15+1) = 12 \quad \text{and} \quad \frac{90}{100}(15+1) = 14.4$$

The value y_{12} is the 75th quantile. The 90th quantile is interpolated by computing a weighted average of the 14th and 15th ranked values as $y_{90} = 0.6y_{14} + 0.4y_{15}$.

Summary Statistics

This section contains statistical details for specific statistics in the Distribution Summary Statistics report.

Mean

The mean is the sum of the nonmissing values divided by the number of nonmissing values. If you assigned a **Weight** or **Freq** variable, the mean is computed as follows:

1. Each column value is multiplied by its corresponding weight or frequency.
2. These values are added and divided by the sum of the weights or frequencies.

Std Dev

The standard deviation measures the spread of a distribution around the mean. It is often denoted as s and is the square root of the sample variance, denoted s^2.

$$s = \sqrt{s^2}$$

where

$$s^2 = \sum_{i=1}^{N} \frac{w_i(y_i - \bar{y}_w)^2}{N-1}$$

$\bar{y}_w$ = weighted mean

Std Err Mean

The standard error mean is computed by dividing the sample standard deviation, *s*, by the square root of *N*. In the launch window, if you specified a column for Weight or Freq, then the denominator is the square root of the sum of the weights or frequencies.

Skewness

Skewness is based on the third moment about the mean and is computed as follows:

$$\sum w_i^{\frac{3}{2}} z_i^3 \frac{N}{(N-1)(N-2)} \quad \text{where} \quad z_i = \frac{x_i - \bar{x}}{s}$$

and w_i is a weight term (= 1 for equally weighted items).

Kurtosis

Kurtosis is based on the fourth moment about the mean and is computed as follows:

$$\frac{n(n+1)}{(n-1)(n-2)(n-3)} \sum_{i=1}^{n} w_i^2 \left(\frac{x_i - \bar{x}}{s}\right)^4 - \frac{3(n-1)^2}{(n-2)(n-3)}$$

where w_i is a weight term (= 1 for equally weighted items). Using this formula, the Normal distribution has a kurtosis of 0. This formula is often referred to as the excess kurtosis.

Normal Quantile Plot

The empirical cumulative probability for each value is computed as follows:

$$\frac{r_i}{N+1}$$

where r_i is the rank of the *i*th observation, and *N* is the number of nonmissing (and nonexcluded) observations.

The normal quantile values are computed as follows:

$$\Phi^{-1}\left(\frac{r_i}{N+1}\right)$$

where Φ is the cumulative probability distribution function for the normal distribution.

These normal quantile values are Van Der Waerden approximations to the order statistics that are expected for the normal distribution.

Wilcoxon Signed Rank Test

The Wilcoxon signed-rank test can be used to test for the median of a single population or to test matched-pairs data for a common median. In the case of matched pairs, the test reduces to testing the single population of paired differences for a median of 0. The test assumes that the underlying population is symmetric.

The Wilcoxon test accommodates tied values. The test statistic is adjusted for differences of zero using a method suggested by Pratt. See Lehmann and D'Abrera (2006), Pratt (1959), and Cureton (1967).

Testing for the Median of a Single Population

- There are N observations:

  $X_1, X_2, \ldots, X_N$
- The null hypothesis is:

  H_0: distribution of X is symmetric around m
- The differences between observations and the hypothesized value m are calculated as follows:

  $D_j = X_j - m$

Testing for the Equality of Two Population Medians with Matched Pairs Data

A special case of the Wilcoxon signed-rank test is applied to matched-pairs data.

- There are N pairs of observations from two populations:

  $X_1, X_2, \ldots, X_N$ and $Y_1, Y_2, \ldots, Y_N$
- The null hypothesis is:

  H_0: distribution of $X - Y$ is symmetric around 0
- The differences between pairs of observations are calculated as follows:

  $D_j = X_j - Y_j$

Wilcoxon Signed-Rank Test Statistic

The test statistic is based on the sum of the signed ranks. Signed ranks are defined as follows:

- The absolute values of the differences, $|D_j|$, are ranked from smallest to largest.
- The ranks start with the value 1, even if there are differences of zero.
- When there are tied absolute differences, they are assigned the average, or *midrank*, of the ranks of the observations.

Denote the rank or midrank for a difference D_j by R_j. Define the signed rank for D_j as follows:

- If the difference D_j is positive, the signed rank is R_j.
- If the difference D_j is zero, the signed rank is 0.
- If the difference D_j is negative, the signed rank is $-R_j$.

The signed-rank statistic is computed as follows:

$$S = \frac{1}{2}\sum_{j=1}^{N} \text{signed ranks}$$

Define the following:

d_0 is the number of signed ranks that equal zero

R^+ is the sum of the positive signed ranks

Then the following holds:

$$S = R^+ - \frac{1}{4}[N(N+1) - d_0(d_0+1)]$$

Wilcoxon Signed-Rank Test P-Values

For $N \leq 20$, exact p-values are calculated.

For $N > 20$, a Student's t approximation to the statistic defined below is used. Note that a correction for ties is applied. See Iman (1974) and Lehmann and D'Abrera (2006).

Under the null hypothesis, the mean of S is zero. The variance of S is given by the following:

$$Var(S) = \frac{1}{24}\left[N(N+1)(2N+1) - d_0(d_0+1)(2d_0+1) - \frac{1}{2}\sum_{i>0} d_i(d_i+1)(d_i-1)\right]$$

The last summation in the expression for *Var(S)* is a correction for ties. The notation d_i for i > 0 represents the number of values in the i^{th} group of nonzero signed ranks. (If there are no ties for a given signed rank, then $d_i = 1$ and the summand is 0.)

The statistic *t* given by the following has an approximate *t* distribution with *N* - 1 degrees of freedom:

$$t = \frac{S}{\sqrt{\frac{N \cdot Var(S) - S^2}{N-1}}}$$

Standard Deviation Test

Here is the formula for calculating the Test Statistic:

$$\frac{(n-1)s^2}{\sigma^2}$$

The Test Statistic is distributed as a Chi-square variable with *n* - 1 degrees of freedom when the population is normal.

The Min PValue is the *p*-value of the two-tailed test, and is calculated as follows:

2*min(p1,p2)

where p1 is the lower one-tail *p*-value and p2 is the upper one-tail *p*-value.

Normal Quantiles

The normal quantile values are computed as follows:

$$\Phi^{-1}\left(\frac{r_i}{N+1}\right)$$

where:

Φ is the cumulative probability distribution function for the normal distribution.

r_i is the rank of the *i*th observation.

N is the number of nonmissing observations.

Saving Standardized Data

The standardized values are computed using the following formula:

$$\frac{X - \overline{X}}{S_X}$$

where:

X is the original column

$\overline{X}$ is the mean of column X

S_X is the standard deviation of column X

Prediction Intervals

The prediction intervals are defined as follows:

- For m future observations:

$$[\underset{\sim}{y}_m, \tilde{y}_m] = \overline{X} \pm t_{(1-\alpha/2m;n-1)} \times \sqrt{1 + \frac{1}{n}} \times s \quad \text{for} \quad m \geq 1$$

- For the mean of m future observations:

$$[Y_l, Y_u] = \overline{X} \pm t_{(1-\alpha/2,\, n-1)} \times \sqrt{\frac{1}{m} + \frac{1}{n}} \times s \quad \text{for} \quad m \geq 1 \quad .$$

- For the standard deviation of m future observations:

$$[s_l, s_u] = \left[s \times \sqrt{\frac{1}{F_{(1-\alpha/2;(n-1,\, m-1))}}},\ s \times \sqrt{F_{(1-\alpha/2;(m-1,\, n-1))}} \right] \quad \text{for} \quad m \geq 2$$

where m = number of future observations, and n = number of points in current analysis sample.

- The one-sided intervals are formed by using 1-α in the quantile functions.

See Meeker et al. (2017, ch. 4).

Tolerance Intervals

This section contains statistical details for one-sided and two-sided tolerance intervals.

Normal Distribution-Based Intervals

One-Sided Interval

The one-sided interval is computed as follows:

Lower Limit = $\bar{x} - k's$

Upper Limit = $\bar{x} + k's$

where

$$k' = t(1-\alpha, n-1, \Phi^{-1}(p)\cdot\sqrt{n})/\sqrt{n}$$

s is the standard deviation

t is the quantile from the non-central t-distribution

Φ^{-1} is the standard normal quantile

Two-Sided Interval

The two-sided interval is computed as follows:

$$[T_{p_L}, T_{p_U}] = [\bar{x} - k_{(1-\alpha/2;p,n)}s, \bar{x} + k_{(1-\alpha/2;p,n)}s]$$

where s is the standard deviation and $k_{(1-\alpha/2;\,p,n)}$ is a constant.

To determine k, consider the fraction of the population captured by the tolerance interval. Tamhane and Dunlop (2000) define this fraction as follows:

$$\Phi\left(\frac{\bar{x}+ks-\mu}{\sigma}\right) - \Phi\left(\frac{\bar{x}-ks-\mu}{\sigma}\right)$$

where Φ denotes the standard normal cdf (cumulative distribution function).

Therefore, k solves the following equation:

$$P\left\{\Phi\left(\frac{\bar{X}+ks-\mu}{\sigma}\right) - \Phi\left(\frac{\bar{X}-ks-\mu}{\sigma}\right) \geq 1-\gamma\right\} = 1-\alpha$$

where 1 - γ is the fraction of all future observations contained in the tolerance interval.

For more information about normal distribution-based tolerance intervals, see Tables J.1a, J.1b, J.6a, and J.6b of Meeker et al. (2017).

Nonparametric Intervals

One-Sided Lower Limit

The lower 100(1 - α)% one-sided tolerance limit to contain at least a proportion β of the sampled distribution from a sample of size n is the order statistic $x_{(l)}$. The index l is computed as follows:

$$l = n - \Phi_{bin}^{-1}(1 - \alpha, n, \beta)$$

where $\Phi^{-1}_{bin}(1\text{-}\alpha, n, \beta)$ is the $(1 - \alpha)^{th}$ quantile of the binomial distribution with n trials and probability of success β.

The actual confidence level is computed as $\Phi_{bin}(n\text{-}l, n, \beta)$, where $\Phi_{bin}(x, n, \beta)$ is the probability of a binomially distributed random variable with n trials and probability of success β being less than or equal to x.

Note that to compute a lower one-sided distribution-free tolerance interval, the sample size n must be at least as large as $(\log\alpha)/(\log\beta)$.

One-Sided Upper Limit

The upper 100(1 - α)% one-sided tolerance limit to contain at least a proportion β of the sampled distribution from a sample of size n is the order statistic $x_{(u)}$. The index u is computed as follows:

$$u = 1 + \Phi_{bin}^{-1}(1 - \alpha, n, \beta)$$

where $\Phi^{-1}_{bin}(1\text{-}\alpha, n, \beta)$ is the $(1 - \alpha)^{th}$ quantile of the binomial distribution with n trials and probability of success β.

The actual confidence level is computed as $\Phi_{bin}(u\text{-}1, n, \beta)$, where $\Phi_{bin}(x, n, \beta)$ is the probability of a binomially distributed random variable with n trials and probability of success β being less than or equal to x.

Note that to compute an upper one-sided distribution-free tolerance interval, the sample size n must be at least as large as $(\log\alpha)/(\log\beta)$.

Two-Sided Interval

The 100(1 - α)% two-sided tolerance interval to contain at least a proportion β of the sampled distribution from a sample of size n is computed as follows:

$$[\tilde{T}_{p_L}, \tilde{T}_{p_U}] = [x_{(l)}, x_{(u)}]$$

where $x_{(i)}$ is the i^{th} order statistic and l and u are computed as follows:

Let $\nu = n - \Phi^{-1}_{bin}(1-\alpha, n, \beta)$, where $\Phi^{-1}_{bin}(1-\alpha, n, \beta)$ is the $(1 - \alpha)^{th}$ quantile of the binomial distribution with n trials and probability of success β. If ν is less than 2, a two-sided distribution-free tolerance interval cannot be computed. If ν is greater than or equal to 2, l = floor(ν/2) and u = floor(n + 1 - ν/2).

The actual confidence level is computed as $\Phi_{bin}(u-l-1, n, \beta)$, where $\Phi_{bin}(x, n, \beta)$ is the probability of a binomially distributed random variable with n trials and probability of success β being less than or equal to x.

Note that to compute a two-sided distribution-free tolerance interval, the sample size n must be at least as large as the n in the following equation:

$$1 - \alpha = 1 - n\beta^{n-1} + (n-1)\beta^{n}$$

For more information about distribution-free tolerance intervals, see Meeker et al. (2017, sec. 5.3).

Continuous Fit Distributions

This section contains statistical details for the options in the Continuous Fit menu. Unless otherwise specified, confidence intervals for parameter estimates use Wald-based calculations.

Fit Normal

The Fit Normal option estimates the two parameters of the normal distribution:

- μ (the mean) defines the location of the distribution on the x-axis
- σ (standard deviation) defines the dispersion or spread of the distribution

The standard normal distribution occurs when μ = 0 and σ = 1.

$$\text{pdf:} \quad \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right] \quad \text{for} \quad -\infty < x < \infty \;;\; -\infty < \mu < \infty \;;\; 0 < \sigma$$

$E(x) = \mu$

$\text{Var}(x) = \sigma^2$

Note: Confidence intervals for the mean estimate are based on the *t* distribution. Confidence intervals for the scale parameter are based on the χ^2 distribution.

Fit Cauchy

The Fit Cauchy option fits a Cauchy distribution with location μ and scale σ.

$$\text{pdf:} \quad \left\{\pi\sigma\left[1+\left(\frac{x-\mu}{\sigma}\right)^2\right]\right\}^{-1} \quad \text{for} \quad -\infty < x < \infty \;;\; -\infty < \mu < \infty \;;\; 0 < \sigma$$

$E(x)$ = undefined

$\text{Var}(x)$ = undefined

Fit Student's t

The Fit Student's *t* option fits a Student's *t* distribution with location μ, scale σ, and degrees of freedom ν.

$$\text{pdf:} \quad \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)}\frac{1}{\sqrt{\nu\pi\sigma^2}}\left[1+\frac{(x-\mu)^2}{\nu\sigma^2}\right]^{-\left(\frac{\nu+1}{2}\right)} \quad \text{for} \quad -\infty < x < \infty \;;\; -\infty < \mu < \infty \;;\; 0 <$$

$\sigma; \quad 1 \le \nu$

Fit SHASH

The Fit SHASH option fits a sinh-arcsinh (SHASH) distribution. The SHASH distribution is based on a transformation of the normal distribution and includes the normal distribution as a special case. It can be symmetric or asymmetric. The shape is determined by the two shape parameters, γ and δ. For more information about the SHASH distribution, see Jones and Pewsey (2009).

$$\text{pdf:} \quad f(x) = \frac{\delta\cosh(w)}{\sqrt{\sigma^2+(x-\theta)^2}}\phi[\sinh(w)] \quad \text{for} \quad -\infty < \gamma, x, \theta < \infty \;;\; 0 < \delta, \sigma$$

where

$\phi(\cdot)$ is the standard normal pdf

$$w = \gamma + \delta \sinh^{-1}\left(\frac{x-\theta}{\sigma}\right)$$

- When $\gamma = 0$ and $\delta = 1$, the SHASH distribution is equivalent to the normal distribution with location θ and scale σ.
- The transformation $\sinh(w)$ is normally distributed with $\mu = 0$ and $\sigma = 1$.

Fit Exponential

The exponential distribution is especially useful for describing events that randomly occur over time, such as survival data. The exponential distribution might also be useful for modeling elapsed time between the occurrence of non-overlapping events. Examples of non-overlapping events include the following: the time between a user's computer query and response of the server, the arrival of customers at a service desk, or calls coming in at a switchboard.

The Exponential distribution is a special case of the two-parameter Weibull when $\beta = 1$ and $\alpha = \sigma$, and also a special case of the Gamma distribution when $\alpha = 1$.

pdf: $\frac{1}{\sigma}\exp\left(-\frac{x}{\sigma}\right)$ for $0 < \sigma;\ 0 \le x$

$E(x) = \sigma$

$Var(x) = \sigma^2$

Devore (1995) notes that an exponential distribution is *memoryless*. Memoryless means that if you check a component after t hours and it is still working, the distribution of additional lifetime (the conditional probability of additional life given that the component has lived until t) is the same as the original distribution.

Fit Gamma

The Fit Gamma option estimates the gamma distribution parameters, $\alpha > 0$ and $\sigma > 0$. The parameter α, called alpha in the fitted gamma report, describes shape or curvature. The parameter σ, called sigma, is the scale parameter of the distribution. The data must be greater than zero.

pdf: $\frac{1}{\Gamma(\alpha)\sigma^{\alpha}} x^{\alpha-1}\exp(-x/\sigma)$ for $0 < x;\ 0 < \alpha,\sigma$

$E(x) = \alpha\sigma$

$Var(x) = \alpha\sigma^2$

- The *standard* gamma distribution has σ = 1. Sigma is called the scale parameter because values other than 1 stretch or compress the distribution along the horizontal axis.
- The Chi-square $\chi^2_{(\nu)}$ distribution occurs when σ = 2 and α = ν/2.
- The exponential distribution occurs when α = 1.

The standard gamma density function is strictly decreasing when $\alpha \leq 1$. When $\alpha > 1$, the density function begins at zero, increases to a maximum, and then decreases.

Fit Lognormal

The Fit Lognormal option estimates the parameters μ (scale) and σ (shape) for the two-parameter lognormal distribution. A variable *Y* is lognormal if and only if $X = \ln(Y)$ is normal. The data must be greater than zero.

$$\text{pdf:} \quad \frac{1}{\sigma\sqrt{2\pi}} \frac{\exp\left[\frac{-(\log(x)-\mu)^2}{2\sigma^2}\right]}{x} \quad \text{for} \quad 0 \leq x \;;\; -\infty < \mu < \infty \;;\; 0 < \sigma$$

$$\text{E}(x) = \exp(\mu + \sigma^2/2)$$

$$\text{Var}(x) = \exp(2(\mu + \sigma^2)) - \exp(2\mu + \sigma^2)$$

Fit Weibull

The Weibull distribution has different shapes depending on the values of α (scale) and β (shape). It often provides a good model for estimating the length of life, especially for mechanical devices and in biology.

The pdf for the Weibull distribution is defined as follows:

$$\text{pdf:} \quad \frac{\beta}{\alpha}\left(\frac{x}{\alpha}\right)^{\beta-1} \exp\left[-\left(\frac{x}{\alpha}\right)^{\beta}\right] \quad \text{for} \quad \alpha,\beta > 0; \;\; 0 < x$$

$$\text{E}(x) = \alpha\Gamma\left(1 + \frac{1}{\beta}\right)$$

$$\text{Var}(x) = \alpha^2\left\{\Gamma\left(1 + \frac{2}{\beta}\right) - \Gamma^2\left(1 + \frac{1}{\beta}\right)\right\}$$

where $\Gamma(\cdot)$ is the Gamma function.

Fit Normal 2 Mixture and Fit Normal 3 Mixture

The Fit Normal 2 Mixture and Fit Normal 3 Mixture options fit a mixture of two or three normal distributions. These flexible distributions are capable of fitting bimodal or multi-modal data. A separate mean, standard deviation, and proportion of the whole is estimated for each group. In the following equations, k equals the number of normal distributions in the mixture.

$$\text{pdf:} \quad \sum_{i=1}^{k} \frac{\pi_i}{\sigma_i} \phi\left(\frac{x-\mu_i}{\sigma_i}\right)$$

$$\mathrm{E}(x) = \sum_{i=1}^{k} \pi_i \mu_i$$

$$\mathrm{Var}(x) = \sum_{i=1}^{k} \pi_i(\mu_i^2 + \sigma_i^2) - \left(\sum_{i=1}^{k} \pi_i \mu_i\right)^2$$

where μ_i, σ_i, and π_i are the respective mean, standard deviation, and proportion for the i^{th} group, and $\phi(\cdot)$ is the standard normal pdf.

Fit Johnson

The Fit Johnson option selects and fits the best-fitting distribution from the Johnson system of distributions, which contains three distributions that are all based on a transformed normal distribution. These three distributions are the following:

- Johnson Su, which is unbounded.
- Johnson Sb, which has bounds on both tails. The bounds are defined by parameters that can be estimated.
- Johnson Sl, which is bounded in one tail. The bound is defined by a parameter that can be estimated. The Johnson Sl family contains the family of lognormal distributions.

Only the fit for the selected distribution is reported. Information about selection procedures and parameter estimation for the Johnson distributions can be found in Slifker and Shapiro (1980). The parameter estimation does not use maximum likelihood.

Johnson distributions are popular because of their flexibility. In particular, the Johnson distribution system is noted for its data-fitting capabilities because it supports every possible combination of skewness and kurtosis. However, the SHASH distribution is also very flexible and is recommended over the Johnson distributions.

If Z is a standard normal variate, then the system is defined as follows:

$$Z = \gamma + \delta f(Y)$$

where, for the Johnson Su:

$$f(Y) = \ln\left(Y + \sqrt{1 + Y^2}\right) = \sinh^{-1} Y$$

$$Y = \frac{X - \theta}{\sigma} \qquad -\infty < X < \infty$$

where, for the Johnson Sb:

$$f(Y) = \ln\left(\frac{Y}{1 - Y}\right)$$

$$Y = \frac{X - \theta}{\sigma} \qquad \theta < X < \theta + \sigma$$

and for the Johnson Sl, where $\sigma = \pm 1$.

$$f(Y) = \ln(Y)$$

$$Y = \frac{X - \theta}{\sigma} \qquad \begin{array}{ll} \theta < X < \infty & \text{if } \sigma = 1 \\ -\infty < X < \theta & \text{if } \sigma = -1 \end{array}$$

Johnson Su

pdf: $\frac{\delta}{\sigma}\left[1 + \left(\frac{x - \theta}{\sigma}\right)^2\right]^{-1/2} \phi\left[\gamma + \delta \sinh^{-1}\left(\frac{x - \theta}{\sigma}\right)\right]$ for $-\infty < x, \theta, \gamma < \infty$; $0 < \theta, \delta$

Johnson Sb

pdf: $\phi\left[\gamma + \delta \ln\left(\frac{x - \theta}{\sigma - (x - \theta)}\right)\right]\left(\frac{\delta\sigma}{(x - \theta)(\sigma - (x - \theta))}\right)$ for $\theta < x < \theta + \sigma$; $0 < \sigma$

Johnson Sl

pdf: $\frac{\delta}{|x - \theta|} \phi\left[\gamma + \delta \ln\left(\frac{x - \theta}{\sigma}\right)\right]$ for $\theta < x$ if $\sigma = 1$; $\theta > x$ if $\sigma = -1$

where $\phi(\cdot)$ is the standard normal pdf.

Fit Beta

The beta distribution is useful for modeling the behavior of random variables that are constrained to fall in the interval 0,1. For example, proportions always fall between 0 and 1. The Fit Beta option estimates two shape parameters, $\alpha > 0$ and $\beta > 0$. The beta distribution has values only in the interval 0,1.

$$\text{pdf:} \quad \frac{1}{B(\alpha, \beta)\sigma^{\alpha+\beta-1}} x^{\alpha-1} x^{\beta-1} \qquad \text{for} \quad 0 < x < 1; \quad 0 < \sigma, \alpha, \beta$$

$$E(x) = \sigma \frac{\alpha}{\alpha + \beta}$$

$$\text{Var}(x) = \frac{\sigma^2 \alpha \beta}{(\alpha + \beta)^2 (\alpha + \beta + 1)}$$

where $B(\cdot)$ is the Beta function.

Fit All

In the Compare Distributions report, the Distribution list is sorted by AICc in ascending order. Use the check boxes to show or hide a fit report and overlay curve for the selected distribution.

The formulas for AICc and BIC are defined as follows:

$$\text{AICc} = -2\text{logL} + 2k + \frac{2k(k+1)}{n-(k+1)}$$

$$\text{BIC} = -2\text{logL} + k\ln(n)$$

where:

- logL is the log-likelihood.
- n is the sample size.
- k is the number of parameters.

The AICc Weight column shows normalized AICc values that sum to one. The AICc weight can be interpreted as the probability that a particular distribution is the true distribution given that one of the fitted distributions is the truth. Therefore, the distribution with the AICc weight closest to one is the better fit. The AICc weights are calculated using only nonmissing AICc values:

AICcWeight = exp[-0.5(AICc-min(AICc))] / sum(exp[-0.5(AICc-min(AICc))])

where min(AICc) is the smallest AICc value among the fitted distributions.

For more information about the measures in the Compare Distributions report, see *Fitting Linear Models*.

Discrete Fit Distributions

This section contains statistical details for the options in the Discrete Fit menu.

Fit Poisson

The Poisson distribution has a single scale parameter $\lambda > 0$.

pmf: $\frac{e^{-\lambda}\lambda^x}{x!}$ for $0 \le \lambda < \infty; \quad x = 0,1,2,...$

$E(x) = \lambda$

$Var(x) = \lambda$

Since the Poisson distribution is a discrete distribution, the overlaid curve is a step function, with jumps that occur at every integer.

Fit Negative Binomial

The negative binomial distribution is useful for modeling the number of successes before a specified number of failures. The following parameterization contains mean parameter λ and dispersion parameter σ.

pmf: $\frac{\Gamma[x + (1/\sigma)]}{\Gamma[x+1]\Gamma[1/\sigma]}\left[\frac{(\lambda\sigma)^x}{(1+\lambda\sigma)^{x+(1/\sigma)}}\right], \; x = 0, 1, 2, \ldots$

$E(x) = \lambda$

$Var(x) = \lambda + \sigma\lambda^2$

where $\Gamma(\cdot)$ is the Gamma function.

Relationship between Negative Binomial and Gamma Poisson Distributions

The negative binomial distribution is equivalent to the Gamma Poisson distribution. The Gamma Poisson distribution is useful when the data are a combination of several Poisson(μ) distributions and each Poisson(μ) distribution has a different μ.

The Gamma Poisson distribution results from assuming that $x|\mu$ follows a Poisson distribution and μ follows a Gamma(α,τ). The Gamma Poisson has parameters $\lambda = \alpha\tau$ and $\sigma = \tau+1$. The parameter σ is a dispersion parameter. If $\sigma > 1$, there is over dispersion, meaning there is more variation in x than explained by the Poisson alone. If $\sigma = 1$, x reduces to Poisson(λ).

$$\text{pmf:} \quad \frac{\Gamma\left(x+\frac{\lambda}{\sigma-1}\right)}{\Gamma(x+1)\Gamma\left(\frac{\lambda}{\sigma-1}\right)}\left(\frac{\sigma-1}{\sigma}\right)^{x}\sigma^{-\frac{\lambda}{\sigma-1}} \qquad \text{for} \quad 0<\lambda; \quad 1\le\sigma; \quad x=0,1,2,\ldots$$

$E(x) = \lambda$

$Var(x) = \lambda\sigma$

where $\Gamma(\cdot)$ is the Gamma function.

The Gamma Poisson is equivalent to a Negative Binomial with $\sigma_{negbin} = (\sigma_{gp} - 1) / \lambda_{gp}$.

Run demoGammaPoisson.jsl in the JMP Samples/Scripts folder to compare a Gamma Poisson distribution with parameters λ and σ to a Poisson distribution with parameter λ.

Fit ZI Poisson

The zero-inflated (ZI) Poisson distribution has scale parameter $\lambda > 0$ and zero-inflation parameter π.

$$\text{pmf:} \quad \begin{cases} \pi + (1-\pi)\exp[-\lambda], \text{ for } x = 0 \\ (1-\pi)\dfrac{\lambda^{x}}{x!}\exp[-\lambda], \text{ for } x = 1, 2, \ldots \end{cases}$$

$E(x) = (1 - \pi)\lambda$

$Var(x) = \lambda(1 - \pi)(1 + \lambda\pi)$

Fit ZI Negative Binomial

The zero-inflated (ZI) negative binomial distribution has scale parameter $\lambda > 0$, dispersion parameter $\sigma > 0$, and zero-inflation parameter π.

$$\text{pmf:} \quad \begin{cases} \pi + (1-\pi)(1+\lambda\sigma)^{-(1/\sigma)}, \text{ for } x = 0 \\ (1-\pi)\dfrac{\Gamma[x+(1/\sigma)]}{\Gamma[x+1]\Gamma[1/\sigma]}\left[\dfrac{(\lambda\sigma)^{x}}{(1+\lambda\sigma)^{x+(1/\sigma)}}\right], \text{ for } x = 1, 2, \ldots \end{cases}$$

$E(x) = (1 - \pi)\lambda$

$Var(x) = \lambda(1 - \pi)[1 + \lambda(\sigma + \pi)]$

Fit Binomial

The Fit Binomial option accepts data in two formats: a constant sample size, or a column containing sample sizes.

pmf: $\binom{n}{x} p^x (1-p)^{n-x}$ for $0 \le p \le 1$; $x = 0,1,2,\ldots,n$

$E(x) = np$

$Var(x) = np(1\text{-}p)$

where n is the number of independent trials.

Note: The confidence interval for the binomial parameter is a Score interval. See Agresti and Coull (1998).

Fit Beta Binomial

The beta binomial distribution is useful when the data are a combination of several Binomial(p) distributions and each Binomial(p) distribution has a different p. One example is the overall number of defects combined from multiple manufacturing lines, when the mean number of defects (p) varies between the lines.

The beta binomial distribution results from assuming that $x \mid \pi$ follows a Binomial(n,π) distribution and π follows a Beta(α,β). The beta binomial has parameters $p = \alpha/(\alpha+\beta)$ and $\delta = 1/(\alpha+\beta+1)$. The parameter δ is a dispersion parameter. When $\delta > 0$, there is over dispersion, meaning there is more variation in x than explained by the Binomial alone. When $\delta < 0$, there is under dispersion. When $\delta = 0$, x is distributed as Binomial(n,p). The beta binomial exists only when $n \ge 2$.

pmf: $$\binom{n}{x} \frac{\Gamma\left(\frac{1}{\delta}-1\right)\Gamma\left[x + p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[n - x + (1-p)\left(\frac{1}{\delta}-1\right)\right]}{\Gamma\left[p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[(1-p)\left(\frac{1}{\delta}-1\right)\right]\Gamma\left(n + \frac{1}{\delta}-1\right)}$$

for $0 \le p \le 1$; $\max\left(-\frac{p}{n-p-1}, -\frac{1-p}{n-2+p}\right) \le \delta \le 1$; $x = 0,1,2,\ldots,n$

$E(x) = np$

$Var(x) = np(1\text{-}p)[1+(n\text{-}1)\delta]$

where $\Gamma(\cdot)$ is the Gamma function.

Remember that $x \mid \pi \sim$ Binomial(n,π), while $\pi \sim$ Beta(α,β). The parameters $p = \alpha/(\alpha+\beta)$ and $\delta = 1/(\alpha+\beta+1)$ are estimated by the platform. To obtain estimates of α and β, use the following formulas:

$$\hat{\alpha} = \hat{p}\left(\frac{1-\hat{\delta}}{\hat{\delta}}\right)$$

$$\hat{\beta} = (1-\hat{p})\left(\frac{1-\hat{\delta}}{\hat{\delta}}\right)$$

If the estimate of δ is 0, the formulas do not work. In that case, the beta binomial has reduced to the Binomial(n,p), and $\hat{p}$ is the estimate of p.

The confidence intervals for the beta binomial parameters are profile likelihood intervals.

Run demoBetaBinomial.jsl in the JMP Samples/Scripts folder to compare a beta binomial distribution with dispersion parameter δ to a Binomial distribution with parameters p and n = 20.

Details for the Legacy Distribution Fitters

Some features of distribution fitting have been updated in JMP 15. This section contains details of the older features from previous JMP releases that have been retained for compatibility purposes. These features are available by selecting **Continuous Fit > Enable Legacy Fitters** in the red triangle menu for a variable.

- "Fit Distributions Options (Legacy)"
- "Statistical Details for Continuous Fit Distributions (Legacy)"
- "Statistical Details for Discrete Fit Distributions (Legacy)"
- "Statistical Details for Fitted Quantiles (Legacy)"
- "Statistical Details for Fit Distribution Options (Legacy)"

Fit Distributions Options (Legacy)

Use the Continuous Fit or Discrete Fit options to fit a distribution to a continuous variable.

Note: Some features of distribution fitting have been updated in JMP 15. This section contains details of the older features from previous JMP releases that have been retained for compatibility purposes. These features are available by selecting **Continuous Fit > Enable Legacy Fitters** in the red triangle menu for a variable.

A curve is overlaid on the histogram, and a Parameter Estimates report is added to the report window. A red triangle menu contains additional options. See "Fit Distribution Options (Legacy)".

Note: The Life Distribution platform also contains options for distribution fitting that might use different parameterizations and allow for censoring. See *Reliability and Survival Methods*.

Continuous Fit (Legacy)

This section describes the distributions in the Legacy Fitters submenu that differ from the corresponding distributions in the updated Continuous Fit options.

- The Weibull distribution, Weibull with threshold distribution, and Extreme Value distribution often provide a good model for estimating the length of life, especially for mechanical devices and in biology.
- The Gamma distribution is bound by zero and has a flexible shape.
- The Beta distribution is useful for modeling the behavior of random variables that are constrained to fall in the interval 0,1. For example, proportions always fall between 0 and 1.
- The Smooth Curve distribution fits a smooth curve using nonparametric density estimation (kernel density estimation). The smooth curve is overlaid on the histogram and a slider appears beneath the plot. Control the amount of smoothing by changing the kernel standard deviation with the slider. The initial Kernel Std estimate is calculated from the standard deviation of the data.
- The Johnson Su, Johnson Sb, and Johnson Sl Distributions are useful for its data-fitting capabilities because it supports every possible combination of skewness and kurtosis.
- The Generalized Log (Glog) distribution is useful for fitting data that are rarely normally distributed and often have non-constant variance, like biological assay data.

Comparing All Distributions

The **All** option fits all applicable continuous distributions to a variable. The Compare Distributions report contains statistics about each fitted distribution. Use the check boxes to show or hide a fit report and overlay curve for the selected distribution. By default, the best fit distribution is selected.

The Show Distribution list is sorted by AICc in ascending order.

If your variable contains negative values, the Show Distribution list does not include those distributions that require data with positive values. Only continuous distributions are fitted by this command. Distributions with threshold parameters, like Beta and Johnson Sb, are not included in the list of possible distributions.

Related Information

For statistical details, see the following sections:

- "Statistical Details for Continuous Fit Distributions (Legacy)"
- "Statistical Details for Fitted Quantiles (Legacy)"
- "Fit Distribution Options (Legacy)"

Discrete Fit (Legacy)

The Discrete Fit option is available when all data values are integers. Use the Discrete Fit options to fit a distribution (such as Poisson or Binomial) to a discrete variable. The following distributions are available:

- Poisson
- Gamma Poisson
- Binomial
- Beta Binomial

Related Information

For statistical details, see the following sections:

- "Statistical Details for Discrete Fit Distributions (Legacy)"
- "Statistical Details for Fitted Quantiles (Legacy)"
- "Fit Distribution Options (Legacy)"

Fit Distribution Options (Legacy)

Each fitted distribution report has a red triangle menu that contains additional options.

Diagnostic Plot Creates a quantile or a probability plot. See "Diagnostic Plot".

Density Curve Uses the estimated parameters of the distribution to overlay a density curve on the histogram.

Goodness of Fit Computes the goodness of fit test for the fitted distribution. See "Goodness of Fit".

Fix Parameters Enables you to fix parameters and re-estimate the non-fixed parameters. An Adequacy LR (likelihood ratio) Test report also appears, which tests your new parameters to determine whether they fit the data.

Quantiles Returns the unscaled and uncentered quantiles for the specific lower probability values that you specify.

Set Spec Limits for K Sigma Use this option when you do not know the specification limits for a process and you want to use its distribution as a guideline for setting specification limits.

Usually, specification limits are derived using engineering considerations. If there are no engineering considerations, and if the data are from a well behaved process, then quantiles from a fitted distribution are often used to help set specification limits. See "Set Spec Limits for K Sigma".

Spec Limits Computes generalizations of the standard capability indices, based on the specification limits and target you specify. See "Spec Limits".

Save Fitted Quantiles Saves the fitted quantile values as a new column in the current data table. See "Statistical Details for Fitted Quantiles (Legacy)".

Save Density Formula Creates a new column in the current data table that contains fitted values that have been computed by the density formula. The density formula uses the estimated parameter values.

Save Spec Limits Saves the specification limits as a column property.

Save Transformed Creates a new column and saves a formula. The formula can transform the column to normality using the fitted distribution. This option is available only when one of the Johnson distributions, the Glog distribution, or the SHASH distribution is fit.

Remove Fit Removes the distribution fit from the report window.

Diagnostic Plot

The **Diagnostic Plot** option creates a quantile or a probability plot. Depending on the fitted distribution, the plot is in one of the following four formats.

The Fitted Quantiles versus the Data

- Weibull with threshold

- Gamma
- Beta
- Poisson
- GammaPoisson
- Binomial
- BetaBinomial

The Fitted Probability versus the Data

- Normal
- Normal Mixtures
- Exponential

The Fitted Probability versus the Data on Log Scale

- Weibull
- LogNormal
- Extreme Value

The Fitted Probability versus the Standard Normal Quantile

- SHASH
- Johnson Sl
- Johnson Sb
- Johnson Su
- Glog

The following options are available in the Diagnostic Plot red triangle menu:

Rotate Reverses the x- and y-axes.

Confidence Limits Draws Lilliefors 95% confidence limits for the Normal Quantile plot, and 95% equal precision bands with $a = 0.001$ and $b = 0.99$ for all other quantile plots (Meeker and Escobar 1998).

Line of Fit Draws the straight diagonal reference line. If a variable fits the selected distribution, the values fall approximately on the reference line.

Median Reference Line Draws a horizontal line at the median of the response.

Goodness of Fit

The **Goodness of Fit** option computes the goodness of fit test for the fitted distribution. The goodness of fit tests are not Chi-square tests, but are EDF (Empirical Distribution Function) tests. EDF tests offer advantages over the Chi-square tests, including improved power and invariance with respect to histogram midpoints.

- For Normal distributions, the Shapiro-Wilk test for normality is reported when the sample size is less than or equal to 2000. The KSL test is computed for samples that are greater than 2000.
- For discrete distributions that have sample sizes less than or equal to 30, the Goodness of Fit test is formed using two one-sided exact Kolmogorov tests combined to form a near-exact test. See Conover (1972). For sample sizes greater than 30, a Pearson Chi-squared goodness of fit test is performed.

Related Information

- For statistical details, see "Fit Distribution Options (Legacy)".

Spec Limits

The **Spec Limits** option opens a window that enables you to enter specification limits and a target. Then generalizations of the standard capability indices are computed. Note that for the normal distribution, 3σ is both the distance from the lower 0.135 percentile to median (or mean) and the distance from the median (or mean) to the upper 99.865 percentile. These percentiles are estimated from the fitted distribution, and the appropriate percentile-to-median distances are substituted for 3σ in the standard formulas.

Related Information

- For statistical details, see "Fit Distribution Options (Legacy)".

Statistical Details for Continuous Fit Distributions (Legacy)

This section contains statistical details for the options in the Continuous Fit menu.

Note: Some features of distribution fitting have been updated in JMP 15. This section contains details of the older features from previous JMP releases that have been retained for compatibility purposes. These features are available by selecting **Continuous Fit > Enable Legacy Fitters** in the red triangle menu for a variable.

Normal

For more information about the normal distribution fit, see "Fit Normal".

LogNormal

For more information about the lognormal distribution fit, see "Fit Lognormal".

Weibull, Weibull with Threshold, and Extreme Value

The Weibull distribution has different shapes depending on the values of α (scale) and β (shape). It often provides a good model for estimating the length of life, especially for mechanical devices and in biology.

The pdf for the Weibull and Weibull with Threshold distributions is defined as follows:

$$\text{pdf:} \quad \frac{\beta}{\alpha}\left(\frac{x-\theta}{\alpha}\right)^{\beta-1}\exp\left[-\left(\frac{x-\theta}{\alpha}\right)^{\beta}\right] \quad \text{for} \quad \alpha,\beta>0; \quad \theta<x$$

$$\mathrm{E}(x) = \theta + \alpha\Gamma\left(1+\frac{1}{\beta}\right)$$

$$\mathrm{Var}(x) = \alpha^2\left\{\Gamma\left(1+\frac{2}{\beta}\right)-\Gamma^2\left(1+\frac{1}{\beta}\right)\right\}$$

where $\Gamma(\cdot)$ is the Gamma function.

The **Weibull** option sets the threshold parameter (θ) to zero. The **Weibull with Threshold** option, estimates the threshold parameter (θ) using the value of the minimum observation and estimates α and β using the rest of the observations. If you know what the threshold should be, set it by using the **Fix Parameters** option. See "Fit Distribution Options".

Note: The Distribution platform uses a different estimation technique for the threshold parameter in the Weibull with Threshold distribution than does the Life Distribution platform. The Life Distribution estimation method is recommended for fitting this distribution. See *Reliability and Survival Methods*.

The Extreme Value distribution is equivalent to a two-parameter Weibull (α, β) distribution re-parameterized as $\delta = 1 / \beta$ and $\lambda = \ln(\alpha)$.

Exponential

For more information about the exponential distribution fit, see "Fit Exponential".

Gamma

The **Gamma** fitting option estimates the gamma distribution parameters, $\alpha > 0$ and $\sigma > 0$. The parameter α, called alpha in the fitted gamma report, describes shape or curvature. The parameter σ, called sigma, is the scale parameter of the distribution. A third parameter, θ, called the Threshold, is the lower endpoint parameter. It is set to zero by default, unless there are negative values. You can also set its value by using the **Fix Parameters** option. See "Fit Distribution Options".

$$\text{pdf:} \quad \frac{1}{\Gamma(\alpha)\sigma^{\alpha}}(x-\theta)^{\alpha-1}\exp(-(x-\theta)/\sigma) \qquad \text{for} \quad 0 \le x; \quad 0 < \alpha,\sigma$$

$E(x) = \alpha\sigma + \theta$

$Var(x) = \alpha\sigma^2$

- The *standard* gamma distribution has $\sigma = 1$. Sigma is called the scale parameter because values other than 1 stretch or compress the distribution along the horizontal axis.
- The Chi-square $\chi^2_{(\nu)}$ distribution occurs when $\sigma = 2$, $\alpha = \nu/2$, and $\theta = 0$.
- The exponential distribution is the family of gamma curves that occur when $\alpha = 1$ and $\theta = 0$.

The standard gamma density function is strictly decreasing when $\alpha \le 1$. When $\alpha > 1$, the density function begins at zero, increases to a maximum, and then decreases.

Beta

The standard beta distribution is useful for modeling the behavior of random variables that are constrained to fall in the interval 0,1. For example, proportions always fall between 0 and 1. The **Beta** fitting option estimates two shape parameters, $\alpha > 0$ and $\beta > 0$, and two threshold parameters, θ and σ. The lower threshold is represented as θ, and the upper threshold is represented as $\theta + \sigma$. The beta distribution has values only in the interval $\theta \le x \le (\theta + \sigma)$. The θ is estimated by the minimum value, and σ is estimated by the range. The standard beta distribution occurs when $\theta = 0$ and $\sigma = 1$.

Set parameters to fixed values by using the **Fix Parameters** option. The upper threshold must be greater than or equal to the maximum data value, and the lower threshold must be less than or equal to the minimum data value. For more information about the Fix Parameters option, see "Fit Distribution Options".

$$\text{pdf:} \quad \frac{1}{B(\alpha,\beta)\sigma^{\alpha+\beta-1}}(x-\theta)^{\alpha-1}(\theta+\sigma-x)^{\beta-1} \qquad \text{for} \quad \theta \le x \le \theta+\sigma; \quad 0 < \sigma,\alpha,\beta$$

$$E(x) = \theta + \sigma\frac{\alpha}{\alpha + \beta}$$

$$Var(x) = \frac{\sigma^2\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}$$

where $B(\cdot)$ is the Beta function.

Normal Mixtures

For more information about the normal mixtures distribution fits, see "Fit Normal 2 Mixture and Fit Normal 3 Mixture".

Smooth Curve

The **Smooth Curve** option fits a smooth curve using nonparametric density estimation (kernel density estimation). The smooth curve is overlaid on the histogram and a slider appears beneath the plot. Control the amount of smoothing by changing the kernel standard deviation with the slider. The initial Kernel Std estimate is calculated from the standard deviation of the data.

SHASH

For more information about the SHASH distribution fit, see "Fit SHASH".

Johnson Su, Johnson Sb, Johnson Sl

The Johnson system of distributions contains three distributions that are all based on a transformed normal distribution. These three distributions are the following:

- Johnson Su, which is unbounded.
- Johnson Sb, which has bounds on both tails. The bounds are defined by parameters that can be estimated.
- Johnson Sl, which is bounded in one tail. The bound is defined by a parameter that can be estimated. The Johnson Sl family contains the family of lognormal distributions.

The S refers to system, the subscript of the range. Although we implement a different method in the legacy fitters, information about selection criteria for a particular Johnson system can be found in Slifker and Shapiro (1980).

Johnson distributions are popular because of their flexibility. In particular, the Johnson distribution system is noted for its data-fitting capabilities because it supports every possible combination of skewness and kurtosis.

If Z is a standard normal variate, then the system is defined as follows:

$$Z = \gamma + \delta f(Y)$$

where, for the Johnson Su:

$$f(Y) = \ln\left(Y + \sqrt{1 + Y^2}\right) = \sinh^{-1} Y$$

$$Y = \frac{X - \theta}{\sigma} \qquad -\infty < X < \infty$$

where, for the Johnson Sb:

$$f(Y) = \ln\left(\frac{Y}{1 - Y}\right)$$

$$Y = \frac{X - \theta}{\sigma} \qquad \theta < X < \theta + \sigma$$

and for the Johnson Sl, where $\sigma = \pm 1$.

$$f(Y) = \ln(Y)$$

$$Y = \frac{X - \theta}{\sigma} \qquad \begin{array}{ll} \theta < X < \infty & \text{if } \sigma = 1 \\ -\infty < X < \theta & \text{if } \sigma = -1 \end{array}$$

Johnson Su

pdf: $\frac{\delta}{\sigma}\left[1 + \left(\frac{x - \theta}{\sigma}\right)^2\right]^{-1/2} \phi\left[\gamma + \delta \sinh^{-1}\left(\frac{x - \theta}{\sigma}\right)\right]$ for $-\infty < x, \theta, \gamma < \infty$; $0 < \theta, \delta$

Johnson Sb

pdf: $\phi\left[\gamma + \delta \ln\left(\frac{x - \theta}{\sigma - (x - \theta)}\right)\right]\left(\frac{\delta\sigma}{(x - \theta)(\sigma - (x - \theta))}\right)$ for $\theta < x < \theta + \sigma$; $0 < \sigma$

Johnson Sl

pdf: $\frac{\delta}{|x - \theta|} \phi\left[\gamma + \delta \ln\left(\frac{x - \theta}{\sigma}\right)\right]$ for $\theta < x$ if $\sigma = 1$; $\theta > x$ if $\sigma = -1$

where $\phi(\cdot)$is the standard normal pdf.

Note the following:

- Parameter estimates might be different between machines due to the order of operations and machine precision.
- The parameter confidence intervals are hidden in the default report. Parameter confidence intervals are not very meaningful for Johnson distributions, because they are transformations to normality. To show parameter confidence intervals, right-click in the report and select **Columns > Lower 95%** and **Upper 95%**.

Generalized Log (Glog)

This distribution is useful for fitting data that are rarely normally distributed and often have non-constant variance, like biological assay data. The Glog distribution is described with the parameters μ (location), σ (scale), and λ (shape).

$$\text{pdf:} \quad \phi\left\{\frac{1}{\sigma}\left[\log\left(\frac{x+\sqrt{x^2+\lambda^2}}{2}\right)-\mu\right]\right\}\frac{x+\sqrt{x^2+\lambda^2}}{\sigma(x^2+\lambda^2+x\sqrt{x^2+\lambda^2})}$$

for $0 \le \lambda;\ \ 0 < \sigma;\ \ -\infty < \mu < \infty$

The Glog distribution is a transformation to normality, and comes from the following relationship:

$$\text{If } z = \frac{1}{\sigma}\left[\log\left(\frac{x+\sqrt{x^2+\lambda^2}}{2}\right)-\mu\right] \sim N(0,1), \text{ then } x \sim \text{Glog}(\mu,\sigma,\lambda).$$

When $\lambda = 0$, the Glog reduces to the LogNormal (μ,σ).

Note: The parameter confidence intervals are hidden in the default report. Parameter confidence intervals are not very meaningful for the GLog distribution, because it is a transformation to normality. To show parameter confidence intervals, right-click in the report and select **Columns > Lower 95%** and **Upper 95%**.

All

In the Compare Distributions report, the Distribution list is sorted by AICc in ascending order.

The AICc is defined as follows:

$$\text{AICc} = -2\text{logL} + 2\nu + \frac{2\nu(\nu+1)}{n-(\nu+1)}$$

where:

- logL is the log-likelihood.
- n is the sample size.
- ν is the number of parameters.

If the column contains negative values, the Distribution list does not include those distributions that require data with positive values. Only continuous distributions are listed. Distributions with threshold parameters, such as Beta and Johnson Sb, are not included in the list of possible distributions.

Statistical Details for Discrete Fit Distributions (Legacy)

This section contains statistical details for the options in the Discrete Fit menu.

Note: Some features of distribution fitting have been updated in JMP 15. This section contains details of the older features from previous JMP releases that have been retained for compatibility purposes. These features are available by selecting **Continuous Fit > Enable Legacy Fitters** in the red triangle menu for a variable.

Poisson

For more information about the Poisson distribution fit, see "Fit Poisson".

Gamma Poisson

This distribution is useful when the data are a combination of several Poisson(μ) distributions and each Poisson(μ) distribution has a different μ. One example is the overall number of accidents combined from multiple intersections, when the mean number of accidents (μ) varies between the intersections.

The Gamma Poisson distribution results from assuming that x|μ follows a Poisson distribution and μ follows a Gamma(α,τ). The Gamma Poisson has parameters $\lambda = \alpha\tau$ and $\sigma = \tau+1$. The parameter σ is a dispersion parameter. If $\sigma > 1$, there is over dispersion, meaning there is more variation in x than explained by the Poisson alone. If $\sigma = 1$, x reduces to Poisson(λ).

pmf: $$\frac{\Gamma\left(x+\frac{\lambda}{\sigma-1}\right)}{\Gamma(x+1)\Gamma\left(\frac{\lambda}{\sigma-1}\right)}\left(\frac{\sigma-1}{\sigma}\right)^{x}\sigma^{-\frac{\lambda}{\sigma-1}} \quad \text{for} \quad 0<\lambda;\quad 1\le\sigma;\quad x=0,1,2,\ldots$$

$E(x) = \lambda$

$\text{Var}(x) = \lambda\sigma$

where $\Gamma(\cdot)$ is the Gamma function.

Remember that $x \mid \mu \sim \text{Poisson}(\mu)$, while $\mu \sim \text{Gamma}(\alpha,\tau)$. The platform estimates $\lambda = \alpha\tau$ and $\sigma = \tau+1$. To obtain estimates for α and τ, use the following formulas:

$$\hat{\tau} = \hat{\sigma} - 1$$

$$\hat{\alpha} = \frac{\hat{\lambda}}{\hat{\tau}}$$

If the estimate of σ is 1, the formulas do not work. In that case, the Gamma Poisson has reduced to the Poisson(λ), and $\hat{\lambda}$ is the estimate of λ.

If the estimate for α is an integer, the Gamma Poisson is equivalent to a Negative Binomial with the following pmf:

$$p(y) = \binom{y+r-1}{y} p^r (1-p)^y \qquad \text{for} \quad 0 \le y$$

with $r = \alpha$ and $(1\text{-}p)/p = \tau$.

Run demoGammaPoisson.jsl in the JMP Samples/Scripts folder to compare a Gamma Poisson distribution with parameters λ and σ to a Poisson distribution with parameter λ.

Binomial

For more information about the binomial distribution fit, see "Fit Binomial".

Beta Binomial

For more information about the beta binomial distribution fit, see "Fit Beta Binomial".

Statistical Details for Fitted Quantiles (Legacy)

Note: Some features of distribution fitting have been updated in JMP 15. This section contains details of the older features from previous JMP releases that have been retained for compatibility purposes. These features are available by selecting **Continuous Fit > Enable Legacy Fitters** in the red triangle menu for a variable.

The fitted quantiles in the Diagnostic Plot and the fitted quantiles saved with the **Save Fitted Quantiles** command are formed using the following method:

1. The data are sorted and ranked. Ties are assigned different ranks.

2. Compute the $p_{[i]} = rank_{[i]}/(n+1)$.
3. Compute the $quantile_{[i]} = Quantile_d(p_{[i]})$ where $Quantile_d$ is the quantile function for the specific fitted distribution, and $i = 1,2,...,n$.

Statistical Details for Fit Distribution Options (Legacy)

This section describes Goodness of Fit tests for fitting distributions and statistical details for specification limits pertaining to fitted distributions.

Note: Some features of distribution fitting have been updated in JMP 15. This section contains details of the older features from previous JMP releases that have been retained for compatibility purposes. These features are available by selecting **Continuous Fit > Enable Legacy Fitters** in the red triangle menu for a variable.

Goodness of Fit

Table 3.2 Descriptions of JMP Goodness of Fit Tests

Distribution	Parameters	Goodness of Fit Test
Normal[a]	μ and σ are unknown	Shapiro-Wilk (for n ≤ 2000) Kolmogorov-Smirnov-Lillefors (for n > 2000)
	μ and σ are both known	Kolmogorov-Smirnov-Lillefors
	either μ or σ is known	(none)
LogNormal	μ and σ are known or unknown	Kolmogorov's *D*
Weibull	α and β known or unknown	Cramér-von Mises W^2
Weibull with threshold	α, β and θ known or unknown	Cramér-von Mises W^2
Extreme Value	α and β known or unknown	Cramér-von Mises W^2
Exponential	σ is known or unknown	Kolmogorov's *D*
Gamma	α and σ are known	Cramér-von Mises W^2
	either α or σ is unknown	(none)

Table 3.2 Descriptions of JMP Goodness of Fit Tests *(Continued)*

Distribution	Parameters	Goodness of Fit Test
Beta	α and β are known	Kolmogorov's *D*
	either α or β is unknown	(none)
Binomial	ρ is known or unknown and *n* is known	Kolmogorov's *D* (for $n \leq 30$) Pearson χ^2 (for $n > 30$)
Beta Binomial	ρ and δ known or unknown	Kolmogorov's *D* (for $n \leq 30$) Pearson χ^2 (for $n > 30$)
Poisson	λ known or unknown	Kolmogorov's *D* (for $n \leq 30$) Pearson χ^2 (for $n > 30$)
Gamma Poisson	λ or σ known or unknown	Kolmogorov's *D* (for $n \leq 30$) Pearson χ^2 (for $n > 30$)

a. For the three Johnson distributions and the Glog distribution, the data are transformed to Normal, then the appropriate test of normality is performed.

Set Spec Limits for K Sigma

Type a K value and select one-sided or two-sided for your process capability analysis. Tail probabilities corresponding to K standard deviations are computed from the Normal distribution. The probabilities are converted to quantiles for the specific distribution that you have fitted. The resulting quantiles are used for specification limits in the process capability analysis. This option is similar to the **Quantiles** option, but you provide K instead of probabilities. K corresponds to the number of standard deviations that the specification limits are away from the mean.

For example, for a Normal distribution, where K = 3, the 3 standard deviations below and above the mean correspond to the 0.00135^{th} quantile and 0.99865^{th} quantile, respectively. The lower specification limit is set at the 0.00135^{th} quantile, and the upper specification limit is set at the 0.99865^{th} quantile of the fitted distribution. A process capability analysis is returned based on those specification limits.

Chapter 4

Introduction to Fit Y by X

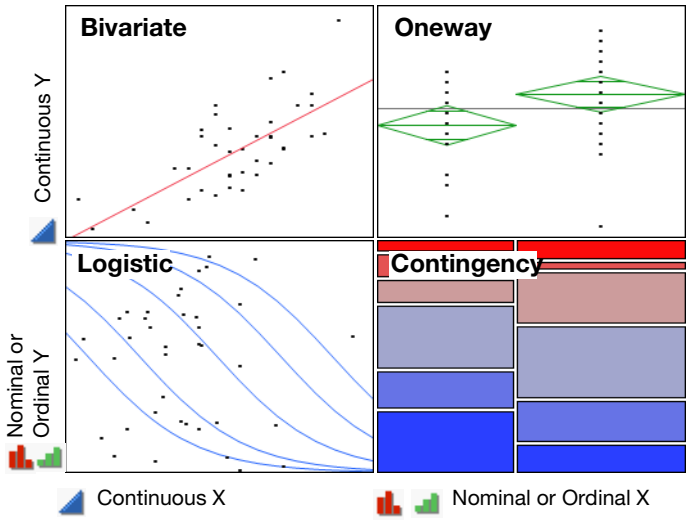
Examine Relationships between Two Variables

Use the Fit Y by X platform to analyze pairs of variables. You can do this using scatter plots, linear regression, ANOVA, multiple comparisons, logistic regression, contingency tables and much more. Specific analyses depend on the modeling types of the variables.

Fit Y by X launches one of four platforms:

- Use Bivariate to analyze a continuous Y and a continuous X
- Use One-way analysis, including ANOVA, to analyze a continuous Y and a categorical X
- Use Logistic regression to analyze a categorical Y and a continuous X
- Use Contingency to analyze a categorical Y and a categorical X

Figure 4.1 Examples of Four Types of Analyses



Contents

Overview of the Fit Y by X Platform 113
Launch the Fit Y by X Platform 113
Launch Specific Analyses from the JMP Starter Window 114

Overview of the Fit Y by X Platform

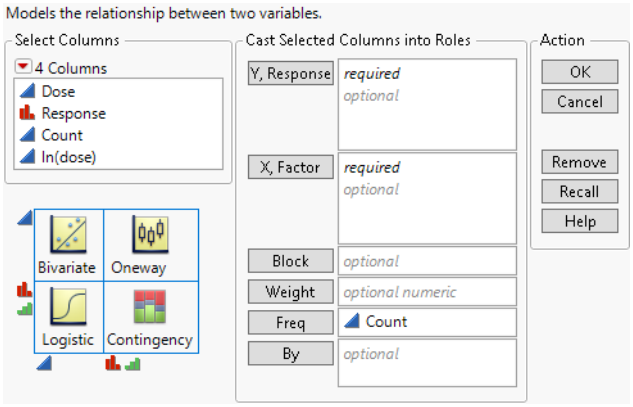
The Fit Y by X platform is a collection of four specific platforms (or types of analyses).

Specific Platform	Modeling Types	Description
Bivariate	Continuous Y by continuous X	Analyzes the relationship between two continuous variables. See "Bivariate Analysis".
Oneway	Continuous Y by nominal or ordinal X	Analyzes how the distribution of a continuous Y variable differs across groups defined by a categorical X variable. See "Oneway Analysis".
Logistic	Nominal or ordinal Y by continuous X	Fits the probabilities for response categories to a continuous X predictor. See "Logistic Analysis".
Contingency	Nominal or ordinal Y by nominal or ordinal X	Analyzes the distribution of a categorical response variable Y as conditioned by the values of a categorical X factor. See "Contingency Analysis".

Launch the Fit Y by X Platform

Launch the Fit Y by X platform by selecting **Analyze > Fit Y by X**.

Figure 4.2 The Fit Y by X Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Bivariate, Oneway, Logistic, Contingency This grid shows which analysis results from the different combinations of data types. Once you have assigned your columns, the applicable platform appears as a label above the grid.

Block (Optional. Applicable only for Oneway and Contingency.):

- For the Oneway platform, specifying a Block variable identifies a second factor, which forms a two-way analysis without interaction. In the Oneway plot, the values of the Y variable are centered by the Block variable. If there are equal counts in each block by group cell, the available Oneway platform options are adapted to the two-way analysis setting. If there are unequal counts in the block by group cells, a Fit Model report is shown below the Oneway plot and the Oneway platform options are not available.
- For the Contingency platform, specifying a Block variable identifies a second factor and performs a Cochran-Mantel-Haenszel test.

For more information about launch windows, see *Using JMP*.

Launch Specific Analyses from the JMP Starter Window

From the JMP Starter window, you can launch a specific analysis (**Bivariate**, **Oneway**, **Logistic**, or **Contingency**). If you select this option, specify the correct modeling types (Y and X variables) for the analysis (Table 4.1).

To launch a specific analysis from the JMP Starter Window, click the **Basic** category, and select a specific analysis.

Most of the platform launch options are the same. However, the naming for some of the Y and X platform buttons is customized for the specific analysis that you are performing.

Table 4.1 Platforms and Buttons

Platform or Analysis	Y Button	X Button
Fit Y by X	Y, Response	X, Factor
Bivariate	Y, Response	X, Regressor
Oneway	Y, Response	X, Grouping
Logistic	Y, Categorical Response	X, Continuous Regressor
Contingency	Y, Response Category	X, Grouping Category

Chapter 5

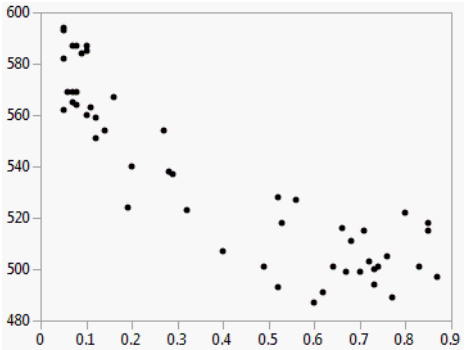
Bivariate Analysis

Examine Relationships between Two Continuous Variables

Use the Bivariate platform to investigate the relationship between two continuous variables. Interactively add a simple linear regression line, polynomial regression curves, or smoothers to a scatter plot. Using the scatterplot, you can see at a glance the relationship between the two variables.

The Bivariate platform is the *continuous by continuous* personality of the Fit Y by X platform. The word bivariate simply means involving two variables instead of one (univariate) or many (multivariate).

Figure 5.1 Example of Bivariate Analysis



Contents

Example of Bivariate Analysis . . . 118
Launch the Bivariate Platform . . . 118
The Bivariate Plot . . . 119
Fitting Options . . . 120
Fitting Options . . . 121
Fitting Option Categories . . . 123
Fit the Same Option Multiple Times . . . 123
Histogram Borders . . . 124
Fit Mean . . . 125
Fit Mean Report . . . 125
Fit Line and Fit Polynomial . . . 126
Linear Fit and Polynomial Fit Reports . . . 126
Fit Special . . . 132
Fit Special Reports and Menus . . . 133
Flexible . . . 134
Fit Spline . . . 134
Kernel Smoother . . . 135
Fit Each Value . . . 136
Fit Orthogonal . . . 137
Orthogonal Fit Ratio Report . . . 137
Robust . . . 138
Fit Robust . . . 138
Fit Cauchy . . . 139
Density Ellipse . . . 139
Bivariate Normal Ellipse Report . . . 140
Nonpar Density . . . 141
Quantile Density Contours Report . . . 141
Group By . . . 142
Fitting Menus . . . 142
Fitting Menu Options . . . 143
Diagnostics Plots . . . 146
Additional Examples of the Bivariate Platform . . . 147
Example of the Fit Special Option . . . 147
Example of the Fit Orthogonal Option . . . 150
Example of the Fit Robust Command . . . 151
Example of Group By Using Density Ellipses . . . 153
Example of Group By Using Regression Lines . . . 154
Example of Grouping Using a By Variable . . . 155
Statistical Details for the Bivariate Platform . . . 157

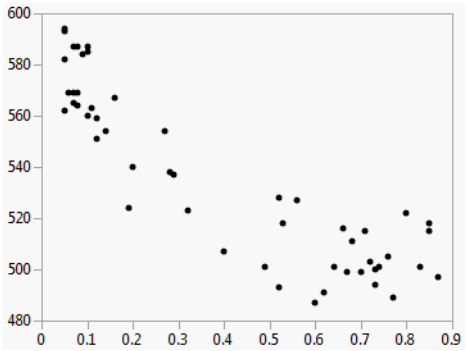
Fit Line 157
Fit Spline 157
Fit Orthogonal 157
Summary of Fit Report 158
Lack of Fit Report 159
Parameter Estimates Report 160
Smoothing Fit Reports 160
Bivariate Normal Ellipse Report 160

Example of Bivariate Analysis

Learn how to construct a scatterplot for two continuous variables. This example uses the SAT.jmp sample data table. SAT test scores for students in the 50 U.S. states, plus the District of Columbia, are divided into two areas: verbal and math. You want to find out how the percentage of students taking the SAT tests is related to verbal test scores for 2004.

1. Select **Help > Sample Data Library** and open SAT.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select 2004 Verbal and click **Y, Response**.
4. Select % Taking (2004) and click **X, Factor.**
5. Click **OK**.

Figure 5.2 Example of SAT Scores by Percent Taking



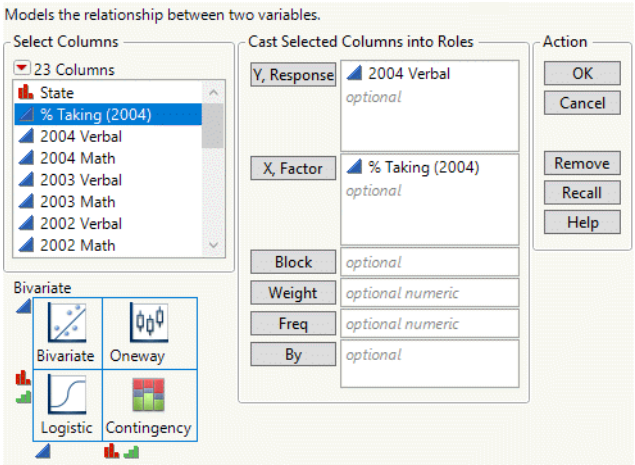
You can see that the verbal scores were higher when a smaller percentage of the population took the test.

Launch the Bivariate Platform

To investigate the relationship between two continuous variables in a bivariate analysis, do the following:

1. Select **Analyze > Fit Y by X**.
2. Enter a continuous column for **Y, Response**.
3. Enter a continuous column for **X, Factor**.

Figure 5.3 The Bivariate Launch Window



The word Bivariate appears above the diagram, to indicate that you are performing a bivariate analysis.

Note: You can also launch a bivariate analysis from the JMP Starter window. Select **View** > **JMP Starter** > **Basic** > **Bivariate**.

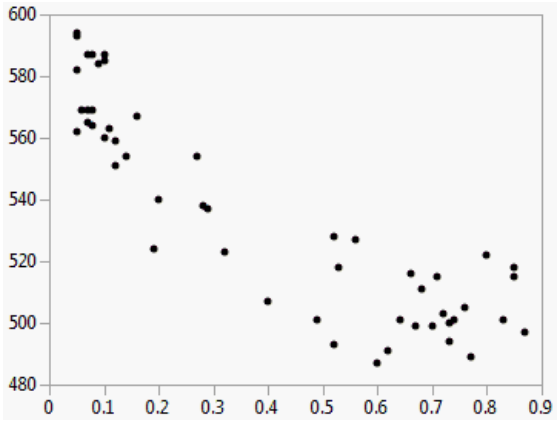
For more information about this launch window, see "Introduction to Fit Y by X". For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Bivariate Plot

The Bivariate report begins with a scatterplot for each pair of *X* and *Y* variables.

Note: To produce the plot shown in Figure 5.4, follow the instructions in "Example of Bivariate Analysis".

Figure 5.4 The Bivariate Plot



Replace variables in the plot by dragging and dropping a variable, in one of two ways: swap existing variables by dragging and dropping a variable from one axis to the other axis; or, click a variable in the Columns panel of the associated data table and drag it onto an axis.

You can interact with this plot just as you can with other JMP plots (for example, resizing the plot, highlighting points with the arrow or brush tool, and labeling points). For more information about these features, see *Using JMP*.

You can fit curves on the plot and view statistical reports and additional menus using the fitting options that are located within the red triangle menu. See "Fitting Options".

Fitting Options

The Bivariate Fit red triangle menu contains display options, fitting options, and control options.

Note: The Fit Group menu appears if you have specified multiple Y or multiple X variables. Menu options enable you to arrange reports or order them by RSquare. See *Fitting Linear Models*.

Show Points Hides or shows the points in the scatterplot. A check mark indicates that points are shown.

Histogram Borders Attaches histograms to the *x*- and *y*-axes of the scatterplot. A check mark indicates that histogram borders are turned on. See "Histogram Borders".

Note: When you apply only the Hidden row state to rows in the data table, the corresponding points do not appear in the scatterplot. However, the histograms are constructed using the hidden rows. If you want to exclude rows from the construction of the histograms and from analysis results, apply the Exclude row state and select **Redo > Redo Analysis** from the Bivariate red triangle menu.

Summary Statistics Shows the summary statistics for the plot, such as the correlation and confidence intervals, mean, and standard deviation.

Group By Lets you select a classification (or grouping) variable. A separate analysis is computed for each level of the grouping variable, and regression curves or ellipses are overlaid on the scatterplot. See "Group By".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

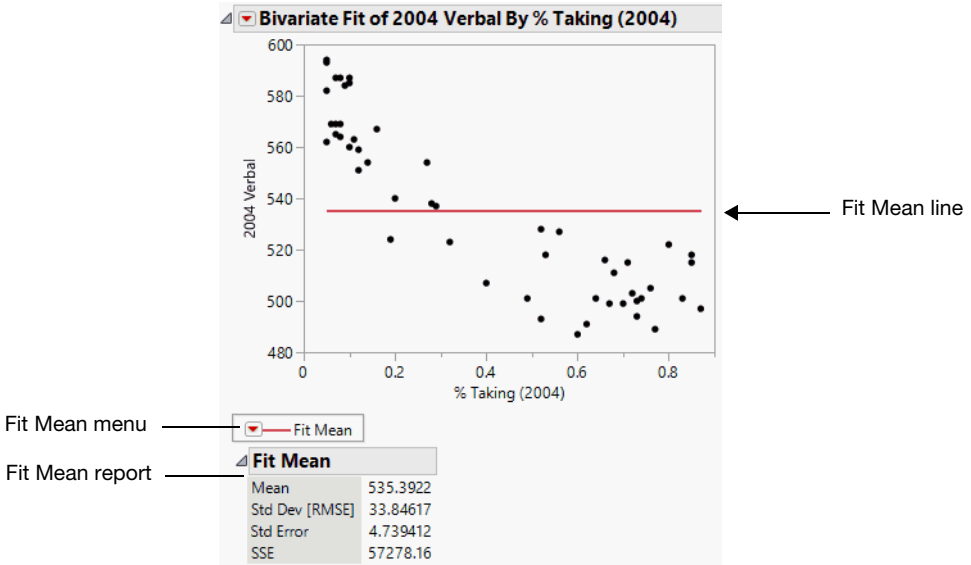
Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Fitting Options

Each fitting option adds the following:

- a line, curve, or distribution to the scatterplot
- a red triangle menu to the report window
- a specific report to the report window

Figure 5.5 Example of the Fit Mean Option



The following fitting options are available:

Fit Mean Adds a horizontal line to the scatterplot that represents the mean of the Y response variable. See "Fit Mean".

Fit Line Adds straight line fits to your scatterplot using least squares regression. See "Fit Line and Fit Polynomial".

Fit Polynomial Fits polynomial curves of a certain degree using least squares regression. See "Fit Line and Fit Polynomial".

Fit Special Transforms *Y* and *X*. Transformations include: log, square root, square, reciprocal, and exponential. You can also turn off center polynomials, constrain the intercept and the slope, and fit polynomial models. See "Fit Special".

Flexible Provides options that enable you to control the smoothness of the estimated regression curve. See "Flexible".

Fit Orthogonal Provides options for orthogonal regression fits, which are useful when X is assumed to vary. This option provides sub options that reflect various assumptions about the variances of X and Y. See "Fit Orthogonal".

Robust Provides options that reduce the influence of outliers in your data set on the fitted model. See "Robust".

Density Ellipse Plots density ellipsoids for the bivariate normal distribution fit to the *X* and *Y* variables. See "Density Ellipse".

Nonpar Density Plots density contours based on a smoothed surface. The contours describe the density of data points. See "Nonpar Density".

Note: You can remove a fit using the **Remove Fit** option. See "Fitting Menu Options".

Fitting Option Categories

Fitting option categories include regression fits and density estimation.

Category	Description	Fitting Options
Regression Fits	Regression methods fit a curve to the observed data points. The fitting methods include least squares fits as well as spline fits, kernel smoothing, orthogonal fits, and robust fits.	**Fit Mean** **Fit Line** **Fit Polynomial** **Fit Special** **Flexible** **Fit Orthogonal** **Robust**
Density Estimation	Density estimation fits a bivariate distribution to the points. You can either select a bivariate normal density, characterized by elliptical contours, or a general nonparametric density.	**Density Ellipse** **Nonpar Density**

Fit the Same Option Multiple Times

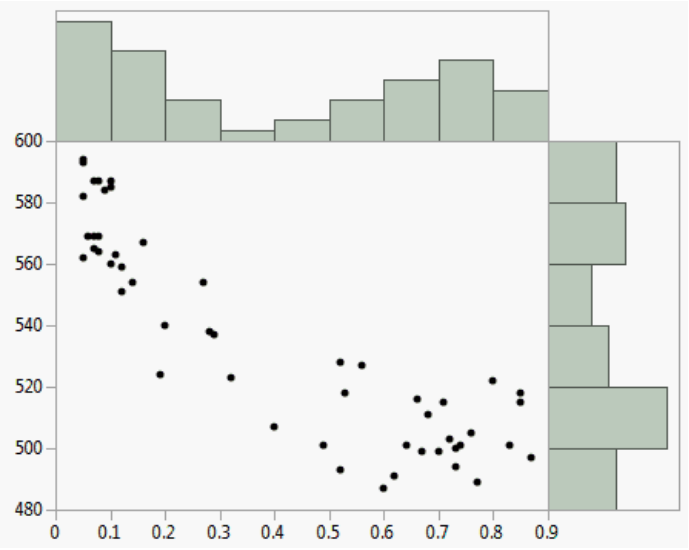
You can select the same fitting option multiple times, and each new fit is overlaid on the scatterplot. You can try fits, exclude points and refit, and you can compare them on the same scatterplot.

To apply a fitting option to multiple analyses in your report window, hold down the Ctrl key and select a fitting option.

Histogram Borders

The **Histogram Borders** option appends histograms to the *x*- and *y*-axes of the scatterplot. You can use the histograms to visualize the marginal distributions of the X and Y variables.

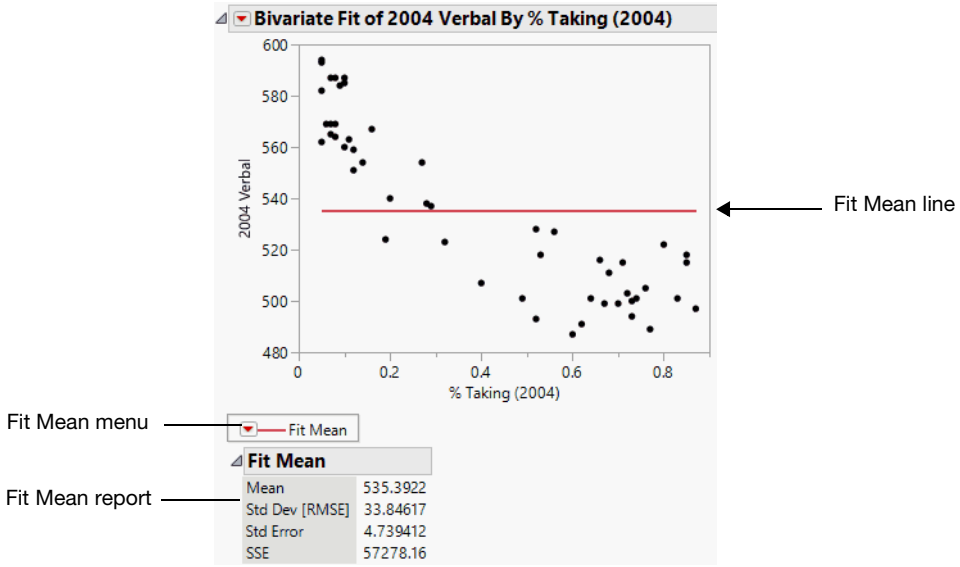
Figure 5.6 Example of Histogram Borders



Fit Mean

Using the **Fit Mean** option, you can add a horizontal line to the scatterplot that represents the mean of the *Y* response variable. You can start by fitting the mean and then use the mean line as a reference for other fits (such as straight lines, confidence curves, polynomial curves, and so on).

Figure 5.7 Example of Fit Mean



Fit Mean Report

The Fit Mean report shows summary statistics about the fit of the mean.

Mean Mean of the response variable. The predicted response when there are no specified effects in the model.

Std Dev [RMSE] Standard deviation of the response variable. Square root of the mean square error, also called the root mean square error (or RMSE).

Std Error Standard deviation of the response mean. Calculated by dividing the RMSE by the square root of the number of values.

SSE Error sum of squares for the simple mean model. Appears as the sum of squares for Error in the analysis of variance tables for each model fit.

For more information about the options in the Fit Mean menu, see "Fitting Menus".

Fit Line and Fit Polynomial

Using the **Fit Line** option, you can add straight line fits to your scatterplot using least squares regression. Using the **Fit Polynomial** option, you can fit polynomial curves of a certain degree using least squares regression.

Figure 5.8 Example of Fit Line and Fit Polynomial

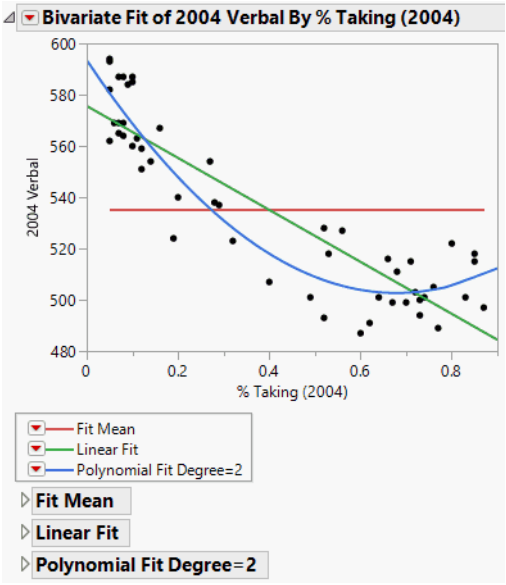


Figure 5.8 shows an example that compares a linear fit to the mean line and to a degree 2 polynomial fit.

Note the following information:

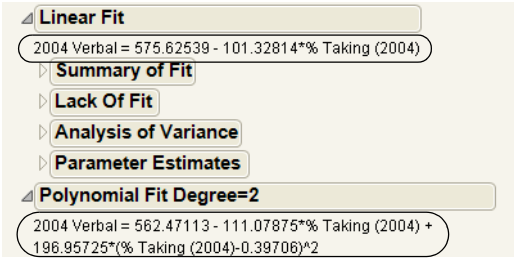
- The **Fit Line** output is equivalent to a polynomial fit of degree 1.
- The **Fit Mean** output is equivalent to a polynomial fit of degree 0.

For more information about the options in the Linear Fit and Polynomial Fit Degree menus, see "Fitting Menus". For statistical details about this fit, see "Fit Line".

Linear Fit and Polynomial Fit Reports

The Linear Fit and Polynomial Fit reports begin with the equation of fit.

Figure 5.9 Example of Equations of Fit



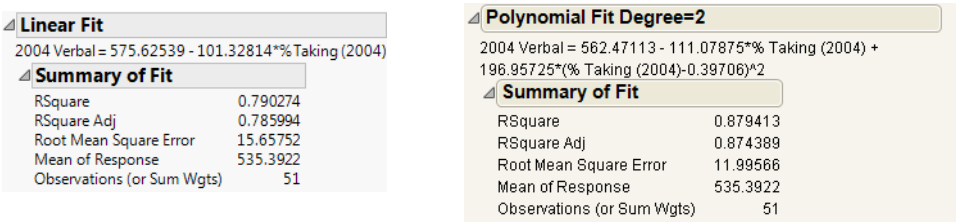
Tip: You can edit the equation by clicking on it.

Each Linear and Polynomial Fit Degree report contains at least three reports. A fourth report, Lack of Fit, appears only if there are *X* replicates in your data.

Summary of Fit Report

The Summary of Fit reports in the Bivariate platform show the numeric summaries of the response for the linear fit and polynomial fit of degree 2 for the same data. You can compare multiple Summary of Fit reports to see the improvement of one model over another, indicated by a larger RSquare value and smaller Root Mean Square Error.

Figure 5.10 Summary of Fit Reports for Linear and Polynomial Fits



The Summary of Fit report contains the following columns:

RSquare Measures the proportion of the variation explained by the model. The remaining variation is not explained by the model and attributed to random error. The RSquare is 1 if the model fits perfectly.

Note: A low RSquare value suggests that there might be variables not in the model that account for the unexplained variation. However, if your data are subject to a large amount of inherent variation, even a useful regression model can have a low RSquare value. Read the literature in your research area to learn about typical RSquare values.

The RSquare values in Figure 5.10 indicate that the polynomial fit of degree 2 gives a small improvement over the linear fit. See "Summary of Fit Report".

RSquare Adj Adjusts the RSquare value to make it more comparable over models with different numbers of parameters by using the degrees of freedom in its computation. See "Summary of Fit Report".

Root Mean Square Error Estimates the standard deviation of the random error. It is the square root of the mean square for Error in the Analysis of Variance report (Figure 5.12).

Mean of Response Provides the sample mean (arithmetic average) of the response variable. This is the predicted response when no model effects are specified.

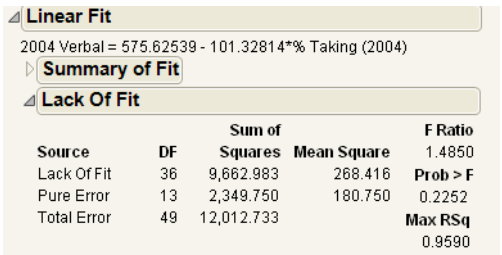
Observations Provides the number of observations used to estimate the fit. If there is a weight variable, this is the sum of the weights.

Lack of Fit Report

Note: The Lack of Fit report appears only if there are multiple rows that have the same *x* value.

Using the Lack of Fit report, you can estimate the error, regardless of whether you have the right form of the model. This occurs when multiple observations occur at the same *x* value. The error that you measure for these exact replicates is called *pure error*. This is the portion of the sample error that cannot be explained or predicted no matter what form of model is used. However, a lack of fit test might not be of much use if it has only a few degrees of freedom for it (few replicated *x* values).

Figure 5.11 Examples of Lack of Fit Reports for Linear and Polynomial Fits

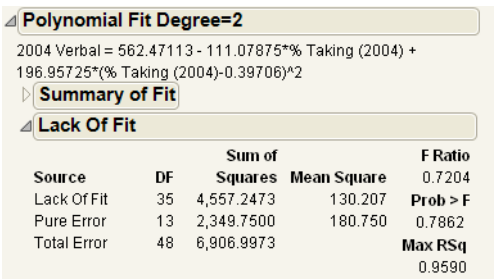

Linear Fit

2004 Verbal = 575.62539 - 101.32814*% Taking (2004)

Summary of Fit

Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
Lack Of Fit	36	9,662.983	268.416	1.4850
Pure Error	13	2,349.750	180.750	Prob > F
Total Error	49	12,012.733		0.2252
				Max RSq
				0.9590


Polynomial Fit Degree=2

2004 Verbal = 562.47113 - 111.07875*% Taking (2004) + 196.95725*(% Taking (2004)-0.39706)^2

Summary of Fit

Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
Lack Of Fit	35	4,557.2473	130.207	0.7204
Pure Error	13	2,349.7500	180.750	Prob > F
Total Error	48	6,906.9973		0.7862
				Max RSq
				0.9590

The difference between the residual error from the model and the pure error is called the *lack of fit error*. The lack of fit error can be significantly greater than the pure error if you have the wrong functional form of the regressor. In that case, you should try a different type of model fit. The Lack of Fit report tests whether the lack of fit error is zero.

The Lack of Fit report contains the following columns:

Source The three sources of variation: Lack of Fit, Pure Error, and Total Error.

DF The *degrees of freedom* (DF) for each source of error.

- The **Total Error** DF is the degrees of freedom found on the Error line of the Analysis of Variance table (shown under the "Analysis of Variance Report"). It is the difference between the **Total** DF and the **Model** DF found in that table. The **Error** DF is partitioned into degrees of freedom for lack of fit and for pure error.
- The **Pure Error** DF is pooled from each group where there are multiple rows with the same values for each effect. See "Lack of Fit Report".
- The **Lack of Fit** DF is the difference between the **Total Error** and **Pure Error** DF.

Sum of Squares The sum of squares (SS for short) for each source of error.

- The **Total Error** SS is the sum of squares found on the **Error** line of the corresponding Analysis of Variance table, shown under "Analysis of Variance Report".
- The **Pure Error** SS is pooled from each group where there are multiple rows with the same value for the *x* variable. This estimates the portion of the true random error that is not explained by model *x* effect. See "Lack of Fit Report".
- The **Lack of Fit** SS is the difference between the **Total Error** and **Pure Error** sum of squares. If the lack of fit SS is large, the model might not be appropriate for the data. The *F*-ratio described below tests whether the variation due to lack of fit is small enough to be accepted as a negligible portion of the pure error.

Mean Square The sum of squares divided by its associated degrees of freedom. This computation converts the sum of squares to an average (mean square). *F*-ratios for statistical tests are the ratios of mean squares.

F Ratio The ratio of mean square for lack of fit to mean square for Pure Error. It tests the hypothesis that the lack of fit error is zero.

Prob > F The probability of obtaining a greater *F*-value if the variation due to lack of fit variance and the pure error variance are the same. A high *p*-value means that there is not a significant lack of fit.

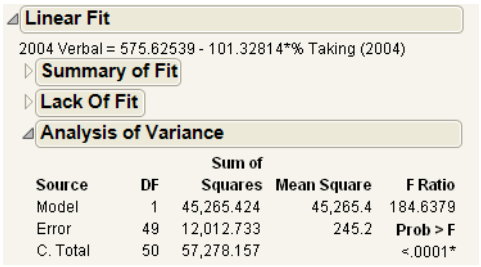
Max RSq The maximum R^2 that can be achieved by a model using only the variables in the model. See "Lack of Fit Report".

Analysis of Variance Report

Analysis of variance (ANOVA) for a regression partitions the total variation of a sample into components. These components are used to compute an *F*-ratio that evaluates the effectiveness of the model. If the probability associated with the *F*-ratio is small, then the model is considered a better statistical fit for the data than the response mean alone.

The Analysis of Variance reports in Figure 5.12 compare a linear fit (**Fit Line**) and a second degree (**Fit Polynomial**). Both fits are statistically better from a horizontal line at the mean.

Figure 5.12 Examples of Analysis of Variance Reports for Linear and Polynomial Fits



Linear Fit

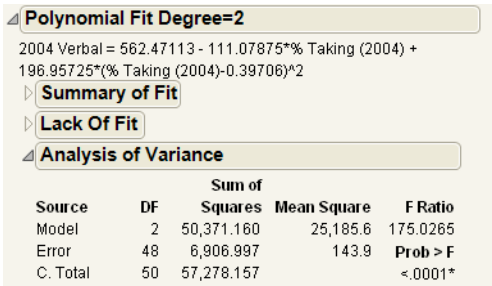
2004 Verbal = 575.62539 - 101.32814*% Taking (2004)

Summary of Fit

Lack Of Fit

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	1	45,265.424	45,265.4	184.6379
Error	49	12,012.733	245.2	Prob > F
C. Total	50	57,278.157		<.0001*



Polynomial Fit Degree=2

2004 Verbal = 562.47113 - 111.07875*% Taking (2004) + 196.95725*(% Taking (2004)-0.39706)^2

Summary of Fit

Lack Of Fit

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	2	50,371.160	25,185.6	175.0265
Error	48	6,906.997	143.9	Prob > F
C. Total	50	57,278.157		<.0001*

The Analysis of Variance Report contains the following columns:

Source The three sources of variation: **Model**, **Error**, and **C. Total**.

DF The degrees of freedom (DF) for each source of variation:

- A degree of freedom is subtracted from the total number of nonmissing values (*N*) for each parameter estimate used in the computation. The computation of the total sample variation uses an estimate of the mean. Therefore, one degree of freedom is subtracted from the total, leaving 50. The total corrected degrees of freedom are partitioned into the Model and Error terms.
- One degree of freedom from the total (shown on the **Model** line) is used to estimate a single regression parameter (the slope) for the linear fit. Two degrees of freedom are used to estimate the parameters (β_1 and β_2) for a polynomial fit of degree 2.
- The **Error** degrees of freedom is the difference between **C. Total** df and **Model** df.

Sum of Squares The sum of squares (SS for short) for each source of variation:

- In this example, the total (**C. Total**) sum of squared distances of each response from the sample mean is 57,278.157, as shown in Figure 5.12. That is the sum of squares for the base model (or simple mean model) used for comparison with all other models.
- For the linear regression, the sum of squared distances from each point to the line of fit reduces from 12,012.733. This is the residual or unexplained **(Error)** SS after fitting the model. The residual SS for a second degree polynomial fit is 6,906.997, accounting for slightly more variation than the linear fit. That is, the model accounts for more variation because the model SS are higher for the second degree polynomial than the linear fit. The **C. total** SS less the **Error** SS gives the sum of squares attributed to the model.

Mean Square The sum of squares divided by its associated degrees of freedom. The *F*-ratio for a statistical test is the ratio of the following mean squares:

- The **Model** mean square for the linear fit is 45,265.4. This value estimates the error variance, but only under the hypothesis that the model parameters are zero.

– The **Error** mean square is 245.2. This value estimates the error variance.

F Ratio The model mean square divided by the error mean square. The underlying hypothesis of the fit is that all the regression parameters (except the intercept) are zero. If this hypothesis is true, then both the mean square for error and the mean square for model estimate the error variance, and their ratio has an *F*-distribution.

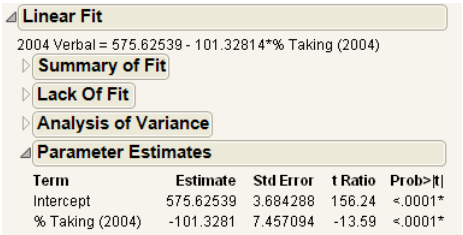
Prob > F The observed significance probability (*p*-value) of obtaining a greater *F*-value if the specified model fits no better than the overall response mean. Observed significance probabilities of 0.05 or less are often considered evidence of a regression effect.

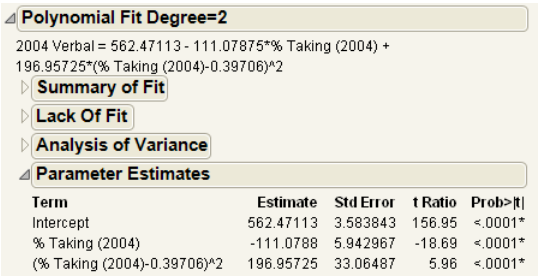
Parameter Estimates Report

The terms in the Parameter Estimates report for a linear fit are the intercept and the single *x* variable.

For a polynomial fit of order *k*, there is an estimate for the model intercept and a parameter estimate for each of the *k* powers of the *X* variable.

Figure 5.13 Examples of Parameter Estimates Reports for Linear and Polynomial Fits





The Parameter Estimates report contains the following columns:

Term Lists the name of each parameter in the requested model. The intercept is a constant term in all models.

Estimate Lists the parameter estimates of the linear model. The prediction formula is the linear combination of these estimates with the values of their corresponding variables.

Std Error Lists the estimates of the standard errors of the parameter estimates. They are used in constructing tests and confidence intervals.

t Ratio Lists the test statistics for the hypothesis that each parameter is zero. It is the ratio of the parameter estimate to its standard error. If the hypothesis is true, then this statistic has a Student's *t*-distribution.

Prob>|t| Lists the observed significance probability calculated from each *t*-ratio. It is the probability of getting a *t*-ratio greater (in absolute value) than the computed value, given a

true null hypothesis. Often, a value below 0.05 (or sometimes 0.01) is interpreted as evidence that the parameter is significantly different from zero.

To reveal additional statistics, right-click in the report and select the **Columns** menu. The following statistics are not shown by default:

Lower 95% The lower endpoint of the 95% confidence interval for the parameter estimate.

Upper 95% The upper endpoint of the 95% confidence interval for the parameter estimate.

Std Beta The standardized parameter estimate. It is useful for comparing the effect of *X* variables that are measured on different scales. See "Parameter Estimates Report".

VIF The variance inflation factor.

Design Std Error The design standard error for the parameter estimate. See "Parameter Estimates Report".

Fit Special

Using the **Fit Special** option, you can transform Y and X. Although data can be transformed for various reasons, it is often done to render data more plausibly normal, so that the appropriate tests can then be conducted. You can also constrain the slope and intercept, fit a polynomial of specific degree, and center the polynomial.

Note: For an example of this option, see "Example of the Fit Special Option".

The Specify Transformation or Constraint Window contains the following options:

Y or X Transformation Use one of these options to transform the Y or X variable:

- Natural logarithm
- Square root
- Square
- Reciprocal
- Exponential

Degree Use this option to fit a polynomial of the specified degree.

Centered Polynomial To turn off polynomial centering, deselect the **Centered Polynomial** check box (Figure 5.19). Note that for transformations of the *X* variable, polynomial centering is not performed. Centering polynomials stabilizes the regression coefficients and reduces multicollinearity.

Constrain Intercept to Select this check box to constrain the model intercept to be the specified value.

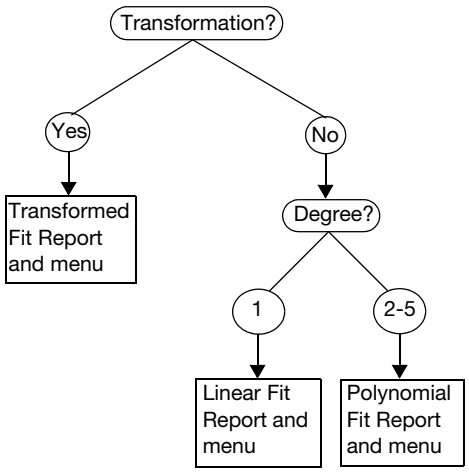
Constrain Slope to Select this check box to constrain the model slope to be the specified value.

For more information about the red triangle options for this fit, see "Fitting Menus".

Fit Special Reports and Menus

Depending on your selections in the Fit Special window, you see different reports and menus. The flowchart in Figure 5.14 shows you what reports and menus you see depending on your choices.

Figure 5.14 Example of Fit Special Flowchart



Transformed Fit Report

The Transformed Fit report contains the reports described in "Linear Fit and Polynomial Fit Reports". However, if you transformed Y, the Fit Measured on Original Scale report appears. This shows the measures of fit based on the original Y variables, and the fitted model transformed back to the original scale.

Flexible

Use the options in the Flexible menu to control the smoothness of the estimated regression curve.

- Fit Spline uses a penalized least squares approach. Adjust the degree of smoothness using the parameter lambda. See "Fit Spline".
- Kernel Smoother is based on locally weighted fits. Control the influence of local behavior using the parameter alpha. See "Kernel Smoother".
- Fit Each Value calculates the mean response at each X value. See "Fit Each Value".

Fit Spline

Using the **Fit Spline** option, you can fit a smoothing spline that varies in smoothness (or flexibility) according to the lambda (λ) value. The lambda value is a tuning parameter in the spline formula. As the value of λ decreases, the error term of the spline model has more weight and the fit becomes more flexible and curved. As the value of λ increases, the fit becomes stiff (less curved), approaching a straight line.

Note the following information:

- The smoothing spline can help you see the expected value of the distribution of *Y* across *X*.
- The points closest to each piece of the fitted curve have the most influence on it. The influence increases as you lower the value of λ, producing a highly flexible curve.
- If you want to use a lambda value that is not listed on the menu, select **Fit Spline > Other**. If the scaling of the *X* variable changes, the fitted model also changes. To prevent this from happening, select the **Standardize X** option. Note that the fitted model remains the same for either the original *X* variable or the scaled *X* variable.
- You might find it helpful to try several λ values. You can use the **Lambda** slider beneath the Smoothing Spline report to experiment with different λ values. However, λ is not invariant to the scaling of the data. For example, the λ value for an *X* measured in inches, is not the same as the λ value for an *X* measured in centimeters.

For more information about the options in the Smoothing Spline Fit menu, see "Fitting Menus". For statistical details about this fit, see "Fit Spline".

Smoothing Spline Fit Report

The Smoothing Spline Fit report contains the R-Square for the spline fit and the Sum of Squares Error. You can use these values to compare the spline fit to other fits, or to compare different spline fits to each other.

R-Square Measures the proportion of variation accounted for by the smoothing spline model. See "Smoothing Fit Reports".

Sum of Squares Error Sum of squared distances from each point to the fitted spline. It is the unexplained error (*residual*) after fitting the spline model.

Change Lambda Enables you to change the λ value, either by entering a number, or by moving the slider.

Kernel Smoother

The **Kernel Smoother** option produces a curve formed by repeatedly finding a locally weighted fit of a simple curve (a line or a quadratic) at sampled points in the domain. The many local fits (512 in total) are combined to produce the smooth curve over the entire domain. This method is also called *Loess* or *Lowess*, which was originally an acronym for Locally Weighted Scatterplot Smoother. The Kernel Smoother option implements the approach of Cleveland (1979) with a small adjustment for cases of near-perfect fits; the 6 * q50 argument in Cleveland's biweight function is replaced by max(6 * q50, 2 * q90), where q50 and q90 are the 50^{th} and 90^{th} percentiles, respectively.

Use this method to quickly see the relationship between variables and to help you determine the type of analysis or fit to perform.

For more information about the options in the Local Smoother menu, see "Fitting Menus".

Local Smoother Report

The Local Smoother report contains the R-Square for the smoother fit and the Sum of Squares Error. You can use these values to compare the smoother fit to other fits, or to compare different smoother fits to each other.

R-Square Measures the proportion of variation accounted for by the smoother model. See "Smoothing Fit Reports".

Sum of Squares Error Sum of squared distances from each point to the fitted smoother. It is the unexplained error (*residual*) after fitting the smoother model.

Local Fit (lambda) Select the polynomial degree for each local fit. Quadratic polynomials can track local bumpiness more smoothly. Lambda is the degree of certain polynomials that are fitted by the method. Lambda can be 0, 1 or 2.

Weight Function Specify how to weight the data in the neighborhood of each local fit. Loess uses tri-cube. The weight function determines the influence that each x_i and y_i has on the fitting of the line. The influence decreases as x_i increases in distance from x and finally becomes zero.

Smoothness (alpha) Controls how many points are part of each local fit. Use the slider or type in a value directly. Alpha is a smoothing parameter. It can be any positive number, but typical values are 1/4 to 1. As alpha increases, the curve becomes smoother.

Sampling Delta Controls the amount of sampling that is used in the fitting process. By default, the sampling delta is zero, which means that none of the points are skipped. As the sampling delta increases, points within delta of the last sample point are skipped in the fitting process. You can use this option to reduce the number of points used when the data are dense.

Robustness Re-weights the points to de-emphasize points that are farther from the fitted curve. Specify the number of times to repeat the process (number of passes). The goal is to converge the curve and automatically filter out outliers by giving them small weights.

Fit Each Value

The **Fit Each Value** option fits a value to each unique *X* value. The fitted values are the means of the response for each unique *X* value.

For more information about the options in the Fit Each Value menu, see "Fitting Menus".

Fit Each Value Report

The Fit Each Value report shows summary statistics about the model fit.

Number of Observations The total number of observations.

Number of Unique Values The number of unique X values.

Degrees of Freedom The pure error degrees of freedom.

Sum of Squares The pure error sum of squares.

Mean Square The pure error mean square.

Fit Orthogonal

The **Fit Orthogonal** option fits linear models that account for variability in *X* as well as *Y*. This is also known as Deming regression.

Note: For an example of this option, see "Example of the Fit Orthogonal Option".

Fit Orthogonal Options

Select one of the following options to specify a variance ratio.

Univariate Variances, Prin Comp Uses the univariate variance estimates computed from the samples of *X* and *Y*. This turns out to be the standardized first principal component. This option is not a good choice in a measurement systems application since the error variances are not likely to be proportional to the population variances.

Equal Variances Uses 1 as the variance ratio, which assumes that the error variances are the same. Using equal variances is equivalent to the non-standardized first principal component line. Suppose that the scatterplot is scaled the same in the *X* and *Y* directions. When you show a normal density ellipse, you see that this line is the longest axis of the ellipse.

Fit X to Y Uses a variance ratio of zero, which indicates that *Y* effectively has no variance.

Specified Variance Ratio Lets you enter any ratio that you want, giving you the ability to use known information about the measurement error in *X* and response error in *Y*.

For more information about the options in the Orthogonal Fit Ratio menu, see "Fitting Menus". For statistical details about this fit, see "Fit Orthogonal".

Orthogonal Fit Ratio Report

The Orthogonal Fit Ratio report shows summary statistics about the orthogonal regression model.

Variable The names of the variables used to fit the line.

Mean The mean of each variable.

Std Dev The standard deviation of each variable.

Variance Ratio The variance ratio used to fit the line.

Correlation The correlation between the two variables.

Intercept The intercept of the fitted line.

Slope The slope of the fitted line.

LowerCL The lower confidence limit for the slope.

UpperCL The upper confidence limit for the slope.

Alpha Enter the alpha level used in computing the confidence interval.

Robust

The **Robust** option provides two methods to reduce the influence of outliers in your data set. Outliers can lead to incorrect estimates and decisions.

Note: For an example of this option, see "Example of the Fit Robust Command".

For more information about the options in the Robust Fit and Cauchy Fit menus, see "Fitting Menus".

Fit Robust

The Fit Robust option reduces the influence of outliers in the response variable. The Huber M-estimation method is used. Huber M-estimation finds parameter estimates that minimize the Huber loss function:

$$l(\underline{e}) = \sum_i \rho(e_i)$$

where

$$\rho(e) = \begin{cases} \frac{1}{2}e^2 & \text{if } |e| < k \\ k|e| - \frac{1}{2}k^2 & \text{if } |e| \geq k \end{cases}$$

e_i refers to the residuals

The Huber loss function penalizes outliers and increases as a quadratic for small errors and linearly for large errors. In the JMP implementation, $k = 2$. For more information about robust fitting, see Huber (1973) and Huber and Ronchetti (2009).

Fit Cauchy

The Cauchy option estimates parameters using maximum likelihood and a Cauchy link function. This method assumes that the errors have a Cauchy distribution. A Cauchy distribution has fatter tails than the normal distribution, resulting in a reduced emphasis on outliers. This option can be useful if you have a large proportion of outliers in your data. However, if your data are close to normal with only a few outliers, this option can lead to incorrect inferences.

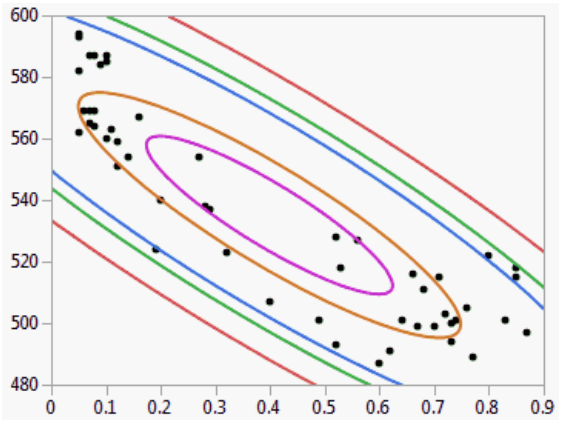
Density Ellipse

Using the **Density Ellipse** option, you can draw an ellipse (or ellipses) that contains the specified mass of points. The number of points is determined by the probability that you select from the **Density Ellipse** menu.

Note: For an example of this option, see "Example of Group By Using Density Ellipses".

For more information about the options in the Bivariate Normal Ellipse menu, see "Fitting Menus".

Figure 5.15 Example of Density Ellipses



The density ellipsoid is computed from the bivariate normal distribution fit to the *X* and *Y* variables. The bivariate normal density is a function of the means and standard deviations of the *X* and *Y* variables and the correlation between them. The **Other** selection lets you specify any probability greater than zero and less than or equal to one.

These ellipses are both density contours and confidence curves. As confidence curves, they show where a given percentage of the data is expected to lie, assuming the bivariate normal distribution.

The density ellipsoid is a good graphical indicator of the correlation between two variables. The ellipsoid collapses diagonally as the correlation between the two variables approaches either 1 or –1. The ellipsoid is more circular (less diagonally oriented) if the two variables are less correlated.

Bivariate Normal Ellipse Report

The Bivariate Normal Ellipse report that accompanies each **Density Ellipse** fit shows the correlation coefficient for the *X* and *Y* variables.

Note: To see a matrix of ellipses and correlations for many pairs of variables, use the **Multivariate** platform in the **Analyze > Multivariate Methods** menu.

Variable The names of the variables used in creating the ellipse

Mean The average of both the *X* and *Y* variable.

Std Dev The standard deviation of both the *X* and *Y* variable.

A discussion of the mean and standard deviation are in the section "The Summary Statistics Report".

Correlation The Pearson correlation coefficient. If there is an exact linear relationship between two variables, the correlation is 1 or –1 depending on whether the variables are positively or negatively related. If there is no relationship, the correlation tends toward zero. See "Bivariate Normal Ellipse Report".

Signif. Prob The probability of obtaining a correlation with greater absolute value than the computed value if no linear relationship exists between the *X* and *Y* variables.

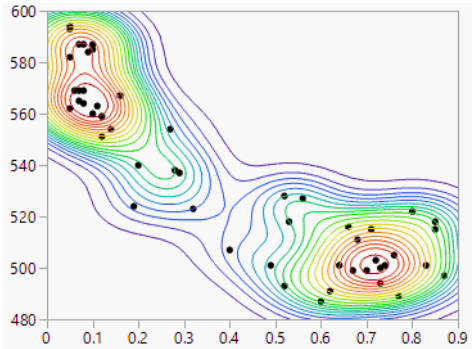
Number The number of observations used in the calculations.

Nonpar Density

When a plot shows thousands of points, the mass of points can be too dark to show patterns in density. Using the Nonpar Density (nonparametric density) option makes it easier to see the patterns.

Nonpar Density estimates a smooth nonparametric bivariate surface that describes the density of data points. The plot adds a set of contour lines showing the density (Figure 5.16). The contour lines are quantile contours in 5% intervals. This means that about 5% of the points generated from the estimated nonparametric distribution are below the lowest contour, 10% are below the next contour, and so on. The highest contour has about 95% of the points below it.

Figure 5.16 Example of Nonpar Density



To change the size of a nonparametric density contour grid, press Shift and select **Nonpar Density** from the Bivariate red triangle menu. Enter a larger value than the default 102 points.

For more information about the options in the Quantile Density Contours menu, see "Fitting Menus".

Quantile Density Contours Report

The Quantile Density Contours report shows the standard deviations used in creating the nonparametric density.

Group By

Using the **Group By** option, you can select a classification (grouping) variable. When a grouping variable is in effect, the Bivariate platform computes a separate analysis for each level of the grouping variable. The scatterplot shows the fitted models (lines, curves, contours, or ellipses) for each grouping variable. The fit for each level of the grouping variable is identified beneath the scatterplot, with individual pop-up menus.

When a grouping variable is in effect, the **Group By** option is checked in the Bivariate Fit red triangle menu. You can change the grouping variable by first selecting the **Group By** option to remove (uncheck) the existing variable. Then, select the **Group By** option again and respond to its window as before.

The **Group By** option enables you to generate separate analyses by a grouping variable while plotting the fits on the same scatterplot. This enables you to visually compare the fits across groups.

Note: For examples of this option, see "Example of Group By Using Density Ellipses" and "Example of Group By Using Regression Lines".

Fitting Menus

In addition to a report, each fitting option adds a fitting menu to the report window. The following table shows the fitting menus that correspond to each fitting option.

Fitting Option	Fitting Menu
Fit Mean	Fit Mean
Fit Line	Linear Fit
Fit Polynomial	Polynomial Fit Degree=X*
Fit Special	Linear Fit Polynomial Fit Degree=X* Transformed Fit X* Constrained Fits
Fit Spline	Smoothing Spline Fit, lambda=X*
Kernel Smoother	Local Smoother

Fitting Option	Fitting Menu
Fit Each Value	Fit Each Value
Fit Orthogonal	Orthogonal Fit Ratio=X*
Fit Robust	Robust Fit
Fit Cauchy	Cauchy Fit
Density Ellipse	Bivariate Normal Ellipse P=X*
Nonpar Density	Quantile Density Contours

*X=variable character or number

Fitting Menu Options

The Fitting menu options depend on the selected fit.

- "Options That Apply to Most Fits"
- "Options That Apply to Multiple Fits"
- "Options That Apply to Bivariate Normal Ellipse"
- "Options That Apply to Quantile Density Contours"

Options That Apply to Most Fits

Line of Fit Displays or hides the line or curve describing the model fit. For the Bivariate Normal Ellipse report, this option shows or hides the ellipse representing the contour border. Not applicable for Quantile Density Contours.

Line Color Lets you select from a palette of colors for assigning a color to each fit. Not applicable for Quantile Density Contours.

Line Style Lets you select from the palette of line styles for each fit. Not applicable for Quantile Density Contours.

Line Width Lets you change the line widths for the line of fit. The default line width is the thinnest line. Not applicable for Quantile Density Contours.

Report Turns the fit's report on and off. Does not modify the Bivariate plot.

Remove Fit Removes the fit from the graph and removes its report.

Options That Apply to Multiple Fits

Confid Curves Fit Displays or hides the confidence limits for the expected value (mean). This option is not available for the Fit Spline, Density Ellipse, Fit Each Value, and Fit Orthogonal fits and is dimmed on those menus.

Confid Curves Indiv Displays or hides the confidence limits for an individual predicted value. The confidence limits reflect variation in the error and variation in the parameter estimates. This option is not available for the Fit Mean, Fit Spline, Density Ellipse, Fit Each Value, and Fit Orthogonal fits and is dimmed on those menus.

Save Predicteds Creates a new column in the current data table called Predicted colname where colname is the name of the *Y* variable. This column includes the prediction formula and the computed sample predicted values. The prediction formula computes values automatically for rows that you add to the table. This option is not available for the Fit Each Value and Density Ellipse fits and is dimmed on those menus.

You can use the **Save Predicteds** and **Save Residuals** options for each fit. If you use these options multiple times or with a grouping variable, it is best to rename the resulting columns in the data table to reflect each fit.

Save Residuals Creates a new column in the current data table called Residuals colname where colname is the name of the *Y* variable. Each value is the difference between the actual (observed) value and its predicted value. Unlike the **Save Predicteds** option, this option does not create a formula in the new column. This option is not available for the Fit Each Value and Density Ellipse fits and is dimmed on those menus.

You can use the **Save Predicteds** and **Save Residuals** options for each fit. If you use these options multiple times or with a grouping variable, it is best to rename the resulting columns in the data table to reflect each fit.

Save Studentized Residuals Creates a new column in the data table containing the result of dividing the residual by the standard error of the residual.

Mean Confidence Limit Formula Creates a new column in the data table containing a formula for the mean confidence intervals.

Indiv Confidence Limit Formula Creates a new column in the data table containing a formula for the individual confidence intervals.

Plot Residuals (Linear, Polynomial, and Fit Special Only) Produces five diagnostic plots: residual by predicted, actual by predicted, residual by row, residual by X, and a normal quantile plot of the residuals. See "Diagnostics Plots".

Set α Level Enables you to set the alpha level used in computing confidence bands for various fits.

Confid Shaded Fit Draws the same curves as the **Confid Curves Fit** option and shades the area between the curves.

Confid Shaded Indiv Draws the same curves as the **Confid Curves Indiv** option and shades the area between the curves.

Save Coefficients Saves the spline coefficients as a new data table that contains columns named X, A, B, C, and D. The X column contains the knot points. A, B, C, and D are the intercept, linear, quadratic, and cubic coefficients of the third-degree polynomial. These coefficients span from the corresponding value in the X column to the next highest value.

Options That Apply to Bivariate Normal Ellipse

Shaded Contour Shades the area inside the density ellipse.

Select Points Inside Selects the points inside the ellipse.

Select Points Outside Selects the points outside the ellipse.

Options That Apply to Quantile Density Contours

Kernel Control Displays a slider for each variable, where you can change the standard deviation that defines the range of X and Y values for determining the density of contour lines.

5% Contours Shows or hides the 5% contour lines.

Contour Lines Shows or hides the 10% contour lines.

Contour Fill Fills the areas between the contour lines.

Color Theme Changes the color theme of the contour lines.

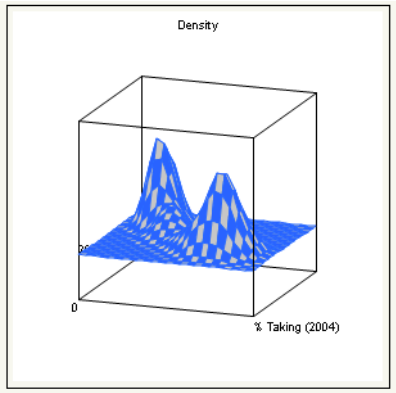
Select Points by Density Selects points that fall in a user-specified quantile range.

Color by Density Quantile Colors the points according to density.

Save Density Quantile Creates a new column containing the density quantile each point is in.

Mesh Plot A three-dimensional plot of the density over a grid of the two analysis variables.

Figure 5.17 Example of a Mesh Plot



Modal Clustering Creates a new column in the current data table and fills it with cluster values.

Note: If you save the modal clustering values first and then save the density grid, the grid table also contains the cluster values. The cluster values are useful for coloring and marking points in plots.

Save Density Grid Saves the density estimates and the quantiles associated with them in a new data table. The grid data can be used to visualize the density in other ways, such as with the Scatterplot 3D or the Contour Plot platforms.

Diagnostics Plots

The **Plot Residuals** option creates residual plots and other plots to diagnose the model fit. The following plots are available:

Residual by Predicted Plot A plot of the residuals versus the predicted values. A histogram of the residuals is also created.

Actual by Predicted Plot A plot of the actual values versus the predicted values.

Residual by Row Plot A plot of the residual values versus the row number.

Residual by X Plot A plot of the residual values versus the X variable.

Residual Normal Quantile Plot A Normal quantile plot of the residuals.

Additional Examples of the Bivariate Platform

These sections provide examples of different bivariate fits.

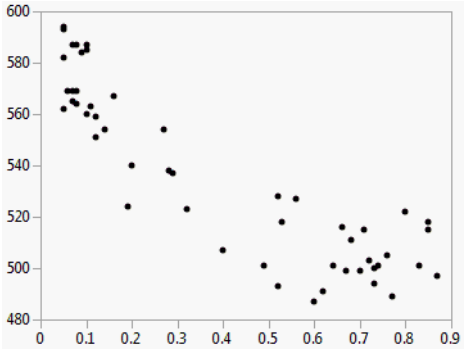
- "Example of the Fit Special Option"
- "Example of the Fit Orthogonal Option"
- "Example of the Fit Robust Command"
- "Example of Group By Using Density Ellipses"
- "Example of Group By Using Regression Lines"
- "Example of Grouping Using a By Variable"

Example of the Fit Special Option

This example shows you how to add transformations to your data. Start by transforming Y as log and X as square root:

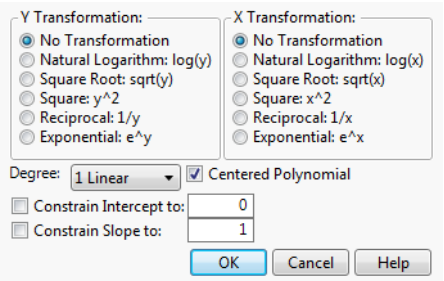
1. Select **Help > Sample Data Library** and open SAT.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select 2004 Verbal and click **Y, Response**.
4. Select % Taking (2004) and click **X, Factor.**
5. Click **OK**.

Figure 5.18 Example of SAT Scores by Percent Taking



6. Click the Bivariate Fit red triangle and select **Fit Special**. The Specify Transformation or Constraint window appears. For a description of this window, see "Fit Special".

Figure 5.19 The Specify Transformation or Constraint Window



7. Within Y Transformation, select **Natural Logarithm: log(y)**.
8. Within X Transformation, select **Square Root: sqrt(x)**.
9. Click **OK**.

Figure 5.20 Example of Fit Special Report

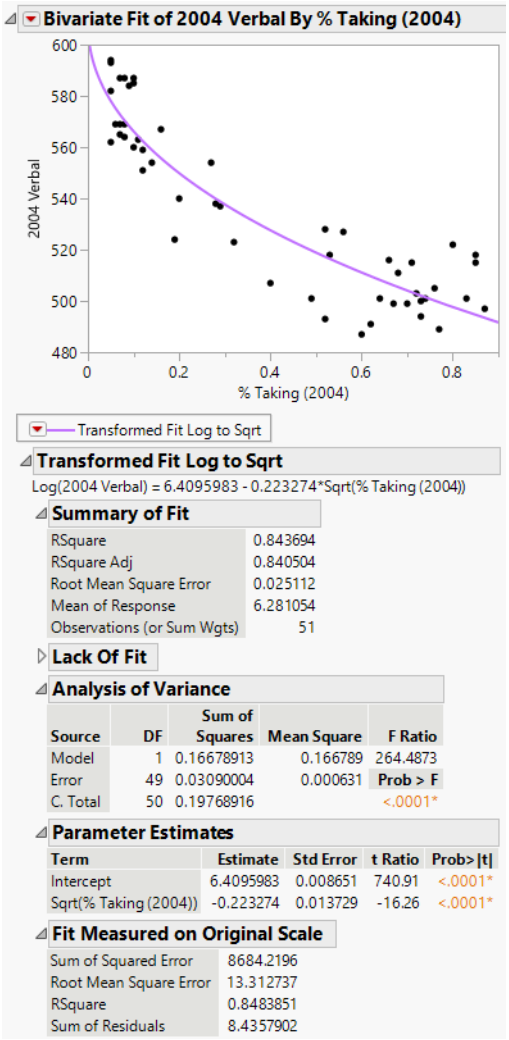


Figure 5.20 shows the fitted line plotted on the original scale. The model appears to fit the data well, as the plotted line goes through the cloud of points.

Example of the Fit Orthogonal Option

This example shows you how to standardize variables using the Distribution platform and then use the standardized variables to fit the orthogonal model.

Standardize the Variables

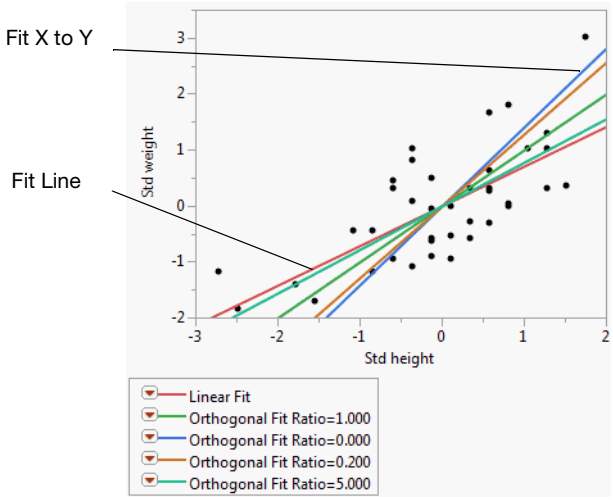
1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Distribution**.
3. Select height and weight and click **Y, Columns**.
4. Click **OK**.
5. Hold down the Ctrl key. Click the height red triangle and select **Save > Standardized**.

   Holding down the Ctrl key broadcasts the operation to all variables in the report window. Notice that in the Big Class.jmp sample data table, two new columns have been added.
6. Close the Distribution report window.

Use the Standardized Variables to Fit the Orthogonal Model

1. From the Big Class.jmp sample data table, select **Analyze > Fit Y by X**.
2. Select Std weight and click **Y, Response**.
3. Select Std height and click **X, Factor**.
4. Click **OK**.
5. Click the red triangle next to Bivariate Fit of Std weight By Std height and select **Fit Line**.
6. Click the red triangle next to Bivariate Fit of Std weight By Std height and select **Fit Orthogonal**. Then select each of the following:
   - **Equal Variances**
   - **Fit X to Y**
   - **Specified Variance Ratio** and type 0.2.
   - **Specified Variance Ratio** and type 5.

Figure 5.21 Example of Orthogonal Fitting Options



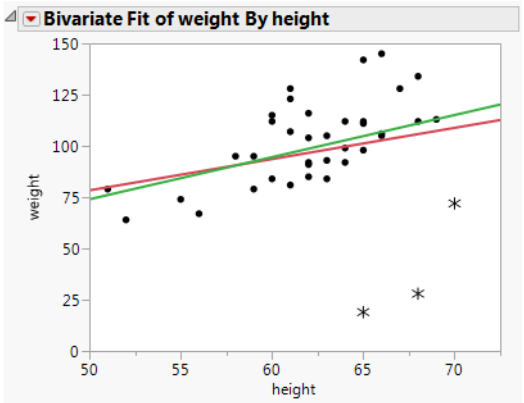
The scatterplot in Figure 5.21 shows the standardized height and weight values with various line fits that illustrate the behavior of the orthogonal variance ratio selections. The standard linear regression (**Fit Line**) occurs when the variance of the *X* variable is considered to be very small. **Fit X to Y** is the opposite extreme, when the variation of the *Y* variable is ignored. All other lines fall between these two extremes and shift as the variance ratio changes. As the variance ratio increases, the variation in the *Y* response dominates and the slope of the fitted line shifts closer to the *Y* by *X* fit. Likewise, when you decrease the ratio, the slope of the line shifts closer to the *X* by *Y* fit.

Example of the Fit Robust Command

This example shows you how to fit a robust model using the Huber M-estimation method. The data in the Weight Measurements.jmp sample data table shows the height and weight measurements taken by 40 students.

1. Select **Help > Sample Data Library** and open Weight Measurements.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response**.
4. Select height and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Bivariate Fit of weight By height and select **Fit Line**.
7. Click the red triangle next to Bivariate Fit of weight By height and select **Robust > Fit Robust**.

Figure 5.22 Example of Robust Fit



Linear Fit

weight = 2.1466733 + 1.5244337*height

Summary of Fit

RSquare	0.062319
RSquare Adj	0.037643
Root Mean Square Error	25.41387
Mean of Response	97.5
Observations (or Sum Wgts)	40

Lack Of Fit

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	1	1631.144	1631.14	2.5255
Error	38	24542.856	645.86	Prob > F
C. Total	39	26174.000		0.1203

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	2.1466733	60.13567	0.04	0.9717
height	1.5244337	0.959253	1.59	0.1203

Robust Fit

Sigma	ChiSquare	PValue	LogWorth
19.9537	3.88052	0.0488*	1.31114

Parameter	Robust Estimate	Std Error
Intercept	-28.62	62.5811
height	2.05316	1.04226

If you look at the standard Analysis of Variance report, you might wrongly conclude that height and weight do not have a linear relationship, since the *p*-value is 0.1203. However, when you look at the Robust Fit report, you would probably conclude that they do have a linear relationship, because the *p*-value there is 0.0489. It appears that some of the measurements are unusually low, perhaps due to incorrect user input. These measurements were unduly influencing the analysis.

Example of Group By Using Density Ellipses

This example shows you how to use a grouping (By) variable and add density ellipses to your data. In the Hot Dogs.jmp sample data table, the Type column identifies three different types of hot dogs: beef, meat, or poultry. You want to group the three types of hot dogs according to their cost variables.

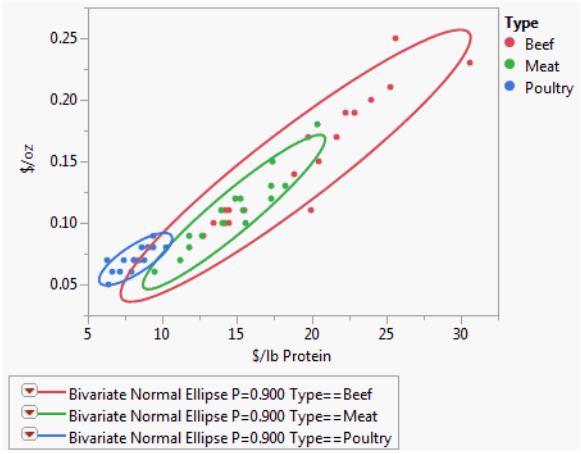
1. Select **Help > Sample Data Library** and open Hot Dogs.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select $/oz and click **Y, Response**.
4. Select $/lb Protein and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Bivariate Fit of $/oz By $/lb Protein and select **Group By**.
7. From the list, select Type.
8. Click **OK**.

   If you look at the **Group By** option again, you see it has a check mark next to it.
9. Click the red triangle next to Bivariate Fit of $/oz By $/lb Protein and select **Density Ellipse > 0.90**.

Color the points according to Type:

10. Right-click the scatterplot and select **Row Legend**.
11. Select Type in the column list and click **OK**.

Figure 5.23 Example of Group By



The ellipses in Figure 5.23 show clearly how the different types of hot dogs cluster with respect to the cost variables.

Example of Group By Using Regression Lines

This example shows you how to use a grouping variable to overlay regression lines to compare slopes of the different groups.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response**.
4. Select height and click **X, Factor**.
5. Click **OK**.

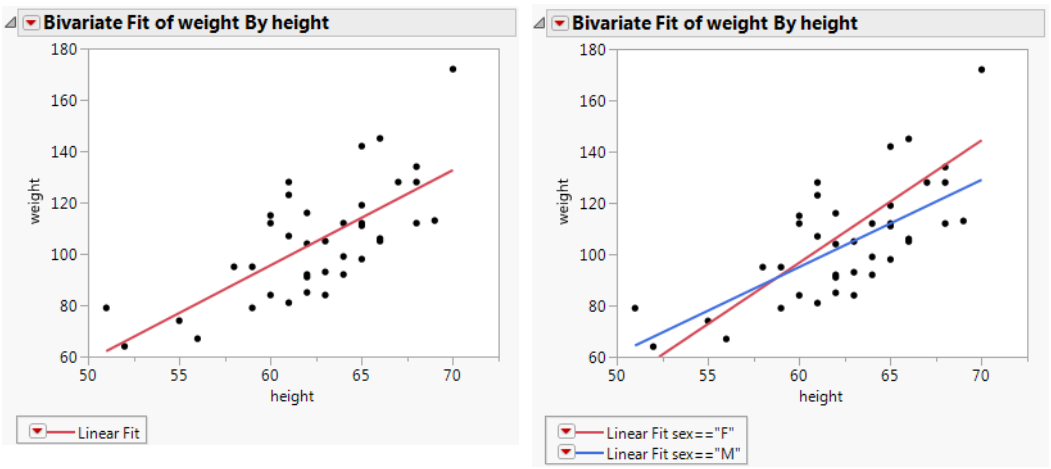
To create the example on the left in Figure 5.24:

6. Click the red triangle next to Bivariate Fit of weight By height and select **Fit Line**.

To create the example on the right in Figure 5.24:

7. From the Linear Fit menu, select **Remove Fit**.
8. Click the red triangle next to Bivariate Fit of weight By height and select **Group By**.
9. From the list, select sex.
10. Click **OK**.
11. Click the red triangle next to Bivariate Fit of weight By height and select **Fit Line**.

Figure 5.24 Example of Regression Analysis for Whole Sample and Grouped Sample



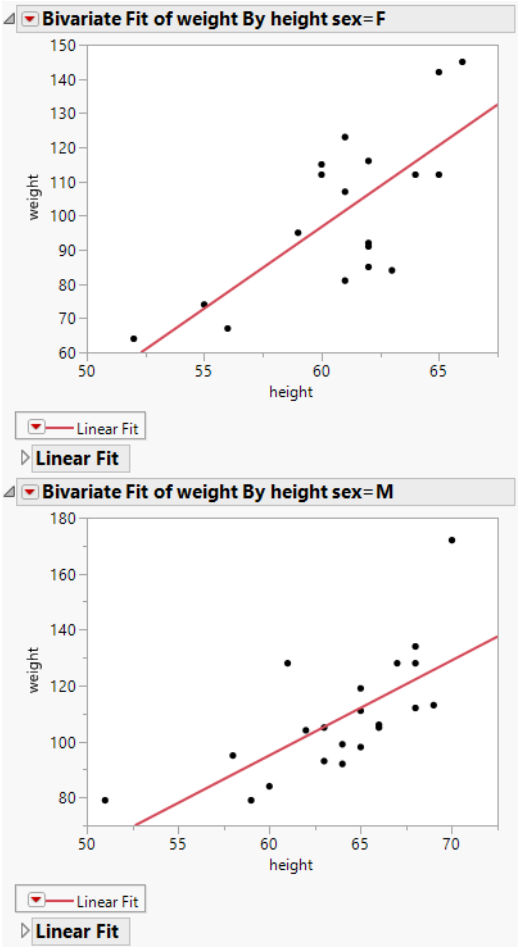
The scatterplot to the left in Figure 5.24 has a single regression line that relates weight to height. The scatterplot to the right shows separate regression lines for males and females.

Example of Grouping Using a By Variable

This example shows another method of grouping by specifying a By variable in the launch window. This results in separate reports and graphics for each level of the By variable (or combinations of By variables).

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response**.
4. Select height and click **X, Factor**.
5. Select sex and click **By**.
6. Click **OK**.
7. Press Ctrl, click the red triangle menu next to Bivariate Fit of weight By height sex =F, and select **Fit Line** from the red triangle menu.

Figure 5.25 Example of By Variable Plots



There are separate analyses for each level of the By variable (sex). So you see a scatterplot for the females and a scatterplot for the males.

Statistical Details for the Bivariate Platform

- "Fit Line"
- "Fit Spline"
- "Fit Orthogonal"
- "Summary of Fit Report"
- "Lack of Fit Report"
- "Parameter Estimates Report"
- "Smoothing Fit Reports"
- "Bivariate Normal Ellipse Report"

Fit Line

The **Fit Line** option finds the parameters β_0 and β_1 for the straight line that fits the points to minimize the residual sum of squares. The model for the i^{th} row is written $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$.

A polynomial of degree 2 is a parabola; a polynomial of degree 3 is a cubic curve. For degree k, the model for the i^{th} observation is defined as follows:

$$y_i = \sum_{j=0}^{k} \beta_j x_i^j + \varepsilon_i$$

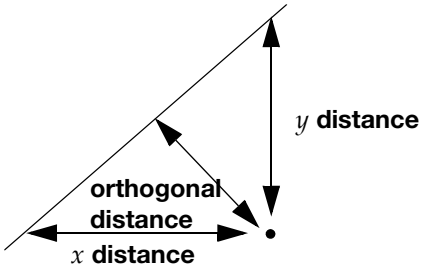
Fit Spline

The cubic spline method uses a set of third-degree polynomials spliced together such that the resulting curve is continuous and smooth at the splices (knot points). The estimation is done by minimizing an objective function that is a combination of the sum of squared errors and a penalty for curvature integrated over the curve extent. See the paper by Reinsch (1967) or the text by Eubank (1999) for a description of this method.

Fit Orthogonal

Standard least square fitting assumes that the X variable is fixed and the Y variable is a function of X plus error. If there is random variation in the measurement of X, you should fit a line that minimizes the sum of the squared perpendicular differences (Figure 5.26). However, the perpendicular distance depends on how X and Y are scaled, and the scaling for the perpendicular is reserved as a statistical issue, not a graphical one.

Figure 5.26 Line Perpendicular to the Line of Fit



The fit requires that you specify the ratio of the variance of the error in Y to the error in X. This is the variance of the error, not the variance of the sample points, so you must choose carefully. The ratio $(\sigma_y^2)/(\sigma_x^2)$ is infinite in standard least squares because σ_x^2 is zero. If you do an orthogonal fit with a large error ratio, the fitted line approaches the standard least squares line of fit. If you specify a ratio of zero, the fit is equivalent to the regression of X on Y, instead of Y on X.

The most common use of this technique is in comparing two measurement systems that both have errors in measuring the same value. Thus, the Y response error and the X measurement error are both the same type of measurement error. Where do you get the measurement error variances? You cannot get them from bivariate data because you cannot tell which measurement system produces what proportion of the error. So, you either must blindly assume some ratio like 1, or you must rely on separate repeated measurements of the same unit by the two measurement systems.

An advantage to this approach is that the computations give you predicted values for both Y and X; the predicted values are the point on the line that is closest to the data point, where closeness is relative to the variance ratio.

Confidence limits are calculated as described in Tan and Iglewicz (1999).

Summary of Fit Report

RSquare

Using quantities from the corresponding analysis of variance table, the RSquare for any continuous response fit is calculated as follows:

$$\frac{\text{Sum of Squares for Model}}{\text{Sum of Squares for C. Total}}$$

RSquare Adj

The RSquare Adj is a ratio of mean squares instead of sums of squares and is calculated as follows:

$$1 - \frac{\text{Mean Square for Error}}{\text{Mean Square for C. Total}}$$

The mean square for Error is in the Analysis of Variance report (Figure 5.12). You can compute the mean square for C. Total as the Sum of Squares for C. Total divided by its respective degrees of freedom.

Lack of Fit Report

Pure Error DF

For the Pure Error DF, consider the cases where more than one observation has the same value for height. In general, if there are g groups having multiple rows with identical values for each effect, the pooled DF, denoted DF_p, is defined as follows:

$$DF_p = \sum_{i=1}^{g} (n_i - 1)$$

where n_i is the number of observations in the ith group.

Pure Error SS

For the Pure Error SS, in general, if there are g groups having multiple rows with the same x value, the pooled SS, denoted SS_p, is defined as follows:

$$SS_p = \sum_{i=1}^{g} SS_i$$

where SS_i is the sum of squares for the ith group corrected for its mean.

Max RSq

Because Pure Error is invariant to the form of the model and is the minimum possible variance, Max RSq is calculated as follows:

$$1 - \frac{\text{SS(Pure error)}}{\text{SS(Total for whole model)}}$$

Parameter Estimates Report

Std Beta

Std Beta is calculated as follows:

$$\hat{\beta}(s_x / s_y)$$

where $\hat{\beta}$ is the estimated parameter, s_x and s_y are the standard deviations of the X and Y variables.

Design Std Error

Design Std Error is calculated as the standard error of the parameter estimate divided by the RMSE.

Smoothing Fit Reports

R-Square is equal to 1-(SSE/C.Total SS), where C.Total SS is available in the Fit Line ANOVA report.

Bivariate Normal Ellipse Report

The Pearson correlation coefficient is denoted *r*, and is computed as follows:

$$r_{xy} = \frac{s_{xy}}{\sqrt{s_x^2 s_y^2}} \quad \text{where} \quad s_{xy} = \frac{\sum w_i (x_i - \bar{x}_i)(y_i - \bar{y}_i)}{df}$$

Where w_i is either the weight of the *i*th observation if a weight column is specified, or 1 if no weight column is assigned.

Chapter 6

Oneway Analysis

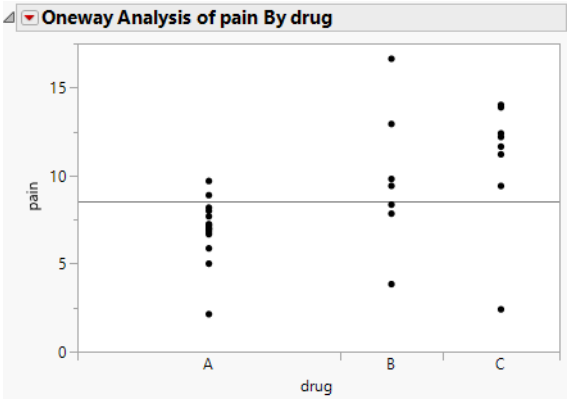
Examine Relationships between a Continuous Y and a Categorical X Variable

Use the Oneway platform to explore how the distribution of a continuous Y variable differs across groups defined by a categorical X variable. You can use ANOVA, nonparametric tests, or multiple comparisons to evaluate differences in means across groups. Density and CDF plots enable you to visualize the distribution of your response by the X categories. For example, you might want to find out how different categories of the same type of drug (X) affect patient pain levels on a numerical scale (Y).

The Oneway platform is the *continuous by nominal or ordinal* personality of the Fit Y by X platform. The analysis results appear in a plot, and you can interactively add additional analyses, such as the following:

- a one-way analysis of variance to fit means and to test that they are equal
- nonparametric tests
- a test for homogeneity of variance
- multiple-comparison tests on means, with means comparison circles
- outlier box plots overlaid on each group
- power details for the one-way layout

Figure 6.1 Oneway Analysis



Contents

Overview of Oneway Analysis 164
Example of Oneway Analysis 164
Launch the Oneway Platform 166
Data Format 167
The Oneway Plot 167
Oneway Platform Options 168
Display Options 171
Quantiles 172
Outlier Box Plots 173
Means/Anova and Means/Anova/Pooled t 174
The Summary of Fit Report 174
The t Test Report 175
The Analysis of Variance Report 176
The Means for Oneway Anova Report 177
The Block Means Report 177
Mean Diamonds and X-Axis Proportional 177
Mean Lines, Error Bars, and Standard Deviation Lines 178
Analysis of Means Methods 179
Analysis of Means for Location 180
Analysis of Means for Scale 180
Analysis of Means Charts 181
Analysis of Means Options 182
Compare Means 183
Using Comparison Circles 184
Each Pair, Student's t 186
All Pairs, Tukey HSD 186
With Best, Hsu MCB 186
With Control, Dunnett's 188
Each Pair Stepwise, Newman-Keuls 188
Compare Means Options 189
Nonparametric Tests 190
The Wilcoxon, Median, Van der Waerden, and Friedman Rank Test Reports 191
Kolmogorov-Smirnov Two-Sample Test Report 193
Nonparametric Multiple Comparisons 194
Unequal Variances 197
Tests That the Variances Are Equal Report 198
Equivalence Tests 199
Equivalence Tests Report 200
Robust 201

Robust Fit 201
Cauchy Fit 202
Power 202
Power Details Window and Reports 203
Normal Quantile Plot 204
CDF Plot 204
Densities 204
Matching Column 205
Additional Examples of the Oneway Platform 206
Example of an Analysis of Means Chart 206
Example of an Analysis of Means for Variances Chart 207
Example of the Each Pair, Student's t Test 208
Example of the All Pairs, Tukey HSD Test 210
Example of the With Best, Hsu MCB Test 212
Example of the With Control, Dunnett's Test 213
Example of the Each Pair Stepwise, Newman-Keuls Test 215
Example Contrasting Four Compare Means Tests 215
Example of the Nonparametric Wilcoxon Test 216
Example of the Unequal Variances Option 219
Example of an Equivalence Test 220
Example of the Robust Fit Option 222
Example of the Power Option 223
Example of a Normal Quantile Plot 225
Example of a CDF Plot 226
Example of the Densities Options 227
Example of the Matching Column Option 228
Example of Stacking Data for a Oneway Analysis 230
Statistical Details for the Oneway Platform 236
Comparison Circles 236
Power 238
Summary of Fit Report 238
Tests That the Variances Are Equal 239
Nonparametric Test Statistics 240

Overview of Oneway Analysis

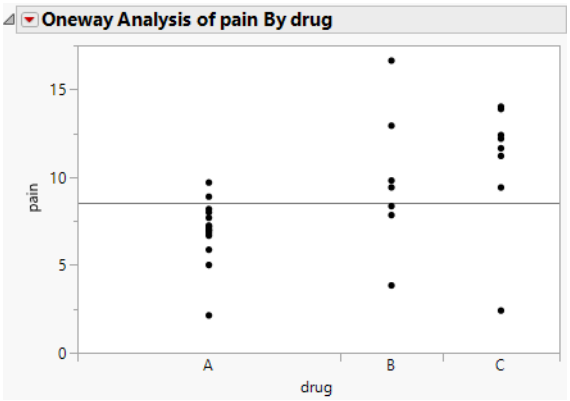
A one-way analysis of variance tests for differences between group means. The total variability in the response is partitioned into two parts: within-group variability and between-group variability. If the between-group variability is large relative to the within-group variability, then the differences between the group means are considered to be significant.

Example of Oneway Analysis

Learn how to perform a one-way analysis of variance in JMP to test if the differences in group means are significant. In this data, thirty-three participants were administered three different types of analgesics (A, B, and C). The participants were asked to rate their pain levels on a sliding scale. You want to find out if the means for A, B, and C are significantly different.

1. Select **Help > Sample Data Library** and open Analgesics.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select pain and click **Y, Response**.
4. Select drug and click **X, Factor**.
5. Click **OK**.

Figure 6.2 Example of Oneway Analysis



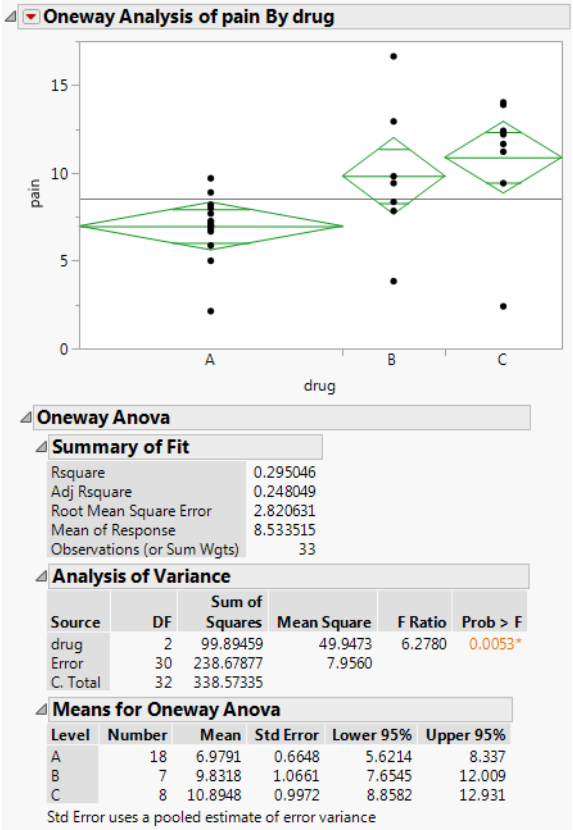
You notice that one drug (A) has consistently lower scores than the other drugs. You also notice that the *x*-axis ticks are unequally spaced. The length between the ticks is proportional to the number of scores (observations) for each drug.

Perform an analysis of variance on the data.

6. Click the red triangle next to Oneway Analysis of pain By drug and select **Means/Anova**.

Note: If the X factor has only two levels, the **Means/Anova** option appears as **Means/Anova/Pooled t,** and adds a Pooled *t* test report to the report window.

Figure 6.3 Example of the Means/Anova Option



Oneway Anova

Summary of Fit

Rsquare	0.295046
Adj Rsquare	0.248049
Root Mean Square Error	2.820631
Mean of Response	8.533515
Observations (or Sum Wgts)	33

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio	Prob > F
drug	2	99.89459	49.9473	6.2780	0.0053*
Error	30	238.67877	7.9560		
C. Total	32	338.57335			

Means for Oneway Anova

Level	Number	Mean	Std Error	Lower 95%	Upper 95%
A	18	6.9791	0.6648	5.6214	8.337
B	7	9.8318	1.0661	7.6545	12.009
C	8	10.8948	0.9972	8.8582	12.931

Std Error uses a pooled estimate of error variance

Note the following observations:

- Mean diamonds representing confidence intervals appear.
  - The line near the center of each diamond represents the group mean. At a glance, you can see that the mean for each drug looks significantly different.
  - The vertical span of each diamond represents the 95% confidence interval for the mean of each group.

  See "Mean Diamonds and X-Axis Proportional".
- The Summary of Fit table provides overall summary information about the analysis.

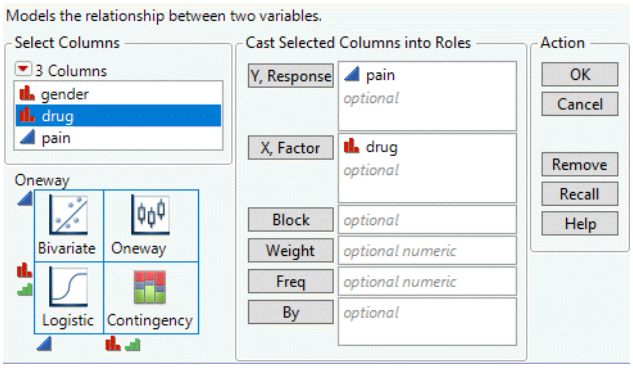
- The Analysis of Variance report shows the standard ANOVA information. You notice that the Prob > F (the *p*-value) is 0.0053, which supports your visual conclusion that there are significant differences between the drugs.
- The Means for Oneway Anova report shows the mean, sample size, and standard error for each level of the categorical factor.

Launch the Oneway Platform

To perform a one-way analysis of variance, do the following:

1. Select **Analyze > Fit Y by X**.
2. Enter a continuous column for **Y, Response**.
3. Enter a nominal or ordinal column for **X, Factor**.

Figure 6.4 The Fit Y by X Launch Window



The word Oneway appears above the diagram, to indicate that you are performing a one-way analysis.

Note: You can also launch a one-way analysis from the JMP Starter window. Select **View > JMP Starter > Basic > Oneway**.

For more information about this launch window, see "Introduction to Fit Y by X". For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Specifying a Block variable identifies a second factor, which forms a two-way analysis without interaction. In the Oneway plot, the values of the Y variable are centered by the Block variable. If there are equal counts in each block by group cell, the available Oneway platform options are adapted to the two-way analysis setting. If there are unequal counts in the block by group cells, a Fit Model report is shown below the Oneway plot and the Oneway platform options are not available.

Data Format

The Oneway platform requires that each row contain information for one or more observations with the same level of the X variable. If a row represents more than one observation, you must use a Weight or Freq variable to indicate how many observations the row represents.

When one-way data are in a format other than a JMP data table, sometimes they are arranged so that a row contains information for multiple observations. To analyze the data in JMP, you must import the data and restructure it so that each row of the JMP data table contains information for a single observation. See "Example of Stacking Data for a Oneway Analysis".

The Oneway Plot

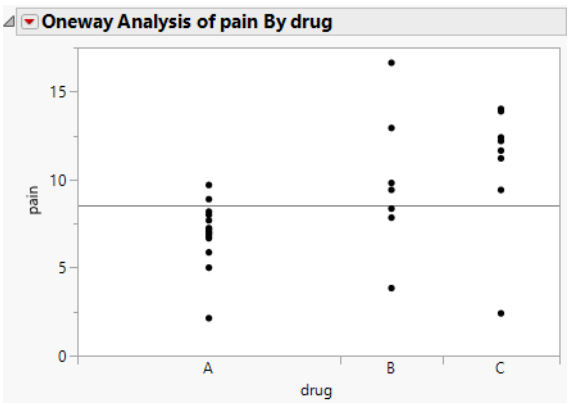
The Oneway plot shows the response points for each X factor value. You can compare the distribution of the response across the levels of the X factor. The distinct values of X are sometimes called levels. If a Block variable is specified in the launch window, the values of the Y variable in the Oneway plot are centered by the Block variable.

Replace variables in the plot in one of two ways: swap existing variables by dragging and dropping a variable from one axis to the other axis; or, click a variable in the Columns panel of the associated data table and drag it onto an axis.

You can add reports, additional plots, and tests to the report window using the options in the red triangle menu for Oneway Analysis. See "Oneway Platform Options".

To produce the plot shown in Figure 6.5, follow the instructions in "Example of Oneway Analysis".

Figure 6.5 The Oneway Plot



Note: Any rows that are excluded in the data table are also hidden in the Oneway plot.

Oneway Platform Options

The Oneway platform red triangle menu contains graph and analyses options. These include ANOVA, nonparametric tests, and multiple comparisons methods to evaluate differences in means across groups. Density and CDF plots enable you to visualize the distribution of your response by the X categories. Many of these options are not available if a Block variable is specified in the launch window.

Note: The Fit Group menu appears if you have specified multiple Y or X variables. Menu options enable you to arrange reports or order them by RSquare. See *Fitting Linear Models*.

Table 6.1 Examples of Options and Elements

Platform Option	Object Added to Plot	Report Added to Report Window
Quantiles	Box plots	Quantiles report
Means/Anova	Mean diamonds	Oneway ANOVA reports
Means and Std Dev	Mean lines, error bars, and standard deviation lines	Means and Std Deviations report
Compare Means	Comparison circles (except Each Pair Stepwise, Newman-Keuls option)	Means Comparison reports

The red triangle menu for Oneway Analysis provides the following options. Some options might not appear unless specific conditions are met.

Quantiles Lists the following quantiles for each group:

- 0% (Minimum)
- 10%
- 25%
- 50% (Median)
- 75%
- 90%

– 100% (Maximum)

Activates **Box Plots** from the **Display Options** menu. See "Quantiles".

Means/Anova Fits means for each group and performs a one-way analysis of variance to test if there are differences among the means. See "Means/Anova and Means/Anova/Pooled t".

Note: If the X factor has two levels, the menu option changes to **Means/Anova/Pooled t** and produces a Pooled *t* test report assuming equal variances.

Means and Std Dev Gives summary statistics for each group. The standard errors for the means use individual group standard deviations rather than the pooled estimate of the standard deviation.

The plot now contains mean lines, error bars, and standard deviation lines. For a brief description of these elements, see "Display Options". For more information about these elements, see "Mean Lines, Error Bars, and Standard Deviation Lines".

t test (Available only if the X factor has two levels.) Produces a *t* test report assuming that the variances are not equal. See "The t Test Report".

Analysis of Means Methods Provides five commands for performing Analysis of Means (ANOM) procedures. There are commands for comparing means, variances, and ranges. See "Analysis of Means Methods".

Compare Means Provides multiple-comparison methods for comparing sets of group means. See "Compare Means".

Nonparametric Provides nonparametric comparisons of group locations. See "Nonparametric Tests".

Unequal Variances Performs four tests for equality of group variances. Also gives the Welch test, which is an ANOVA test for comparing means when the variances within groups are not equal. See "Unequal Variances".

Equivalence Tests Tests that a difference is less than a specified threshold value. You can specify using a pooled variance or unequal variances for the variance assumption in the test. See "Equivalence Tests".

Robust Provides two methods for reducing the influence of outliers on your data. See "Robust".

Power Provides calculations of statistical power and other details about a given hypothesis test. See "Power".

The Power Details window and reports also appear within the Fit Model platform. For further discussion and examples of power calculations, see *Fitting Linear Models*.

Set α Level You can select an option from the most common alpha levels or specify any level with the **Other** selection. Changing the alpha level results in the following actions:

- recalculates confidence limits
- adjusts the mean diamonds on the plot (if they are showing)
- modifies the upper and lower confidence level values in reports
- changes the critical number and comparison circles for all Compare Means reports
- changes the critical number for all Nonparametric Multiple Comparison reports

Normal Quantile Plot Provides the following options for plotting the quantiles of the data in each group:

Plot Actual by Quantile Appends a quantile plot to the Oneway Analysis plot. The quantile plot shares the Oneway Analysis vertical axis for the response variable. The cumulative probabilities for each group are on the horizontal axis. The plot shows quantiles computed within each level of the categorical X factor.

Plot Quantile by Actual Generates a quantile plot with the response variable on the horizontal axis and cumulative probabilities on the vertical axis. The plot shows quantiles computed within each level of the categorical X factor.

Line of Fit Shows or hides a reference line fit to the data for each level of the X variable.

Normal Quantile Label Shows or hides labels inside the right vertical axis of the Normal Quantile Plot and inside the top axis of the Actual by Quantile plot. The labels represent the normal quantile scale.

CDF Plot Plots the cumulative distribution function for all of the groups in the Oneway report. See "CDF Plot".

Densities Compares densities across groups. See "Densities".

Matching Column Specify a matching variable to perform a matching model analysis. Use this option when the data in your Oneway analysis comes from matched (paired) data, such as when observations in different groups come from the same participant.

The plot now contains matching lines that connect the matching points. See "Matching Column".

Save Saves the following quantities as new columns in the current data table:

Save Residuals Saves values computed as the response variable minus the mean of the response variable within each level of the factor variable.

Save Standardized Saves standardized values of the response variable computed within each level of the factor variable. This is the centered response divided by the standard deviation within each level.

Save Normal Quantiles Saves normal quantile values computed within each level of the categorical factor variable.

Save Predicted Saves the predicted mean of the response variable for each level of the factor variable.

Display Options Adds or removes elements from the plot. See "Display Options".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Display Options

Using Display Options, you can add or remove elements from the Oneway data plot. Some options might not appear unless they are relevant.

All Graphs Shows or hides all graphs.

Points Shows or hides data points on the plot.

Box Plots Shows or hides outlier box plots for each group. For an example, see "Conduct the Oneway Analysis".

Mean Diamonds Draws a horizontal line through the mean of each group proportional to its horizontal axis. The top and bottom points of the mean diamond show the upper and lower 95% confidence points for each group. See "Mean Diamonds and X-Axis Proportional".

Mean Lines Draws a line at the mean of each group. See "Mean Lines, Error Bars, and Standard Deviation Lines".

Mean CI Lines Draws lines at the upper and lower 95% confidence levels for each group.

Mean Error Bars Identifies the mean of each group and shows error bars one standard error above and below the mean. See "Mean Lines, Error Bars, and Standard Deviation Lines".

Grand Mean Draws the overall mean of the *Y* variable on the plot.

Std Dev Lines Shows lines one standard deviation above and below the mean of each group. See "Mean Lines, Error Bars, and Standard Deviation Lines".

Comparison Circles Shows or hides comparison circles. This option is available only when one of the **Compare Means** options is selected. See "Comparison Circles". For an example, see "Conduct the Oneway Analysis".

Connect Means Connects the group means with a straight line.

Mean of Means Draws a line at the mean of the group means.

X-Axis proportional Makes the spacing on the *x*-axis proportional to the sample size of each level. See "Mean Diamonds and X-Axis Proportional".

Points Spread Spreads points over the width of the interval

Points Jittered Adds small spaces between points that overlay on the same *y* value. The horizontal adjustment of points varies from 0.375 to 0.625 with a $4*(\text{Uniform}(0,1)-0.5)^5$ distribution.

Matching Lines (Appears only when the **Matching Column** option is selected.) Connects matching points.

Matching Dotted Lines (Appears only when the **Matching Column** option is selected.) Draws dotted lines to connect cell means from missing cells in the table. The values used as the endpoints of the lines are obtained using a two-way ANOVA model.

Histograms Draws side-by-side histograms to the right of the original plot.

Robust Mean Lines (Appears only when a Robust option is selected.) Draws a line at the robust mean of each group.

Legend Displays a legend for the Normal Quantile Plot, CDF Plot, and Densities options.

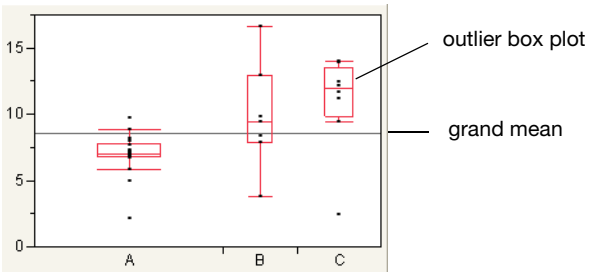
Quantiles

The Quantiles report lists selected percentiles for each level of the X factor variable. The median is the 50th percentile, and the 25th and 75th percentiles are called the *quartiles*.

The **Quantiles** option adds the following elements to the plot:

- the grand mean representing the overall mean of the Y variable
- outlier box plots summarizing the distribution of points at each factor level

Figure 6.6 Outlier Box Plot and Grand Mean



Note: To hide these elements, click the red triangle next to Oneway Analysis and select **Display Options > Box Plots** or **Grand Mean**.

Outlier Box Plots

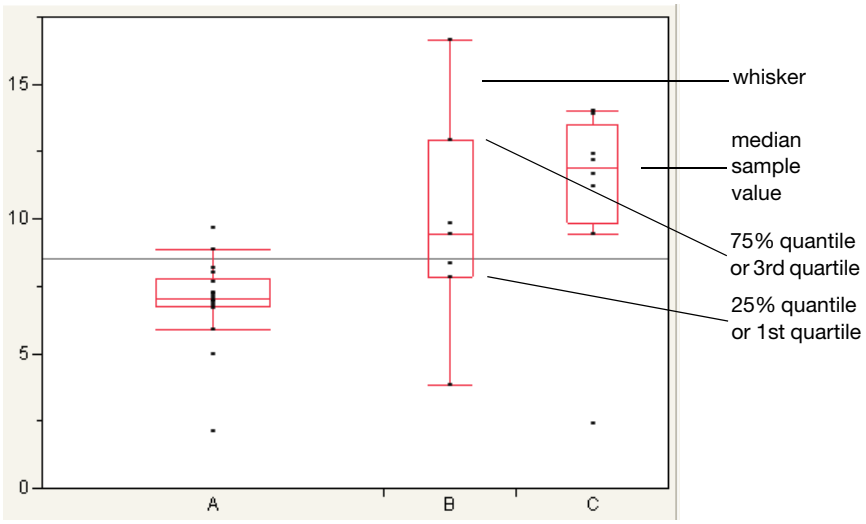
The outlier box plot is a graphical summary of the distribution of data. Note the following aspects about outlier box plots (Figure 6.7):

- The horizontal line within the box represents the median sample value.
- The ends of the box represent the 75th and 25th quantiles, also expressed as the 3rd and 1st quartile, respectively.
- The difference between the 1st and 3rd quartiles is called the *interquartile range*.
- Each box has lines, sometimes called *whiskers*, that extend from each end. The whiskers extend from the ends of the box to the outermost data point that falls within these distances:

  ```
  3rd quartile + 1.5*(interquartile range)
  1st quartile - 1.5*(interquartile range)
  ```

  If the data points do not reach the computed ranges, then the whiskers are determined by the upper and lower data point values (not including outliers).

Figure 6.7 Examples of Outlier Box Plots



Means/Anova and Means/Anova/Pooled t

The **Means/Anova** option performs an analysis of variance. If the X factor contains exactly two levels, this option appears as **Means/Anova/Pooled t**. In addition to the other reports, a Pooled *t* test report assuming pooled (or equal) variances appears.

Mean diamonds are added to the Oneway plot See "Display Options" and "Mean Diamonds and X-Axis Proportional".

Reports See "The Summary of Fit Report", "The Analysis of Variance Report", "The Means for Oneway Anova Report", "The t Test Report", and "The Block Means Report".

- The *t* test report appears only if the **Means/Anova/Pooled t** option is selected.
- The Block Means report appears only if you have specified a Block variable in the launch window and there are equal counts in each combination of block and X factor level.

The Summary of Fit Report

The Summary of Fit report in the Oneway platform shows a summary for a one-way analysis of variance.

Rsquare Measures the proportion of the variation accounted for by fitting means to each factor level. The remaining variation is attributed to random error. The R^2 value is 1 if fitting the group means account for all the variation with no error. An R^2 of 0 indicates that

the fit serves no better as a prediction model than the overall response mean. See the statistical details for the "Summary of Fit Report".

R^2 is also called the *coefficient of determination*.

Note: A low RSquare value suggests that there might be variables not in the model that account for the unexplained variation. However, if your data are subject to a large amount of inherent variation, even a useful ANOVA model can have a low RSquare value. Read the literature in your research area to learn about typical RSquare values.

Adj Rsquare Adjusts R^2 to make it more comparable over models with different numbers of parameters by using the degrees of freedom in its computation. See the statistical details for the "Summary of Fit Report".

Root Mean Square Error Estimates the standard deviation of the random error. It is the square root of the mean square for Error found in the Analysis of Variance report.

Mean of Response The overall mean (arithmetic average) of the response variable.

Observations (or Sum Wgts) Number of observations used in estimating the fit. If weights are used, this is the sum of the weights. See the statistical details for the "Summary of Fit Report".

The t Test Report

There are two types of *t* tests: equal variances and unequal variances.

- If you select the **Means/Anova/Pooled t** option, a Pooled t Test report appears. This *t* test assumes equal variances.
- If you select the **t Test** option from the red triangle menu, a *t* test report appears. This *t* test assumes unequal variances.

The t Test report contains the following columns:

t Test plot Shows the sampling distribution of the difference in the means, assuming that the null hypothesis is true. The vertical red line is the actual difference in the means. The shaded areas correspond to the *p*-values.

Difference Shows the estimated difference between the two X levels. In the plots, the Difference value appears as a red line that compares the two levels.

Std Err Dif Shows the standard error of the difference.

Upper CL Dif Shows the upper confidence limit for the difference.

Lower CL Dif Shows the lower confidence limit for the difference.

Confidence Shows the level of confidence (1-alpha). To change the level of confidence, select a new alpha level from the **Set** α **Level** command from the platform red triangle menu.

t Ratio Value of the *t*-statistic.

DF The degrees of freedom used in the *t* test.

Prob > |t| The *p*-value associated with a two-tailed test.

Prob > t The *p*-value associated with an upper-tailed test.

Prob < t The *p*-value associated with a lower-tailed test.

The Analysis of Variance Report

The Analysis of Variance report partitions the total variation of a sample into two components. The ratio of the two mean squares forms the *F* ratio. If the probability associated with the *F* ratio is small, then the model is a better fit statistically than the overall response mean.

Note: If you specified a **Block** column, then the Analysis of Variance report includes the **Block** variable.

Source Lists the three sources of variation. These sources are the model source, **Error**, and **C. Total** (corrected total).

DF Records an associated degrees of freedom (DF for short) for each source of variation:

- The degrees of freedom for **C. Total** are N - 1, where N is the total number of observations used in the analysis.
- If the X factor has k levels, then the model has k - 1 degrees of freedom.

The **Error** degrees of freedom is the difference between the **C. Total** degrees of freedom and the **Model** degrees of freedom (in other words, N - k).

Sum of Squares Records a sum of squares (SS for short) for each source of variation:

- The total (**C. Total**) sum of squares of each response from the overall response mean. The **C. Total** sum of squares is the base model used for comparison with all other models.
- The sum of squared distances from each point to its respective group mean. This is the remaining unexplained **Error** (residual) SS after fitting the analysis of variance model.

The total SS minus the error SS gives the sum of squares attributed to the model. This tells you how much of the total variation is explained by the model.

Mean Square Is a sum of squares divided by its associated degrees of freedom:

- The **Model** mean square estimates the variance of the error, but only under the hypothesis that the group means are equal.
- The **Error** mean square estimates the variance of the error term independently of the model mean square and is unconditioned by any model hypothesis.

F Ratio The model mean square divided by the error mean square. If the hypothesis that the group means are equal (there is no real difference between them) is true, then both the mean square for error and the mean square for model estimate the error variance. Their ratio has an *F* distribution. If the analysis of variance model results in a significant reduction of variation from the total, the *F* ratio is higher than expected.

Prob>F Probability of obtaining an *F* value greater than the one calculated if there is no difference in the population group means. Observed significance probabilities of 0.05 or less are often considered evidence that there are differences in the group means.

The Means for Oneway Anova Report

The Means for Oneway Anova report summarizes response information for each level of the nominal or ordinal factor.

Level Lists the levels of the X variable.

Number Lists the number of observations in each group.

Mean Lists the mean of each group.

Std Error Lists the estimates of the standard deviations for the group means. This standard error is estimated assuming that the variance of the response is the same in each level. It is the root mean square error found in the Summary of Fit report divided by the square root of the number of values used to compute the group mean.

Lower 95% and Upper 95% Lists the lower and upper 95% confidence interval for the group means.

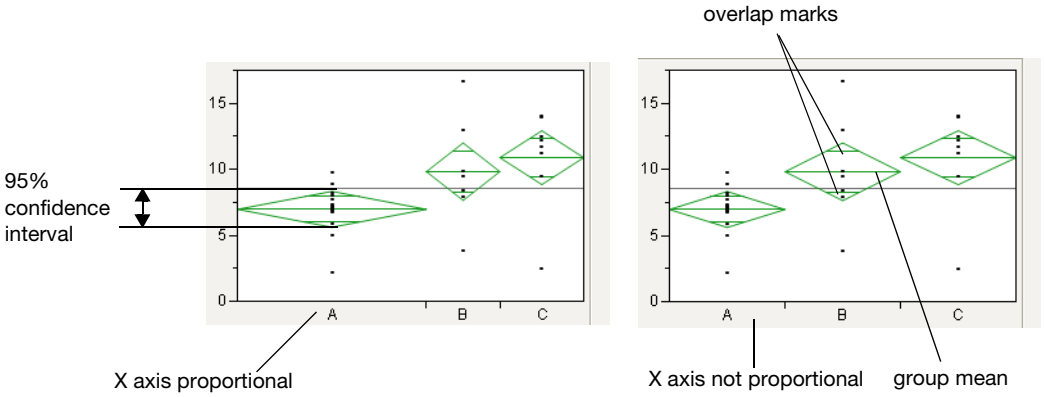
The Block Means Report

If you have specified a Block variable on the launch window, the **Means/Anova** and **Means/Anova/Pooled t** commands produce a Block Means report. This report shows the means for each block and the number of observations in each block.

Mean Diamonds and X-Axis Proportional

A mean diamond illustrates a sample mean and confidence interval.

Figure 6.8 Examples of Mean Diamonds and X-Axis Proportional Options



Note the following observations:

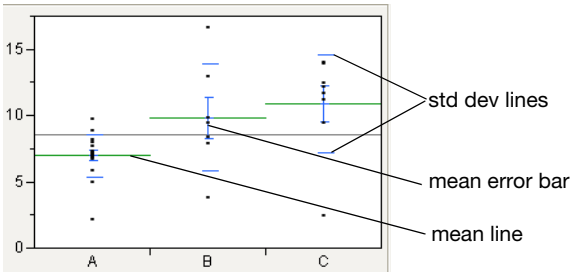
- The top and bottom of each diamond represent the (1-alpha)x100 confidence interval for each group. The confidence interval computation assumes that the variances are equal across observations. Therefore, the height of the diamond is proportional to the reciprocal of the square root of the number of observations in the group.
- If the **X-Axis proportional** option is selected, the horizontal extent of each group along the horizontal axis (the horizontal size of the diamond) is proportional to the sample size for each level of the X variable. Therefore, the narrower diamonds are usually taller, because fewer data points results in a wider confidence interval.
- The mean line across the middle of each diamond represents the group mean.
- Overlap marks appear as lines above and below the group mean. For groups with equal sample sizes, overlapping marks indicate that the two group means are not significantly different at the given confidence level. Overlap marks are computed as group mean $\pm \ (\sqrt{2})/2 \times \mathrm{CI}/2$. Overlap marks in one diamond that are closer to the mean of another diamond than that diamond's overlap marks indicate that those two groups are not different at the given confidence level.
- The mean diamonds automatically appear when you select the **Means/Anova/Pooled t** or **Means/Anova** option from the platform menu. However, you can show or hide them at any time by selecting **Display Options > Mean Diamonds** from the red triangle menu.

Mean Lines, Error Bars, and Standard Deviation Lines

Show mean lines by selecting **Display Options > Mean Lines**. Mean lines indicate the mean of the response for each level of the X variable.

Mean error bars and standard deviation lines appear when you select the **Means and Std Dev** option from the red triangle menu. To turn each option on or off singly, select **Display Options > Mean Error Bars** or **Std Dev Lines**.

Figure 6.9 Mean Lines, Mean Error Bars, and Std Dev Lines



Analysis of Means Methods

Analysis of means (ANOM) is a multiple comparison method for means, proportions, or variances. You might want to use these ANOM under these circumstances:

- to test whether any of the group means are statistically different from the overall (sample) mean
- to test whether any of the group standard deviations are statistically different from the root mean square error (RMSE)
- to test whether any of the group ranges are statistically different from the overall mean of the ranges

Note: Within the Contingency platform, you can use the **Analysis of Means for Proportions** when the response has two categories. See "Contingency Analysis".

For a description of ANOM methods and to see how JMP implements ANOM, see the book by Nelson et al. (2005).

Note: If you specify a Block variable in the launch window and there are equal counts for each combination of Block and X variable level, the ANOM chart is the only Analysis of Means chart available. If you specify a Block variable in the launch window and there are unequal counts, none of the Analysis of Means options are available.

Analysis of Means for Location

You can test whether groups have a common mean or center value using the following options:

- ANOM
- ANOM with Transformed Ranks

ANOM

Use ANOM to compare group means to the overall mean. This method assumes that your data are approximately normally distributed. See "Example of an Analysis of Means Chart".

ANOM with Transformed Ranks

This is the nonparametric version of the **ANOM** analysis. Use this method if your data is clearly non-normal and cannot be transformed to normality. ANOM with Transformed Ranks compares each group's mean transformed rank to the overall mean transformed rank. The ANOM test involves applying the usual ANOM procedure and critical values to the transformed observations.

Transformed Ranks

Suppose that there are n observations. The transformed observations are computed as follows:

- Rank all observations from smallest to largest, accounting for ties. For tied observations, assign each one the average of the block of ranks that they share.
- Denote the ranks by R_1, R_2,..., R_n.
- The transformed rank corresponding to the i^{th} observations is:

$$\text{Transformed } R_i = Normal\ Quantile\left[\left(\frac{R_i}{2n+1}\right) + 0.5\right]$$

The ANOM procedure is applied to the values Transformed R_i. Since the ranks have a uniform distribution, the transformed ranks have a folded normal distribution. See Nelson et al. (2005).

Analysis of Means for Scale

You can test for homogeneity of variation within groups using the following options:

- ANOM for Variances
- ANOM for Variances with Levene (ADM)
- ANOM for Ranges

ANOM for Variances

Use this method to compare group standard deviations (or variances) to the root mean square error (or mean square error). This method assumes that your data is approximately normally distributed. To use this method, each group must have at least four observations. For more information about the ANOM for Variances test, see Wludyka and Nelson (1997) and Nelson et al. (2005). For an example, see "Example of an Analysis of Means for Variances Chart".

ANOM for Variances with Levene (ADM)

This method provides a robust test that compares the group means of the *absolute deviations from the median* (ADM) to the overall mean ADM. Use ANOM for Variances with Levene (ADM) if you suspect that your data is non-normal and cannot be transformed to normality. ANOM for Variances with Levene (ADM) is a nonparametric analog of the ANOM for Variances analysis. For more information about the ANOM for Variances with Levene (ADM) test, see Levene (1960) or Brown and Forsythe (1974).

ANOM for Ranges

Use this test to compare group ranges to the mean of the group ranges. This is a test for scale differences based on the range as the measure of spread. See Wheeler (2003).

Note: ANOM for Ranges is available only for balanced designs and specific group sizes. See "Restrictions for ANOM for Ranges Test".

Restrictions for ANOM for Ranges Test

Unlike the other ANOM decision limits, the decision limits for the ANOM for Ranges chart uses only tabled critical values. For this reason, ANOM for Ranges is available only for the following:

- groups of equal sizes
- groups specifically of the following sizes: 2–10, 12, 15, and 20
- number of groups between 2 and 30
- alpha levels of 0.10, 0.05, and 0.01

Analysis of Means Charts

Each Analysis of Means Methods option adds a decision chart to the report window that shows the following:

- an upper decision limit (UDL)
- a lower decision limit (LDL)

- a horizontal (center) line that falls between the decision limits; the center line position is determined by the chart type:
  - ANOM: the overall mean
  - ANOM with Transformed Ranks: the overall mean of the transformed ranks
  - ANOM for Variances: the root mean square error (or MSE when in variance scale)
  - ANOM for Variances with Levene (ADM): the overall absolute deviation from the mean
  - ANOM for Ranges: the mean of the group ranges

If a group's plotted statistic falls outside of the decision limits, then the test indicates that there is a statistical difference between that group's statistic and the overall average of the statistic for all the groups.

Analysis of Means Options

Each Analysis of Means Methods option adds an Analysis of Means red triangle menu to the report window.

Set Alpha Level Select an option from the most common alpha levels or specify any level with the **Other** selection. Changing the alpha level modifies the upper and lower decision limits.

Note: For ANOM for Ranges, only the selections 0.10, 0.05, and 0.01 are available.

Show Summary Report The reports are based on the Analysis of Means method:

- For ANOM, creates a report showing group means and decision limits.
- For ANOM with Transformed Ranks, creates a report showing group mean transformed ranks and decision limits.
- For ANOM for Variances, creates a report showing group standard deviations (or variances) and decision limits.
- For ANOM for Variances with Levene (ADM), creates a report showing group mean ADMs and decision limits.
- For ANOM for Ranges, creates a report showing group ranges and decision limits.

Graph in Variance Scale (Available only for **ANOM for Variances**.) Changes the scale of the vertical axis from standard deviations to variances.

Display Options Contains the following options to customize the display:

Show Decision Limits Shows or hides decision limit lines.

Show Decision Limit Shading Shows or hides decision limit shading.

Show Center Line Shows or hides the center line statistic.

Point Options: Show Needles Shows the needles. This is the default option. **Show Connected Points** shows a line connecting the means for each group. **Show Only Points** shows only the points representing the means for each group.

Compare Means

Use the Compare Means options to perform multiple comparisons of group means. Methods include the Tukey-Kramer all pairs test and the Dunnett's multiple comparison test with a control. All of these methods use pooled variance estimates for the means. Each Compare Means option, except for the Each Pair Stepwise, Newman-Keuls method, adds comparison circles next to the plot and specific reports to the report window. For more information about comparison circles, see "Using Comparison Circles".

Note: Another method for comparing means is ANOM. See "Analysis of Means Methods".

Option	Description	Nonparametric Menu Option
Each Pair, Student's t	Computes individual pairwise comparisons using Student's *t* tests. If you make many pairwise tests, there is no protection across the inferences. Therefore, the alpha-size (Type I error rate) across the hypothesis tests is higher than that for individual tests. See "Each Pair, Student's t".	**Nonparametric > Nonparametric Multiple Comparisons > Wilcoxon Each Pair**
All Pairs, Tukey HSD	Shows a test that is sized for all differences among the means. This is the *Tukey* or *Tukey-Kramer* HSD (honestly significant difference) test (Tukey 1953; Kramer 1956). This test is an exact alpha-level test if the sample sizes are the same, and conservative if the sample sizes are different (Hayter 1984). See "All Pairs, Tukey HSD".	**Nonparametric > Nonparametric Multiple Comparisons > Steel-Dwass All Pairs**

Option	Description	Nonparametric Menu Option
With Best, Hsu MCB	Tests whether the means are less than the unknown maximum or greater than the unknown minimum. This is the Hsu MCB test (Hsu 1996; Hsu 1981). See "With Best, Hsu MCB".	none
With Control, Dunnett's	Tests whether the means are different from the mean of a control group. This is Dunnett's test (Dunnett 1955). See "With Control, Dunnett's".	**Nonparametric > Nonparametric Multiple Comparisons > Steel With Control**
Each Pair Stepwise, Newman-Keuls	Tests whether there are differences between the means using the Studentized range test in a stepwise procedure. This is the Newman-Keuls or Student-Newman-Keuls method (Keuls, 1952). This test is less conservative than a Tukey HSD test. See "Each Pair Stepwise, Newman-Keuls".	none

Note: If you have specified a **Block** column, then the multiple comparison methods are performed on data that has been adjusted for the Block means.

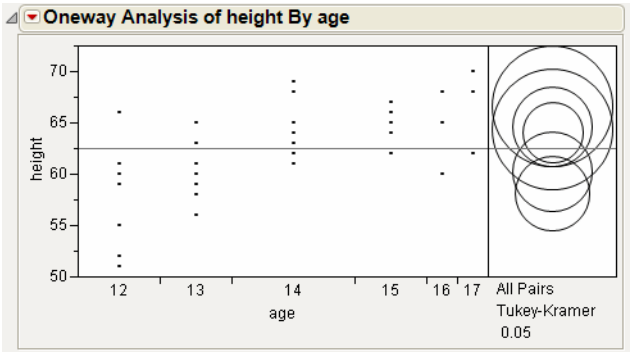
For an example showing all of these tests, see "Example Contrasting Four Compare Means Tests".

Using Comparison Circles

Each multiple comparison test, except for the Each Pair Stepwise, Newman-Keuls method, begins with a *comparison circles* plot, which is a visual representation of group mean comparisons. Figure 6.10 shows the comparison circles for the All Pairs, Tukey HSD method. Other comparison tests lengthen or shorten the radii of the circles.

Note: To permanently hide the comparison circles plot, select **File** > **Preferences** > **Platforms** > **Oneway** and deselect the **Comparison Circles** option.

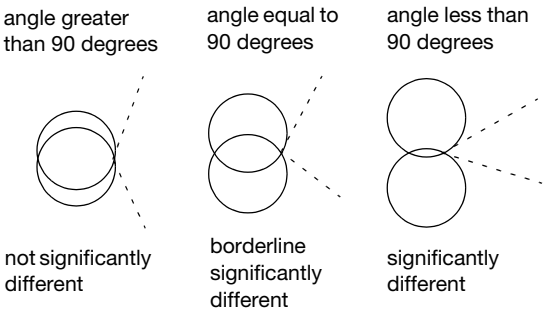
Figure 6.10 Visual Comparison of Group Means



Compare each pair of group means visually by examining the intersection of the comparison circles. The outside angle of intersection tells you whether the group means are significantly different (Figure 6.11).

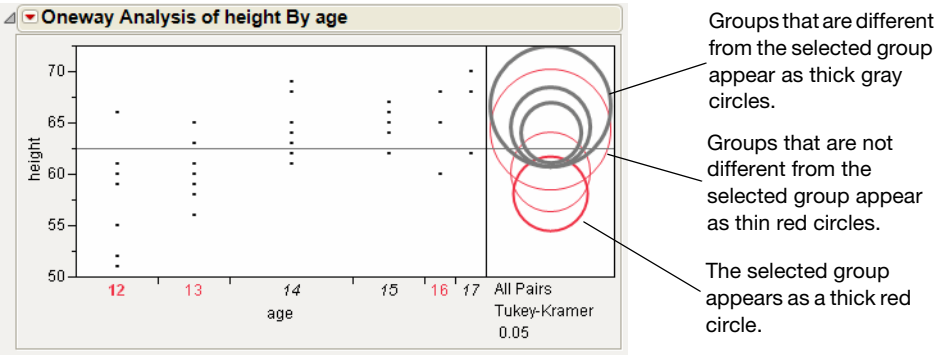
- Circles for means that are significantly different either do not intersect, or intersect slightly, so that the outside angle of intersection is less than 90 degrees.
- If the circles intersect by an angle of more than 90 degrees, or if they are nested, the means are not significantly different.

Figure 6.11 Angles of Intersection and Significance



If the intersection angle is close to 90 degrees, you can verify whether the means are significantly different by clicking on the comparison circle to select it (Figure 6.12). To deselect circles, click in the white space outside the circles.

Figure 6.12 Highlighting Comparison Circles



Related Information

- "Comparison Circles"

Each Pair, Student's t

The **Each Pair, Student's t** test shows the Student's *t* test for each pair of group levels and tests only individual comparisons. For an example of this test, see "Example of the Each Pair, Student's t Test".

All Pairs, Tukey HSD

The **All Pairs, Tukey HSD** test (also called Tukey-Kramer) protects the significance tests of all combinations of pairs, and the HSD intervals become greater than the Student's *t* pairwise LSDs. Graphically, the comparison circles become larger and differences are less significant.

The q statistic is calculated as q* = (1/sqrt(2)) * q where q is the required percentile of the studentized range distribution. For an example of this test, see "Example of the All Pairs, Tukey HSD Test".

With Best, Hsu MCB

The **With Best, Hsu MCB** test determines whether the mean for a given level exceeds the maximum mean of the remaining levels, or is smaller than the minimum mean of the remaining levels. See Hsu (1996). For an example of this test, see "Example of the With Best, Hsu MCB Test".

The quantiles for the Hsu MCB test vary by the level of the categorical variable. Unless the sample sizes are equal across levels, the comparison circle technique is not exact. The radius of a comparison circle is given by the standard error of the level multiplied by the largest quantile value. Use the *p*-values of the tests to obtain precise assessments of significant differences. See "Comparison with Max and Min".

Note: Means that are not regarded as the maximum or the minimum by MCB are also the means that are not contained in the selected subset of Gupta (1965) of potential maximums or minimum means.

Confidence Quantile

This report gives the quantiles for each level of the categorical variable. These correspond to the specified value of Alpha.

Comparison with Max and Min

The report shows *p*-values for one-sided Dunnett tests. For each level other than the best, the *p*-value given is for a test that compares the mean of the sample best level to the mean of each remaining level treated as a control (potentially best) level. The *p*-value for the sample best level is obtained by comparing the mean of the second sample best level to the mean of the sample best level treated as a control.

The report shows three columns.

Level The level of the categorical variable.

with Max p-Value For each level of the categorical variable, this column gives a *p*-value for a test that the mean of that level exceeds the maximum mean of the remaining levels. Use the tests in this column to screen out levels whose means are significantly smaller than or equal to the (unknown) largest true mean.

with Min p-Value For each level of the categorical variable, this column gives a *p*-value for a test that the mean of that level is smaller than the minimum mean of the remaining levels. Use the tests in this column to screen out levels whose means are significantly greater than or equal to the (unknown) smallest true mean.

LSD Threshold Matrix

The first report shown is for the maximum and the second is for the minimum.

For the *maximum* report, a column shows the row mean minus the column mean minus the LSD. If a value is positive, the row mean is significantly higher than the mean for the column, and the mean for the column is not the maximum.

For the *minimum* report, a column shows the row mean minus the column mean plus the LSD. If a value is negative, the row mean is significantly less than the mean for the column, and the mean for the column is not the minimum.

With Control, Dunnett's

The **With Control, Dunnett's** test compares a set of means against the mean of a control group. The LSDs that it produces are between the Student's *t* and Tukey-Kramer LSDs, because they are sized to refrain from an intermediate number of comparisons. For an example of this test, see "Example of the With Control, Dunnett's Test".

In the Dunnett's report, the $|d|$ quantile appears, and can be used in a manner similar to a Student's *t*-statistic. The LSD threshold matrix shows the absolute value of the difference minus the LSD. If a value is positive, its mean is more than the LSD apart from the control group mean and is therefore significantly different.

Each Pair Stepwise, Newman-Keuls

The **Each Pair Stepwise, Newman-Keuls** test compares the sample means using an iterative, stepwise procedure. At each iteration, Tukey's HSD test is used to test the difference between two group means. For an example of this test, see "Example of the Each Pair Stepwise, Newman-Keuls Test".

Caution: The Newman-Keuls test does not control the family-wise error rate. Use caution when interpreting the results of this procedure.

The following procedure is used for testing *J* group means:

Define the following:

J = number of groups (sorted in ascending order of group means)

N = number of observations

d = degrees of freedom, calculated as *N* - *J*

i = index of smallest group mean involved in a comparison

j = index of largest group mean involved in a comparison

k = minimum value of *j* in any comparison during the procedure

At the beginning of the procedure, set $i = 1$, $j = J$, and $k = 2$.

1. Perform Tukey's HSD test for groups *i* and *j*, where the number of groups for finding the appropriate quantile equals *j* - i + 1.
   – If the test is significant, groups *i* and *j* are determined to be significantly different. Decrease *j* by 1.

If this causes j to be less than k, then increase i by 1, set $k = \max(i, j) + 1$, set $j = J$, and continue to step 2.

If this causes j to be greater than or equal to k, then continue to step 2.

– If the test is not significant, groups i and j are not determined to be significantly different. Increase i by 1, set $k = j + 1$, set $j = J$, and continue to step 2.

2. Determine whether the procedure continues or stops based on the value of k.
   – If k is greater than i and k is less than or equal to J, repeat step 1.
   – If k is less than or equal to i or k is greater than J, stop the procedure. Any remaining untested ranges are deemed not to be significantly different.

The quantile used for Tukey's HSD is different for each test and is based on the number of group means between the sorted means being tested. In the Newman-Keuls report, the Smallest Quantile Considered (labeled Smallest q*) is the smallest studentized range quantile used in the above procedure divided by the square root of 2.

The test results are reported in the Connecting Letters Report.

For more information about the Newman-Keuls test, see Howell (2013).

Note: There are no mean circles added to the Comparison Circles graph when you use the **Each Pair Stepwise, Newman-Keuls** test. This is because each comparison has a different cut-off depending on the number of means between the two means being tested. Therefore, each circle would be a different size.

Compare Means Options

The Means Comparisons reports contain a red triangle menu with the following options:

Difference Matrix Shows or hides a table of all differences of means.

Confidence Quantile Shows or hides the critical value(s) and significance level (α) used for the means comparison procedure.

LSD Threshold Matrix (Not available for the Newman-Keuls test.) Shows or hides a matrix of pairwise differences of means minus the least significant difference for those means. A positive value indicates a pair of means that are significantly different.

Connecting Letters Report (Available only for the Student's t, Tukey's HSD, and the Newman-Keuls tests.) Shows or hides the traditional letter-coded report where means that do not share a letter are significantly different.

Ordered Differences Report (Available only for the Student's t and Tukey's HSD tests.) Shows or hides all pairwise positive-side differences, standard error of the difference, confidence intervals, p-values, and a plot of the magnitude of the difference with overlaid

confidence intervals. Confidence intervals that do not fully contain their corresponding difference bar indicate means that are significantly different from each other.

Detailed Comparisons Report (Available only for the Student's *t* test.) Shows or hides a detailed report for each comparison. Each section shows the difference between the levels, standard error and confidence intervals, *t*-ratios, *p*-values, and degrees of freedom. A plot illustrating the comparison appears on the right of each report.

Nonparametric Tests

Nonparametric tests are useful when the usual analysis of variance assumption of normality is not viable. The Nonparametric options provide several methods for testing the hypothesis of equal means or medians across groups. Nonparametric multiple comparison procedures are also available to control the overall error rate for pairwise comparisons. Nonparametric tests use functions of the response ranks, called rank scores. See Hajek (1969) and SAS Institute Inc. (2020a).

Notes:

- For the Wilcoxon, Median, Van der Waerden, and Friedman Rank tests, if the X factor has more than two levels, a chi-square approximation to the one-way test is performed.
- If you specify a Block variable in the launch window and there are equal counts for each combination of Block and X variable level, the Friedman Rank Test is the only Nonparametric option available. If you specify a Block variable in the launch window and there are unequal counts, none of the Nonparametric options are available.

Wilcoxon Test Performs a test based on Wilcoxon rank scores. The Wilcoxon rank scores are the simple ranks of the data. The Wilcoxon test is the most powerful rank test for errors with logistic distributions. If the factor has more than two levels, the Kruskal-Wallis test is performed. For information about the report, see "The Wilcoxon, Median, Van der Waerden, and Friedman Rank Test Reports". For an example, see "Example of the Nonparametric Wilcoxon Test".

The Wilcoxon test is also called the Mann-Whitney test.

Median Test Performs a test based on Median rank scores. The Median rank scores are either 1 or 0, depending on whether a rank is above or below the median rank. The Median test is the most powerful rank test for errors with double-exponential distributions. For information about the report, see "The Wilcoxon, Median, Van der Waerden, and Friedman Rank Test Reports".

van der Waerden Test Performs a test based on Van der Waerden rank scores. The Van der Waerden rank scores are the ranks of the data divided by one plus the number of observations transformed to a normal score by applying the inverse of the normal

distribution function. The Van der Waerden test is the most powerful rank test for errors with normal distributions. For information about the report, see "The Wilcoxon, Median, Van der Waerden, and Friedman Rank Test Reports".

Kolmogorov Smirnov Test (Available only when the X factor has two levels.) Performs a test based on the empirical distribution function, which tests whether the distribution of the response is the same across the groups. Both an approximate and an exact test are given. For information about the report, see "Kolmogorov-Smirnov Two-Sample Test Report".

Friedman Rank Test (Available only when a Block variable is specified in the launch window.) Performs a test based on Friedman Rank scores. The Friedman Rank scores are the ranks of the data within each level of the blocking variable. The parametric version of this test is a repeated measures ANOVA. For information about the report, see "The Wilcoxon, Median, Van der Waerden, and Friedman Rank Test Reports".

Note: There must be an equal number of observations within each block.

JMP PRO **Exact Test** Provides options for performing exact versions of the Wilcoxon, Median, van der Waerden, and Kolmogorov-Smirnov tests. These options are available only when the X factor has two levels. Results for both the approximate and the exact test are given.

For information about the report, see "2-Sample, Exact Test". For an example involving the Wilcoxon Exact Test, see "Example of the Nonparametric Wilcoxon Test".

The Wilcoxon, Median, Van der Waerden, and Friedman Rank Test Reports

For each test, the report shows the descriptive statistics followed by the test results. Test results appear in the 1-Way Test, ChiSquare Approximation report and, if the X variable has exactly two levels, a 2-Sample Test, Normal Approximation report also appears. The descriptive statistics are the following:

Level The levels of X.

Count The frequencies of each level.

Score Sum The sum of the rank score for each level.

Expected Score The expected score under the null hypothesis that there is no difference among class levels.

Score Mean The mean rank score for each level.

(Mean-Mean0)/Std0 The standardized score. Mean0 is the mean score expected under the null hypothesis. Std0 is the standard deviation of the score sum expected under the null

hypothesis. The null hypothesis is that the group means or medians are in the same location across groups.

2-Sample Test, Normal Approximation

When you have exactly two levels of X, a 2-Sample Test, Normal Approximation report appears. This report gives the following:

S Gives the sum of the rank scores for the level with the smaller number of observations.

Z Gives the test statistic for the normal approximation test. See "Two-Sample Normal Approximations".

Prob>|Z| Gives the *p*-value, based on a standard normal distribution, for the normal approximation test.

1-Way Test, ChiSquare Approximation

This report gives results for a chi-square test for location. See Conover (1999).

ChiSquare Gives the values of the chi-square test statistic. See "One-Way ChiSquare Approximations".

DF Gives the degrees of freedom for the test.

Prob>ChiSq Gives the *p*-value for the test. The *p*-value is based on a ChiSquare distribution with degrees of freedom equal to the number of levels of X minus 1.

JMP PRO 2-Sample, Exact Test

If your data are sparse, skewed, or heavily tied, exact tests might be more suitable than approximations based on asymptotic behavior. When you have exactly two levels of X, JMP Pro computes test statistics for exact tests. Select **Nonparametric > Exact Test** and select the test of your choice. A 2-Sample: Exact Test report appears. This report gives the following:

S Gives the sum of the rank scores for the observations in the smaller group. If the two levels of X have the same numbers of observations, then the value of S corresponds to the last level of X in the value ordering.

Prob ≤ S Gives a one-sided *p*-value for the test.

Prob ≥ |S-Mean| Gives a two-sided *p*-value for the test.

Kolmogorov-Smirnov Two-Sample Test Report

The Kolmogorov-Smirnov test is available only when X has exactly two levels. The report shows descriptive statistics followed by test results. The descriptive statistics are the following:

Level The two levels of X.

Count The frequencies of each level.

EDF at Maximum For a level of X, gives the value of the empirical cumulative distribution function (EDF) for that level at the value of X for which the difference between the two EDFs is a maximum. For the row named Total, gives the value of the pooled EDF (the EDF for the entire data set) at the value of X for which the difference between the two EDFs is a maximum.

Deviation from Mean at Maximum For each level, gives the value obtained by the following steps:

- Compute the difference between the EDF at Maximum for the given level and the EDF at maximum for the pooled data set (Total).
- Multiply this difference by the square root of the number of observations in that level, given as Count.

Kolmogorov-Smirnov Asymptotic Test

This report gives the details for the test.

KS A Kolmogorov-Smirnov statistic computed as follows:

$$\text{KS} = \max_j \sqrt{\frac{1}{n}\sum_i n_i (F_i(x_j) - F(x_j))^2}$$

The formula uses the following notation:

- x_j, $j = 1,\ldots, n$ are the observations
- n_i is the number of observations in the ith level of X
- F is the pooled cumulative empirical distribution function
- F_i is the cumulative empirical distribution function for the i^{th} level of X

This version of the Kolmogorov-Smirnov statistic applies even when there are more than two levels of X. Note, however, that JMP performs the Kolmogorov-Smirnov analysis only when X has only two levels of X.

KSa An asymptotic Kolmogorov-Smirnov statistic computed as $\text{KS}\sqrt{n}$, where n is the

total number of observations.

D=max|F1-F2| The maximum absolute deviation between the EDFs for the two levels. This is the version of the Kolmogorov-Smirnov statistic typically used to compare two samples.

Prob > D The *p*-value for the test. This is the probability that D exceeds the computed value under the null hypothesis of no difference between the levels.

D+ = max(F1-F2) A one-sided test statistic for the alternative hypothesis that the level of the first group exceeds the level of the second group.

Prob > D+ The *p*-value for the test of D+.

D- = max(F2-F1) A one-sided test statistic for the alternative hypothesis that the level of the second group exceeds the level of the first group

Prob > D- The *p*-value for the test of D-.

JMP PRO Kolmogorov-Smirnov Exact Test

For the Kolmogorov-Smirnov exact test, the report gives the same statistics as does the asymptotic test, but the *p*-values are computed to be exact.

Nonparametric Multiple Comparisons

This option provides several methods for performing nonparametric multiple comparisons. These tests are based on ranks and, except for the Wilcoxon Each Pair test, control for the overall experimentwise error rate. For more information about these tests, see See Dunn (1964) and Hsu (1996). For information about the reports, see "Nonparametric Multiple Comparisons Procedures".

Nonparametric Multiple Comparisons Procedures

Wilcoxon Each Pair Performs the Wilcoxon test on each pair. This procedure does not control for the overall alpha level. This is the nonparametric version of the **Each Pair, Student's t** option found on the Compare Means menu. See "Wilcoxon Each Pair, Steel-Dwass All Pairs, and Steel with Control".

Steel-Dwass All Pairs Performs the Steel-Dwass test on each pair. This is the nonparametric version of the **All Pairs, Tukey HSD** option found on the Compare Means menu. See "Wilcoxon Each Pair, Steel-Dwass All Pairs, and Steel with Control".

Steel With Control Compares each level to a control level. This is the nonparametric version of the **With Control, Dunnett's** option found on the Compare Means menu. See "Wilcoxon Each Pair, Steel-Dwass All Pairs, and Steel with Control".

Dunn All Pairs for Joint Ranks Performs a comparison of each pair, similar to the Steel-Dwass All Pairs option. The Dunn method computes ranks for all the data, not just the pair being compared. The reported *p*-value reflects a Bonferroni adjustment. It is the unadjusted *p*-value multiplied by the number of comparisons. If the adjusted *p*-value exceeds 1, it is reported as 1. See "Dunn All Pairs for Joint Ranks and Dunn with Control for Joint Ranks".

Dunn With Control for Joint Ranks Compares each level to a control level, similar to the Steel With Control option. The Dunn method computes ranks for all the data, not just the pair being compared. The reported *p*-value reflects a Bonferroni adjustment. It is the unadjusted *p*-value multiplied by the number of comparisons. If the adjusted *p*-value exceeds 1, it is reported as 1. See "Dunn All Pairs for Joint Ranks and Dunn with Control for Joint Ranks".

Wilcoxon Each Pair, Steel-Dwass All Pairs, and Steel with Control

The reports for these multiple comparison procedures give test results and confidence intervals. For these tests, observations are ranked within the sample obtained by combining only the two levels used in a given comparison.

q* The quantile used in computing the confidence intervals.

Alpha The alpha level used in computing the confidence interval. You can change the confidence level by selecting the Set α Level option from the Oneway menu.

Level The first level of the X variable used in the pairwise comparison.

- Level The second level of the X variable used in the pairwise comparison.

Score Mean Difference The mean of the rank score of the observations in the first level (Level) minus the mean of the rank scores of the observations in the second level (-Level), where a continuity correction is applied.

Denote the number of observations in the first level by n_1 and the number in the second level by n_2. The observations are ranked within the sample consisting of these two levels. Tied ranks are averaged. Denote the sum of the ranks for the first level by ScoreSum_1 and for the second level by ScoreSum_2.

If the difference in mean scores is positive, then the Score Mean Difference is defined as follows:

$$\text{Score Mean Difference} = (\text{ScoreSum}_1 - 0.5)/n_1 - (\text{ScoreSum}_2 + 0.5)/n_2$$

If the difference in mean scores is negative, then the Score Mean Difference is defined as follows:

$$\text{Score Mean Difference} = (\text{ScoreSum}_1 + 0.5)/n_1 - (\text{ScoreSum}_2 - 0.5)/n_2$$

Std Error Dif The standard error of the Score Mean Difference.

Z The standardized test statistic, which has an asymptotic standard normal distribution under the null hypothesis of no difference in means.

p-Value The *p*-value for the asymptotic test based on Z.

Hodges-Lehmann The Hodges-Lehmann estimator of the location shift. All paired differences consisting of observations in the first level minus observations in the second level are constructed. The Hodges-Lehmann estimator is the median of these differences. The Difference Plot bar chart shows the size of the Hodges-Lehmann estimate.

Lower CL The lower confidence limit for the Hodges-Lehmann statistic.

Note: Not computed if group sample sizes are large enough to cause memory issues.

Upper CL The upper confidence limit for the Hodges-Lehmann statistic.

Note: Not computed if group sample sizes are large enough to cause memory issues.

Dunn All Pairs for Joint Ranks and Dunn with Control for Joint Ranks

These comparison procedures are based on the rank of an observation in the entire data set. For the Dunn with Control for Joint Ranks tests, you must select a control level.

Level The first level of the X variable used in the pairwise comparison.

- Level The second level of the X variable used in the pairwise comparison.

Score Mean Difference The mean of the rank score of the observations in the first level (Level) minus the mean of the rank scores of the observations in the second level (-Level), where a continuity correction is applied. The ranks are obtained by ranking the observations within the entire sample. Tied ranks are averaged. The continuity correction is described in "Score Mean Difference".

Std Error Dif The standard error of the Score Mean Difference.

Z The standardized test statistic, which has an asymptotic standard normal distribution under the null hypothesis of no difference in means.

p-Value The *p*-value for the asymptotic test based on Z.

Unequal Variances

JMP provides four tests for equality of group variances. When the variances across groups are not equal, the usual assumptions for analysis of variance are not satisfied. For example, the ANOVA *F* test is not valid and an analysis that does not assume equal group variances should be used. The concept behind the first three tests of equal variances is to perform an analysis of variance on a new response variable constructed to measure the spread in each group. The fourth test is Bartlett's test, which is similar to the likelihood ratio test under normal distributions. The Unequal Variances option is not available when a Block variable is specified in the launch window.

Note: Another method to test for unequal variances is ANOMV. See "Analysis of Means Methods".

The following Tests for Equal Variances are available:

O'Brien Constructs a dependent variable so that the group means of the new variable equal the group sample variances of the original response. An ANOVA on the O'Brien variable is actually an ANOVA on the group sample variances (O'Brien 1979; Olejnik and Algina 1987).

Brown-Forsythe Shows the *F* test from an ANOVA where the response is the absolute value of the difference of each observation and the group median (Brown and Forsythe 1974).

Levene Shows the *F* test from an ANOVA where the response is the absolute value of the difference of each observation and the group mean (Levene 1960). The spread is measured as $z_{ij} = |y_{ij} - \bar{y}_i|$ (as opposed to the SAS default $z_{ij}^2 = (y_{ij} - \bar{y}_i)^2$).

Bartlett Compares the weighted arithmetic average of the sample variances to the weighted geometric average of the sample variances. The geometric average is always less than or equal to the arithmetic average with equality holding only when all sample variances are equal. The more variation there is among the group variances, the more these two averages differ. A function of these two averages is created, which approximates a χ^2-distribution (or, in fact, an *F* distribution under a certain formulation). Large values correspond to large values of the arithmetic or geometric ratio, and therefore to widely varying group variances. Dividing the Bartlett Chi-square test statistic by the degrees of freedom gives the *F* value shown in the table. Bartlett's test is not very robust to violations of the normality assumption (Bartlett and Kendall 1946).

F Test 2-sided (Available only if there are two levels of the X variable.) If there are only two groups tested, then a standard *F* test for unequal variances is also performed. The *F* test is the ratio of the larger to the smaller variance estimate. The *p*-value from the *F* distribution is doubled to make it a two-sided test.

Note: If you have specified a **Block** column, then the variance tests are performed on data after it has been adjusted for the Block means.

See "Example of the Unequal Variances Option".

Tests That the Variances Are Equal Report

The Tests That the Variances Are Equal report shows the differences between group means to the grand mean and to the median, and gives a summary of testing procedures.

If the equal variances test reveals that the group variances are significantly different, use Welch's test instead of the regular ANOVA test. The Welch statistic is based on the usual ANOVA *F* test. However, the means are weighted by the reciprocal of the group mean variances (Welch 1951; Brown and Forsythe 1974; Asiribo and Gurland 1990). If there are only two levels, the Welch ANOVA is equivalent to an unequal variance *t* test.

Description of the Tests That the Variances Are Equal Report

Level Lists the factor levels occurring in the data.

Count Records the frequencies of each level.

Std Dev Records the standard deviations of the response for each factor level. The standard deviations are equal to the means of the O'Brien variable. If a level occurs only once in the data, no standard deviation is calculated.

MeanAbsDif to Mean Records the mean absolute difference of the response and group mean. The mean absolute differences are equal to the group means of the Levene variable.

MeanAbsDif to Median Records the absolute difference of the response and group median. The mean absolute differences are equal to the group means of the Brown-Forsythe variable.

Test Lists the names of the tests performed.

F Ratio Records a calculated *F* statistic for each test. See "Tests That the Variances Are Equal".

DFNum Records the degrees of freedom in the numerator for each test. If a factor has *k* levels, the numerator has *k* - 1 degrees of freedom. Levels occurring only once in the data are not used in calculating test statistics for O'Brien, Brown-Forsythe, or Levene. The numerator degrees of freedom in this situation is the number of levels used in calculations minus one.

DFDen Records the degrees of freedom used in the denominator for each test. For O'Brien, Brown-Forsythe, and Levene, a degree of freedom is subtracted for each factor level used

in calculating the test statistic. If a factor has k levels, the denominator degrees of freedom is n - k.

p-Value Probability of obtaining an F-ratio value larger than the one calculated if the variances are equal across all levels.

Note: A warning appears if any level of the X variable contains fewer than 5 observations. For more information about the performance of the above tests with small sample sizes, see Brown and Forsythe (1974) and Miller (1972).

Description of the Welch's Test Report

F Ratio Shows the F test statistic for the equal means test.

DFNum Records the degrees of freedom in the numerator of the test. If a factor has k levels, the numerator has k - 1 degrees of freedom. Levels occurring only once in the data are not used in calculating the Welch ANOVA. The numerator degrees of freedom in this situation is the number of levels used in calculations minus one.

DFDen Records the degrees of freedom in the denominator of the test. See "Tests That the Variances Are Equal".

Prob>F Probability of obtaining an F value larger than the one calculated if the means are equal across all levels. Observed significance probabilities of 0.05 or less are considered evidence of unequal means across the levels.

t Test Shows the relationship between the F ratio and the t Test. Calculated as the square root of the F ratio. Appears only if the X factor has two levels.

Equivalence Tests

Equivalence tests are useful when you want to detect differences in means that are of *practical* interest. You must specify a threshold difference for which smaller differences are considered practically equivalent. In other words, if two group means differ by this amount or less, you are willing to consider them equivalent.

See "Example of an Equivalence Test".

Equivalence Tests Report

Once you have specified the threshold value and variance assumption, the Equivalence Tests report appears. The full title of the report is Equivalence Tests with Pooled Variance or the Equivalence Tests with Unequal Variances, depending on the specified variance assumption. The specified threshold value implies upper and lower bounds, which are shown at the top of the report. The top of the report also contains the α level of the equivalence tests. The report consists of a table that contains the equivalence tests and a scatterplot that displays them. The equivalence tests and confidence intervals are based on Student's *t* critical values.

Tip: To change the α level of the equivalence tests, use the **Set α Level** option in the Oneway Analysis red triangle menu before you select the Equivalence Tests option.

The red triangle menu next to Equivalence Tests with Pooled Variance or Equivalence Tests with Unequal Variances contains the following options:

Equivalence TOST Tests Shows or hides the Equivalence TOST Tests report.

Equivalence Tests Scatterplot Shows or hides the Equivalence Tests Scatterplot report.

Equivalence Tests Pairwise Comparisons Shows or hides the Practical Equivalence reports for all pairwise comparisons.

Remove Removes the Equivalence Tests report from the Oneway Analysis report window.

Equivalence TOST Tests

The Two One-Sided Tests (TOST) method is used to test for a practical difference between the means (Schuirmann 1987). Two one-sided *t* tests using the specified variance assumption are constructed for the null hypotheses that the true difference exceeds the threshold values. If both tests reject, the difference in the means does not statistically exceed either threshold value. Therefore, the groups are considered practically equivalent. If only one or neither test rejects, then the groups might not be practically equivalent.

For each comparison, the Equivalence TOST Tests report gives the following information:

Difference Estimated difference in the means.

Std Error of Difference Estimated standard error of the difference in the means.

Lower Bound t Ratio, Upper Bound t Ratio Lower and upper bound *t* ratios for the two one-sided pooled-variance significance tests.

Lower Bound p-Value, Upper Bound p-Value *p*-values corresponding to the lower and upper bound *t* ratios.

Maximum p-Value Maximum of the lower and upper bound *p*-values.

Lower and Upper Limits for a 1–2α confidence interval for the difference in the means.

Equivalence Tests Scatterplot

Using colors, this scatterplot indicates which means are practically equivalent and which are not practically equivalent as determined by the equivalence test. This plot is sometimes called a *diffogram* or a mean-mean scatterplot.

The plot shows a solid reference line on the diagonal as well as a shaded reference band. The width of the band is twice the practical difference. The coordinates of the point on the line segment are the means for the corresponding groups. There is an implied third axis on the diagonal where each line segment corresponds to a 1–2α confidence interval for a pairwise comparison. Hover over one of these points to show a tooltip that indicates the groups being compared and the estimated difference. When a line segment is entirely contained within the diagonal band, it follows that the means are practically equivalent.

The Equivalence Tests Scatterplot has the following option:

Show Reference Lines Displays reference lines for the points on the scatterplot. This is not recommended if there are many points in the scatterplot. If there are many points, it is better to hover over the points to view the labels.

Robust

The **Robust** option provides two methods to reduce the influence of outliers or extreme data points in your data set: Robust Fit and Cauchy Fit. The Robust options are not available when a Block variable is specified in the launch window.

Robust Fit

The Robust Fit option reduces the influence of outliers in the response variable. The Huber M-estimation method is used. Huber M-estimation finds parameter estimates that minimize the Huber loss function:

$$l(\underset{\sim}{e}) = \sum_i \rho(e_i)$$

where

$$\rho(e) = \begin{cases} \frac{1}{2}e^2 & \text{if } |e| < k \\ k|e| - \frac{1}{2}k^2 & \text{if } |e| \geq k \end{cases}$$

e_i refers to the residuals

The Huber loss function penalizes outliers and increases as a quadratic for small errors and linearly for large errors. For more information about robust fitting, see Huber (1973) and Huber and Ronchetti (2009). See "Example of the Robust Fit Option".

Cauchy Fit

The Cauchy fit option assumes that the errors have a Cauchy distribution. A Cauchy distribution has fatter tails than the normal distribution, resulting in a reduced emphasis on outliers. This option can be useful if you have a large proportion of outliers in your data. However, if your data are close to normal with only a few outliers, this option can lead to incorrect inferences. The Cauchy option estimates parameters using maximum likelihood and a Cauchy link function.

Power

The **Power** option calculates statistical power and other details about a given hypothesis test. See "Example of the Power Option". For statistical details, see "Power".

- *LSV* (the Least Significant Value) is the value of some parameter or function of parameters that would produce a certain *p*-value alpha. Said another way, you want to know how small an effect would be declared significant at some *p*-value alpha. The LSV provides a measuring stick for significance on the scale of the parameter, rather than on a probability scale. It shows how sensitive the design and data are.
- *LSN* (the Least Significant Number) is the total number of observations that would produce a specified *p*-value alpha given that the data has the same form. The LSN is defined as the number of observations needed to reduce the variance of the estimates enough to achieve a significant result with the given values of alpha, sigma, and delta (the significance level, the standard deviation of the error, and the effect size). If you need more data to achieve significance, the LSN helps tell you how many more. The LSN is the total number of observations that yields approximately 50% power.

- *Power* is the probability of getting significance (p-value < alpha) when a real difference exists between groups. It is a function of the sample size, the effect size, the standard deviation of the error, and the significance level. The power tells you how likely your experiment is to detect a difference (effect size), at a given alpha level.

Note: When there are only two groups in a one-way layout, the LSV computed by the power facility is the same as the least significant difference (LSD) shown in the multiple-comparison tables.

Power Details Window and Reports

The Power Details window and reports are the same as those in the general fitting platform launched by the Fit Model platform. For more information about power calculation, see *Fitting Linear Models*.

For each of four columns Alpha, Sigma, Delta, and Number, fill in a single value, two values, or the start, stop, and increment for a sequence of values (Figure 6.32). Power calculations are performed on all possible combinations of the values that you specify.

Alpha (α) Significance level, between 0 and 1 (usually 0.05, 0.01, or 0.10). Initially, a value of 0.05 shows.

Sigma (σ) Standard error of the residual error in the model. Initially, RMSE, the estimate from the square root of the mean square error is supplied here.

Delta (δ) Raw effect size. For more information about effect size computations, see *Fitting Linear Models*. The first position is initially set to the square root of the sums of squares for the hypothesis divided by n (that is, $\delta = \sqrt{SS/n}$).

Number (n) Total sample size across all groups. Initially, the actual sample size is put in the first position.

Solve for Power Solves for the power (the probability of a significant result) as a function of all four values: α, σ, δ, and n.

Solve for Least Significant Number Solves for the number of observations needed to achieve approximately 50% power given α, σ, and δ.

Solve for Least Significant Value Solves for the value of the parameter or linear test that produces a p-value of α. This is a function of α, σ, n, and the standard error of the estimate. This feature is available only when the X factor has two levels and is usually used for individual parameters.

Adjusted Power and Confidence Interval When you look at power retrospectively, you use estimates of the standard error and the test parameters.

– Adjusted power is the power calculated from a more unbiased estimate of the non-centrality parameter.

– The confidence interval for the adjusted power is based on the confidence interval for the non-centrality estimate.

Adjusted power and confidence limits are computed only for the original Delta, because that is where the random variation is.

Normal Quantile Plot

You can create two types of normal quantile plots:

- **Plot Actual by Quantile** creates a plot of the response values versus the normal quantile values. The quantiles are computed and plotted separately for each level of the X variable.
- **Plot Quantile by Actual** creates a plot of the normal quantile values versus the response values. The quantiles are computed and plotted separately for each level of the X variable.

The **Line of Fit** option shows or hides the lines of fit on the quantile plots. The **Normal Quantile Label** option shows or hides labels inside the right vertical axis of the Normal Quantile Plot and inside the top axis of the Actual by Quantile plot. The labels represent the normal quantile scale.

The Normal Quantile Plot options are not available when a Block variable is specified in the launch window.

See "Example of a Normal Quantile Plot".

CDF Plot

The CDF plot shows the cumulative distribution function for all of the groups in the Oneway report. CDF plots are useful if you want to compare the distributions of the response across levels of the X factor. The CDF Plot option is not available when a Block variable is specified in the launch window. See "Example of a CDF Plot".

Densities

The **Densities** options provide several ways to visually compare the distribution and composition of the response across the levels of the X factor. There are three density options:

- **Compare Densities** shows a smooth curve estimating the density of each group. The smooth curve is the density estimate for each group.
- **Composition of Densities** shows the summed densities, weighted by each group's counts. At each X value, the Composition of Densities plot shows how each group contributes to the total.
- **Proportion of Densities** shows the contribution of the group as a proportion of the total at each X level.

The Densities options are not available when a Block variable is specified in the launch window.

See "Example of the Densities Options".

Matching Column

Use the **Matching Column** option to specify a matching (ID) variable for a matching model analysis. The **Matching Column** option addresses the case when the data in a one-way analysis come from matched (paired) data. Matched data can occur when observations in different groups come from the same participant. The Matching Column option is not available when a Block variable is specified in the launch window. See "Example of the Matching Column Option".

Note: A special case of matching leads to the paired *t* test. The **Matched Pairs** platform handles this type of data, but the data must be organized with the pairs in different columns, not in different rows.

The **Matching Column** option performs two primary actions:

- It fits an additive model (using an iterative proportional fitting algorithm) that includes both the grouping variable (the X variable in the Fit Y by X analysis) and the matching variable that you select. The iterative proportional fitting algorithm makes a difference if there are hundreds of participants, because the equivalent linear model would be very slow and would require huge memory resources.
- It draws lines between the points that match across the groups. If there are multiple observations with the same matching ID value, lines are drawn from the mean of the group of observations.

The **Matching Column** option automatically activates the **Matching Lines** option connecting the matching points. To turn the lines off, select **Display Options > Matching Lines**.

The Matching Fit report shows the effects with *F* tests. These are equivalent to the tests that you get with the Fit Model platform if you run two models, one with the interaction term and one without. If there are only two levels, then the *F* test is equivalent to the paired *t* test.

Note: For more information about the Fit Model platform, see *Fitting Linear Models*.

Additional Examples of the Oneway Platform

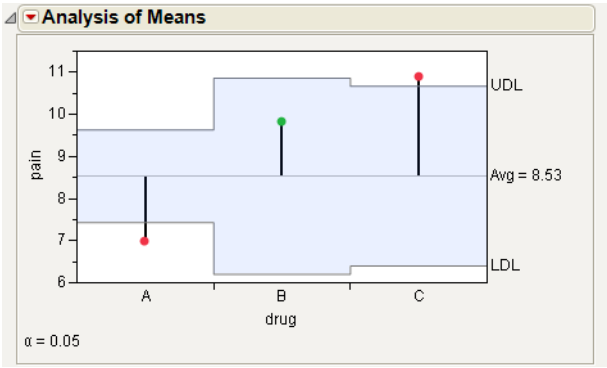
- "Example of an Analysis of Means Chart"
- "Example of an Analysis of Means for Variances Chart"
- "Example of the Each Pair, Student's t Test"
- "Example of the All Pairs, Tukey HSD Test"
- "Example of the With Best, Hsu MCB Test"
- "Example of the With Control, Dunnett's Test"
- "Example of the Each Pair Stepwise, Newman-Keuls Test"
- "Example Contrasting Four Compare Means Tests"
- "Example of the Nonparametric Wilcoxon Test"
- "Example of the Unequal Variances Option"
- "Example of an Equivalence Test"
- "Example of the Robust Fit Option"
- "Example of the Power Option"
- "Example of a Normal Quantile Plot"
- "Example of a CDF Plot"
- "Example of the Densities Options"
- "Example of the Matching Column Option"
- "Example of Stacking Data for a Oneway Analysis"

Example of an Analysis of Means Chart

1. Select **Help > Sample Data Library** and open Analgesics.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select pain and click **Y, Response**.
4. Select drug and click **X, Factor**.

5. Click **OK**.
6. Click the Analysis of Means red triangle and select **Analysis of Means Methods > ANOM**.

Figure 6.13 Example of Analysis of Means Chart



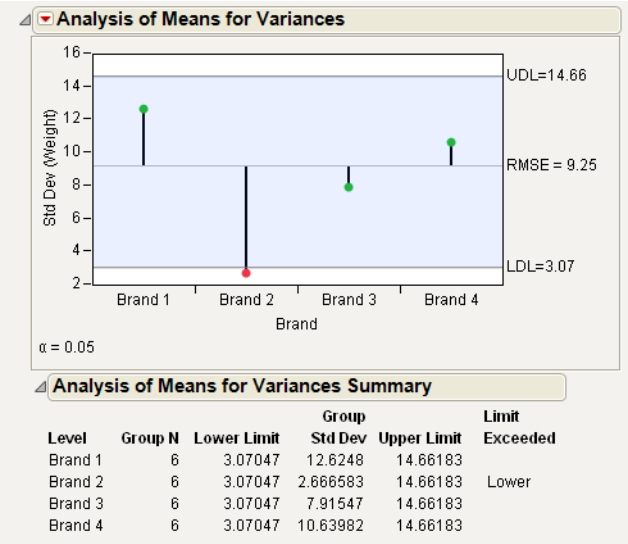
For the example, the means for drug A and C are statistically different from the overall mean. The drug A mean is lower and the drug C mean is higher. Note the decision limits for the drug types are not the same, due to different sample sizes.

Example of an Analysis of Means for Variances Chart

Use ANOMV to test if the variability is significantly different between brands of springs. Four different brands of springs were tested to see what weight is required to extend a spring 0.10 inches. Six springs of each brand were tested. The data was checked for normality, since the ANOMV test is not robust to non-normality. Examine the brands to determine whether the variability is significantly different between brands.

1. Select **Help > Sample Data Library** and open Spring Data.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Weight and click **Y, Response**.
4. Select Brand and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of Weight By Brand and select **Analysis of Means Methods > ANOM for Variances**.
7. Click the red triangle next to Analysis of Means for Variances and select **Show Summary Report**.

Figure 6.14 Example of Analysis of Means for Variances Chart



Analysis of Means for Variances Summary

Level	Group N	Lower Limit	Group Std Dev	Upper Limit	Limit Exceeded
Brand 1	6	3.07047	12.6248	14.66183	
Brand 2	6	3.07047	2.666583	14.66183	Lower
Brand 3	6	3.07047	7.91547	14.66183	
Brand 4	6	3.07047	10.63982	14.66183	

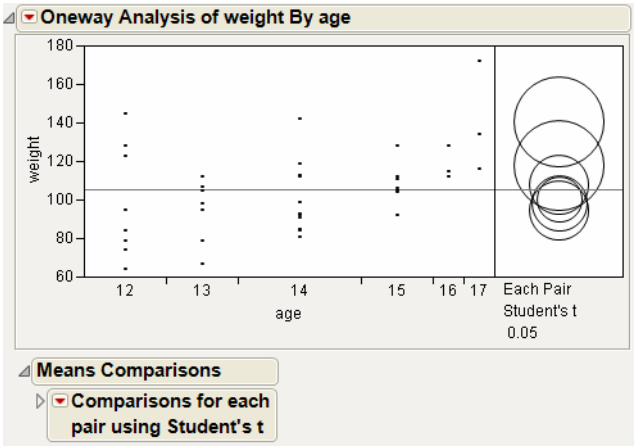
Note that the standard deviation for Brand 2 exceeds the lower decision limit. Therefore, Brand 2 has significantly lower variance than the other brands.

Example of the Each Pair, Student's t Test

This example illustrates the use of all possible *t* tests.

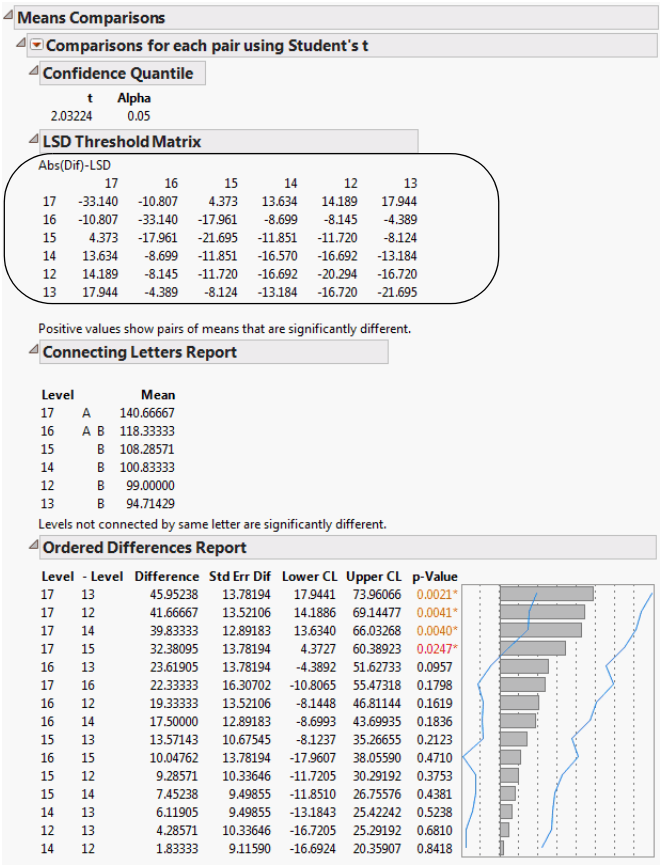
1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response.**
4. Select age and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of weight By age and select **Compare Means > Each Pair, Student's t**.

Figure 6.15 Example of Each Pair, Student's t Comparison Circles



The means comparison method can be thought of as seeing if the actual difference in the means is greater than the difference that would be significant. This difference is called the LSD (least significant difference). The LSD term is used for Student's *t* intervals and in context with intervals for other tests. In the comparison circles graph, the distance between the circles' centers represent the actual difference. The LSD is what the distance would be if the circles intersected at right angles.

Figure 6.16 Example of Means Comparisons Report for Each Pair, Student's *t*



Means Comparisons

Comparisons for each pair using Student's t

Confidence Quantile

t	Alpha
2.03224	0.05

LSD Threshold Matrix

Abs(Dif)-LSD

	17	16	15	14	12	13
17	-33.140	-10.807	4.373	13.634	14.189	17.944
16	-10.807	-33.140	-17.961	-8.699	-8.145	-4.389
15	4.373	-17.961	-21.695	-11.851	-11.720	-8.124
14	13.634	-8.699	-11.851	-16.570	-16.692	-13.184
12	14.189	-8.145	-11.720	-16.692	-20.294	-16.720
13	17.944	-4.389	-8.124	-13.184	-16.720	-21.695

Positive values show pairs of means that are significantly different.

Connecting Letters Report

Level		Mean
17	A	140.66667
16	A B	118.33333
15	B	108.28571
14	B	100.83333
12	B	99.00000
13	B	94.71429

Levels not connected by same letter are significantly different.

Ordered Differences Report

Level	- Level	Difference	Std Err Dif	Lower CL	Upper CL	p-Value
17	13	45.95238	13.78194	17.9441	73.96066	0.0021*
17	12	41.66667	13.52106	14.1886	69.14477	0.0041*
17	14	39.83333	12.89183	13.6340	66.03268	0.0040*
17	15	32.38095	13.78194	4.3727	60.38923	0.0247*
16	13	23.61905	13.78194	-4.3892	51.62733	0.0957
17	16	22.33333	16.30702	-10.8065	55.47318	0.1798
16	12	19.33333	13.52106	-8.1448	46.81144	0.1619
16	14	17.50000	12.89183	-8.6993	43.69935	0.1836
15	13	13.57143	10.67545	-8.1237	35.26655	0.2123
16	15	10.04762	13.78194	-17.9607	38.05590	0.4710
15	12	9.28571	10.33646	-11.7205	30.29192	0.3753
15	14	7.45238	9.49855	-11.8510	26.75576	0.4381
14	13	6.11905	9.49855	-13.1843	25.42242	0.5238
12	13	4.28571	10.33646	-16.7205	25.29192	0.6810
14	12	1.83333	9.11590	-16.6924	20.35907	0.8418

In Figure 6.16, the LSD threshold table shows the difference between the absolute difference in the means and the LSD (least significant difference). If the values are positive, the difference in the two means is larger than the LSD, and the two groups are significantly different.

Example of the All Pairs, Tukey HSD Test

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response.**
4. Select age and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of weight By age and select **Compare Means > All Pairs, Tukey HSD**.

Figure 6.17 Example of All Pairs, Tukey HSD Comparison Circles

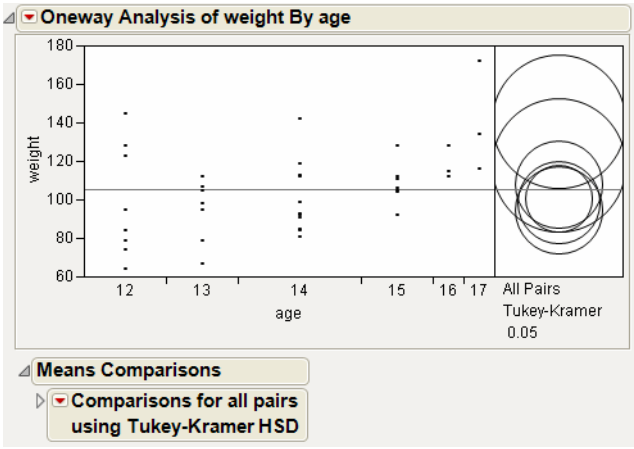


Figure 6.18 Example of Means Comparisons Report for All Pairs, Tukey HSD

Means Comparisons

Comparisons for all pairs using Tukey-Kramer HSD

Confidence Quantile

q*	Alpha
3.01825	0.05

LSD Threshold Matrix

Abs(Dif)-HSD

	17	16	15	14	12	13
17	-49.219	-26.885	-9.216	0.923	0.857	4.355
16	-26.885	-49.219	-31.550	-21.411	-21.477	-17.978
15	-9.216	-31.550	-32.221	-21.217	-21.912	-18.650
14	0.923	-21.411	-21.217	-24.609	-25.681	-22.550
12	0.857	-21.477	-21.912	-25.681	-30.140	-26.912
13	4.355	-17.978	-18.650	-22.550	-26.912	-32.221

Positive values show pairs of means that are significantly different.

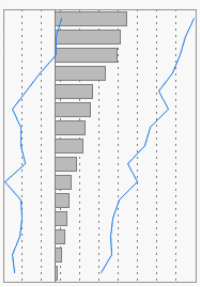
Connecting Letters Report

Level		Mean
17	A	140.66667
16	A B	118.33333
15	A B	108.28571
14	B	100.83333
12	B	99.00000
13	B	94.71429

Levels not connected by same letter are significantly different.

Ordered Differences Report

Level	- Level	Difference	Std Err Dif	Lower CL	Upper CL	p-Value
17	13	45.95238	13.78194	4.3551	87.54968	0.0233*
17	12	41.66667	13.52106	0.8568	82.47657	0.0431*
17	14	39.83333	12.89183	0.9226	78.74404	0.0423*
17	15	32.38095	13.78194	-9.2163	73.97825	0.2029
16	13	23.61905	13.78194	-17.9783	65.21635	0.5325
17	16	22.33333	16.30702	-26.8853	71.55192	0.7443
16	12	19.33333	13.52106	-21.4766	60.14323	0.7091
16	14	17.50000	12.89183	-21.4107	56.41071	0.7512
15	13	13.57143	10.67545	-18.6497	45.79256	0.7982
16	15	10.04762	13.78194	-31.5497	51.64492	0.9769
15	12	9.28571	10.33646	-21.9123	40.48369	0.9442
15	14	7.45238	9.49855	-21.2166	36.12134	0.9683
14	13	6.11905	9.49855	-22.5499	34.78801	0.9867
12	13	4.28571	10.33646	-26.9123	35.48369	0.9983
14	12	1.83333	9.11590	-25.6807	29.34736	1.0000



In Figure 6.18, the Tukey-Kramer HSD Threshold matrix shows the actual absolute difference in the means minus the HSD. This value represents the difference that would be significant. Pairs with a positive value are significantly different. The q* (appearing above the HSD Threshold Matrix table) is the quantile that is used to scale the HSDs. It has a computational role comparable to a Student's *t*.

Example of the With Best, Hsu MCB Test

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response.**
4. Select age and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of weight By age and select **Compare Means > With Best, Hsu MCB**.

Figure 6.19 Examples of With Best, Hsu MCB Comparison Circles

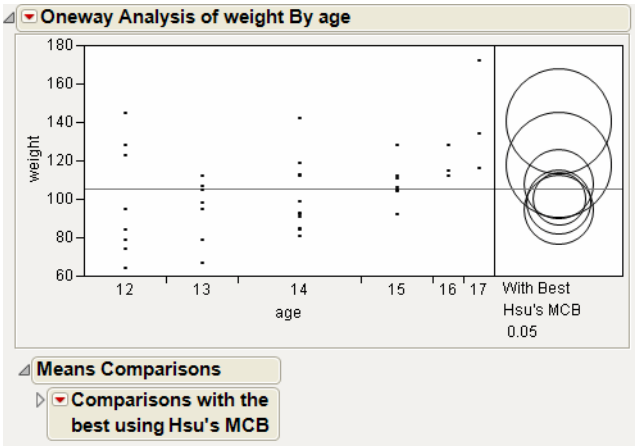


Figure 6.20 Example of Means Comparisons Report for With Best, Hsu MCB

Comparisons with the best using Hsu's MCB

Confidence Quantile

d	Alpha
2.23157	0.05
2.23157	
2.34056	
2.38159	
2.35278	
2.34056	

Comparisons with Max and Min

Level	with Max p-Value	with Min p-Value
17	0.9863	0.0039*
16	0.2219	0.1304
15	0.0491*	0.3172
14	0.0093*	0.6599
12	0.0091*	0.7216
13	0.0046*	0.9408

LSD Threshold Matrix

Mean[i]-Mean[j]-LSD

	17	16	15	14	12	13
17	-36.390	-14.057	0.124	9.130	9.855	13.695
16	-58.724	-36.390	-22.210	-13.203	-12.479	-8.638
15	-63.136	-40.803	-24.987	-15.169	-15.034	-11.415
14	-68.602	-46.269	-29.684	-19.418	-19.614	-16.113
12	-71.840	-49.507	-33.479	-23.544	-23.495	-19.907
13	-76.708	-54.374	-38.558	-28.741	-28.605	-24.987

If a column has any positive values, the mean is significantly less than the max.

Mean[i]-Mean[j]+LSD

	17	16	15	14	12	13
17	36.390	58.724	64.638	70.536	73.479	78.210
16	14.057	36.390	42.305	48.203	51.145	55.876
15	-1.626	20.708	24.987	30.074	33.605	38.558
14	-11.064	11.269	14.780	19.418	23.281	28.351
12	-11.493	10.840	14.907	19.877	23.495	28.479
13	-15.197	7.136	11.415	16.503	20.034	24.987

If a column has any negative values, the mean is significantly greater than the min.

The Comparison with Max and Min report compares the mean of each level to the maximum and the minimum of the means of the remaining levels. For example, the mean for age 15 differs significantly from the maximum of the means of the remaining levels. The mean for age 17 differs significantly from the minimum of the means of the remaining levels. The maximum mean could occur for age 16 or age 17, because neither mean differs significantly from the maximum mean. By the same reasoning, the minimum mean could correspond to any of the ages other than age 17.

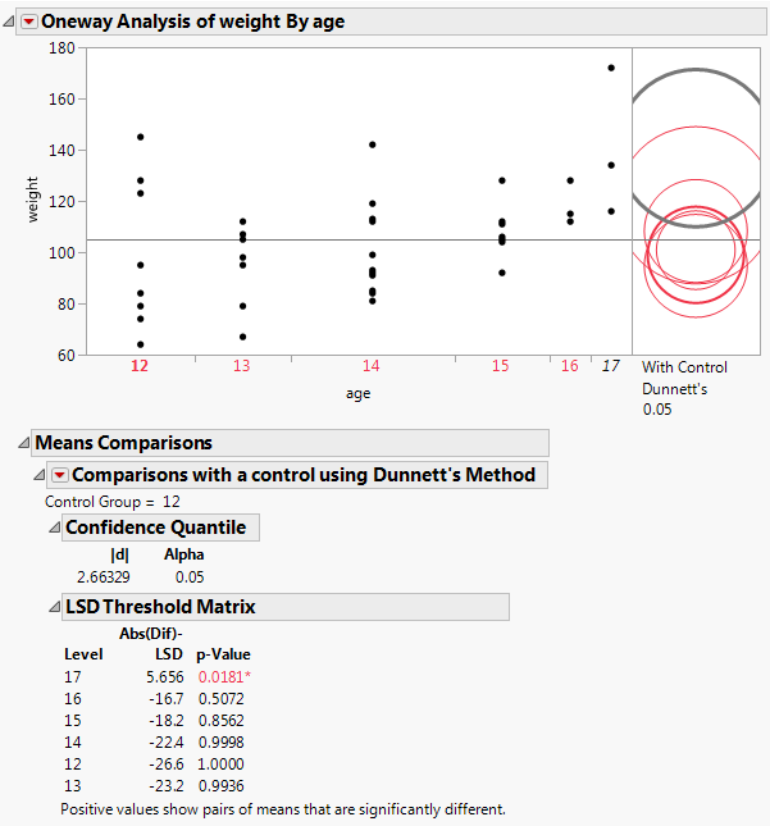
Example of the With Control, Dunnett's Test

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response.**

4. Select age and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of weight By age and select **Compare Means > With Control, Dunnett's**.
7. Select the group to use as the control group. In this example, select age 12.

   Alternatively, click a row to highlight it in the scatterplot before selecting the **Compare Means > With Control, Dunnett's** option. The test uses the selected row as the control group.
8. Click **OK**.

Figure 6.21 Example of With Control, Dunnett's Comparison Circles

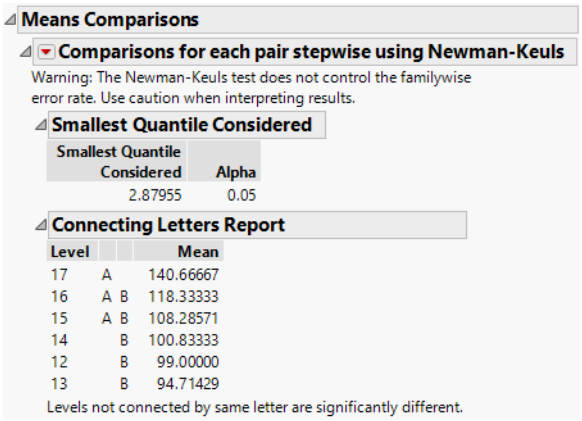


Using the comparison circles, you can conclude that level 17 is the only level that is significantly different from the control level of 12.

Example of the Each Pair Stepwise, Newman-Keuls Test

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response**.
4. Select age and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of weight By age and select **Compare Means > Each Pair Stepwise, Newman-Keuls**.

Figure 6.22 Example of Means Comparisons Report for Each Pair Stepwise, Newman-Keuls



Means Comparisons

Comparisons for each pair stepwise using Newman-Keuls

Warning: The Newman-Keuls test does not control the familywise error rate. Use caution when interpreting results.

Smallest Quantile Considered

Smallest Quantile Considered	Alpha
2.87955	0.05

Connecting Letters Report

Level			Mean
17	A		140.66667
16	A	B	118.33333
15	A	B	108.28571
14		B	100.83333
12		B	99.00000
13		B	94.71429

Levels not connected by same letter are significantly different.

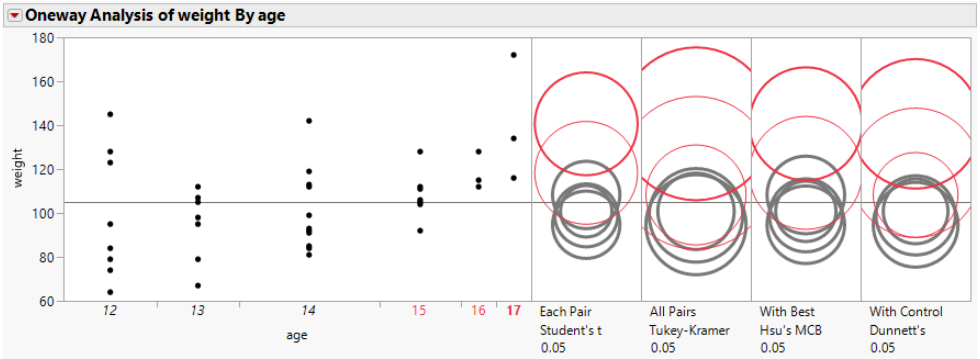
The Connecting Letters Report shows that Level 17 is significantly different from all other levels except 16 and 15.

Example Contrasting Four Compare Means Tests

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select weight and click **Y, Response.**
4. Select age and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of weight By age and select each one of the **Compare Means** options. For the With Control, Dunnett's option, select age 17 as the control group.

The four methods all test differences between group means. Each test is used for a specific hypothesis and different findings can occur.

Figure 6.23 Comparison Circles for Four Multiple Comparison Tests



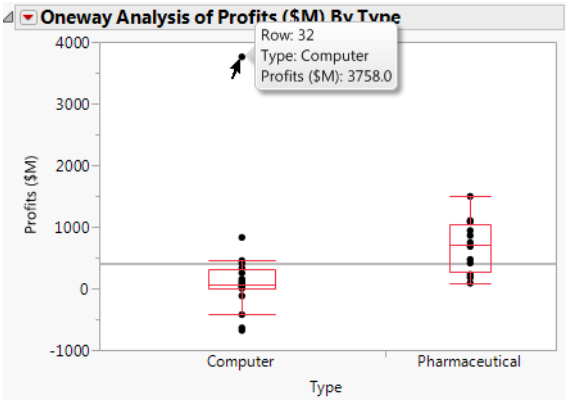
In Figure 6.23, age group 17 is highlighted. The other control circles are colored in relation to age group 17. Notice that for the Student's t and Hsu methods, age group 15 (the third circle from the top) is gray. This indicates that it is significantly different from age group 17. However, for the Tukey and Dunnett methods, age group 15 is red, which indicates that it is not significantly different from age group 17.

Example of the Nonparametric Wilcoxon Test

Use a Wilcoxon test to determine if the mean profit earned by companies differs by type of company. In Companies.jmp, the data consist of various metrics on two types of companies, Pharmaceutical (12 companies) and Computer (20 companies).

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Profits ($M) and click **Y, Response.**
4. Select Type and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of Profits ($M) By Type and select **Display Options > Box Plots**.

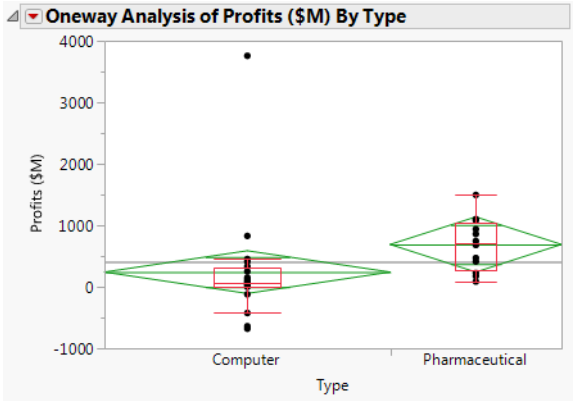
Figure 6.24 Computer Company Profit Distribution



The box plots suggest that the distributions are not normal or even symmetric. There is a very large value for the company in row 32 that might affect parametric tests.

7. Click the red triangle next to Oneway Analysis of Profits ($M) By Type and select **Means/ANOVA/Pooled t**.

Figure 6.25 Company Analysis of Variance

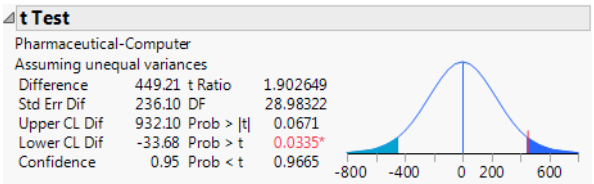


Source	DF	Sum of Squares	Mean Square	F Ratio	Prob > F
Type	1	1513422	1513422	2.6152	0.1163
Error	30	17360863	578695		
C. Total	31	18874286			

The *F* test shows no significance because the *p*-value is large ($p = 0.1163$). This might be due to the large value in row 32 and the possible violation of the normality assumption.

8. Click the red triangle next to Oneway Analysis of Profits ($M) By Type and select **t Test**.

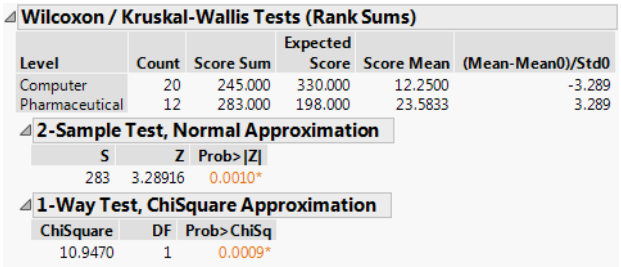
Figure 6.26 t Test Results



The Prob > |t| for a two-sided test is 0.0671. The *t* test does not assume equal variances, but the unequal variances *t* test is also a parametric test.

9. Click the red triangle next to Oneway Analysis of Profits ($M) By Type and select **Nonparametric > Wilcoxon Test**.

Figure 6.27 Wilcoxon Test Results



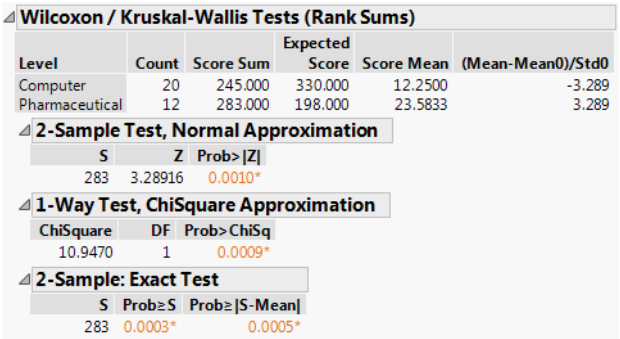
The Wilcoxon test is a nonparametric test. It is based on ranks, so it is resistant to outliers. Also, it does not require normality.

Both the normal and the chi-square approximations for the Wilcoxon test statistic indicate significance at a *p*-value of 0.0010. You conclude that there is a significant difference in the location of the distributions, and conclude that mean profit differs based on company type.

The normal and chi-square tests are based on the asymptotic distributions of the test statistics. If you have JMP Pro, you can conduct an exact test.

10. JMP PRO Click the red triangle next to Oneway Analysis of Profits ($M) By Type and select **Nonparametric > Exact Test > Wilcoxon Exact Test**.

Figure 6.28 Wilcoxon Exact Test Results



Wilcoxon / Kruskal-Wallis Tests (Rank Sums)

Level	Count	Score Sum	Expected Score	Score Mean	(Mean-Mean0)/Std0
Computer	20	245.000	330.000	12.2500	-3.289
Pharmaceutical	12	283.000	198.000	23.5833	3.289

2-Sample Test, Normal Approximation

S	Z	Prob>|Z|
283	3.28916	0.0010*

1-Way Test, ChiSquare Approximation

ChiSquare	DF	Prob>ChiSq
10.9470	1	0.0009*

2-Sample: Exact Test

S	Prob≥S	Prob≥|S-Mean|
283	0.0003*	0.0005*

The observed value of the test statistic is S = 283. This is the sum of the ranks for the level of Type with the smaller sample size (pharmaceuticals). The probability of observing an absolute difference from the mean midrank that exceeds the absolute value of S minus the mean of the midranks is 0.0005. This is a two-sided test for a difference in location and supports rejecting the hypothesis that profits do not differ by type of company.

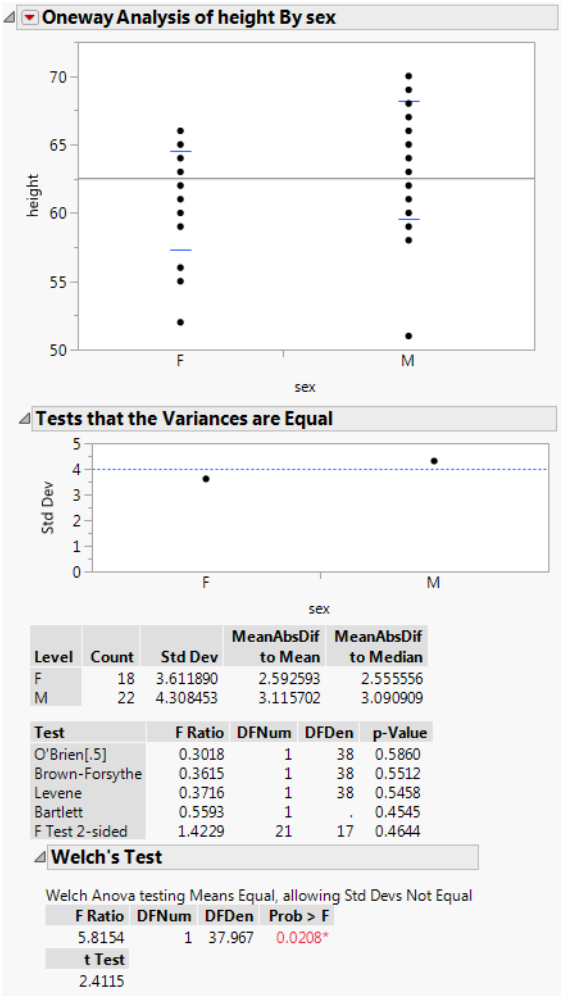
In this example, the nonparametric tests are more appropriate than the normality-based ANOVA test and the unequal variances *t* test. The nonparametric tests are resistant to the large value in row 32 and do not require the assumption of normality.

Example of the Unequal Variances Option

In this example, suppose you want to test whether two groups (males and females) have equal variance.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select height and click **Y, Response.**
4. Select sex and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of height By sex and select **Unequal Variances**.

Figure 6.29 Example of the Unequal Variances Report



Level	Count	Std Dev	MeanAbsDif to Mean	MeanAbsDif to Median
F	18	3.611890	2.592593	2.555556
M	22	4.308453	3.115702	3.090909

Test	F Ratio	DFNum	DFDen	p-Value
O'Brien[.5]	0.3018	1	38	0.5860
Brown-Forsythe	0.3615	1	38	0.5512
Levene	0.3716	1	38	0.5458
Bartlett	0.5593	1	.	0.4545
F Test 2-sided	1.4229	21	17	0.4644

Welch's Test

Welch Anova testing Means Equal, allowing Std Devs Not Equal

F Ratio	DFNum	DFDen	Prob > F
5.8154	1	37.967	0.0208*

t Test
2.4115

Since the *p*-value from the 2-sided *F* Test is large, you cannot conclude that the variances are unequal.

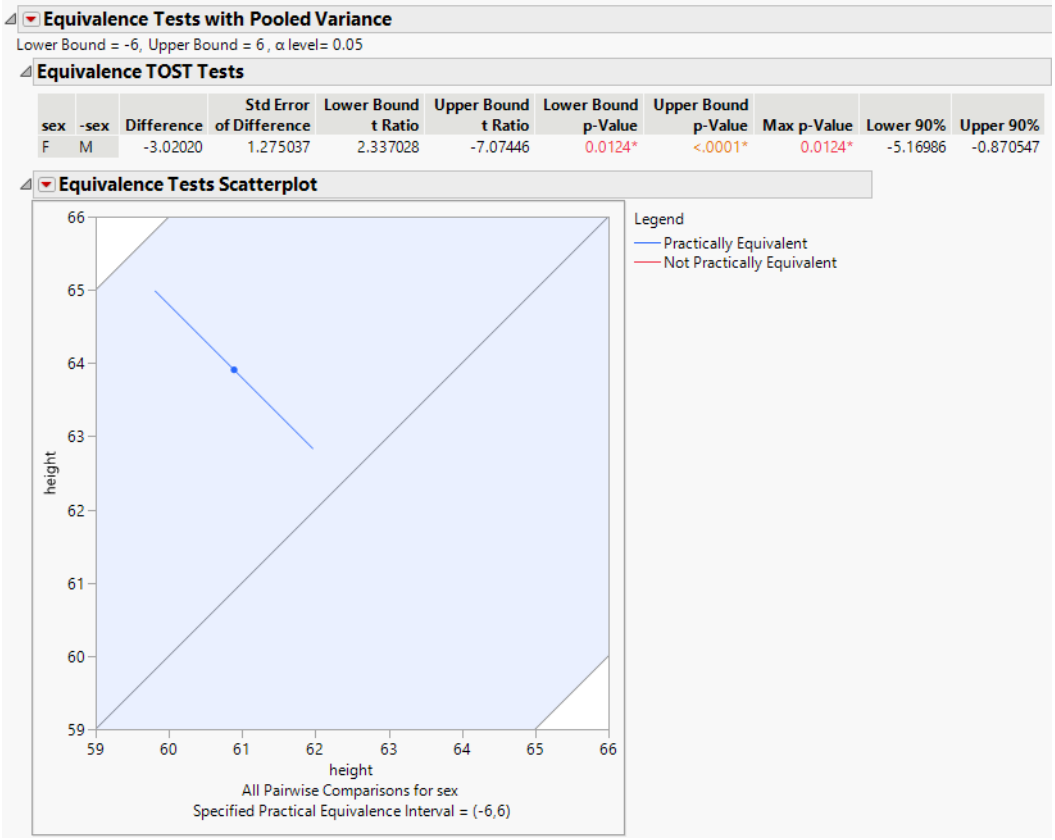
Example of an Equivalence Test

In this example, you investigate whether the difference in height between males and females is less than 6 inches.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select height and click **Y, Response.**

4. Select sex and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle Oneway Analysis of height By sex and select **Equivalence Tests**.
7. Type 6 as the difference considered practically zero.
8. Click **OK**.

Figure 6.30 Example of an Equivalence Test



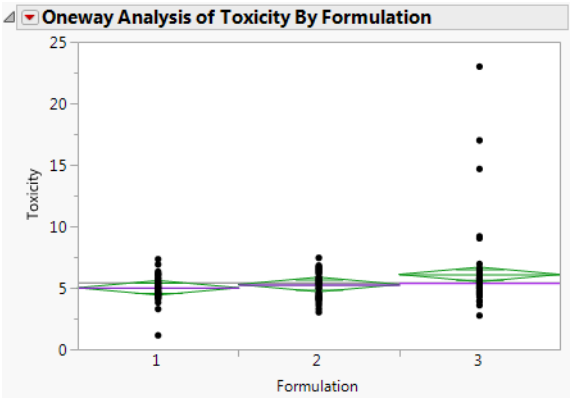
Using two one-sided tests, you can see that the *p*-value is small for both. Therefore, you can conclude that the difference in population means is located somewhere between -6 and 6. For your purposes, you can declare the means to be practically equivalent.

Example of the Robust Fit Option

Use a Robust fit in this example where one of three groups contains outliers. The data in the Drug Toxicity.jmp sample data table shows the toxicity levels for three different formulations of a drug.

1. Select **Help > Sample Data Library** and open Drug Toxicity.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Toxicity and click **Y, Response**.
4. Select Formulation and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of Toxicity By Formulation and select **Means/Anova**.
7. Click the red triangle next to Oneway Analysis of Toxicity By Formulation and select **Robust > Robust Fit**.

Figure 6.31 Example of Robust Fit



Oneway Anova

Summary of Fit

Rsquare	0.045786
Adj Rsquare	0.032803
Root Mean Square Error	2.128216
Mean of Response	5.460866
Observations (or Sum Wgts)	150

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio	Prob > F
Formulation	2	31.94736	15.9737	3.5267	0.0319*
Error	147	665.80744	4.5293		
C. Total	149	697.75480			

Means for Oneway Anova

Level	Number	Mean	Std Error	Lower 95%	Upper 95%
1	50	5.01002	0.30098	4.4152	5.6048
2	50	5.27760	0.30098	4.6828	5.8724
3	50	6.09498	0.30098	5.5002	6.6898

Std Error uses a pooled estimate of error variance

Robust Fit

Sigma	ChiSquare	PValue	LogWorth
1.01391	3.05069	0.2175	0.66245

Level	Robust Mean	Std Error
1	5.04222	0.12241
2	5.27785	0.15014
3	5.39175	0.17766

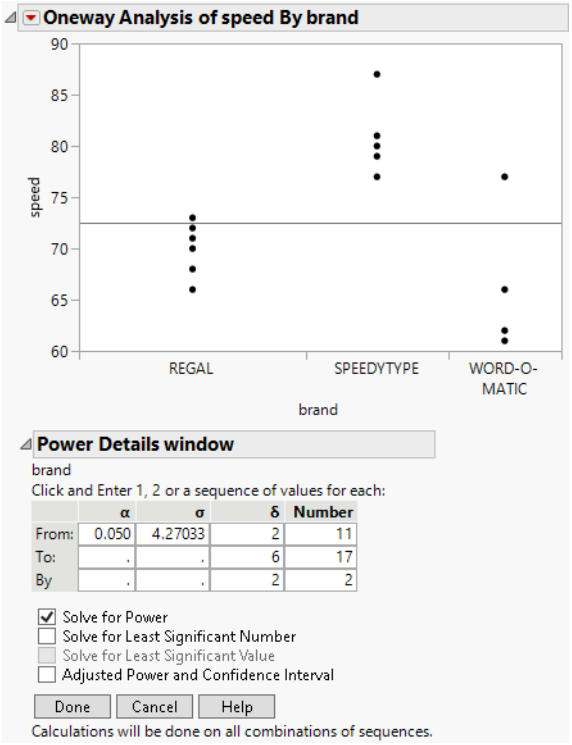
If you look at the standard Analysis of Variance report, you might wrongly conclude that there is a difference between the three formulations, since the *p*-value is 0.0319. However, when you look at the Robust Fit report, you would not conclude that the three formulations are significantly different, because the *p*-value there is 0.2175. It appears that the toxicity for a few of the observations is unusually high, creating the undue influence on the data.

Example of the Power Option

1. Select **Help > Sample Data Library** and open Typing Data.jmp.
2. Select **Analyze > Fit Y by X**.

3. Select speed and click **Y, Response.**
4. Select brand and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of speed By brand and select **Power**.
7. Within the From row, type 2 for Delta (the third box) and type 11 for Number.
8. Within the To row, type 6 for Delta, and type 17 in the Number box.
9. Within the By row, type 2 for both Delta and Number.
10. Select the **Solve for Power** check box.

Figure 6.32 Example of the Power Details Window



11. Click **Done**.

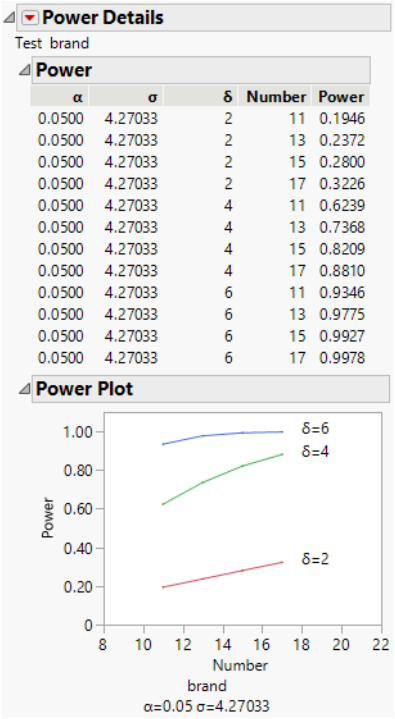
Note: The **Done** button remains dimmed until all of the necessary options are applied.

Power is computed for each combination of Delta and Number, and appears in the Power report.

To plot the Power values:

12. Click the red triangle below the table and select **Power Plot**.

Figure 6.33 Example of the Power Report



13. You might need to click and drag vertically on the Power axis to see all of the data in the plot.

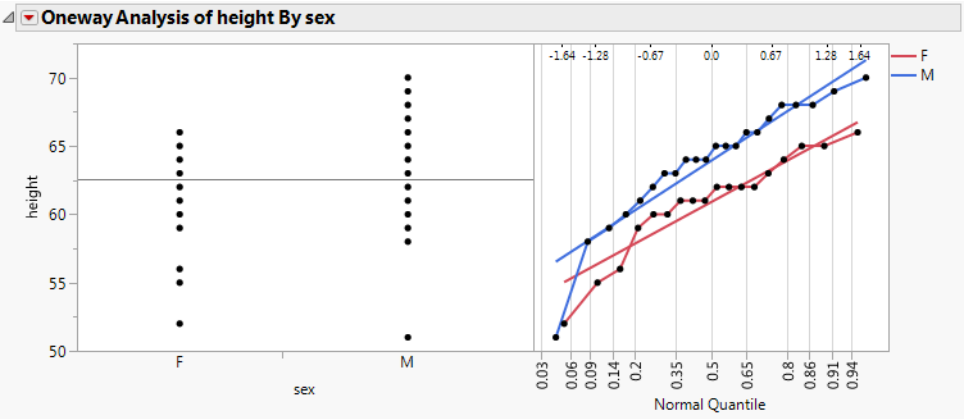
Power is plotted for each combination of Delta and Number. As you might expect, the power rises for larger Number (sample sizes) values and for larger Delta values (difference in means).

Example of a Normal Quantile Plot

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select height and click **Y, Response.**
4. Select sex and click **X, Factor**.
5. Click **OK**.

6. Click the red triangle next to Oneway Analysis of height By sex and select **Normal Quantile Plot > Plot Actual by Quantile**.

Figure 6.34 Example of a Normal Quantile Plot



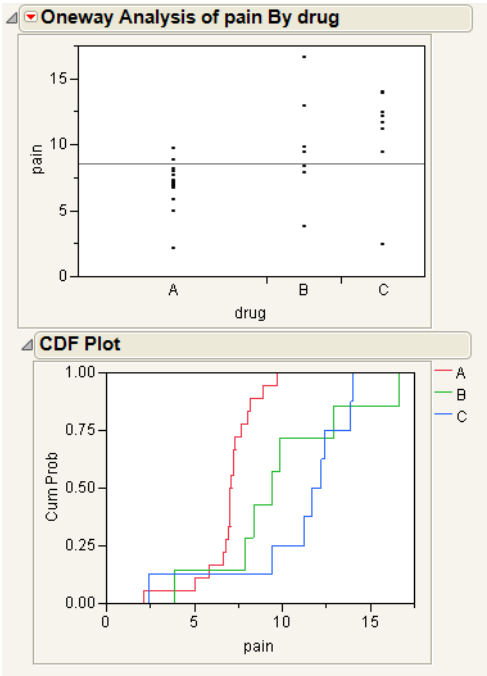
Note the following:

- The Line of Fit appears by default.
- The data points track very closely to the line of fit, indicating a normal distribution.

Example of a CDF Plot

1. Select **Help > Sample Data Library** and open Analgesics.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select pain and click **Y, Response**.
4. Select drug and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of pain By drug and select **CDF Plot**.

Figure 6.35 Example of a CDF Plot

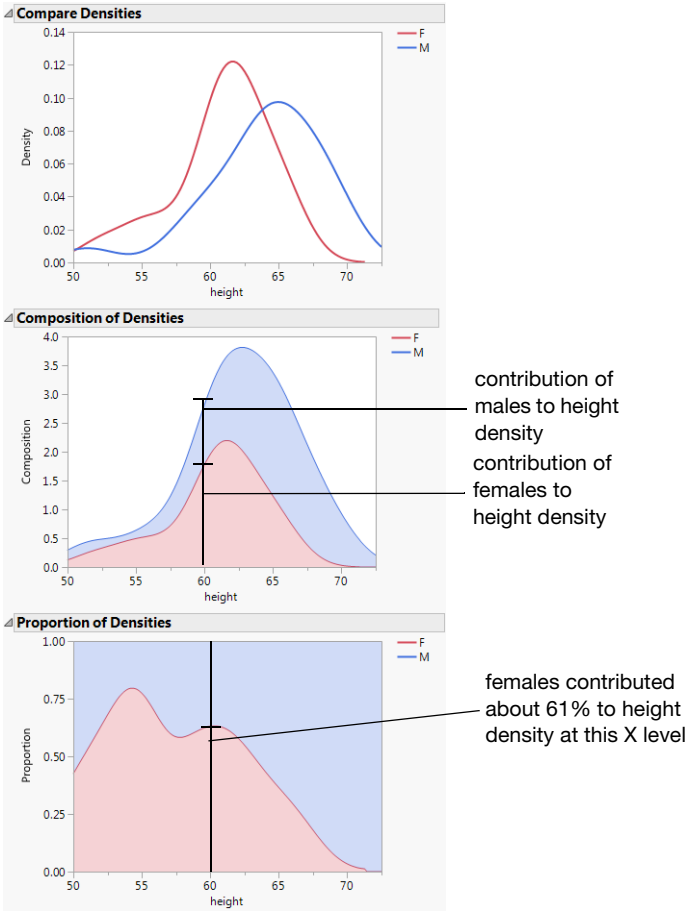


The levels of the X variables in the initial Oneway analysis appear in the CDF plot as different curves. The horizontal axis of the CDF plot uses the *y* value in the initial Oneway analysis.

Example of the Densities Options

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select height and click **Y, Response.**
4. Select sex and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of height By sex and select all three options: **Densities > Compare Densities**, **Densities > Composition of Densities**, and **Densities > Proportion of Densities**.

Figure 6.36 Example of the Densities Options



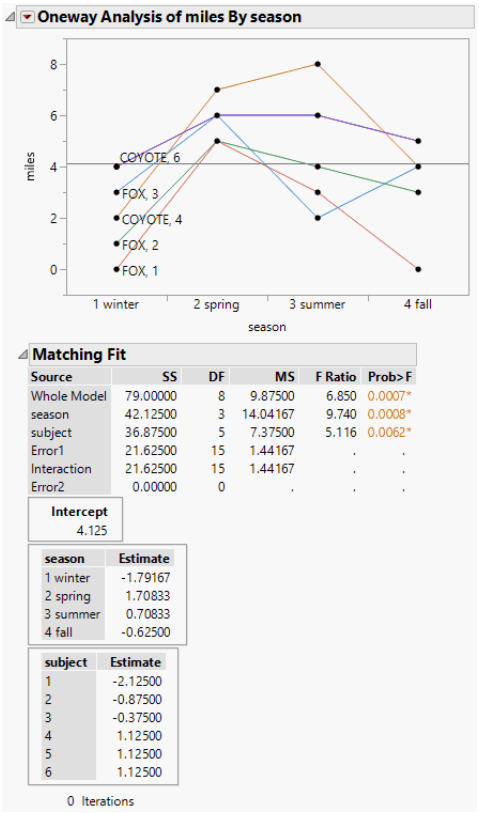
Example of the Matching Column Option

This example uses the Matching.jmp sample data table, which contains data on six animals and the miles that they travel during different seasons.

1. Select **Help > Sample Data Library** and open Matching.jmp.
2. Select **Analyze** > **Fit Y by X**.
3. Select miles and click **Y, Response**.
4. Select season and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of miles By season and select **Matching Column**.

7. Select subject as the matching column.
8. Click **OK**.

Figure 6.37 Example of the Matching Column Report



Matching Fit

Source	SS	DF	MS	F Ratio	Prob>F
Whole Model	79.00000	8	9.87500	6.850	0.0007*
season	42.12500	3	14.04167	9.740	0.0008*
subject	36.87500	5	7.37500	5.116	0.0062*
Error1	21.62500	15	1.44167	.	.
Interaction	21.62500	15	1.44167	.	.
Error2	0.00000	0	.	.	.

Intercept
4.125

season	Estimate
1 winter	-1.79167
2 spring	1.70833
3 summer	0.70833
4 fall	-0.62500

subject	Estimate
1	-2.12500
2	-0.87500
3	-0.37500
4	1.12500
5	1.12500
6	1.12500

0 Iterations

The plot graphs the miles traveled by season, with subject as the matching variable. The labels next to the first measurement for each subject on the graph are determined by the species and subject variables.

The Matching Fit report shows the season and subject effects with *F* tests. These are equivalent to the tests that you get with the Fit Model platform if you run two models, one with the interaction term and one without. If there are only two levels, then the *F* test is equivalent to the paired *t* test.

Note: For more information about the Fit Model platform, see *Fitting Linear Models*.

Example of Stacking Data for a Oneway Analysis

When your data are in a format other than a JMP data table, sometimes they are arranged so that a row contains information for multiple observations. To analyze the data in JMP, you must import the data and restructure it so that each row of the JMP data table contains information for a single observation. For example, suppose that your data are in a spreadsheet. The data for parts produced on three production lines are arranged in three sets of columns. In your JMP data table, you need to stack the data from the three production lines into a single set of columns so that each row represents the data for a single part.

Description and Goals

This example uses the file Fill Weights.xlsx, which contains the weights of cereal boxes randomly sampled from three different production lines. Figure 6.38 shows the format of the data.

- The ID columns contain an identifier for each cereal box that was measured.
- The Line columns contain the weights (in ounces) for boxes sampled from the corresponding production line.

Figure 6.38 Data Format

Weights					
ID	Line A	ID	Line B	ID	Line C
215	12.42	705	13.63	254	11.73
287	12.49	670	12.56	282	11.40
381	12.80	715	12.87	938	12.78
		683	13.09	597	12.19
		514	13.31	179	12.25
		517	12.64		
		946	12.75		

The target fill weight for the boxes is 12.5 ounces. Although you are interested in whether the three production lines are meeting the target, initially you want to see whether the three lines are achieving the same mean fill rate. You can use Oneway to test for differences among the mean fill weights.

To use the Oneway platform, you need to do the following:

1. Import the data into JMP. See "Import the Data".
2. Reshape the data so that each row in the JMP data table reflects only a single observation. Reshaping the data requires that you stack the cereal box IDs, the line identifiers, and the weights into columns. See "Stack the Data".

Import the Data

This example illustrates two ways to import data from Microsoft Excel into JMP. Select one method or explore both:

- Use the **File > Open** option to import data from a Microsoft Excel file using the Excel Import Wizard. See "Import the Data Using the Excel Import Wizard". This method is convenient for any Excel file.
- Copy and paste data from Microsoft Excel into a new JMP data table. See "Copy and Paste the Data from Excel". You can use this method with small data files.

For more information about how to import data from Microsoft Excel, see *Using JMP*.

Import the Data Using the Excel Import Wizard

1. Select **Help > Sample Data Library** and open Fill Weights.xlsx located in the Samples/Import Data folder.

   The file opens in the Excel Import Wizard.
2. Type 3 next to **Column headers start on row**.

   In the Excel file, row 1 contains information about the table and row 2 is blank. The column header information starts on row 3.
3. Type 2 for **Number of rows with column headers**.

   In the Excel file, rows 3 and 4 both contain column header information.
4. Click **Import**.

Figure 6.39 JMP Table Created Using Excel Import Wizard

	Weights-ID	Weights-Line A	Weights-ID 2	Weights-Line B	Weights-ID 3	Weights-Line C
1	215	12.42	705	13.63	254	11.73
2	287	12.49	670	12.56	282	11.40
3	381	12.80	715	12.87	938	12.78
4	•	•	683	13.09	597	12.19
5	•	•	514	13.31	179	12.25
6	•	•	517	12.64	•	•
7	•	•	946	12.75	•	•

The data are placed in seven rows and multiple IDs appear in each row. For each of the three lines, there are an ID and Weight column, giving a total of six columns.

Notice that the "Weights" part of the ID column name is unnecessary and misleading. You could rename the columns now, but it will be more efficient to rename the columns after you stack the data.

5. Proceed to "Stack the Data".

Copy and Paste the Data from Excel

1. Open Fill Weights.xlsx in Microsoft Excel.
2. Select the data inside the table but exclude the unnecessary "Weights" heading.
3. Right-click and select **Copy**.
4. In JMP, select **File > New > Data Table**.
5. Select **Edit > Paste with Column Names**.

   The **Edit > Paste with Column Names** option is used when your column names are included in the selection on the clipboard.

Figure 6.40 JMP Table Created Using Paste with Column Names

	ID	Line A	ID 2	Line B	ID 3	Line C
1	215	12.42	705	13.63	254	11.73
2	287	12.49	670	12.56	282	11.40
3	381	12.80	715	12.87	938	12.78
4	•	•	683	13.09	597	12.19
5	•	•	514	13.31	179	12.25
6	•	•	517	12.64	•	•
7	•	•	946	12.75	•	•

6. Proceed to "Stack the Data".

Stack the Data

Use the Stack option to place one observation in each row of a new data table. For more information about the Stack option, see *Using JMP*.

1. In the JMP data table, select **Tables > Stack**.
2. Select all six columns and click **Stack Columns**.
3. Select **Multiple Series Stack**.

   You are stacking two series, ID and Line, so you do not change the Number of Series, which is set to 2 by default. The columns that contain the series are not contiguous. They alternate (ID, Line A, ID, Line B, ID, Line C). For this reason, you do not check Contiguous.
4. Deselect **Stack By Row**.
5. Select **Eliminate Missing Rows**.
6. Enter Stacked next to **Output table name**.
7. Click **OK**.

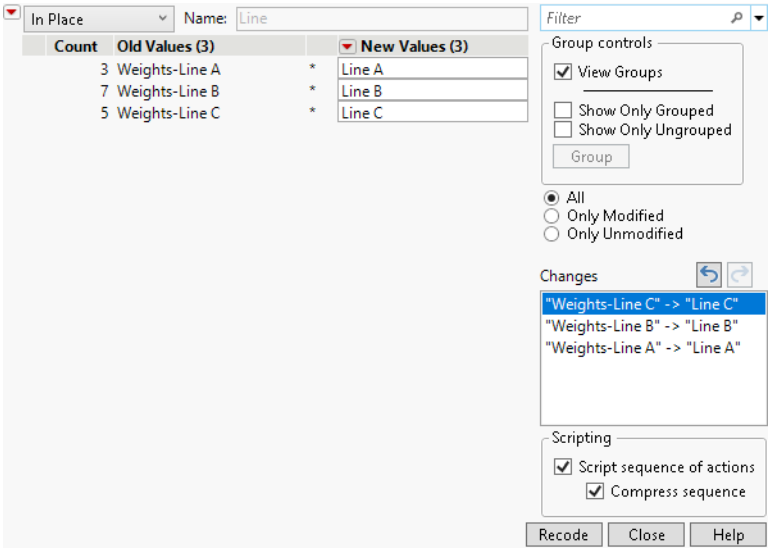
   In the new data table, Data and Data 2 are columns containing the ID and Weight data.
8. Right-click the Label column heading and select **Delete Columns**.

The entries in the Label column were the column headings for the box IDs in the imported data table. These entries are not needed.

9. Rename each column by double-clicking on the column header:
   - Data to ID
   - Label 2 to Line
   - Data 2 to Weight
10. In the Columns panel, click the icon to the left of ID and select **Nominal**.

    Although ID is given as a number, it is an identifier and should be treated as nominal when modeling. This is not an issue in this example, but it is good practice to assign the appropriate modeling type to a column.
11. (Applies only if you imported the data from Excel using **File > Open.)** Do the following:
    1. Click the Line column header to select the column and select **Cols > Recode**.
    2. Click **New Column** and select **In Place**.
    3. Change the values in the **New Values** column to match those in Figure 6.41 below.

Figure 6.41 Recode Column Values



4. Click **Recode**.

Your new data table is now properly structured for JMP analysis. Each row contains data for a single cereal box. The first column gives the box ID, the second gives the production line, and the third gives the weight of the box (Figure 6.42).

Figure 6.42 Recoded Data Table

	ID	Line	Weight
1	215	Line A	12.42
2	287	Line A	12.49
3	381	Line A	12.8
4	705	Line B	13.63
5	670	Line B	12.56
6	715	Line B	12.87
7	683	Line B	13.09
8	514	Line B	13.31
9	517	Line B	12.64
10	946	Line B	12.75
11	254	Line C	11.73
12	282	Line C	11.4
13	938	Line C	12.78
14	597	Line C	12.19
15	179	Line C	12.25

Conduct the Oneway Analysis

This part of the example contains the following tasks:

- Conduct a Oneway Analysis of Variance to test for differences in the mean fill weights among the three production lines.
- Obtain Comparison Circles to explore which lines might differ.
- Label points by ID in case you want to reweigh or further examine their boxes.

Before beginning, verify that you are using the Stacked data table.

1. Select **Analyze > Fit Y by X**.
2. Select Weight and click **Y, Response**.
3. Select Line and click **X, Factor**.
4. Click **OK**.
5. Click the red triangle next to Oneway Analysis of Weight By Line and select **Means/Anova**.

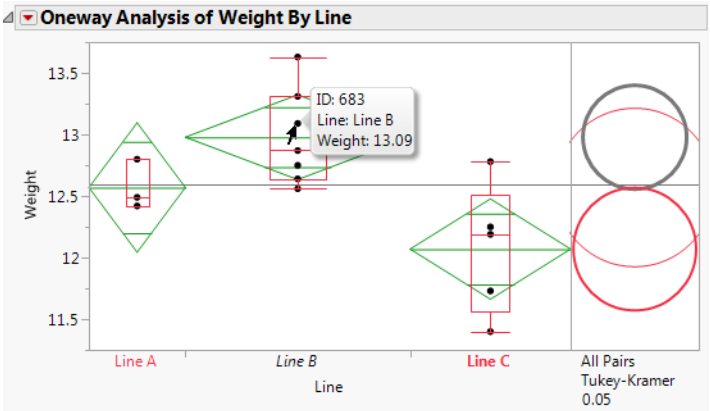
   The mean diamonds in the plot show 95% confidence intervals for the production line means. The points that fall outside the mean diamonds might seem like outliers. However, they are not. To see this, add box plots to the plot.
6. Click the red triangle next to Oneway Analysis of Weight By Line and select **Display Options > Box Plots**.

   All points fall within the box plots boundaries. Therefore, they are not outliers.
7. From the data table, in the Columns panel, right-click ID and select **Label/Unlabel**.

8. In the plot, hover over the points to see their ID values, as well as their Line and Weight data (Figure 6.43).
9. Click the red triangle next to Oneway Analysis of Weight By Line and select **Compare Means > All Pairs, Tukey HSD**.

   Comparison circles appear in a panel to the right of the plot.
10. Click the bottom comparison circle.

Figure 6.43 Oneway Analysis of Weight by Line



Oneway Anova

Summary of Fit

Rsquare	0.534629
Adj Rsquare	0.457068
Root Mean Square Error	0.418099
Mean of Response	12.594
Observations (or Sum Wgts)	15

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio	Prob > F
Line	2	2.4098743	1.20494	6.8930	0.0102*
Error	12	2.0976857	0.17481		
C. Total	14	4.5075600			

Means for Oneway Anova

Level	Number	Mean	Std Error	Lower 95%	Upper 95%
Line A	3	12.5700	0.24139	12.044	13.096
Line B	7	12.9786	0.15803	12.634	13.323
Line C	5	12.0700	0.18698	11.663	12.477

Std Error uses a pooled estimate of error variance

Welch's Test

Means Comparisons

In the Analysis of Variance report, the *p*-value of 0.0102 provides evidence that the means are not all equal. In the plot, the comparison circle for Line C is selected and appears red. Since the circle for Line B appears as thick gray, the mean for Line C differs from the mean for Line B at the 0.05 significance level. The means for Lines A and B do not show a statistically significant difference.

The mean diamonds shown in the plot span 95% confidence intervals for the means. The numeric bounds for the 95% confidence intervals are given in the Means for Oneway ANOVA report. Both of these indicate that the confidence intervals for Lines B and C do not contain the target fill weight of 12.5: Line B seems to overfill and Line C seems to underfill. For these two production lines, the underlying causes that result in off-target fill weights must be addressed.

Statistical Details for the Oneway Platform

- "Comparison Circles"
- "Power"
- "Summary of Fit Report"
- "Tests That the Variances Are Equal"
- "Nonparametric Test Statistics"

Comparison Circles

Comparison circles are a graphical representation of the *least significant difference* (LSD) in a multiple comparison test. This least significant difference is a Student's *t*-statistic multiplied by the standard error of the difference of the two means and is defined as follows:

$$\text{LSD} = t_{\alpha/2}\text{std}(\hat{\mu}_1 - \hat{\mu}_2)$$

The standard error of the difference of two independent means is calculated from the following relationship:

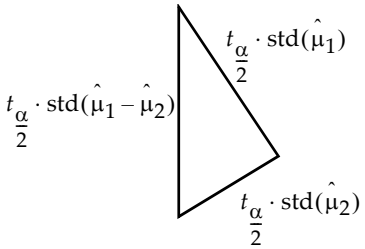
$$[\text{std}(\hat{\mu}_1 - \hat{\mu}_2)]^2 = [\text{std}(\hat{\mu}_1)]^2 + [\text{std}(\hat{\mu}_2)]^2$$

When the means are uncorrelated, these quantities have the following relationship:

$$\text{LSD}^2 = [t_{\alpha/2}\text{std}((\hat{\mu}_1 - \hat{\mu}_2))]^2 = [t_{\alpha/2}\text{std}(\hat{\mu}_1)]^2 + [t_{\alpha/2}\text{std}\hat{\mu}_2]^2$$

These squared values form a Pythagorean relationship, illustrated graphically by the right triangle shown in Figure 6.44.

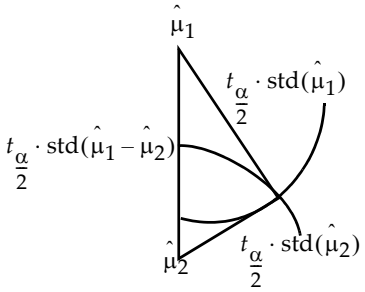
Figure 6.44 Relationship of the Difference between Two Means



The hypotenuse of this triangle is a measuring stick for comparing means. The means are significantly different if and only if the actual difference is greater than the hypotenuse (LSD).

Suppose that you have two means that are exactly on the borderline, where the actual difference is the same as the least significant difference. Draw the triangle with vertices at the means measured on a vertical scale. Also, draw circles around each mean so that the diameter of each is equal to the confidence interval for that mean.

Figure 6.45 Geometric Relationship of *t* Test Statistics



The radius of each circle is the length of the corresponding leg of the triangle, which is $t_{\alpha/2}\text{std}(\mu_i)$.

The circles must intersect at the same right angle as the triangle legs, giving the following relationship:

- If the means differ exactly by their least significant difference, then the confidence interval circles around each mean intersect at a right angle. That is, the angle of the tangents is a right angle.

Now, consider how these circles must intersect if the means are different by greater than or less than the least significant difference:

- If the circles intersect so that the outside angle is greater than a right angle, then the means *are not* significantly different. If the circles intersect so that the outside angle is less than a right angle, then the means *are* significantly different. An outside angle of less than 90 degrees indicates that the means are farther apart than the least significant difference.

- If the circles do not intersect, then they are significantly different. If they nest, they are not significantly different (Figure 6.11).

The same graphical technique works for many multiple-comparison tests, substituting a different probability quantile value for the Student's *t*.

Power

To compute power, you use the noncentral *F* distribution. The formula (O'Brien and Lohr 1984) is defined as follows:

Power = Prob($F > F_{crit}$, ν_1, ν_2, nc)

where:

F is distributed as the noncentral $F(nc, \nu_1, \nu_2)$ and $F_{crit} = F_{(1-\alpha, \nu1, \nu2)}$ is the 1 - α quantile of the F distribution with ν_1 and ν_2 degrees of freedom.

$\nu_1 = r - 1$ is the numerator df.

$\nu_2 = r(n - 1)$ is the denominator df.

n is the number per group.

r is the number of groups.

$nc = n(CSS)/\sigma^2$ is the non-centrality parameter.

$CSS = \sum_{g=1}^{r} (\mu_g - \mu)^2$ is the corrected sum of squares.

μ_g is the mean of the g^{th} group.

μ is the overall mean.

σ^2 is estimated by the mean squared error (MSE).

Summary of Fit Report

Rsquare

Using quantities from the Analysis of Variance report for the model, the R^2 for any continuous response fit is calculated as follows:

$$\frac{\text{Sum of Squares (Model)}}{\text{Sum of Squares (C Total)}}$$

Adj Rsquare

Adj Rsquare is a ratio of mean squares instead of sums of squares and is calculated as follows:

$$1 - \frac{\text{Mean Square (Error)}}{\text{Mean Square (C Total)}}$$

The mean square for Error is found in the Analysis of Variance report and the mean square for **C. Total** can be computed as the **C. Total** Sum of Squares divided by its respective degrees of freedom. See "The Analysis of Variance Report".

Tests That the Variances Are Equal

F Ratio

O'Brien's test constructs a dependent variable so that the group means of the new variable equal the group sample variances of the original response. The O'Brien variable is computed as follows:

$$r_{ijk} = \frac{(n_{ij} - 1.5)n_{ij}(y_{ijk} - \overline{y_{ij}})^2 - 0.5s_{ij}^2(n_{ij} - 1)}{(n_{ij} - 1)(n_{ij} - 2)}$$

where n represents the number of y_{ijk} observations.

Brown-Forsythe is the model F statistic from an ANOVA on $z_{ij} = |y_{ij} - \tilde{y}_i|$ where $\tilde{y}_i$ is the median response for the i^{th} level.

The Levene F is the model F statistic from an ANOVA on $z_{ij} = |y_{ij} - \overline{y_{i.}}|$ where $\overline{y_{i.}}$ is the mean response for the i^{th} level.

Bartlett's test is calculated as follows:

$$T = \frac{v\log\left(\sum_i \frac{v_i}{v} s_i^2\right) - \sum_i v_i \log(s_i^2)}{1 + \left(\frac{\sum_i \frac{1}{v_i} - \frac{1}{v}}{3(k-1)}\right)} \quad \text{where } v_i = n_i - 1 \text{ and } v = \sum_i v_i$$

and n_i is the count on the ith level and s_i^2 is the response sample variance on the i^{th} level. The Bartlett statistic has a χ^2-distribution. Dividing the Chi-square test statistic by the degrees of freedom results in the reported F value.

Welch's Test F Ratio

The Welch's Test F Ratio is computed as follows:

$$F = \frac{\left[\frac{\sum_i w_i(\bar{y}_i - \tilde{y}_{..})^2}{k-1}\right]}{\left\{1 + \frac{2(k-2)}{k^2-1}\left[\sum_i \frac{\left(1-\frac{w_i}{u}\right)^2}{n_i - 1}\right]\right\}} \quad \text{where} \quad w_i = \frac{n_i}{s_i^2}, \quad u = \sum_i w_i, \quad \tilde{y}_{..} = \sum_i \frac{w_i \bar{y}_{i.}}{u},$$

and n_i is the count on the ith level, $\bar{y}_{i.}$ is the mean response for the i^{th} level, and s_i^2 is the response sample variance for the i^{th} level.

Welch's Test DF Den

The Welch approximation for the denominator degrees of freedom is defined as follows:

$$df = \frac{1}{\left(\frac{3}{k^2-1}\right)\left[\sum_i \frac{\left(1-\frac{w_i}{u}\right)^2}{n_i - 1}\right]}$$

where w_i, n_i, and u are defined as in the F ratio formula.

Nonparametric Test Statistics

This section provides formulas for the test statistics used in the Wilcoxon, Median, van der Waerden, and Friedman Rank tests.

Notation

The tests are based on scores and use the following notation.

$j = 1, \ldots, n$ The observations in the entire sample.

$i = 1, \ldots, k$ The levels of X, where k is the total number of levels.

$n_1, n_2, \ldots, n_k$ The number of observations in each of the k levels of X.

R_j The midrank of the j^{th} observation. The *midrank* is the observation's rank if it is not tied and its average rank if it is tied.

α A function of the midranks used to define scores for the various tests.

The following notation is used when a Block variable is specified in the launch window.

$b = 1,\ldots, B$ The levels of the blocking variable, where B is the total number of blocks.

R_{bi} The midrank of the i^{th} level of X within block b.

The function α defines scores as follows:

Wilcoxon Scores

$$\alpha(R_j) = R_j$$

Median Scores

$$\alpha(R_j) = \begin{cases} 1 \text{ if } R_j > \text{median} \\ 0 \text{ if } R_j < \text{median} \\ t \text{ if } R_j = \text{median} \end{cases}$$

Let n_t denote the number of observations tied at the median and let n_u denote the number of observations greater than the median. Then t is given by the following:

$$t = \frac{\text{floor}(n/2) - n_u}{n_t}$$

van der Waerden Scores

$$\alpha(R_j) = \text{Standard Normal Quantile}(R_j/(n+1))$$

Friedman Rank Scores

$$\alpha(R_{bi}) = R_{bi}$$

Two-Sample Normal Approximations

Tests based on the normal approximation are given only when X has exactly two levels. The notation used in this section is defined in "Notation". The statistics that appear in the Two-Sample Normal Approximation report are defined below.

S The statistic *S* is the sum of the values $\alpha(R_j)$ for the observations in the smaller group. If the two levels of X have the same numbers of observations, then the value of S corresponds to the last level of X in the value ordering.

Z The value of Z is defined as follows:

$$Z = (S - E(S)) / \sqrt{Var(S)}$$

Note: The Wilcoxon test adds a continuity correction. If (*S - E(S)*) is greater than zero, then 0.5 is subtracted from the numerator. If (*S - E(S)*) is less than zero, then 0.5 is added to the numerator.

E(*S*) The expected value of S under the null hypothesis. Denote the number of observations in the smaller level, or in the last level in the value ordering if the two groups have the same number of observations, by n_l:

$$E(S) = \frac{n_l}{n} \sum_{j=1}^{n} \alpha(R_j)$$

Var(*S*) Define *ave* to be the average score across all observations. Then the variance of S is defined as follows:

$$Var(S) = \frac{n_1 n_2}{n(n-1)} \sum_{j=1}^{n} (\alpha(R_j) - ave)^2$$

Two-Sample Normal Approximations for Friedman Rank Test

When you use the Friedman Rank test, the calculations for the two-sample normal approximation is the same as above, except that the variance of S is different. The variance of S is computed as follows:

$$Var(S) = \frac{B}{(n-1)} \sum_{j=1}^{n} (\alpha(R_j) - ave)^2$$

One-Way ChiSquare Approximations

Note: The ChiSquare test based on the Wilcoxon scores is known as the Kruskal-Wallis test.

The notation used in this section is defined in "Notation". The following quantities are used in calculating the ChiSquare statistic:

T_i The total of the scores for the i^{th} level of X.

$E(T_i)$ The expected value of the total score for level i under the null hypothesis of no difference in levels, defined as follows:

$$E(T_i) = \frac{n_i}{n} \sum_{j=1}^{n} \alpha(R_j)$$

*Var***(T)** Define *ave* to be the average score across all observations. Then the variance of T is defined as follows:

$$Var(T) = \frac{1}{(n-1)} \sum_{j=1}^{n} (\alpha(R_j) - ave)^2$$

The value of the test statistic is given below. This statistic is asymptotically ChiSquare on *k - 1* degrees of freedom.

$$C = \left(\sum_{i=1}^{k} (T_i - E(T_i))^2 / n_i \right) / Var(T)$$

One-Way ChiSquare Approximations for Friedman Rank Test

The ChiSquare test statistic for the Friedman Rank Test is calculated as follows:

$$C = \frac{\sum_{i=1}^{k} (T_i - E(T_i))^2 / n_i}{\frac{1}{(k-1)} \sum_{j=1}^{n} (\alpha(R_j) - ave)^2 / n_i}$$

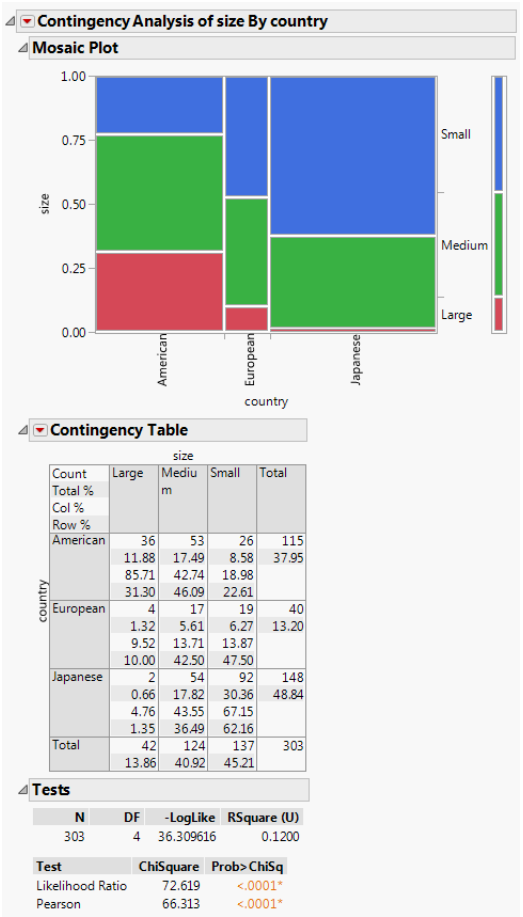
Chapter 7

Contingency Analysis

Examine Relationships between Two Categorical Variables

Use the Contingency platform to investigate the relationship between two categorical variables. The categorical variables can be nominal or ordinal. The analysis results include a mosaic plot, a contingency table of frequency counts and proportions, and Chi-squared tests of significance. You can interactively perform additional analyses and tests on your data, such as Analysis of Means for Proportions, correspondence analysis and association analyses. The Contingency platform is the *categorical by categorical* personality of the Fit Y by X platform.

Figure 7.1 Example of Contingency Analysis



Contents

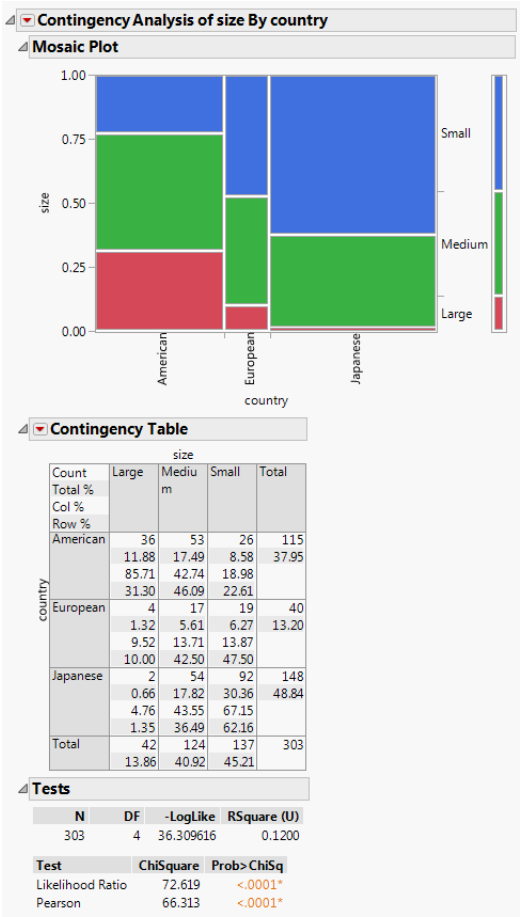
Example of Contingency Analysis . . . 247
Launch the Contingency Platform . . . 248
Data Format . . . 249
The Contingency Report . . . 250
Contingency Platform Options . . . 251
Mosaic Plot . . . 253
Pop-Up Menu . . . 254
Contingency Table . . . 255
Description of the Contingency Table . . . 256
Tests . . . 257
Description of the Tests Report . . . 257
Fisher's Exact Test . . . 258
Analysis of Means for Proportions . . . 258
Correspondence Analysis . . . 259
Understanding Correspondence Analysis Plots . . . 259
Correspondence Analysis Options . . . 259
The Details Report . . . 259
Cochran-Mantel-Haenszel Test . . . 260
Agreement Statistic . . . 261
Relative Risk . . . 261
Two Sample Test for Proportions . . . 262
Measures of Association . . . 262
Cochran Armitage Trend Test . . . 264
Exact Test . . . 264
Additional Examples of the Contingency Platform . . . 265
Example of Analysis of Means for Proportions . . . 265
Example of Correspondence Analysis . . . 266
Example of a Cochran Mantel Haenszel Test . . . 268
Example of the Agreement Statistic Option . . . 269
Example of the Relative Risk Option . . . 270
Example of a Two Sample Test for Proportions . . . 272
Example of the Measures of Association Option . . . 273
Example of the Cochran Armitage Trend Test . . . 274
Statistical Details for the Contingency Platform . . . 275
Agreement Statistic Option . . . 275
Odds Ratio Option . . . 276
Tests Report . . . 276
Details Report in Correspondence Analysis . . . 277

Example of Contingency Analysis

Learn how to examine the relationship between two categorical variables in JMP. This example uses data collected from car polls. The data include respondent attributes: sex, marital status, and age. The data also include attributes of the respondent's car: country of origin, the size, and the type. Examine the relationship between car sizes (small, medium, and large) and the cars' country of origin.

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select size and click **Y, Response**.
4. Select country and click **X, Factor**.
5. Click **OK**.

Figure 7.2 Example of Contingency Analysis



From the mosaic plot and legend, notice the following:

- Very few Japanese cars fall into the Large size category.
- The majority of the European cars fall into the Small and Medium size categories.
- The majority of the American cars fall into the Large and Medium size categories.

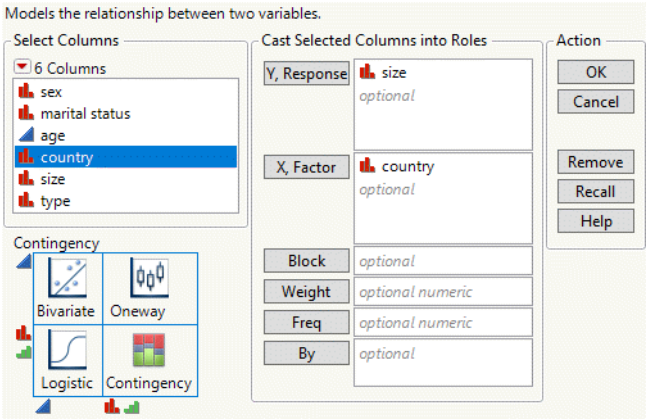
Launch the Contingency Platform

To perform a contingency analysis, do the following:

1. Select **Analyze > Fit Y by X**.
2. Enter a nominal or ordinal column for **Y, Response**.

3. Enter a nominal or ordinal column for **X, Factor**.

Figure 7.3 The Fit Y by X Launch Window



The word Contingency appears above the diagram, to indicate that you are performing a contingency analysis.

Note: You can also launch a contingency analysis from the JMP Starter window. Select **View** > **JMP Starter** > **Basic** > **Contingency**.

For more information about this launch window, see "Introduction to Fit Y by X". For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Data Format

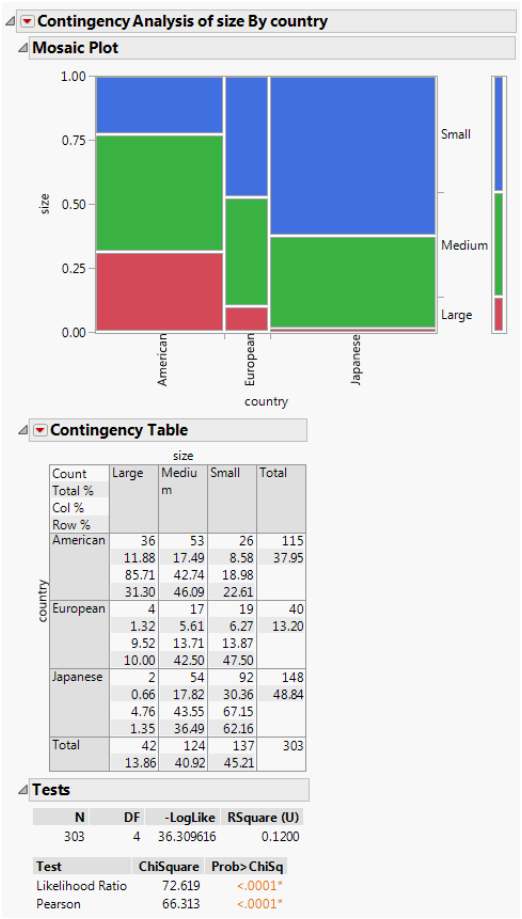
Categorical data are often presented in summary form, where there is only one row in the data table for each combination of levels of the Y and X variables. In this situation, use the Freq or Weight variable to indicate the number of observations that each row represents. For an example of summarized categorical data, see "Example of Analysis of Means for Proportions".

The Contingency Report

The Contingency report initially shows a Mosaic Plot, a Contingency Table, and a Tests report. You can run additional analyses and tests using the red triangle menu options. For more information about all of these reports and options, see "Contingency Platform Options".

To produce the plot shown in Figure 7.4, follow the instructions in "Example of Contingency Analysis".

Figure 7.4 Example of a Contingency Report



Note: Any rows that are excluded in the data table are also hidden in the Mosaic Plot.

Contingency Platform Options

Use the options within the Contingency Analysis red triangle menu to perform additional analyses.

Note: The Fit Group menu appears if you have specified multiple Y variables. Menu options enable you to arrange reports or order them by RSquare. See *Fitting Linear Models*.

Mosaic Plot A graphical representation of the data in the Contingency Table. See "Mosaic Plot".

Contingency Table A two-way frequency table. There is a row for each factor level and a column for each response level. See "Contingency Table".

Tests Analogous to the Analysis of Variance table for continuous data. The tests show that the response level rates are the same across X levels. See "Tests".

Set α level Changes the alpha level used in confidence intervals. Select one of the common values (0.10, 0.05, 0.01) or select a specific value using the **Other** option.

Analysis of Means for Proportions (Appears only if the response has exactly two levels.) Compares response proportions for the X levels to the overall response proportion. See "Analysis of Means for Proportions".

Correspondence Analysis Shows which rows or columns of a frequency table have similar patterns of counts. In the correspondence analysis plot, there is a point for each row and for each column of the contingency table. See "Correspondence Analysis".

Cochran Mantel Haenszel Tests if there is a relationship between two categorical variables after blocking across a third classification. See "Cochran-Mantel-Haenszel Test".

Agreement Statistic (Appears only when the X and Y variables have the same levels.) Displays the Kappa statistic (Agresti 1990), its standard error, confidence interval, hypothesis test, and Bowker's test of symmetry, also know as McNemar's test. See "Agreement Statistic".

Relative Risk (Appears only when the X and Y variables have two levels.) Calculates risk ratios. See "Relative Risk".

Odds Ratio (Appears only when the X and Y variables have two levels.) Produces a report of the odds ratio. See "Odds Ratio Option".

The report also gives a confidence interval for this ratio. You can change the alpha level using the **Set α Level** option.

Two Sample Test for Proportions (Appears only when the X and Y variables have two levels.) Performs a two-sample test for proportions. This test compares the proportions of the Y variable between the two levels of the X variable. See "Two Sample Test for Proportions".

Measures of Association Describes the association between the variables in the contingency table. See "Measures of Association".

Cochran Armitage Trend Test (Appears only when one variable has two levels and the other variable is ordinal.) Tests for trends in binomial proportions across levels of a single variable. See "Cochran Armitage Trend Test".

JMP PRO **Exact Test** Provides exact versions of the following tests:

- Fisher's Test
- Cochran Armitage Trend Test
- Agreement Statistic

See "Exact Test".

Display Options > Horizontal Mosaic Rotates the mosaic plot horizontally or vertically.

Make Into Data Table Creates a JMP data table from the report table.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

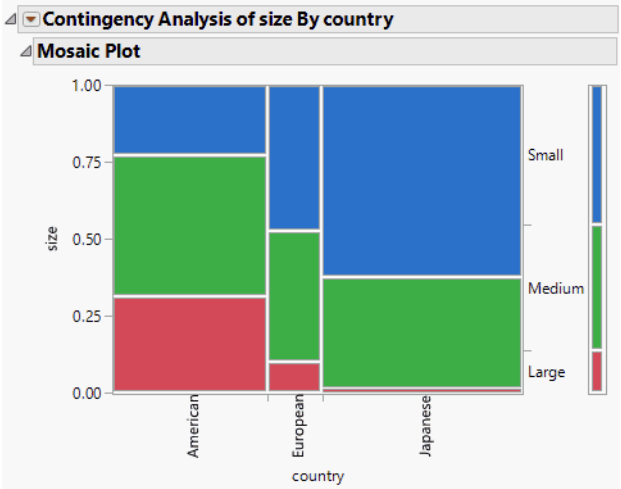
Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Mosaic Plot

The mosaic plot is a graphical representation of the two-way frequency table or Contingency Table. A mosaic plot is divided into rectangles; the vertical length of each rectangle is proportional to the proportions of the Y variable in each level of the X variable. The mosaic plot was introduced by Hartigan and Kleiner (1981) and refined by Friendly (1994).

To produce the plot shown in Figure 7.5, follow the instructions in "Example of Contingency Analysis".

Figure 7.5 Example of a Mosaic Plot



Note the following about the mosaic plot:

- The proportions on the horizontal axis represent the number of observations for each level of the X variable, which is country.
- The proportions on the vertical axis at right represent the overall proportions of Small, Medium, and Large cars for the combined levels (American, European, and Japanese).
- The scale of the vertical axis at left shows the response probability. The whole axis is equivalent to a probability of one (representing the total sample).

Clicking on a rectangle in the mosaic plot highlights the selection and highlights the corresponding data in the associated data table.

Replace variables in the mosaic plot by dragging and dropping a variable, in one of two ways: swap existing variables by dragging and dropping a variable from one axis to the other axis; or, click a variable in the Columns panel of the associated data table and drag it onto an axis.

Pop-Up Menu

Right-click the mosaic plot to change colors and label the cells.

Set Colors Shows the current assignment of colors to levels. See "Set Colors".

Cell Labeling Specify a label to be drawn in the mosaic plot. Select one of the following options:

No Labels Shows no labels, and removes any of the other options.

Label by Count Shows the number of observations in each cell.

Label by Percent Shows the percent of observations in each cell.

Label by Value Shows the levels of the Y variable corresponding to each cell.

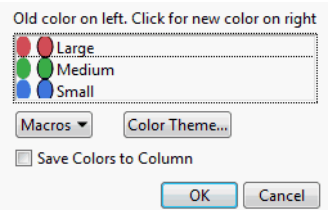
Label by Row Shows the row labels for all of the rows represented by the cell.

Note: For descriptions of the remainder of the right-click options, see *Using JMP*.

Set Colors

When you select the **Set Colors** option, the Select Colors for Values window appears.

Figure 7.6 Select Colors for Values Window



The default mosaic colors depend on whether the response column is ordinal or nominal, and whether there is an existing Value Colors column property. To change the color for any level, click the oval in the second column of colors and select a new color.

Description of the Select Colors for Values Window

Macros Computes a color gradient between any two levels:

Gradient Between Ends Applies a gradient to all levels of the variable.

Gradient Between Selected Points Applies a color gradient to the levels that you have selected. You can select a range of levels by dragging the cursor over the levels that you want to select, or by pressing the Shift key and clicking the first and last level.

Reverse Colors Reverses the color order.

Revert to Old Colors Reverts any of your changes.

Color Theme Changes the colors for each value based on a color theme.

Save Colors to Column If you change the default color theme, then select this check box, a new column property (**Value Colors**) is added to the column in the associated data table. To edit this property from the data table, select **Cols > Column Info**.

Contingency Table

The Contingency Table is a two-way frequency table. There is a row for each factor level and a column for each response level.

To produce the plot shown in Figure 7.7, follow the instructions in "Example of Contingency Analysis".

Figure 7.7 Example of a Contingency Table

Contingency Table

country	Count Total % Col % Row %	size Large	Mediu m	Small	Total
American		36	53	26	115
		11.88	17.49	8.58	37.95
		85.71	42.74	18.98	
		31.30	46.09	22.61	
European		4	17	19	40
		1.32	5.61	6.27	13.20
		9.52	13.71	13.87	
		10.00	42.50	47.50	
Japanese		2	54	92	148
		0.66	17.82	30.36	48.84
		4.76	43.55	67.15	
		1.35	36.49	62.16	
Total		42	124	137	303
		13.86	40.92	45.21	

Note the following about Contingency tables:

- The Count, Total%, Col%, and Row% correspond to the data within each cell that has row and column headings (such as the cell under American and Large).
- The last column contains the total counts for each row and percentages for each row.
- The bottom row contains total counts for each column and percentages for each column.

In Figure 7.7, focus on the cars that are large and come from America. The following table explains the conclusions that you can make about these cars using the Contingency Table.

Table 7.1 Conclusions Based on Example of a Contingency Table

Number	Description	Label in Table
36	Number of cars that are both large and come from America	Count
11.88%	Percentage of all cars that are both large and come from America (36/303)[a].	Total%
85.71%	Percentage of large cars that come from America (36/42)[b]	Col%
31.30%	Percentage of American cars that are large (36/115)[c].	Row%
37.95%	Percentage of all cars that come from America (115/303).	(none)
13.86%	Percentage of all cars that are large (42/303).	(none)

a. 303 is the total number of cars in the poll.
b. 42 is the total number of large cars in the poll.
c. 115 is the total number of American cars in the poll.

Tip: To show or hide data in the Contingency Table, from the red triangle menu next to Contingency Table, select the option that you want to show or hide.

Description of the Contingency Table

Count Cell frequency, margin total frequencies, and grand total (total sample size).

Total% Percent of cell counts and margin totals to the grand total.

Row% Percent of each cell count to its row total.

Col% Percent of each cell count to its column total.

Expected Expected frequency (E) of each cell under the assumption of independence. Computed as the product of the corresponding row total and column total divided by the grand total.

Deviation Observed cell frequency (O) minus the expected cell frequency (E).

Cell Chi Square Chi-square values computed for each cell as $(O - E)^2 / E$.

Col Cum Cumulative column total.

Col Cum% Cumulative column percentage.

Row Cum Cumulative row total.

Row Cum% Cumulative row percentage.

Tests

The Tests report shows the results for two tests to determine whether the response level rates are the same across X levels.

To produce the report shown in Figure 7.8, follow the instructions in "Example of Contingency Analysis".

Figure 7.8 Example of a Tests Report

Tests

N	DF	-LogLike	RSquare (U)
303	4	36.309616	0.1200

Test	ChiSquare	Prob>ChiSq
Likelihood Ratio	72.619	<.0001*
Pearson	66.313	<.0001*

Note the following about the Chi-square statistics:

- When both categorical variables are responses (Y variables), the Chi-square statistics test that they are independent.
- You might have a Y variable with a fixed X variable. In this case, the Chi-square statistics test that the distribution of the Y variable is the same across each X level.

Description of the Tests Report

N Total number of observations.

DF Records the degrees of freedom associated with the test. The degrees of freedom are equal to $(c - 1)(r - 1)$, where c is the number of columns and r is the number of rows.

-LogLike Negative log-likelihood, which measures fit and uncertainty (much like sums of squares in continuous response situations). See *Fitting Linear Models*.

Rsquare (U) Portion of the total uncertainty attributed to the model fit.

- An R^2 of 1 means that the factors completely predict the categorical response.
- An R^2 of 0 means that there is no gain from using the model instead of fixed background response rates.

See "Tests Report".

Test Lists two Chi-square statistical tests of the hypothesis that the response rates are the same in each sample category. See "Tests Report".

Prob>ChiSq Lists the probability of obtaining a Chi-square value greater than the one computed if no relationship exists between the response and factor. If both variables have only two levels, Fisher's exact probabilities for the one-tailed tests and the two-tailed test also appear.

Fisher's Exact Test

This report gives the results of Fisher's exact test for a 2x2 table. The results appear automatically for 2x2 tables. For more information about Fisher's exact test and the test for *r* x *c* tables, see "Exact Test".

Analysis of Means for Proportions

The analysis of means for proportions (ANOMP) is a multiple comparison method for testing if individual group proportions differ from the overall proportion.

Note: For a description of Analysis of Means methods, see the document by Nelson et al. (2005). See also "Example of Analysis of Means for Proportions".

If the response has two levels, you can use this option to compare response proportions for the X levels to the overall response proportion. This method uses the normal approximation to the binomial. Therefore, if the sample sizes are too small, a warning appears in the results.

The following options appear in the red triangle menu:

Set Alpha Level Selects the alpha level used in the analysis.

Show Summary Report Produces a report that shows the response proportions with decision limits for each level of the X variable. The report indicates whether a limit has been exceeded.

Switch Response Level for Proportion Changes the response category used in the analysis.

Display Options Shows or hides the decision limits, decision limit shading, center line, and point options.

Correspondence Analysis

The Correspondence Analysis option in the Contingency platform provides a graphical technique to show which rows or columns of a frequency table have similar patterns of counts. In the correspondence analysis plot, there is a point for each row and for each column. Use Correspondence Analysis when you have many levels, making it difficult to derive useful information from the mosaic plot.

Note: See also "Example of Correspondence Analysis".

Understanding Correspondence Analysis Plots

The *row profile* can be defined as the set of row-wise rates, or in other words, the counts in a row divided by the total count for that row. If two rows have very similar row profiles, their points in the correspondence analysis plot are close together. Squared distances between row points are approximately proportional to Chi-square distances that test the homogeneity between the pair of rows.

Column and row profiles are alike because the problem is defined symmetrically. The distance between a row point and a column point has no meaning. However, the directions of columns and rows from the origin are meaningful, and the relationships help interpret the plot.

Correspondence Analysis Options

Use the options in the Correspondence Analysis red triangle menu to add a 3-D scatterplot to the Contingency report and to add column properties to the data table.

3D Correspondence Analysis Produces a 3-D scatterplot.

Save Value Order Takes the order of the levels sorted by the first correspondence score coefficient and makes a column property for both the X and Y columns.

The Details Report

The Details report contains statistical information about the correspondence analysis and shows the values used in the plot.

Singular Value Provides the singular value decomposition of the contingency table. For the formula, see "Details Report in Correspondence Analysis".

Inertia Lists the square of the singular values, reflecting the relative variation accounted for in the canonical dimensions.

Portion Portion of inertia with respect to the total inertia.

Cumulative Shows the cumulative portion of inertia. If the first two singular values capture the bulk of the inertia, then the 2-D correspondence analysis plot is sufficient to show the relationships in the table.

X variable c1, c2, c3 The values plotted on the Correspondence Analysis plot (Figure 7.11).

Y variable c1, c2, c3 The values plotted on the Correspondence Analysis plot (Figure 7.11).

Cochran-Mantel-Haenszel Test

The Cochran-Mantel-Haenszel test evaluates the relationship between two categorical variables after blocking across a third classification.

Note: See also "Example of a Cochran Mantel Haenszel Test".

Correlation of Scores Applicable when both Y or X are ordinal or interval. The alternative hypothesis is that there is a linear association between Y and X in at least one level of the blocking variable.

Row Score by Col Categories Applicable when Y is ordinal or interval. The alternative hypothesis is that, for at least one level of the blocking variable, the mean scores of the *r* rows are unequal.

Col Score by Row Categories Applicable when X is ordinal or interval. The alternative hypothesis is that, for at least one level of the blocking variable, the mean scores of the *c* columns are unequal.

General Assoc. of Categories Tests that for at least one level of the blocking variable, there is some type of association between X and Y.

Agreement Statistic

When the two variables have the same levels, the **Agreement Statistic** option is available. This option shows the Kappa statistic (Agresti 1990), its standard error, confidence interval, hypothesis test, and Bowker's test of symmetry.

The Kappa statistic and associated *p*-value given in this section are approximate. An exact version of the agreement statistic is available. See "Exact Test".

Note: For statistical details, see "Agreement Statistic Option". See also "Example of the Agreement Statistic Option".

Kappa Shows the Kappa statistic.

Std Err Shows the standard error of the Kappa statistic.

Lower 95% Shows the lower endpoint of the confidence interval for Kappa.

Upper 95% Shows the upper endpoint of the confidence interval for Kappa.

Prob>Z Shows the *p*-value for a one-sided test for Kappa. The null hypothesis tests if Kappa equals zero.

Prob>|Z| Shows the *p*-value for a two-sided test for Kappa.

ChiSquare Shows the test statistic for Bowker's test. For Bowker's test of symmetry, the null hypothesis is that the probabilities in the square table satisfy symmetry, or that $p_{ij}=p_{ji}$ for all pairs of table cells. When both X and Y have two levels, this test is equal to McNemar's test.

Prob>ChiSq Shows the *p*-value for the Bowker's test.

Relative Risk

Note: See also "Example of the Relative Risk Option".

Calculate risk ratios for 2x2 contingency tables using the **Relative Risk** option. Confidence intervals also appear in the report. You can find more information about this method in Agresti (1990, sect. 3.4.2).

The Choose Relative Risk Categories window appears when you select the **Relative Risk** option. You can select a single response and factor combination, or you can calculate the risk ratios for all combinations of response and factor levels.

Two Sample Test for Proportions

When both the X and Y variables have two levels, you can construct a confidence interval for the difference between the two proportions. It also computes the test corresponding to the confidence interval.

Note: See also "Example of a Two Sample Test for Proportions".

Description Shows the test being performed.

Proportion Difference Shows the difference in the proportions between the levels of the X variable.

Lower 95% Shows the lower endpoint of the confidence interval for the difference. Based on the adjusted Wald confidence interval.

Upper 95% Shows the upper endpoint of the confidence interval for the difference. Based on the adjusted Wald confidence interval.

Adjusted Wald Test Shows two-tailed and one-tailed tests.

Prob Shows the *p*-values for the tests.

Response <variable> category of interest Select which response level to use in the test.

Measures of Association

The **Measures of Association** option provides association statistics.

Note: See also "Example of the Measures of Association Option".

Gamma Based on the number of concordant and discordant pairs and ignores tied pairs. Takes values in the range -1 to 1.

Kendall's Tau-b Similar to Gamma and uses a correction for ties. Takes values in the range -1 to 1.

Stuart's Tau-c Similar to Gamma and uses an adjustment for table size and a correction for ties. Takes values in the range -1 to 1.

Somers' D An asymmetric modification of Tau-b.

– The C|R denotes that the row variable X is regarded as an independent variable and the column variable Y is regarded as dependent.

– Similarly, the R|C denotes that the column variable Y is regarded as an independent variable and the row variable X is dependent.

Somers' D differs from Tau-b in that it uses a correction for ties only when the pair is tied on the independent variable. It takes values in the range -1 to 1.

Lambda Asymmetric Differs for C|R and R|C.

– For C|R, is interpreted as the probable improvement in predicting the column variable Y given knowledge of the row variable X.

– For R|C, is interpreted as the probable improvement in predicting the row variable X given knowledge about the column variable Y.

Takes values in the range 0 to 1.

Lambda Symmetric Loosely interpreted as the average of the two Lambda Asymmetric measures. Takes values in the range 0 to 1.

Uncertainty Coef

– For C|R, is the proportion of uncertainty in the column variable Y that is explained by the row variable X.

– For R|C, is interpreted as the proportion of uncertainty in the row variable X that is explained by the column variable Y.

Takes values in the range 0 to 1.

Uncertainty Coef Symmetric Symmetric version of the two Uncertainty Coef measures. Takes values in the range 0 to 1.

Notes:

- Each statistic appears with its standard error and confidence interval.
- Gamma, Kendall's Tau-b, Stuart's Tau-c, and Somers' D are measures of ordinal association that consider whether the variable Y tends to increase as X increases. They classify pairs of observations as concordant or discordant. A pair is concordant if an observation with a larger value of X also has a larger value of Y. A pair is discordant if an observation with a larger value of X has a smaller value of Y. These measures are appropriate only when both variables are ordinal.
- The Lambda and Uncertainty measures are appropriate for ordinal and nominal variables.

For computational details about the measures of association statistics, see the FREQ Procedure chapter in SAS Institute Inc. (2020b). The following references also contain additional information:

- Brown and Benedetti (1977)
- Goodman and Kruskal (1979)
- Kendall and Stuart (1979)

- Snedecor and Cochran (1980)
- Somers (1962)

Cochran Armitage Trend Test

Note: See also "Example of the Cochran Armitage Trend Test".

This Cochran Armitage Trend tests for trends in binomial proportions across the levels of a single variable. This test is appropriate only when one variable has two levels and the other variable is ordinal. The two-level variable represents the response, and the other represents an explanatory variable with ordered levels. The null hypothesis is the hypothesis of no trend, which means that the binomial proportion is the same for all levels of the explanatory variable.

The test statistic and *p*-values given in this test are approximate. An exact version of the trend test is available. See "Exact Test".

Exact Test

JMP PRO The following Exact tests are available in the Contingency platform:

Fisher's Exact Test Performs Fisher's Exact test for an *r* x *c* table. This is a test for association between two variables. Fisher's exact test assumes that the row and column totals are fixed, and uses the hypergeometric distribution to compute probabilities.

This test does not depend on any large-sample distribution assumptions. This means it is appropriate for situations where the Likelihood Ratio and Pearson tests become less reliable, like for small sample sizes or sparse tables.

The report includes the following information:

Table Probability (P) Gives the probability for the observed table. This is not the *p*-value for the test.

Two-sided Prob ≤ P Gives the *p*-value for the two-sided test.

For 2x2 tables, the Fisher's Exact test is automatically performed, unless one row or column contains all zeros (in this case, the test cannot be calculated). See "Tests".

Exact Cochran Armitage Trend Test Performs the exact version of the Cochran Armitage Trend Test. This test is available only when one of the variables has two levels. For more information about the trend test, see "Cochran Armitage Trend Test".

Exact Agreement Statistic Performs an exact test for testing agreement between variables. This is an exact test for the Kappa statistic. This is available only when the two variables have the same levels. For more information about agreement testing, see "Agreement Statistic".

Note: If the overall sample size is larger than 32767 and the contingency table is larger than 2x2, the exact test options are not available.

Additional Examples of the Contingency Platform

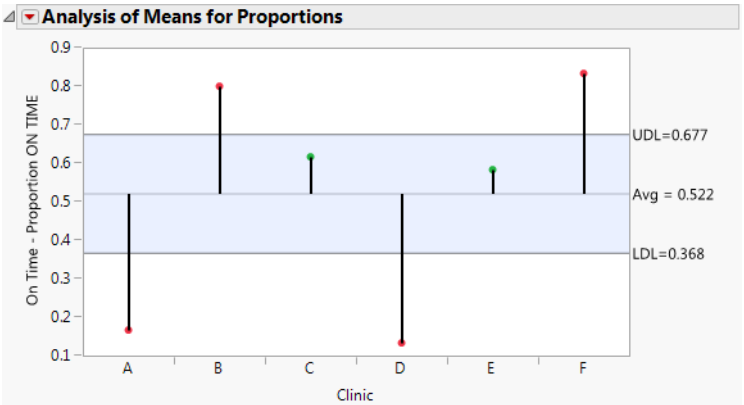
- "Example of Analysis of Means for Proportions"
- "Example of Correspondence Analysis"
- "Example of a Cochran Mantel Haenszel Test"
- "Example of the Agreement Statistic Option"
- "Example of the Relative Risk Option"
- "Example of a Two Sample Test for Proportions"
- "Example of the Measures of Association Option"
- "Example of the Cochran Armitage Trend Test"

Example of Analysis of Means for Proportions

This example shows you how to examine the proportion of patients that arrived on-time to their appointments. The Office Visits.jmp sample data table records late and on-time appointments for six clinics in a geographic region. 60 random appointments were selected from 1 week of records for each of the six clinics. To be considered on-time, the patient must be taken to an exam room within five minutes of their scheduled appointment time.

1. Select **Help > Sample Data Library** and open Office Visits.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select On Time and click **Y, Response**.
4. Select Clinic and click **X, Factor**.
5. Select Frequency and click **Freq**.
6. Click **OK**.
7. Click the red triangle next to Contingency Analysis of On Time By Clinic and select **Analysis of Means for Proportions**.
8. Click the red triangle next to Analysis of Means for Proportions and select **Show Summary Report** and **Switch Response Level for Proportion**.

Figure 7.9 Example of Analysis of Means for Proportions



Analysis of Means for Proportions Summary

Level	Group N	Lower Limit	Group Proportion	Upper Limit	Limit Exceeded
A	60	0.36791	0.166667	0.676534	Lower
B	60	0.36791	0.8	0.676534	Upper
C	60	0.36791	0.616667	0.676534	
D	60	0.36791	0.133333	0.676534	Lower
E	60	0.36791	0.583333	0.676534	
F	60	0.36791	0.833333	0.676534	Upper

Figure 7.9 shows the proportion of patients who were on-time from each clinic. From Figure 7.9, notice the following:

- The proportion of on-time arrivals is the highest for clinic F, followed by clinic B.
- Clinic D has the lowest proportion of on-time arrivals, followed by clinic A.
- Clinic E and clinic C are close to the average, and do not exceed the decision limits.

Example of Correspondence Analysis

This example uses the Cheese.jmp sample data table. This table contains data from the Newell cheese tasting experiment; the data were reported in McCullagh and Nelder (1989). The experiment records the counts of nine different response levels across four different cheese additives.

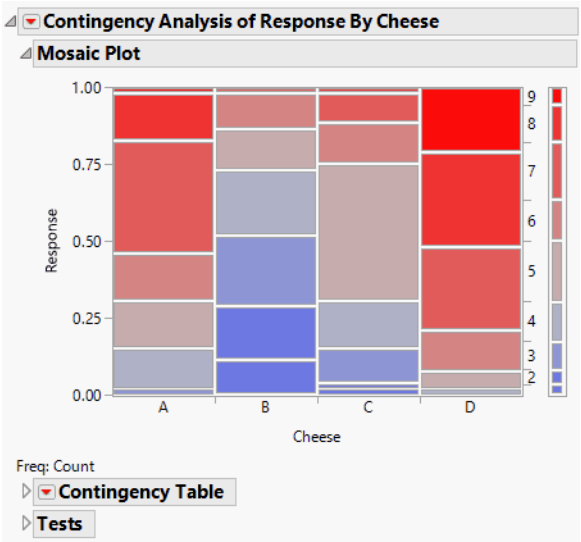
1. Select **Help > Sample Data Library** and open Cheese.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Response and click **Y, Response**.

   The Response values range from one to nine, where one is the least liked, and nine is the best liked.
4. Select Cheese and click **X, Factor**.

   A, B, C, and D represent four different cheese additives.

5. Select Count and click **Freq**.
6. Click **OK**.

Figure 7.10 Mosaic Plot for the Cheese Data



From the mosaic plot, you notice that the distributions do not appear alike. However, it is challenging to make sense of the mosaic plot across nine levels. A correspondence analysis can help define relationships in this type of situation.

7. To see the correspondence analysis plot, click the red triangle next to Contingency Analysis of Response By Cheese and select **Correspondence Analysis**.

Figure 7.11 Example of a Correspondence Analysis Plot

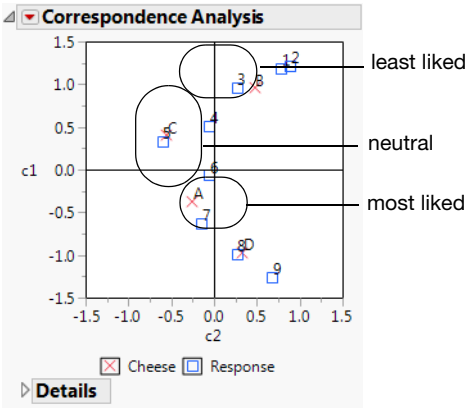


Figure 7.11 shows the correspondence analysis graphically, where the plot axes are labeled c1 and c2. Notice the following:

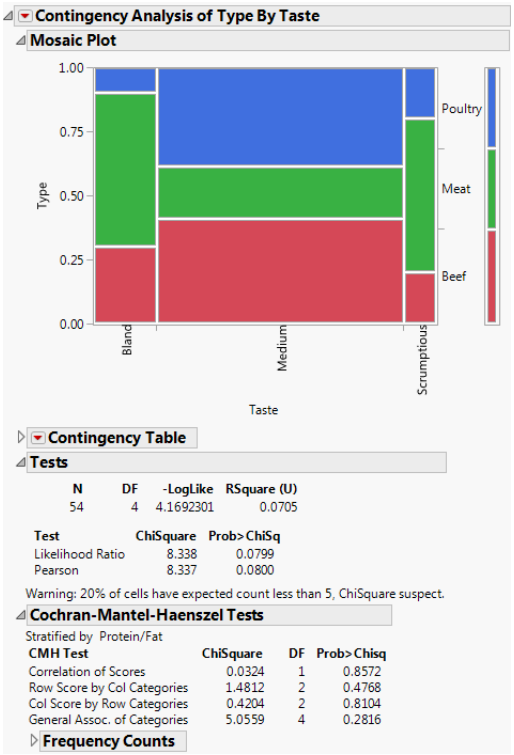
- c1 seems to correspond to a general satisfaction level. The cheeses on the c1 axis go from least liked at the top to most liked at the bottom.
- Cheese D is the most liked cheese, with responses of 8 and 9.
- Cheese B is the least liked cheese, with responses of 1,2, and 3.
- Cheeses C and A are in the middle, with responses of 4,5,6, and 7.

Example of a Cochran Mantel Haenszel Test

This example examines the relationship between hot dog type and taste.

1. Select **Help > Sample Data Library** and open Hot Dogs.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Type and click **Y, Response**.
4. Select Taste and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Contingency Analysis of Type By Taste and select **Cochran Mantel Haenszel**.
7. Select Protein/Fat as the grouping variable and click **OK**.

Figure 7.12 Example of a Cochran-Mantel-Haenszel Test



Notice the following:

- The Tests report shows a marginally significant Chi-square probability of about 0.0799, indicating some significance in the relationship between hot dog taste and type.
- The Cochran Mantel Haenszel report shows that the *p*-value for the general association of categories is 0.2816, which is much larger than 5%.

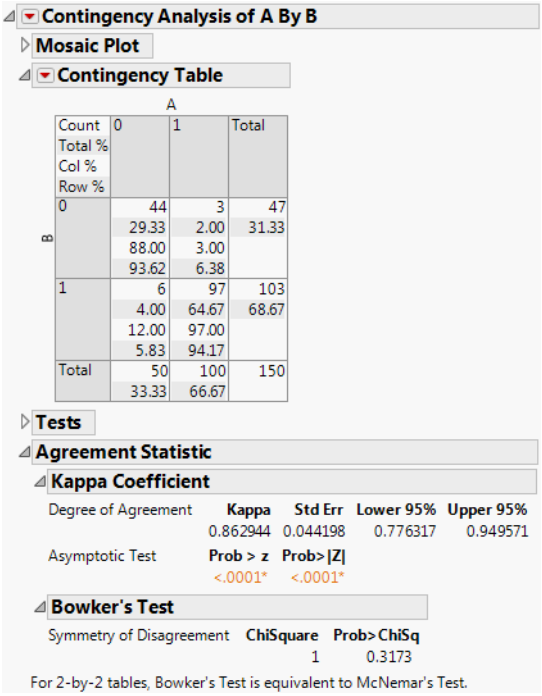
Example of the Agreement Statistic Option

This example examines the relationship between two raters. The data gives results from three people (raters) rating fifty parts three times each. Examine the relationship between raters A and B.

1. Select **Help > Sample Data Library** and open Attribute Gauge.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select A and click **Y, Response**.
4. Select B and click **X, Factor**.
5. Click **OK**.

6. Click the red triangle next to Contingency Analysis of A By B and select **Agreement Statistic**.

Figure 7.13 Example of the Agreement Statistic Report



You notice that the agreement statistic of 0.86 is high (close to 1) and the *p*-value of <.0001 is small. This reinforces the high agreement seen by looking at the diagonal of the contingency table. Agreement between the raters occurs when both raters give a rating of 0 or both give a rating of 1.

Example of the Relative Risk Option

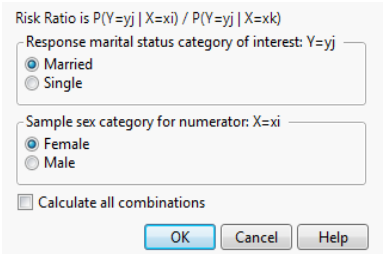
This example examines the relative probabilities of being married and single for the participants in a poll.

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select marital status and click **Y, Response**.
4. Select sex and click **X, Factor**.
5. Click **OK**.

6. Click the red triangle next to Contingency Analysis of marital status By sex and select **Relative Risk**.

   The Choose Relative Risk Categories window appears.

Figure 7.14 The Choose Relative Risk Categories Window



Note the following about the Choose Relative Risk Categories window:

- If you are interested in only a single response and factor combination, you can select that here. For example, if you clicked **OK** in the window in Figure 7.14, this is the calculation:

$$\frac{P(Y = \text{Married} | X = \text{Female})}{P(Y = \text{Married} | X = \text{Male})}$$

- If you would like to calculate the risk ratios for all ($2 \times 2 = 4$) combinations of response and factor levels, select the **Calculate All Combinations** check box (Figure 7.15).

7. Ask for all combinations by selecting the **Calculate All Combinations** check box. Leave all other default selections as is.

Figure 7.15 Example of the Risk Ratio Report

Contingency Analysis of marital status By sex

Mosaic Plot

Contingency Table

	marital status		
Count Total % Col % Row %	Married	Single	Total
sex: Female	95 31.35 48.47 68.84	43 14.19 40.19 31.16	138 45.54
Male	101 33.33 51.53 61.21	64 21.12 59.81 38.79	165 54.46
Total	196 64.69	107 35.31	303

Tests

Relative Risk

Description	Relative Risk	Lower 95%	Upper 95%
P(Married|Female)/P(Married|Male)	1.124623	0.953193	1.326885
P(Married|Male)/P(Married|Female)	0.889187	0.753645	1.049105
P(Single|Female)/P(Single|Male)	0.803329	0.587179	1.099046
P(Single|Male)/P(Single|Female)	1.24482	0.90988	1.703057

The relative risk is calculated as follows:

1. Examine the first entry in the Relative Risk report, which is P(Married | Female)/P(Married | Male).
2. You can find these probabilities in the Contingency Table. Since the probabilities are computed based on two levels of sex, which differs across the rows of the table, use the Row% to read the probabilities:

   P(Married | Female) = 0.6884

   P(Married | Male) = 0.6121

   Therefore, the relative risk is calculated as follows:

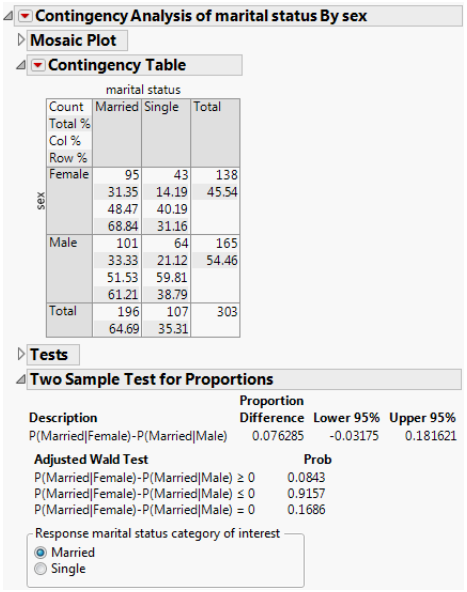
   $$\text{P(Married | Female)/P(Married | Male)} = \frac{0.6884}{0.6121} = 1.1247$$

Example of a Two Sample Test for Proportions

This example examines the probability of being married for males and females.

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select marital status and click **Y, Response**.
4. Select sex and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Contingency Analysis of marital status By sex and select **Two Sample Test for Proportions**.

Figure 7.16 Example of the Two Sample Test for Proportions Report



Contingency Analysis of marital status By sex

Mosaic Plot

Contingency Table

	marital status		
Count Total % Col % Row %	Married	Single	Total
sex: Female	95 31.35 48.47 68.84	43 14.19 40.19 31.16	138 45.54
Male	101 33.33 51.53 61.21	64 21.12 59.81 38.79	165 54.46
Total	196 64.69	107 35.31	303

Tests

Two Sample Test for Proportions

Description	Proportion Difference	Lower 95%	Upper 95%
P(Married|Female)-P(Married|Male)	0.076285	-0.03175	0.181621

Adjusted Wald Test	Prob
P(Married|Female)-P(Married|Male) ≥ 0	0.0843
P(Married|Female)-P(Married|Male) ≤ 0	0.9157
P(Married|Female)-P(Married|Male) = 0	0.1686

Response marital status category of interest
- (•) Married
- ( ) Single

In this example, you are comparing the probability of being married between females and males. See the Row% in the Contingency Table to obtain the following:

P(Married | Female) = 0.6884

P(Married | Male) = 0.6121

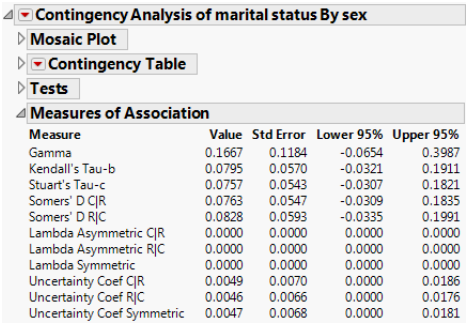
The difference between these two numbers, 0.0763, is the Proportion Difference shown in the report. The two-sided confidence interval is [-0.03175, 0.181621]. The *p*-value by the adjusted Wald method corresponding to the confidence interval is 0.1686, which is close to the *p*-value (0.1665) by Pearson's Chi-square test. Generally, Pearson's Chi-square test is more popular than the modified Wald's test for testing the difference of two proportions.

Example of the Measures of Association Option

This example examines the association of being married with gender.

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select marital status and click **Y, Response**.
4. Select sex and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Contingency Analysis of marital status By sex and select **Measures of Association**.

Figure 7.17 Example of the Measures of Association Report



Contingency Analysis of marital status By sex

Mosaic Plot

Contingency Table

Tests

Measures of Association

Measure	Value	Std Error	Lower 95%	Upper 95%
Gamma	0.1667	0.1184	-0.0654	0.3987
Kendall's Tau-b	0.0795	0.0570	-0.0321	0.1911
Stuart's Tau-c	0.0757	0.0543	-0.0307	0.1821
Somers' D C|R	0.0763	0.0547	-0.0309	0.1835
Somers' D R|C	0.0828	0.0593	-0.0335	0.1991
Lambda Asymmetric C|R	0.0000	0.0000	0.0000	0.0000
Lambda Asymmetric R|C	0.0000	0.0000	0.0000	0.0000
Lambda Symmetric	0.0000	0.0000	0.0000	0.0000
Uncertainty Coef C|R	0.0049	0.0070	0.0000	0.0186
Uncertainty Coef R|C	0.0046	0.0066	0.0000	0.0176
Uncertainty Coef Symmetric	0.0047	0.0068	0.0000	0.0181

Since the variables that you want to examine (sex and marital status) are nominal, use the Lambda and Uncertainty measures. All of them are small, so it seems that there is a weak association between sex and marital status.

Example of the Cochran Armitage Trend Test

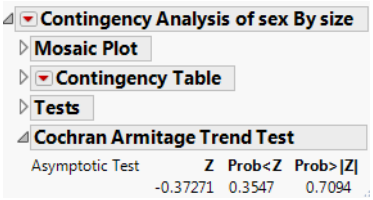
This example investigates whether there is a relationship between the proportion of males and females that buy different sizes of cars.

1. Select **Help > Sample Data Library** and open Car Poll.jmp.

For the purposes of this test, change size to an ordinal variable:

2. In the Columns panel, right-click the icon next to size and select **Ordinal**.
3. Select **Analyze > Fit Y by X**.
4. Select sex and click **Y, Response**.
5. Select size and click **X, Factor**.
6. Click **OK**.
7. Click the red triangle next to Contingency Analysis of sex By size and select **Cochran Armitage Trend Test**.

Figure 7.18 Example of the Cochran Armitage Trend Test Report



Contingency Analysis of sex By size

Mosaic Plot

Contingency Table

Tests

Cochran Armitage Trend Test

Asymptotic Test	Z	Prob<Z	Prob>|Z|
	-0.37271	0.3547	0.7094

The two-sided *p*-value (0.7094) is large. From this, you cannot conclude that there is a relationship in the proportion of male and females that purchase different sizes of cars.

Statistical Details for the Contingency Platform

- "Agreement Statistic Option"
- "Odds Ratio Option"
- "Tests Report"
- "Details Report in Correspondence Analysis"

Agreement Statistic Option

Viewing the two response variables as two independent ratings of the n subjects, the Kappa coefficient equals +1 when there is complete agreement of the raters. When the observed agreement exceeds chance agreement, the Kappa coefficient is positive and its magnitude reflects the strength of agreement. Although unusual in practice, Kappa is negative when the observed agreement is less than chance agreement. The minimum value of Kappa is between -1 and 0, depending on the marginal proportions.

The Kappa coefficient is computed as follows:

$$\hat{\kappa} = \frac{P_0 - P_c}{1 - P_c} \quad \text{where} \quad P_0 = \sum_i p_{ii} \quad \text{and} \quad P_c = \sum_i p_{i+} p_{+i}$$

Note that p_{ij} is the proportion of subjects in the (i, j) th cell, such that $\sum_i \sum_j p_{ij} = 1$.

The asymptotic variance of the simple kappa coefficient is estimated by the following:

$$\text{var} = \frac{A + B - C}{(1 - P_c)^2 n} \quad \text{where} \quad A = \sum_i p_{ii}[1 - (p_{i+} + p_{+i})(1 - \hat{\kappa})]^2 ,$$

$$B = (1 - \hat{\kappa})^2 \sum_{i \neq j} \sum p_{ij} (p_{+i} + p_{j+})^2 \quad \text{and}$$

$$C = [\hat{\kappa} - P_c(1 - \hat{\kappa})]^2$$

See Cohen (1960) and Fleiss et al. (1969).

For Bowker's test of symmetry, the null hypothesis is that the probabilities in the two-by-two table satisfy symmetry (p_{ij}=p_{ji}).

Odds Ratio Option

The Odds Ratio is calculated as follows:

$$\frac{p_{11} \times p_{22}}{p_{12} \times p_{21}}$$

where p_{ij} is the count in the i^{th} row and j^{th} column of the 2x2 table.

Tests Report

Rsquare (U)

Rsquare (U) is computed as follows:

$$\frac{-\text{log likelihood for Model}}{-\text{log likelihood for Corrected Total}}$$

The total negative log-likelihood is found by fitting fixed response rates across the total sample.

Test

The two Chi-square tests are computed as follows:

The Likelihood Ratio Chi-square test statistic is computed as twice the negative log-likelihood for Model in the Tests table. Some books use the notation G^2 for this statistic. The difference of two negative log-likelihoods, one with *whole-population* response probabilities and one with *each-population* response rates, is defined as follows:

$$G^2 = 2\left[\sum_{ij}(-n_{ij})\ln(p_j) - \sum_{ij} -n_{ij}\ln(p_{ij})\right] \quad \text{where} \quad p_{ij} = \frac{n_{ij}}{N} \quad \text{and} \quad p_j = \frac{N_j}{N}$$

This formula can be more compactly written as follows:

$$G^2 = 2\sum_i \sum_j n_{ij} \ln\left(\frac{n_{ij}}{e_{ij}}\right)$$

The Pearson Chi-square test statistic is calculated by summing the squares of the differences between the observed and expected cell counts. The Pearson Chi-square test exploits the property that frequency counts tend to a normal distribution in very large samples. The familiar form of this Chi-square statistic is defined as follows:

$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

where O is the observed cell counts and E is the expected cell counts. The summation is over all cells. There is no continuity correction done here, as is sometimes done in 2-by-2 tables.

Details Report in Correspondence Analysis

Lists values from the following singular value decomposition (SVD):

$$\mathbf{D}_r^{-0.5}(\mathbf{P} - rc')\mathbf{D}_c^{-0.5} = \mathbf{U}Diag(\Lambda)\mathbf{V}'$$

where:

- **P** is the matrix of counts divided by the total frequency
- *r* and *c* are row and column sums of **P**
- the **D** matrices are diagonal matrices of the values of *r* and *c*
- Λ is the column vector of the singular values reported in the details report

For more information about singular value decomposition, see *Multivariate Methods*.

The row coordinates (*rc*) and column coordinates (*cc*) in the details report are computed as follows:

$$rc = \mathbf{D}_r^{-0.5}\mathbf{U}Diag(\Lambda)$$

$$cc = \mathbf{D}_c^{-0.5}\mathbf{V}Diag(\Lambda)$$

Chapter 8

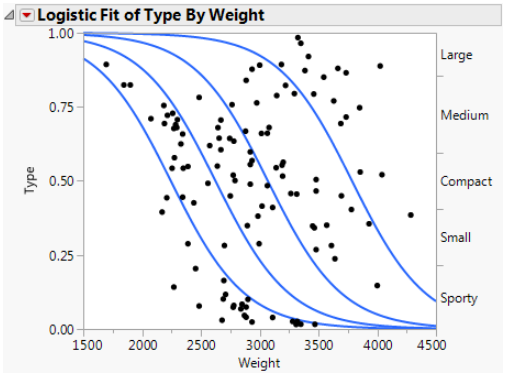
Logistic Analysis

Examine Relationships between a Categorical Y and a Continuous X Variable

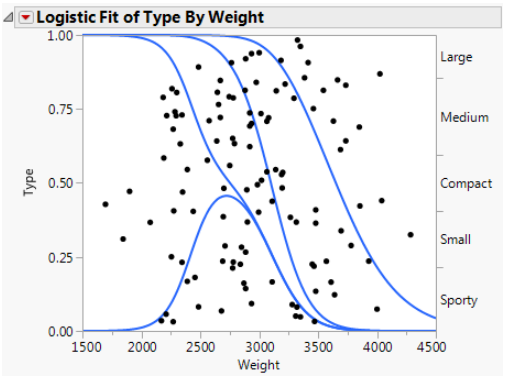
Use the Logistic platform to fit a logistic regression model to a categorical response with a continuous *x* predictor. Options include a ROC curve, lift curve, and odds ratio estimates. The fitted model estimates probabilities for each *x* value. The Logistic platform is the *nominal* or *ordinal* by *continuous* personality of the Fit Y by X platform. There is a distinction between nominal and ordinal responses on this platform:

- Nominal logistic regression estimates a set of curves to partition the probability among the responses.
- Ordinal logistic regression models the probability of being less than or equal to a given response. This has the effect of estimating a single logistic curve, which is shifted horizontally to produce probabilities for the ordered categories. This model is less complex and is recommended for ordered responses.

Figure 8.1 Examples of Logistic Regression



Ordinal Logistic Regression



Nominal Logistic Regression

Contents

Overview of Logistic Regression . . . 281
Nominal Logistic Regression . . . 281
Ordinal Logistic Regression . . . 281
Example of Nominal Logistic Regression . . . 282
Launch the Logistic Platform . . . 283
Data Structure . . . 284
The Logistic Report . . . 285
Logistic Plot . . . 286
Iterations . . . 286
Whole Model Test . . . 286
Fit Details . . . 288
Parameter Estimates . . . 289
Logistic Platform Options . . . 289
ROC Curves . . . 290
Save Probability Formula . . . 291
Inverse Prediction . . . 291
Additional Examples of Logistic Regression . . . 292
Example of Ordinal Logistic Regression . . . 292
Additional Example of a Logistic Plot . . . 294
Example of ROC Curves . . . 296
Example of Inverse Prediction Using the Crosshair Tool . . . 297
Example of Inverse Prediction Using the Inverse Prediction Option . . . 298
Statistical Details for the Logistic Platform . . . 300

Overview of Logistic Regression

Logistic regression has a long tradition with widely varying applications such as modeling dose-response data and purchase-choice data. Unfortunately, many introductory statistics courses do not cover this fairly simple method. Many texts in categorical statistics cover it (Agresti 1990), in addition to texts on logistic regression (Hosmer and Lemeshow 1989). Some analysts use the method with a different distribution function, the normal. In that case, it is called *probit analysis*. Some analysts use discriminant analysis instead of logistic regression because they prefer to think of the continuous variables as *Y*s and the categories as *X*s and work backward. However, discriminant analysis assumes that the continuous data are normally distributed random responses, rather than fixed regressors.

Simple logistic regression is a more graphical and simplified version of the general facility for categorical responses in the Fit Model platform. For examples of more complex logistic regression models, see *Fitting Linear Models*.

Nominal Logistic Regression

Nominal logistic regression estimates the probability of choosing one of the response levels as a smooth function of the *x* factor. The fitted probabilities must be between 0 and 1, and must sum to 1 across the response levels for a given factor value.

In a logistic probability plot, the vertical axis represents probability. For *k* response levels, *k* - 1 smooth curves partition the total probability (which equals 1) among the response levels. The fitting principle for a logistic regression minimizes the sum of the negative natural logarithms of the probabilities fitted to the response events that occur (that is, maximum likelihood).

Ordinal Logistic Regression

When *Y* is ordinal, a modified version of logistic regression is used for fitting. The cumulative probability of being at or below each response level is modeled by a curve. The curves are the same for each level except that they are shifted to the right or left.

The ordinal logistic model fits a different intercept, but the same slope, for each of *r* - 1 cumulative logistic comparisons, where *r* is the number of response levels. Each parameter estimate can be examined and tested individually, although this is seldom of much interest.

The ordinal model is preferred to the nominal model when it is appropriate because it has fewer parameters to estimate. In fact, it is practical to fit ordinal responses with hundreds of response levels.

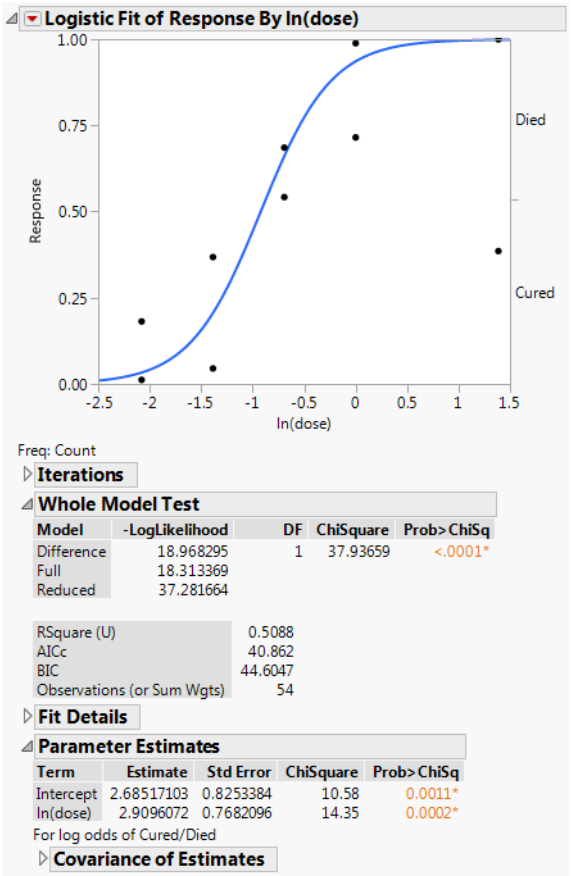
Example of Nominal Logistic Regression

Learn how to examine the relationship between a nominal Y response and a continuous X factor. The data in this example comes from an experiment where 5 groups, each containing 12 rabbits, were injected with streptococcus bacteria. Once the rabbits were confirmed to have the bacteria in their system, they were given different doses of penicillin. You want to find out whether the natural log (ln(dose)) of dosage amounts has any effect on whether the rabbits are cured.

1. Select **Help > Sample Data Library** and open Penicillin.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Response and click **Y, Response**.
4. Select ln(dose) and click **X, Factor**.

   Notice that JMP automatically fills in Count for **Freq**. Count was previously assigned the role of Freq.
5. From the Target Level list, select Cured.
6. Click **OK**.

Figure 8.2 Example of Nominal Logistic Report



The plot shows the fitted model as a function of ln(dose). The fitted model is the predicted probability of being cured. The *p*-value is significant, indicating that the dosage amounts have a significant effect on whether the rabbits are cured.

Tip: To change the response level that is analyzed, specify a Target Level in the launch window or use the Value Order column property.

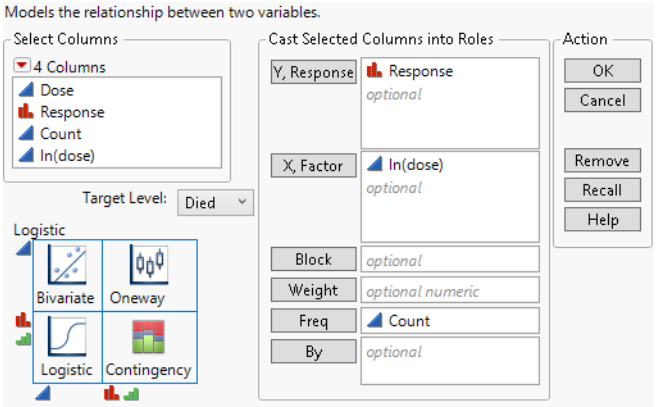
Launch the Logistic Platform

To perform a logistic analysis, do the following:

1. Select **Analyze > Fit Y by X**.
2. Enter a nominal or ordinal column for **Y, Response**.

3. Enter a continuous column for **X, Factor**.

Figure 8.3 Fit Y by X Launch Window



The word Logistic appears above the image, to indicate that you are performing a logistic analysis.

Note: You can also launch a logistic analysis from the JMP Starter window. Select **View** > **JMP Starter** > **Basic** > **Logistic**.

When the response is binary and has a nominal modeling type, a Target Level menu appears in the launch window. Use this menu to specify the level of the response whose probability you want to model.

Tip: The default value is the higher of the two levels based on the order of the levels. Check that the default value is the level that you want to model.

For more information about the Fit Y by X launch window, see "Introduction to Fit Y by X". For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Data Structure

Your data can consist of unsummarized or summarized data:

Unsummarized data There is one row for each observation containing its X value and its Y value.

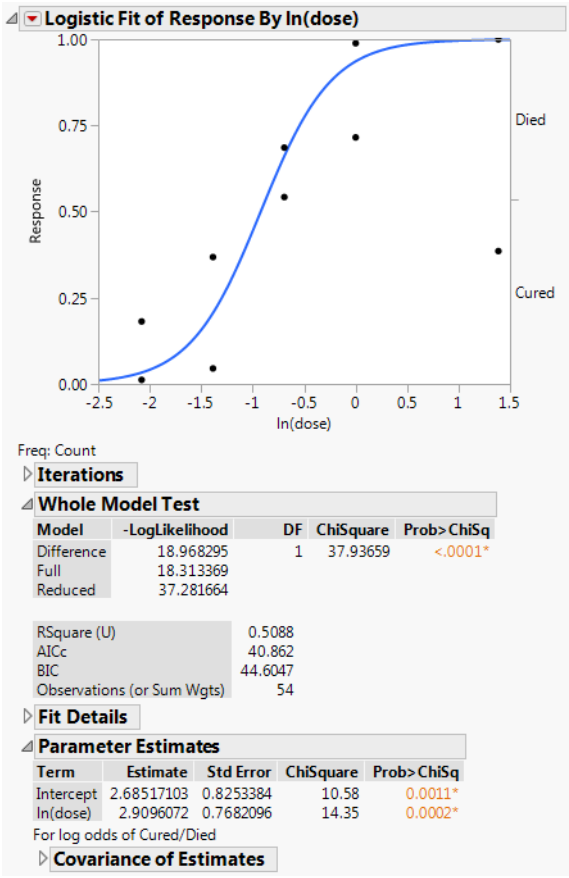
Summarized data Each row represents a set of observations with common X and Y values. The data table must contain a column that gives the counts for each row. Enter this column as Freq in the launch window.

The Logistic Report

The Logistic report window contains the Logistic plot, the Iterations report, the Whole Model Test report, the Fit Details report, and the Parameter Estimates report.

To produce the plot shown in Figure 8.4, follow the instructions in "Example of Nominal Logistic Regression".

Figure 8.4 Example of a Logistic Report



Note: The red triangle menu provides more options that can add to the initial report window. See "Logistic Platform Options".

Logistic Plot

The logistic probability plot illustrates the logistic fit. At each value on the horizontal axis, the probability scale in the vertical direction is partitioned into probabilities for each response category. The probabilities are measured as the vertical distance between the curves, with the total across all response category probabilities summing to 1.

The points in the logistic plot represent the observations from the data table. The horizontal position of each point is determined by its value of continuous factor. The vertical position of each point is randomly chosen to be between curves that correspond to the value of its response category. This jittering of the points makes it easier to see where the points are most dense, but the vertical position does not correspond to the values on the vertical axis. Because a fixed random seed is used, the vertical positions do not differ across multiple fits of the same model.

You can replace variables in the plot by clicking on a variable in the Columns panel of the associated data table and dragging it onto an axis.

Note: See also "Additional Example of a Logistic Plot".

Iterations

The Iterations report shows each iteration and the evaluated criteria that determine whether the logistic model has converged. Iterations appear only for nominal logistic regression.

Whole Model Test

The Whole Model Test report shows if the model fits better than constant response probabilities. This report is analogous to the Analysis of Variance report for a continuous response model. It is a specific likelihood ratio Chi-square test that evaluates how well the categorical model fits the data.

The negative sum of natural logs of the observed probabilities is called the negative log-likelihood (–LogLikelihood). The negative log-likelihood for categorical data plays the same role as sums of squares in continuous data: twice the difference in the negative log-likelihood from the model fitted by the data and the model with equal probabilities is a Chi-square statistic. This test statistic examines the hypothesis that the x variable has no effect on the responses.

Values of the **RSquare (U)** (sometimes denoted as R^2) range from 0 to 1. High R^2 values are indicative of a good model fit, and are rare in categorical models.

The Whole Model Test report contains the following columns:

Model Sometimes called Source.

- The **Reduced** model contains only an intercept.
- The **Full** model contains all of the effects as well as the intercept.
- The **Difference** is the difference of the log-likelihoods of the full and reduced models.

DF Records the degrees of freedom associated with the model.

–LogLikelihood Measures variation, sometimes called *uncertainty*, in the sample.

Full (the full model) is the negative log-likelihood (or uncertainty) calculated after fitting the model. The fitting process involves predicting response rates with a linear model and a logistic response function. This value is minimized by the fitting process.

Reduced (the reduced model) is the negative log-likelihood (or uncertainty) for the case when the probabilities are estimated by fixed background rates. This is the background uncertainty when the model has no effects.

The difference of these two negative log-likelihoods is the reduction due to fitting the model. Two times this value is the likelihood ratio Chi-square test statistic.

See *Fitting Linear Models*.

Chi-Square The likelihood ratio Chi-square test of the hypothesis that the model fits no better than fixed response rates across the whole sample. It is twice the –LogLikelihood for the Difference Model. It is two times the difference of two negative log-likelihoods, one with whole-population response probabilities and one with each-population response rates. See "Statistical Details for the Logistic Platform".

Prob>ChiSq The observed significance probability, often called the *p*-value, for the Chi-square test. It is the probability of getting a Chi-square value greater than the one computed. Models are often judged significant if this probability is below 0.05.

RSquare (U) The proportion of the total uncertainty that is attributed to the model fit, defined as the **Difference** negative log-likelihood value divided by the **Reduced** negative log-likelihood value. An RSquare (U) value of 1 indicates that the predicted probabilities for events that occur are equal to one: There is no uncertainty in predicted probabilities. Because certainty in the predicted probabilities is rare for logistic models, RSquare (U) tends to be small. See "Statistical Details for the Logistic Platform".

Note: RSquare (U) is also known as *McFadden's pseudo R-square*.

AICc The corrected Akaike Information Criterion. See *Fitting Linear Models*.

BIC The Bayesian Information Criterion. See *Fitting Linear Models*.

Observations Sometimes called Sum Wgts. The total sample size used in computations. If you specified a **Weight** variable, this is the sum of the weights.

Fit Details

The Fit Details report in the Logistic Platform contains the following statistics:

Measure Contains the following measures of fit:

Entropy RSquare Compares the log-likelihoods from the fitted model and the constant probability model. This is the same as Rsquare (U). See "Statistical Details for the Logistic Platform".

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991).

Mean -Log p The average of -log(p), where p is the fitted probability associated with the event that occurred.

RASE The root average square error, where the differences are between the response and p (the fitted probability for the event that actually occurred).

Mean Abs Dev The average of the absolute values of the differences between the response and p (the fitted probability for the event that actually occurred).

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category.

N The number of observations.

For Entropy RSquare and Generalized RSquare, values closer to 1 indicate a better fit. For Mean -Log p, RASE, Mean Abs Dev, and Misclassification Rate, smaller values indicate a better fit.

Training The value of the measure of fit.

Definition The algebraic definition of the measure of fit.

Parameter Estimates

The nominal logistic model fits a parameter for the intercept and slope for each of $k - 1$ logistic comparisons, where k is the number of response levels. The Parameter Estimates report lists these estimates. Each parameter estimate can be examined and tested individually, although this is seldom of much interest.

Term Lists each parameter in the logistic model. There is an intercept and a slope term for the factor at each level of the response variable, except the last level.

Estimate Lists the parameter estimates given by the logistic model.

Std Error Lists the standard error of each parameter estimate. They are used to compute the statistical tests that compare each term to zero.

Chi-Square Lists the Wald tests for the hypotheses that each of the parameters is zero. The Wald Chi-square is computed as (Estimate / Std Error)^2.

Prob>ChiSq Lists the observed significance probabilities for the Chi-square tests.

Covariance of Estimates

Reports the estimated variances of the parameter estimates, and the estimated covariances between the parameter estimates. The square root of the variance estimates is the same as those given in the **Std Error** section.

Logistic Platform Options

The Logistic Fit red triangle menu contains options that affect the Logistic plot. If you have specified multiple Y response variables, the Fit Group red triangle menu appears instead. See *Fitting Linear Models*.

Odds Ratios Adds odds ratios to the Parameter Estimates report. See *Fitting Linear Models*.

This option is available only for a response with two levels.

Inverse Prediction Prediction of x values from given y values. See "Inverse Prediction".

This option is available only for two-level nominal responses.

Logistic Plot Shows or hides the logistic plot.

Plot Options Contains the following options:

Show Points Shows or hides the points in the logistic plot.

Show Rate Curve Is useful only if you have several points for each x-value. In these cases, you get reasonable estimates of the rate at each value, and compare this rate with the fitted logistic curve. To prevent too many degenerate points, usually at zero or one, JMP shows only the rate value if there are at least three points at the x-value.

Line Color Enables you to select the color of the plot curves.

ROC Curve Produces a Receiver Operating Characteristic (ROC) curve. The ROC curve is a plot of sensitivity versus (1 - specificity) for each value of x. See "ROC Curves".

Lift Curve Produces a lift curve for the model. A lift curve shows the same information as a ROC curve, but in a way to dramatize the richness of the ordering at the beginning. The vertical axis shows the ratio of how rich that portion of the population is in the chosen response level compared to the rate of that response level as a whole. See *Fitting Linear Models* for more information about lift curves.

Save Probability Formula Creates new data table columns that contain formulas. See "Save Probability Formula".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

ROC Curves

When you do a simple logistic regression with a binary outcome, there is a platform option for a ROC curve for that analysis. After selecting the ROC Curve option, you must specify which level to use as the *positive* response.

Note: See also "Example of ROC Curves".

Suppose you have an x value that is a diagnostic measurement and you want to determine a threshold value of x that indicates the following:

- A condition exists if the x value is greater than the threshold.

- A condition does not exist if the x value is less than the threshold.

For example, you could measure a blood component level as a diagnostic test to predict a type of cancer. Now consider the diagnostic test as you vary the threshold and thus cause more or fewer false positives and false negatives. You then plot those rates. The ideal is to have a very narrow range of x criterion values that best divides true negatives and true positives. The Receiver Operating Characteristic (ROC) curve shows how rapidly this transition happens. The goal of the ROC curve is to have diagnostics that maximize the area under the curve.

Two standard definitions are used in medicine:

- *Sensitivity*, the probability that a given x value (a test or measure) correctly predicts an existing condition. For a given x, the probability of incorrectly predicting the existence of a condition is 1 – sensitivity.
- *Specificity*, the probability that a test correctly predicts that a condition does not exist.

A ROC curve is a plot of sensitivity by (1 – specificity) for each value of x. The area under the ROC curve is a common index used to summarize the information contained in the curve.

If a test predicted perfectly, it would have a value above which the entire abnormal population would fall and below which all normal values would fall. It would be perfectly sensitive and then pass through the point (0,1) on the grid. The closer the ROC curve comes to this ideal point, the better its discriminating ability. A test with no predictive ability produces a curve that follows the diagonal of the grid (DeLong et al. 1988).

The ROC curve is a graphical representation of the relationship between false-positive and true-positive rates. A standard way to evaluate the relationship is with the area under the curve, shown below the plot in the report. In the plot, a yellow line is drawn at a 45-degree angle tangent to the ROC Curve. This marks a good cutoff point under the assumption that false negatives and false positives have similar costs.

Save Probability Formula

The **Save Probability Formula** option creates new data table columns. These data table columns save the following:

- formulas for linear combinations (typically called logits) of the x factor
- prediction formulas for the response level probabilities
- a prediction formula that gives the most likely response

Inverse Prediction

Inverse prediction is the opposite of prediction. It is the prediction of x values from given y values. But in logistic regression, instead of a y value, you have the probability attributed to one of the Y levels. This feature works only for two-level nominal responses.

The Fit Model platform also has an option that gives an inverse prediction with confidence limits. See *Fitting Linear Models* for more information about inverse prediction.

Note: See also "Example of Inverse Prediction Using the Crosshair Tool" and "Example of Inverse Prediction Using the Inverse Prediction Option".

Additional Examples of Logistic Regression

This section contains additional examples using logistic regression in the Logistic platform.

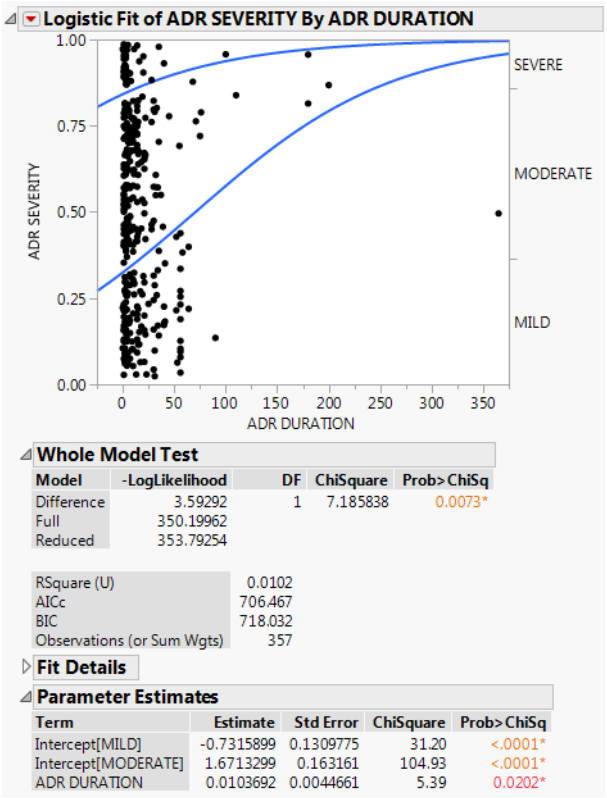
- "Example of Ordinal Logistic Regression"
- "Additional Example of a Logistic Plot"
- "Example of ROC Curves"
- "Example of Inverse Prediction Using the Crosshair Tool"
- "Example of Inverse Prediction Using the Inverse Prediction Option"

Example of Ordinal Logistic Regression

This example shows you how to examine the relationship between an ordinal Y response and a continuous X factor. In this example, suppose you want to model the severity of an adverse event as a function of duration of adverse events.

1. Select **Help > Sample Data Library** and open AdverseR.jmp.
2. Right-click the icon to the left of ADR SEVERITY and change the modeling type to ordinal.
3. Select **Analyze > Fit Y by X**.
4. Select ADR SEVERITY and click **Y, Response**.
5. Select ADR DURATION and click **X, Factor**.
6. Click **OK**.

Figure 8.5 Example of Ordinal Logistic Report



You interpret this report the same way as the nominal report. See "The Logistic Report".

In the plot, markers for the data are drawn at their x-coordinate. When several data points appear at the same y position, the points are jittered. That is, small spaces appear between the data points so that you can see each point more clearly.

Where there are many points, the curves are pushed apart. Where there are few to no points, the curves are close together. The data pushes the curves in that way because the criterion that is maximized is the product of the probabilities fitted by the model. The fit tries to avoid points attributed to have a small probability, which are points crowded by the curves of fit. See *Fitting Linear Models* for more information about computational details.

For more information about the Whole Model Test report and the Parameter Estimates report, see "The Logistic Report". In the Parameter Estimates report, an intercept parameter is estimated for every response level except the last, but there is only one slope parameter. The intercept parameters show the spacing of the response levels. They always increase monotonically.

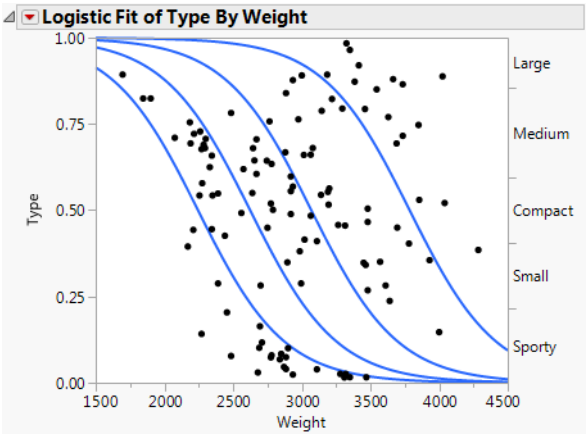
Additional Example of a Logistic Plot

In this example, suppose you want to use car weight to predict car size (Type) for 116 cars. Car size can be one of the following, from smallest to largest: Sporty, Small, Compact, Medium, or Large.

1. Select **Help > Sample Data Library** and open Car Physical Data.jmp.
2. In the Columns panel, right-click the icon to the left of Type, and select **Ordinal**.
3. Right-click Type and select **Column Info**.
4. From the Column Properties menu, select **Value Order**.
5. Verify that the data are in the following top-down order: Sporty, Small, Compact, Medium, Large.
6. Click **OK**.
7. Select **Analyze > Fit Y by X**.
8. Select Type and click **Y, Response**.
9. Select Weight and click **X, Factor**.
10. Click **OK**.

    The report window appears.

Figure 8.6 Example of Type by Weight Logistic Plot



Note the following observations:

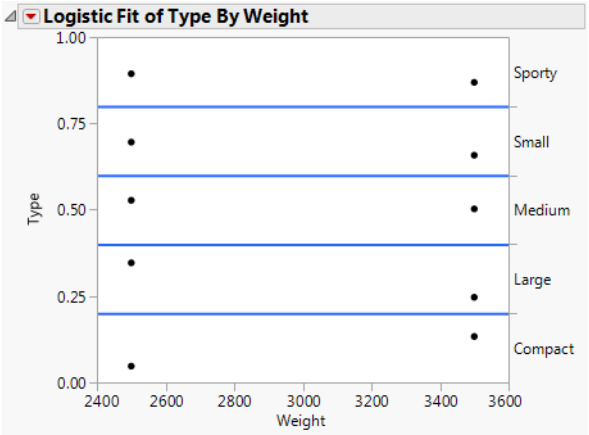
- The first (bottom) curve represents the probability that a car at a given weight is Sporty.
- The second curve represents the probability that a car is Small or Sporty. Looking only at the distance between the first and second curves corresponds to the probability of being Small.

- As you might expect, heavier cars are more likely to be Large.
- Markers for the data are drawn at their x-coordinate. The y position is jittered randomly within the range corresponding to the response category for that row.

If the x -variable has no effect on the response, then the fitted lines are horizontal and the probabilities are constant for each response across the continuous factor range. Figure 8.7 shows a logistic plot where Weight is not useful for predicting Type.

Figure 8.7 Examples of Sample Data Table and Logistic Plot Showing No y by x Relationship

	Type	Weight
1	Large	2500
2	Medium	2500
3	Compact	2500
4	Small	2500
5	Sporty	2500
6	Large	3500
7	Medium	3500
8	Compact	3500
9	Small	3500
10	Sporty	3500

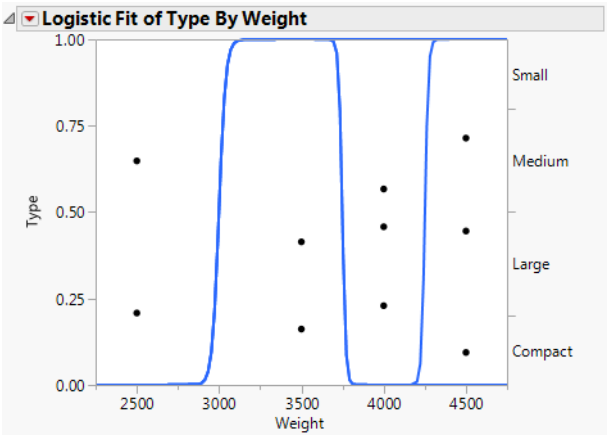


Note: To re-create the plots in Figure 8.7 and Figure 8.8, you must first create the data tables shown here, and then perform steps 7-10 at the beginning of this section.

If the response is completely predicted by the value of the factor, then the logistic curves are effectively vertical. The prediction of a response is near certain (the probability is almost 1) at each of the factor levels. Figure 8.8 shows a logistic plot where Weight almost perfectly predicts Type.

Figure 8.8 Examples of Sample Data Table and Logistic Plot Showing an Almost Perfect *y* by *x* Relationship

	Type	Weight
1	Small	2500
2	Small	2500
3	Compact	3500
4	Compact	3500
5	Medium	4000
6	Medium	4000
7	Medium	4000
8	Large	4500
9	Large	4500
10	Large	4500



In this case, the parameter estimates become very large and are labeled *unstable* in the regression report. In these cases, you might consider using the Generalized Linear Model personality with Firth bias-adjusted estimates. See *Fitting Linear Models*.

Example of ROC Curves

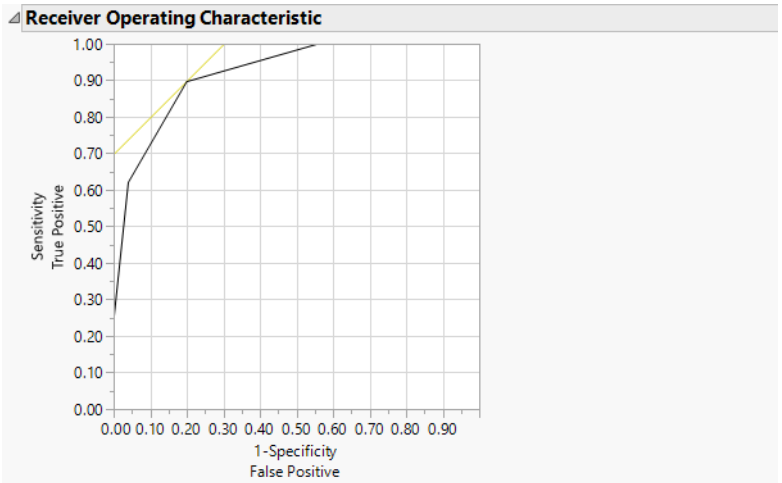
1. Select **Help > Sample Data Library** and open Penicillin.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Response and click **Y, Response**.
4. Select ln(dose) and click **X, Factor**.

   Notice that JMP automatically fills in Count for **Freq**. Count was previously assigned the role of Freq.
5. From the Target Level list, select Cured.
6. Click **OK**.
7. Click the red triangle next to Logistic Fit of Response By ln(dose) and select **ROC Curve**.

Note: This example shows a ROC Curve for a nominal response. For more information about ordinal ROC curves, see *Predictive and Specialized Modeling*.

The results for the response by ln(dose) example are shown here. The ROC curve plots the probabilities described above, for predicting response. Note that in the ROC Table, the row with the highest Sens-(1-Spec) is marked with an asterisk.

Figure 8.9 Examples of ROC Curve and Table



Using Response='Cured' to be the positive level

AUC
0.92000

ROC Table

X	Prob	1-Specificity	Sensitivity	Sens-(1-Spec)		True Pos	True Neg	False Pos	False Neg
.	.	0.0000	0.0000	0.0000		0	25	0	29
1.38629	0.9988	0.0000	0.2414	0.2414		7	25	0	22
0.00000	0.9361	0.0400	0.6207	0.5807		18	24	1	11
-0.69315	0.6611	0.2000	0.8966	0.6966	*	26	20	5	3
-1.38629	0.2061	0.5600	1.0000	0.4400		29	11	14	0
-2.07944	0.0334	1.0000	1.0000	0.0000		29	0	25	0
-2.07944	0.0334	1.0000	1.0000	0.0000		29	0	25	0

Since the ROC curve is well above a diagonal line, you conclude that the model has good predictive ability.

Example of Inverse Prediction Using the Crosshair Tool

This example shows you how to use the crosshair tool in JMP to visually approximate an inverse prediction. In a study of rabbits who were given penicillin, you want to know what dose of penicillin results in a 0.5 probability of curing a rabbit. In this case, the inverse prediction for 0.5 is called the *ED50*, the effective dose corresponding to a 50% survival rate.

Determine which dose is equally likely either to cure or to be lethal.

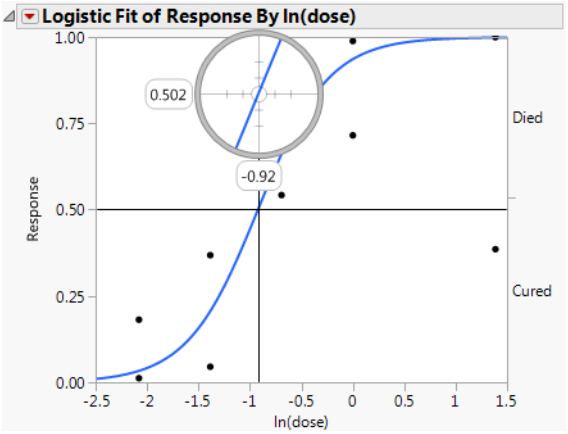
1. Select **Help > Sample Data Library** and open Penicillin.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Response and click **Y, Response**.
4. Select ln(dose) and click **X, Factor**.

Notice that JMP automatically fills in Count for **Freq**. Count was previously assigned the role of Freq.

5. From the Target Level list, select Cured.
6. Click **OK**.
7. Click the crosshairs tool.
8. Place the horizontal crosshair line at about 0.5 on the vertical (Response) probability axis.
9. Move the cross-hair intersection to the prediction line, and read the ln(dose) value that shows on the horizontal axis.

In this example, a rabbit with a ln(dose)of approximately -0.9 is equally likely to be cured as it is to die.

Figure 8.10 Example of Crosshair Tool on Logistic Plot



Example of Inverse Prediction Using the Inverse Prediction Option

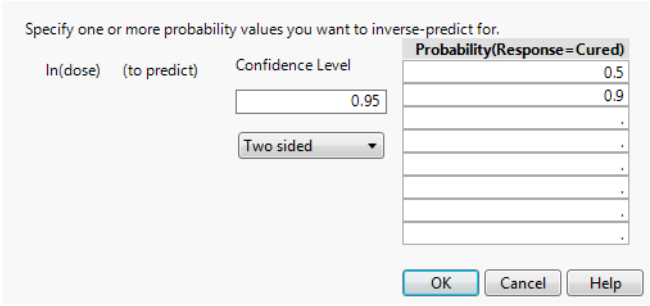
If your response has exactly two levels, the **Inverse Prediction** option enables you to request an exact inverse prediction. You are given the *x* value corresponding to a given probability of the target category, as well as a confidence interval for that *x* value.

1. Select **Help > Sample Data Library** and open Penicillin.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Response and click **Y, Response**.
4. Select ln(dose) and click **X, Factor**.

   Notice that JMP automatically fills in Count for **Freq**. Count was previously assigned the role of Freq.
5. From the Target Level list, select Cured.

6. Click **OK**.
7. Click the red triangle next to Logistic Fit of Response By ln(dose) and select **Inverse Prediction** (Figure 8.11).
8. Type 0.95 for the **Confidence Level**.
9. Select **Two sided** for the confidence interval.
10. Request the response probability of interest. Type 0.5 and 0.9 for this example, which indicates you are requesting the values for ln(dose) that correspond to a 0.5 and 0.9 probability of being cured.

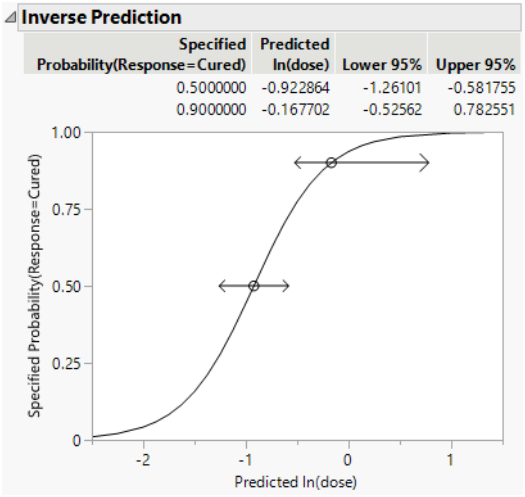
Figure 8.11 Inverse Prediction Window



11. Click **OK**.

    The Inverse Prediction plot appears.

Figure 8.12 Example of Inverse Prediction Plot



The estimates of the x values and the confidence intervals are shown in the report as well as in the probability plot. For example, the value of ln(dose) that results in a 90% probability of being cured is estimated to be between -0.526 and 0.783.

Statistical Details for the Logistic Platform

This section contains statistical details for the Whole Model Test report.

Chi-Square

The Chi-Square statistic is sometimes denoted G^2 and is defined as follows:

$$G^2 = 2(\sum -\ln p(\text{background}) - \sum -\ln p(\text{model}))$$

where the summations are over all observations instead of all cells.

RSquare (U)

The ratio of this test statistic to the background log-likelihood is subtracted from 1 to calculate R^2. More simply, RSquare (U) is computed as follows:

$$\frac{\text{negative log-likelihood for Difference}}{\text{negative log-likelihood for Reduced}}$$

using quantities from the Whole Model Test report.

Note: RSquare (U) is also known as *McFadden's pseudo R-square*.

Chapter 9

Tabulate

Create Summary Tables Interactively

Use the Tabulate platform to interactively construct summary tables, or pivot tables, of descriptive statistics. The Tabulate platform is an easy and flexible way to present summary data in tabular form. Tables are built from assigning data table columns as rows or columns in the tabulation and then assigning the desired summary statistics.

Figure 9.1 Tabulate Examples

	sex											
	F						M					
	age						age					
	12	13	14	15	16	17	12	13	14	15	16	17
	weight	weight	weight	weight	weight	weight	weight	weight	weight	weight	weight	weight
Min	64	67	81	92	112	116	79	79	92	104	128	134
Mean	100.2	95.3	96.6	102.0	113.5	116.0	97.0	94.3	103.9	110.8	128.0	153.0
Max	145	112	142	112	115	116	128	105	119	128	128	172

		sex							
		Female				Male			
		marital status				marital status			
		Married		Single		Married		Single	
		Mean	Std Dev	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
country	size	age	age	age	age	age	age	age	age
American	Large	33.6	8.107	41.0	.	34.7	3.931	32.0	6.265
	Medium	31.4	5.827	29.0	9.258	31.3	5.413	32.1	11.05
	Small	31.0	5.657	29.0	9.539	31.8	4.813	26.5	6.455
European	Large	34.0	7.071	28.0	.	.	.	26.0	.
	Medium	31.0	5.06	28.7	5.508	32.3	5.62	31.0	10.13
	Small	29.8	6.611	28.0	1.414	33.8	4.381	25.7	2.517
Japanese	Large	25.0	.	.	.	32.0	.	.	.
	Medium	30.5	4.993	28.0	3.071	32.3	3.878	27.4	5.016
	Small	29.6	4.251	31.1	9.562	29.8	5.357	28.7	4.739
country									
American		31.9	6.452	30.0	9.115	32.6	4.919	31.0	8.179
European		31.0	5.612	28.3	3.559	33.3	4.608	28.4	7.328
Japanese		29.8	4.54	30.1	8.113	30.9	4.822	28.3	4.781

		Mean		
Type	Size Co	Profits ($M)	Sales ($M)	profit/emp
Computer	big	1089.9	20597.48	4530.478
	medium	-85.75	3018.85	-3462.51
	small	44.94	1758.06	7998.815
	All	240.87	5652.02	6159.015
Pharmaceutical	big	894.42	7474.04	17140.70
	medium	698.98	4261.06	24035.11
	small	156.95	1083.75	38337.19
	All	690.08	5070.25	23546.12
All	All	409.32	5433.86	12679.18
Type				
Computer		240.87	5652.02	6159.015
Pharmaceutical		690.08	5070.25	23546.12
All		409.32	5433.86	12679.18

Contents

Example of the Tabulate Platform . . . 303
Launch the Tabulate Platform . . . 308
Use the Dialog . . . 310
Add Statistics . . . 311
The Tabulate Output . . . 314
Analysis Columns . . . 315
Grouping Columns . . . 315
Column and Row Tables . . . 316
Edit Tables . . . 317
Tabulate Platform Options . . . 318
Show Test Build Panel . . . 319
Right-Click Menu for Columns . . . 319
Additional Examples of the Tabulate Platform . . . 320
Example of Creating Different Tables and Rearranging Contents . . . 320
Example of Combining Columns into a Single Table . . . 324
Example Using a Page Column . . . 326

Example of the Tabulate Platform

Learn how to summarize data in a tabular format using the Tabulate platform. In this example, you have data containing height measurements for male and female students. You want to create a table that shows the mean height for males and females and the aggregate mean for both sexes.

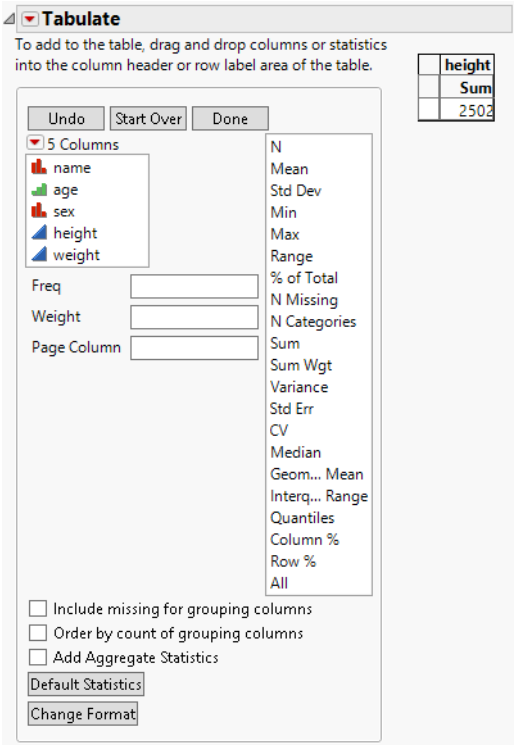
Figure 9.2 Table Showing Mean Height

	height
sex	Mean
F	60.9
M	63.9
All	62.6

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Tabulate**.

   Since height is the variable that you are examining, you want it to appear at the top of the table.
3. Click height and drag it into the Drop zone for columns.

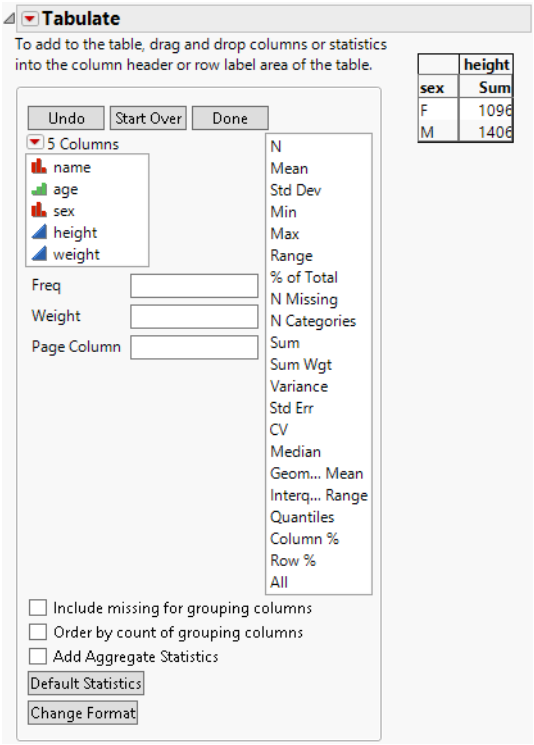
Figure 9.3 Height Variable Added



You want the statistics by sex, and you want sex to appear on the side.

4. Click sex and drag it into the blank cell next to the number 2502.

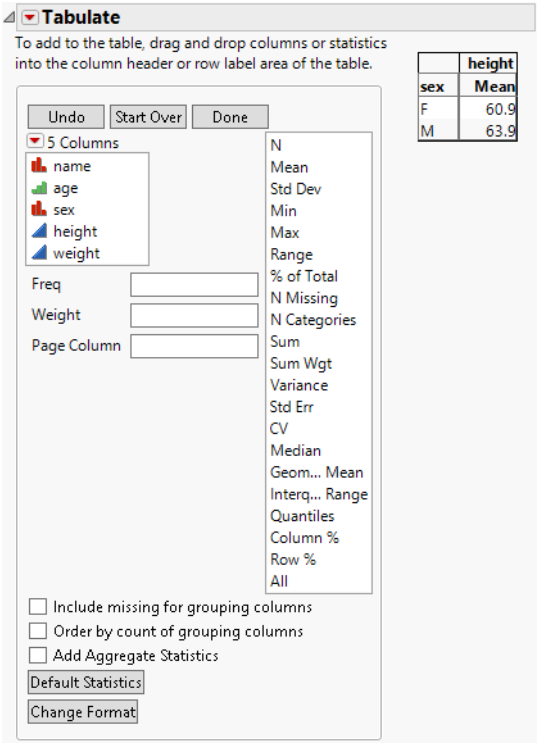
Figure 9.4 Sex Variable Added



Instead of the sum, you want it to show the mean.

5. Click Mean and drag it on top of Sum.

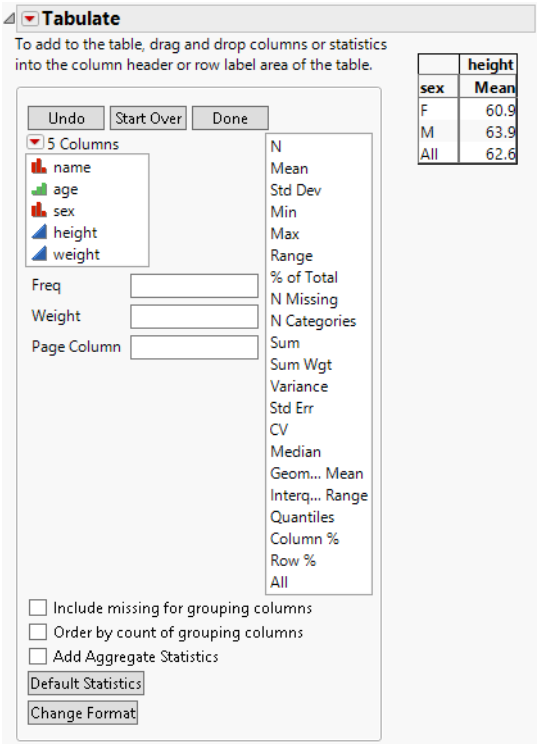
Figure 9.5 Mean Statistic Added



You also want to see the combined mean for males and females.

6. Click All and drag it on top of sex. Or, you can simply select the **Add Aggregate Statistics** check box.

Figure 9.6 All Statistic Added



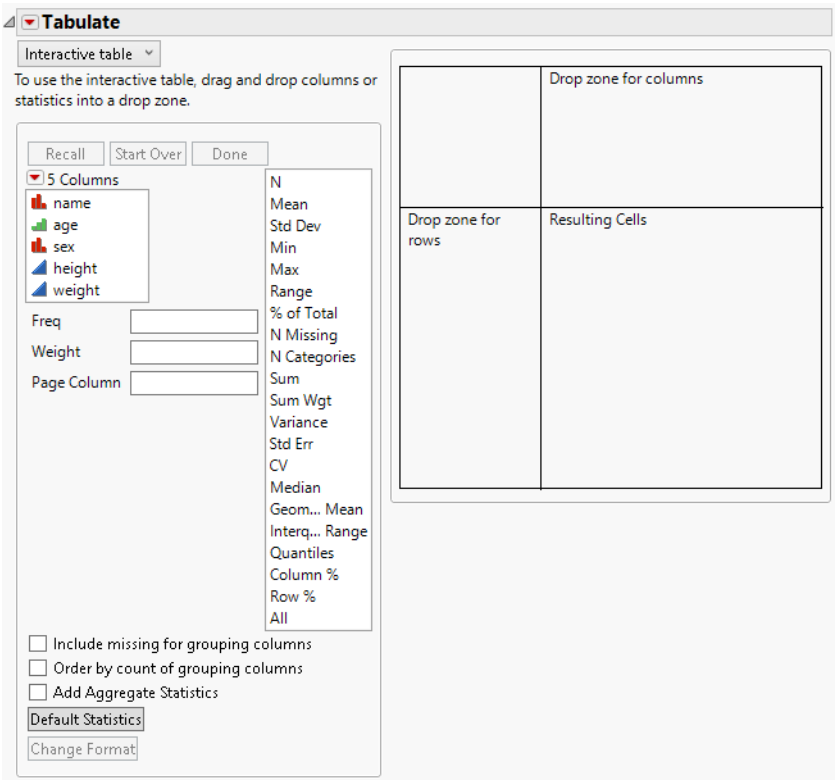
7. (Optional) Click **Done**.

The completed table shows the mean height for females, males, and the combined mean height for both.

Launch the Tabulate Platform

To launch the Tabulate platform, select **Analyze > Tabulate**.

Figure 9.7 The Tabulate Interactive Table



Note: For more information about red triangle options, see "Tabulate Platform Options". For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Tabulate window contains the following options:

Interactive table/Dialog Switch between the two modes. Use the interactive table mode to drag and drop items, creating a custom table. Use the dialog mode to create a simple table using a fixed format. See "Use the Dialog".

Statistics options Lists standard statistics. Drag any statistic from the list to the table to incorporate it. See "Add Statistics".

Drop zone for columns Drag and drop columns or statistics here to create columns.

Note: If the data table contains columns with names equal to those in the Statistics options, be sure to drag and drop the column name from the column list. Otherwise, JMP might substitute the statistic of the same name in the table.

Drop zone for rows Drag and drop columns or statistics here to create rows.

Tip: You can also select one or more columns in the columns list, select one or more of the statistics, and then Alt-click (Option-click on macOS) on a drop zone to create rows or columns in the table.

Resulting cells Shows the resulting cells based on the columns or statistics that you drag and drop.

Freq Identifies the data table column whose values assign a frequency to each row. This option is useful when a frequency is assigned to each row in summarized data.

Weight Identifies the data table column whose variables assign weight (such as importance or influence) to the data.

Page Column Generates separate tables for each category of a nominal or ordinal column. See "Example Using a Page Column".

Include missing for grouping columns Creates a separate group for missing values in grouping columns. When unchecked, missing values are not included in the table. Note that any missing value codes that you have defined as column properties are taken into account.

Order by count of grouping columns Changes the order of the table to be in descending order of the values of the grouping columns.

Add Aggregate Statistics Adds aggregate statistics for all rows and columns.

Default Statistics Enables you to change the default statistics that appear when you drag and drop analysis or non-analysis (for example, grouping) columns.

Change Format Enables you to change the numeric format for displaying specific statistics. See "Change Numeric Formats".

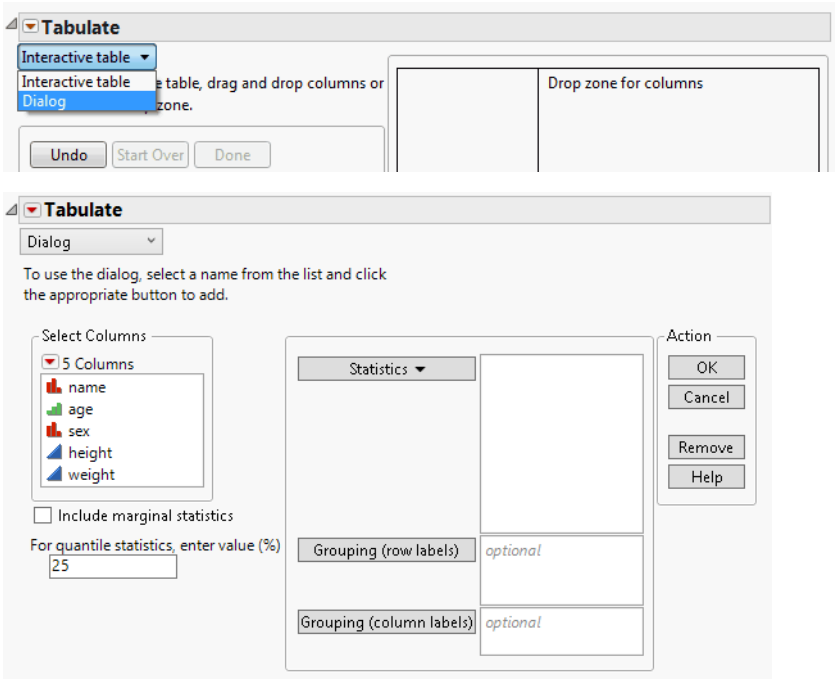
Change Plot Scale (Appears only if **Show Chart** is selected from the red triangle menu.) Enables you to specify a uniform custom scale.

Uniform plot scale (Appears only if **Show Chart** is selected from the red triangle menu.) Deselect this box for each column of bars to use the scale determined separately from the data in each displayed column.

Use the Dialog

If you prefer not to drag and drop and build a table interactively, you can create a simple table using the Dialog interface in the Tabulate platform. After selecting **Analyze > Tabulate**, select **Dialog** from the menu, as shown in Figure 9.8. You can make changes to the table by selecting **Show Control Panel** from the red triangle menu, and then drag and drop new items into the table.

Figure 9.8 Using the Dialog



The dialog contains the following options:

Include marginal statistics Aggregates summary information for categories of a grouping column.

For quantile statistics, enter value (%) Enter the value at which the specific percentage of the argument is less than or equal to. For example, 75% of the data is less than the 75th quantile. This applies to all grouping columns.

Statistics Once you have selected a column, select a standard statistic to apply to that column. See "Add Statistics".

Grouping (row labels) Select the column to use as the row label.

Grouping (column labels) Select the column to use as the column label.

Add Statistics

Tabulate supports a list of standard statistics that appear in the control panel. You can drag any keyword from that list to the table, just like you do with the columns. Note the following:

Tip: You can select both a column and a statistic at the same time and drag them into the table.

- The statistics associated with each cell are calculated on values of the analysis columns from all observations in that category, as defined by the grouping columns.
- All of the requested statistics have to reside in the same dimension, either in the row table or in the column table.
- If you drag a continuous column into a data area, it is treated as an analysis column.

Note: Analysis columns are numeric, continuous columns for which you want to compute statistics. See "Analysis Columns".

Tabulate uses the following keywords:

N Provides the number of nonmissing values in the column. This is the default statistic when there is no analysis column.

Mean Provides the arithmetic mean of a column's values. It is the sum of nonmissing values (and if defined, multiplied by the **weight** variable) divided by the **Sum Wgt**.

Std Dev Provides the sample standard deviation, computed for the nonmissing values. It is the square root of the sample variance.

Min Provides the smallest nonmissing value in a column.

Max Provides the largest nonmissing value in a column.

Range Provides the difference between **Max** and **Min**.

% of Total Computes the percentage of total of the whole population. The denominator used in the computation is the total of all the included observations, and the numerator is the total for the category. If there is no analysis column, the % of Total is the percentage of total of counts. If there is an analysis column, the % of Total is the percentage of the total of the sum of the analysis column. Thus, the denominator is the sum of the analysis column over all the included observations, and the numerator is the sum of the analysis column for that category. You can request different percentages by dragging the keyword into the table.

  – Dropping one or more grouping columns from the table to the % of Total heading changes the denominator definition. For this, Tabulate uses the sum of these grouping columns for the denominator.

– To get the percentage of the column total, drag all the grouping columns on the row table and drop them onto the **% of Total** heading (same as Column %). Similarly, to get the percentage of the row total, drag all grouping columns on the column table and drop them onto the **% of Total** heading (same as Row %).

N Missing Provides the number of missing values.

N Categories Provides the number of distinct categories in the analysis column.

Sum Provides the sum of all values in the column. This is the default statistic for analysis columns when there are no other statistics for the table.

Sum Wgt Provides the sum of all weight values in a column. Or, if no column is assigned the weight role, **Sum Wgt** is the total number of nonmissing values.

Variance Provides the sample variance, computed for the nonmissing values. It is the sum of squared deviations from the mean, divided by the number of nonmissing values minus one.

Std Err Provides the standard error of the mean. It is the standard deviation divided by the square root of **N**. If a column is assigned the role of weight, then the denominator is the square root of the sum of the weights.

CV (Coefficient of Variation) Provides the measure of dispersion, which is the standard deviation divided by the mean multiplied by one hundred.

Median Provides the 50th percentile, which is the value where half the data are below and half are above or equal to the 50th quantile (median).

Geometric Mean The *n*th root of the product of the data. For example, geometric means are often used to calculate interest rates. The statistic is also helpful when the data contains a large value in a skewed distribution.

Note: Negative values result in missing numbers, and zero values (with no negative values) result in zero.

Interquartile Range Provides the difference between the 3rd quartile and 1st quartile.

Quantiles Provides the value at which the specific percentage of the argument is less than or equal to. For example, 75% of the data is less than the 75th quantile. You can request different quantiles by clicking and dragging the **Quantiles** keyword into the table, and then entering the quantile into the box that appears.

Column % Provides the percent of each cell count to its column total if there is no analysis column. If there is an analysis column, the Column % is the percent of the column total of the sum of the analysis column. For tables with statistics on the top, you can add Column % to tables with multiple row tables (stacked vertically).

Row % Provides the percent of each cell count to its row total if there is no analysis column. If there is an analysis column, the Row % is the percent of the row total of the sum of the analysis column. For tables with statistics on the side, you can add Row % to tables with multiple column tables (side by side tables).

All Aggregates summary information for categories of a grouping column.

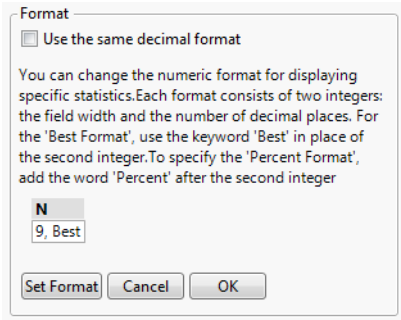
Change Numeric Formats

The formats of each cell depend on the analysis column and the statistics. For counts, the default format has no decimal digits. For each cell defined by some statistics, JMP tries to determine a reasonable format using the format of the analysis column and the statistics requested. To override the default format:

1. Click the **Change Format** button at the bottom of the Tabulate window.
2. In the panel that appears, enter the field width, a comma, and then the number of decimal places that you want displayed in the table (Figure 9.9).
3. To exhibit the cell value in Percent format, add a comma after the number of decimal places and type the word **Percent**.
4. (Optional) If you would like JMP to determine the best format for you to use, type the word **Best** in the text box.

   JMP now considers the precision of each cell value and selects the best way to show it.
5. Click **OK** to implement the changes and close the Format section, or click **Set Format** to see the changes implemented without closing the Format section.

Figure 9.9 Changing Numeric Formats



The Tabulate Output

The Tabulate output consists of one or more column tables concatenated side by side, and one or more row tables concatenated top to bottom. The output might have only a column table or a row table.

Figure 9.10 Tabulate Output

	sex											
	F						M					
	age						age					
	12	13	14	15	16	17	12	13	14	15	16	17
	weight	weight	weight	weight	weight	weight	weight	weight	weight	weight	weight	weight
Min	64	67	81	92	112	116	79	79	92	104	128	134
Mean	100.2	95.3	96.6	102.0	113.5	116.0	97.0	94.3	103.9	110.8	128.0	153.0
Max	145	112	142	112	115	116	128	105	119	128	128	172

		sex							
		Female				Male			
		marital status				marital status			
		Married		Single		Married		Single	
		Mean	Std Dev	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
country	size	age	age	age	age	age	age	age	age
American	Large	33.6	8.107	41.0	.	34.7	3.931	32.0	6.265
	Medium	31.4	5.827	29.0	9.258	31.3	5.413	32.1	11.05
	Small	31.0	5.657	29.0	9.539	31.8	4.813	26.5	6.455
European	Large	34.0	7.071	28.0	.	.	.	26.0	.
	Medium	31.0	5.06	28.7	5.508	32.3	5.62	31.0	10.13
	Small	29.8	6.611	28.0	1.414	33.8	4.381	25.7	2.517
Japanese	Large	25.0	.	.	.	32.0	.	.	.
	Medium	30.5	4.993	28.0	3.071	32.3	3.878	27.4	5.016
	Small	29.6	4.251	31.1	9.562	29.8	5.357	28.7	4.739
country									
American		31.9	6.452	30.0	9.115	32.6	4.919	31.0	8.179
European		31.0	5.612	28.3	3.559	33.3	4.608	28.4	7.328
Japanese		29.8	4.54	30.1	8.113	30.9	4.822	28.3	4.781

		Mean		
Type	Size Co	Profits ($M)	Sales ($M)	profit/emp
Computer	big	1089.9	20597.48	4530.478
	medium	-85.75	3018.85	-3462.51
	small	44.94	1758.06	7998.815
	All	240.87	5652.02	6159.015
Pharmaceutical	big	894.42	7474.04	17140.70
	medium	698.98	4261.06	24035.11
	small	156.95	1083.75	38337.19
	All	690.08	5070.25	23546.12
All	All	409.32	5433.86	12679.18
Type				
Computer		240.87	5652.02	6159.015
Pharmaceutical		690.08	5070.25	23546.12
All		409.32	5433.86	12679.18

Creating a table interactively is an iterative process:

- Click the items (columns or statistics) from the appropriate list, and drag them into the drop zone (for rows or columns). See "Edit Tables", and "Column and Row Tables".
- Add to the table by repeating the drag and drop process. The table updates to reflect the latest addition. If there are already column headings or row labels, you can decide where the addition goes relative to the existing items.

Note the following about clicking and dragging:

- JMP uses the modeling type to determine a column's role. Continuous columns are assumed to be analysis columns. See "Analysis Columns". Ordinal or nominal columns are assumed to be grouping columns. See "Grouping Columns".
- When you drag and drop multiple columns into the initial table:

  - If the columns share a set of common values, they are combined into a single table. A crosstabulation of the column names and the categories gathered from these columns is generated. Each cell is defined by one of the columns and one of the categories.
  - If the columns do not share common values, they are put into separate tables.
  - You can always change the default action by right-clicking on a column and selecting **Combine Tables** or **Separate Tables**. See "Right-Click Menu for Columns".
- To nest columns, create a table with the first column, and then drag the additional columns into the first column.
- In a properly created table, all grouping columns are together, all analysis columns are together, and all statistics are together. Therefore, JMP does not intersperse a statistics keyword within a list of analysis columns. JMP also does not insert an analysis column within a list of grouping columns.
- You can drag columns from the Table panel in the data table onto a Tabulate table instead of using the Tabulate Control Panel.

Note: The Tabulate table is updated when you add data to the open data table, delete rows, and recode the data.

Analysis Columns

Analysis columns in the Tabulate platform are any numeric (continuous) columns for which you want to compute statistics. Tabulate computes statistics on the analysis columns for each category formed from the grouping columns.

Note: All the analysis columns must reside in the same dimension, either in the row table or in the column table.

Grouping Columns

Grouping columns in the Tabulate platform are columns (nominal or ordinal) that you want to use to classify your data into categories of information. They can have character, integer, or even decimal values, but the number of unique values should be limited.

Note the following:

- If grouping columns are nested, Tabulate constructs distinct categories from the hierarchical nesting of the values of the columns. For example, from the grouping columns Sex with values F and M, and the grouping column Marital Status with values Married and Single, Tabulate constructs four distinct categories: F and Married, F and Single, M and Married, M and Single.

- You can specify grouping columns for column tables as well as row tables. Together they generate the categories that define each table cell.
- Tabulate does not include observations with a missing value for one or more grouping columns by default. You can include them by checking the **Include missing for grouping columns** option.
- To specify codes or values that should be treated as missing, use the Missing Value Codes column property. You can include these by checking the **Include missing for grouping columns** option. For more information about Missing Value Codes, see *Using JMP*.

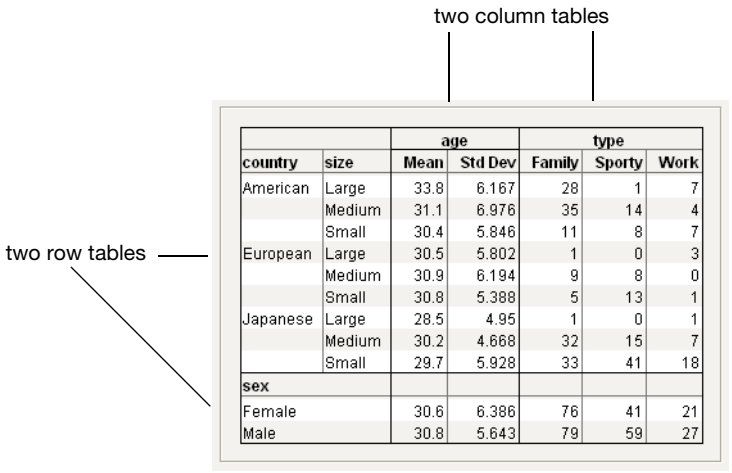
Column and Row Tables

In the Tabulate platform, a table is defined by its column headings and row labels. These sub-tables are referred to as *row tables* and *column tables* (Figure 9.11).

Example of Row and Column Tables in the Tabulate Platform

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Tabulate**.
3. Drag size into the Drop zone for rows.
4. Drag country to the left of the size heading.
5. Drag Mean over the N heading.
6. Drag Std Dev below the Mean heading.
7. Drag age above the Mean heading.
8. Drag type to the far right of the table.
9. Drag sex under the table.

Figure 9.11 Row and Column Tables



		age		type		
country	size	Mean	Std Dev	Family	Sporty	Work
American	Large	33.8	6.167	28	1	7
	Medium	31.1	6.976	35	14	4
	Small	30.4	5.846	11	8	7
European	Large	30.5	5.802	1	0	3
	Medium	30.9	6.194	9	8	0
	Small	30.8	5.388	5	13	1
Japanese	Large	28.5	4.95	1	0	1
	Medium	30.2	4.668	32	15	7
	Small	29.7	5.928	33	41	18
sex						
Female		30.6	6.386	76	41	21
Male		30.8	5.643	79	59	27

For multiple column tables, the labels on the side are shared across the column tables. In this instance, country and sex are shared across the tables. Similarly, for multiple row tables, the headings on the top are shared among the row tables. In this instance, both age and type are shared among the tables.

Edit Tables

There are several ways to edit items that you add to a table. You can delete items, remove column labels, or edit statistical key words or labels.

Delete Items

After you add items to the table, you can remove them in any one of the following ways:

- Drag the item away from the table.
- To remove the last item, click **Undo**.
- Right-click an item and select **Delete**.

Remove Column Labels

Grouping columns display the column name on top of the categories associated with that column. For some columns, the column name might seem redundant. Remove the column name from the column table by right-clicking on the column name and selecting **Remove Column Label**. To re-insert the column label, right-click one of its associated categories and select **Restore Column Label**.

Edit Statistical Key Words and Labels

You can edit a statistical key word or a statistical label. For example, instead of Mean, you might want to use the word Average. Right-click the word that you want to edit and select **Change Item Label**. In the box that appears, enter the new label. Alternatively, you can type directly into the edit box.

If you change one statistics keyword to another statistics keyword, JMP assumes that you actually want to change the statistics, not just the label. It would be as if you have deleted the statistics from the table and added the latter.

Tabulate Platform Options

The following options are available from the red triangle menu next to Tabulate:

Show Control Panel Displays the control panel for further interaction.

Show Table Displays the summarized data in tabular form.

Show Chart Displays the summarized data in bar charts that mirrors the table of summary statistics. The simple bar chart enables visual comparison of the relative magnitude of the summary statistics. By default, all columns of bars share the same scale. You can have each column of bars use the scale determined separately from the data in each displayed column, by clearing the **Uniform plot scale** check box. You can specify a uniform custom scale using the **Change Plot Scale** button. The charts are either 0-based or centered on 0. If the data are all nonnegative, or all non-positive, the charts baseline is at 0. Otherwise, the charts are centered on 0.

Show Shading Displays gray shading boxes in the table when there are multiple rows.

Show Tooltip Displays tips that appear when you hover over areas of the table.

Show Test Build Panel Displays the control area that lets you create a test build using a random sample from the original table. This is particularly useful when you have large amounts of data. See "Show Test Build Panel".

Make Into Data Table Makes a data table from the report. There is one data table for each row table, because labels of different row tables might not be mapped to the same structure.

Full Path Column Name Uses the fully qualified column names of grouping columns for the column name in the created data table.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

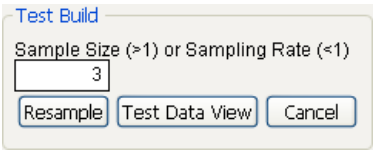
For a description of the options in the Select Columns red triangle menu, see *Using JMP*.

Show Test Build Panel

If you have a very large data table, you might want to use a small subset of the data table to try out different table layouts to find one that best shows the summary information. In this case, JMP generates a random subset of the size as specified and uses that subset when it builds the table. To use the test build feature:

1. Click the Tabulate red triangle and select **Show Test Build Panel**.
2. Enter the size of the sample that you want in the box under **Sample Size (>1) or Sampling Rate (<1)**, as shown in Figure 9.12. The size of the sample can be either the proportion of the active table that you enter or the number of rows from the active table.

Figure 9.12 The Test Build Panel



3. Click **Resample**.
4. To see the sampled data in a JMP data table, click the **Test Data View** button. When you dismiss the test build panel, Tabulate uses the full data table to regenerate the tables as designed.

Right-Click Menu for Columns

Right-click a column in a table to see the following options:

Delete Deletes the selected column.

Use as Grouping column Changes the analysis column to a grouping column.

Use as Analysis column Changes the grouping column to an analysis column.

Change Item Label (Appears only for separate or nested columns.) Enter a new label.

Combine Tables (Columns by Categories) (Appears only for separate or nested columns.) Combines separate or nested columns. See "Example of Combining Columns into a Single Table".

Nest Grouping Columns Nests grouping columns vertically or horizontally.

Separate Tables (Appears only for combined tables.) Creates a separate table for each column.

Remove Column Label Removes the column name from the column table.

Restore Column Label Restores a hidden column name to the column table.

Order By Count Orders the levels of a grouping column by the count of each level, from most to least. This option can also be used to override the setting of the Order by count of grouping columns option for a particular grouping column.

Additional Examples of the Tabulate Platform

- "Example of Creating Different Tables and Rearranging Contents"
- "Example of Combining Columns into a Single Table"
- "Example Using a Page Column"

Example of Creating Different Tables and Rearranging Contents

This example contains the following steps:

1. "Create a Table of Counts"
2. "Create a Table Showing Statistics"
3. "Rearrange the Table Contents"

Create a Table of Counts

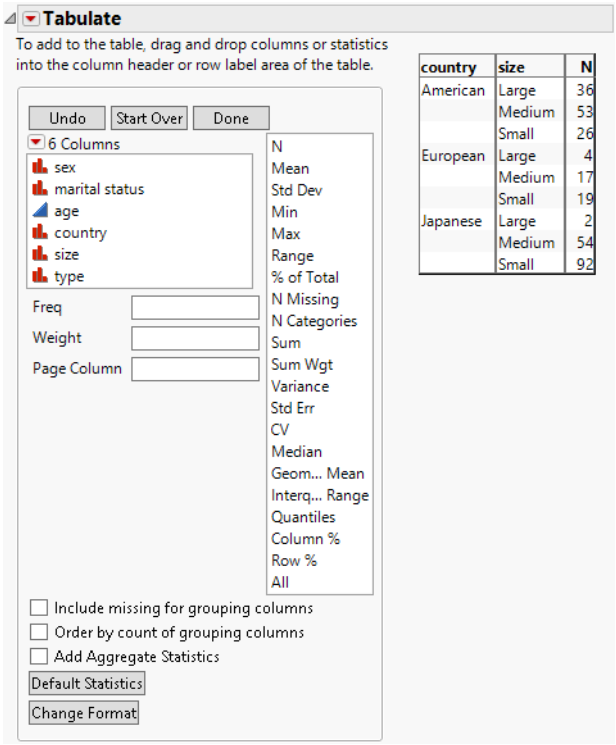
Suppose that you would like to create a table that contains counts for how many people in a survey own Japanese, European, and American cars. You also want the counts broken down by the size of the car.

Figure 9.13 Table Showing Counts of Car Ownership

country	size	N
American	Large	36
	Medium	53
	Small	26
European	Large	4
	Medium	17
	Small	19
Japanese	Large	2
	Medium	54
	Small	92

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Tabulate**.
3. Click country and drag it into the Drop zone for rows.
4. Click size and drag it to the right of the country heading.

Figure 9.14 Country and Size Added to the Table



Create a Table Showing Statistics

Suppose that you would like to see the mean (average) and the standard deviation of the age of people who own each size of car.

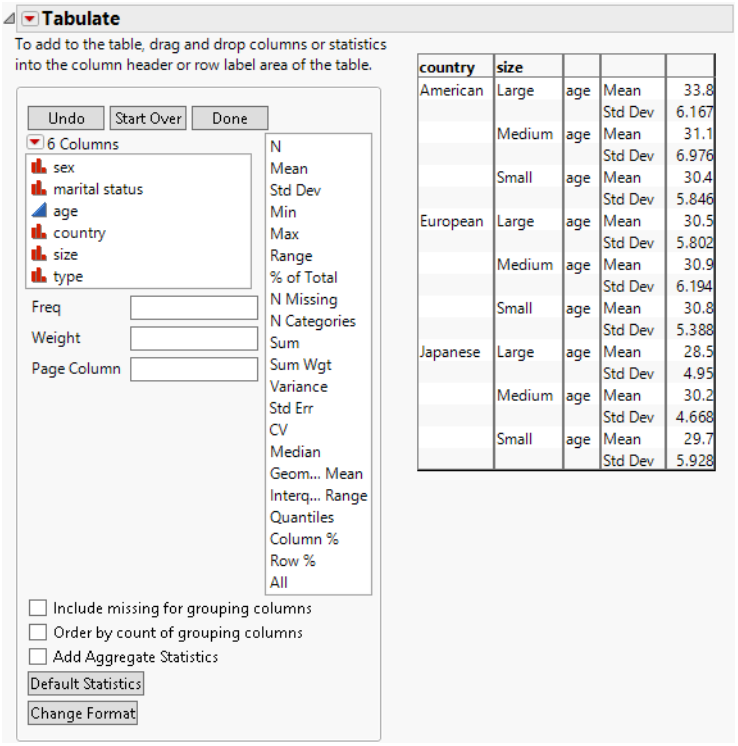
Figure 9.15 Table Showing Mean and Standard Deviation by Age

country	size			
American	Large	age	Mean	33.8
			Std Dev	6.2
	Medium	age	Mean	31.1
			Std Dev	7.0
	Small	age	Mean	30.4
			Std Dev	5.8
European	Large	age	Mean	30.5
			Std Dev	5.8
	Medium	age	Mean	30.9
			Std Dev	6.2
	Small	age	Mean	30.8
			Std Dev	5.4
Japanese	Large	age	Mean	28.5
			Std Dev	4.9
	Medium	age	Mean	30.2
			Std Dev	4.7
	Small	age	Mean	29.7
			Std Dev	5.9

1. Start from Figure 9.14. Click age and drag it to the right of the size heading.
2. Click Mean and drag it over Sum.
3. Click Std Dev and drag it below Mean.

   Std Dev is placed below Mean in the table. Dropping Std Dev above Mean places Std Dev above Mean in the table.

Figure 9.16 Age, Mean, and Std Dev Added to the Table



country	size			
American	Large	age	Mean	33.8
			Std Dev	6.167
	Medium	age	Mean	31.1
			Std Dev	6.976
	Small	age	Mean	30.4
			Std Dev	5.846
European	Large	age	Mean	30.5
			Std Dev	5.802
	Medium	age	Mean	30.9
			Std Dev	6.194
	Small	age	Mean	30.8
			Std Dev	5.388
Japanese	Large	age	Mean	28.5
			Std Dev	4.95
	Medium	age	Mean	30.2
			Std Dev	4.668
	Small	age	Mean	29.7
			Std Dev	5.928

Rearrange the Table Contents

Suppose that you would prefer size to be on top, showing a crosstab layout.

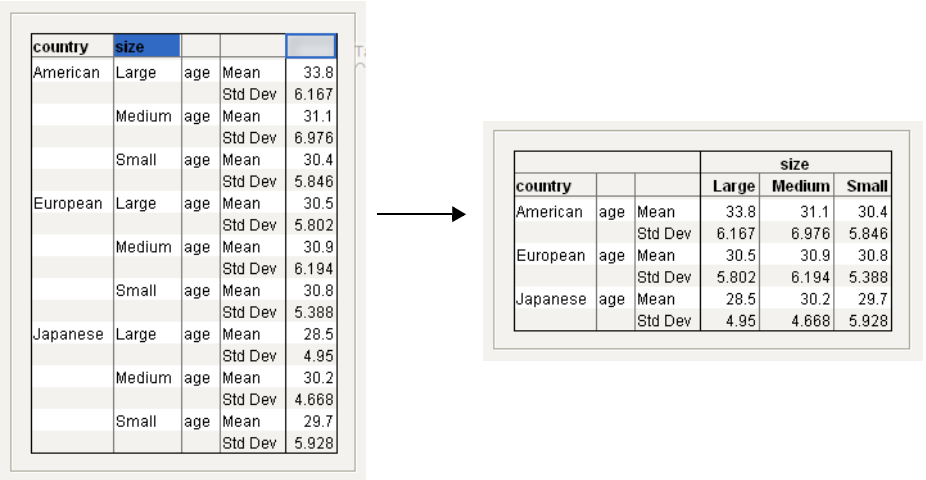
Figure 9.17 Size on Top

	size					
	Large		Medium		Small	
	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
country	age	age	age	age	age	age
American	33.8	6.167	31.1	6.976	30.4	5.846
European	30.5	5.802	30.9	6.194	30.8	5.388
Japanese	28.5	4.95	30.2	4.668	29.7	5.928

Rearrange the table contents.

1. Start from Figure 9.16. Click the size heading and drag it to the right of the table headings.

Figure 9.18 Moving size



country	size			
American	Large	age	Mean	33.8
			Std Dev	6.167
	Medium	age	Mean	31.1
			Std Dev	6.976
	Small	age	Mean	30.4
			Std Dev	5.846
European	Large	age	Mean	30.5
			Std Dev	5.802
	Medium	age	Mean	30.9
			Std Dev	6.194
	Small	age	Mean	30.8
			Std Dev	5.388
Japanese	Large	age	Mean	28.5
			Std Dev	4.95
	Medium	age	Mean	30.2
			Std Dev	4.668
	Small	age	Mean	29.7
			Std Dev	5.928

			size		
country			Large	Medium	Small
American	age	Mean	33.8	31.1	30.4
		Std Dev	6.167	6.976	5.846
European	age	Mean	30.5	30.9	30.8
		Std Dev	5.802	6.194	5.388
Japanese	age	Mean	28.5	30.2	29.7
		Std Dev	4.95	4.668	5.928

2. Click age and drag it under the Large Medium Small heading.
3. Select both Mean and Std Dev, and then drag them under the Large heading.

Now your table clearly presents the data. It is easier to see the mean and standard deviation of the car owner age broken down by car size and country.

Example of Combining Columns into a Single Table

In this example, you have data from students indicating the importance of self-reported factors in children's popularity (grades, sports, looks, money). Suppose that you want to see all of these factors in a single, combined table with additional statistics and factors.

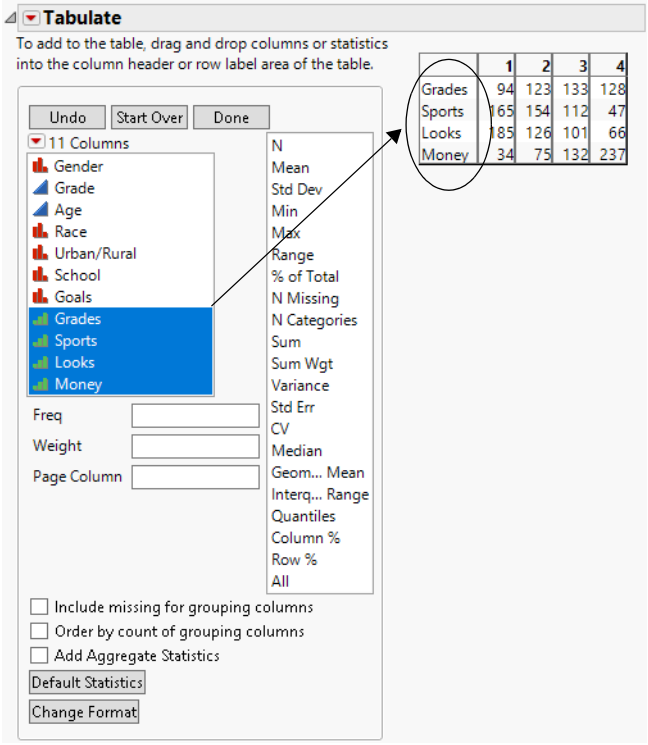
Figure 9.19 Adding Demographic Data

		% of Total														
		Urban/Rural														
		Rural					Suburban					Urban				
Gender		1	2	3	4	All	1	2	3	4	All	1	2	3	4	All
boy	Grades	2.30%	2.93%	3.56%	5.02%	13.81%	2.72%	5.86%	5.02%	5.02%	18.62%	3.14%	3.97%	5.44%	2.51%	15.06%
	Sports	6.49%	3.97%	2.72%	0.63%	13.81%	11.30%	4.60%	2.30%	0.42%	18.62%	8.79%	3.97%	1.46%	0.84%	15.06%
	Looks	3.14%	4.60%	2.72%	3.35%	13.81%	3.77%	5.86%	5.44%	3.56%	18.62%	2.30%	5.02%	4.18%	3.56%	15.06%
	Money	1.88%	2.30%	4.81%	4.81%	13.81%	0.84%	2.30%	5.86%	9.62%	18.62%	0.84%	2.09%	3.97%	8.16%	15.06%
girl	Grades	4.39%	4.39%	4.39%	4.18%	17.36%	2.51%	3.35%	2.93%	4.18%	12.97%	4.60%	5.23%	6.49%	5.86%	22.18%
	Sports	2.30%	6.69%	5.44%	2.93%	17.36%	1.67%	3.56%	5.65%	2.09%	12.97%	3.97%	9.41%	5.86%	2.93%	22.18%
	Looks	9.62%	3.35%	3.35%	1.05%	17.36%	7.95%	2.93%	1.26%	0.84%	12.97%	11.92%	4.60%	4.18%	1.46%	22.18%
	Money	1.05%	2.93%	4.18%	9.21%	17.36%	0.84%	3.14%	3.14%	5.86%	12.97%	1.67%	2.93%	5.65%	11.92%	22.18%

1. Select **Help > Sample Data Library** and open Children's Popularity.jmp.
2. Select **Analyze > Tabulate**.

3. Select Grades, Sports, Looks, and Money and drag them into the Drop zone for rows.

Figure 9.20 Columns by Categories



Notice that a single, combined table appears.

Tabulate the percentage of the one to four ratings of each category.

4. Drag Gender into the empty heading at left.
5. Drag % of Total above the numbered headings.
6. Drag All beside the number 4.

Figure 9.21 Gender, % of Total, and All Added to the Table

		% of Total				
Gender		**1**	**2**	**3**	**4**	**All**
boy	Grades	8.16%	12.76%	14.02%	12.55%	47.49%
	Sports	26.57%	12.55%	6.49%	1.88%	47.49%
	Looks	9.21%	15.48%	12.34%	10.46%	47.49%
	Money	3.56%	6.69%	14.64%	22.59%	47.49%
girl	Grades	11.51%	12.97%	13.81%	14.23%	52.51%
	Sports	7.95%	19.67%	16.95%	7.95%	52.51%
	Looks	29.50%	10.88%	8.79%	3.35%	52.51%
	Money	3.56%	9.00%	12.97%	26.99%	52.51%

Break down the tabulation further by adding demographic data.

7. Drag Urban/Rural below the % of Total heading.

Figure 9.22 Urban/Rural Added to the Table

		% of Total														
		Urban/Rural														
		Rural					Suburban					Urban				
Gender		1	2	3	4	All	1	2	3	4	All	1	2	3	4	All
boy	Grades	2.30%	2.93%	3.56%	5.02%	13.81%	2.72%	5.86%	5.02%	5.02%	18.62%	3.14%	3.97%	5.44%	2.51%	15.06%
	Sports	6.49%	3.97%	2.72%	0.63%	13.81%	11.30%	4.60%	2.30%	0.42%	18.62%	8.79%	3.97%	1.46%	0.84%	15.06%
	Looks	3.14%	4.60%	2.72%	3.35%	13.81%	3.77%	5.86%	5.44%	3.56%	18.62%	2.30%	5.02%	4.18%	3.56%	15.06%
	Money	1.88%	2.30%	4.81%	4.81%	13.81%	0.84%	2.30%	5.86%	9.62%	18.62%	0.84%	2.09%	3.97%	8.16%	15.06%
girl	Grades	4.39%	4.39%	4.39%	4.18%	17.36%	2.51%	3.35%	2.93%	4.18%	12.97%	4.60%	5.23%	6.49%	5.86%	22.18%
	Sports	2.30%	6.69%	5.44%	2.93%	17.36%	1.67%	3.56%	5.65%	2.09%	12.97%	3.97%	9.41%	5.86%	2.93%	22.18%
	Looks	9.62%	3.35%	3.35%	1.05%	17.36%	7.95%	2.93%	1.26%	0.84%	12.97%	11.92%	4.60%	4.18%	1.46%	22.18%
	Money	1.05%	2.93%	4.18%	9.21%	17.36%	0.84%	3.14%	3.14%	5.86%	12.97%	1.67%	2.93%	5.65%	11.92%	22.18%

You can see that for boys in rural, suburban, and urban areas, sports are the most important factor for popularity. For girls in rural, suburban, and urban areas, looks are the most important factor for popularity.

Example Using a Page Column

In this example, you have data containing height measurements for male and female students. You want to create a table that shows the mean of the heights by the age of the students. Then you want to stratify your data by sex in different tables. To do so, add the stratification column as a page column, which builds the pages for each group.

Figure 9.23 Mean Height of Students by Sex

sex = F

	height
age	Mean
12	58.6
13	59.0
14	62.6
15	63.0
16	62.5
17	62.0

Females

sex = M

	height
age	Mean
12	57.3
13	61.3
14	65.3
15	65.2
16	68.0
17	69.0

Males

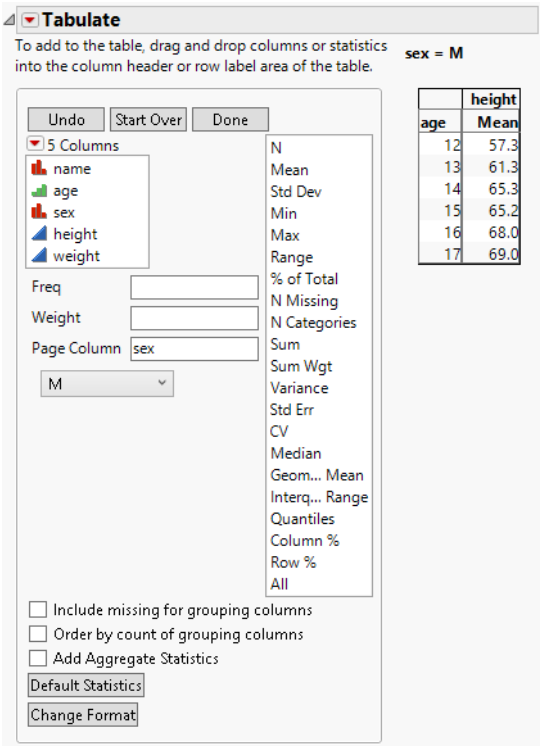
1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Tabulate**.

   Since height is the variable that you are examining, you want it to appear at the top of the table.
3. Click height and drag it into the Drop zone for columns.

You want the statistics by age, and you want age to appear on the side.

4. Click age and drag it into the blank cell next to the number 2502.
5. Click Mean and drag it into the cell that says Sum.
6. Click sex and drag it into **Page Column**.
7. Select F from the Page Column list to show the mean of the heights for only females.
8. Select M from the Page Column list to show the mean of the heights for only males. You can also select **None Selected** to show all values.

Figure 9.24 Using a Page Column



Chapter 10

JMP PRO Simulate

Answer Challenging Questions with Parametric Resampling

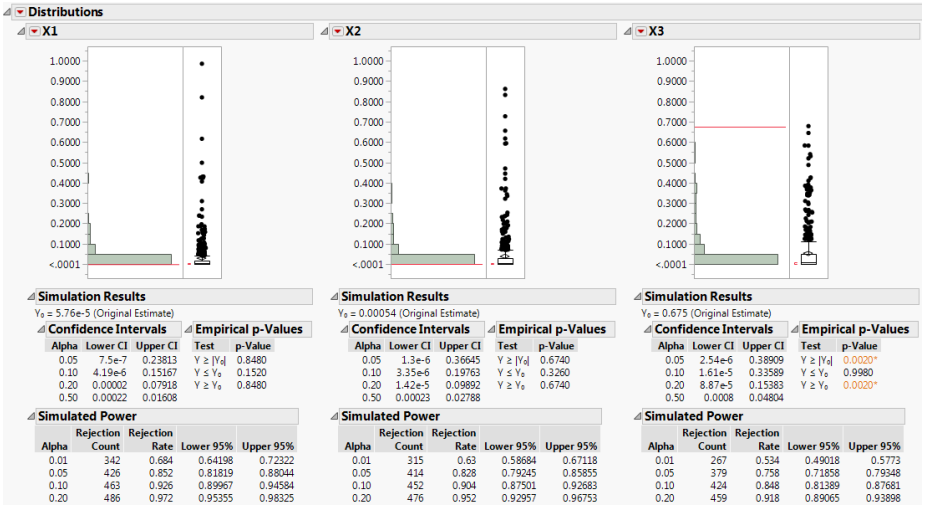
The Simulate platform is available only in JMP Pro.

The Simulate feature provides powerful parametric and nonparametric simulation capability. Use Simulate to do the following:

- Expand on the bootstrap to provide parametric bootstrapping.
- Obtain power calculations in nonstandard situations.
- Approximate the distribution of statistics, such as predicted values, and confidence intervals, in nonstandard situations.
- Conduct permutation tests.
- Explore the effect of assumptions about predictors on models.
- Explore various “what if” scenarios relative to your models.
- Evaluate new or existing statistical methods.

The Simulate option is available in many reports, including all of those that support Bootstrap. To access the Simulate option, right-click in a report.

Figure 10.1 Power Analysis Using Simulate



Contents

Overview of the Simulate Platform 331
Examples That Use Simulate 332
Construct Semiparametric Confidence Intervals for Variance Components 332
Conduct a Permutation Test 338
Explore Retaining a Factor in Generalized Regression 340
Conduct Prospective Power Analysis for a Nonlinear Model 345
Launch the Simulate Window 354
The Simulate Window 354
The Simulate Results Table 355
Simulation Results Report 356
Simulated Power Report 356

JMP PRO Overview of the Simulate Platform

The Simulate platform provides simulated results for a column of statistics in a report. Right-click a column of statistics in a report and select Simulate. In the Simulate window, specify a column in your data table that forms the basis for your simulation. This is the column that you *switch out*. This column can have any role in the analysis. In particular, it can be a response or a predictor in a model. You then specify a column in your data table that contains a formula that you want to use for the simulation. This is the column that you *switch in*. It functions as a surrogate for the column that you switched out.

Note: Your data table must contain a column that has a random component.

The method works as follows. A column of simulated values is generated based on the formula in the formula column that you switch in. The entire analysis that generated the report containing the statistics of interest is rerun using this new column of simulated values to replace the column that you switched out. This process is repeated N times, where N is the total number of samples that you specify.

The Simulate analysis produces an output data table showing a summary of the analysis.

- Each row of the data table represents the results of the analysis for one column of simulated values.
- There is a column for each row of the report table involved in the simulation.
- There are scripts to facilitate your analysis.

Tip: The Simulate platform reruns the entire analysis that appears in the platform report from which Simulate is invoked. As a result, Simulate might run slowly for your selected column because of extraneous analyses in the report. If Simulate is taking a long time, remove extraneous options from the platform report before running Simulate.

Simulate is available in all statistical platforms except the following: Association Analysis, Diagram, Multidimensional Scaling, Multiple Factor Analysis, Reliability Block Diagram, Reliability Forecast, Repairable Systems Simulation, Response Screening, and Text Explorer.

JMP PRO Examples That Use Simulate

Learn how to get simulated results for your data in different scenarios.

- "Construct Semiparametric Confidence Intervals for Variance Components"
- "Conduct a Permutation Test"
- "Explore Retaining a Factor in Generalized Regression"
- "Conduct Prospective Power Analysis for a Nonlinear Model"
- For an example that shows how to simulate a confidence interval for Ppk and the percent nonconforming for a non-normal variable, see *Quality and Process Methods*.

JMP PRO Construct Semiparametric Confidence Intervals for Variance Components

In this example, your goal is to obtain semiparametric confidence intervals for the variance components of hard-to-change variables. For this data, you are interested in the effects of temperature, time, and the amount of catalyst on a reaction. Temperature is a very-hard-to-change variable (whole plot factor), time is hard-to-change (subplot factor), and the amount of catalyst is easy-to-change. For information about whole plot and subplot factors, see the *Design of Experiments Guide*.

Previous studies have suggested that the whole-plot standard deviation is about twice the error standard deviation, and the sub-plot error is about 1.5 times the error standard deviation. The Wald intervals given in the REML report, which assume that the variance components are asymptotically normal, have poor coverage properties. You obtain confidence intervals using percentiles of the simulated distributions of the variance components.

JMP PRO Construct the Design

In this section, you construct a custom design for your split-split-plot experiment.

Tip: If you prefer to skip the steps in this section, select **Help > Sample Data Library** and open Design Experiment/Catalyst Design.jmp. In the Catalyst Design.jmp data table, click the green triangle next to the **DOE Simulate** script. Then go to "Fit the Model".

1. Select **DOE > Custom Design**.
2. In the Factors outline, type 3 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Double-click to rename these factors Temperature, Time, and Catalyst.

Keep the default Values of –1 and 1 for these factors.

5. For Temperature, click **Easy** and select **Very Hard**.

   This defines Temperature to be a whole plot factor.

6. For Time, click **Easy** and select **Hard** for Time.

   This defines Time to be a subplot factor.

7. Click **Continue**.
8. In the Model outline, select **Interactions > 2nd**.

   This adds all two-way interactions to the model.

9. Click the Custom Design red triangle and select **Simulate Responses**.

   This opens the Simulate Responses window after you select Make Table to construct the design table.

   Note: Setting the Random Seed in step 10 and Number of Starts in step 11 reproduces the same design shown in this example. In constructing a design on your own, these steps are not necessary.

10. (Optional) Click the Custom Design red triangle and select **Set Random Seed**. Type 12345 and click **OK**.
11. (Optional) Click the Custom Design red triangle and select **Number of Starts**. Type 1000 and click **OK**.
12. Click **Make Design**.
13. Click **Make Table**.

    Note: The entries in your Y and Y Simulated columns might differ from those that appear in Figure 10.2.

Figure 10.2 Design Table

Catalyst Design
Locked File C:\Program File
Design Custom Design
Criterion D Optimal
Model
Model for Y Simulated
Evaluate Design
DOE Simulate
DOE Dialog

Columns (7/0)
Whole Plots ✱
Subplots ✱
Temperature ✱
Time ✱
Catalyst ✱
Y ✱
Y Simulated

Rows
All rows 24
Selected 0
Excluded 0

	Whole Plots	Subplots	Temperature	Time	Catalyst	Y	Y Simulated
1	1	1	1	1	-1	-0.046768023	2.38362212
2	1	1	1	1	1	6.539690219	6.934627
3	1	1	1	1	1	6.5397108257	6.6911769
4	1	2	1	-1	1	0.494801276	-1.5879342
5	1	2	1	-1	-1	-2.65841644	-3.2406153
6	1	2	1	-1	-1	-0.200740508	-2.5118393
7	2	3	-1	1	1	3.1335699565	2.10432757
8	2	3	-1	1	1	2.2739679213	2.36399771
9	2	3	-1	1	-1	1.1704395547	0.23356541
10	2	4	-1	-1	-1	-0.693941035	1.29953351
11	2	4	-1	-1	-1	2.3525055616	2.18979623
12	2	4	-1	-1	1	-1.545149968	-0.0606938
13	3	5	-1	1	-1	-1.4838427	-1.08124
14	3	5	-1	1	-1	-1.375362439	-2.0077657
15	3	5	-1	1	1	0.7434199911	-0.3799921
16	3	6	-1	-1	1	-1.11989643	-3.4434001
17	3	6	-1	-1	-1	3.4346315881	1.28011609
18	3	6	-1	-1	1	0.2987179478	-2.752904
19	4	7	1	-1	1	-1.245681444	-1.7403239
20	4	7	1	-1	1	0.4951842993	0.74829587
21	4	7	1	-1	-1	-0.6602484	-1.3436848
22	4	8	1	1	-1	0.7779088461	1.23261462
23	4	8	1	1	1	7.9432230978	5.9851515
24	4	8	1	1	-1	1.0668195454	1.35743939

Figure 10.3 Simulate Responses Window

Simulate Responses

Effects	Y
Intercept	1
Temperature	1
Time	1
Catalyst	1
Temperature*Time	1
Temperature*Catalyst	1
Time*Catalyst	1

Reset coefficients

Distribution

Normal
Error σ: 1
Whole Plots σ: 1
Subplots σ: 1
Binomial
Poisson

Apply

The design table and a Simulate Responses window appear. Notice that the design table contains a **DOE Simulate** script. At any time, you can run this script to specify different parameter values.

Continue to the next section, where you specify standard deviations for the whole plot and subplot errors, and fit a REML model to the first set of simulated values.

JMP PRO Fit the Model

In this section, you fit a model using the REML method. Assume that the whole plot and subplot errors are normal. Based on your estimates of their standard deviations, if the error standard deviation is about 1 unit, the whole plot standard deviation is about 2 units and the subplot standard deviation is about 1.5 units. Since you are interested only in the whole- and sub-plot variation, you do not need to change the values assigned to Effects in the Simulate Responses outline.

1. In the Distribution panel (Figure 10.3), next to **Whole Plots** σ, type 2.

   Notice that the Normal distribution is selected by default. As a result, normal error is added to the formula.
2. Next to **Subplots** σ, type 1.5.
3. Click **Apply**.

   In the data table, the formula for Y Simulated updates to reflect your specifications. To view the formula, click the plus sign to the right of the column name in the Columns panel.
4. In the data table, click the green triangle next to the **Model** script.
5. Click the Y variable next to the **Y** button and click **Remove**.
6. Click Y Simulated and click the **Y** button.

   This action replaces Y with a column that contains a simulation formula.
7. Click **Run**.

   The model that is fit is based on a single set of simulated responses.

Note: Because the values in Y Simulated are randomly generated, the entries in your report might differ from those that appear in Figure 10.4.

Figure 10.4 REML Report Showing Wald Confidence Intervals

REML Variance Component Estimates

Random Effect	Var Ratio	Var Component	Std Error	95% Lower	95% Upper	Wald p-Value	Pct of Total
Whole Plots	3.8107856	1.556364	1.8007472	-1.973036	5.0857636	0.3874	68.445
Subplots	0.7568606	0.3091097	0.4661521	-0.604532	1.222751	0.5073	13.594
Residual		0.4084103	0.160228	0.2146172	1.060295		17.961
Total		2.2738839	1.8035082	0.7468087	28.115033		100.000

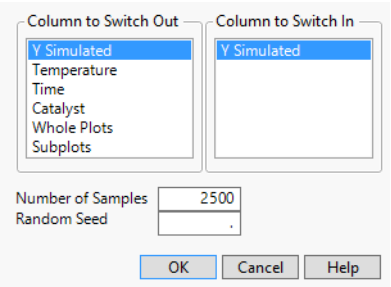
-2 LogLikelihood = 63.890021519
Note: Total is the sum of the positive variance components.
Total including negative estimates = 2.2738839

JMP PRO Generate Confidence Intervals

In this section, you simulate variance component estimates and use these to construct simulated percentile confidence intervals.

1. In the REML Variance Components Estimates outline, right-click in the **Var Component** column and select **Simulate**.

Figure 10.5 Simulate Window



In your simulations, you replace the column Y Simulated, which you used to run your model, with a new instance of the column Y Simulated, which generates a new column of simulated values for each simulation. The column on which you right-clicked and that appears as selected, **Var Component**, is simulated for each effect listed in the Parameter Estimates table.

2. Next to **Number of Samples**, enter **200**.
3. (Optional) Next to **Random Seed**, enter 456.

   This reproduces the values shown in Figure 10.6, except for the values in row 1.
4. Click **OK**.

   The entries in your row 1 might differ from those that appear in Figure 10.6.

Figure 10.6 Table of Simulated Results for Var Component (Partial View)

Fit Least Squares Simulat...
Random Seed 456
Make Combined Data Table
Distribution

Columns (6/0)
Y *
SimID•
Residual
Subplots
Total
Whole Plots

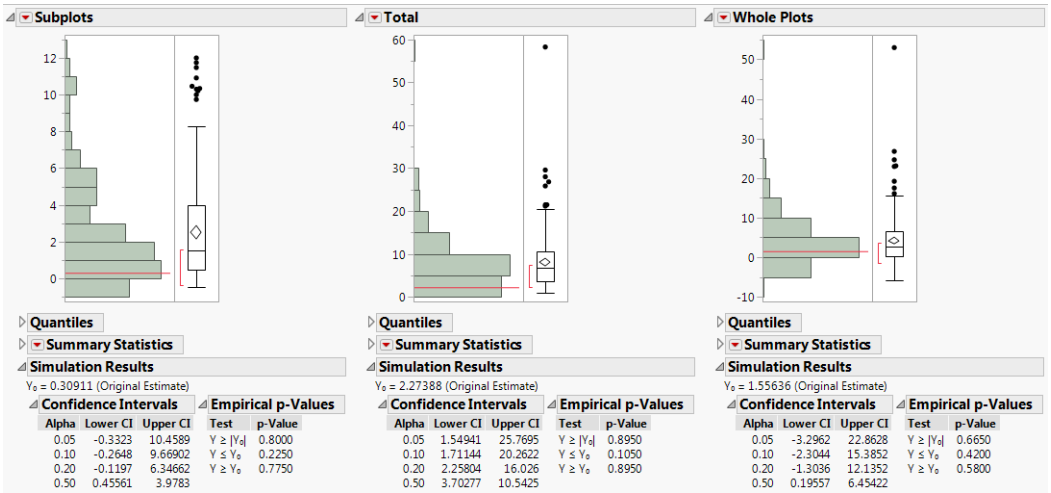
Rows
All rows 201
Selected 0
Excluded 1
Hidden 0
Labeled 0

		Y	SimID•	Residual	Subplots	Total	Whole Plots
×	1	Y Simulated	0	0.4084102799	0.3091096508	2.2738839302	1.5563639995
	2	Y Simulated	1	1.7118437836	9.9978320675	11.825684937	0.1160090855
	3	Y Simulated	2	0.7259348275	4.5617762065	8.6529959081	3.3652848741
	4	Y Simulated	3	0.8416572537	3.0383907272	26.836360491	22.95631251
	5	Y Simulated	4	0.9872034755	-0.176636748	13.143778421	12.156574946
	6	Y Simulated	5	1.3514412755	-0.153702674	1.3514412755	-0.060658009
	7	Y Simulated	6	0.9837455205	1.1063669307	13.894478258	11.804365807
	8	Y Simulated	7	0.8559296173	0.1630909822	3.7015003812	2.6824797817
	9	Y Simulated	8	1.0422128078	0.2154635485	28.022776455	26.765100099
	10	Y Simulated	9	1.0302645404	4.0615625437	17.850598404	12.75877132
	11	Y Simulated	10	1.2569403028	1.8169128804	3.0738531832	-0.433818317
	12	Y Simulated	11	1.6719518802	6.9696566519	11.105096973	2.4634884406
	13	Y Simulated	12	1.0326091926	1.9758939136	7.2251377019	4.2166345958
	14	Y Simulated	13	0.7050109246	1.1584152266	4.6661579836	2.8027318323
	15	Y Simulated	14	0.9281404099	5.2888052789	6.53640214	0.3194564512
	16	Y Simulated	15	1.6586718874	1.4827905134	3.1414624009	-0.270350632
	17	Y Simulated	16	0.9471174937	0.7636963822	1.7108138759	-0.192580268
	18	Y Simulated	17	0.9782751744	1.1919618966	2.927052512	0.756815441
	19	Y Simulated	18	1.5524955876	1.0391874862	7.4052160139	4.8135329401

The first row of the Fit Least Squares Simulate Results (Var Component) data table contains the initial values of **Var Component** and is excluded. The remaining rows contain simulated values.

5. Run the **Distribution** script.

Figure 10.7 Distribution Plots for Variance Components (Partial View)



For each variance component, confidence intervals at various confidence levels are shown in the Simulation Results report. Compare the 95% intervals in the Alpha=0.05 row of each table to the intervals given in the REML report (Figure 10.4):

- The simulated 95% confidence interval for the whole-plot variance component is -3.296 to 22.863. The Wald interval given in the REML report is -1.973 to 5.086.
- The simulated 95% confidence interval for the sub-plot variance component is -0.332 to 10.459. The Wald interval given in the REML report is -0.605 to 1.223.

The intervals that you obtain using simulation are considerably wider than the REML interval calculated from your single set of values. For more precise intervals, consider running a larger number of simulations.

JMP PRO Conduct a Permutation Test

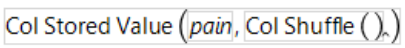
In this example, you use Simulate to conduct a randomization or permutation test. You are studying the effects of three drugs on pain, and are interested in whether they differ in their effects. Because you have a very small sample size and are somewhat concerned about violations of the usual ANOVA assumptions, you can use Simulate to conduct a permutation test.

First, you construct a formula that randomly shuffles the pain measurements among the three drugs. Under the null hypothesis of no effect, any of these allocations is as likely as any other. It follows that the *F* ratios obtained in this manner approximate the distribution of *F* ratios under the null hypothesis. Finally, you compare the observed value of the *F* ratio to the null distribution obtained by simulation.

JMP PRO Define the Simulation Formula

1. Select **Help > Sample Data Library** and open Analgesics.jmp.
2. Select **Cols > New Columns**.
3. Type Pain Shuffled for Column Name.
4. From the Column Properties list, select **Formula**.
5. In the function list, select **Row > Col Stored Value**.
6. In the Columns list, double-click pain.
7. Click the insert key (^) in the list of symbols above the editor panel.
8. From the list of functions, select **Random > Col Shuffle**.

Figure 10.8 Completed Formula



This formula randomly shuffles the entries in the pain column.

9. Click **OK** in the Formula Editor window.
10. Click **OK** in the Column Info window.

Perform the Permutation Test

1. Select **Analyze > Fit Y by X**.
2. Select pain and click **Y, Response**.
3. Select drug and click **X, Factor**.
4. Click **OK**.
5. Click the Oneway Analysis red triangle and select **Means/Anova**.

Figure 10.9 Analysis of Variance Report

Analysis of Variance

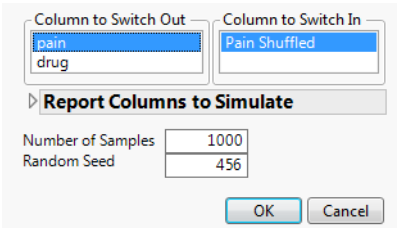
Source	DF	Sum of Squares	Mean Square	F Ratio	Prob > F
drug	2	99.89459	49.9473	6.2780	0.0053*
Error	30	238.67877	7.9560		
C. Total	32	338.57335			

Notice that the *F* ratio is 6.2780.

6. In the Analysis of Variance outline, right-click the F Ratio column and select **Simulate**.
7. In the Column to Switch Out list, click pain.
8. In the Column to Switch In list, click Pain Shuffled.
9. Next to **Number of Samples**, enter 1000.
10. (Optional) Next to **Random Seed**, enter 456.

    This reproduces the values in this example.

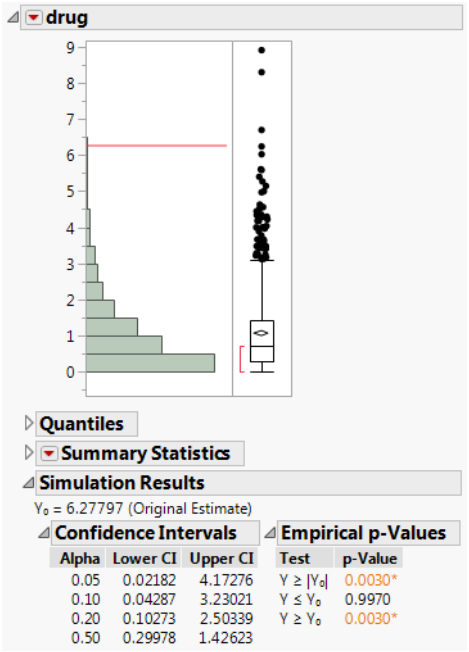
Figure 10.10 Completed Simulate Window



11. Click **OK**.

    In the table of simulated results, the C. Total and Error columns are empty, since the *F* Ratio value in the Analysis of Variance table applies only to drug.
12. In the table of simulated values, run the **Distribution** script.

Figure 10.11 Simulated Distribution of *F* Ratios under the Null Distribution



The observed *F* ratio value of 6.2780 is represented with a red line in the histogram. This value falls in the upper 0.5% of the simulated null distribution of *F* ratios. This presents strong evidence that the three drugs differ in their effects on pain.

JMP PRO Explore Retaining a Factor in Generalized Regression

In this example, you construct a generalized regression model and use the active parameters to fit a reduced model. Based on this reduced model, you use simulation to explore the likelihood that one of the factors is included in the model.

A pharmaceutical manufacturer has historical information about the dissolution of a tablet inside the body and various factors that can affect the dissolution rate. A tablet with a dissolution rate below 70 is considered defective. You want to understand which factors affect dissolution rate.

JMP PRO Fit the Model

In this section, you fit a model using generalized regression.

Tip: If you prefer not to work through the steps in this section, click the green triangle next to the **Generalized Regression** script in the Tablet Production.jmp data table to obtain the model.

1. Select **Help > Sample Data Library** and open Tablet Production.jmp.
2. Select **Analyze > Fit Model**.
3. Click Dissolution and click **Y**.
4. Select Mill Time through Atomizer Pressure and click **Add**.
5. From the Personality list, select **Generalized Regression**.
6. Click **Run**.
7. In the Model Launch panel, select the **Adaptive** box.
8. In the Model Launch panel, click **Go**.

Figure 10.12 Model Based on Adaptive Lasso

Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	108.6406	24.332975	19.933977	<.0001*	60.948851	156.33236
Mill Time	0.130278	0.028185	21.365139	<.0001*	0.0750364	0.1855195
Screen Size[3-5]	4.1877616	0.541493	59.8106	<.0001*	3.1264548	5.2490685
Screen Size[4-5]	2.3907729	0.567217	17.765534	<.0001*	1.2790481	3.5024977
Mag. Stearate Supplier[Jones Inc-Smith Ind]	0	0	0	1.0000	0	0
Lactose Supplier[Bond Inc-James Ind]	0	0	0	1.0000	0	0
Sugar Supplier[Sour-Sweet]	0	0	0	1.0000	0	0
Talc Supplier[Rough-Smooth]	0	0	0	1.0000	0	0
Blend Time	0.7223423	0.1381848	27.325338	<.0001*	0.4515051	0.9931796
Blend Speed	0.2130888	0.2389205	0.7954524	0.3725	-0.255187	0.6813644
Compressor[Compress1-Compress2]	-0.538112	0.3993673	1.8155148	0.1778	-1.320857	0.2446338
Force	0	0	0	1.0000	0	0
Coating Supplier[Coat-Mac]	0	0	0	1.0000	0	0
Coating Supplier[Down-Mac]	0	0	0	1.0000	0	0
Coating Viscosity	0.1834341	0.0500838	13.414242	0.0002*	0.0852717	0.2815965
Inlet Temp	0	0	0	1.0000	0	0
Exhaust Temp	0	0	0	1.0000	0	0
Spray Rate	-0.204356	0.0420142	23.658267	<.0001*	-0.286702	-0.12201
Atomizer Pressure	0	0	0	1.0000	0	0

Normal Distribution Parameters	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Scale	2.0224259	0.1641142	151.86339	<.0001*	1.700768	2.3440838

You are interested in the parameter estimates shown in the Normal Adaptive Lasso with AICc Validation report. Based on the nonzero parameter estimates, the model suggests that Mill Time, Screen Size, Blend Time, Blend Speed, Compressor, Coating Viscosity, and Spray Rate are related to Dissolution.

JMP PRO Reduce the Model

Before reducing the model, ensure that no columns are selected in the Tablet Production.jmp data table. Selected columns are not deselected in the first step below. Ensuring that no columns are selected prevents the inadvertent inclusion of columns with zeroed terms.

If you prefer not to work through the steps in this section, click the green triangle next to the **Generalized Regression Reduced Model** script in the Tablet Production.jmp data table to obtain the reduced model.

1. Click the red triangle next to Normal Adaptive Lasso with AICc Validation and select **Relaunch Active Set > Relaunch with Active Effects**.

   This opens a Fit Model window that places the terms with nonzero coefficient estimates in the Parameter Estimates reports into the Construct Model Effects list. The response is entered as Y. The Generalized Regression personality is selected.
2. Click **Run**.
3. In the Model Launch panel, select the **Adaptive** box.
4. In the Model Launch panel, click **Go**.

Figure 10.13 Reduced Model Using Adaptive Lasso

Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	95.142391	23.430178	16.489099	<.0001*	49.220085	141.0647
Mill Time	0.1394103	0.0275609	25.585971	<.0001*	0.0853919	0.1934288
Screen Size[3-5]	4.3323833	0.534237	65.763638	<.0001*	3.285298	5.3794685
Screen Size[4-5]	2.6331283	0.5457852	23.275584	<.0001*	1.563409	3.7028476
Blend Time	0.7583048	0.1385246	29.966364	<.0001*	0.4868017	1.029808
Blend Speed	0.4575802	0.2289168	3.9955744	0.0456*	0.0089116	0.9062488
Compressor[Compress1-Compress2]	-0.877986	0.4127286	4.5252866	0.0334*	-1.686919	-0.069053
Coating Viscosity	0.198673	0.0486798	16.656386	<.0001*	0.1032624	0.2940836
Spray Rate	-0.212753	0.0410929	26.805238	<.0001*	-0.293294	-0.132213

Normal Distribution Parameters	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Scale	1.9982636	0.1574951	160.97992	<.0001*	1.689579	2.3069483

Notice that the estimate for Blend Speed has a confidence interval (Lower 95%) that comes very close to including zero. Next, perform a simulation study to see how often Blend Speed would be included in the model if other data values from the dissolution distribution have been observed.

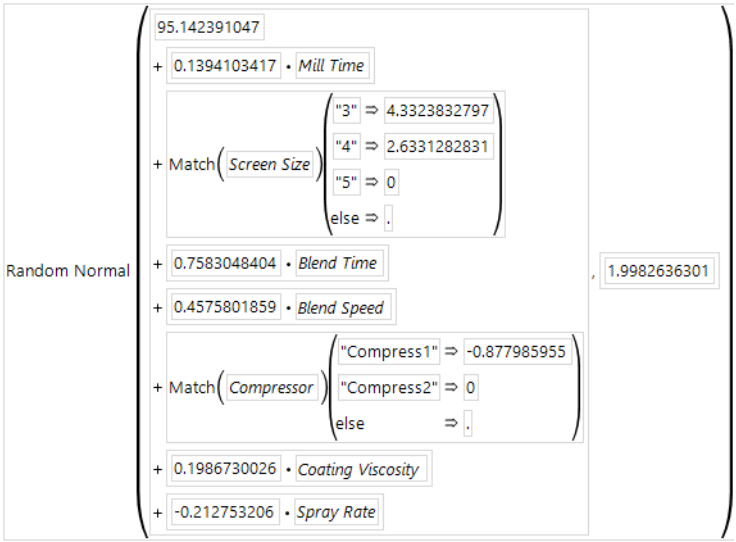
JMP PRO Explore the Inclusion of Blend Speed in the Model

Use the report for the reduced model (Figure 10.13) in the steps below.

1. Click the red triangle next to Normal Adaptive Lasso with AICc Validation and select **Save Columns > Save Simulation Formula**.

   This adds a new column called Dissolution Simulation Formula to the Tablet Production.jmp data table.
2. (Optional) In the data table Columns panel, click the plus sign to the right of Dissolution Simulation Formula.

Figure 10.14 Simulation Formula



For each row, this formula simulates a value that could be obtained given the model and the distribution of Dissolution, which is estimated to be Normal with standard deviation about 1.998.

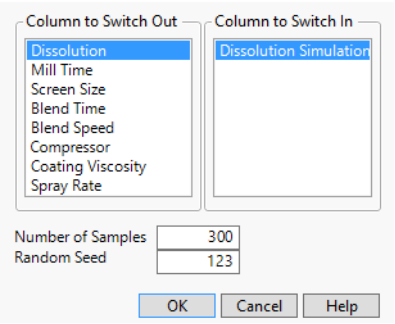
3. Click **Cancel**.
4. Go back to the reduced model report window. In the Parameter Estimates for Original Predictors report, right-click in the Estimate column and select **Simulate**.

   Make sure that Dissolution is selected in the Column to Switch Out list.
5. Next to **Number of Samples**, enter 300.

   For the simulation, you ask JMP to replace the Dissolution column in each of 300 analyses with values simulated using the Dissolution Simulation Formula column.
6. (Optional) Set the **Random Seed** to 123.

   This reproduces the values in this example.

Figure 10.15 Completed Simulation Window

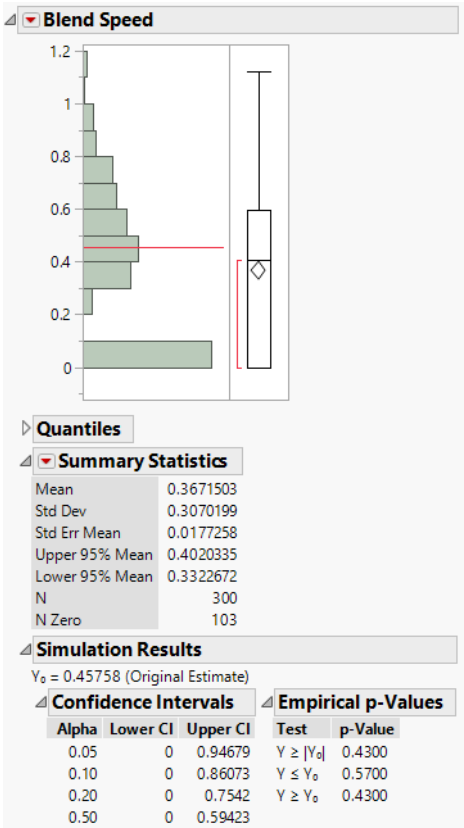


7. Click **OK**.

   The first row of the table contains the initial values of the Estimates and is excluded. The remaining rows contain simulated values.

8. Run the **Distribution** script.
9. Press Ctrl, click the Blend Speed red triangle and select **Display Options > Customize Summary Statistics**.
10. Select **N Zero**.
11. Click **OK**.
12. Scroll to the Distribution report for Blend Speed.

Figure 10.16 Histogram of Simulated Blend Speed Coefficient Estimates



The Summary Statistics report shows that for 103/300 = 34.3% of the simulations, the Blend Speed estimates are zero.

JMP PRO Conduct Prospective Power Analysis for a Nonlinear Model

In this example, you are interested in the main effects of six continuous factors on whether a part passes or fails inspection. The response is binomial and you can afford a total of 60 runs.

Plan for the Example

You model the probability of a success using a generalized linear model with the logit as a link function. The logit link function fits a logistic model:

$$\pi(\mathbf{X}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \ldots + \beta_6 X_6)}}$$

where $\pi(\mathbf{X})$ denotes the probability that a part passes at the given design settings $\mathbf{X} = (X_1, X_2, ..., X_6)$.

Denote the linear predictor by $L(\mathbf{X})$:

$$L(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \ldots + \beta_6 X_6$$

Next, you explore power for the following values of the coefficients of the linear predictor:

Coefficient	Value
β_0	0
β_1	1
β_2	0.9
β_3	0.8
β_4	0.7
β_5	0.6
β_6	0.5

Because the intercept in the linear predictor is 0, when all factors are set to 0, the probability of a passing part equals 50%. The probabilities associated with the levels of the i^{th} factor, when all other factors are set to 0, are given below.

Factor	Percent Passing at $X_i = 1$	Percent Passing at $X_i = -1$	Difference
X_1	73.11%	26.89%	46.2%
X_2	71.09%	28.91%	42.2%
X_3	69.00%	31.00%	38.0%
X_4	66.82%	33.18%	33.6%
X_5	64.56%	35.43%	29.1%
X_6	62.25%	37.75%	24.5%

For example, when all factors other than X_1 are set to 0, the difference in pass rates that you want to detect is 46.2%. The smallest difference in pass rates that you want to detect occurs when all factors other than X_6 are set to zero and that difference is 24.5%.

JMP PRO Construct the Design

In this section, you construct a custom design for your experiment.

Note: If you prefer to skip the steps in this section, select **Help > Sample Data Library** and open Design Experiment/Binomial Experiment.jmp. Click the green triangle next to the **DOE Simulate** script and then go to "Define Simulated Responses".

1. Select **DOE > Custom Design**.

   Note: Although a custom design is not optimal for a non-linear situation, in this example, for simplicity, you can use the Custom Design platform rather than the Nonlinear Design platform. For an example illustrating why a design constructed using the Nonlinear Design platform is better than an orthogonal design, see the *Design of Experiments Guide*.

2. In the Factors outline, type 6 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Click **Continue**.

   You are constructing a main effects design, so do not make any changes to the Model outline.

5. Under Number of Runs, type 60 next to **User Specified**.
6. Click the Custom Design red triangle and select **Simulate Responses**.

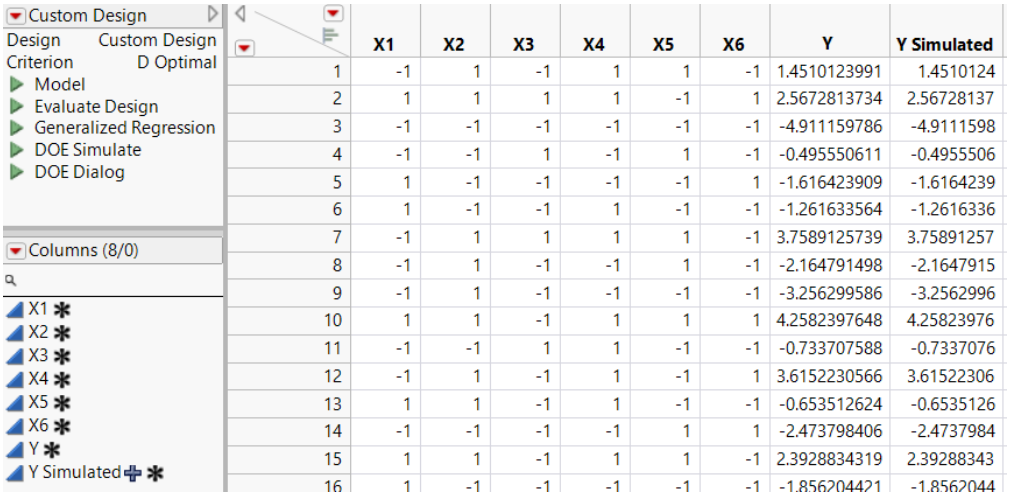
   This opens the Simulate Responses window after you select Make Table to construct the design table.

   Note: Setting the Random Seed in step 7 and Number of Starts in step 8 reproduces the same design shown in this example. In constructing a design on your own, these steps are not necessary.

7. (Optional) Click the Custom Design red triangle and select **Set Random Seed**. Type 12345 and click **OK**.
8. (Optional) Click the Custom Design red triangle and select **Number of Starts**. Type 1 and click **OK**.
9. Click **Make Design**.
10. Click **Make Table**.

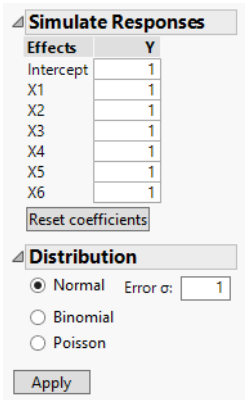
Note: The entries in your Y and Y Simulated columns might differ from those that appear in Figure 10.17.

Figure 10.17 Partial View of Design Table



	X1	X2	X3	X4	X5	X6	Y	Y Simulated
1	-1	1	-1	1	1	-1	1.4510123991	1.4510124
2	1	1	1	1	-1	1	2.5672813734	2.56728137
3	-1	-1	-1	-1	-1	-1	-4.911159786	-4.9111598
4	-1	-1	1	-1	1	-1	-0.495550611	-0.4955506
5	1	-1	-1	-1	-1	1	-1.616423909	-1.6164239
6	1	-1	-1	1	-1	-1	-1.261633564	-1.2616336
7	-1	1	1	1	1	-1	3.7589125739	3.75891257
8	-1	1	-1	-1	1	-1	-2.164791498	-2.1647915
9	-1	1	-1	-1	-1	-1	-3.256299586	-3.2562996
10	1	1	-1	1	1	1	4.2582397648	4.25823976
11	-1	-1	1	1	-1	-1	-0.733707588	-0.7337076
12	-1	1	-1	1	-1	1	3.6152230566	3.61522306
13	1	1	-1	1	-1	-1	-0.653512624	-0.6535126
14	-1	-1	-1	-1	1	1	-2.473798406	-2.4737984
15	1	1	-1	1	1	-1	2.3928834319	2.39288343
16	1	-1	-1	-1	-1	-1	-1.856204421	-1.8562044

Figure 10.18 Simulate Responses Window



The design table and a Simulate Responses window appear. Two columns are added to the design table:

- Y contains a set of values simulated according to the specifications in the Simulate Responses window.
- Y Simulated contains a formula that calculates its values using the formula for the model that is specified in the Simulate Responses window. To view the formula, click the plus sign to the right of the column name in the Columns panel.

Continue to the next section, where you simulate binomial responses and fit a generalized linear model to these simulated responses.

JMP PRO Define Simulated Responses

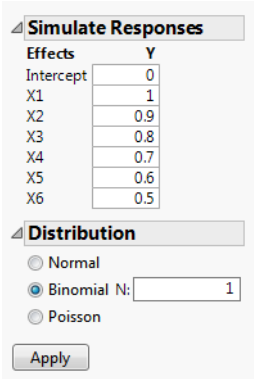
Your plan is to simulate binomial response data where the probability of success is given by a logistic model. For more information about Simulate Response, see the *Design of Experiments Guide*.

Note: If you prefer to skip the steps in this section, click the green triangle next to the **Simulate Model Responses** script. Then go to "Fit the Generalized Linear Model".

1. In the Simulate Responses window (Figure 10.18), enter the following values under Y:
   - Next to Intercept, type 0.
   - Next to X1, 1 is entered by default. Keep that value.
   - Next to X2, type 0.9.
   - Next to X3, type 0.8.
   - Next to X4, type 0.7.
   - Next to X5, type 0.6.
   - Next to X6, type 0.5.
2. In the Distribution outline, select **Binomial**.

   Leave the value for **N** set to 1, indicating that there is only one unit per trial.

Figure 10.19 Completed Simulate Responses Window

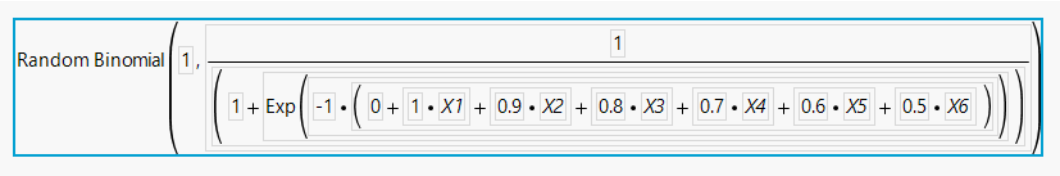


3. Click **Apply**.

In the design data table, the Y Simulated column is replaced with a formula column that generates binomial values. A column called Y N Trials indicates the number of trials for each run.

4. (Optional) Click the plus sign to the right of Y Simulated in the Columns panel.

Figure 10.20 Random Binomial Formula for Y Simulated



5. Click **Cancel**.

JMP PRO Fit the Generalized Linear Model

In this section, you fit a logistic model using the Generalized Linear Model personality.

1. In the data table, click the green triangle next to the **Model** script.
2. Click the Y variable next to the **Y** button and click **Remove**.
3. Click Y Simulated and click the **Y** button.

   You are replacing Y with a column that contains randomly generated binomial values.
4. From the Personality list, select **Generalized Linear Model**.
5. From the Distribution list, select **Binomial**.

   Notice that the Logit function appears in the Link Function menu.
6. Click **Run**.

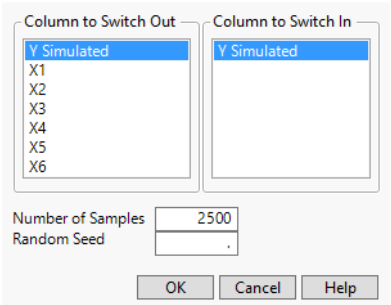
   The model that is fit is based on a single set of simulated binomial responses.

JMP PRO Explore Power

In this section, you simulate likelihood ratio test *p*-values to explore the power of detecting a difference over a range of probability values that is determined by the linear predictor with the coefficient values given in "Plan for the Example".

1. In the Effect Tests outline, right-click in the **Prob>ChiSq** column and select **Simulate**.

Figure 10.21 Simulate Window



Make sure the Y Simulated column is selected in the Column to Switch Out list. This column contains the values that were used to fit the model. When you select the column Y Simulated under Column to Switch In, for each simulation, you are telling JMP to replace the values in Y Simulated with a new column of values that are simulated using the formula in the column Y Simulated.

The column that you have selected in the report, **Prob>ChiSq**, is the *p*-value for a likelihood ratio test of whether the associated main effect is 0. The Prob>ChiSq value is simulated for each effect listed in the Effect Tests table.

2. Next to **Number of Samples**, type **500**.
3. Click **OK**.

A Generalized Linear Model Simulate Results data table appears.

Note: Because response values are simulated, your simulated *p*-values might differ from those shown in Figure 10.22.

Figure 10.22 Table of Simulated Results, Partial View

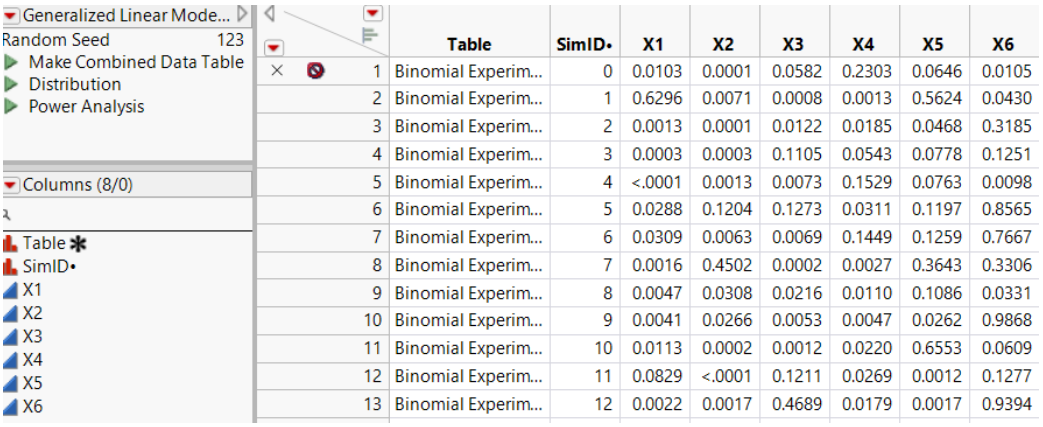


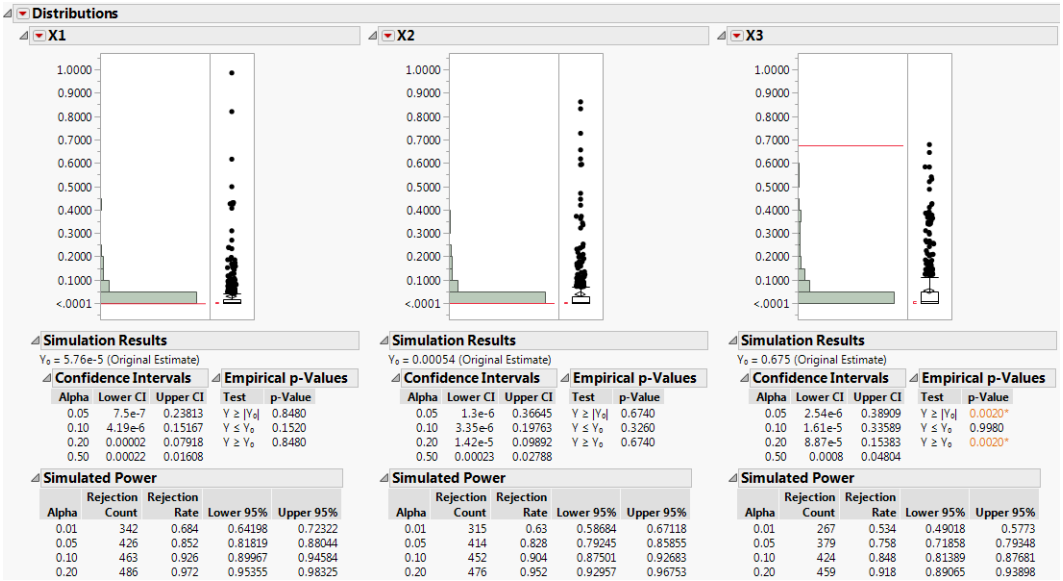
	Table	SimID•	X1	X2	X3	X4	X5	X6
1	Binomial Experim...	0	0.0103	0.0001	0.0582	0.2303	0.0646	0.0105
2	Binomial Experim...	1	0.6296	0.0071	0.0008	0.0013	0.5624	0.0430
3	Binomial Experim...	2	0.0013	0.0001	0.0122	0.0185	0.0468	0.3185
4	Binomial Experim...	3	0.0003	0.0003	0.1105	0.0543	0.0778	0.1251
5	Binomial Experim...	4	<.0001	0.0013	0.0073	0.1529	0.0763	0.0098
6	Binomial Experim...	5	0.0288	0.1204	0.1273	0.0311	0.1197	0.8565
7	Binomial Experim...	6	0.0309	0.0063	0.0069	0.1449	0.1259	0.7667
8	Binomial Experim...	7	0.0016	0.4502	0.0002	0.0027	0.3643	0.3306
9	Binomial Experim...	8	0.0047	0.0308	0.0216	0.0110	0.1086	0.0331
10	Binomial Experim...	9	0.0041	0.0266	0.0053	0.0047	0.0262	0.9868
11	Binomial Experim...	10	0.0113	0.0002	0.0012	0.0220	0.6553	0.0609
12	Binomial Experim...	11	0.0829	<.0001	0.1211	0.0269	0.0012	0.1277
13	Binomial Experim...	12	0.0022	0.0017	0.4689	0.0179	0.0017	0.9394

The first row of the table contains the initial values of **Prob>ChiSq** and is excluded. The remaining 500 rows contain simulated values.

4. Run the **Power Analysis** script.

Note: Because response values are simulated, your simulated power results might differ from those shown in Figure 10.23.

Figure 10.23 Distribution Plots for the First Two Effects



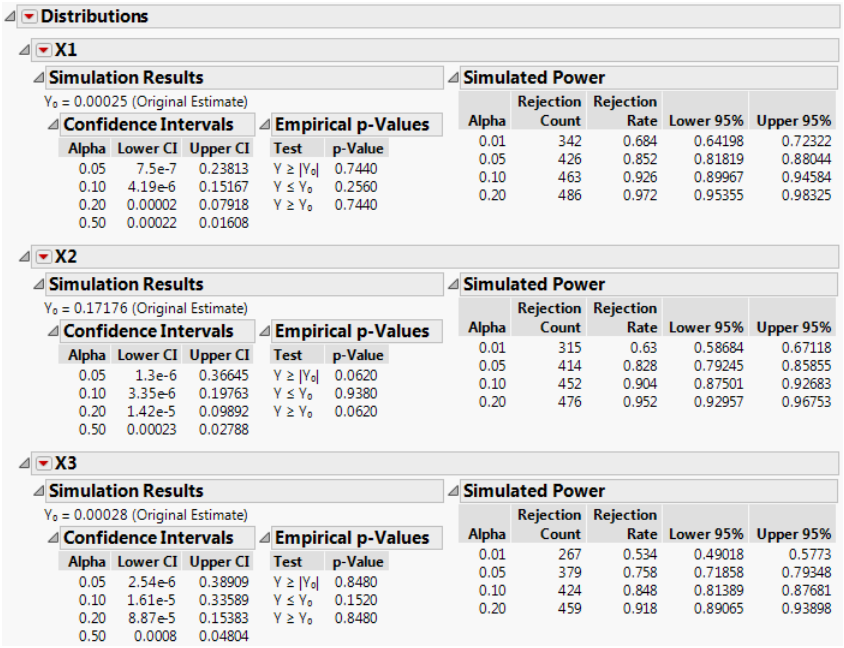
The histograms plot the 500 simulated Prob>ChiSq values for each main effect. The Simulated Power outline shows the simulated Rejection Rate in the 500 simulations.

For easier viewing, stack the reports and de-select the plots.

5. Click the Distributions red triangle and select **Stack**.
6. Press Ctrl and click the X1 red triangle, and de-select **Outlier Box Plot**.
7. Press Ctrl and click the X1 red triangle, select **Histogram Options**, and de-select **Histogram**.

Note: Because response values are simulated, your simulated power results might differ from those shown in Figure 10.24.

Figure 10.24 Power Results for the First Three Effects



Distributions

X1

Simulation Results

Y_0 = 0.00025 (Original Estimate)

Confidence Intervals

Alpha	Lower CI	Upper CI
0.05	7.5e-7	0.23813
0.10	4.19e-6	0.15167
0.20	0.00002	0.07918
0.50	0.00022	0.01608

Empirical p-Values

Test	p-Value
$Y \geq \lvert Y_0 \rvert$	0.7440
$Y \leq Y_0$	0.2560
$Y \geq Y_0$	0.7440

Simulated Power

Alpha	Rejection Count	Rejection Rate	Lower 95%	Upper 95%
0.01	342	0.684	0.64198	0.72322
0.05	426	0.852	0.81819	0.88044
0.10	463	0.926	0.89967	0.94584
0.20	486	0.972	0.95355	0.98325

X2

Simulation Results

Y_0 = 0.17176 (Original Estimate)

Confidence Intervals

Alpha	Lower CI	Upper CI
0.05	1.3e-6	0.36645
0.10	3.35e-6	0.19763
0.20	1.42e-5	0.09892
0.50	0.00023	0.02788

Empirical p-Values

Test	p-Value
$Y \geq \lvert Y_0 \rvert$	0.0620
$Y \leq Y_0$	0.9380
$Y \geq Y_0$	0.0620

Simulated Power

Alpha	Rejection Count	Rejection Rate	Lower 95%	Upper 95%
0.01	315	0.63	0.58684	0.67118
0.05	414	0.828	0.79245	0.85855
0.10	452	0.904	0.87501	0.92683
0.20	476	0.952	0.92957	0.96753

X3

Simulation Results

Y_0 = 0.00028 (Original Estimate)

Confidence Intervals

Alpha	Lower CI	Upper CI
0.05	2.54e-6	0.38909
0.10	1.61e-5	0.33589
0.20	8.87e-5	0.15383
0.50	0.0008	0.04804

Empirical p-Values

Test	p-Value
$Y \geq \lvert Y_0 \rvert$	0.8480
$Y \leq Y_0$	0.1520
$Y \geq Y_0$	0.8480

Simulated Power

Alpha	Rejection Count	Rejection Rate	Lower 95%	Upper 95%
0.01	267	0.534	0.49018	0.5773
0.05	379	0.758	0.71858	0.79348
0.10	424	0.848	0.81389	0.87681
0.20	459	0.918	0.89065	0.93898

In the Simulated Power outlines, the Rejection Rate for each row gives the proportion of *p*-values that are smaller than the corresponding Alpha. For example, for X3, which corresponds to a coefficient value of 0.8 and a probability difference of 38%, the simulated power for a 0.05 significance level is 379/500 = 0.758. Table 10.1 summarizes the estimated power at the 0.05 significance level for all effects. Notice how power decreases as the Difference to Detect decreases. Also notice that the power to detect an effect as large as 24.5% (X6) is only approximately 0.37.

Note: Because response values are simulated, your simulated power results might differ from those shown in Table 10.1.

Table 10.1 Simulated Power at Significance Level 0.05

Factor	Percent Passing at $X_i = 1$	Percent Passing at $X_i = -1$	Difference to Detect	Simulated Power (Rejection Rate) at Alpha=0.05
X_1	73.11%	26.89%	46.2%	0.852
X_2	71.09%	28.91%	42.2%	0.828
X_3	69.00%	31.00%	38.0%	0.758

Table 10.1 Simulated Power at Significance Level 0.05 *(Continued)*

Factor	Percent Passing at $X_i = 1$	Percent Passing at $X_i = -1$	Difference to Detect	Simulated Power (Rejection Rate) at Alpha=0.05
X_4	66.82%	33.18%	33.6%	0.654
X_5	64.56%	35.43%	29.1%	0.488
X_6	62.25%	37.75%	24.5%	0.372

JMP PRO Launch the Simulate Window

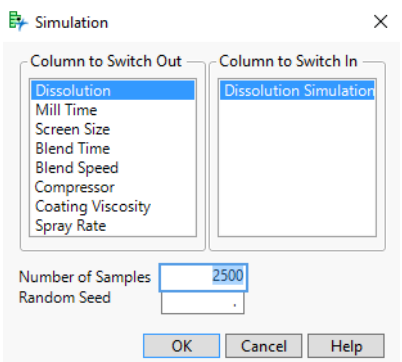
To launch the Simulate platform, right-click a column of calculated values in a report window and select Simulate. The Simulate platform is available in many reports, including all reports that support bootstrapping. To use the Simulate platform, the data table must contain a formula with a random component that simulates data.

Note: The Simulate platform is not available in reports that use a By variable.

JMP PRO The Simulate Window

When you click OK to run the simulation, a window that contains a progress bar and a Stop Early button appears. The number of the sample being simulated is shown above the progress bar. If you click Stop Early, the simulated values that have been computed up to that point are presented in a Simulate Results table. The window also shows you which analyses are being run at any given time.

Figure 10.25 Simulate Window for Tablet Production.jmp



The Simulate window contains these panels and options:

Column to Switch Out The column that is replaced by the Column to Switch In.

Column to Switch In The column that replaces the Column to Switch Out. The analysis is repeated with values simulated according to the formula in the Column to Switch In. Only columns with formulas are listed in the Column to Switch In panel.

Number of Samples Number of times that the report is re-run for a set of simulated data. The default value is 2500.

Random Seed A value that controls the simulated results. The random seed makes the results reproducible.

JMP PRO The Simulate Results Table

Simulate results appear in a table. Note the following:

- The first row of the table contains the values for the table items that appear in the report. For this reason, the first row is always excluded.
- The remaining rows give the simulation results. The number of remaining rows is equal to the Number of Samples that you specified in the Simulate launch window.
- The rows in the report are identified by the first column in the report table that contains the selected column of calculated values. A column appears in the simulated results table for each item in this first column.
- The table contains a **Distribution** script that constructs a Distribution report. This report contains histograms, quantiles, summary statistics, and simulation results for each column in the simulated results data table. In addition to the standard Distribution report, the report contains the following items:
  - A red line that denotes the original estimate appears on the histogram.
  - A Simulation Results report containing the original estimate, as well as confidence intervals and empirical *p*-values for the simulation. See "Simulation Results Report".
  - If the values in the simulated results data table have a PValue format, a Simulated Power report is also provided. See "Simulated Power Report".
- The table contains a **Power Analysis** script only if you have simulated a column of *p*-values. This script constructs a Distribution report showing histograms of *p*-values and provides a Simulated Power report. See "Simulated Power Report".

JMP PRO Simulation Results Report

Original Estimate The value of the original estimate, also shown in the first row of the Simulate Results data table. This estimate is labeled Y_0.

Confidence Intervals Lower and upper limits for quantile-based confidence intervals at the following significance levels: 0.05, 0.10, 0.20, and 0.50.

Empirical p-Values The empirical *p*-values for a two-sided test and both one-sided tests that compare the simulated values to the original estimate. These *p*-values are computed as the proportions of the simulated values that fall in the ranges that are specified in the Test column of the report.

JMP PRO Simulated Power Report

Alpha The significance level: 0.01, 0.05, 0.10, and 0.20.

Rejection Count The number of simulations where the test rejects at the corresponding significance level.

Rejection Rate The proportion of simulations where the test rejects at the corresponding significance level.

Lower 95% and Upper 95% Lower and upper limits for a 95% confidence interval for the simulated rejection rate. The interval is computed using the Wilson score method. See Wilson (1927).

Tip: Increase the Number of Samples for a narrower confidence interval.

Chapter 11

JMP PRO Bootstrapping

Approximate the Distribution of a Statistic through Resampling

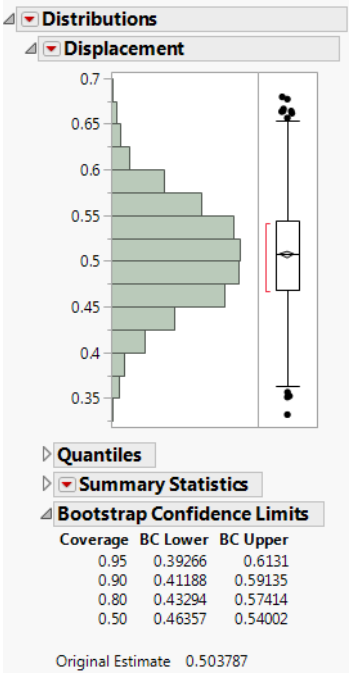
Bootstrapping is available only in JMP Pro.

Bootstrapping is a resampling method for approximating the sampling distribution of a statistic. You can use bootstrapping to estimate the distribution of a statistic and its properties, such as its mean, bias, standard error, and confidence intervals. Bootstrapping is especially useful in the following situations:

- The theoretical distribution of the statistic is complicated or unknown.
- Inference using parametric methods is not possible because of violations of assumptions.

Note: Bootstrap is available only from a right-click in a report. It is not a platform command.

Figure 11.1 Bootstrapping Results for a Slope Parameter



Contents

Overview of Bootstrapping 359
Platforms that Support Bootstrapping 360
Example of Bootstrapping 361
Bootstrapping Window Options 363
Stacked Bootstrap Results Table 365
Unstacked Bootstrap Results Table 366
Analysis of Bootstrap Results 367
Additional Example of Bootstrapping 368
Statistical Details for Bootstrapping 373
Calculation of Fractional Weights 373
Bias-Corrected Percentile Intervals 373

JMP PRO Overview of Bootstrapping

Bootstrapping repeatedly resamples the observations that are used in your report to construct an estimate of the distribution of a statistic or statistics. The observations are assumed to be independent.

In the simple bootstrap, the n observations are resampled with replacement to produce a bootstrap sample of size n. Note that some observations might not appear in the bootstrap sample, and others might appear multiple times. The number of times that an observation occurs in the bootstrap sample is called its *bootstrap weight*. For each bootstrap iteration, the entire analysis that produced the statistic of interest is rerun with these changes:

- the bootstrap sample of n observations is the data set
- the bootstrap weight is a frequency variable in the analysis platform

This process is repeated to produce a distribution of values for the statistic or statistics of interest.

However, the simple bootstrap can sometimes be inadequate. For example, suppose your data set is small or you have a logistic regression setting where you can encounter separation issues. In such cases, JMP enables you to conduct Bayesian bootstrapping using fractional weights. When fractional weights are used, a fractional weight is associated with each observation. The fractional weights sum to n. The statistic of interest is computed by treating the fractional weights as a frequency variable in the analysis platform. For information about fractional weights, see "Fractional Weights" and "Calculation of Fractional Weights".

To run a bootstrap analysis in a report, right-click in a table column that contains the statistic that you want to bootstrap and select Bootstrap.

Note: Bootstrap is available only from a right-click in a report. It is not a platform command.

JMP provides bootstrapping in many statistical platforms. See "Platforms that Support Bootstrapping" for a complete list. The observations that comprise the sample are all observations that are used in the calculations for the statistics of interest. If the report uses a frequency column, the observations in that column are treated as if they were repeated the number of times indicated by the Freq variable. If the report uses a Weight variable, Bootstrap treats it as it was treated in the calculations for the report.

Tip: Bootstrap reruns the entire analysis that appears in the platform report from which Bootstrap is invoked. As a result, Bootstrap might run slowly for your selected column because of extraneous analyses in the report. If Bootstrap is running slowly, remove extraneous options from the platform report before running Bootstrap.

Platforms that Support Bootstrapping

Bootstrapping is available in the following statistical platforms:

- Boosted Tree
- Bootstrap Forest
- Categorical
- Destructive Degradation
- Discriminant
- Distribution
- Fit Curve
- Fit Life by X
- Fit Parametric Survival
- Fit Proportional Hazards
- Fit Y by X
- Generalized Linear Model
- Generalized Regression
- Life Distribution
- Logistic
- Loglinear Variance
- Multiple Correspondence Analysis
- Multivariate
- Neural
- Nonlinear
- Parametric Survival
- Partial Least Squares
- Partition
- Principal Components
- Proportional Hazard
- Standard Least Squares
- Survival
- Uplift

JMP PRO Example of Bootstrapping

This example uses the Car Physical Data.jmp sample data table. A tire manufacturer wants to predict an engine's horsepower from the engine's displacement (in^3). The company is most interested in estimating the slope of the relationship between the variables. The slope values help the company predict the corresponding change in horsepower when the displacement changes.

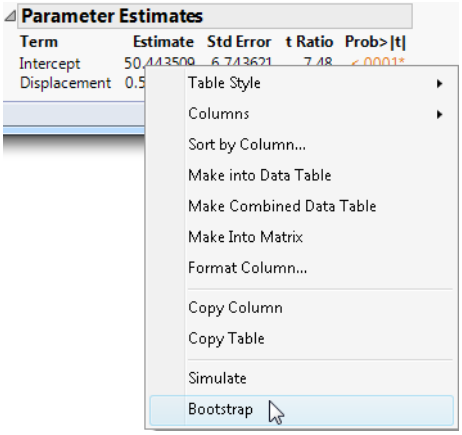
In this example, the regression assumption of homogeneity of variance is violated, so the confidence limits from the regression analysis for the slope might be misleading. For this reason, the company uses a bootstrap estimate of the confidence interval for the slope.

1. Select **Help > Sample Data Library** and open Car Physical Data.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Horsepower and click **Y, Response**.
4. Select Displacement and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Bivariate Fit of Horsepower By Displacement and select **Fit Line**.

   The slope estimate is 0.503787, approximately 0.504.
7. (Optional) Right-click in the **Parameter Estimates** report and select **Columns > Lower 95%**.
8. (Optional) Right-click in the **Parameter Estimates** report and select **Columns > Upper 95%**.

   The confidence limits from the regression analysis for the slope are 0.4249038 and 0.5826711.
9. Right-click the **Estimate** column in the **Parameter Estimates** report and select **Bootstrap**.

Figure 11.2 The Bootstrap Option



The column that you right-click is relevant when the **Split Selected Column** option is selected. See "Bootstrapping Window Options".

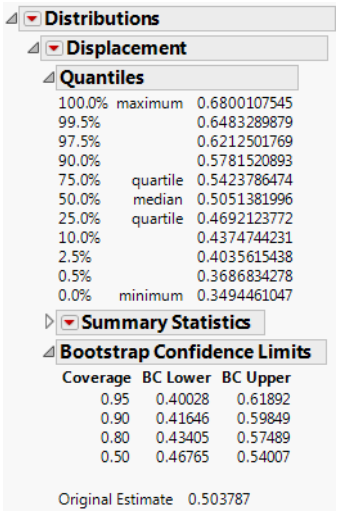
10. Type 1000 for the **Number of Bootstrap Samples**.
11. (Optional) To match the results in Figure 11.3, type 12345 for the **Random Seed**.
12. Click **OK**.

    The bootstrap process runs and produces a Bootstrap Results data table with unstacked results for the slope and intercept.

    Next, analyze the bootstrapped slope.
13. In the Bootstrap Results table, run the Distribution script.

    The Distribution report includes the Bootstrap Confidence Limits report.

Figure 11.3 Bootstrap Report



Distributions

Displacement

Quantiles

100.0%	maximum	0.6800107545
99.5%		0.6483289879
97.5%		0.6212501769
90.0%		0.5781520893
75.0%	quartile	0.5423786474
50.0%	median	0.5051381996
25.0%	quartile	0.4692123772
10.0%		0.4374744231
2.5%		0.4035615438
0.5%		0.3686834278
0.0%	minimum	0.3494461047

Summary Statistics

Bootstrap Confidence Limits

Coverage	BC Lower	BC Upper
0.95	0.40028	0.61892
0.90	0.41646	0.59849
0.80	0.43405	0.57489
0.50	0.46765	0.54007

Original Estimate 0.503787

The estimate of the slope (step 6) is 0.504. Based on the bootstrap results for 95% coverage, the company can estimate the slope to be between 0.40028 and 0.61892. When the displacement is changed by one unit, with 95% confidence, the horsepower changes by some amount between 0.40028 and 0.61892. The bootstrap confidence interval for the slope (0.400 to 0.619) is slightly wider than the confidence interval (0.425 to 0.583) obtained using the usual regression assumptions in step 7 and step 8.

Note: The BC Lower and BC Upper columns in the Bootstrap Confidence Limits report refer to *bias-corrected intervals*. See "Bias-Corrected Percentile Intervals".

JMP PRO Bootstrapping Window Options

To perform a bootstrap analysis, right-click a numeric column of sample statistics in a table in a report window and select **Bootstrap**. The selected column is highlighted, and the Bootstrapping window appears. After you select options and click **OK** in the Bootstrapping window, bootstrap results for every statistic in the column appear in the default results table.

Note: The Bootstrap option is not available in reports that use a By variable.

The Bootstrapping window contains the following options:

Number of Bootstrap Samples Sets the number of times that you want to resample the data and compute the statistics. A larger number results in more precise estimates of the statistics' properties. By default, the number of bootstrap samples is set to 2500.

Random Seed Sets a random seed that you can re-enter in subsequent runs of the bootstrap analysis to duplicate your current results. By default, no seed is set.

Fractional Weights Performs a Bayesian bootstrap analysis. In each bootstrap iteration, each observation is assigned a weight that is calculated as described in "Calculation of Fractional Weights". The weighted observations are used in computing the statistics of interest. By default, the Fractional Weights option is not selected and a simple bootstrap analysis is conducted.

Tip: Use the Fractional Weights option if the number of observations that are used in your analysis is small or if you are concerned about separation in a logistic regression setting.

Suppose that the Fractional Weights option is selected. For each bootstrap iteration, each observation that is used in the report is assigned a nonzero weight. These weights sum to n, the number of observations used in the calculations of the statistics of interest. For more information about how the weights are calculated and used, see "Calculation of Fractional Weights".

Split Selected Column Places bootstrap results for each statistic in the column that you selected for bootstrapping into a *separate* column in the Bootstrap Results table. Each row of the Bootstrap Results table (other than the first) corresponds to a single bootstrap sample.

If you deselect this option, a Stacked Bootstrap Results table appears. For each bootstrap iteration, this table contains results for the entire report table that contains the column that you selected for bootstrapping. Results for each row of the report table appear as rows in the Stacked Bootstrap Results table. Each column in the report table defines a column in the Stacked Bootstrap Results table. For an example, see "Stacked Bootstrap Results Table".

Discard Stacked Table if Split Works (Applicable only if the **Split Selected Column** option is selected.) Determines the number of results tables produced by Bootstrap.

If the Discard Stacked Table if Split Works option *is not* selected, then two Bootstrap tables are shown:

- The Stacked Bootstrap Results table, which contains bootstrap results for each row of the table containing the column that you selected for bootstrapping. This table gives bootstrap results for every statistic in the report, where each column is defined by a statistic.
- The unstacked Bootstrap Results table, which is obtained by splitting the stacked table. This table provides results only for the column that is selected in the original report.

If the Discard Stacked Table if Split Works option *is* selected and if the **Split Selected Column** operation is successful, the Stacked Bootstrap Results table is not shown.

JMP PRO Stacked Bootstrap Results Table

The initial results of a bootstrap analysis appear in a stacked results table (Figure 11.4). This table might not appear if you have selected the **Discard Stacked Table if Split Works** option. Figure 11.4 shows a bootstrap table that is based on the Parameter Estimates report obtained by fitting a Bivariate model in Fit Y by X to Car Physical Data.jmp. See "Overview of Bootstrapping".

Figure 11.4 Stacked Bootstrap Results Table

	X	Y	Term	~Bias	Estimate	Std Error	t Ratio	Prob>|t|	BootID•
1	Displacement	Horsepower	Intercept		50.443509471	6.7436209999	7.48	<.0001	0
2	Displacement	Horsepower	Displacement		0.5037874592	0.0398202611	12.65	<.0001	0
3	Displacement	Horsepower	Intercept		45.385604758	5.4724273046	8.29	<.0001	1
4	Displacement	Horsepower	Displacement		0.5451674092	0.0306151772	17.81	<.0001	1
5	Displacement	Horsepower	Intercept		40.843862813	7.0508187326	5.79	<.0001	2
6	Displacement	Horsepower	Displacement		0.5854988173	0.0457041828	12.81	<.0001	2
7	Displacement	Horsepower	Intercept		47.765642104	5.4677610087	8.74	<.0001	3
8	Displacement	Horsepower	Displacement		0.4943459579	0.0305765677	16.17	<.0001	3
9	Displacement	Horsepower	Intercept		59.758060908	7.0436720036	8.48	<.0001	4
10	Displacement	Horsepower	Displacement		0.4385540785	0.042053201	10.43	<.0001	4
11	Displacement	Horsepower	Intercept		66.341062413	6.4663383596	10.26	<.0001	5
12	Displacement	Horsepower	Displacement		0.3969640071	0.0393100058	10.10	<.0001	5
13	Displacement	Horsepower	Intercept		41.234989734	5.8543901826	7.04	<.0001	6
14	Displacement	Horsepower	Displacement		0.5723548142	0.034172131	16.75	<.0001	6
15	Displacement	Horsepower	Intercept		43.876867815	7.8085946052	5.62	<.0001	7
16	Displacement	Horsepower	Displacement		0.5731377467	0.0439823115	13.03	<.0001	7

Note the following about the stacked results table:

- For each bootstrap sample, there is a row for each value given in the first column of the report table. These values are shown in a column whose name is the name of the first column in the report table. In this example, for each bootstrap sample there is a row containing results for each Term: Intercept and Displacement, which appear in the Term column.
- The data table columns that are used in the analysis appear in the table. In this example, X is Displacement, and Y is Horsepower.
- There is a column for every column in the report table that you are bootstrapping. In this example, the columns are ~Bias, Estimate, Std Error, t Ratio, and Prob>|t|. Note that ~Bias is a column in the Fit Y by X report that is hidden unless one of the parameter estimates is biased.
- The BootID• column identifies the bootstrap sample. The rows where BootID• = 0 correspond to the original estimates. Those rows are marked with an X and have the excluded row state. In this example, each bootstrap sample is used to calculate results for two rows: the results for Intercept and the results for Displacement.

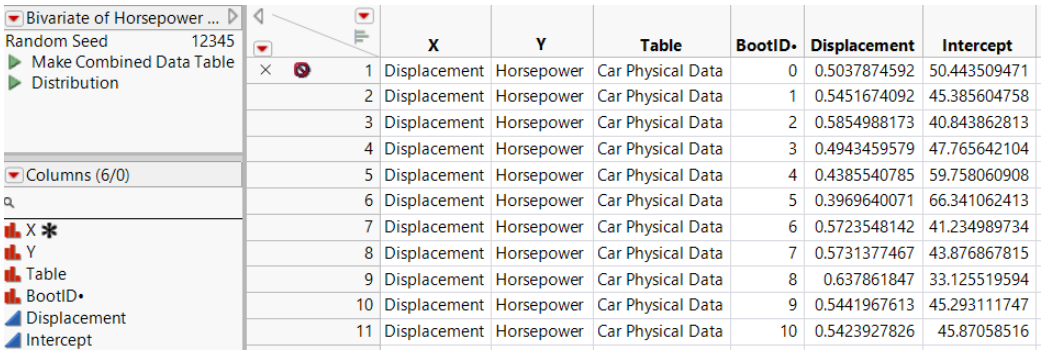
- The data table name begins with “Stacked Bootstrap Results”.

If you selected the **Split Selected Column** option, an unstacked results table might also appear. See “Unstacked Bootstrap Results Table”.

JMP PRO Unstacked Bootstrap Results Table

Select **Split Selected Column** to create a bootstrap table that contains separate columns for the report column that you selected. Each column corresponds to a Term in the report table. For example, in Figure 11.5, the Estimate column from Figure 11.4 is split into two columns (Displacement and Intercept), one for each level of Term.

Figure 11.5 Unstacked Bootstrap Results Table



	X	Y	Table	BootID•	Displacement	Intercept
1	Displacement	Horsepower	Car Physical Data	0	0.5037874592	50.443509471
2	Displacement	Horsepower	Car Physical Data	1	0.5451674092	45.385604758
3	Displacement	Horsepower	Car Physical Data	2	0.5854988173	40.843862813
4	Displacement	Horsepower	Car Physical Data	3	0.4943459579	47.765642104
5	Displacement	Horsepower	Car Physical Data	4	0.4385540785	59.758060908
6	Displacement	Horsepower	Car Physical Data	5	0.3969640071	66.341062413
7	Displacement	Horsepower	Car Physical Data	6	0.5723548142	41.234989734
8	Displacement	Horsepower	Car Physical Data	7	0.5731377467	43.876867815
9	Displacement	Horsepower	Car Physical Data	8	0.637861847	33.125519594
10	Displacement	Horsepower	Car Physical Data	9	0.5441967613	45.293111747
11	Displacement	Horsepower	Car Physical Data	10	0.5423927826	45.87058516

Note the following about the unstacked results table:

- There is a single row for each bootstrap sample.
- The data table columns used in the analysis appear in the table. In this example, X is Displacement, and Y is Horsepower.
- There is a column for each row of the report that was bootstrapped.
- If you specified a Random Seed in the Bootstrapping window, the bootstrap results table contains a table variable called Random Seed that gives its value.
- The unstacked bootstrap results table contains a Source table script and a Distribution table script. The Distribution table script enables you to quickly obtain statistics based on the bootstrap samples, including bootstrap confidence intervals.
- The BootID• column identifies the bootstrap sample. The row where BootID• = 0 corresponds to the original estimates. That row is marked with an X and has the excluded row state. In the unstacked bootstrap table, each row is calculated from a single bootstrap sample.

- The data table name ends with "Bootstrap Results (<colname>)", where <colname> identifies the column in the report that was bootstrapped.

JMP PRO Analysis of Bootstrap Results

Analyze your bootstrap results using the Distribution platform:

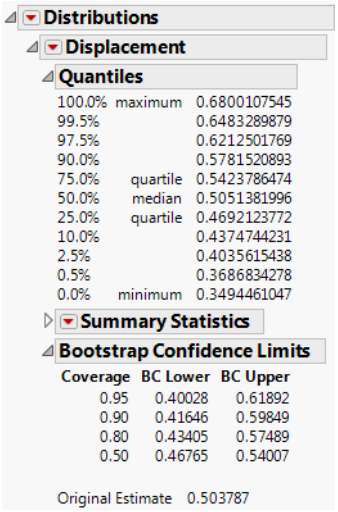
- If your analysis produced an unstacked bootstrap results table, run the Distribution script in the table.
- If your analysis produced a stacked bootstrap results table, select **Analyze > Distribution** and assign the columns of interest to the appropriate roles. In most cases, it is appropriate to assign the column that corresponds to the first column in the report table to the By role.

The Distribution platform provides summary statistics for your bootstrap results. It also produces a Bootstrap Confidence Limits report for any table that contains a BootID• column (Figure 11.6).

You can use the Distribution report to obtain two types of bootstrap confidence intervals:

- The Quantiles report provides *percentile intervals*. For example, to construct a 95% confidence interval using the percentile method, use the 2.5% and 97.5% quantiles as the interval bounds.
- The Bootstrap Confidence Limits report provides *bias-corrected percentile intervals*. The report shows intervals with 95%, 90%, 80%, and 50% coverage levels. The BC Lower and BC Upper columns show the lower and upper endpoints, respectively. For more information about the computation of the bias-corrected percentile intervals, see "Bias-Corrected Percentile Intervals".

Figure 11.6 Bootstrap Confidence Limits Report



Distributions

Displacement

Quantiles

100.0%	maximum	0.6800107545
99.5%		0.6483289879
97.5%		0.6212501769
90.0%		0.5781520893
75.0%	quartile	0.5423786474
50.0%	median	0.5051381996
25.0%	quartile	0.4692123772
10.0%		0.4374744231
2.5%		0.4035615438
0.5%		0.3686834278
0.0%	minimum	0.3494461047

Summary Statistics

Bootstrap Confidence Limits

Coverage	BC Lower	BC Upper
0.95	0.40028	0.61892
0.90	0.41646	0.59849
0.80	0.43405	0.57489
0.50	0.46765	0.54007

Original Estimate 0.503787

The **Original Estimate** at the bottom of the Bootstrap Confidence Limits report is the estimate of the statistic using the original data.

For more information about interpreting the Bootstrap Confidence Limits report, see "Overview of Bootstrapping". Efron (1981) describes the methods for both the percentile interval and the bias-corrected percentile interval.

JMP PRO Additional Example of Bootstrapping

This example illustrates the benefits of the Fractional Weights (Bayesian Bootstrap) option for a small data table. The data consist of a response, Y, measured on three samples of each of seven different soil types. A scientist is interested in finding a confidence interval for the mean response for the wabash soil type.

Because each soil type has only three observations, the simple bootstrap has the potential to exclude all three of the observations for wabash from a bootstrap sample. The Fractional Weights option ensures that all observations for every soil type are represented in all bootstrap samples.

The scientist examines the distribution of wabash sample means from both bootstrap methods:

- "Simple Bootstrap Analysis"
- "Bayesian Bootstrap Analysis"

Simple Bootstrap Analysis

1. Select **Help > Sample Data Library** and open Snapdragon.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Y and click **Y, Response**.
4. Select Soil and click **X, Factor**.
5. Click **OK**.
6. Click the red triangle next to Oneway Analysis of Y By Soil and select **Means/Anova**.
7. In the **Means for Oneway Anova** report, right-click the **Mean** column and select **Bootstrap**.
8. Type 1000 for the **Number of Bootstrap Samples**.
9. (Optional) To match the results in Figure 11.7, type 12345 for the **Random Seed**.
10. Click **OK**.

Figure 11.7 Bootstrap Results for a Simple Bootstrap

	X	Y	BootID	clarion	clinton	compost	knox	o'neill	wabash	webster
1	Soil	Y	0	32.1667	30.3000	29.6667	34.9000	33.8000	35.9667	31.1000
2	Soil	Y	1	32.2333	30.9000	28.0000	35.7000	34.3500	31.9000	31.1000
3	Soil	Y	2	32.4333	30.3000	30.2000	33.9667	31.2000	38.0000	•
4	Soil	Y	3	32.5000	30.7500	29.2000	34.0333	32.8000	38.0000	31.1000
5	Soil	Y	4	31.5000	29.4000	29.9000	35.7500	35.0400	34.9500	•
6	Soil	Y	5	32.5000	30.6000	31.8000	35.7000	34.3000	35.0500	31.5667
7	Soil	Y	6	32.4000	30.1000	29.8500	34.4500	36.0000	38.2000	30.4000
8	Soil	Y	7	31.9000	29.3400	•	34.4000	33.6000	35.9667	31.4500
9	Soil	Y	8	32.1400	31.3500	29.6667	33.1000	35.1000	37.9333	30.6333
10	Soil	Y	9	32.7000	30.8000	30.2000	35.2600	31.2000	34.8500	32.5000
11	Soil	Y	10	32.1000	32.1000	28.0000	33.1000	34.1143	34.0000	31.8000
12	Soil	Y	11	31.5000	30.3000	•	33.1000	34.4000	37.2125	30.6333
13	Soil	Y	12	32.7000	30.4200	31.8000	34.0333	35.1000	35.0500	•
14	Soil	Y	13	32.1667	•	30.1000	34.9000	33.9000	38.2000	31.8000
15	Soil	Y	14	32.3000	•	29.2000	33.9667	34.4000	35.6000	31.9400
16	Soil	Y	15	32.2500	29.1000	29.6667	33.1000	35.1000	31.9000	•
17	Soil	Y	16	31.5000	30.4200	30.0000	•	35.2800	37.8000	31.1000
18	Soil	Y	17	32.4333	29.7000	29.2000	33.1000	34.4000	34.0000	31.1000
19	Soil	Y	18	32.3800	32.1000	29.9000	35.8000	34.6800	35.0500	31.8000
20	Soil	Y	19	32.0600	30.1000	30.2800	•	33.1500	31.9000	31.1000
21	Soil	Y	20	31.9000	29.1000	28.4000	35.9000	34.8000	34.8500	30.8200

The missing values in Figure 11.7 represent bootstrap iterations in which none of the observations for a given soil type were selected for the bootstrap sample.

11. Select **Analyze > Distribution**.
12. Select wabash and click **Y, Columns**.

13. Click **OK**.

Figure 11.8 Distribution of wabash Means from a Simple Bootstrap

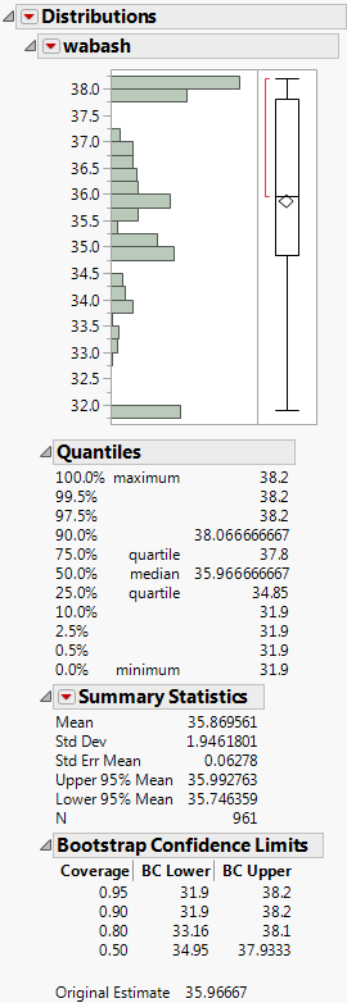


Figure 11.8 shows the distribution of wabash means from the simple bootstrap analysis. Notice the following:

– The Summary Statistics report indicates that the number of rows containing bootstrap means for wabash is N = 961. Although you conducted 1,000 iterations, 39 bootstrap samples did not contain any of the three observations for wabash.

– The histogram of sample means is not smooth, with peaks at the two extremes. The three values for wabash are 38.2, 37.8, and 31.9. The peak at the low end of the distribution results from bootstrap samples that contain only the value 31.9. The peak

at the high end results from bootstrap samples that contain one or both of the values 38.2 and 37.8.

Next, use the Fractional Weights (Bayesian Bootstrap) option to avoid obtaining missing values from the bootstrap samples and to smooth the distribution of bootstrapped means.

Bayesian Bootstrap Analysis

1. In the Oneway Analysis report, right-click the **Mean** column in the **Means for Oneway Anova** report and select **Bootstrap**.
2. Type 1000 for the **Number of Bootstrap Samples**.
3. (Optional) To match the results in Figure 11.9, type 12345 for the **Random Seed**.
4. Select the **Fractional Weights** option.
5. Click **OK**.

Figure 11.9 Bootstrap Results for a Bayesian Bootstrap

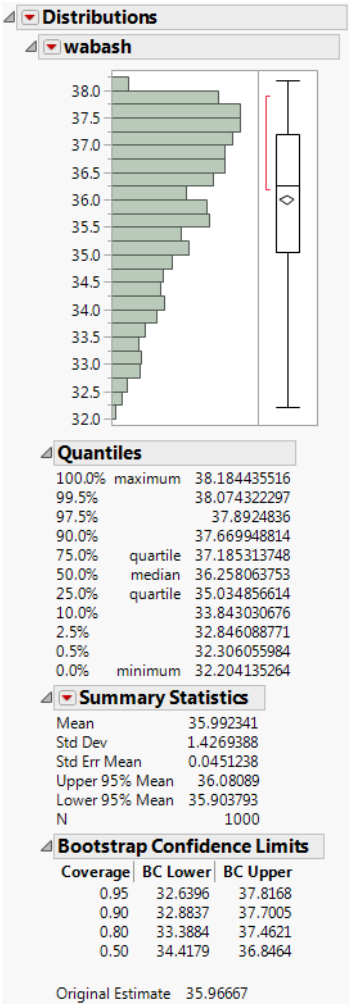
	X	Y	BootID•	clarion	clinton	compost	knox	o'neill	wabash	webster
1	Soil	Y	0	32.1667	30.3000	29.6667	34.9000	33.8000	35.9667	31.1000
2	Soil	Y	1	31.9365	31.3493	29.3234	35.4497	34.7105	33.7270	31.4071
3	Soil	Y	2	32.4189	30.3474	29.5143	35.6281	32.7006	34.1027	32.1212
4	Soil	Y	3	32.2339	30.2001	31.4102	34.8758	33.6674	38.0389	31.5428
5	Soil	Y	4	32.4054	30.3242	30.6227	33.8495	32.9495	36.8344	31.9607
6	Soil	Y	5	32.2262	30.8672	29.7999	33.6759	32.2022	35.5792	31.9058
7	Soil	Y	6	32.3222	31.9732	28.8823	35.6307	34.8863	35.3014	30.1325
8	Soil	Y	7	31.9948	30.8828	29.2516	35.3424	33.2094	36.3367	30.8386
9	Soil	Y	8	31.6254	29.7677	28.6390	34.4697	33.0662	36.6183	31.4667
10	Soil	Y	9	32.3499	29.9416	29.5732	35.2564	31.9583	35.8246	29.9302
11	Soil	Y	10	32.5228	30.4506	28.4859	34.9088	35.8126	34.6317	31.6100
12	Soil	Y	11	31.9057	30.0711	29.0693	35.4018	34.4654	33.2086	31.0309
13	Soil	Y	12	31.7275	29.6189	29.1609	34.3984	33.6840	35.0815	31.5446
14	Soil	Y	13	32.5893	30.5210	28.6054	33.4594	34.0958	33.7692	31.7129
15	Soil	Y	14	32.2473	30.5199	31.6080	35.5617	34.1706	35.7146	31.8023
16	Soil	Y	15	32.2329	29.7275	30.1770	35.3286	32.7820	37.4866	30.9706
17	Soil	Y	16	31.5831	29.7508	28.7122	33.5304	34.5348	37.1092	31.2752
18	Soil	Y	17	32.3545	31.3237	29.1542	35.5890	32.2606	37.2005	30.8430
19	Soil	Y	18	32.3811	29.6241	30.6138	35.4308	33.2024	33.0787	31.2926
20	Soil	Y	19	31.7488	29.7763	28.7327	34.7007	33.8910	34.0573	29.9064

There are no missing values in the Bayesian Bootstrap results table. All 21 rows in the Snapdragon.jmp data table are included, with varying bootstrap weights, in each bootstrap sample.

6. Select **Analyze > Distribution**.
7. Select wabash and click **Y, Columns**.

8. Click **OK**.

Figure 11.10 Distribution of wabash Means from a Bayesian Bootstrap



The Bayesian Bootstrap produces a much smoother distribution for the wabash sample means. All 1,000 bootstrap samples include the three observations for wabash. For each iteration, the wabash sample mean is calculated using different fractional weights.

The Bootstrap Confidence Limits report shows that a 95% confidence interval for the mean is 32.6396 to 37.8168.

JMP PRO Statistical Details for Bootstrapping

JMP PRO Calculation of Fractional Weights

The Fractional Weights option is based on the Bayesian bootstrap (Rubin 1981). The number of times that an observation occurs in a given bootstrap sample is called its *bootstrap weight*. In the simple bootstrap, the bootstrap weights for each bootstrap sample are determined using simple random sampling with replacement.

In the Bayesian approach, sampling probabilities are treated as unknown parameters and their posterior distribution is obtained using a non-informative prior. Estimates of the probabilities are obtained by sampling from this posterior distribution. These estimates are used to construct the bootstrap weights:

- Randomly generate a vector of n values from a gamma distribution with shape parameter equal to $(n - 1)/n$ and scale parameter equal to 1.

  Note: Rubin (1981) uses 1 as the gamma shape parameter. The shape parameter that is used in JMP Pro ensures that the mean and variance of the fractional weights are equal to the mean and variance of the simple bootstrap weights.

- Compute S as the sum of the n values.
- Compute the fractional weights by multiplying the vector of n values by N / S, where N equals the number of rows or the sum of the frequencies if a Freq variable is specified.

Note: If a Freq variable is specified for the analysis, multiply the shape parameter for the gamma distribution by the Freq values on a row-by-row basis. The sum of the values of the Freq variable must be greater than 1. Then the shape parameters are equal to $f_i(N - 1)/N$, where f_i is the Freq value for the i^{th} row and N equals the sum of the Freq values.

This procedure scales the fractional weights for each row to have mean and variance over bootstrap sampling equal to those of the simple bootstrap weights. The fractional bootstrap weights in each bootstrap sample are positive, sum to N, and have a mean of 1.

JMP PRO Bias-Corrected Percentile Intervals

This section describes the calculation of the bias-corrected (BC) confidence intervals that appear in the Bootstrap Confidence Limits report when you run the Distribution script in the Bootstrap Results table. Bias-corrected percentile intervals improve on the ability of percentile intervals in accounting for asymmetry in the bootstrap distribution. See Efron (1981).

Notation

- p^* is the proportion of bootstrap samples with an estimate of the statistic of interest that is less than or equal to the original estimate.
- z_0 is the p^* quantile of a standard normal distribution.
- z_α is the α quantile of a standard normal distribution.

Bias-Corrected Confidence Interval Endpoints

The endpoints of a (1 - α) bias-corrected confidence intervals are given by quantiles of the bootstrap distribution:

- The lower endpoint is the following quantile:

$$\Phi\left(2z_0 + z_{\frac{\alpha}{2}}\right)$$

- The upper endpoint is the following quantile:

$$\Phi\left(2z_0 + z_{1-\frac{\alpha}{2}}\right)$$

Chapter 12

Text Explorer

Explore Unstructured Text in Your Data

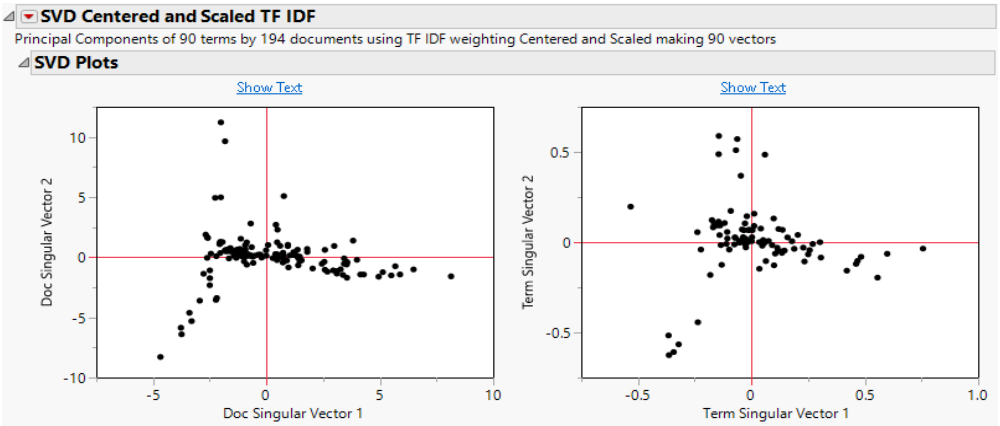
Many features in this platform are available only in JMP Pro and noted with this icon.

The Text Explorer platform enables you to analyze unstructured text, such as comment fields in surveys or incident reports. Interact with the text data by using tools to combine similar terms, recode misspecified terms, and understand the underlying patterns in your textual data.

The JMP Pro version of the platform contains analysis tools that use singular value decomposition (SVD) to group similar documents into topics. You can cluster text documents or cluster terms that are in a collection of documents. You can also cluster documents using latent class analysis.

The JMP Pro version of the platform also contains tools for identifying important terms and sentiments in your documents. Term selection enables you to identify the terms that best explain different responses. Sentiment analysis enables you to identify sentiment terms in document using lexical analysis and scores documents for positive, negative, and overall sentiment. The Sentiment Analysis feature includes basic natural language processing (NLP) support.

Figure 12.1 SVD Plots in Text Explorer



Contents

Overview of the Text Explorer Platform 377
Text Processing Steps 379
Example of the Text Explorer Platform 380
Launch the Text Explorer Platform 383
Customize Regex in the Regular Expression Editor 385
The Text Explorer Report 391
Summary Counts Report 391
Term and Phrase Lists 392
Text Explorer Platform Options 396
Text Preparation Options 396
Text Analysis Options 403
Save Options 404
Report Options 406
Latent Class Analysis 407
Latent Semantic Analysis (SVD) 408
SVD Report 409
SVD Report Options 410
Topic Analysis 412
Topic Analysis Report 413
Topic Analysis Report Options 413
Discriminant Analysis 414
Discriminant Analysis Report 415
Discriminant Analysis Report Options 416
Term Selection 416
Term Selection Settings 417
Term Selection Report 418
Term Selection Report Options 420
Sentiment Analysis 421
Sentiment Analysis Report 422
Sentiment Analysis Report Options 424
Additional Example of the Text Explorer Platform 426

Overview of the Text Explorer Platform

The Text Explorer platform enables you to explore unstructured text in order to better understand its meaning. Unstructured text data are common. For example, unstructured text data could result from a free response field in a survey, product review comments, or incident reports.

Text analysis is often an iterative process, so you might alternate between curating and analyzing the list of terms.

Curating the List of Terms

Text analysis uses some unique terminology. A *term* or *token* is the smallest piece of text, similar to a word in a sentence. However, you can define terms in many ways, including through the use of regular expressions; the process of breaking the text into terms is called *tokenization*.

- A *phrase* is a short collection of terms; the platform has options to manage phrases that are specified as terms in and of themselves.
- A *document* refers to a collection of words; in a JMP data table, the unstructured text in each row of the text column corresponds to a document.
- A *corpus* refers to a collection of documents.

It is often desirable to exclude some common words from the analysis. These excluded words are called *stop words*. The platform has a default list of stop words, but you can also add specific words as stop words. Although stop words are not eligible to be terms, they can be used in phrases.

You can also recode terms; this is useful for combining synonyms into one common term.

Stemming is the process of combining words with identical beginnings (*stems*) by removing the endings that differ. This results in “jump”, “jumped”, and “jumping” all being treated as the term “jump·”. The stemming procedure is similar to the procedure used in the Snowball string processing language. When a phrase is stemmed, each word in the phrase is stemmed as it would be stemmed as a stand-alone term.

Analyzing the List of Terms

Text analysis in the Text Explorer platform uses a *bag of words* approach. Other than in the formation of phrases, the order of terms is ignored. The analysis is based on the term counts.

After you curate the list of terms through the use of regular expressions, stop words, recoding, and stemming, you can perform analyses on the curated list of terms. The analysis options in the platform are based on the *document term matrix* (DTM). Each row in the DTM corresponds to a document (a cell in a text column of a JMP data table). Each column in the DTM corresponds to a term from the curated term list. This approach implements the bag of words approach since it ignores word ordering. In its simplest form, each cell of the DTM contains the frequency (number of occurrences) of the column's term in the row's document. There are various other weighting schemes for the DTM; these are described in "Save Options".

JMP PRO The analysis options that are available in the platform first perform a singular value decomposition (SVD) on the document term matrix. This can greatly reduce the number of columns needed to represent the term information in the data. For more information about singular value decomposition, see *Multivariate Methods*. Hierarchical clustering options are available for clustering the terms and for clustering the documents. These options enable you to group similar terms or documents together.

Text Explorer Platform Workflow

These are the expected steps for using the Text Explorer platform:

1. Specify the method for tokenizing (either built-in or customized regular expression).
2. Use the report to specify additional stop words, add phrases to the term list, perform recodes of terms, and specify exceptions to stemming rules.
3. Specify the preference for stemming.
4. Use word and phrase counts, SVD, and clustering approaches to identify important terms and phrases.

   Note: JMP PRO The SVD and clustering options are available only in JMP Pro.

5. Save results for use in further analysis: the term table, the DTM, the singular vectors, or other results.

   Note: JMP PRO The option to save the singular vectors is available only in JMP Pro.

6. Save Phrase, Recode, and Stop Words properties for use in future analyses of similar text data.

Text Processing Steps

In the JMP Text Explorer platform, text is processed in three stages: tokenizing, phrasing, and terming.

Tokenizing Stage

The Tokenizing stage performs the following operations:

1. Convert text to lowercase.
2. Apply Tokenizing method (either Basic Words or Regex) to group characters into tokens.
3. Recode tokens based on specified recode definitions. Note that recoding occurs before stemming.

Note: Recode operations are processed internally in one pass regardless of the order that they are specified in the report window.

Phrasing Stage

The Phrasing stage collects phrases that occur in the corpus (collection of documents) and enables you to specify that individual phrases be treated as terms. Phrases cannot start or end with a stop word, but they can contain a stop word.

Terming Stage

The Terming stage creates the Term List from the tokens and phrases that result from the previous stages.

For each token, the Terming stage performs the following operations:

1. Check that the minimum and maximum length requirements specified in the launch window are met. Tokens that contain only numbers are excluded from this operation.
2. Check that the token is qualified to become a term; tokens parsed by the Basic Words tokenization method must contain at least one alphabetical or Unicode character. Tokens that contain only numbers are excluded from this operation. The Regex tokenization method uses regular expressions to determine what characters are part of a token.
3. Check that the token is not a stop word.
4. Apply stemming and stem exceptions.

For each phrase that you add, the Terming stage performs the following operations:

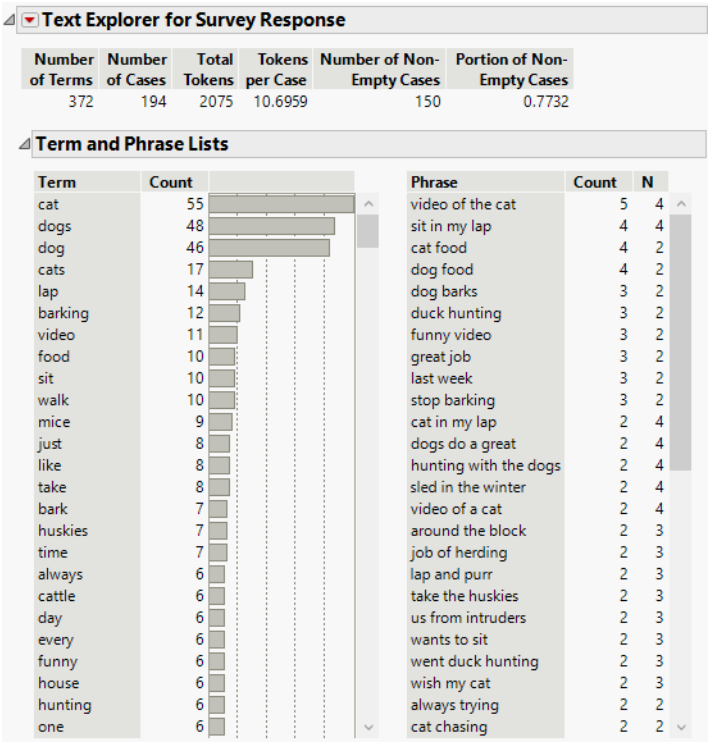
1. Add the phrase to the Term List. Phrases should apply stemming to each word in the phrase that is stemmed in the Term List. Phrases that have different raw tokens but the same stems are combined in the Term List.
2. Remove token term occurrences that appear in the phrase.

Example of the Text Explorer Platform

Learn how to explore text responses in JMP. In this example, you want to explore the text responses from a survey about pets.

1. Select **Help > Sample Data Library** and open Pet Survey.jmp.
2. Select **Analyze > Text Explorer**.
3. Select Survey Response and click **Text Columns**.
4. From the Language list, select **English**.
5. Click **OK**.

Figure 12.2 Example of Initial Text Explorer Report



Text Explorer for Survey Response

Number of Terms	Number of Cases	Total Tokens	Tokens per Case	Number of Non-Empty Cases	Portion of Non-Empty Cases
372	194	2075	10.6959	150	0.7732

Term and Phrase Lists

Term	Count
cat	55
dogs	48
dog	46
cats	17
lap	14
barking	12
video	11
food	10
sit	10
walk	10
mice	9
just	8
like	8
take	8
bark	7
huskies	7
time	7
always	6
cattle	6
day	6
every	6
funny	6
house	6
hunting	6
one	6

Phrase	Count	N
video of the cat	5	4
sit in my lap	4	4
cat food	4	2
dog food	4	2
dog barks	3	2
duck hunting	3	2
funny video	3	2
great job	3	2
last week	3	2
stop barking	3	2
cat in my lap	2	4
dogs do a great	2	4
hunting with the dogs	2	4
sled in the winter	2	4
video of a cat	2	4
around the block	2	3
job of herding	2	3
lap and purr	2	3
take the huskies	2	3
us from intruders	2	3
wants to sit	2	3
went duck hunting	2	3
wish my cat	2	3
always trying	2	2
cat chasing	2	2

At a glance, you can see that there are 372 unique terms in 194 documents. In all, there are 2075 tokenized terms. The most common term is “cat”, and it occurs 55 times.

6. Click the red triangle next to Text Explorer for Survey Response and select **Term Options > Stemming > Stem All Terms**.
7. In the Phrase List table, select **cat food** and **dog food**, right-click the selection, and select **Add Phrase**.

The terms cat food and dog food are included in the Term List.

8. Scroll down in the Term List and find the cat and dog food entries.

   You can see that there are four occurrences of each phrase.

Figure 12.3 Term List after Modifications and Scrolling

Term	Count
barn·	4
cat· food·	4
couch·	4
dog· food·	4
door·	4
duck·	4
get·	4
great·	4
guard·	4
job·	4
know·	4
made·	4
make·	4
now·	4
watch·	4
anymor·	3
ate·	3
back·	3
bath·	3
best·	3
box·	3
eat·	3

In the Phrase List, cat food and dog food are gray, since they are now locally being treated as terms in this Text Explorer report.

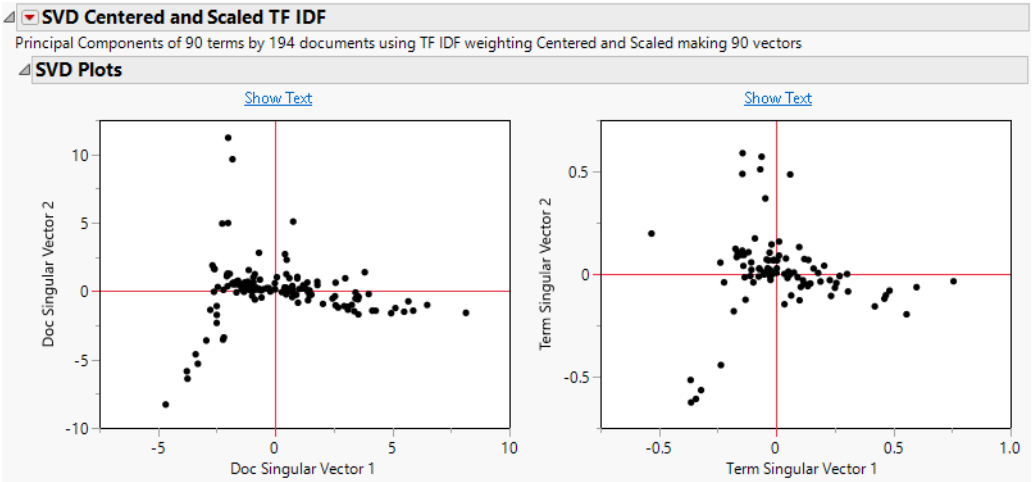
JMP PRO *The remaining steps of this example can be completed only in JMP Pro.*

9. JMP PRO Click the red triangle next to Text Explorer for Survey Response and select **Latent Semantic Analysis, SVD**.

10. JMP PRO Click **OK** to accept the default values.

    Two SVD Plots appear in the report. The one on the left shows the first two singular vectors in the document space. The one on the right shows the first two singular vectors in the term space.

Figure 12.4 SVD Plots



11. JMP PRO Select the seven right-most points in the left SVD Plot.

    These seven points represent survey responses that are clustered away from the rest of the points. To further investigate this cluster, you read the text of these responses.

12. JMP PRO Click the **Show Text** button that is above the left SVD Plot.

Figure 12.5 Text of Selected Documents

```
There was this funny video of a cat trying to jump into someones lap, but fell into the pool instead.
[56]

My kids are always trying to make videos of the cats doing funny things. [78]

The cat jumps out of my lap when he hears someone shake that box of cat food. [79]

The cats are always trying to sit in my lap while I am sitting down to dinner. [93]

The funny cat video where the cats jumped through the window right into the bathtub was hilarious.
[142]

We made this funny video of the cat trying to climb the wall to chase a laser pointer. [153]

My cat always enjoys sitting in my lap and watching cat videos on my tablet. [165]
```

A window appears that contains the text of the seven documents represented by the selected points. These survey responses are similar in that they all refer to some combination of “funny”, “cat”, and “video”. These documents have larger positive values for the first singular vector than the rest of the documents. These larger values indicate that they are different from the rest of the documents in that dimension.

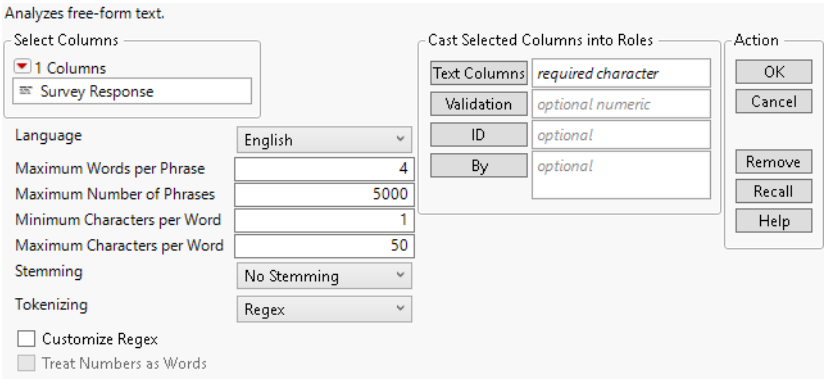
Further investigation of the singular vector dimensions could lead to interpretations of what the dimensions represent. For example, many of the documents on the far right of the plot are responses that are about cats. On the far left, many of the responses are about

dogs. Therefore, the first singular vector is picking up differences based on whether the response was about a cat or a dog.

Launch the Text Explorer Platform

Launch the Text Explorer platform by selecting **Analyze > Text Explorer**.

Figure 12.6 The Text Explorer Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*. The Text Explorer launch window contains the following options:

Text Columns Assigns the columns that contain text data. If you specify multiple columns, a separate analysis is created for each column.

JMP PRO **Validation** In JMP Pro, you can enter a Validation column. If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see *Predictive and Specialized Modeling*.

The specification of a Validation column does not affect the calculation of the document-term matrix. However, when a Validation column is specified, only the training set is used for the Latent Class Analysis, Latent Semantic Analysis, Topic Analysis, and Discriminant Analysis options. The Validation column is used as the Generalized Regression validation method for the Term Selection option.

ID Assigns a column used to identify separate respondents in the Save Stacked DTM for Association output data table. This output data table is suitable for association analysis. This column is also used to identify separate respondents in the Latent Class Analysis report.

By Identifies a column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Note: If you specify a By variable, the Customize Regex option and settings apply to all levels of the By variables.

Language Specifies the language used for text processing. This affects stemming and the built-in lists of stop words, recodes, and phrases. This option is independent of the language in which JMP is running. Unless the Language platform preference is set, the Language option is set according to the JMP Display Language preference. However, the Language option in Text Explorer does not support Korean. If the JMP Display Language is Korean, this option defaults to English.

Maximum Words per Phrase Specifies a maximum number of words that a phrase can contain to be included as a phrase in the analysis.

Maximum Number of Phrases Specifies the maximum number of phrases that appear in the Phrase List.

Minimum Characters per Word Specifies the number of characters that a word must contain to be included as a term in the analysis.

Maximum Characters per Word Specifies the largest number of characters (up to 2000) that a word can contain to be included as a term in the analysis.

Stemming (Available only when the Language option is set to English, German, Spanish, French, or Italian.) Specifies a method for combining terms with similar beginning characters but different endings. The following options are available:

No Stemming No terms are combined.

Stem for Combining Stems only the terms where two or more terms stem to the same term.

Stem All Terms Stems all terms.

Note: The use of the Stemming option also affects phrases that have been added to the Term List. Phrase identification occurs after terms within a phrase have been stemmed. For example, “dogs bark” and “dog barks” would both match the specified phrase “dog· bark·”.

Tokenizing (Available only when the Language option is set to English, German, Spanish, French, or Italian.) Specifies a method for parsing the text into terms or tokens. The following tokenization options are available:

Regex Parses text using a default set of built-in regular expressions. If you want to add to, remove, or edit the set of regular expressions used to parse the text, select the **Customize Regex** option. See "Customize Regex in the Regular Expression Editor".

Basic Words Text is parsed into words based on a set of characters that typically separate words. These characters include spaces, tabs, new lines, and most punctuation marks. If you want numbers to be parsed into terms for the analysis, select the **Treat Numbers as Words** option. If you do not select this option, pieces of text between delimiters that contain only numbers are ignored in the tokenizing step.

Tip: You can view the default set of delimiters using the **Display Options > Show Delimiters** option in a Text Explorer report that uses the Basic Words Tokenizing method.

Customize Regex (Available only with the Regex Tokenizing method.) Enables you to use the Text Explorer Regular Expression Editor window to modify the Regex settings. Use this option to accommodate non-traditional words. Examples include phone numbers or words formed by a combination of characters and numbers. Using the Customize Regex option is not recommended unless the default Regex method is not giving you the results that you need. This can happen when your text contains structures that the default Regex method does not recognize. See "Customize Regex in the Regular Expression Editor".

Treat Numbers as Words (Available only with the Basic Words Tokenizing method.) Allows numbers to be tokenized as terms in the analysis. When this option is selected, the Minimum Characters per Word setting is ignored for terms that contain numeric digits.

After you click **OK** on the launch window, the Text Explorer Regular Expression Editor window appears if you selected **Customize Regex** in the launch window. Otherwise, the Text Explorer report appears.

Note: The processing of text input is not case-sensitive. All text is converted to lowercase internally prior to tokenization and all analysis steps. This conversion affects the processing of regular expressions and the aggregation of terms in the Text Explorer output.

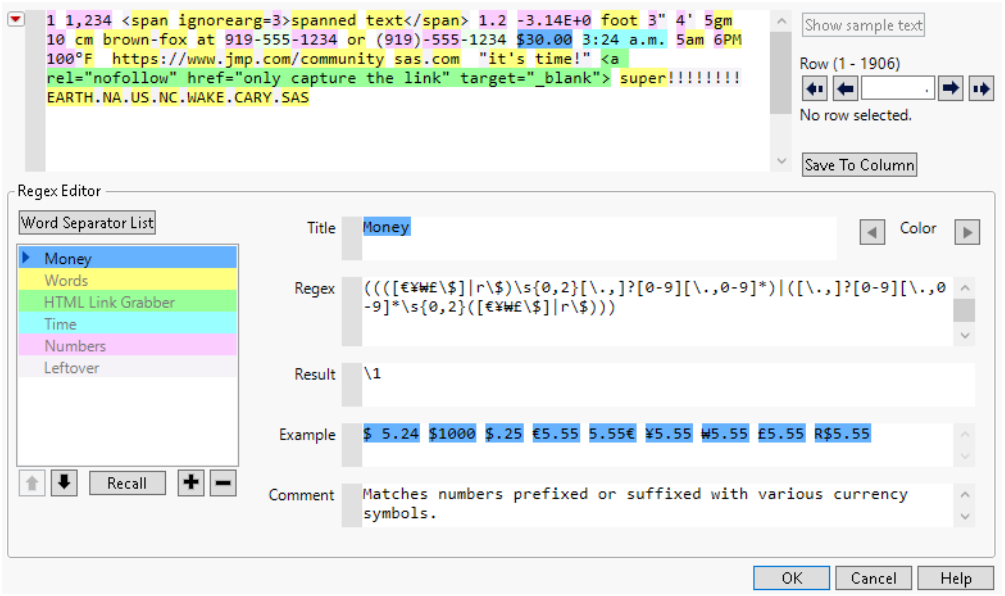
Customize Regex in the Regular Expression Editor

When you select the **Customize Regex** option, the Text Explorer Regular Expression Editor appears. Use this window to parse text documents using a wide variety of built-in regular expressions, such as phone numbers, times, or monetary values. You can also create your own regular expression definitions.

Note: Using the Customize Regex option is recommended only if you are not getting desired results from the default Regex method. This can happen when your text contains structures that the default Regex method does not recognize.

Tip: If Japanese, Chinese (Simplified), or Chinese (Traditional) is specified as the Language option in the launch window, the list of Regex patterns contains a single Regex for the specified language. If you want to add other Regex patterns, it is recommended that you add them after the single Regex pattern. You should avoid using the Words pattern before the language-specific Regex pattern, because the Words pattern can gather long runs of Asian language characters into single words.

Figure 12.7 Text Explorer Regular Expression Editor



Parsing with the Script Editor Box

The script editor box at the top of the window shows you how the parsing would proceed for sample text. The results of parsing the regular expressions in the Regex Editor list are highlighted in colors that correspond to the colors in the Regex Editor list.

- Click the **First**, **Previous**, **Next**, and **Last** row buttons to populate the script editor box with text from your own data. This enables you to see how a given row of text data is parsed. You can also enter a row number in the edit box to populate the script editor box with text from a specific row in the data table.
- Click the **Save to Column** button to save a new column to the data table that contains the result of the regular expression tokenization. For more information about specifying the result of the regular expression, see "Editing the Regular Expressions". The **Save to Column** button does not appear if you access the Regular Expression Editor through **Cols >Utilities > New Column by Text Matching**.

Note: The **Save to Column** button uses only the regular expression to match text. The following settings are not used: stop words, recodes, stemming, phrases, or minimum and maximum characters per word to modify the output of the regular expression.

Adding Regular Expressions

To add a regular expression to be used in tokenization, click the plus sign below the list. The Regex Library Selections window appears. This window contains all the built-in regular expressions as well as any recently modified regular expressions that you created in previous instances of the Regular Expression Editor. Built-in regular expressions are labeled. Custom regular expressions that are saved in your library are labeled with the name that you specified. Only the most recent expression for a given name is stored in the Regex Library.

Click the **Recall** button to populate the regular expressions list with the regular expressions from the most recent instance of the Regular Expression Editor. The recalled regular expressions are ones that were present in the previous instance of the editor when either the **Save to Column** button or the **OK** button was clicked.

Select one or more regular expressions in the list and click **OK** to add the selected regular expressions to be used in tokenization. Use the **Delete Selected Item** button to remove one or more custom regular expressions from the Regex Library. The Regex Library for each user is stored as a JSL file in a directory called TextExplorer. The location of this directory is based on your computer's operating system:

- Windows: "C:/Users/<username>/AppData/Roaming/SAS/JMP/TextExplorer/"
- macOS: "/Users/<username>/Library/Application Support/JMP/TextExplorer/"

These files can be shared with other users, but you should not edit the file directly. Use the Regular Expression Editor instead.

Editing the Regular Expressions

Terms are tokenized by processing the regular expressions in the order specified in the Regex Editor panel. To change the order of the regular expressions, select a regular expression in the list and click the up or down arrow buttons below the list. You can also drag and drop items in the regular expression list to change the order of execution. A blue triangle represents the currently selected regular expression. To remove a regular expression and exclude it from the tokenization, select it in the list and click the minus sign below the list. The "Leftover" regular expression cannot be removed and must appear last in the sequence of regular expressions.

When you select a regular expression in the list, the editable fields in the Regex Editor panel refer to the selected regular expression. Click and type in any of these fields to edit them.

Each regular expression has the following attributes:

Title Specifies a name used to identify the regular expression in the current window (as well as in the Regex Library later).

Regex Specifies the regular expression definition. The regular expression must have at least one set of parentheses to designate the regular expression capture.

Result Specifies what replaces the text matched by the regular expression. This value can be static text, blank, or the value of the regular expression capture. The regular expression capture is defined as the result of the Regex definition:

- To replace the matched text with static text, specify the static text in the Result field.
- To ignore the matched text, leave the Result field blank.
- To keep the text that results from the outer-most parentheses in the regular expression, use "\1" (without quotation marks) in the Result field.
- To keep the entire result of the regular expression, use "\0" (without quotation marks) in the Result field.

Example (Optional) Specifies an example text string with colors indicating the behavior of the regular expression.

Comment (Optional) Specifies a comment to explain the regular expression and its behavior.

Color Specifies the color used to identify matches of the regular expression in the text in the Script Editor box and in the Example field. Use the arrow buttons to change the color.

Note: If the regular expression definition in the Regex field is invalid, a red X appears next to the name of the regular expression in the list of regular expressions.

Creating a Custom Regular Expression

Follow these steps to create your own custom regular expression:

1. Click the plus sign below the list.
2. In the Regex Library Selections window, note that the Blank regular expression is selected.
3. Click **OK**.
4. Edit the Regex definition in the Regex Editor panel.
5. Give your custom regular expression a unique name in the Title field.

Tip: When editing the Regex definition field, it is helpful to have the Log window open and visible. Some error messages appear only in the Log window. To open the Log window, select **View > Log**. There are many internet resources available for troubleshooting regular expressions, such as https://regexr.com/.

The Word Separator List

The **Word Separator List** button enables you to specify a list of characters that occur between words in the tokenization process. The *between-word characters* cannot begin a word, but they can appear inside a word if one of the regular expressions allows it. You can add or remove characters from the list in the window that appears when you click the button. By default, the only character in the list is a whitespace character. In the Separator Characters window, click the **Reset** button to undo any modifications to the list of separator characters. Modifications to the list of separator characters are applied only to the current regular expression tokenization.

The following steps describe the processing of the specified regular expressions and the required "Leftover" regular expression:

1. Compare the current character in the text stream to the list of separator characters.
   - If the character is in the list of separator characters, ignore the character, process any accumulated characters in the "Leftover" temporary string, move to the next character, and repeat step 1.
   - If the character is not in the list of separator characters, go to step 2.
2. Compare the string starting at the current character to each regular expression (one at a time, up to, but not including, the "Leftover" regular expression).
   - If the string starting at the current character matches one of the regular expressions, the following events occur. Any accumulated characters in the "Leftover" temporary string are processed. The value of the Result field is saved as a term. The current character in the text stream becomes the character following the matched string. The processing returns to step 1.
   - If the string starting at the current character does not match any of the regular expressions up to the "Leftover" regular expression, go to step 3.
3. Collect characters into the "Leftover" temporary string by appending the current character and setting the current character to the next character in the text stream. Return to step 1.
   - The "Leftover" temporary string is accumulated one character at a time, until one of the other regular expressions produces a match.
   - The default Result of the "Leftover" regular expression is to discard the accumulated "Leftover" temporary string.

Tips:

- If you set the Result of the "Leftover" regular expression to \1, you might want to add more separator characters, such as punctuation marks. This ensures that your results do not include the specified punctuation marks.
- Instead of changing the Result of the "Leftover" regular expression to \1, you might want to consider one or more of the following actions to capture terms of interest:

- Add more regular expressions from the Regex Library.
- Create custom regular expressions.

The processing follows the above steps until reaching the end of the text string for each row in the data table.

Saving the Results to a Column in the Data Table

Click the **Save to Column** button to save to the data table a new column that contains the results of the regular expression tokenization. The new column is a character column with the same name as the text column specified in the Text Explorer launch window; a number is appended to the name so that the column names are unique. You can also use a stand-alone Regular Expression utility under Cols > Utilities > New Column by Text Matching. See *Using JMP*.

Note: When you save the results of the custom regular expression tokenization to a column in the data table, the regular expression process is run on the original text in each row of the data table. It is not run on the version of the text string that was converted to lowercase.

Closing the Text Explorer Regular Expression Editor

After you click **OK** in the Text Explorer Regular Expression Editor window, the following events occur:

1. The custom regular expressions defined in the Text Explorer Regular Expression Editor window are saved to the Regex Library.

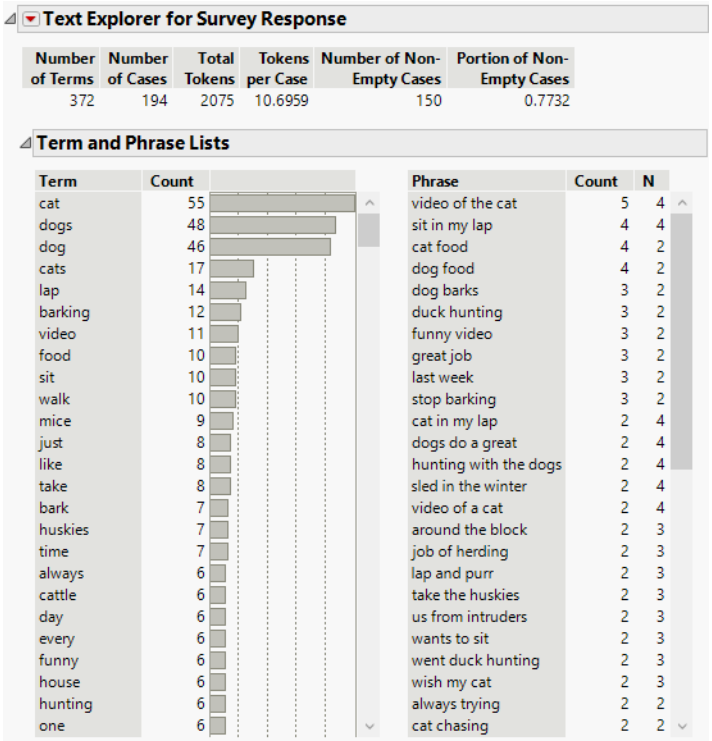
   Caution: The custom Regex Library is saved only when you click **OK** and there are customized regular expressions. The most recently saved regular expressions will be available next time. Use unique names to keep additional regular expressions in the Regex Library. To ensure that a regular expression is available later, you can save a script from the Text Explorer report window.

2. The Text Explorer report appears. The report shows the result of using the specified regular expression settings to tokenize the text.

The Text Explorer Report

The Text Explorer report window contains the Summary Counts report and the Term and Phrase Lists report.

Figure 12.8 Example of a Text Explorer Report



Text Explorer for Survey Response

Number of Terms	Number of Cases	Total Tokens	Tokens per Case	Number of Non-Empty Cases	Portion of Non-Empty Cases
372	194	2075	10.6959	150	0.7732

Term and Phrase Lists

Term	Count
cat	55
dogs	48
dog	46
cats	17
lap	14
barking	12
video	11
food	10
sit	10
walk	10
mice	9
just	8
like	8
take	8
bark	7
huskies	7
time	7
always	6
cattle	6
day	6
every	6
funny	6
house	6
hunting	6
one	6

Phrase	Count	N
video of the cat	5	4
sit in my lap	4	4
cat food	4	2
dog food	4	2
dog barks	3	2
duck hunting	3	2
funny video	3	2
great job	3	2
last week	3	2
stop barking	3	2
cat in my lap	2	4
dogs do a great	2	4
hunting with the dogs	2	4
sled in the winter	2	4
video of a cat	2	4
around the block	2	3
job of herding	2	3
lap and purr	2	3
take the huskies	2	3
us from intruders	2	3
wants to sit	2	3
went duck hunting	2	3
wish my cat	2	3
always trying	2	2
cat chasing	2	2

Summary Counts Report

The first table in the Text Explorer report window contains the following summary statistics:

Number of Terms The number of terms in the Term List.

Number of Cases The number of documents in the corpus.

Total Tokens The total number of terms in the corpus.

Tokens per Case The number of tokens divided by the number of cases.

Number of Non-Empty Cases The number of documents in the corpus that contain at least one term.

Portion of Non-Empty Cases The proportion of documents in the corpus that contain at least one term.

Term and Phrase Lists

The Term and Phrase Lists report contains tables of terms and phrases found in the text after tokenization has occurred. See Figure 12.8 for an example of the Term and Phrase Lists report. The Count column in the Term List indicates the number of occurrences of the term in the corpus. The Count column in the Phrase List indicates the number of occurrences of the phrase in the corpus; the N column indicates the number of words in the phrase.

By default, the Terms List is sorted in descending count order; terms that are tied in count are sorted alphabetically. The Phrases List is sorted in descending count order; phrases that are tied in count are then sorted in descending length (N) order. Further ties in the Phrases List are sorted alphabetically. The sort order of each list can be changed to alphabetical sorting using the options in each list.

The phrases that appear in the Phrase List are determined by the settings of the **Maximum Words per Phrase** and **Maximum Number of Phrases** options in the launch window. Phrases that occur only one time in the data table do not appear in the Phrase List.

Phrases can be specified as terms at various scopes. Phrases in the Phrase List that have been specified as terms are colored based on the scope of the phrase specification (Table 12.1). For more information about specifying phrases in different scopes, see "Term Options Management Windows".

Table 12.1 Colors for Specified Phrases

Scope	Color
Built-in	Red
User Library	Green
Project	Blue
Column Property	Orange
Local	Gray

Actions for Terms and Phrases

You can access options in the Term List and Phrase List tables by selecting items and then right-clicking in the left-most column of each table. You can save each table as a data table by right-clicking in the Count column of each table and selecting Make into Data Table.

Term List Pop-up Menu Options

When you right-click in the Term column of the Term List table, a pop-up menu appears with the following options:

Select Rows Selects rows in the data table that contain the selected terms.

Show Text Shows the documents that contain the selected terms.

> **Note:** By default, only the first 10,000 documents are shown. If there are more than 10,000 documents that contain a selected term, a window appears that enables you to increase this limit.

Alphabetical Order Specifies the sort order of the Term List. When this option is selected, the terms are sorted in alphabetical order. When this option is not selected, the terms are sorted in descending Count order.

Numerical Order (Available only when the Alphabetical Order option is selected.) Specifies the sort order of the Term List. When this option is selected, the items are split into string and numeric segments, and the numeric segments are then sorted in numerical order. For more information about the sorting rules used by the Numerical Order option, see *Using JMP*.

Copy Places the selected terms onto the clipboard.

Color Enables you to assign a color to the selected terms.

Label Places labels on the corresponding points in the Term SVD Plot for the selected terms.

Containing Phrases Selects the phrases in the Phrase List table that contain the selected terms.

Save Indicators Saves an indicator column to the data table for each term selected in the Term List. The value of the indicator column for each row is 1 if the document in that row contains the term and 0 otherwise.

Save Formula Saves a column formula to the data table for each term selected in the Term List. The column formula for each row evaluates to 1 if the document in that row contains the term and 0 otherwise. This is useful for new documents.

Recode Enables you to change the values for one or more terms. Select the terms in the list before selecting this option. After you select this option, the Recode window appears. See *Using JMP*.

Add Stop Word Adds the selected terms to the list of stop words and removes those terms from the Term List. This action also updates the Phrase List.

Note: If you add a stemmed word as a stop word, all of the tokens that correspond to that stem are added as stop words.

Add Stem Exception (Available only when the Language option is set to English, German, Spanish, French, or Italian.) Adds the selected terms to the list of terms that are excluded from stemming.

Remove Phrase (Available only when a specified phrase is selected in the Term List.) Removes the selected phrase from the set of specified phrases and updates the Term Counts accordingly.

Note: If a phrase as been added as a Sentiment Phrase, the Remove Phrase option also removes the phrase from the list of sentiment terms in the current Sentiment Analysis report.

JMP PRO **Add Sentiment** (Available only when a Sentiment Analysis report is open in the current report window.) Adds the selected terms to the list of sentiment terms in the current Sentiment Analysis report.

Note: If you add a stemmed word as a sentiment term, all of the tokens that correspond to that stem are added as sentiment terms.

Show Filter Shows or hides a search filter above the Term List. See "Search Filter Options".

Make into Data Table Creates a JMP data table from the report table.

Make Combined Data Table Searches the report for other tables like the one you selected and combines them into a single JMP data table.

Phrase List Pop-up Menu Options

When you right-click in the Phrase column of the Phrase List table, a pop-up menu appears with the following options:

Select Rows Selects rows in the data table that contain the selected phrases.

Show Text Shows the documents that contain the selected phrases.

Save Indicators Saves an indicator column to the data table for each phrase selected in the Phrase List. The value of the indicator column for each row is 1 if the document in that row contains the phrase and 0 otherwise.

Alphabetical Order Specifies the sort order of the Phrase List. When this option is selected, the terms are sorted in alphabetical order. When this option is not selected, the terms are sorted in descending Count order.

Numerical Order (Available only when the Alphabetical Order option is selected.) Specifies the sort order of the Phrase List. When this option is selected, the items are split into string and numeric segments, and the numeric segments are then sorted in numerical order. For more information about the sorting rules used by the Numerical Order option, see *Using JMP*.

Copy Places the selected phrases onto the clipboard.

Select Contains Selects larger phrases in the Phrase List that contain the selected phrase.

Select Contained Selects smaller phrases in the Phrase List and terms in the Term List that are contained by the selected phrase.

Add Phrase Adds the selected phrases to the Term List and updates the Term Counts accordingly.

Add Stop Word Adds the selected phrases to the list of stop words. This action also updates the Term List.

JMP PRO **Add Sentiment Phrase** (Available only when a Sentiment Analysis report is open in the current report window.) Adds the selected phrases to the Term List and to the list of sentiment terms in the current Sentiment Analysis report.

Show Filter Shows or hides a search filter above the Phrase List. See "Search Filter Options".

Make into Data Table Creates a JMP data table from the report table.

Make Combined Data Table Searches the report for other tables like the one you selected and combines them into a single JMP data table.

Search Filter Options

Click the down arrow button next to the search box to refine your search.

Contains Terms Returns items that contain a part of the search criteria. A search for "ease oom" returns messages such as "Release Zoom".

Contains Phrase Returns items that contain the exact search criteria. A search for "text box" returns entries that contain "text" followed directly by "box" (for example, "Context Box" and "Text Box").

Starts With Phrase Returns items that start with the search criteria.

Ends With Phrase Returns items that end with the search criteria.

Whole Phrase Returns items that consist of the entire string. A search for "text box" returns entries that contain only "text box".

Regular Expression Enables you to use the wildcard (*) and period (.) in the search box. Searching for “get.*name” looks for items that contain “get” followed by one or more words. It returns “Get Color Theme Names”, “Get Name Info”, and “Get Effect Names”, and so on.

Invert Result Returns items that do not match the search criteria.

Match All Terms Returns items that contain both strings. A search for “t test” returns elements that contain either or both of the search strings: “Pat Test”, “Shortest Edit Script” and “Paired t test”.

Ignore Case Ignores the case in the search criteria.

Match Whole Words Returns items that contain each word in the string based on the Match All Terms setting. If you search for “data filter”, and Match All Terms is selected, entries that contain both “data” and “filter” are returned.

Text Explorer Platform Options

This section describes the options available in the Text Explorer platform.

- “Text Preparation Options”
- “Text Analysis Options”
- “Save Options”
- “Report Options”

Text Preparation Options

The Text Explorer red triangle menu contains the following options for text preparation:

Display Options Shows a submenu of options to control the report display.

Show Word Cloud Shows or hides the Word Cloud report. The Word Cloud red triangle menu enables you to change the layout and font for the word cloud. See “Word Cloud Options”.

The word cloud can be interactively resized by changing the width. The height is then determined automatically. The rows in the Term List are linked to the terms in the Word Cloud.

Show Term List Shows or hides the Term List.

Show Phrase List Shows or hides the Phrase List.

Show Term and Phrase Options Shows buttons in the Term and Phrase Lists report corresponding to the options available in the pop-up menus for each list. See "Term and Phrase Lists".

Show Summary Counts Shows or hides the Summary Counts table. See "Summary Counts Report".

Show Stop Words Shows or hides a list of the stop words used in the analysis. A built-in list of stop words is used initially. To add a stop word, right-click it in the Term List and select **Add Stop Word** from the pop-up menu. See "Term Options Management Windows".

Show Recodes Shows or hides a list of the recoded terms. See "Term Options Management Windows".

Show Specified Phrases Shows or hides a list of the phrases that have been specified by the user to be treated as terms. See "Term Options Management Windows".

Show Stem Exceptions (Available only when the Language option is set to English, German, Spanish, French, or Italian.) Shows or hides the terms that are excluded from stemming. See "Term Options Management Windows".

Show Delimiters (Available only when the Language option is set to English, German, Spanish, French, or Italian and the selected Tokenizing method is Basic Words.) Shows or hides the delimiters used by the Basic Words Tokenizing method. To modify the set of delimiters used, you must use the `Add Delimiters()` or `Set Delimiters()` messages in JSL.

Show Stem Report (Available only when the Language option is set to English, German, Spanish, French, or Italian and the selected Stemming method is not No Stemming.) Shows or hides the Stemming report that contains two tables of stemming results. The table on the left maps each stem to the corresponding terms. The table on the right maps each term to its corresponding stem.

Show Selected Rows Opens a window that contains the text of the documents that are in the currently selected rows.

Show Filters for All Tables Shows or hides filters that can be used for searching tables in the report. This option applies to the following tables: Stop Words, Specified Phrases, Stem Exceptions, Term List, Phrase List, and the Stem Report. For more information about the filter tool, see "Search Filter Options".

Term Options Shows a submenu of options that apply to the Term List.

Stemming (Available only when the Language option is set to English, German, Spanish, French, or Italian.) See the description of stemming options in "Launch the Text Explorer Platform".

Include Builtin Stop Words Specifies if the stop words used in the tokenizing process include built-in stop words or not.

Include Builtin Phrases Specifies if the phrases used in the tokenizing process include built-in phrases or not.

Manage Stop Words Shows a window that enables you to add or remove stop words. The changes made can be applied at the User, Column, and Local levels. You can also specify Local Exceptions that exclude stop words that are specified in any of the other levels. See "Term Options Management Windows".

Manage Recodes Shows a window that enables you to add or remove recodes. The changes made can be applied at the User, Column, and Local levels. You can also specify Local Exceptions that exclude recodes that are specified in any of the other levels. See "Term Options Management Windows".

Manage Phrases Shows a window that enables you to add or remove the phrases that are treated as terms. The changes made can be applied at the User, Column, and Local levels. You can also specify Local Exceptions that exclude phrases that are specified in any of the other levels. See "Term Options Management Windows".

Manage Stem Exceptions (Available only when the Language option is set to English, German, Spanish, French, or Italian.) Shows a window that enables you to add or remove exceptions to stemming. The changes made can be applied at the User, Column, and Local levels. You can also specify Local Exceptions that exclude stem exceptions that are specified in any of the other levels. See "Term Options Management Windows".

Parsing Options Shows a submenu of options that apply to parsing and tokenization.

Tokenizing (Available only when the Language option is set to English, German, Spanish, French, or Italian.) See the description of tokenizing options in "Launch the Text Explorer Platform".

Customize Regex (Available only with the Regex Tokenizing method.) Shows the Customize Regex window. This option enables you to modify the Regex settings for the current Text Explorer report.

Note: If you specified a By variable in the platform launch window, the Customize Regex option automatically broadcasts to all level of the By variables.

Treat Numbers as Words (Available only when the Language option is set to English, German, Spanish, French, or Italian and Basic Words is the selected Tokenizing method.) Allows numbers to be tokenized as terms in the analysis. Note that this option is affected by the setting for Minimum characters per word.

Word Cloud Options

The Word Cloud red triangle menu contains the following options:

Layout Specifies the arrangement of the terms in the Word Cloud. By default, the Layout is set to Ordered.

Ordered Presents the terms in horizontal lines ordered from most to least frequent.

Alphabetical Presents the terms in horizontal lines sorted in ascending alphabetical order.

Centered Presents the terms in a cloud and sized by frequency.

Coloring Specifies the coloring of the terms in the Word Cloud. By default, the Coloring is set to None.

None Colors each term the same color as it is colored in the Term List.

Uniform Color Colors each term the same color. You can change this color in the Legend.

Arbitrary Grays Colors each term in varying shades of gray.

Arbitrary Colors Colors each term in various colors. You can adjust the colors in the Legend.

By column values Colors each term on a gradient color scale. The scale is based on the score for a term generated by the Score Terms by Column option. You can adjust the colors and gradient in the Legend.

Font Specifies the font, style, and size of the terms in the Word Cloud.

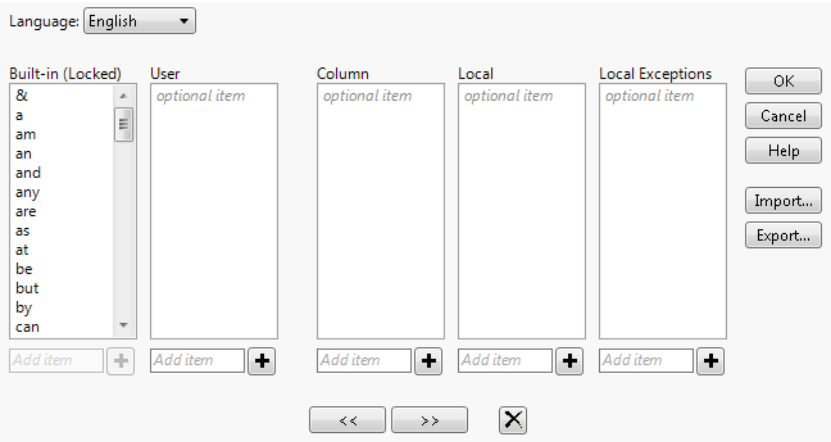
Show Legend Shows or hides the legend for the Word Cloud.

Term Options Management Windows

Stop word, recode, phrase, and stem exception information can be specified for many different scopes. They can be stored in the following locations: the Text Explorer user library (User scope), the current project, a column property for the analysis column (Column scope), or in a platform script (Local scope). You can save the local specifications and local exceptions for a specific instance of Text Explorer by saving the script for the Text Explorer report.

The Term Options management windows are four similar windows that enable you to manage the collections of stop words, recodes, phrases, and stem exceptions. Figure 12.9 shows the Manage Stop Words window. The Manage Phrases and Manage Stem Exceptions are identical to the Manage Stop Words window. The Manage Recodes window differs slightly. See "Manage Recodes".

Figure 12.9 Manage Stop Words Window



Manage Stop Words

The Manage Stop Words window contains multiple lists of stop words that represent the different scopes (or locations) of specified stop words. Below each list is a text edit box and an add button. These controls enable you to add custom stop words to each scope. You can move stop words from one scope to another by dragging them. You can copy and paste items from one list to another list. Two buttons at the bottom of the window move the selected items from one scope to the next, either left or right. The X button removes the selected items from their current scope. You can edit existing items in the lists by double-clicking on an item and changing the text.

Language Specifies the list of built-in stop words and to which language the user library selections are saved. If you select Apply Items for Language, the changes are saved to the main user library. The Language setting applies only to the Built-in, User, and Project scopes.

Built-in (Locked) Lists the built-in list of stop words for the specified language. You can exclude a built-in stop word by placing it in the Local Exceptions list.

User Lists the stop words in the user library for the specified language.

Project (Available only when Text Explorer is launched within a project.) Lists the stop words in the current project for the specified language.

Column Lists the stop words in the “Stop Words” column property for the text column.

Local Lists the stop words in the local scope. You can specify them when Text Explorer is launched via JSL. These stop words are used only in the current Text Explorer platform report.

Local Exceptions Lists words that are not treated as stop words in the current Text Explorer platform. You can specify them when Text Explorer is launched via JSL. The words listed in Local Exceptions override words listed in all of the other scopes.

Import Enables you to import stop words from a text file. The stop words are copied to the clipboard. You can paste them into any of the lists other than Built-in.

Export Enables you to export stop words to the clipboard or to a text file. An Export window appears that enables you to select the scopes for which you would like to export stop words and the location of the export.

The user library files are located in a TextExplorer directory. The location of this directory is based on your computer's operating system:

- Windows: "C:/Users/<username>/AppData/Roaming/SAS/JMP/TextExplorer/<lang>/"
- macOS: "/Users/<username>/Library/Application Support/JMP/TextExplorer/<lang>/"

The main user library files are located in the TextExplorer directory itself. These files are not language-specific.

When you click OK, changes to the User, Project, and Column lists are saved to the user library, the project, and the column properties, respectively. Anything specified in the Local and Local Exceptions lists is saved only when you save the script of the Text Explorer report.

If you are saving stop words to the user library, the file is named stopwords.txt. If you are saving to a column property, the property is called "Stop Words".

Manage Recodes

The Manage Recodes window differs slightly from the Manage Stop Words window. Instead of one text edit box below each list, there are two text edit boxes. The old value (specified in the top box) is recoded to the new value (specified in the bottom box).

If you are saving recodes to the user library, the file is named recodes.txt. If you are saving to a column property, the property is called "Recodes".

Manage Phrases

If you are saving phrases to the user library, the file is named phrases.txt. If you are saving to a column property, the property is called "Phrases".

Manage Stem Exceptions

If you are saving stem exceptions to the user library, the file is named stemExceptions.txt. If you are saving to a column property, the property is called "Stem Exceptions".

Note: The Local Exceptions list in the Manage Stem Exceptions window lists stem exceptions that are excluded from the stem exception list. The words in this list are involved in the stemming operation.

JMP PRO Manage Negation Terms

The Manage Negation Terms window is available in the Sentiment Analysis report. See "Sentiment Analysis".

If you are saving negation terms to the user library, the file is named negations.txt. If you are saving to a column property, the property is called "Negation Terms".

Note: Terms that appear in the Local or Local Exceptions lists apply only to the current Sentiment Analysis report.

JMP PRO Manage Intensifier Terms

The Manage Intensifier Terms window, which is available in the Sentiment Analysis report, differs slightly from the Manage Stop Words window. In addition to the text edit box below each list, there is a Multiplier control. The Multiplier control enables you specify an intensification multiplier for a term when you add it to the set of intensifier terms. See "Sentiment Analysis".

If you are saving intensifier terms to the user library, the file is named intensifiers.txt. If you are saving to a column property, the property is called "Intensifier Terms".

Note: Terms that appear in the Local or Local Exceptions lists apply only to the current Sentiment Analysis report.

JMP PRO Manage Sentiment Terms

The Manage Sentiment Terms window, which is available in the Sentiment Analysis report, differs slightly from the Manage Stop Words window. In addition to the text edit box below each list, there is a Score control. The Score control enables you specify a sentiment score for a term when you add it to the set of sentiment terms. See "Sentiment Analysis".

If you are saving sentiment terms to the user library, the file is named sentiments.txt. If you are saving to a column property, the property is called "Sentiment Terms".

Note: Terms that appear in the Local or Local Exceptions lists apply only to the current Sentiment Analysis report.

JMP PRO Text Analysis Options

The Text Explorer red triangle menu contains the following analysis options:

Latent Class Analysis Performs a latent class analysis on the binary weighted document term matrix using sparse matrix routines. See "Latent Class Analysis".

When you select Latent Class Analysis from the Text Explorer red triangle menu, a Specifications window appears with the following options:

Maximum Number of Terms The maximum number of terms included in the latent class analysis.

Minimum Term Frequency The minimum number of occurrences a term must have to be included in the latent class analysis.

Number of Clusters The number of clusters in the latent class analysis.

Latent Semantic Analysis, SVD Performs a partial singular value decomposition of the document term matrix. See "Latent Semantic Analysis (SVD)".

Discriminant Analysis Predicts membership of each document in a group or category based on the document term matrix. See "Discriminant Analysis".

Term Selection Analyzes which terms best explain different responses. Term Selection can also be useful for sentiment analysis when the responses are ratings. See "Term Selection".

Sentiment Analysis (Available only when the Language option is set to English.) Identifies sentiment terms in document using lexical analysis and scores documents for positive, negative, and overall sentiment. See "Sentiment Analysis".

JMP PRO Singular Value Decomposition Specifications Windows

The analysis options in the Text Explorer platform are based on the Document Term Matrix (DTM). The DTM is formed by creating a column for each term in the Term List (up to a specified Maximum Number of Terms). Each text document (equivalent to a row in the data table) corresponds to a row of the DTM. The values in the cells of the DTM depend on the type of weighting specified by the user in the Specifications window.

Figure 12.10 shows the Singular Value Decomposition Specifications window. When you select options from the Text Explorer red triangle menu that perform a singular value decomposition on the document term matrix, the Specifications window appears with the following options:

Maximum Number of Terms The maximum number of terms included in the singular value decomposition.

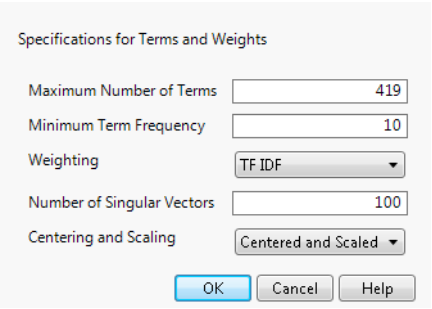
Minimum Term Frequency The minimum number of occurrences a term must have to be included in the singular value decomposition.

Weighting The weighting scheme that determines the values that go into the cells of the document term matrix. The weighting scheme options are described in "Document Term Matrix Specifications Window".

Number of Singular Vectors The number of singular vectors in the singular value decomposition. The default value is the minimum of the number of documents, the number of terms, or 100.

Centering and Scaling Options for centering and scaling of the document term matrix. You can choose between **Centered and Scaled**, **Centered**, and **Uncentered**. By default, the document term matrix is both centered and scaled.

Figure 12.10 SVD Specification Window



Save Options

The Text Explorer red triangle menu contains the following options to save information to data tables, table columns, and column properties:

Save Document Term Matrix Saves columns to the data table for each column of the document term matrix (up to a specified Maximum Number of Terms).

JMP PRO **Save Stacked DTM for Association** Saves a stacked version of the document-term matrix to a JMP data table. The stacked format is appropriate for analysis in the Association Analysis platform. See *Predictive and Specialized Modeling*. If you specify an ID variable in the Text Explorer launch window, the ID variable is used to identify the rows that each term came from in the original text data table. The stacked table also contains a table script to launch Association Analysis.

Save DTM Formula Saves a formula column with the Vector modeling type to the data table. The length of the vector depends on user-specified options for the maximum number of terms, the minimum term frequency, and the weighting. The resulting column uses the

`Text Score()` JSL function. For more information about this function, see Help > Scripting Index.

Save Term Table Creates a JMP data table that contains each term from the Term List, the number of occurrences, and the number of documents that contain each term. If you select the Score Terms by Column option after selecting Save Term Table, a column containing scores for each term is added to the data table created by the Save Term Table option.

Score Terms by Column Saves scores based on values in a specified column to the JMP data table created by the Save Term Table option. The scores for each term are the mean value of the specified column weighted by the number of occurrences of the term in each row. If you have already selected the Save Term Table option, the Score Terms by Column option adds a column containing scores to the data table created by the Save Term Table option. Otherwise, the JMP data table for the term table is created. When the specified column is not Continuous, columns containing scores for each level in the specified column are created.

Document Term Matrix Specifications Window

When you select the Save Document Term Matrix and Save DTM Formula options from the Text Explorer red triangle menu, the Document Term Matrix Specifications window appears with the following options:

Maximum Number of Terms The maximum number of terms included in the document term matrix.

Minimum Term Frequency The minimum number of occurrences a term must have to be included in the document term matrix.

Weighting The weighting scheme that determines the values that go into the cells of the document term matrix.

The following options are available for Weighting:

Binary Assigns 1 if a term occurs in each document and 0 otherwise. This is the default weighting, unless an SVD analysis has previously been run.

Ternary Assigns 2 if a term occurs more than once in each document, 1 if it occurs only once and 0 otherwise.

Frequency Assigns the count of a term's occurrence in each document.

Log Freq Assigns $log10(1 + x)$, where x is the count of a term's occurrence in each document.

TF IDF Assigns *TF * log10(nDoc / nDocTerm)*. Abbreviation for *term frequency - inverse document frequency*. This is the default weighting for an SVD analysis. The terms in the formula are defined as follows:

TF = frequency of the term in the document

nDoc = number of documents in the corpus

nDocTerm = number of documents that contain the term

Note: If you select Save Document Term Matrix or Save DTM Formula after you have run an SVD analysis, the Specifications window contains the specifications from the most recent SVD analysis.

Report Options

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Latent Class Analysis

In the Text Explorer platform, latent class analysis enables you to group the documents from the corpus into clusters of similar documents. The Latent Class Analysis report contains the model specifications, the Bayesian Information Criterion (BIC) value for the model and a Show Text button. If one or more clusters in the Cluster Mixture Probabilities table is selected, the Show Text button opens a window that contains the text of the documents that are deemed most likely to belong to the selected cluster.

The Latent Class Analysis red triangle menu contains the following options:

Display Options Specifies the contents of the Latent Class Analysis report. By default, all of the report options are shown except for the word clouds for each cluster.

Cluster Mixture Probabilities Shows or hides a table of the probability of an observation belonging to each cluster.

Tip: You can select one or more rows in the Mixture Probabilities by Cluster table to select the observations assigned to the corresponding clusters.

Term Probabilities by Cluster Shows or hides a table of terms with an estimate for each cluster of the conditional probability that a document contains the term, given that the document belongs to a particular cluster. By default, the terms in this table are sorted by descending frequency in the corpus.

The Cluster Most Characteristic column shows the cluster that the term occurs in at the highest rate.

The Cluster Most Probable column shows the cluster in which a randomly chosen document that contains the term is most likely to be found.

Top Terms by Cluster Show or hides a table of the ten terms with the highest scores in each cluster. The score $S_{t,c}$ for term t in cluster c is calculated as follows:

$$S_{t,c} = 100 \bullet \text{mean}(p_t) \bullet \log 10\left(\frac{p_{t,c}}{\text{mean}(p_t)}\right)$$

where mean(p_t) is the mean of the term probabilities by cluster for term t and $p_{t,c}$ is the term probability by cluster for term t in cluster c.

MDS Plot Shows or hides a multidimensional scaling plot, which is a two-dimensional representation of the proximity of the clusters. For more information about MDS plots, see *Multivariate Methods*. The Show Text button opens a window that contains the text of the selected documents.

Cluster Probabilities by Row Shows or hides the Mixture Probabilities table, which displays probabilities of cluster membership for each row. The Most Likely Cluster column indicates the cluster with the highest probability of membership for each row.

Word Clouds by Cluster Shows or hides a matrix of word clouds, one for each cluster.

Rename Clusters Enables you to add descriptive names for one or more of the clusters.

Save Probabilities Saves the values in the Mixture Probabilities table to the corresponding rows in the data table.

Save Probability Formulas Saves a formula column to the data table for each cluster as well as a formula column for the most likely cluster.

The score formula that is saved uses the `Text Score()` JSL function with the weighting argument set to "LCA".

Color by Cluster Colors each row in the data table according to its most likely cluster.

Remove Removes the Latent Class Analysis report from the Text Explorer report.

For more information about latent class analysis, see *Multivariate Methods*.

Note: The LCA algorithm that is used in the Text Explorer platform takes advantage of the sparsity of the document term matrix. For this reason, the LCA results in the Text Explorer platform do not exactly match the results in the Latent Class Analysis platform.

JMP PRO Latent Semantic Analysis (SVD)

In the Text Explorer platform, latent semantic analysis is centered around computing a partial singular value decomposition (SVD) of the document term matrix (DTM). This decomposition reduces the text data into a manageable number of dimensions for analysis. Latent semantic analysis is equivalent to performing principal components analysis (PCA).

The partial singular value decomposition approximates the DTM using three matrices: **U**, **S**, and **V**′. The relationship between these matrices is defined as follows:

$$DTM \approx \mathbf{U} * \mathbf{S} * \mathbf{V}'$$

Define *nDoc* as the number of documents (rows) in the DTM, *nTerm* as the number of terms (columns) in the DTM, and *nVec* as the specified number of singular vectors. Note that *nVec* must be less than or equal to min(*nDoc*, *nTerm*). It follows that **U** is an *nDoc* by *nVec* matrix that contains the left singular vectors of the DTM. **S** is a diagonal matrix of dimension *nVec*. The diagonal entries in **S** are the singular values of the DTM. **V**′ is an *nVec* by *nTerm* matrix. The rows in **V**′ (or columns in **V**) are the right singular vectors.

The right singular vectors capture connections among different terms with similar meanings or topic areas. If three terms tend to appear in the same documents, the SVD is likely to produce a singular vector in **V**′ with large values for those three terms. The **U** singular vectors represent the documents projected into this new term space.

Latent semantic analysis also captures indirect connections. If two words never appear together in the same document, but they generally appear in documents with another third word, the SVD is able to capture some of that connection. If two documents have no words in common but contain words that are connected in the dimension-reduced space, they map to similar vectors in the SVD output.

The SVD transforms text data into a fixed-dimensional vector space, making it amenable to all types of clustering, classification, and regression techniques. The Save options enable you to export this vector space to be analyzed in other JMP platforms.

The DTM, by default, is centered, scaled, and divided by *nDoc* minus 1 before the singular value decomposition is carried out. This analysis is equivalent to a PCA of the correlation matrix of the DTM.

You can also specify Centered or Uncentered in the Specifications window.

- If you specify Centered, the DTM is centered and divided by *nDoc* minus 1 before the singular value decomposition. This analysis is equivalent to a PCA of the covariance matrix of the DTM.
- If you specify Uncentered, the DTM is divided by *nDoc* before the singular value decomposition. This analysis is equivalent to a PCA of the unscaled DTM.

The SVD implementation takes advantage of the sparsity of the DTM even when the DTM is centered.

JMP PRO SVD Report

The Latent Semantic Analysis option in the Text Explorer platform produces two SVD plots and a table of the singular values from the singular value decomposition.

JMP PRO SVD Plots

The first plot contains a point for each document. For a given document, the point that is plotted is defined by the document's values in the first two singular vectors (the first two columns of the **U** matrix) multiplied by the diagonal singular values matrix (**S**). This plot is equivalent to the Score Plot in the Principal Components platform. Each point in this plot represents a document (row of the data table). You can select the points in this plot to select the corresponding rows in the data table.

The second plot contains a point for each term. For a given term, the point that is plotted is defined by the term's values in the first two singular vectors (the first two rows of the **V'** matrix) multiplied by the diagonal singular values matrix (**S**). This plot is equivalent to the Loadings Plot in the Principal Components platform. In this plot, the points correspond to rows in the Term List table.

Above each of the SVD Plots, you can click a Show Text button to open a window that contains the text of the selected points in the plot.

JMP PRO Singular Values

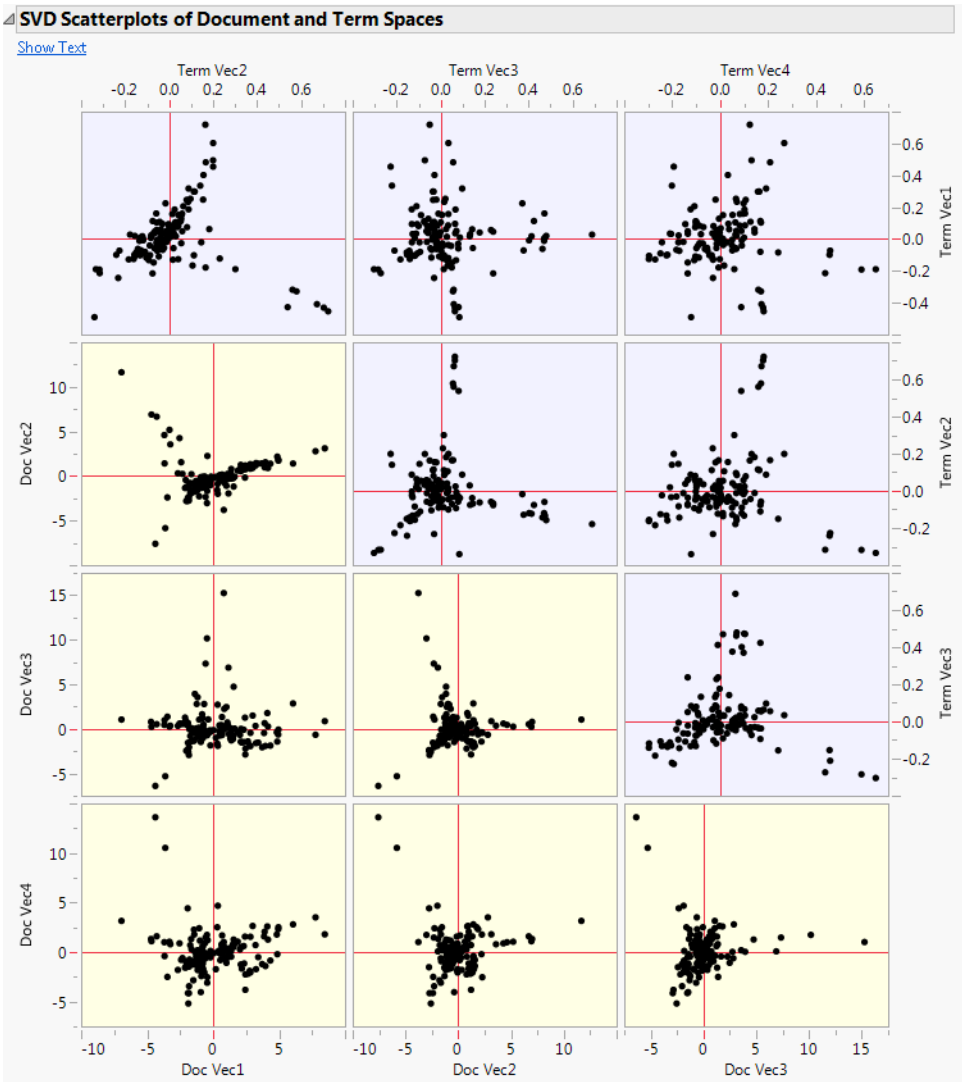
Below the document and term SVD plots, a table of the singular values appears. These are the diagonal entries of the **S** matrix in the singular value decomposition of the document term matrix. The Singular Values table also contains a column of corresponding eigenvalues for the equivalent principal components analysis. Like in the Principal Components platform, there are columns for the percent and cumulative percent of variation explained by each eigenvalue (or singular value). You can use the Cum Percent column to decide what percent of variance from the DTM you want to preserve, and then use the corresponding number of singular vectors.

JMP PRO SVD Report Options

The SVD red triangle menu in the Text Explorer platform contains the following options:

SVD Scatterplot Matrix Shows or hides a scatterplot matrix of the term and document singular value decomposition vectors. You are prompted to select the size of the scatterplot matrix when you select this option. This scatterplot matrix enables you to visualize more than the first two dimensions of the singular value decomposition. The Show Text button opens a window that contains the text of the selected documents.

Figure 12.11 SVD Scatterplots of Document and Term Spaces



Topic Analysis, Rotated SVD Performs a varimax rotated partial singular value decomposition of the document term matrix to produce groups of terms called topics. You can select this option multiple times to find different numbers of topics. See "Topic Analysis".

Cluster Terms Shows or hides a hierarchical clustering analysis of the terms in the data. To the right of the dendrogram, there are options to set the number of clusters and save the clusters to a data table. For each term, this data table contains its frequency, the number of documents that contain it, and its assigned cluster. For more information about hierarchical clustering and dendrograms, see *Multivariate Methods*.

Cluster Documents Shows or hides a hierarchical clustering analysis of the documents in the data. To the right of the dendrogram, there are options to do the following: set the number of clusters, save the clusters to a column in the data table, and show the documents in a selected branch of the dendrogram plot.

Select Near Neighbors Finds and selects the nearest neighbors of the selected points in the document SVD plot. The algorithm uses the **U** matrix from the singular value decomposition to find the nearest neighbors. When you select this option, you must specify the number of nearest neighbors to select. By default, the option selects the 10 nearest neighbors.

Save Document Singular Vectors Saves a user-specified number of singular vectors from the document singular value decomposition as columns to the data table. The first two saved columns represent the points plotted in the document SVD plot. See "Latent Semantic Analysis (SVD)".

Save Singular Vector Formula Saves a formula column with the Vector modeling type that contains the document singular value decomposition to the data table. The resulting column uses the `Text Score()` JSL function. For more information about this function, see Help > Scripting Index.

Save Term Singular Vectors Saves a user-specified number of singular vectors from the terms singular value decomposition as columns to a new data table where each row corresponds to a term. If a Term Table data table is already open, this option saves the columns to that data table. The first two saved columns represent the points plotted in the term SVD plot. See "Latent Semantic Analysis (SVD)".

Remove Removes the SVD report from the Text Explorer report window.

JMP PRO Topic Analysis

The Topic Analysis, Rotated SVD option performs a varimax rotation on the partial singular value decomposition (SVD) of the document term matrix (DTM). You must specify a number of rotated singular vectors, which corresponds to the number of topics that you want to retain from the DTM. After you specify a number of topics, the Topic Analysis report appears.

Topic analysis is equivalent to a rotated principal component analysis (PCA). The varimax rotation takes a set of singular vectors and rotates them to make them point more directly in the coordinate directions (toward the terms). This rotation makes the vectors help explain the text as each rotated vector orients toward a set of terms. Negative values indicate a repulsion force. The terms with negative values occur in a topic less frequently compared to the terms with positive values.

JMP PRO Topic Analysis Report

The Topic Analysis report shows the terms that have the largest loadings in each topic after rotation. There are additional reports that show the components of the rotated singular value decomposition.

The Top Loadings by Topic report shows a table of terms for each topic. The terms in each table are the ones that have the largest loadings in absolute value for each topic. Each table is sorted in descending order by the absolute value of the loading. These tables can be used to determine conceptual themes that correspond to each topic.

The Topic Analysis report also contains the following reports:

Topic Loadings Contains a matrix of the loadings across topics for each term. This matrix is equivalent to the factor loading matrix in a rotated PCA.

Word Clouds by Topic Contains a matrix of word clouds, one for each topic.

Topic Scores Contains a matrix of document scores for each topic. Documents with higher scores in a topic are more likely to be associated with that topic.

Topic Scores Plots Contains a Show Text button and a plot of topic scores for each document. The Show Text button opens a window that contains the text of the selected documents.

The Topic Scores Plots report is a visual representation of the matrix in the Topic Scores report. Each panel in the plot corresponds to one of the topics, or one of the columns of the Topic Scores matrix. Within each panel, each point corresponds to one of the documents in the corpus, or one of the rows of the Topic Scores matrix.

Variance Explained by Each Topic Contains a table of the variance explained by each topic. The table also contains columns for the percent and cumulative percent of the variation explained by each topic.

Rotation Matrix Contains the rotation matrix for the varimax rotation.

JMP PRO Topic Analysis Report Options

The Topic Analysis red triangle menu in the Text Explorer platform contains the following options:

Topic Scatterplot Matrix Shows or hides a scatterplot matrix of the rotated singular value decomposition vectors. The Show Text button opens a window that contains the text of the selected documents.

Display Options Contains options to show or hide content that appears in the Topic Analysis report. See "Topic Analysis Report".

Rename Topics Enables you to add descriptive names for one or more of the topics.

Save Document Topic Vectors Saves a user-specified number of singular vectors from the rotated singular value decomposition as columns to the data table.

Save Topic Vector Formula Saves a formula column with the Vector modeling type that contains the rotated singular value decomposition to the data table. The resulting column uses the `Text Score()` JSL function. For more information about this function, see Help > Scripting Index.

Save Term Topic Vectors Saves the topic vectors as columns to the data table created by the Save Term Table option.

Remove Removes the Topic Analysis report from the SVD report.

JMP PRO Discriminant Analysis

Discriminant analysis predicts membership of each document in a group or category based on the columns in the document term matrix (DTM). Specifically, discriminant analysis predicts a classification of each document into a category of a response column. When you select the Discriminant Analysis option, you must select a response column that contains categories or groups. Group membership is predicted by the columns of the DTM. For more information about discriminant analysis, see *Multivariate Methods*.

The discriminant analysis method in the Text Explorer platform is based on a singular value decomposition of the centered DTM. Each group of the response column has its own group mean that is used to center the DTM. The discriminant analysis method in the Text Explorer platform is faster than the Discriminant Analysis platform because it takes advantage of the sparsity of the DTM.

JMP PRO Discriminant Analysis Specifications Window

The Discriminant Analysis option in the Text Explorer platform is based on the Document Term Matrix (DTM). The DTM is formed by creating a column for each term in the Term List (up to a specified Maximum Number of Terms). Each text document (equivalent to a row in the data table) corresponds to a row of the DTM. The values in the cells of the DTM depend on the type of weighting specified by the user in the Specifications window.

When you select the Discriminant Analysis option from the Text Explorer red triangle menu, the Specifications window appears with the following options:

Maximum Number of Terms The maximum number of terms included in the discriminant analysis.

Minimum Term Frequency The minimum number of occurrences a term must have to be included in the discriminant analysis.

Weighting The weighting scheme that determines the values that go into the cells of the document term matrix. The weighting scheme options are described in "Document Term Matrix Specifications Window".

Number of Singular Vectors The number of singular vectors in the discriminant analysis. The default value is the minimum of the number of documents, the number of terms, or 100.

JMP PRO Discriminant Analysis Report

By default, the Discriminant Analysis report in the Text Explorer platform contains two open reports: the Classification Summary and the Discriminant Scores. The other reports are initially closed.

The Discriminant Analysis report contains the following reports:

Term Means Provides a table of the terms used in the discriminant analysis. The terms correspond to the columns of the DTM. The table contains the means in each group for each term, as well as the overall mean and weighted standard deviation for each term.

Squared Distances to Each Group Provides a table that contains the squared Mahalanobis distances to each group for each document. For more information about Mahalanobis distances, see *Multivariate Methods*.

Probabilities to Each Group Provides a table that contains the probability that a document belongs to each group.

Classification Summary Provides a report that summarizes the discriminant scores. This report corresponds to the Score Summaries report in the Discriminant Analysis platform report.

Discriminant Scores Provides a table of the predicted classification of each document and other supporting information. This table corresponds to the Discriminant Scores table in the Discriminant Analysis platform report.

JMP PRO Discriminant Analysis Report Options

The Discriminant Analysis red triangle menu contains the following options:

Canonical Plot Shows or hides a plot of the documents and group means in canonical space. Canonical space is the space that most separates the groups. If there are more than two levels of the response variable, you must specify the number of canonical coordinates. If you specify more than two canonical coordinates, this option produces a matrix of canonical plots.

Save Probabilities Saves a probability column to the data table for each response level as well as a column that contains the most likely response. The Most Likely response column contains the level with the highest probability based on the model.

Each probability column gives the posterior probability of an observation's membership in that level of the response. The Response Probability column property is saved to each probability column. For more information about the Response Probability column property, see *Using JMP*.

Save Probability Formulas Saves formula columns to the data table for the prediction of the most likely response. The first saved column contains a formula that uses the `Text Score()` function to calculate the probability for each response level. There are also columns that contain probabilities for each response level as well as a column that contains the predicted response.

Save Canonical Scores Saves columns to the data table that contain the scores from canonical space for each observation. Canonical space is the space that most separates the groups. The column for the k^{th} canonical score is named Canonical<k>.

Remove Removes the Discriminant Analysis report from the Text Explorer report window.

JMP PRO Term Selection

Term selection identifies which terms best explain different responses. The analysis uses the Generalized Regression platform to perform variable selection on the document term matrix (DTM) and to identify terms that most impact the response. Term selection can be used with binary responses, similar to sentiment analysis, as well as other types of responses. The fitted model uses an appropriate response distribution for the specified response column.

Tip: For an example of Term Selection, select **Help > Sample Data Library**, open Chips.jmp, and run the Text Explorer - Term Selection table script.

JMP PRO Term Selection Settings

The Settings report enables you to select a response column, specify the target level of the response, and adjust the settings for the model. When you have specified the model settings, click the Run button to run the model. The fitted model then appears in the Summary report. See "Term Selection Summary Report".

Target Level

After you choose a response column, the Target Level outline appears.

- For nominal responses, choose one level of the response to be the target level in a logistic regression model; the response in the logistic regression model is the target level versus all of the other levels combined.
- For ordinal responses, all response levels are initially included in the model. Using the local data filter, you can select levels of the response to be excluded from the model; the underlying numeric values of the included levels are modeled with a normal response distribution.

  Note: For ordinal responses, the term selection model can be fit only when the data type of the response column is numeric.

- For continuous responses, use the local data filter histogram to select values of the response to be excluded from the model; the included values are modeled with a normal response distribution.
- For response columns with the Multiple Response modeling type, choose one or more levels of the response to be the target level in a binary logistic regression model. If you choose more than one level, a document belongs to the target level if any of the levels are present in the response column for that document. Select the **Combine with AND** option to require that all selected levels are present in a document's response column for that document to be included in the target level.

Model Settings

By default, the Generalized Regression model uses the Elastic Net estimation method with early stopping and the AICc validation method. You can change these settings in the Model Settings outline. See *Fitting Linear Models*.

Note: If a Validation column is specified in the Text Explorer launch window, the Generalized Regression platform in the Term Selection report uses the Validation column as the Validation method.

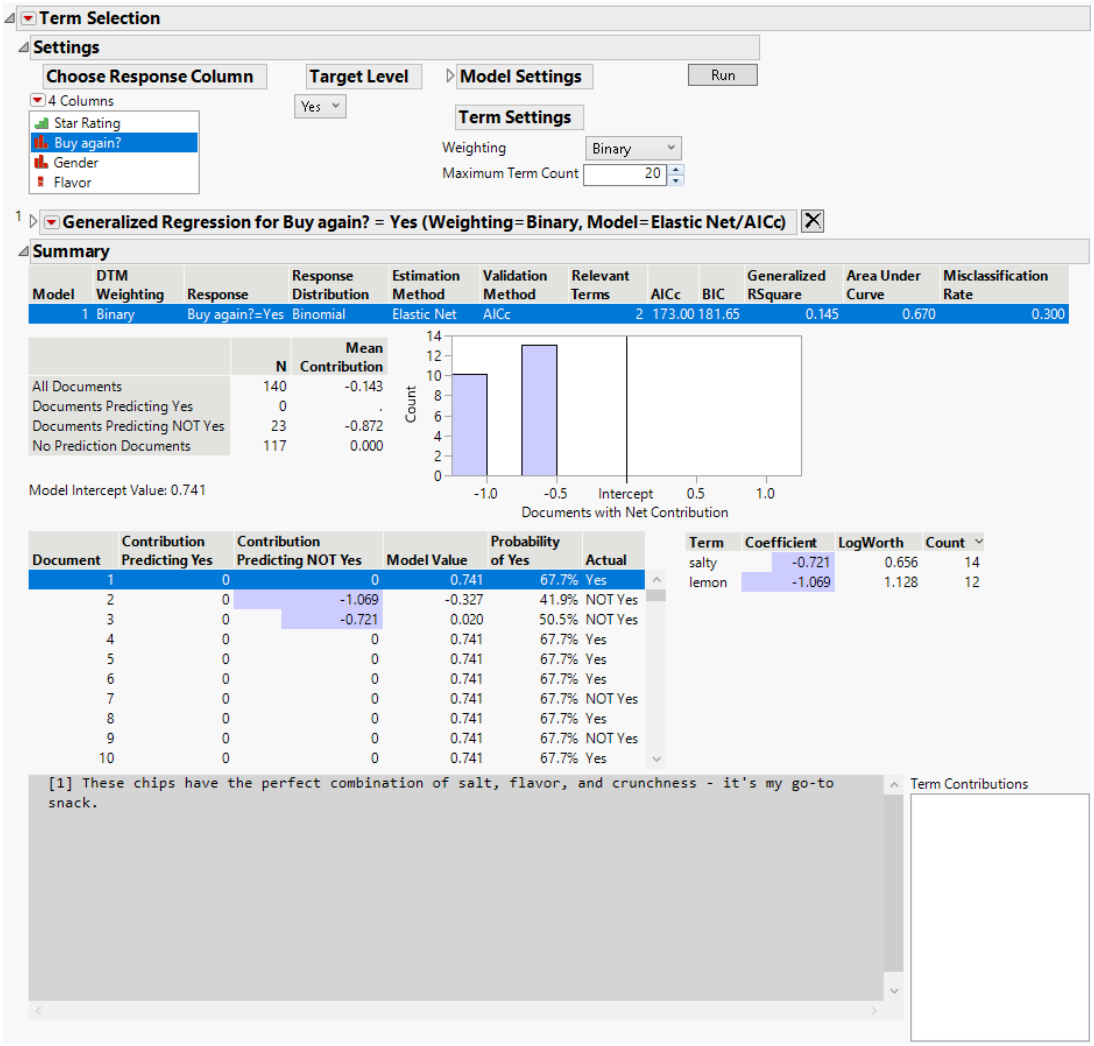
Term Settings

The Term Settings define the document term matrix (DTM) that is used in the regression model. You can change the weighting technique as well as the maximum number of terms included in the DTM; each term corresponds to a column of the DTM. Note that terms that have fewer than 10 occurrences in the corpus are not included in the DTM used by the model. For more information about the DTM options, see "Document Term Matrix Specifications Window".

JMP PRO Term Selection Report

After you run an analysis, the Term Selection report consists of three sections. The Settings report contains controls for specifying an analysis. See "Term Selection Settings". Below the Settings report, there are initially closed Generalized Regression reports for each analysis that you have run. See *Fitting Linear Models*. The last section of the report is the Summary report.

Figure 12.12 Term Selection Report



Term Selection Summary Report

The Summary report contains a Model Comparison table, a Summary table and histogram, a Document Scores table, a Term Scores table, and a text box.

The Model Comparison table contains a row for each fitted model. The rest of the Summary report shows results from the currently selected model in this table.

The Summary table shows counts and mean scores for the documents, overall and by the predicted value of the response from the model. The Mean Contribution is the average of the contribution values in the Document Scores table. The Summary histogram shows the distribution of the overall contribution values of the documents. The histogram is interactive, so you can click on a bar to highlight the corresponding documents in the Document Scores table.

The Document Scores table shows the positive and negative contribution values for each document, as well as predicted and actual values for each document. For binomial response models, the predicted values are probabilities of the document being in the target level; for normal response models, the predicted values are the predictions from the fitted model for each document. If you select a row of the table, the text of the corresponding document appears in the text box below the table.

The Term Scores table lists each term that was selected by the fitted model, its coefficient from the model, its LogWorth, and the count of occurrences of the term in the corpus. If you select a row of the table, the text of the corresponding document appears in the text box below the table.

The text box shows the text of documents that are selected in the Document Scores table or the context of terms that are selected in the Term Scores table.

JMP PRO Term Selection Report Options

The Term Selection red triangle menu contains the following options:

Save Document Scores (Available only when an analysis is selected in the Summary table.) Saves the columns from the Document Scores table to new columns in the data table. The new columns contain the positive and negative contributions, as well as the predicted value for each document.

Save Term Score DTM (Available only when an analysis is selected in the Summary table.) Saves columns to the data table for each relevant term in the currently selected analysis. The columns contain the term scores for each document, using the Weighting specified in the Term Selection Term Settings.

Save Prediction Formulas (Available only when an analysis is selected in the Summary table.) Saves columns to the data table that contain the prediction formulas for the currently selected analysis.

Show Term Cloud Shows or hides a word cloud in the Summary report. The word cloud shows the coefficient terms in the currently selected analysis. The words are sized by the absolute value of their coefficients and colored by the sign of their coefficients.

Remove Removes the Term Selection report from the Text Explorer report window.

JMP PRO Sentiment Analysis

Sentiment analysis identifies sentiment terms in documents using lexical analysis and scores documents for positive, negative, and overall sentiment. The analysis assumes that each document is free text with binary sentiment on a single topic. Sentiment Analysis incorporates basic natural language processing (NLP) into the results. For more information about natural language processing, see https://opennlp.apache.org/. If you prefer to not use NLP, deselect the Parse Documents option.

Tip: For an example of Sentiment Analysis, select **Help > Sample Data Library**, open Chips.jmp, and run the Text Explorer - Sentiment Analysis table script.

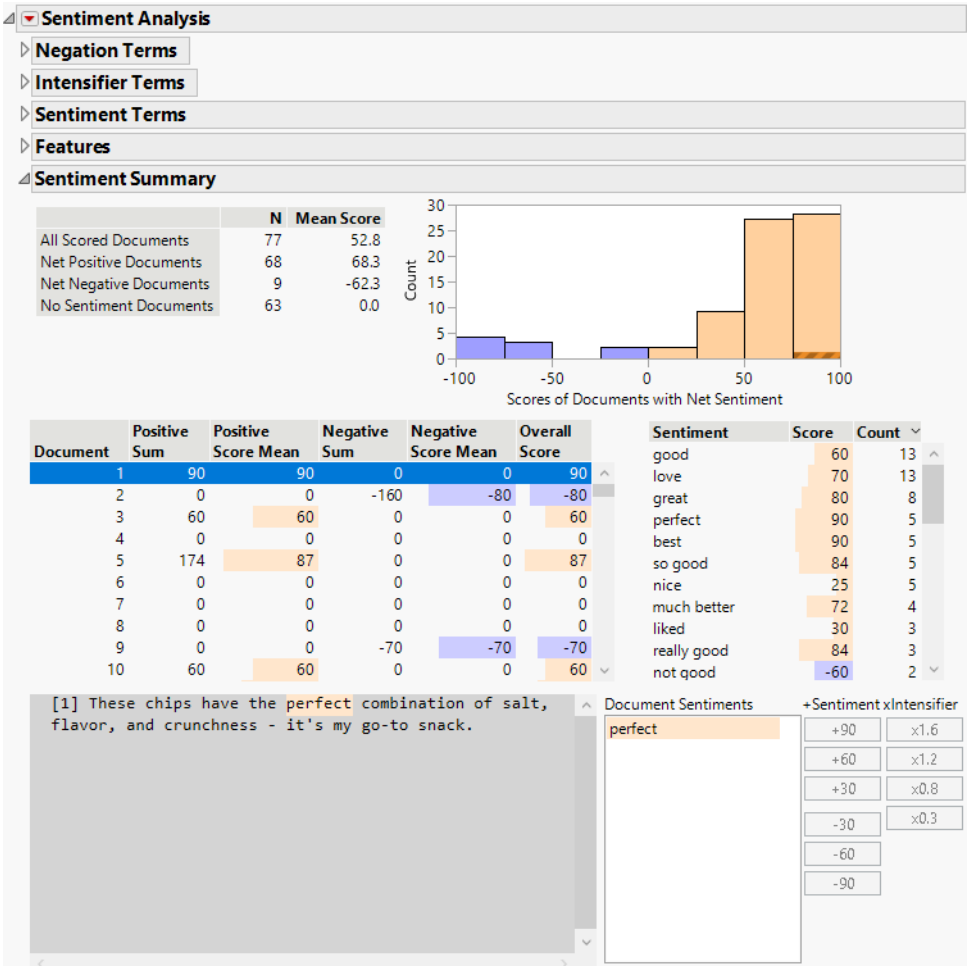
Notes:

- In Sentiment Analysis, a word can take effect in only one class of terms: negation, intensifier, or sentiment.
- Sentiment Analysis recognizes some emoticons, or sequences of characters that are treated as a single unit. You can see the built-in emoticons and their default sentiment scores in the Sentiment Terms report or in the Manage Sentiment Terms window.
- If you specify a word as a negation, intensifier, or sentiment term and it is already specified as a stop word, the word is temporarily removed as a stop word as long as the Sentiment Analysis report is open. This temporary removal affects the entire Text Explorer report. It is restored as a stop word when the Sentiment Analysis report closes.

JMP PRO Sentiment Analysis Report

By default, the Sentiment Analysis report in the Text Explorer platform contains one open report: the Sentiment Summary. The other reports are initially closed.

Figure 12.13 Sentiment Analysis Report



The Sentiment Analysis report contains the following reports:

Negation Terms

Contains a list of the negation terms in the current sentiment analysis. Right-click the list to see a menu of additional options. You can select terms in the list to remove them.

Intensifier Terms

Contains a list of the intensifier terms and their corresponding multiplier values. Right-click the list to see a menu of additional options. You can select terms in the list to remove them.

Sentiment Terms

Contains a list of the sentiment terms and their corresponding score values. This report also enables you to add new sentiment terms. The Possible Sentiment table contains counts of terms that you might consider adding as sentiment terms. To add a term as a sentiment term, select it in the Possible Sentiment table and click one of the buttons below +Sentiment. To choose a sentiment score value that is not listed, you can edit the score value after you have added it to the list of sentiment terms.

When you select a term in the Possible Sentiment table, documents that contain that term appear on the right side of the Sentiment Terms report. This provides context for how the term is used in the corpus.

Features

Contains options to score features in the corpus. A feature is something that is being described by the sentiment terms. Click the **Search** button to generate a list of possible feature terms. When you select one or more terms from the Possible Feature table, excerpts of the documents that contain those terms appear in the text box to the right of the table. Click the **Score Selected Features** button to update the Sentiment Summary report to show the results of scoring the selected feature terms.

Note: If you select the Parse Documents option, the Features report scores words when they occur within the same umbrella clause as a sentiment.

Sentiment Summary

Contains the results of the sentiment analysis based on the current settings. This report contains a Summary table and histogram, a Document Scores table, a Sentiment Terms table, a text box, and a control panel that enables you to add more sentiment and intensifier terms.

The Summary table shows counts and mean scores for the documents, broken down by how the documents were scored. The Mean Score is determined by the setting of the Scoring option. See "Sentiment Analysis Report Options". The Summary histogram shows the distribution of the overall sentiment scores of the documents. The histogram is interactive, so you can click on a bar to highlight the corresponding documents in the Document Scores table.

The Document Scores table shows the positive and negative sentiment score sums and means, as well as the overall sentiment score for each document. If you select a row of the table, the text of the corresponding document appears in the text box below the table. If you specify a Score Column, the table contains the values from the scoring column.

Tip: You can hover over the cells in the Document Scores table to see the scoring calculations that were used to produce the table results.

The Sentiment Terms table lists each sentiment term, its score value, and the count of occurrences of the term in the corpus.

Tip: For sentiment terms with multiple words, you can hover over the cells in the Score column to see the calculations that were used to produce the score.

The text box shows the text of documents that are selected in the Document Scores table or the context of terms that are selected in the Sentiment Terms table. When you select a document in the Document Scores table, a list of the sentiments in that document appears to the right of the text box.

Tip: When you hover over a term in the text box that is classified as a negation, intensifier, or sentiment term, a box appears that shows the classification and contains a Remove button. Click the **Remove** button to quickly remove that term from the list of negation, intensifier, or sentiment terms.

The control panel activates when you select a term in the text box. To add a term as a sentiment term, select it in the text box and click one of the buttons below +Sentiment. To add a term as an intensifier term, select it in the text box and click one of the buttons below ×Intensifier.

JMP PRO Sentiment Analysis Report Options

The Sentiment Analysis red triangle menu contains the following options:

Scoring Contains the following options for calculating the Overall Scores for documents:

- **Scaled** Scores of positive and negative phrases are summed. The sum is then divided by the number of phrases in the document to determine the Overall Score.
- **Min Max** The Overall Score is calculated as the sum of the maximum positive score and the minimum negative score.

Score Column Specifies a data table column that contains known information that can be compared with the calculated sentiment. The score column is added to the Document Scores table.

Tip: You can visually compare the Overall Score column and the score column to assess the sentiment scoring.

Parse Documents Specifies if natural language processing (NLP) is used to parse the documents. For more information about natural language processing, see https://opennlp.apache.org/.

Save Document Scores Saves the columns from the Document Scores table to new columns in the data table. The new columns contain the positive and negative sentiment score sums and means, as well as the overall sentiment score for each document.

Save Count of Sentiment Scores by Document Saves a column to the data table for each sentiment term. Each column contains counts of the occurrences of each sentiment term in each document.

Show Negation Terms Shows or hides the Negation Terms report.

Show Intensifier Terms Shows or hides the Intensifier Terms report.

Show Sentiment Terms Shows or hides the Sentiment Terms report.

Show Feature Finder Shows or hides the Features report.

Show Sentiment Cloud Shows or hides a word cloud of the sentiment terms in the Sentiment Summary report.

Include Builtin Negation Terms Specifies if the negation terms used in the sentiment analysis include built-in negation terms or not.

Include Builtin Intensifier Terms Specifies if the intensifier terms used in the sentiment analysis include built-in intensifier terms or not.

Include Builtin Sentiment Terms Specifies if the sentiment terms used in the sentiment analysis include built-in sentiment terms or not.

Manage Negation Terms Shows a window that enables you to add or remove negation terms. The changes made can be applied at the User, Column, and Local levels. You can also specify Local Exceptions that exclude negation terms that are specified in any of the other levels. See “Term Options Management Windows”.

Manage Intensifier Terms Shows a window that enables you to add or remove intensifier terms. The changes made can be applied at the User, Column, and Local levels. You can also specify Local Exceptions that exclude intensifier terms that are specified in any of the other levels. See “Term Options Management Windows”.

Manage Sentiment Terms Shows a window that enables you to add or remove sentiment terms. The changes made can be applied at the User, Column, and Local levels. You can

also specify Local Exceptions that exclude sentiment terms that are specified in any of the other levels. See "Term Options Management Windows".

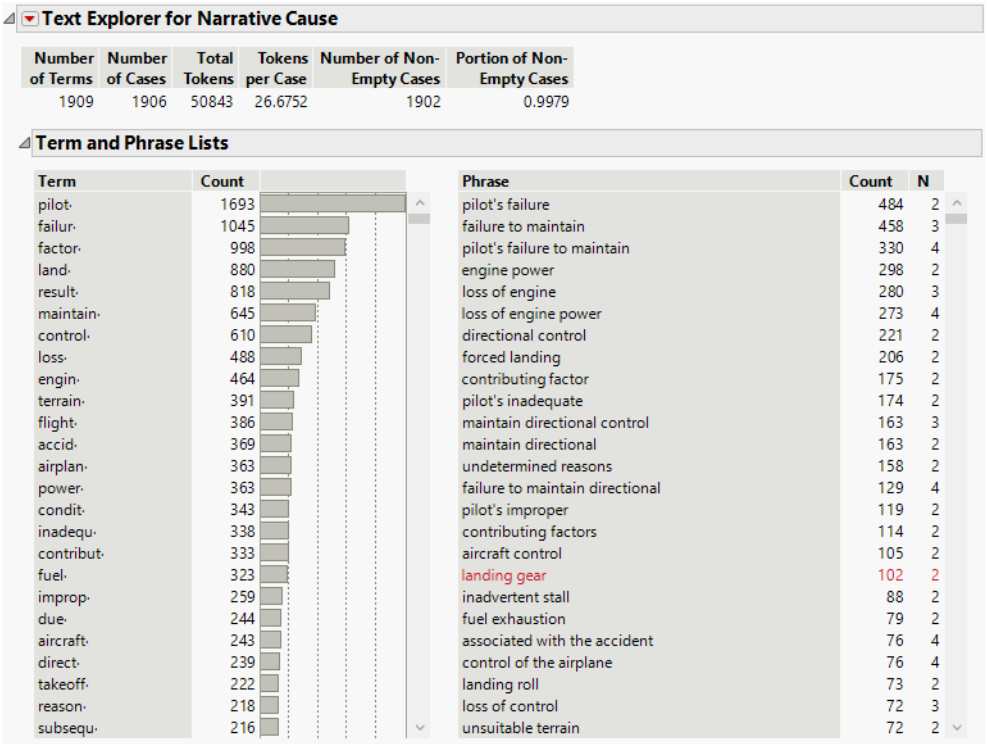
JMP PRO Additional Example of the Text Explorer Platform

This example looks at aircraft incident reports from the National Transportation Safety Board for events occurring in 2001 in the United States. You want to explore the text that contains a description of the results of the investigation into the cause of each incident. You also want to find themes in the collection of incident reports.

1. Select **Help > Sample Data Library** and open Aircraft Incidents.jmp.
2. Select **Rows > Color or Mark by Column**.
3. Select Fatal from the columns list and click **OK**.

   The rows that contain accidents involving fatalities are colored red.
4. Select **Analyze > Text Explorer**.
5. Select Narrative Cause from the Select Columns list and click **Text Columns**.
6. From the Language list, select **English**.
7. From the Stemming list, select **Stem All Terms**.
8. From the Tokenizing list, select **Basic Words**.
9. Click **OK**.

Figure 12.14 Text Explorer Report for Narrative Cause



Text Explorer for Narrative Cause

Number of Terms	Number of Cases	Total Tokens	Tokens per Case	Number of Non-Empty Cases	Portion of Non-Empty Cases
1909	1906	50843	26.6752	1902	0.9979

Term and Phrase Lists

Term	Count
pilot·	1693
failur·	1045
factor·	998
land·	880
result·	818
maintain·	645
control·	610
loss·	488
engin·	464
terrain·	391
flight·	386
accid·	369
airplan·	363
power·	363
condit·	343
inadequ·	338
contribut·	333
fuel·	323
improp·	259
due·	244
aircraft·	243
direct·	239
takeoff·	222
reason·	218
subsequ·	216

Phrase	Count	N
pilot's failure	484	2
failure to maintain	458	3
pilot's failure to maintain	330	4
engine power	298	2
loss of engine	280	3
loss of engine power	273	4
directional control	221	2
forced landing	206	2
contributing factor	175	2
pilot's inadequate	174	2
maintain directional control	163	3
maintain directional	163	2
undetermined reasons	158	2
failure to maintain directional	129	4
pilot's improper	119	2
contributing factors	114	2
aircraft control	105	2
landing gear	102	2
inadvertent stall	88	2
fuel exhaustion	79	2
associated with the accident	76	4
control of the airplane	76	4
landing roll	73	2
loss of control	72	3
unsuitable terrain	72	2

From the report, you see that there are almost 51,000 tokens and about 1,900 unique terms.

10. Right-click pilot· in the Term List and select **Select Rows**.

    From the number of selected rows in the data table, you see that some form of the word "pilot" occurs in more than 1,300 of the incident reports.

11. Right-click pilot· and select **Add Stop Word**.

    Because some form of the word "pilot" occurs frequently compared to other terms, these terms do not provide much information to differentiate among documents. All of the terms that stem to pilot· are added to the stop word list.

JMP PRO *The remaining steps of this example can be completed only in JMP Pro.*

12. JMP PRO Click the red triangle next to Text Explorer for Narrative Cause and select **Latent Semantic Analysis, SVD**.

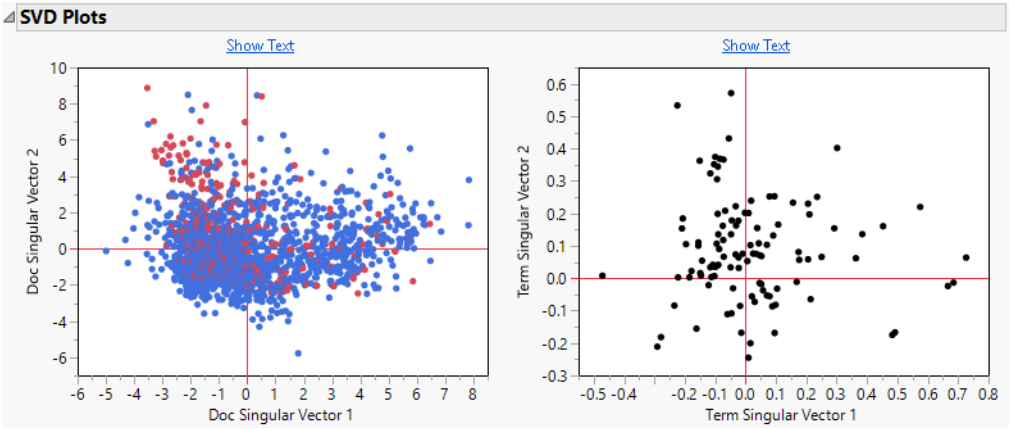
    This is the first analysis step toward topic analysis, which performs a rotation of the SVD.

13. JMP PRO In the Specifications window, type 50 for Minimum Term Frequency.

    Because there are approximately 51,000 tokens, this frequency is equivalent to a term that represents at least 0.1% of all the terms.

14. JMP PRO Click **OK**.

Figure 12.15 SVD Plots for Narrative Cause



There is not a lot of difference in the document SVD plot between fatal and non-fatal incidents.

15. JMP PRO Click the red triangle next to SVD Centered and Scaled TF IDF and select **Topic Analysis, Rotated SVD**.

    You want to look for groups of terms that form topics.

16. JMP PRO Type 5 for Number of Topics.
17. JMP PRO Click **OK**.

Figure 12.16 Top Loadings by Topics for Narrative Cause

Top Loadings by Topic

Topic 1		Topic 2		Topic 3		Topic 4		Topic 5	
Term	Loading	Term	Loading	Term	Loading	Term	Loading	Term	Loading
power·	0.67567	altitud·	0.48052	factor·	0.5093	control·	0.5125	fuel·	0.4864
loss·	0.66539	low·	0.45137	condit·	0.4677	direct·	0.4957	personnel·	0.4630
forc·	0.62046	dark·	0.42289	unsuit·	0.3984	experi·	0.4382	mainten·	0.4546
engin·	0.61866	night·	0.40408	accid·	0.3909	student·	0.4273	result·	0.4153
suitabl·	0.58926	maintain·	0.39283	select·	0.3842	lack·	0.3625	preflight·	0.3930
lack·	0.53292	instrument·	0.39239	associ·	0.3806	maintain·	0.3616	exhaust·	0.3663
reason·	0.47828	clearanc·	0.37881	area·	0.3628	instructor·	0.3559	inspect·	0.3640
undetermin·	0.46599	airspe·	0.33612	compens·	0.3352	supervis·	0.3282	plan·	0.3474
terrain·	0.37013	condit·	0.33473	failur·	-0.3207	power·	-0.3269	reason·	-0.3468
total·	0.31932	stall·	0.33010	wind·	0.3202	failur·	0.3171	undetermin·	-0.3438
land·	0.29402	flight·	0.32838	inadequ·	0.3080	reason·	-0.3142	improp·	0.3389
		continu·	0.31729	result·	-0.2845	undetermin·	-0.3101	inadequ·	0.3353
		maneuv·	0.30931	terrain·	0.2663	crosswind·	0.3085	subsequ·	0.3246
		weather·	0.29484	airspe·	-0.2542	aircraft·	0.3007	maintain·	-0.3117
		adequ·	0.28082			factor·	0.2787	due·	0.2882

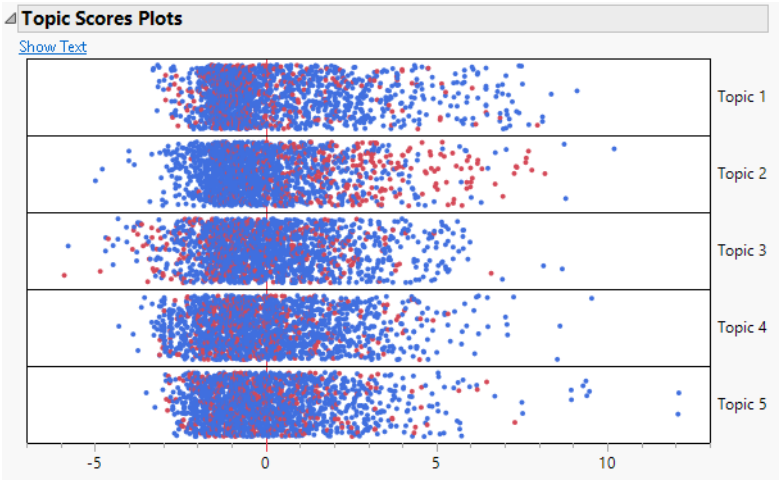
The terms for each topic with the highest loadings enable you to interpret whether the topic is capturing a theme in the incident reports.

For example, Topic 1 has high loadings for power, loss, and engine, indicating a theme of losing power to the engine as a cause of the incident. This corresponds to the phrase "loss of engine power" occurring 273 times in the set of incident reports.

Based on the words with high loadings in Topic 2, it can be described as being related to incidents that involved darkness or low altitude.

18. JMP PRO Click the gray disclosure icon next to Topic Scores Plots.

Figure 12.17 Topic Scores Plots for Narrative Cause



Each topic score plot contains a point for each document in the corpus. You can select points in these plots to further examine the text of specific documents.

You want to further explore the subject matter of Topic 2.

19. JMP PRO Select the three right-most points in the Topic 2 plot and click the **Show Text** button at the top left of the graph.

The text of the three documents with the highest scores in Topic 2 appear in a new window. From these, you can confirm that Topic 2 relates to low altitude.

At this stage of the text analysis, you have many choices in how to proceed. Text analysis is an iterative process, so you might use topic information to further curate your term list by adding stop words or specifying phrases. You might save the weighted document-term matrix, the vectors from the SVD or rotated SVD as numeric columns in your data table and use them in other JMP analysis platforms. When you use these columns in other platforms, you can also include other columns from your data table in further analyses.

Appendix A

References

Agresti, A. (1990). *Categorical Data Analysis*. New York: John Wiley & Sons.

Agresti, A., and Coull, B. A. (1998). "Approximate is Better Than 'Exact' for Interval Estimation of Binomial Proportions." *American Statistician* 52:119–126.

Asiribo, O., and Gurland, J. (1990). "Coping with Variance Heterogeneity." *Communication in Statistics: Theory and Methods* 19:4029–4048.

Bartlett, M. S., and Kendall, D. G. (1946). "The Statistical Analysis of Variance-Heterogeneity and the Logarithmic Transformation." *Supplement to the Journal of the Royal Statistical Society* 8:128–138.

Bissell, A. F. (1990). "How Reliable is Your Capability Index?" *Applied Statistics* 30:331–340.

Brown, M. B., and Benedetti, J. K. (1977). "Sampling Behavior of Tests for Correlation in Two-Way Contingency Tables." *Journal of the American Statistical Association* 72:305–315.

Brown, M. B., and Forsythe, A. B. (1974). "Robust Tests for Equality of Variances." *Journal of the American Statistical Association* 69:364–367.

Chen, S.-X., and Hall, P. (1993). "Empirical Likelihood Confidence Intervals for Quantiles." *The Annals of Statistics* 21:1166–1181.

Chou, Y.-M., Owen, D. B., and Borrego, S. A. (1990). "Lower Confidence Limits on Process Capability Indices." *Journal of Quality Technology* 22:223–229.

Cleveland, W. S. (1979). "Robust Locally Weighted Regression and Smoothing Scatterplots." *Journal of the American Statistical Association* 74:829–836.

Cohen, J. (1960). "A Coefficient of Agreement for Nominal Scales." *Education Psychological Measurement* 20:37–46.

Conover, W. J. (1972). "A Kolmogorov Goodness-of-fit Test for Discontinuous Distributions." *Journal of the American Statistical Association* 67:591–596.

Conover, W. J. (1980). *Practical Nonparametric Statistics*. New York: John Wiley & Sons.

Conover, W. J. (1999). *Practical Nonparametric Statistics*. 3rd ed. New York: John Wiley & Sons.

Cureton, E. E. (1967). "The Normal Approximation to the Signed-Rank Sampling Distribution when Zero Differences are Present." *Journal of the American Statistical Association* 62:1068–1069.

DeLong, E. R., DeLong, D. M., and Clarke-Pearson, D. L. (1988). "Comparing the Areas under Two or More Correlated Receiver Operating Characteristic Curves: A Nonparametric Approach." *Biometrics* 44:837–845.

Devore, J. L. (1995). *Probability and Statistics for Engineering and the Sciences*. Pacific Grove, CA: Duxbury Press.

Dunn, O. J. (1964). "Multiple Comparisons Using Rank Sums." *Technometrics* 6:241–252.

Dunnett, C. W. (1955). "A Multiple Comparisons Procedure for Comparing Several Treatments with a Control." *Journal of the American Statistical Association* 50:1096–1121.

Efron, B. (1981). "Nonparametric Standard Errors and Confidence Intervals." *The Canadian Journal of Statistics* 9:139–158.

Eubank, R. L. (1999). *Nonparametric Regression and Spline Smoothing*. 2nd ed. Boca Raton, Florida: CRC.

Fleiss, J. L., Cohen, J., and Everitt, B. S. (1969). "Large-Sample Standard Errors of Kappa and Weighted Kappa." *Psychological Bulletin* 72:323–327.

Friendly, M. (1994). "Mosaic Displays for Multi-Way Contingency Tables." *Journal of the American Statistical Association* 89:190–200.

Goodman, L. A., and Kruskal, W. H. (1979). *Measures of Association for Cross Classification*. New York: Springer-Verlag.

Gupta, S. S. (1965). "On Some Multiple Decision (Selection and Ranking) Rules. *Technometrics* 7:225–245.

Hajek, J. (1969). *A Course in Nonparametric Statistics*. San Francisco: Holden-Day.

Hartigan, J. A., and Kleiner, B. (1981). "Mosaics for Contingency Tables." In *Computer Science and Statistics: Proceedings of the Thirteenth Symposium on the Interface*, edited by W. F. Eddy, 268–273. New York: Springer-Verlag.

Hayter, A. J. (1984). "A Proof of the Conjecture That the Tukey-Kramer Method Is Conservative." *Annals of Mathematical Statistics* 12: 61–75.

Hosmer, D. W., and Lemeshow, S. (1989). *Applied Logistic Regression*. New York: John Wiley & Sons.

Howell, D. C. (2013). *Statistical Methods for Psychology*. 8th ed. Belmont, California: Wadsworth.

Hsu, J. (1981). "Simultaneous Confidence Intervals for All Distances from the 'Best'." *Annals of Statistics* 9:1026–1034.

Hsu, J. C. (1996). *Multiple Comparisons: Theory and Methods*. London: Chapman and Hall.

Huber, P. J. (1973). "Robust Regression: Asymptotics, Conjecture, and Monte Carlo." *Annals of Statistics* 1:799–821.

Huber, P. J., and Ronchetti, E. M. (2009). *Robust Statistics.* 2nd ed. New York: John Wiley & Sons.

Iman, R. L. (1974). "Use of a t-statistic as an Approximation to the Exact Distribution of Wilcoxon Signed Ranks Test Statistic." *Communications in Statistics—Simulation and Computation* 3:795–806.

Jones, M. C., and Pewsey, A. (2009). "Sinh-Arcsinh Distributions." *Biometrika* 96:761–780.

Kendall, M., and Stuart, A. (1979). *The Advanced Theory of Statistics*. 4th ed. Vol. 2. New York: Macmillan.

Keuls, M. (1952). "The Use of the 'Studentized Range' in Connection with an Analysis of Variance." *Euphytica* 1.2:112–122.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Lehmann, E. L., and D'Abrera, H. J. M. (2006). *Nonparametrics: Statistical Methods Based on Ranks*. Rev. ed. San Francisco: Holden-Day.

Levene, H. (1960). "Robust Tests for the Equality of Variance." In *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, edited by I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, and H. B. Mann. Palo Alto, CA: Stanford University Press.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. London: Chapman and Hall.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Meeker, W. Q., Hahn, G. J., and Escobar, L. A. (2017). *Statistical Intervals: A Guide for Practitioners and Researchers*. 2nd ed. New York: John Wiley & Sons.

Miller, A. J. (1972). "Letter to the Editor." *Technometrics* 14:507.

Nagelkerke, N. J. D. (1991). "A Note on a General Definition of the Coefficient of Determination." *Biometrika* 78:691–692.

Nelson, P. R., Wludyka, P. S., and Copeland, K. A. F. (2005). *The Analysis of Means: A Graphical Method for Comparing Means, Rates, and Proportions*. Philadelphia: Society for Industrial and Applied Mathematics.

Neter, J., Wasserman, W., and Kutner, M. H. (1996). *Applied Linear Statistical Models*. 4th ed. Boston: Irwin.

O'Brien, R. G. (1979). "A General ANOVA Method for Robust Tests of Additive Models for Variances." *Journal of the American Statistical Association* 74:877–880.

O'Brien, R., and Lohr, V. (1984). "Power Analysis For Linear Models: The Time Has Come." *Proceedings of the Ninth Annual SAS User's Group International Conference*, 840–846. Cary, NC: SAS Institute Inc.

Olejnik, S. F., and Algina, J. (1987). "Type I Error Rates and Power Estimates of Selected Parametric and Nonparametric Tests of Scale." *Journal of Educational Statistics* 12:45–61.

Pratt, J. W. (1959). "Remarks on Zeros and Ties in the Wilcoxon Signed Rank Procedures." *Journal of the American Statistical Association* 54:655–667.

Reinsch, C. H. (1967). "Smoothing by Spline Functions." *Numerische Mathematik* 10:177–183.

Rousseeuw, P. J., and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. New York: John Wiley & Sons.

Rubin, D. (1981). "The Bayesian Bootstrap." *The Annals of Statistics* 9:130–134.

SAS Institute Inc. (2020a). "Introduction to Nonparametric Analysis." In *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/intronpar.pdf.

SAS Institute Inc. (2020b). "The FREQ Procedure." In *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/freq.pdf.

Schuirmann, D. J. (1987). "A Comparison of the Two One-Sided Tests Procedure and the Power Approach for Assessing the Equivalence of Average Bioavailability." *Journal of Pharmacokinetics and Biopharmaceutics* 15:657–680.

Slifker, J. F., and Shapiro, S. S. (1980). "The Johnson System: Selection and Parameter Estimation." *Technometrics* 22:239–246.

Snedecor, G. W., and Cochran, W. G. (1980). *Statistical Methods*. 7th ed. Ames, Iowa: Iowa State University Press.

Somers, R. H. (1962). "A New Asymmetric Measure of Association for Ordinal Variables." *American Sociological Review* 27:799–811.

Stephens, M. A. (1974). "EDF Statistics for Goodness of Fit and Some Comparisons." *Journal of the American Statistical Association* 69:730–737.

Tan, C. Y., and Iglewicz, B. (1999). "Measurement-Methods Comparisons and Linear Statistical Relationship." *Technometrics* 41:192–201.

Tamhane, A. C., and Dunlop, D. D. (2000). *Statistics and Data Analysis*. Englewood Cliffs, NJ: Prentice-Hall.

Tukey, J. W. (1953). "The Problem of Multiple Comparisons." In *Multiple Comparisons, 1948–1983*, edited by H. I. Braun, vol. 8 of *The Collected Works of John W. Tukey* (published 1994), 1–300. London: Chapman & Hall. Unpublished manuscript.

Welch, B. L. (1951). "On the Comparison of Several Mean Values: An Alternative Approach." *Biometrika* 38:330–336.

Wheeler, D. J. (2003). *Range Based Analysis of Means*. Knoxville, TN: SPC Press.

Wilson, E. B. (1927). "Probable Inference, the Law of Succession, and Statistical Inference." *Journal of the American Statistical Association* 22:209–212.

Wludyka, P. S., and Nelson, P. R. (1997). "An Analysis-of-Means-Type Test for Variances From Normal Populations." *Technometrics* 39:274–285.

Appendix B

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR

ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
- You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Essential Graphing

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Essential Graphing*. Cary, NC: SAS Institute Inc.

JMP® 16 Essential Graphing

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Essential Graphing

1 Learn about JMP 11
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 13
JMP Help 14
JMP Documentation Library 14
Additional Resources for Learning JMP 20
JMP Tutorials 20
Sample Data Tables 20
Learn about Statistical and JSL Terms 21
Learn JMP Tips and Tricks 21
JMP Tooltips 21
JMP User Community 22
Free Online Statistical Thinking Course 22
JMP New User Welcome Kit 22
Statistics Knowledge Portal 22
JMP Training 22
JMP Books by Users 23
The JMP Starter Window 23
JMP Technical Support 23

2 Introduction to Interactive Graphing 25
Overview of Data Visualization

3 Graph Builder 27
Interactive Data Visualization
How to Use Graph Builder 30
About the Graph Builder Window 30
Example of Features in Graph Builder 32
Move or Remove Variables in Zones 39
Work with Axes 40
Merge Variables on a Common Axis 40
Create Separate Axes in the X and Y Zones 41
Create a Second Y Axis 43

Create Nested Axes for Character Variables ... 45
Order the Levels of a Categorical Variable ... 49
Element Types and Options ... 53
Points ... 55
Smoother ... 57
Line of Fit ... 58
Ellipse ... 60
Contour ... 61
Line ... 64
Bar ... 67
Area ... 78
Box Plot ... 81
Histogram ... 85
Heatmap ... 87
Pie ... 89
Treemap ... 91
Mosaic ... 94
Caption Box ... 95
Formula ... 97
Map Shapes ... 98
Parallel ... 99
Graph Zones ... 103
Categorical and Continuous Variables in Group X or Y Zones ... 107
Move Group X and Group Y Labels ... 108
Red Triangle Options ... 108
Graph Builder Buttons ... 110
Right-Click Menus ... 111
Graph Options ... 111
Grouping Zones Options ... 112
Options for Columns in the Variables Panel ... 112
Options for Axes, Axes Variable Labels, or Graph Titles ... 113
Options for the Dividing Line between Multiple Graphs ... 114
Legend Options ... 115

4 Graph Builder Examples ... 121
Interactive Data Visualization

Example of an Ordered Bar Chart ... 123
Example of an Area and Line Chart ... 126
Example of a Map Chart ... 131
Example of a Butterfly Chart ... 134

Example of a Scatterplot ... 138
Example of a Packed Bar Chart ... 143
Example of an Overlaid Histogram and Ridgeline Chart ... 145
Example of a Bullet Bar Chart ... 147
Example of a Forecast Plot ... 149

5 Bubble Plots ... 151
View Patterns in Multidimensional Data Using Bubble Plots

Example of a Dynamic Bubble Plot ... 153
Launch the Bubble Plot Platform ... 155
Specifying Two ID Variables ... 156
Specifying a Time Variable ... 156
Interact with the Bubble Plot ... 157
Control Animation for Dynamic Bubble Plots ... 157
Select Bubbles ... 158
Use the Brush Tool ... 159
Bubble Plot Platform Options ... 159
Show Roles ... 162
Additional Examples of the Bubble Plot Platform ... 163
Example of Specifying Only a Time Variable ... 163
Example of Specifying Only ID Variables and Splitting a Bubble ... 165
Example of a Static Bubble Plot ... 167
Example of a Bubble Plot with a Categorical Y Variable ... 171

6 Scatterplot Matrix ... 173
View Multiple Bivariate Relationships Simultaneously

Example of a Scatterplot Matrix ... 175
Example using the Scatterplot Matrix Platform ... 175
Example of a Scatterplot Matrix in Graph Builder ... 176
Launch the Scatterplot Matrix Platform ... 177
Change the Matrix Format ... 178
The Scatterplot Matrix Window ... 179
Scatterplot Matrix Platform Options ... 179
Example Using a Grouping Variable ... 181
Create a Grouping Variable ... 182

7 Parallel Plots ... 185
View Patterns in Multidimensional Data by Plotting Parallel Coordinates

Example of a Parallel Plot ... 187
Launch the Parallel Plot Platform ... 189
The Parallel Plot ... 190

Interpreting Parallel Plots ... 190
Parallel Plot Platform Options ... 192
Additional Examples of the Parallel Plot Platform ... 193
Examine Iris Measurements ... 193
Examine Student Measurements ... 194

8 Cell Plots ... 197
View Color-Intensity Plots of Variables

Example of a Cell Plot ... 199
Launch the Cell Plot Platform ... 200
The Cell Plot ... 200
Cell Plot Platform Options ... 201
Right-Click Menu for Cell Plots ... 202
Additional Example of the Cell Plot Platform ... 203

9 Scatterplot 3D ... 205
Create a Rotating Three-Dimensional View of Data

Example of a 3D Scatterplot ... 207
Launch the Scatterplot 3D Platform ... 208
The Scatterplot 3D Report ... 209
Spin the 3D Scatterplot ... 211
Change Variables on the Axes ... 212
Adjust the Axes ... 212
Assign Colors and Markers to Data Points ... 213
Assign Colors and Markers in the Data Table ... 213
Scatterplot 3D Platform Options ... 213
Normal Contour Ellipsoids ... 216
Nonparametric Density Contours ... 216
Pop-Up Menu ... 218
Additional Examples of the Scatterplot 3D Platform ... 221
Example of an Ungrouped Normal Contour Ellipsoid ... 221
Example of Grouped Normal Contour Ellipsoids ... 222
Example of a Grouped Nonparametric Density Contour ... 223

10 Contour Plots ... 227
View Multidimensional Relationships in Two Dimensions

Example of a Contour Plot ... 229
Example Using the Contour Platform ... 229
Example of a Contour Plot in Graph Builder ... 230
Launch the Contour Plot Platform ... 231
The Contour Plot ... 233

Contour Plot Platform Options ... 233
Fill Areas ... 235
Contour Specification ... 235
Contour Plot Save Options ... 239
Use Formulas for Specifying Contours ... 240
Additional Example of Contour Plots ... 241

11 Ternary Plots ... 243
View Plots for Compositional or Mixture Data

Example of a Ternary Plot ... 245
Launch the Ternary Plot Platform ... 247
The Ternary Plot ... 248
Mixtures and Constraints ... 249
Ternary Plot Platform Options ... 249
Additional Examples of the Ternary Plot Platform ... 250
Example Using Mixture Constraints ... 250
Example Using a Contour Function ... 251

12 Maps ... 253
Add Maps or Custom Shapes to Enhance Data Visualization

Overview of Mapping ... 255
Example of Creating a Map in Graph Builder ... 256
Graph Builder ... 257
Map Shape ... 257
Color ... 259
Size ... 260
Custom Map Files ... 263
Background Maps ... 269
Images in Maps ... 271
Boundaries ... 276
Examples of Creating Maps ... 277
Louisiana Parishes Example ... 277
Hurricane Tracking Examples ... 285
Office Temperature Study ... 290
Boundary Map with Clipped Elements ... 295

A Legacy Platforms ... 299
Treemaps ... 299
Charts ... 300
Overlay Plots ... 300

B References 301
C Technology License Notices 303

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 13
JMP Help . . . 14
JMP Documentation Library . . . 14
Additional Resources for Learning JMP . . . 20
JMP Tutorials . . . 20
Sample Data Tables . . . 20
Learn about Statistical and JSL Terms . . . 21
Learn JMP Tips and Tricks . . . 21
JMP Tooltips . . . 21
JMP User Community . . . 22
Free Online Statistical Thinking Course . . . 22
JMP New User Welcome Kit . . . 22
Statistics Knowledge Portal . . . 22
JMP Training . . . 22
JMP Books by Users . . . 23
The JMP Starter Window . . . 23
JMP Technical Support . . . 23

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction to Interactive Graphing

Overview of Data Visualization

Essential Graphing describes all of the different graphs and elements you can use to visualize your data:

- Graph Builder interactively creates many different types of graphs. See "Graph Builder" and "Graph Builder Examples".
- Bubble Plot creates a scatterplot that represents its points as circles, or bubbles. Bubble plots can be dynamic (animated over time) or static (fixed bubbles that do not move). See "Bubble Plots".
- Scatterplot Matrix shows an ordered collection of bivariate graphs. See "Scatterplot Matrix".
- Parallel Plot draws connected line segments that represent each row in a data table. See "Parallel Plots".
- Cell Plot draws a rectangular array of cells where each cell corresponds to a data table entry. See "Cell Plots".
- 3D Scatterplot shows the values of numeric columns in the associated data table in a rotatable, three-dimensional view. See "Scatterplot 3D".
- Contour Plot constructs contours of a response in a rectangular coordinate system. See "Contour Plots".
- Ternary Plot display the distribution and variability of three-part compositional data. See "Ternary Plots".
- Maps can be used in Graph Builder, but also in other platforms, as background maps. See "Maps".
- Treemaps, charts, and overlay plots are deprecated platforms and can be created using Graph Builder. See "Legacy Platforms".

Chapter 3

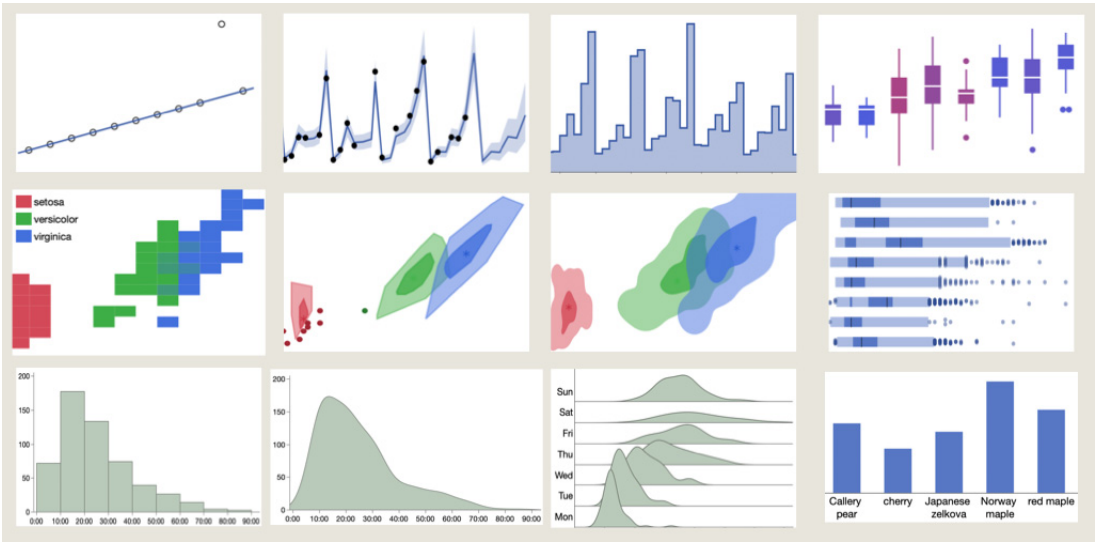
Graph Builder

Interactive Data Visualization

Use Graph Builder to interactively explore your data. You can quickly create and experiment with plots until you find the one you want. Then, you can share your results with others.

This chapter shows you how to use Graph Builder. For detailed examples using Graph Builder, see "Graph Builder Examples".

Figure 3.1 Graphs Created in Graph Builder



Contents

How to Use Graph Builder 30
About the Graph Builder Window 30
Example of Features in Graph Builder 32
Move or Remove Variables in Zones 39
Work with Axes 40
Merge Variables on a Common Axis 40
Create Separate Axes in the X and Y Zones 41
Create a Second Y Axis 43
Create Nested Axes for Character Variables 45
Order the Levels of a Categorical Variable 49
Element Types and Options 53
Points 55
Smoother 57
Line of Fit 58
Ellipse 60
Contour 61
Line 64
Bar 67
Area 78
Box Plot 81
Histogram 85
Heatmap 87
Pie 89
Treemap 91
Mosaic 94
Caption Box 95
Formula 97
Map Shapes 98
Parallel 99
Graph Zones 103
Categorical and Continuous Variables in Group X or Y Zones 107
Move Group X and Group Y Labels 108
Red Triangle Options 108
Graph Builder Buttons 110
Right-Click Menus 111
Graph Options 111
Grouping Zones Options 112
Options for Columns in the Variables Panel 112
Options for Axes, Axes Variable Labels, or Graph Titles 113

Options for the Dividing Line between Multiple Graphs 114
Legend Options 115

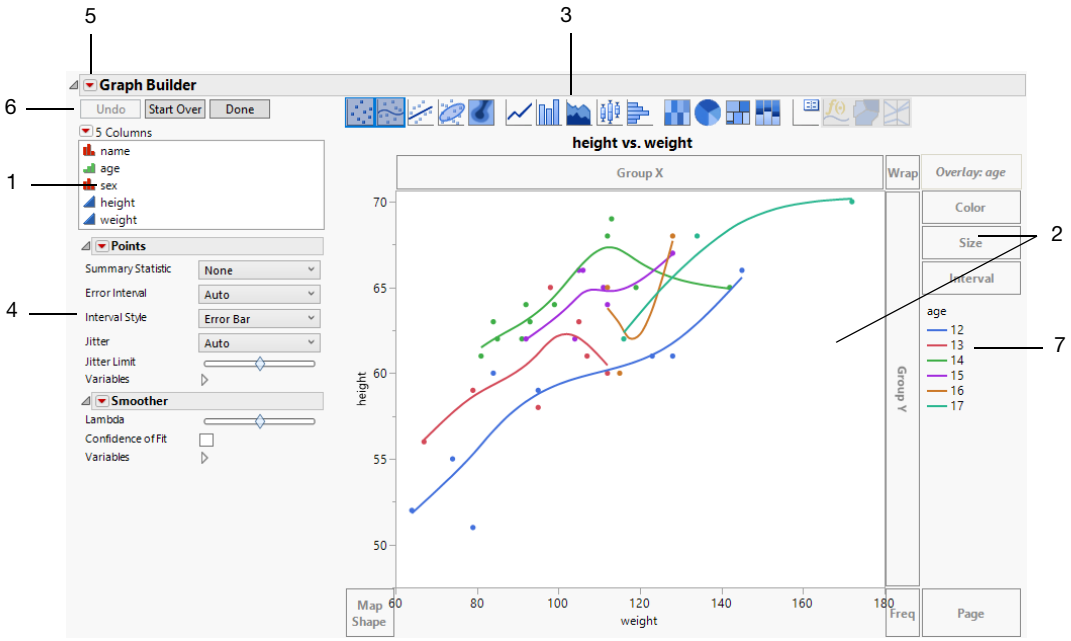
How to Use Graph Builder

1. Open the data table containing the data that you want to graph.
2. Select **Graph** > **Graph Builder**.
3. Drag columns from the Variables list into zones. See "Move or Remove Variables in Zones" and "Graph Zones".
4. Click element type icons to choose different types of graphs or elements. See "Element Types and Options".
5. (Optional) Customize the selected element types. See "Element Types and Options".
6. (Optional) Customize the legend. See "Legend Options".
7. When you are satisfied with the graph, click **Done**.

About the Graph Builder Window

To launch Graph Builder, open your data table and then select **Graph > Graph Builder**. The Variables list contains the columns in your data table, which you can drag into zones.

Figure 3.2 Graph Builder Window for Big Class.jmp



Number	Description
1	The columns in your data table are listed in the Variables panel. Drag columns from the Columns list into zones, or select the column in the Variables list and click the empty zone. You can create a virtual column by right-clicking a column name in the Variables list and selecting from the options provided. • The transformed column appears at the bottom of the list in italics. • You can drag that column into Graph Builder zones. • Right-click the transformed column for further options. **Tip:** For descriptions of the column options in the Variables red triangle menu, see *Using JMP*.
2	The graph area and zones are the main body of Graph Builder. Your graph appears in the middle, and you drag variables into different zones. See "Graph Zones". **Note:** Any excluded rows are hidden in graphs.
3	Click an element type icon to choose the type of graph that you want. You can apply several types of elements to your data simultaneously (using the Shift key). See "Element Types and Options".
4	Customize the graph with specific options for the type of graph you have selected. See "Element Types and Options".
5	Customize the Graph Builder window. See "Red Triangle Options".
6	Undo actions, start over, and more. See "Graph Builder Buttons".
7	If your graph has a legend, it appears here. To customize the legend, see "Legend Options".

To re-create and modify the graph in Figure 3.2, follow these steps:

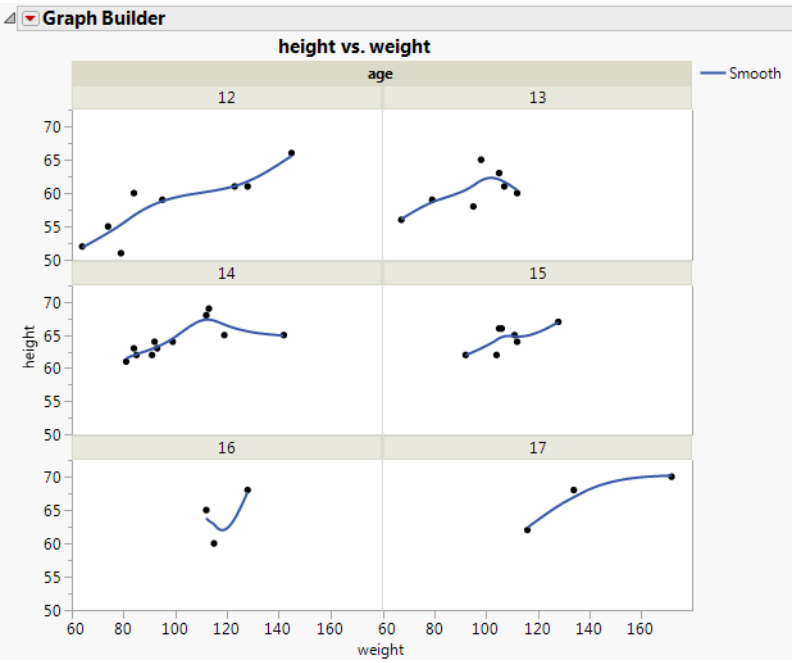
1. Select **Help** > **Sample Data Library** and open Big Class.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select height and drag it to the **Y** zone.
4. Select weight and drag it to the **X** zone.
5. Select age and drag it to the **Overlay** zone.

The graph in Figure 3.2 appears. The relationship between height and weight for each value of age is shown using color in a single graph.

Now you want to see separate graphs for each value of age.

6. Select age in the **Overlay** zone and drag it to the **Wrap** zone.

Figure 3.3 Height and Weight Grouped by Age



For each value of age, the relationship between height and weight is shown in a separate graph.

Example of Features in Graph Builder

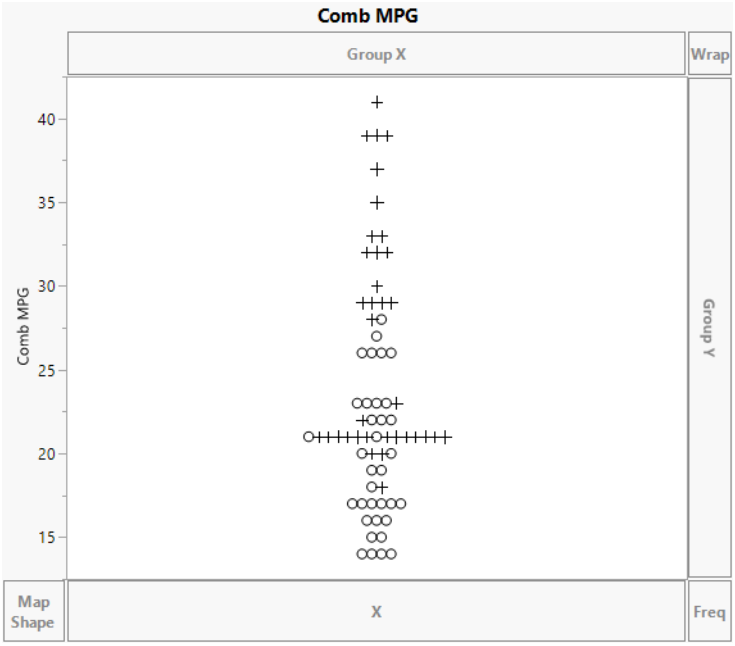
The following example gets you started using some of the features in Graph Builder. This example uses fuel economy data collected for hybrid and non-hybrid cars. You want to get an understanding of miles per gallon (MPG), a measure of fuel economy, and to see which factors might be influencing MPG. This example shows you how to plot several factors of interest:

- "Get an Overall Picture of Combined Miles per Gallon"
- "Plot Mileage Stratified by Engine and Driving Type"
- "Find the Relationship between Hwy and City MPG by Engine Type"
- "Find the Relationship between Hwy and City MPG by Engine Type"

Get an Overall Picture of Combined Miles per Gallon

1. Select **Help > Sample Data Library** and open Hybrid Fuel Economy.jmp.
2. Select **Graph > Graph Builder**.
3. Select Comb MPG (combined miles per gallon) and drag it to the **Y** zone.

Figure 3.4 Initial Graph of Combined MPG



Because markers are assigned to rows in the data table, points are represented by those markers. The value of Comb MPG for each row is plotted at its value on the vertical axis. To avoid over-plotting points that correspond to observations with the same Comb MPG values, JMP automatically jitters the points.

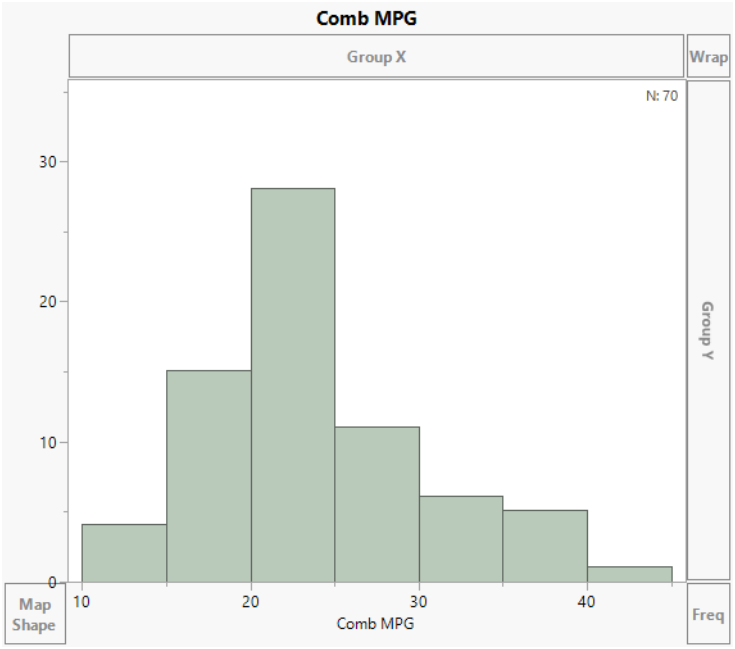
You can get a cleaner picture of the distribution by plotting a histogram.

4. Click the Histogram icon .

   You want the histogram to be vertical, so you need to move Comb MPG to the X zone.

5. Hover over Comb MPG in the Y axis. The cursor turns into a hand. Select and drag Comb MPG into the **X** zone.
6. Click the Caption Box icon to show the mean Comb MPG on the graph.

Figure 3.5 Histogram of Combined MPG



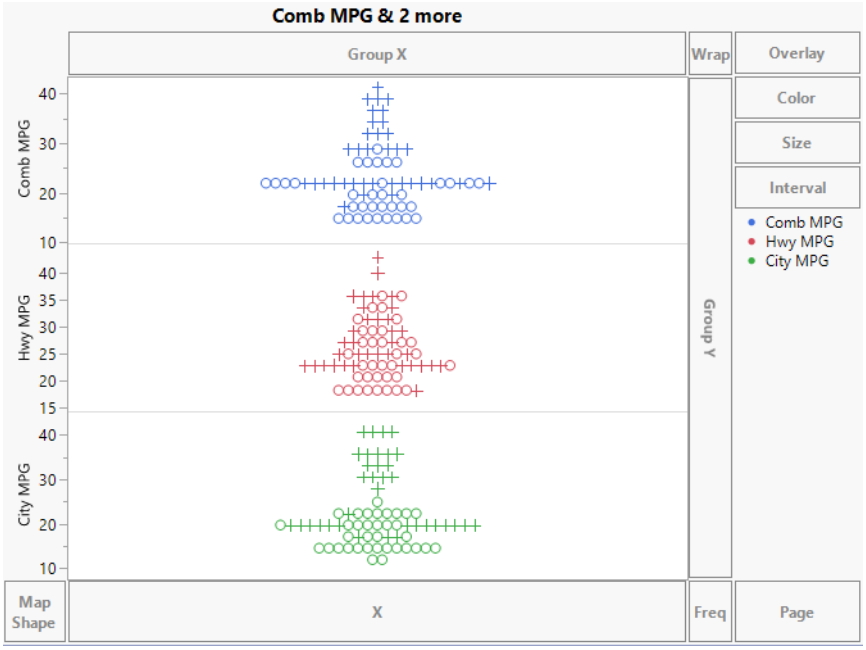
The histogram shows that Comb MPG can range widely, from about 10 to 45. However, a lot of vehicles get mileage between 15 and 30. The mean MPG is 23.3286.

Plot Mileage Stratified by Engine and Driving Type

1. Click **Start Over**.
2. Select City MPG, Hwy MPG, and Comb MPG.
3. Hover over the Y zone, hold down the Shift key and click in the Y zone.

   This creates separate Y axes for City MPG, Hwy MPG, and Comb MPG.

Figure 3.6 MPG Variables with Separate Y Zone Axes

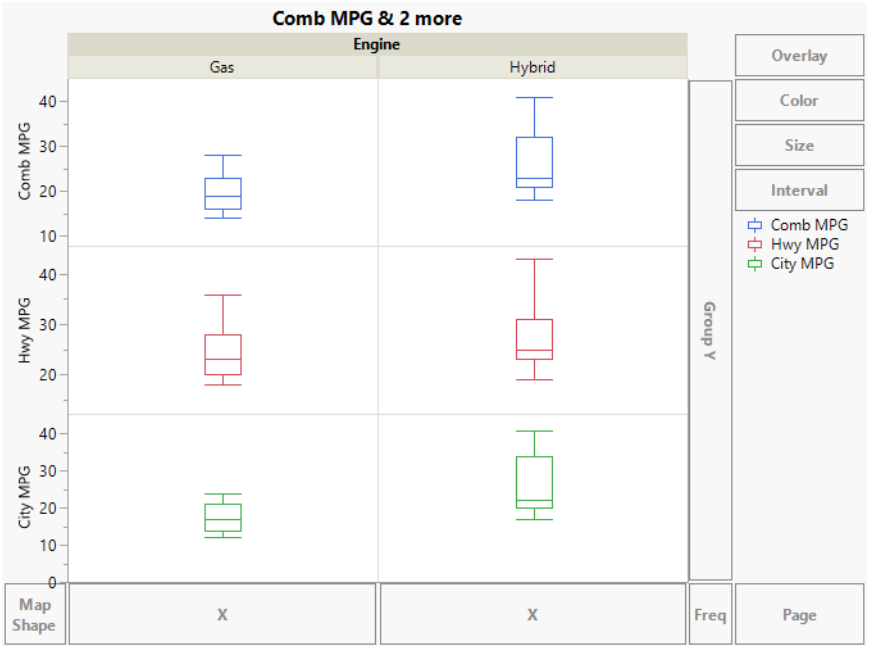


In the legend at right, markers are colored according to the driving type.

Tip: To change the color assigned to a driving type, right-click the marker in the legend.

4. Drag Engine to the Group X zone.
5. Click the Box Plot icon .

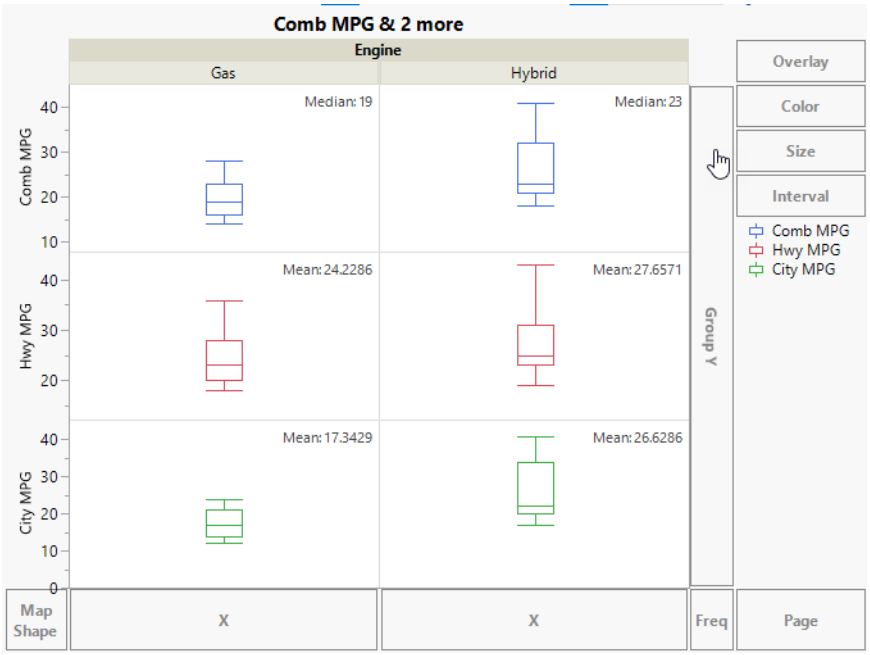
Figure 3.7 Box Plots of MPG Variables by Engine Type



The box plots show the distributions in a very compact form. You can see that all three types of MPG values are substantially lower for the gas vehicles than for the hybrids in the study. From the horizontal lines in the centers of the two City MPG box plots, you can estimate that the median difference is about five miles per gallon. However, you can use the Caption element to be sure.

6. Click the Caption Box icon .
7. In the Caption Box options at the left of the plot, select **Median** from the Summary Statistic list for Comb MPG, Hwy MPG, and City MPG.

Figure 3.8 Box Plots of MPG Variables with Captions

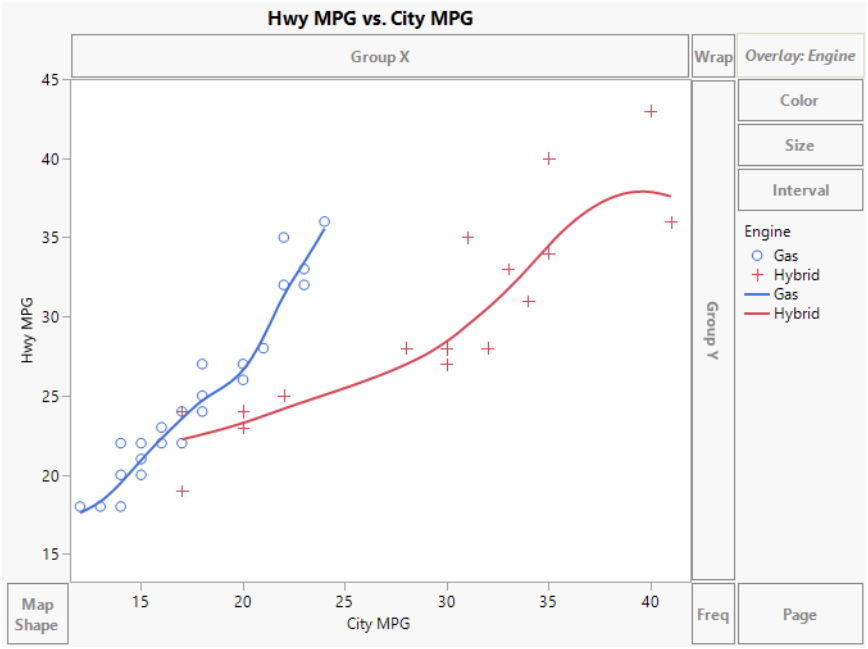


The median city MPG for gas engines is 17 and for hybrid engines is 22.

Find the Relationship between Hwy and City MPG by Engine Type

1. Click **Start Over**.
2. Select City MPG and drag it into the **X** zone.
3. Select Hwy MPG and drag it into the **Y** zone.
4. Select Engine and drag it into the **Overlay** zone.

Figure 3.9 Hwy and City MPG by Engine Type



A Smoother appears for each type of engine, Gas, and Hybrid.

Tip: To change the line properties (color, width, and so on), right-click a line in the legend.

5. Click the Line of Fit icon .

   For each engine type, the smoother is removed and a least squares line, together with confidence bands for the predicted mean, is added.

   Tip: To add the y = x line, right-click in the graph and select **Customize** > **+** > **Templates** > **Y Function**. Replace `_function_of_x_` with `y = x` and click **OK**.

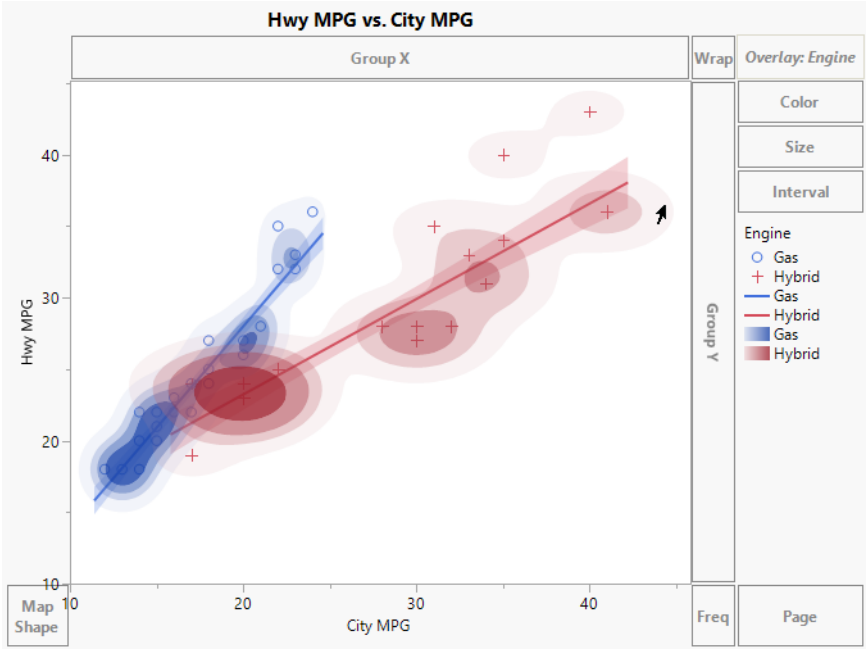
6. Drag a rectangle around the "o" marker with the smallest value of City MPG and check the number of rows selected in the Rows panel of the data table.

   You see that two rows are represented by this marker. You can check that over-plotting happens for other markers as well. When there is over-plotting of observations, namely when several observations are represented by a single point on a plot, density contour plots can help you see the density of points.

7. Drag the Contour icon into the graph.

   Dragging the Contour icon into the graph retains the Line of Fit for each engine type.

Figure 3.10 Density Contour for Hwy and City MPG by Engine Type



The contours indicate that for both engine types, there are more vehicles in the study with lower MPG values than higher MPG values.

Move or Remove Variables in Zones

There are several ways that you can move a variable into a zone:

- To move a variable from one zone to another, click and drag it from the old zone into the new zone.
- To replace a variable in a zone, drag the new variable over the old one.
- To switch a variable from one zone to another, right-click the variable in the zone that you want to move and select **Swap**. Then, select the variable that you want to switch places with.
- To merge variables on the same axis, see "Merge Variables on a Common Axis".

To remove a variable, drag it into empty space, or right-click the name of the variable in the zone and select **Remove**.

Work with Axes

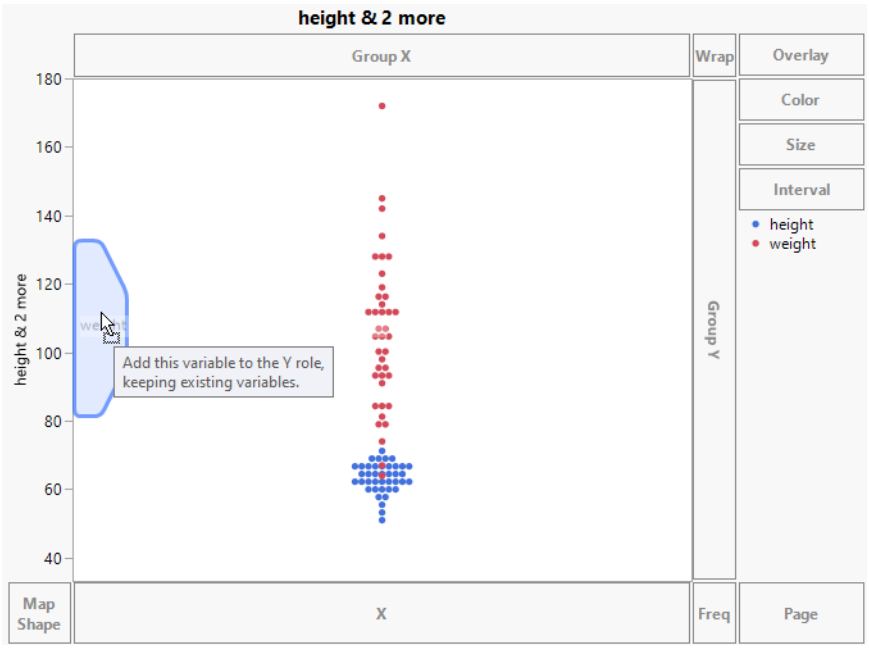
- "Merge Variables on a Common Axis"
- "Create Separate Axes in the X and Y Zones"
- "Create a Second Y Axis"
- "Create Nested Axes for Character Variables"
- "Order the Levels of a Categorical Variable"

Merge Variables on a Common Axis

Merging variables places both variables on the same axis and creates a single graph for both variables. Elements in the plot are colored by the variables, and a legend appears to the right of the plot. When points are plotted, they are randomly jittered for each value or category defined by the combination of the axes.

- If you are starting with an empty zone, highlight multiple variables from the list and drag them into the zone at the same time.
- If you already have variables in the zone and want to add more, drag the new variables to the inner side of the zone, in the graph area, until a symmetric polygonal shape appears.

Figure 3.11 New Variable Merged with Existing Variable



Results Based on Data and Modeling Type

- If only variables with continuous modeling types are merged, their values are plotted against a common axis.
- If variables with categorical (nominal or ordinal) modeling types are merged with a continuous variable, the values of all variables are plotted against a common axis.

  Before you can merge a variable with the character data type with numeric variables, the levels of the character variable have to be assigned numeric values. Integer values are assigned based on the Value Order column property or the default value ordering. These integer values range from 0 (highest level in value ordering) to the number of levels minus one (lowest value in value ordering). These integer values are then plotted against the common axis.
- If only variables with categorical modeling types are merged, then separate, *nested*, axes are constructed for each variable. See "Create Nested Axes for Character Variables".

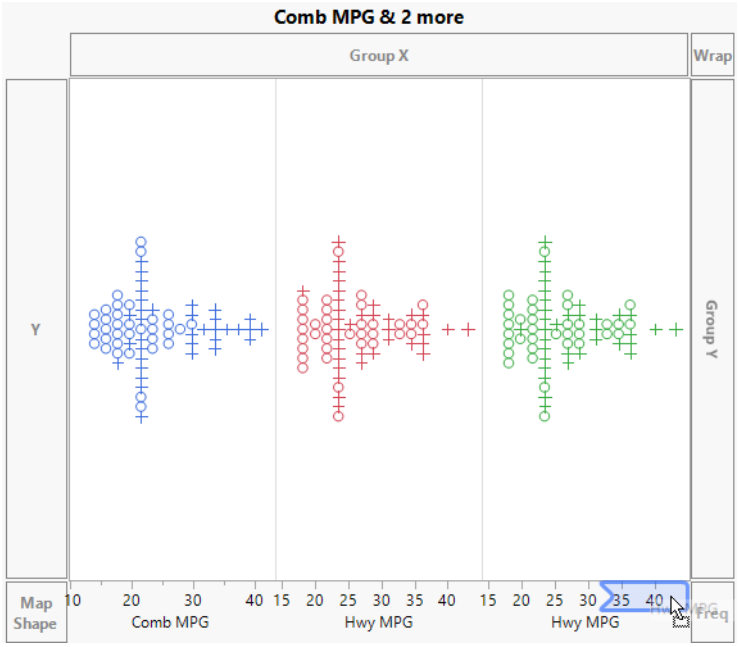
Create Separate Axes in the X and Y Zones

To quickly plot multiple variables against separate axes in the X zone:

1. Select the columns in the Variables list.
2. Click **Shift** and then click the **X** zone.

If you want more control over where the variables appear, drag them individually. Drag additional variables above, below, or between existing variables in the X zone.

Figure 3.12 Dragging a Variable to the Right of the Existing Variable



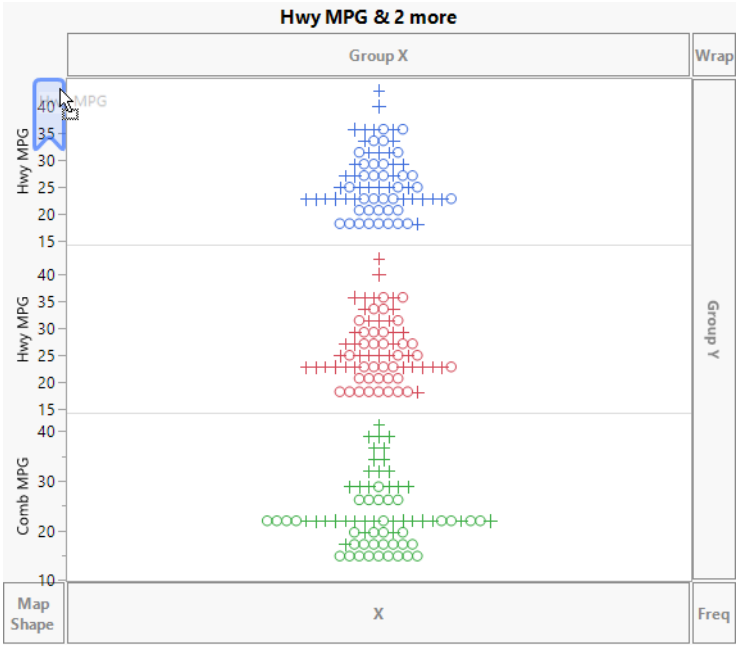
A single polygonal shape indicates when your variable is in the appropriate target zone.

To quickly plot multiple variables against separate axes in the Y zone:

1. Select the columns in the Variables list.
2. Click **Shift** and then click the **Y** zone.

If you want more control over where the variables appear, drag them individually. Drag additional variables above, below, or between existing variables in the Y zone.

Figure 3.13 Dragging a Variable above the Existing Variable



A single polygonal shape indicates when your variable is in the appropriate target zone.

Create a Second Y Axis

If you have two or more Y variables on the same axis, you can reflect the scaling of a second set of variables by creating a second Y axis. This can be useful when two variables measure the same underlying quantities, but have different scales. In general, it is unwise to use a second Y axis in any other situation. See Few (2008).

To create a second Y axis:

1. Right-click one of the Y variable names in the Y zone and select **Move Right**.
2. Select the variable or variables that you want to move to the new axis.

   The new axis is scaled according to the values of the specified variable, and the selected variable is plotted against this axis.
3. Repeat the process to plot additional variables against the new axis.

   The new axis adjusts to accommodate the values of the additional variables.

Change the Graph for a Second Y Axis

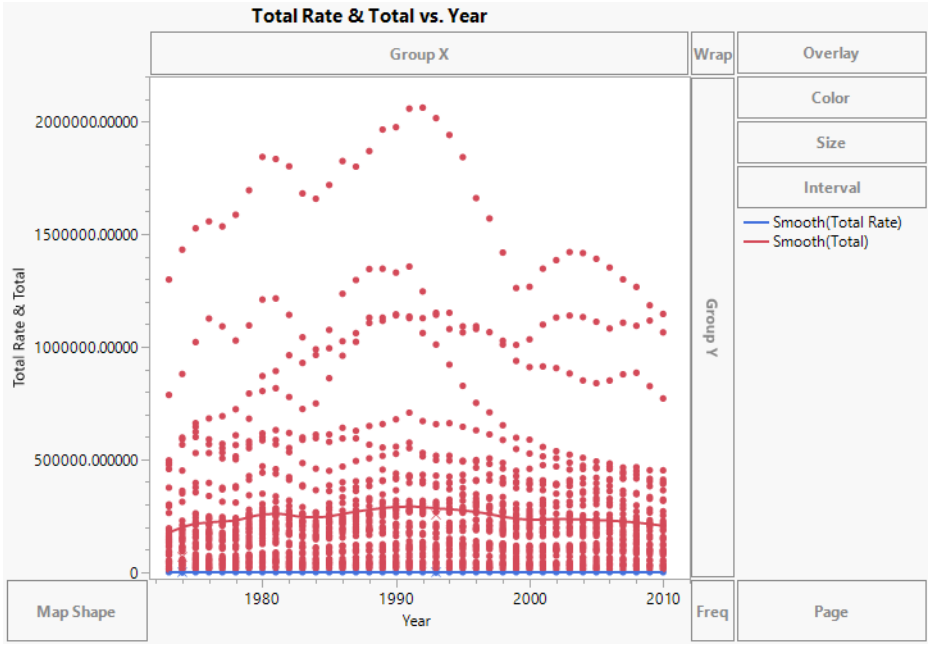
When you click an element type, it applies only to the variables on the left Y axis. To apply an element to the variables on the right Y axis, right-click and change the second element, which corresponds to the right axis.

Example of Creating a Second Y Axis

The CrimeData.jmp sample data table contains data on various types of crime for the 50 US states yearly, from 1973 to 2010. The Total column sums the number of incidents, and the Total Rate column gives a population-adjusted rate. You want to show these two variables on a single chart.

1. Select **Help > Sample Data Library** and open CrimeData.jmp.
2. Select **Graph > Graph Builder**.
3. Select Total Rate and Total and drag them to the Y zone.
4. Select Year and drag it to the X zone.

Figure 3.14 Total Rate and Total Merged

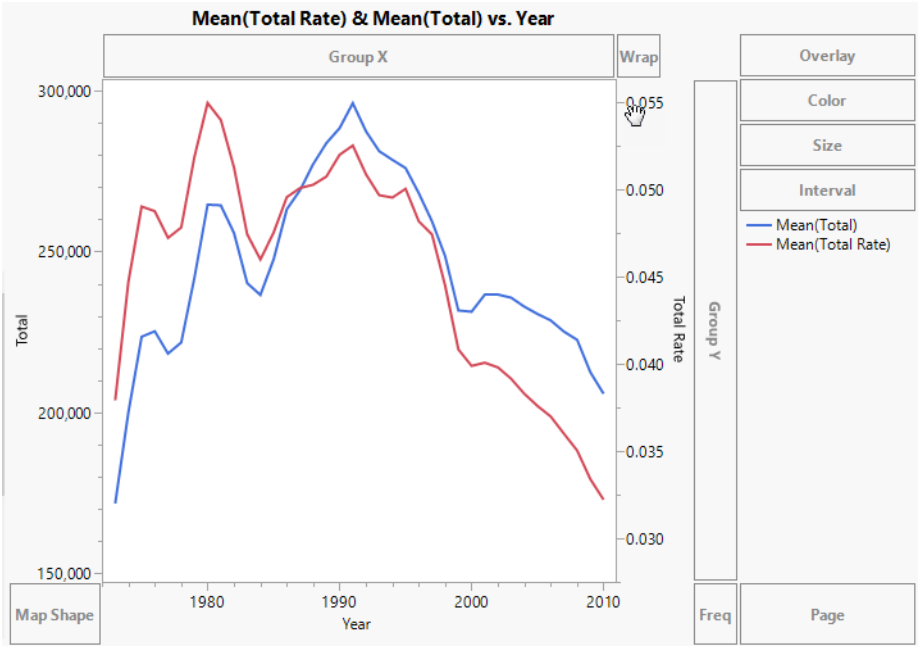


The Total Rate values are all between 0 and 1, and they are barely visible. Calculate yearly means and then use a second Y axis to make these values visible.

5. In the options panel for Points, select **Mean** next to **Summary Statistic**.
6. Click the Line element .

7. Right-click the variable names in the Y zone and select **Move Right** > **Total Rate**.

Figure 3.15 Second Y Axis Added



An axis for Total Rate is added on the right, and the axes rescale to show both sets of values. You can now see relationships between the two measures. For example, both measures began to decrease in 1991.

8. (Optional) Click **Done**.

Create Nested Axes for Character Variables

If you merge variables that all have categorical modeling types, then an individual axis is constructed for each variable. The outermost axis corresponds to the first variable selected, the next to the second, and so on.

To nest the axes in a specific order, drag the variables to the zone individually. First, drag the variable for the innermost axis to the zone. Then drag the variable for the next axis to the outside of the preceding variable and drop the variable once a trapezoid shape appears.

The following example illustrates nested axes.

After changes are made to improve a measurement process, a measurement systems analysis study is conducted to study repeatability and reproducibility with Part, Operator, and Instrument as factors. Each of three operators measures each of eight parts with four instruments. Of particular interest is the consistency of the instruments. The measured quantity is called new Y.

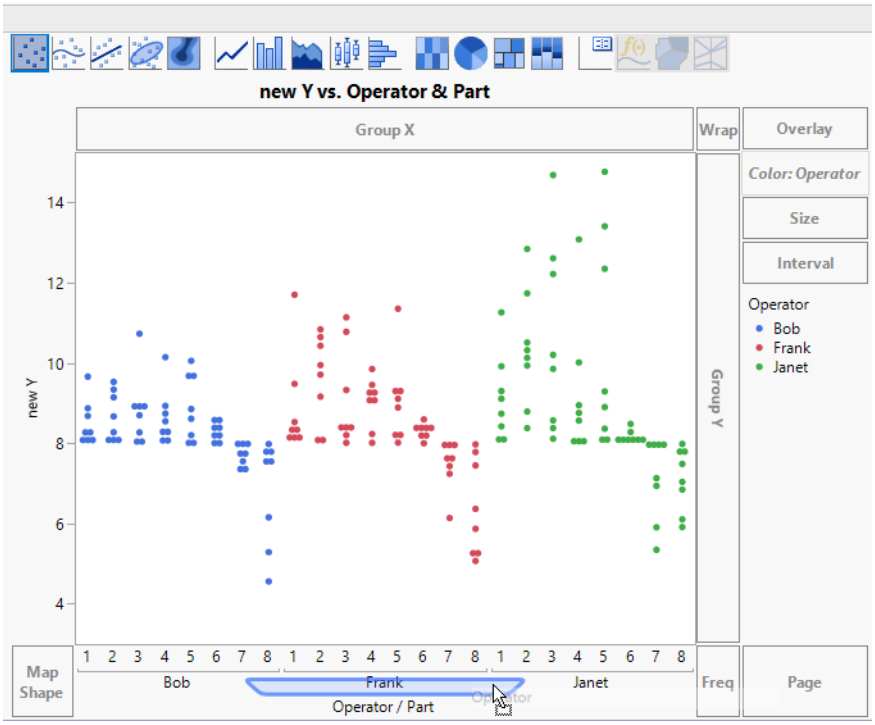
1. Select **Help > Sample Data Library** and open Variability Data/3 Factors Crossed.jmp.
2. Select **Graph > Graph Builder**.
3. Select new Y and drag it to the **Y** zone.
4. Select Part and drag it to the X zone.

   The plot shows variation in the values that are measured for each of the eight parts. There are systematic differences among the parts, which is to be expected. For example, measurements for parts 7 and 8 are lower than those for parts 1 through 6.
5. Drag Operator to the **Color** zone.

   The new Y values are colored by Operator, using the legend at the right of the graph. It appears that Janet might be measuring higher values than the other two operators on most parts. But the Operator effect is not easy to visualize, so you create a separate Operator axis.
6. Select Operator and drag it beneath Part in the X zone.

   The label Operator/Part appears, indicating that Part is associated with the topmost axis, and Operator is associated with the lower axis.

Figure 3.16 Drag Operator to Add a Second Axis

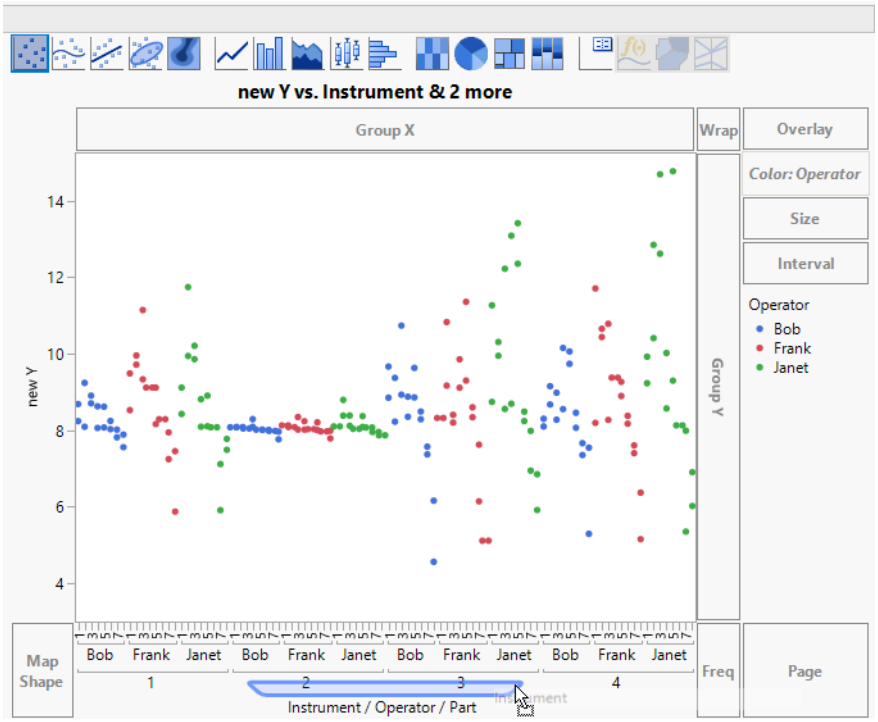


Now it is easier to see that Janet tends to obtain higher measurements for the same parts than do Bob and Frank. But what about the effect of Instrument?

7. Select Instrument and drag it beneath Operator/Part in the X zone.

   The label Instrument/Operator/Part appears, indicating that a third axis for Instrument has been added beneath the Operator axis.

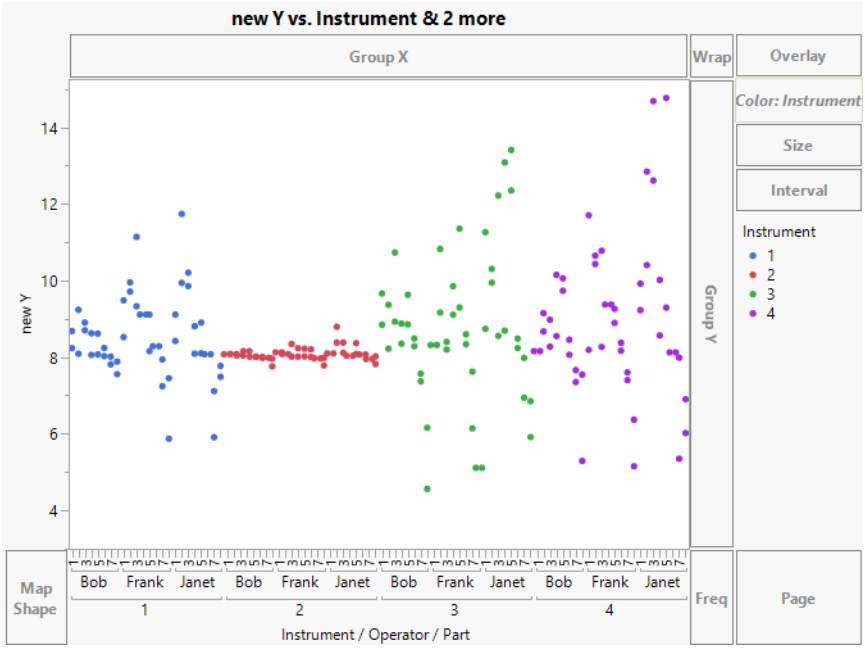
Figure 3.17 Drag Instrument to Add a Third Axis



It is clear that Instrument 2 leads to much more consistent measurements than the other three instruments. For Instrument 2, there is comparatively little variation between or within operators.

8. Select Instrument and drag it to the **Color** zone.

Figure 3.18 Three Nested Axes



Now the new Y values are colored by Instrument, and it is easy to see Instrument differences.

By nesting the axes for the three factors in study, you are able to obtain a visual understanding of the variation attributable to the factors.

Order the Levels of a Categorical Variable

JMP orders the levels of a categorical variable according to the following rules:

- Numeric, nominal data are sorted numerically.
  - White space around numbers is compared: “vt 1” is sorted before “vt1”.
- Character data that are only digits (numbers) are sorted numerically.
- Character data are sorted alphabetically, with the following exceptions:
  - Months and days of the week are in chronological order.
  - Ratings are sorted from low to high:

    Low to high: Very Low, Low, Medium Low, Medium, Medium High, High, Very High

    Agreement: Strongly Disagree, Disagree, Neutral, Indifferent, Agree, Strongly Agree

    Bad or good: Failing, Unacceptable, Very Poor, Poor, Bad, Acceptable, Average, Good, Better, Very Good, Excellent, Best

- Character data that have a character prefix and a numeric suffix are sorted first by prefix and then by suffix. For example, lots 1 through 12 are ordered “lot1”, “lot2”, “lot3”,...”lot10”, “lot11”, and “lot12”.
- If the numeric suffix starts with a “0”, it does not follow numeric ordering by suffix. (“lot1” is ordered *after* “lot02”).

Note: The preceding rules apply only to graphs with categorical axes.

To change the order of the levels of a categorical variable on a graph axis, click and hold a level on the axis to enter edit mode. In the edit mode, denoted by boxes around each label, you can drag one or more labels (using shift-click for multiple selection) to rearrange them. As you rearrange labels, the value order column property is updated. You can also edit labels, which updates the value label column property. Press Esc to exit the edit mode.

You can also order the levels of a categorical (nominal or ordinal) variable on axes using the Value Order column property, which takes precedence over all other ordering rules. To change the ordering, you can do one of the following:

- Use the Value Order column property to ensure that values are ordered as you intend. For more information about the Value Order column property, see *Using JMP*.
- Use the values of a numeric variable already in the graph. In Graph Builder, right-click the categorical axis and select **Order By**.
- Use the values of an arbitrary numeric variable. In Graph Builder, drag the numeric variable next to the categorical variable in the zone until a trapezoid appears and then drop the variable. The axis label changes to <categorical variable> ordered by <numeric variable>. See “Example of Ordering the Levels of a Categorical Variable Using a Numeric Variable”.

Change the order of the levels or the statistic:

1. Right-click the axis label and select **Order By**.
2. Select one of the ascending or descending options or **other** to select a numerical column.

   You can order by a variable’s Order Statistic (these options are prefixed by the variable’s name) or by the number of observations in each level of the categorical variable (Count).
3. The default ordering statistic is the mean. To use another statistic, right-click the axis label and select **Order Statistic**.

Note: If you try to order the values of a numeric variable using another numeric variable, JMP merges the variables. See “Merge Variables on a Common Axis”.

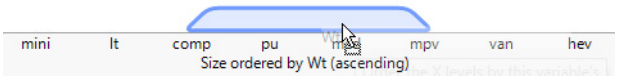
Example of Ordering the Levels of a Categorical Variable Using a Numeric Variable

To order a nominal or ordinal variable by a numeric variable, consider data about vehicle types. You want to see the vehicle types arranged in a meaningful order.

1. Select **Help > Sample Data Library** and open Cars.jmp.
2. Select **Graph > Graph Builder**.
3. Select Size and drag it into the **X** zone.

   This variable represents the type of the vehicle. Eight levels are listed alphabetically on the X axis: compact (comp), heavy (hev), lightweight (lt), medium (med), mini, multi-purpose (mpv), pick-up truck (pu), and van. Since the levels are listed alphabetically, they are not ordered in a meaningful way. For example, heavy comes before mini and lightweight. You want to order the levels by Wt (weight).
4. Select Wt and drag it to the middle of the X axis. Drop it just above the **X** axis. Before you drop the variable, a blue quadrilateral appears.

Figure 3.19 Merging Wt and Size

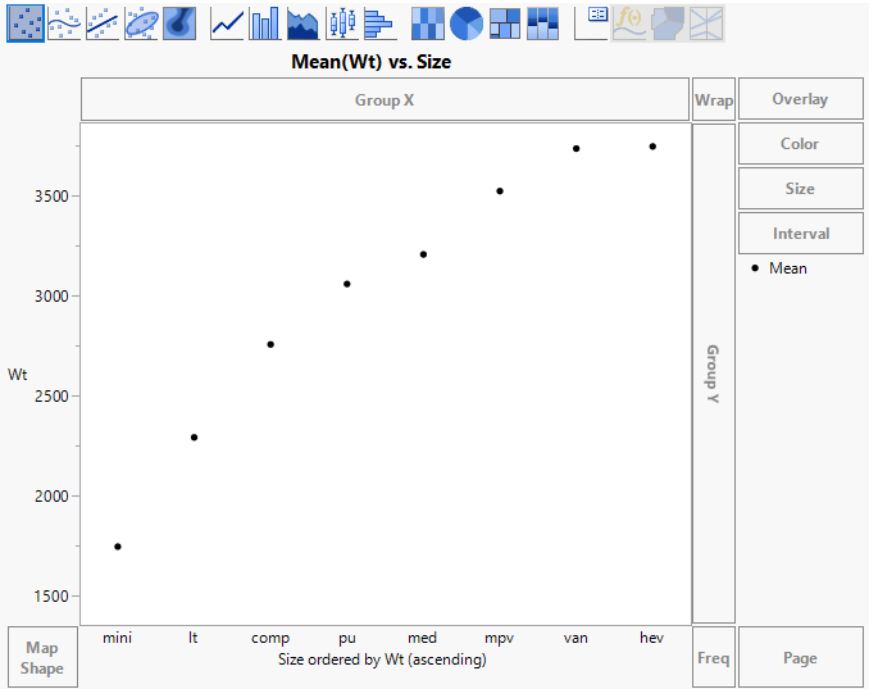


The levels of Size are now arranged in increasing order according to the average Wt of all vehicles in the levels. Notice that mini and lt (lightweight) are now ordered before hev (heavy). The axis label is updated, signifying that an ordering variable is in use.

Next, verify that Size is actually ordered by Wt.

5. Select Wt and drag it to the **Y** zone.
6. Select **Mean** from the Summary Statistic list.

Figure 3.20 Example of Size Ordered by Wt, Ascending



You can see that the average Wt increases from left to right.

Change the order from ascending to descending.

7. Right-click in the **X** zone and select **Order By** > **Wt, descending**.

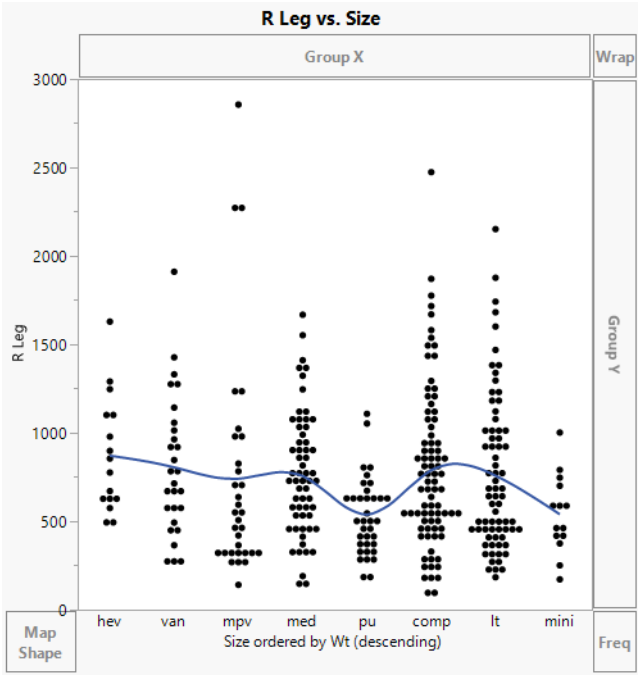
   Now the levels of Size are arranged in decreasing order of mean Wt. You want to see whether the right femur load (R Leg) decreases with vehicle weight.

8. Right-click Wt in the Y zone and select Remove.
9. Select R Leg and drag it to the Y zone.
10. Select **None** from the Summary Statistic list.

    This selection replaces the means with points for all the individual observations.

11. Click the Smoother element .

Figure 3.21 Example of R Leg Ordered by Wt, Descending



R Leg seems unrelated to the weight classes.

The default ordering statistic is the mean. To use another statistic, right-click in the **X** zone, select **Order Statistic** and change it to the statistic that you want.

Element Types and Options

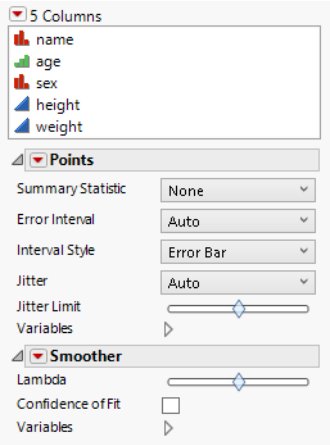
You can change how your data appear by clicking an element type icon, such as a bar chart, line, or histogram.

Figure 3.22 Element Type Icons

Note: Supported element types vary depending on the variable type and zones that are selected. Element types that are not applicable appear dimmed.

For each element that you add to a graph, an outline appears in the Properties area at left. This is where you specify and change properties for each element type.

Figure 3.23 Properties Areas for Points and Smoother

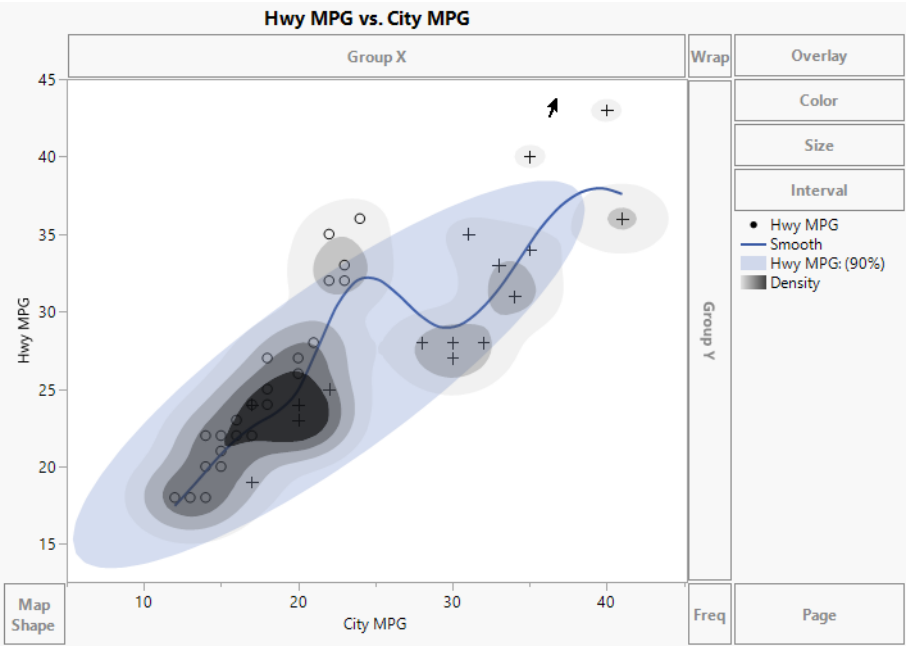


You can also right-click in any graph to change the element, or customize elements in the graph.

Select Multiple Elements at Once

To select and overlay multiple elements, hold down the Shift key and click the elements. Alternatively, you can drag the elements into the graph one at a time.

Figure 3.24 Graph Showing Points, Smoother, Ellipse, and Contour Elements



Labels for Points

In most graphs that display points, when you hover over a point, a label appears showing information about the corresponding row. Hover over the label and right-click for more options. You can pin the label to the graph, copy the contents, or close to remove the label. You can also pin the label by clicking the pin icon in the upper right of the label. See *Using JMP*.

Hidden and Excluded Rows

Hide and exclude are two different row states:

- The goal of a hidden row is to impact visibility but not analytical results.
- The goal of an excluded row is to impact the analytical result.

For graphs, an analytical value is usually part of the appearance. Therefore, excluding rows typically impacts the graph appearance and any analytical result included as part of the graph. Hiding a row typically impacts only the graph appearance.

For example, consider a graph with points and a line of fit:

- If rows are hidden, those rows are included in the calculations to obtain the line of fit but they are not plotted.
- If rows are excluded, those rows are excluded from both the calculations and the display.

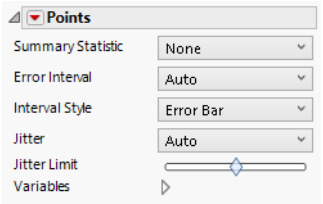
In the two scenarios, the number of points that appear is the same, but the line of fit is different.

When graphing groups, if all rows in a group are hidden, then the visual for that group is also hidden.

Points

The **Points** element shows data values as points.

Figure 3.25 Points Options



Summary Statistic Changes the statistic being plotted. The statistic is calculated using the variable on the Response Axis. None is the default setting, indicating that the data values themselves are plotted.

Error Interval Adds or removes specified error interval in the graph.

Interval Style Draws error bars or error bands for the selected error interval.

Jitter Jitter adds random noise to values to reduce over-plotting. Choose from the following types of jitter:

None No adjustments are made.

Auto Adds various types of jitter when categorical variables are involved. No jitter is added when you have only continuous variables.

Random Uniform Random offset with uniform distribution.

Random Normal Random offset with Gaussian distribution.

Packed Places markers tightly to preserve any non-jittered dimensions.

Centered Grid Similar to Packed, but adjusts the non-jittered dimensions to fall into a grid.

Positive Grid Similar to Center Grid, but in the positive direction.

Density Random For one-dimensional jitter, this option places markers randomly within the bounds of a violin plot. For two-dimensional jitter, this option places markers randomly within an oval.

You can also customize jitter by right-clicking and selecting **Customize** > **Marker**.

Tip: To create a dot plot, select the Positive Grid option. You can move the dot plot to start at the bottom by changing the Y-axis values to 0-1. You can also resize the markers to a larger value, such as 20, to adjust the binning of the dots.

Jitter Limit Controls the spread or amount of overlap.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Points

Response Axis Changes the axis that is associated with the response variable to X (horizontal), Y (vertical), or automatically determines the response direction.

Set Shape Expression Enables the use of a JSL expression to define a marker shape.

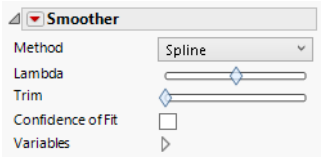
Set Shape Column Sets a column to use as the marker shape.

Smoother

The **Smoother** element shows a smooth curve through the data.

Tip: To label a smoother, right-click on the smoother element in the legend.

Figure 3.26 Smoother Options



Method Specifies the method used for generating the smooth fit. Options include Spline, Local Kernel, Savitzky-Gloay, Moving Average, and Moving Box.

Lambda (Available for Spline.) Enables you to adjust the value of lambda. The default value is 0.05. For more information about lambda, see *Basic Analysis*.

Degree (Available for Savitzky-Golay.) Specifies the degree of the smoother.

Local Width (Not available for Spline.) Enables you to control the smoother fit in terms of a local to a global fit. Use the red triangle Local Region option for further control.

Trim Enables you to trim the sections of the smoother that are based on incomplete or imbalanced data ranges. The slider is scaled from 0 to 1. At 1, all of the incomplete or

imbalanced regions are trimmed with the exception of the spline where 1 corresponds to the penultimate edge knots.

Confidence of Fit Shows or hides the bootstrap confidence region for each fit.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Smoother

Response Axis Changes the axis that is associated with the response variable to X (horizontal), Y (vertical), or Auto (the Y axis for Smoother).

Local Region Specifies options for controlling the region used for the local width option.

Local Weighting Specifies weighting options.

Adapt to Axis Scale For log and other axis transformations, applies computations on the transformed coordinates.

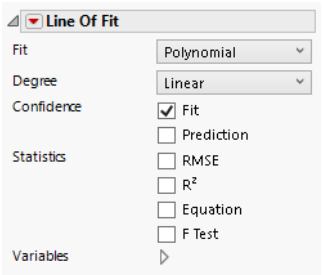
Save Formula Saves fit formulas and upper and lower confidence intervals for a single variable to the data table.

Line of Fit

The **Line of Fit** element shows a linear regression line with confidence intervals for the fit.

Tip: To label a line of fit, right-click on the line element in the legend.

Figure 3.27 Line of Fit Options



Fit Specifies the type of fit:

Polynomial A simple linear regression.

Robust Cauchy A linear regression assuming Cauchy distributed residuals, to de-emphasize outliers.

Time Series Smoothing for equally spaced X values with optimal seasonality. Includes options for a forecast model and the number of seasonal and forecast periods. The smoothing model is selected from a subset of state space smoothing models defined by Hyndman et al. (2008). See *Predictive and Specialized Modeling*.

Degree Specifies the polynomial degree of the linear regression fit, which can be linear, quadratic, or cubic.

Confidence Shows or hides confidence intervals for the predicted value (Fit) or for individual values (Prediction). Both types of intervals are fixed at 95% confidence.

Statistics Shows various selections on the graph. You can show the root mean square error (RMSE), R-square, the equation of the regression line, and the F Test value.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.
– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Line of Fit

Response Axis Specifies the axis for the variable that is used as the response in the calculation of the linear regression line. Because standard least squares typically uses the Y (vertical) axis as the response axis, the Auto setting defaults to the Y axis.

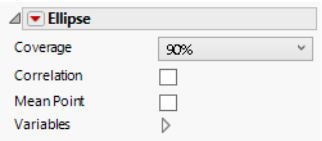
Adapt to Axis Scale For log and other axis transformations, applies computations on the transformed coordinates.

Save Formula Saves fit formulas and upper and lower confidence intervals for a single variable to the data table.

Ellipse

The **Ellipse** element shows a bivariate normal density ellipse.

Figure 3.28 Ellipse Options



Coverage Specifies the coverage percentage (99%, 95%, 90%, or 50%) for the density ellipse.

Correlation Shows the Pearson correlation coefficient for the X and Y variables on the graph.

Mean Point Shows the mean on the graph.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Option for Ellipse

Adapt to Axis Scale For log and other axis transformations, applies computations on the transformed coordinates.

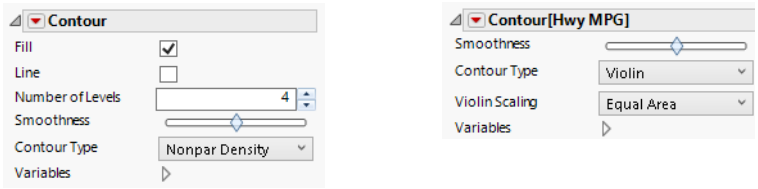
Contour

The **Contour** element shows regions of density (or value contours when used with a Color variable). Density contours are useful when you have a scatterplot with many points where the mass of points makes it difficult to see patterns in density. Multiple contour types are available. The default is a smooth bivariate nonparametric density surface that is fit to reflect the density of the data points. The nonparametric density surface estimates the bivariate probability density function at each point, providing a continuous analog of a bivariate histogram.

- For two continuous variables, four contours are plotted by default. These contours are 100%, 75%, 50%, and 25% density contours. You can specify the number of contour levels to display. Alternatively, you can select a Bagplot or High Density Region (HDR) contours.
- For only one continuous variable, a *violin* plot appears instead of a contour plot. A violin plot illustrates the density of the data by plotting symmetric kernel densities around a common vertical axis. The kernel density estimates the probability density function at each point, providing a continuous analog of the histogram. The violin plot is similar to a box plot with symmetric kernel densities replacing the box and whiskers. Alternatively, you can select High Density Region (HDR) contours.
- If you add a Color variable to a contour plot, the plot shows *value contours* that reflect the levels of the Color variable. The value contours are computed using Delaunay triangulation. You can select an option (Transform) to show a plot where the X and Y ranges have been normalized.

Contour Options

Figure 3.29 Contour Options for a Contour Plot or a Violin Plot



Fill (Not available for Bagplots.) Fills in the contours.

Line (Not available for Bagplots.) Adds lines around the contours.

Number of Levels (Available for Nonpar Density.) For density contours, specifies the number of contours that appear. The number can be between 1 and 1000, the default is 4 contours.

Boundary (Available only when you have a Color variable.) Adds a line around the outside boundary of the contour.

Alpha (Available only when you have a Color variable.) Controls the hull of value contours. Increasing alpha can eliminate some of the long, skinny, or large triangles where interpolation might be undesirable.

Smoothness (Not available for Bagplots.) Smooths the boundaries of the contour plots. The smoothness value is normalized between -1 and 1. This value can be interpreted as a smoothing kernel radius. The original data are interpolated to a grid, and then a Gaussian smoother is applied.

Transform (Available only when you have a Color variable.) Transforms the triangulation to use a normalized scale for X and Y by selecting Range Normalized. This causes both the X and Y values to be scaled to [0,1] before computing the Delaunay triangulation. This option might be desirable in cases where the X and Y units are very different. In these cases, if points are stretched in one dimension and not the other, Delaunay triangulation tries to minimize long, skinny triangles, which can obscure some features.

Contour Type Enables selection of contour types

Nonpar Density (Available only for bivariate plots.) A smooth bivariate nonparametric density surface.

Bagplot Draws a Bagplot, also known as a bivariate box plot. A Bagplot consists of two polygons, a set of outlier points, and a median point. All computations are based on first computing the Tukey depth (bivariate depth) of each point in the data. The median point is the average of all points at maximum depth, which is plotted as an asterisk. The inner polygon is the *bag*, which contains at most 50% of the data points. Not shown in the plot is the *fence*, which is the bag polygon inflated three times relative to the median point. The outer polygon is the convex hull of all points contained within the fence. This is referred to as the *loop*. Points that lie outside the fence are designated outliers, and are shown as points on the plot. For more information about Bagplots, see Rousseeuw (1999).

HDR Draws highest density region rectangles for univariate and contours for bivariate data. The lighter shaded region represents the 99% probability region; the darker shaded region represents the 50% probability region. (Note that the regions can be noncontiguous and may not contain exactly 50% or 99% of the input data.) The density mode within the 50% probability region is represented by a line (univariate data) or an asterisk (bivariate data). If you remove points from the plot, the points that remain represent outliers relative to the 99% probability region. You can adjust the smoothness of the regions using the Smoothness option.

The HDR regions are based on a nonparametric density estimated by applying a Gaussian kernel to the data after the points have been interpolated to a grid. The default bandwidth for determining the smoothness of the HDR regions is computed as follows:

$$\frac{0.9s}{\sqrt[5]{n}}$$

The resulting equally spaced bins are used to compute the density regions. The contours are evenly spaced within the density regions that separate the space into equal-probability regions. For more information about highest density regions, see Hyndman (1996).

Violin (Available only for univariate plots.) Draws a violin plot of the density of the data by plotting symmetric kernel densities around a common vertical axis. The densities are defined using the same procedure as the HDR regions.

Jitter Turns jitter on or off. Jitter adds random noise to data, and can help reduce over-plotting. For descriptions of the specific jitter options, see "Points".

Outliers (Available for Bagplot and HDR contours.) Plots outliers.

Violin Scaling For multiple violin plots, choose a scaling option. Equal area or width means that the areas or maximum width of the violins are the same. Weighted area means that the areas of the violins are proportional to the number of observations in each violin.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Contour

Adapt to Axis Scale For log and other axis transformations, applies computations on the transformed coordinates.

Contour Placement (Available with grouping variables.) Enables you to align or offset the contours.

Line

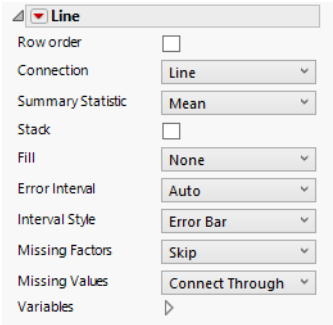
The **Line** element uses line segments to connect summary statistic values (or raw values) of the variable plotted on the Response Axis across the values or categories of another variable.

For an example that combines a line chart with an area plot to emphasize a line, see "Example of an Area and Line Chart".

Notes:

- If any of the rows used in calculating a point on the plot are hidden and not excluded, the point is not shown, but the position that it would occupy remains. The line segment connects the adjoining points.
- To label a line, right-click on the line element in the legend.

Figure 3.30 Line Options



Row order Connects points with line segments in the order of their row numbers.

Connection Changes the connection type to a line, curve, step, centered step, horizontal, or vertical line. If you select a curve, you can also change the smoothness.

Smoothness Changes the smoothness of a Curve connection. To use this option, select Curve for the Connection type.

Summary Statistic Specifies the statistic that is plotted. The statistic is computed for observations at each distinct value of the variable in the X zone (or the Y zone, if X is specified as the Response Axis).

Stack (Available for multiple Y variables.) Accumulates, or stacks, the Y values. Use to build an area chart.

Fill Specify how to fill the area below or between the lines.

Error Interval Adds or removes specified error interval in the graph.

Interval Style Draws error bars or error bands for the selected error interval.

Missing Factors Provides options for connections when all values are missing for a setting of the variable on the X axis:

Skip Skips over the missing factor value, connecting the previous factor to the next factor.

Treat as Missing Skips over the missing factor value, connecting the previous factor to the next factor. Draws the connection in the style specified for Missing Values.

Treat as zero Treats the missing factor values as zero.

Missing Values Provides options for connections when all values are missing for a setting of the variable on the Response axis:

Connect Through Connects the points adjacent to the missing values with the connector type used for the remaining points.

Connect Faded Connects the points adjacent to the missing values with a faded connection.

Connect Dashed Connects the points adjacent to the missing values with a dashed connection.

No Connection Shows no connection between the points adjacent to the missing values.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.
– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Line

Response Axis Specifies the axis for the variable that is used as the response in plotting the line segments. The Auto setting is the Y axis.

Save Formula Saves fit formulas and upper and lower confidence intervals for a single variable to the data table.

Bar

The **Bar** element shows a bar chart of summary statistic values of one or more variables across the values or categories of one or more stratifying variables. The graph can display a single variable or multiple variables of any type arranged in the X and Y zones.

- If you add a single categorical variable, then each bar shows the count of observations in the corresponding level of the variable.
- If you add two variables, with one in each zone, each bar represents the value of the specified summary statistic for the variable on the specified Response Axis. There is a bar for each value or level of the stratifying variable.
- If you add multiple variables in each zone, multiple bars are plotted for each value or combination of levels of the stratifying variables.

Response Axis Variables with Character Data Type

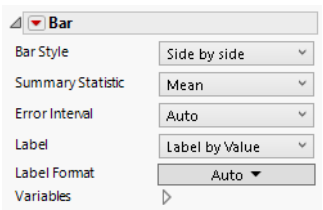
If the variable on the Response Axis has the Character data type, then this variable is assigned numeric values:

- Integer values are assigned based on the Value Order column property or the default value ordering.
- These integer values range from 0 (highest level in value ordering) to the number of levels minus one (lowest value in value ordering).
- These integer values are used in calculating the Summary Statistic.

Note: If any of the rows used in constructing a bar on the plot are hidden and not excluded, the bar is not shown, but the position that it would occupy remains.

For an example of a graph that uses the bar element, see "Example of an Overlaid Histogram and Ridgeline Chart" and "Example of a Packed Bar Chart".

Figure 3.31 Bar Options



Bar Style You can change the appearance of the bars that describe the levels of the stratifying variable. Many bar styles are applicable only when there are merged variables on the response axis, or if an overlay variable is present. For bar style descriptions, see Table 3.1. To change bar widths and space, right-click in the graph and select **Customize** > **Bar**.

Packed Bar Style Options

Packed Primaries Specifies the number of primary bars.

Packed Placement Specifies where the secondary bars are stacked. Options include the following:

First Stack—Start on the top row and fill it before moving to the next row.

Smallest Stack—Place each bar at the stack that is currently the smallest.

Separate Stack—Make a stack called Other where all secondary bars appear.

Packed Ordering Specifies the order in which secondary bars are placed. Can be by size. The biggest bars appear first or by label, which is in natural order and usually alphabetical.

Packed Coloring Specifies how the secondary bars are colored. Options include the following:

Bar color—Applies the same color as the primary bar color.

Faded bar color—Applies a faded version of the primary bar color.

Grays—Applies random shades of light gray.

Packed Labeling Controls the percentage of secondary bars that are labeled.

Packed Primary Labels Specifies where primary labels should be placed, either inside bars or on the axis.

Summary Statistic Specifies the statistic that is used as bar labels and controls the response axis scale. Available statistics depend on the type of bar chart used.

Error Interval Adds or removes error intervals in the graph when applicable, otherwise it is not available. To customize error bars, right-click in the graph and select **Customize** > **Error Bar**.

Label Adds or removes labels in the bar chart. You can label by value, percent of total value, or by row. When **Label by Value** is selected the values are determined by the summary statistic setting. Click and drag to reposition a label.

Label Format (Available for value and percent labels.) Enables you to set the format for the cell labels.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Bar

Response Axis Specifies the axis for the variable that is used as the response in plotting the bars. This option is available only when each axis contains at least one variable with a Continuous modeling type. If only one variable is Continuous, its axis is the response axis. The Auto setting is the Y axis.

Save Formula Saves fit formulas and upper and lower confidence intervals for a single variable to the data table.

Overlap Specifies the amount of overlap (Auto, None, Half, Full) between multiple bars.

Table 3.1 Bar Styles and Descriptions

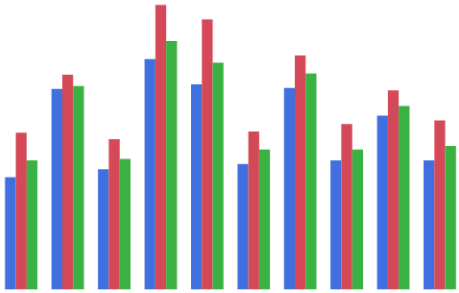
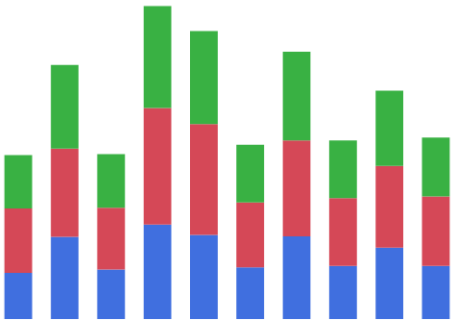
Bar Style	Description
Side by side	When you have two or more merged variables on the response axis, a bar is plotted for each of the merged variables. The bars are arranged next to each other for each level of the stratifying variables. They are colored and a legend appears. 
Stacked	When you have two or more merged variables on the response axis, a single bar is plotted for each level of the stratifying variables. Each bar stacks the bars for each of the merged variables. The part of the bar corresponding to each merged variable is colored and a legend appears. 

Table 3.1 Bar Styles and Descriptions *(Continued)*

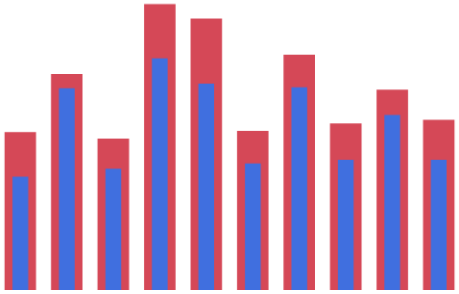
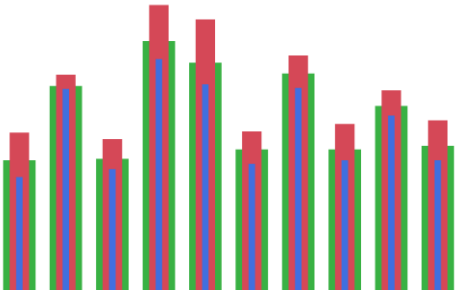
Bar Style	Description
Bullet	When you have two or more merged variables on the response axis, a narrow bar is plotted within a wider and taller bar. The widths of the bars depend on the order of the variables. The part of the bar corresponding to each merged variable is colored and a legend appears. If you have only a single variable, this style plots narrow bars. 
Nested	When you have two or more merged variables on the response axis, the bars for each variable are nested within each other. The narrowest bar corresponds to the first variable listed in the legend, the next narrowest to the second, and so on. **Tip:** You can change the legend order using the arrows in the Variables panel. 

Table 3.1 Bar Styles and Descriptions *(Continued)*

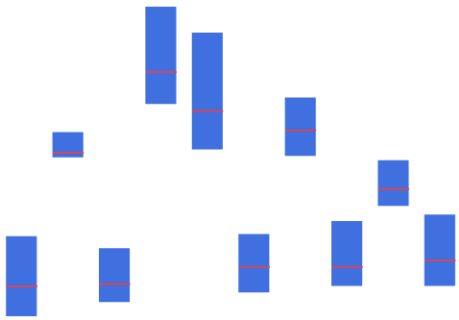
Bar Style	Description
Range	When you have two or more merged variables on the response axis, a rectangle and a line are plotted for each level of the stratifying variables. For each level of the stratifying variables: • The values of the summary statistic are calculated for each of the merged variables. • A rectangle is plotted with bounds equal to the summary statistic values for the first two of variables, as indicated in the legend. • Lines are plotted at the values of the summary statistic for the other variables. 

Table 3.1 Bar Styles and Descriptions *(Continued)*

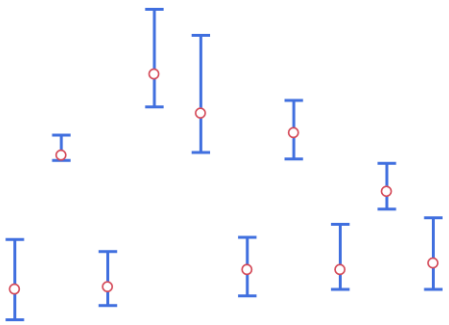
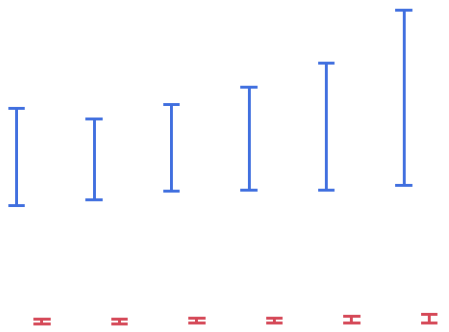
Bar Style	Description
Interval	When you have two or more merged variables on the response axis, a line with boundaries and a circle is plotted for each level of the stratifying variables. For each level of the stratifying variables: • The values of the summary statistic are calculated for each of the merged variables. • A line with boundaries is plotted with bounds equal to the summary statistic values for the first two of variables, as indicated in the legend. • A circle is plotted at the overall values of the summary statistic for the other variables. 
Side by side intervals	In this style, each pair of variables defines an interval. 

Table 3.1 Bar Styles and Descriptions *(Continued)*

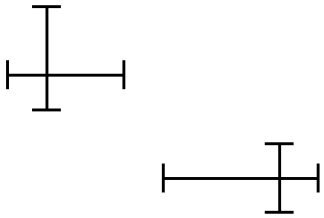
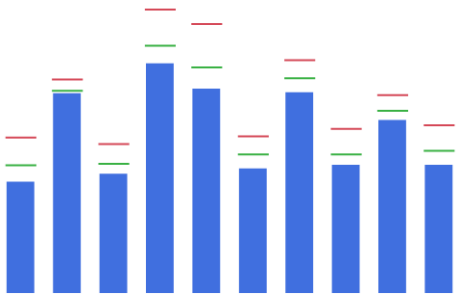
Bar Style	Description
Two-way interval	This style is for two-dimensional bar charts with bars in each direction. 
Single	For each level of the stratifying variables, a bar is plotted. When you have two or more merged variables on the response axis, the bar represents the value of the summary statistic for the first variable in the legend and lines are shown for the values of the summary statistic for the remaining variables. **Tip:** You can change the legend order using the arrows in the Variables panel. 

Table 3.1 Bar Styles and Descriptions *(Continued)*

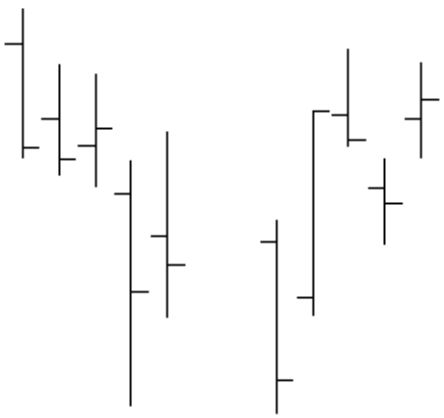
Bar Style	Description
Stock	When you have two or more merged variables on the response axis, line segments are plotted at the levels of the stratifying variables. For each level of the stratifying variables: • The values of the summary statistic are calculated for each of the merged variables. • A line segment connecting the summary statistic values for two of the variables, as indicated in the legend. • Bars are plotted at right angles to the line segment at the remaining values of the summary statistic. 

Table 3.1 Bar Styles and Descriptions *(Continued)*

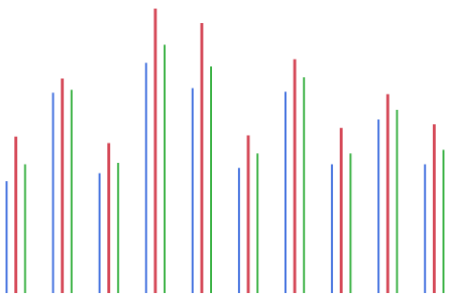
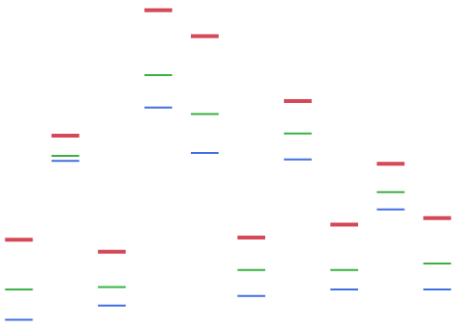
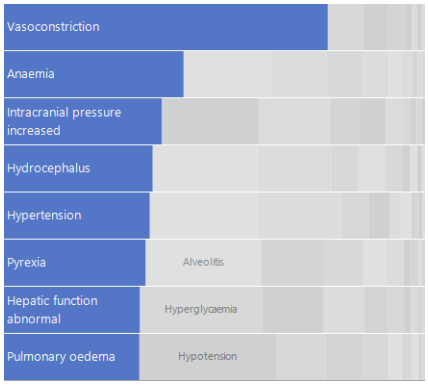
Bar Style	Description
Box Plot	When you have two or more merged variables on the response axis, box plots are shown for the levels of the stratifying variables. For each level of the stratifying variables: • The values of the summary statistic are calculated for each of the merged variables. • An outlier box plot is constructed for these values.
Needle	When you have two or more merged variables on the response axis, needle-shaped bars are plotted side-by-side for each of the merged variables. The bars are colored and a legend appears. This style is useful when you have many levels of the stratifying variables. 

Table 3.1 Bar Styles and Descriptions *(Continued)*

Bar Style	Description
Float	When you have two or more merged variables on the response axis, bars are plotted at the levels of the stratifying variables. For each level of the stratifying variables: • The values of the summary statistic are calculated for each of the merged variables. • A bar is plotted at each of these values. This style is useful when you have many merged variables. 
Packed	This bar style shows the top categories as a bar chart and stacks the other categories as bars with an approximately rectangular fill. The top categories are the focus and the other bars provide context. This style is useful when you have many categories and skewed response data. 

Area

Note: You can also create an area plot using the Line element .

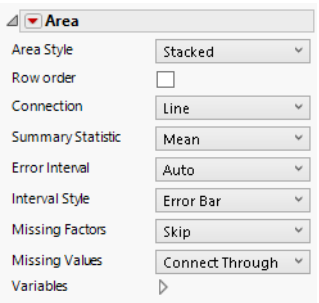
The **Area** element shows a contiguous area that represents the area beneath summary statistic values of a variable across the values or categories of another variable. The Area element can be considered a continuous representation of the Bar element.

- Both variables might be nominal or ordinal. The summary statistic represents the count of observations in a level of the variable on the axis that is not specified as the Response Axis.
- One of the variables might be continuous and the other nominal or ordinal. The summary statistic represents the value of the specified summary statistic for the continuous variable for the corresponding level of the categorical variable.
- If both variables are continuous, the summary statistic is computed for the continuous variable on the axis that is specified as the Response Axis.

Note: The plot is not affected in any way if rows are hidden and not excluded.

For an example of a graph that uses the Area element, see "Example of an Area and Line Chart".

Figure 3.32 Area Options



Area Style Changes how the area is displayed when there are merged variables on the response axis. For area style descriptions, see Table 3.2.

Row order Connects points with line segments in the order of their row numbers.

Connection Changes the connection type to a line, curve, step, centered step, horizontal, or vertical line. If you select a curve, you can also change the smoothness.

Summary Statistic Specifies the statistic that is plotted. The statistic is computed for observations at each distinct value of the variable in the X zone (or the Y zone, if X is specified as the Response Axis).

Error Interval Adds or removes specified error interval in the graph.

Interval Style Draws error bars or error bands for the selected error interval.

Missing Factors Provides options for connections when all values are missing for a setting of the variable on the X axis:

Skip Skips over the missing factor value, connecting the previous factor to the next factor.

Treat as Missing Skips over the missing factor value, connecting the previous factor to the next factor. Draws the connection in the style specified for Missing Values.

Treat as zero Treats the missing factor values as zero.

Missing Values Provides options for connections when all values are missing for a setting of the variable on the Response axis:

Connect Through Connects the points adjacent to the missing values with the connector type used for the remaining points.

Connect Faded Connects the points adjacent to the missing values with a faded connection.

Connect Dashed Connects the points adjacent to the missing values with a dashed connection.

No Connection Shows no connection between the points adjacent to the missing values.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Area

Response Axis Specifies the axis for the variable that is used as the response in plotting the areas. Available only when both variables have a Continuous modeling type. If only one variable is Continuous, its axis is the response axis. The Auto setting is the Y axis.

Save Formula Saves fit formulas and upper and lower confidence intervals for a single variable to the data table.

Table 3.2 Area Styles and Descriptions

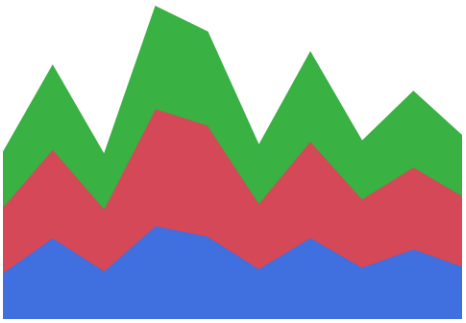
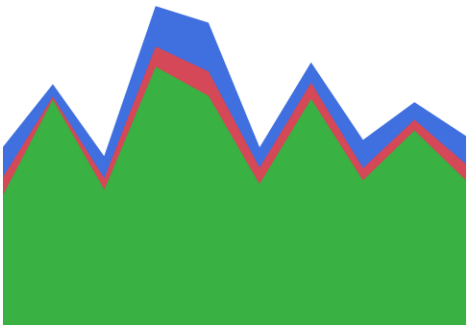
Area Style	Description
Stacked	Shows areas that correspond to summary statistics for each of the merged variables. The plots are colored and a legend appears in the Legend zone. 
Overlaid	Shows the area plots for the merged variables superimposed over each other. The plots are colored and a legend appears in the Legend zone. 

Table 3.2 Area Styles and Descriptions *(Continued)*

Area Style	Description
Range	Shows the area corresponding to the difference in the summary statistic for the merged variables.
Stacked Range	Shows the areas corresponding to the difference in the summary statistic after the variables are stacked.

Box Plot

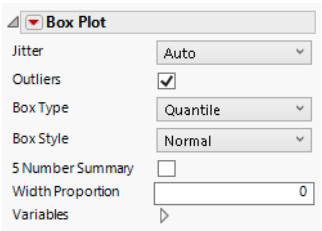
The **Box Plot** element shows outlier or quantile box plots. A box plot provides a compact view of a distribution of values. The box extends from the 25th percentile to the 75th percentile where the distance between the 75th and 25th percentiles is the interquartile range (IQR). The median is marked within the box. For outlier box plots, whiskers extend to the last point that is within 1.5*IQR from the ends of the box. For quantile box plots the whiskers cover the full range of observations with tick marks at specific quantiles. For more information about outlier and quantile box plots, see *Basic Analysis*.

- If one of the variables is continuous and the other is nominal or ordinal, box plots are constructed for values of the continuous variable. There is a box plot for each level of the categorical variable.

- If both variables are continuous, box plots are constructed for values of the continuous variable on the axis specified as the Response Axis. There is a box plot for each level of the other continuous variable.

For an example of a graph using box plots, see "Example of Features in Graph Builder".

Figure 3.33 Box Plot Options



Jitter Turns jitter on or off. Jitter adds random noise to data, and can help reduce over-plotting.

Outliers Shows or hides values that extend beyond the whiskers.

Box Type Specifies whether the box plot is an outlier box plot or a quantile box plot.

Box Style Changes the style the box plot. For box style descriptions, see Table 3.3.

5 Number Summary Adds statistics like the median, maximum, minimum, and quantiles 1 and 3 to the graph.

Width Proportion adjusts the width of the box plots.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Options for Box Plots

Confidence Diamond Shows or hides a confidence diamond about the mean. The diamond covers the 95th confidence interval for the mean.

Notched Displays a box plot that is notched at the median where the notches span:

$$\text{median} \pm (1.58 \times \text{IQR}) / (\sqrt{n})$$

Fences Shows or hides vertical lines, or fences, at the end of the box plot whiskers.

Shortest Half Shows or hides a bracket that indicates the location of the shortest interval that contains 50% of the data.

Shortest Half Color Enables you to select a color for the shortest half bracket.

Response Axis (Available only when the X and Y axes variables both have a continuous modeling type.) Specifies the axis for the variable that is used as the response in constructing the box plots. Available only when both variables have a continuous modeling type. If only one variable is continuous, its axis is the response axis. The Auto setting is the Y axis.

Box Placement (Available with grouping variables.) Enables you to align or offset the box plots.

Tip: When you have a categorical grouping variable for your box plot, right-click on the categorical axis and select **Size By > Count** to draw box plots that are proportional to the number of observations in each category.

Table 3.3 Box Styles and Descriptions

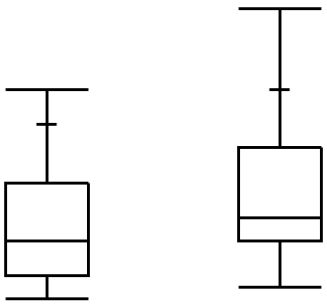
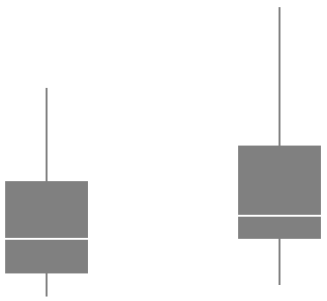
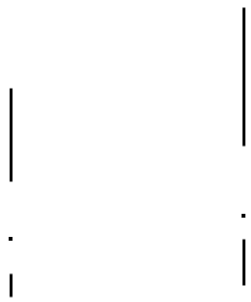
Box Style	Description
Normal	Shows the box defined by the quartiles as hollow with only a line at the median. 
Solid	Shows the box defined by the quartiles with solid fill. The line at the median is shown by white space. 

Table 3.3 Box Styles and Descriptions *(Continued)*

Box Style	Description
Thin	The box defined by the quartiles is not shown. It is discernible due to the whiskers. The median is denoted by a point.



Histogram

The **Histogram** element shows a variable's distribution using binning. For more information about histograms, see *Basic Analysis*.

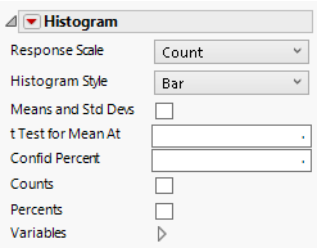
Note: The bars of the histogram align with the tick marks on the axis for the zone containing the variable. If you customize the axis, the histogram adjusts to match the new tick marks.

You can change the primary direction of the graph to X (horizontal), Y (vertical), or Auto using the Response Axis option.

- If both variables are nominal or ordinal, the histogram shows the count of observations in a level of the variable on the axis that is not specified as the Response Axis.
- If one of the variables is continuous and the other is nominal or ordinal, the plot shows a histogram for the continuous variable for each level of the categorical variable.
- If the same variable is specified for both *X* and *Y*, then the *Y* role is ignored and a single histogram appears.
- If both variables are continuous, then the variable on the X axis is treated as discrete. You might need to rescale the X axis to view the graph clearly. For an alternative visual, use a scatterplot with the Contour element.

Tip: You can overlay histograms with transparent color so that you can see the individual histograms. To do this, assign the variable of interest as X or Y. Assign your overlay variable to the **Overlay** zone. Then, click the **Histogram** element icon. See "Example of an Overlaid Histogram and Ridgeline Chart".

Figure 3.34 Histogram Options



Response Scale Specifies the scale for the response axis. The options are count, percent, or fill. Fill has no response axis label, fills the display space, and scales multiple histograms independently.

Overlap (Available with a categorical response.) Adjusts the amount of histogram overlap between vertical categories.

Histogram Style Specifies the histogram style.

Bar Default traditional histogram bars with a height based on the number or percent of observations that fall within each bar or bin.

Polygon Connects the peaks of each histogram bar to construct a polygon representation of the distribution of the data.

Kernel Density Density curve with a smoother control.

Shadowgram Overlays histograms with different bin widths.

Means and Std Devs Shows the means and standard deviations for the levels of the variables in the X or Y zone.

t Test for Mean At Performs a *t* test for the specified mean.

Confid Percent Specifies the confidence interval for the mean.

Counts Displays counts on the histogram bars.

Percents Displays percents on the histogram bars.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Option for Histogram

Response Axis Specifies the axis for the variable that is used as the response in constructing the histograms. Available only when both variables have a Continuous modeling type. If only one variable is Continuous, the other axis is the response axis.

Heatmap

The **Heatmap** element shows the counts or average values for groups of observations using a color intensity scale. If you plot a single variable as either Y or X, your plot appears as bars. If you plot a cross tabulation of two variables Y and X, your plot shows rectangles.

- For a categorical variable, the levels define the groups of observations.
- For a continuous variable, Graph Builder constructs non-overlapping intervals of values. These intervals define the groups of observations.

Note: If all of the rows used in constructing a bar or rectangle on the plot are hidden and not excluded, the rectangle is not shown, but the position that it would occupy remains.

When you create a heatmap, an intensity legend appears in the Legend area. See "Discrete or Continuous Legend Items". The intensity legend shows counts or average values:

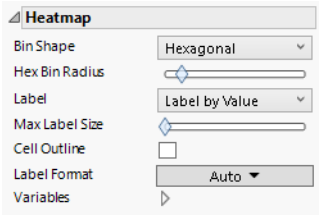
- If no Color variable is assigned, the counts in each bar or rectangle define the scale for the intensity legend.

- If a Color variable is assigned, the average of that variable for the observations in each bar or rectangle define the scale for the intensity legend.

Tip: Hover over a cell to see labels. Click a cell to select the corresponding rows.

To color or size a heatmap by two variables, drag the first variable into the Color or Size zone and then drag the second variable into the top or bottom corner of the zone.

Figure 3.35 Heatmap Options



Bin Shape (Available for continuous variables). Specifies the shape of the bin.

Hex Bin Radius (Available for hexagonal bins.) Enables you to adjust the radius of the bin.

Label Specifies labels for the heatmap. You can label by value, percent of total, or by row.

Max Label Size Enables you to adjust the size of the label.

Cell Outline (Available when a size variable is used.) Enables you to add outlines for the original bin sizes. This highlights the size scaling on the bins.

Label Format (Available for value and percent labels.) Enables you to set the format for the cell labels.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Use a Heatmap to Apply Background Colors

You can also use the heatmap to control the background color when you have plot elements that use the Wrap, Group X, or Group Y zones. To do so:

1. Select the variable for the background color and drag it to the **Color** zone.
2. Select the Heatmap element .
3. Construct the plot elements.

In the Heatmap panel, under Variables, deselect all check boxes except the one for Color.

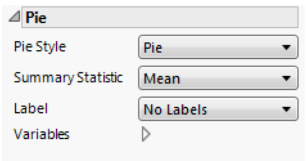
Pie

The **Pie** element plots summary statistics for groups of observations, representing their values as the size of the slices or rings on a pie chart.

Note: If any of the rows used in constructing a slice of the pie chart are hidden and not excluded, the slice is not shown, but the position that it would occupy remains.

- For a single variable, the slice sizes the number of observations in each category.
- For two variables, the variable in the Y zone is used to size the slices according to the selected Summary Statistic. A legend appears in the Legend zone showing the values of the variable in the X zone.

Figure 3.36 Pie Options



When you add an Overlay variable, a ring chart is constructed for each level of the Overlay variable. The outer ring represents the smallest value in the value ordering, and the inner ring represents the largest value.

Pie Style Changes the appearance of the pie chart. For pie style descriptions, see Table 3.4.

Summary Statistic Changes the statistic that is plotted.

Label Adds or removes labels in the pie chart.

Label Format (Available for value and percent labels.) Enables you to set the format for the cell labels.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Table 3.4 Pie Style Descriptions

Pie Style	Description
Pie	Traditional pie chart with each slice sized by the Summary Statistic.

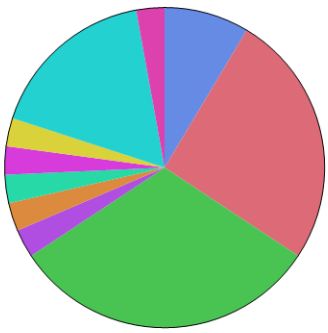
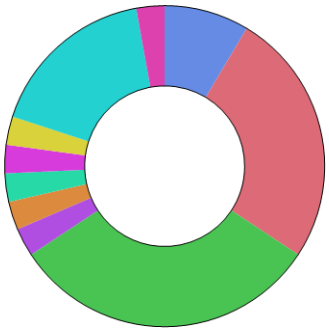
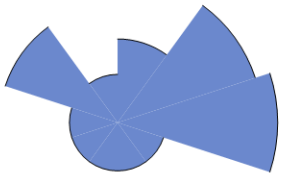


Table 3.4 Pie Style Descriptions *(Continued)*

Pie Style	Description
Ring	Each variable or level of a stratifying variable is represented by a concentric ring. The sections are sized by the Summary Statistic. Ring charts can help you visualize hierarchical data, using concentric rings. 
Coxcomb	The central angles for all slices are equal. The size of each slice is determined by the Summary Statistic. Also known as an *exploded pie chart*, a coxcomb chart can help you see smaller areas. 

Treemap

The **Treemap** element shows the levels or values of a variable as rectangles within a rectangular display. The rectangle sizes represent summary statistic values of the variable in the **Size** zone across the levels or values of one or more X variables.

To construct a treemap that shows summary values across the levels of two or more X variables, merge the variables in the X zone. The rectangles are sized to represent the values of the summary statistic without leaving unused space in the overall rectangular layout.

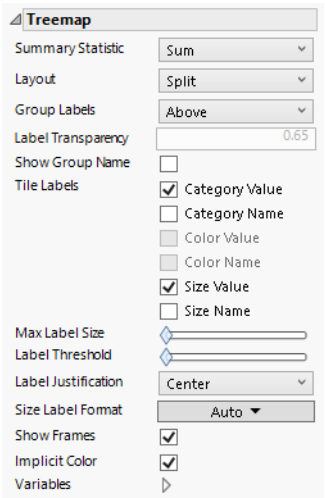
By default, the data table order is used when ordering the tiles in the treemap using the Split layout. Alternatively, you can specify one or two continuous variables as Y variables for ordering:

- If one Y variable is specified, it orders both the X and Y axes.
- If two Y variables are specified, the first orders the X axis and the second orders the Y axis.

If the Squarify layout is specified, then the order of the tiles is determined by the size, and ordering variables are ignored.

Note: The plot is not affected in any way if rows are hidden and not excluded.

Figure 3.37 Treemap Options



Summary Statistic Specifies the summary statistic, corresponding to the variable in the Y zone, that is used to size the rectangles.

Layout Arranges rectangles to the extent possible by the value ordering of the X variable or by the size of the rectangle.

Split Arranges the rectangles according to the value ordering of the levels or values of the X variable. Split is the default setting.

Squarify Arranges the rectangles according to the values of the summary statistic, sorted in descending order. This places the largest rectangles in the top left corner of the plot and the smallest in the bottom right corner.

Mixed Applies only when you have two or more X variables. It applies Split to the outermost variable and Squarify to the other variables. It follows that the large groupings are ordered according to their value ordering, whereas the inscribed rectangles are ordered according to the summary statistic values.

Group Labels (Available when you have more than one X variable.) Shows or hides the additional group labels above each category or floating in the center of each category.

Tip: To filter data in a group, click a group label.

Label Transparency (Available with floating group labels.) Specifies the transparency of the floating group labels.

Show Group Name Adds the variable title to the group labels.

Tile Labels Shows or hides the following labels:

Category Value The values of the X variable.

Category Name The name of the X variable.

Color Value The values of the Color variable.

Color Name The name of the Color variable.

Size Value The values of the Size variable, or the Y variable if no Size variable exists (Size Value).

Size Name The name of the Size variable, or the Y variable if no Size variable exists (Size Name).

Tip: Hover over a rectangle to see details.

Max Label Size Increases or decreases the size of the labels.

Label Threshold Removes labels based on the size of the rectangle. By default, all labels are shown. Move the slider to the right to remove labels for progressively larger rectangles.

Label Justification Places the label in the center, left, or right.

Color Label Format (Available when a color variable is used.) Enables you to set the format for the cell labels based on the color variable.

Size Label Format (Available when Size Value is selected.) Enables you to set the format for the cell labels based on the size value variable.

Show Frames Shows or hides the borders around the rectangles.

Tip: Use the treemap option in the right-click customize menu to adjust frame color, style, and thickness.

Implicit Color Shows or hides the color. If no variable is specified in the Color zone, all rectangles have the same color.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

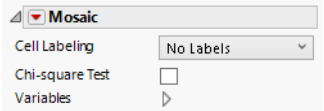
Mosaic

The **Mosaic** element uses rectangle sizes to represent the proportion of observations in categories for one variable across categories of another variable. Although variables can be categorical or continuous, they are treated as categorical. For more information about mosaic plots, see *Basic Analysis*.

The default Response Axis is the Y axis. This means that, for each level of the X variable, vertical stacked bars show the proportion of values in each level of the Y variable. You can change the primary direction of the graph to X (horizontal).

Note: The plot is not affected in any way if rows are hidden and not excluded.

Figure 3.38 Mosaic Options



Cell Labeling Add labels to cells based on counts, percents, values, or row.

Label Format (Available for percent or value labels.) Enables you to set the format for the cell labels.

Chi-square Test Performs a Pearson chi-square test and shows results in a label on the graph. This option appears if each side has 2 or more levels.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Option for Mosaic

Response Axis Changes the axis that is associated with the response variable to X (horizontal), Y (vertical), or Auto (the Y axis for Mosaic). The variable on the Response Axis is used to calculate the proportion of observations in each level of the variable on the other axis.

Caption Box

The **Caption Box** element displays summary statistics on the plot. The summary statistic corresponds to the variable that corresponds to the axis that you specify as the Response Axis in the Caption Box options.

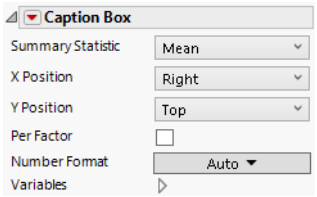
Summary Statistic for Categorical Variables

For a categorical variable, the Summary Statistic that appears is calculated by assigning numerical values to the levels of the categorical variable. The levels are placed in the order defined by their value ordering. They are assigned integer values ranging from 0 to the number of levels minus one.

Note: When using categorical values in zones associated with summarization roles like Y and Color, summary statistics are calculated based on numerical values assigned to the variable levels. (The integer values range from 0 to the number of levels minus one.) The final result is then mapped back as a level to a corresponding categorical value or value range. This approach allows the use of ordinal variables in summarization, but might not lead to clear results for nominal variables.

For an example illustrating the Caption Box element, see "Example of Features in Graph Builder".

Figure 3.39 Caption Box Options



Summary Statistic Specifies the summary statistic to be displayed for the variable defined by the Response Axis setting. The default Summary Statistic is the Mean. For more information about how summary statistics are computed for categorical variables, see "Summary Statistic for Categorical Variables".

X Position Specifies the horizontal position of the caption.

Y Position Specifies the vertical position of the caption.

Per Factor Shows a caption for each X, or each Y if horizontal.

Number Format Enables you to set the format of the summary statistic.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Option for Caption Box

Response Axis Specifies the axis for the variable whose summary statistic is displayed. The Y axis is the default. Notice that this axis can differ from the Response Axis specified to define the plot.

Formula

The **Formula** element shows the graphs of a function, its inverse, or a parametric curve. The function or functions are defined by column formulas.

- $y = f(x)$:
  - To plot a function of a single argument across the values of its single argument, drag the column that contains the formula to the Y zone and the column that contains the values of its single argument to the X zone.
  - Make sure that the Response Axis is set to Auto or Y.
- $y = f^{-1}(x)$:
  - To plot the inverse of a function of a single argument for a column of values, drag the column that contains the formula to the X zone. Drag the column that contains the values of interest to the Y zone.
  - Set the Response Axis to X.
- $x = f(t)$ and $y = g(t)$: To plot the parametric curve defined by two functions, drag one to the X zone and the other to the Y zone.

Note: Because the Formula element plots the values of a function, the plot is not affected in any way if rows are hidden and not excluded.

Figure 3.40 Formula Option

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Red Triangle Option for Formula

Response Axis Specifies the axis for the formula variable.

Map Shapes

The Map Shapes element shows areas defined by the variable in the Map Shape zone.

When a column in your data table contains the names of geographical regions (such as countries, states, provinces, counties), you can assign the column to the Map Shape zone. When a variable is dropped in the Map Shape zone, Graph Builder searches for internal maps that correspond to the values in the column. If it finds an appropriate map, it draws the map in the graph area.

Alternatively, you can define the Map Role column property for the column of interest. This property tells JMP how to connect the values in the column with map shape data. It is especially useful when you create custom maps. See "Custom Map Files".

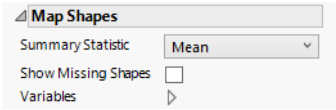
If you have a variable in the Map Shape zone, the X and Y zones disappear. You can add information from other variables using Color and Size.

- To color the map shapes by the values of a summary statistic, drag the column of interest to the **Color** zone. The categorical or continuous color theme selected in your Preferences is applied to each shape.

- To size the map shapes by the values of a summary statistic, drag the column of interest to the **Size** zone. This scales the map shapes according to the summary statistic value of the size variable, minimizing distortion.

See "Maps". For examples, see "Examples of Creating Maps", or run the associated scripts in these sample data tables: PopulationByMSA.jmp or SAT.jmp.

Figure 3.41 Map Shapes Options



Summary Statistic Changes the statistic that is plotted as the Color or Size variable.

Show Missing Shapes Shows or hides missing data from the map.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

– Show or hide the elements corresponding to a variable in a zone.

– Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Parallel

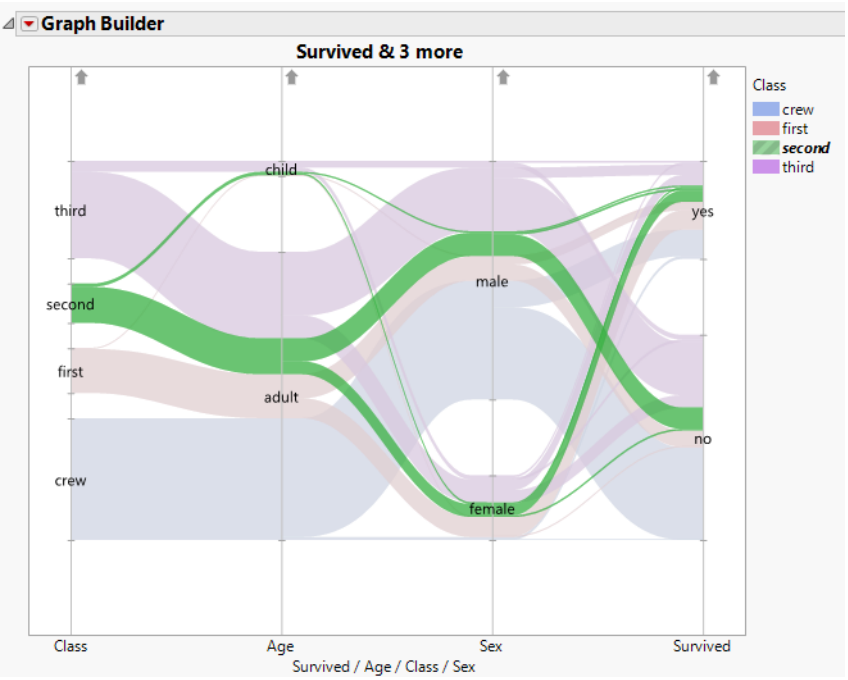
The **Parallel** element connects the values in a row across two or more variables. Drag two or more variables together to either the X or Y zone. The variable names appear as axis labels in the zone to which they were dragged.

- Values for each continuous variable are plotted along lines (axes) parallel to the other axis.

- Levels of categorical variables are represented by intervals on lines (axes) parallel to the other axis. The sizes of the intervals are proportional to the number of observations in each level of the categorical variable.
- The values (for continuous variables) or bands (for categorical variables) are joined with continuous curves.

When all variables are categorical, there is a band for every combination of levels of the categorical variables. The bands split as they move from left to right. In the first interval corresponding to a categorical variable, there is a band for every level of that categorical variable. In the last interval corresponding to categorical variables, there is a band for every possible combination of the categorical variables.

Figure 3.42 Example of Categorical Bands Using Titanic.jmp



In Figure 3.42, the band containing all second-class passengers is selected. The parallel plot shows that most were adults, there were more males than females, and slightly fewer survived than did not survive.

The values or levels of the variables are connected with lines or curves using the Curve Lines option.

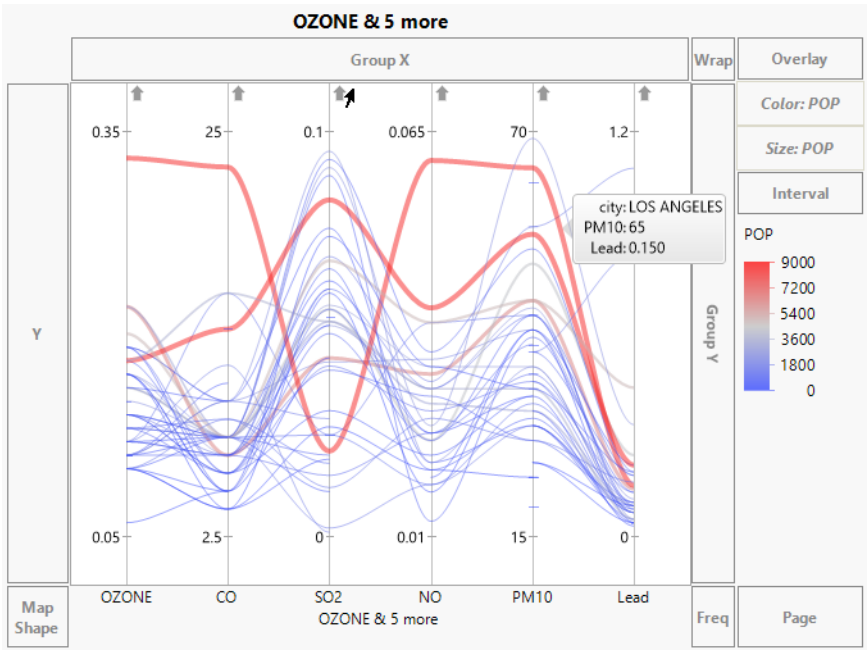
- To color the curves by the values of a variable, drag the column of interest to the **Color** zone. The categorical or continuous color theme selected in your Preferences is shown in the Legend zone.

- To size the curves by the values of a variable, drag the column of interest to the **Size** zone.
- To change the direction of a variable's axis, click the arrow at the top of the vertical line denoting the variable's axis.
- To move a variable and its axis, click and drag the axis.

Tip: Hover over a curve to see a label giving information about the corresponding row.

Figure 3.43 shows a parallel plot for six variables in the Cities.jmp data table. The variable POP is used both as a Color and Size variable. The curve for Los Angeles is labeled.

Figure 3.43 Parallel Plot for Pollution Data in Cities.jmp



Tips:

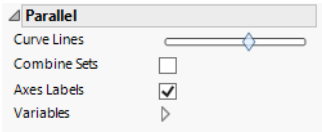
- You can add a line element to a parallel plot to show a summary relationship across the parallel plot.
- You can add reference lines for specification limits. For information, see *Using JMP*.

Scaling

By default, the scales for the values of the variables are adjusted so that the minimum and maximum values are plotted at the same level. For example, in Figure 3.43, the values of each of the variables have an identical vertical spread. Each vertical line is labeled by the minimum and maximum values of the variables.

In Figure 3.43, the scales for CO and PM10 differ greatly from the scales of the other variables. When your variables are measured on very different scales, this scaling enables you to see differences clearly.

Figure 3.44 Parallel Options



Curve Lines Adjusts the amount of curvature of the curves that connect points. Place the slider all the way to the left for lines. As you move the slider to the right, the degree of curvature increases.

Combine Sets (Applicable only if you have three or more categorical factors.) For categorical variables, when Combine Sets is not selected, bands split as they move from left to right. In the last interval that corresponds to a categorical variable, there is a band for every possible combination of the categorical variables. Selecting the Combine Sets option causes the bands not to split. Each interval that follows a categorical variable shows a band for each level of that variable.

Axes Labels Removes the axis labels from the display.

Variables Shows or hides graph elements for variables, or re-orders the display of variables.

Note: These options do not apply to variables in the Group X, Group Y, Wrap, or Page zones.

Check boxes are followed by the zone designation and the name of the variable. Use check boxes to do the following:

- Show or hide the elements corresponding to a variable in a zone.
- Add or remove the effect of applying the Color, Size, Shape, or Freq variable to the variable in the zone.

Tip: If you have multiple graphs, you can color or size each graph by different variables. Drag a second variable to the **Color** or **Size** zone, and drop it in a corner. In the Variables option, select the specific color or size variable to apply to each graph.

Use arrows to re-order the display if there are multiple variables in a zone. Highlight a variable name and click an arrow to reposition it.

For an example using Variables, see "Example of an Area and Line Chart".

Graph Zones

The main element in the Graph Builder window is the graph area. The graph area contains zones, and you drag and drop variables from the Variables box into the zones.

Tip: If you drop variables into the center of the graph, JMP guesses which drop zone to put them into, based on whether the variables are continuous, ordinal, or nominal.

There are two types of zones into which you can drag variables:

- *Data zones* include **X**, **Y**, **Map Shape**, **Freq**, **Color**, **Size**, and **Interval**. The X, Y, and Map Shape zones are positional, and influence the types of graph elements that are available. The Freq, Color, Size, and Interval zones modify certain graph elements.
- *Grouping zones* partition the data into subsets that repeat the graph for each possible subset. Grouping zones include **Group X**, **Group Y**, **Wrap**, **Overlay**, and **Page**.

The following zones are available in Graph Builder:

Table 3.5 Graph Builder Zones

Zone	Description
X and Y	Constructs plots that are based on the values, levels, or categories of the variables that you drop into these zones. Use the X zone for a horizontal layout and the Y zone for a vertical layout.

Table 3.5 Graph Builder Zones *(Continued)*

Group X and Group Y	Stratifies your data by the levels of the nominal or ordinal variable and constructs a plot for each level. Group X produces a horizontal display, and Group Y produces a vertical display. The type of variable that you put in Group X or Y also determines aspects of the display: • Adding a continuous variable creates non-overlapping interval subgroups and plots for the values in each interval. • Adding a categorical variable creates plots for each level of the categorical variable. You can add multiple variables to Group X or Group Y: • To add a variable *above* a variable in one of the Group zones, drag it to the left of that variable in the Group X zone or above that variable in the Group Y zone. • To add a variable *below* a variable in one of the zones, drag it to the right of that variable in the Group X zone or below that variable in the Group Y zone. To stratify by the levels of two variables, drag a variable into both Group X and Group Y. To change the number of levels that appear, right-click in the axis border and select Levels in View. For a description of all options, see "Options for Axes, Axes Variable Labels, or Graph Titles".

Table 3.5 Graph Builder Zones *(Continued)*

Map Shape	Adds a map. Your variable must contain geographic place names, such as countries, regions, states, or provinces, or you must provide information in a custom shape file. Note the following: • If you drag and drop columns labeled Latitude and Longitude, JMP automatically assigns them to the Y and X roles (respectively). • When you drop a variable into the Map Shape zone, the X and Y zones disappear. • The Map Shape element can be used with the Points element to place points within the shape. For more information about creating map shapes, see "Red Triangle Option for Formula" and "Graph Builder".
Wrap	Creates rows of plots for the levels of a categorical variable or for the interval subgroups of a continuous variable. Use Wrap when your stratifying variable has many levels. To change the number of levels that appear, right-click in the axis border and select **Levels in View**. For a description of all options, see "Grouping Zones Options". **Note:** Once a variable is placed here, no variable can be placed in Group X.
Overlay	Subgroups and colors the plot elements according to the levels of the Overlay variable. • Adding a nominal or ordinal variable stratifies and colors the plot elements by the levels of the variable. • Adding a continuous variable creates interval subgroups. The plot elements are stratified and colored according to the subgroups. A legend appears to the right of the plot. Right-click on legend items for customization options.

Table 3.5 Graph Builder Zones *(Continued)*

Color	Adds color to points, map shapes, and other objects. • Adding a nominal or ordinal variable colors the plot elements by the variable's levels. • Adding a continuous variable colors the plot elements according to an intensity scale. A legend appears to the right of the plot. Right-click in the legend to change colors or to customize the intensity scale. **Tip:** Once you have added a variable in the Color zone, you can show or hide color using the Variables option in the element properties panel.
Size	Sizes graph elements by a summarizing statistic or other size variable. A legend appears to the right of the plot. Right-click on legend items for customization options.
Interval	Add interval variables to draw custom error bars. • A single interval variable is treated as a delta to compute upper and lower ends of the interval. • Two interval variables are treated as lower and upper values for the interval.
Freq	Add a frequency or weight variable. When all values are integer values, the Freq variable is treated as a frequency, otherwise it is treated as a weight. A footer note indicates how the variable is treated. The Freq variable affects summary statistics.

Table 3.5 Graph Builder Zones *(Continued)*

Page	Drop a By group variable to the Page zone to show each level of the group on a separate graph. By default, the Page zone is limited to 200 pages. The first 200 categories are displayed. Adjust this limit in the Graph Builder preferences. See *Using JMP*. **Tips:** • Once you click Page, you can select the red triangle menu option Link Page Axes. Use this option to link or unlink graph axis scales across levels of the By group variable in the Page zone. • To copy individual graphs using the selection tool, first save the Graph Builder output to a journal (Edit > Journal).

For more information about the legend, see "Legend Options".

Note: When you use categorical values in zones associated with summarization roles like Y and Color, summary statistics are calculated based on numerical values assigned to the variable levels (integer values ranging from 0 to the number of levels minus one). The final result is then mapped back as a level to a corresponding categorical value or value range. This approach allows the use of ordinal variables in summarization, but might not lead to clear results for nominal variables.

Categorical and Continuous Variables in Group X or Y Zones

When a categorical variable is added to the Group X or Group Y zone, a partition is created for each level of the variable.

When a continuous variable is added to a grouping zone, Graph Builder uses quantiles of the data to divide the variable into five groups. To change the number of groups:

1. Right-click the grouping variable label and select **Number of Levels**.
2. Type in the number of levels that you want to display.
3. Click **OK**.

Move Group X and Group Y Labels

Grouping variable labels can be relocated to another position on the graph. The Group X labels can be either on the top or the bottom of the graph. The Group Y labels can be either on the right or the left of the graph.

- To relocate a Group X label, right-click the variable in the Group X zone and select **X Group Edge > Top** or **Bottom**.
- To relocate a Group Y label, right-click the variable in the Group Y zone and select **Y Group Edge > Left** or **Right**.

Red Triangle Options

The red triangle menu for Graph Builder contains these options:

Show Control Panel Shows or hides the platform buttons, the Select Columns box, and the drop zone borders.

Show

Title Shows or hides the graph title. Right-click to change the alignment or the span of the title.

Subtitle Shows or hides the graph subtitle. Right-click to change the alignment or the span of the subtitle.

Legend Shows or hides the legend.

Footer Shows or hides the footer, which contains informative messages such as missing map shapes, error bar notes, frequency notes, and WHERE clauses.

X Axis Shows or hides the X axis.

Y Axis Shows or hides the Y axis.

X Axis Title Shows or hides the X-axis title.

Y Axis Title Shows or hides the Y-axis title.

Legend Position Sets the position of the legend. The legend appears on the right by default. Putting the legend at the bottom places it in the center below the graph. The legend items then appear horizontally instead of vertically. Or, you can put the legend inside the graph, on the left or right.

Legend Settings Opens a window where you can modify legend settings, such as the title, font, and title position.

Color Settings Contains the following options:

Continuous Color Theme Selects the color theme for continuous variables.

Categorical Color Theme Selects the color theme for categorical variables.

Note: For more information about color themes, see *Using JMP*.

Lighten large fills Automatically lightens colors for elements that fill large areas, such as pie, treemap, and mosaic plots.

Use row colors for levels When every level has a unique color, initialize legend levels with row colors.

Lock Scales Prevents axis scales and gradient legend scales from automatically adjusting in response to data or filtering changes.

Error Bar Offset Adjust the offset for overlaid error bars.

Link Page Axes Links or unlinks graph axis scales across levels of the By group variable in the Page zone.

Fit to Window Specifies whether the graph is resized as you resize the JMP window. The default setting is Auto, which is usually set to On except when the window is shared with other content or if there is a Page variable. To always fit the graph inside the window, keep the setting at On. You can also maintain the aspect ratio. To prevent the graph from resizing, change the setting to Off.

Sampling Uses a random sample of the data to speed up graph drawing. If the sample size is zero, or greater than or equal to the number of rows in the data table, then sampling is turned off.

Graph Spacing Sets the amount of space between graph panels.

Include Missing Categories Enables a graph to collect and display missing values for categorical variables.

Set Alpha Level Enables you to set the Alpha level for graphs and statistics that use an alpha value for confidence intervals.

Launch Analysis Launches the Fit Model platform with the variables on the graph placed into roles. It launches the Distribution platform when only one variable is placed.

Make into Data Table Creates a new data table that contains the results from the graph.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Graph Builder Buttons

The Graph Builder window contains these buttons:

Recall Populates the Graph Builder window with the last graph that you created. This button changes into the Undo button once you perform an action.

Dialog Opens the Graph Builder launch window. This button changes into the Start Over button once you perform an action. You can add a By variable in the Graph Builder launch window.

Done Hides the buttons, Columns list, and all drop zone outlines. Equivalent to deselecting the Show Control Panel option.

Use this presentation-friendly format for pasting into other applications. To copy the graph, select **Edit > Copy**. To restore the window to the interactive mode, click the Graph Builder red triangle and select **Show Control Panel**.

Undo Reverses the last change made to the window.

Start Over Returns the window to the default condition, removing all data and graph elements from the window, and all variables from the drop zones.

Right-Click Menus

Numerous options are contained in Graph Builder right-click menus. The specific menu options depend on where in a graph you click, and the types and number of variables in your graph. Descriptions for the right click menu options are grouped in the following categories.

- "Graph Options"
- "Grouping Zones Options"
- "Options for Columns in the Variables Panel"
- "Options for Axes, Axes Variable Labels, or Graph Titles"
- "Options for the Dividing Line between Multiple Graphs"
- "Legend Options"

For multiple graphs (created with grouping zones or the Wrap zone), note the following:

- Generally, any options that you apply to one graph element apply to all graphs, across all grouping and wrap variables.
- To apply an option to only one graph, right-click in that graph area and select the option from the Graph menu.
- To apply an option to all of your multiple graphs simultaneously, first hold down the Ctrl key and right-click and then select the option from the Graph menu.

Graph Options

Right-clicking in a graph shows the following:

1. Element types that appear in the plot (such as Points, Line, and so on). Sub-menus provide options that are specific to each element type, and general options. Alternatively, you can change element-specific options in each element's properties panel, below the Variables panel.
2. The Add option, where you can select other elements to add to the graph. Alternatively, you can add more elements to the graph by dragging the desired element icons.
3. The Rows, Graph, Customize, and Edit menus. For descriptions of these options, see *Using JMP*.

Tip: Select **Rows > Name Selection in Column** to create a new column to name and label selected points in a graph.

Grouping Zones Options

Right-clicking in the X Group, Y Group, Wrap, Color, Size, and Overlay zones can show the following options:

Levels in View Changes the number of levels of the grouping variable that are visible. Use the arrow buttons to scroll forward and backward.

Number of Levels Changes the number of levels. See "Categorical and Continuous Variables in Group X or Y Zones".

Order By Orders the levels of a variable. See "Order the Levels of a Categorical Variable".

Show Title Shows or hides the variable title.

Title Orientation Changes the orientation of the variable text to horizontal or vertical.

Level Orientation Changes the orientation of the level values to horizontal or vertical.

Levels per Row Changes the number of columns included in the graph. Use with a **Wrap** variable.

Color Changes the background color of the grouping zone.

X or Y Group Edge Moves the grouping variable labels. See "Move Group X and Group Y Labels".

Summary Statistic Applies color and or size variables based on a summary statistic. Available only for a summarized Y response.

Swap Swaps the position of two variables. See "Move or Remove Variables in Zones".

Remove Removes a variable.

Options for Columns in the Variables Panel

Right-clicking on a column in the Variables panel shows options that depend on the column's modeling type. For more information about these menus, see *Using JMP*.

Options for Axes, Axes Variable Labels, or Graph Titles

Right-clicking on axes, axes variable labels, or graph titles show options for customizing your graph. Select **Show Properties** from a right-click menu to access the properties panel for additional customizations, such as tick line thickness or color. The following options are included in the right-click menus:

Graph Titles

Alignment Changes the graph title alignment.

Span Changes the width of the graph title span.

Additional options for customizing fonts, font colors, and the visibility of the title are also available in the right-click menu.

Axes

Axes have two levels of right-click menus, one for the axis title and one for the axis labels.

Combine Scales Combines scales when you have multiple variables in the same zone. Auto does the following:

- If there are 4 or more variables on one axis and no variables on the other axis, elements that support parallel mode are combined as parallel scales. If all variables are continuous and overlapping, Parallel Merged is used.
- If all variables are categorical, use Nested.

In addition, consider the following:

  - Merged creates one scale that covers all variables.
  - Nested creates a nested scale with one level per variable.
  - Parallel Merged is for parallel coordinates with a shared scale. You can use this option with the following elements: Points, Contour (Violin), Histogram, Box Plot and Parallel.
  - Parallel Independent is for parallel coordinates with independent scales. You can use this option with the following elements: Points, Contour (Violin), Histogram, Box Plot and Parallel.

Date/Time Bin (Appears only for a date or time axis variable.) Enables you to select a level of granularity to bin the data. For example, you might choose to bin at a high level, by year, or at a low level, by second.

Date/Time Wrap (Appears only for a date or time axis variable.) Enables you to select a cyclic period for your date data.

Swap Switches a variable from one zone to another. Select the variable that you want to switch places with.

Remove Removes the variable from the zone.

Move Right (Appears only if you have more than one variable in the Y zone.) Creates a second Y axis (on the right) for the variable that you select.

Move Left (Appears only after you have moved a variable to the right.) Moves the variable back to the left.

Align Ticks with Left Axis Aligns the major tick marks with those on the left axis.

Order By (Appears only if you have categorical variables in the X or Y zone.) Orders the levels of a categorical variable. See "Order the Levels of a Categorical Variable".

Note: Select **Other** to indicate a continuous variable as the ordering variable.

Order Statistic Enables you to select a statistic to use for the ordering.

Save Value Order (Available once an ordering has been defined.) Saves the defined order to the Value Order column property.

Note: You can edit value ordering from Graph Builder. Right-click on an axis and select **Edit Value Order**.

Size By (Available for a categorical axis.) Enables you to size your box plots by the count of observations within a group or by a numerical column.

Size Statistic (Available when the Size By option indicates a numerical column as the sizing variable.) Enables you to select a statistic to use for the sizing.

For descriptions of the general axis options below the line, see *Using JMP*.

Note: The **Revert Axis** option reverts the axis back to the default settings. The **Revert Scale** option reverts the scale-specific items: min, max, major interval, n minor.

Options for the Dividing Line between Multiple Graphs

This menu appears only if you have two or more graphs in the graph area. Right-click the line that separates graphs. Options appear that change the formatting of the line, such as color, transparency, spacing, and borders.

Tip: If the line between the graphs is too thin, you can resize it. Click the Graph Builder red triangle, select **Graph Spacing**, and enter a larger value.

Legend Options

A legend consists of elements, such as lines, fill colors, and gradients. Right-click on legend elements for customization properties. These properties include color and labeling options when available. Additional legend settings are found in the Legend Settings window.

To change legend settings or revert to initial settings, right-click to the right of the legend or on the title (if applicable). To undo recent changes to the legend, click Undo.

Tip: When you have both an overlay and a color variable, the color variable levels are shown by rounded color icons in the legend.

Legend Settings Change certain aspects of the graphics elements (line, bars, and so on) such as color, fill, transparency, and gradient. You can also add a title to the legend. See "Legend Settings Window".

Revert Legend Returns the legend to its initial settings.

Move the Legend

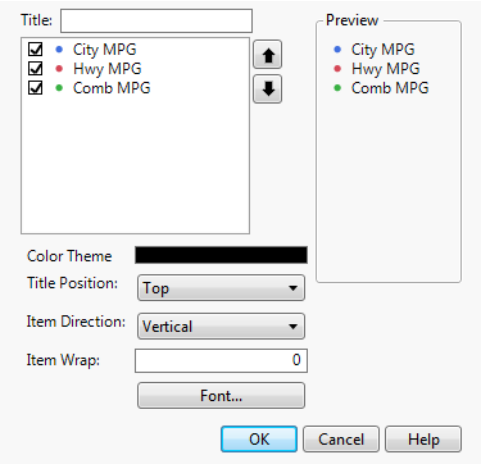
To move the position of the legend, click and drag to the desired location. The location can be a hotspot corresponding to the locations accessible from the Graph Builder red triangle menu or the legend or it can be any other location within the graph.

To use the menu options, click the Graph Builder red triangle and select **Legend Position**. To make the legend vertical or horizontal, change the Item Direction in the Legend Settings window.

Legend Settings Window

Change aspects of the legend through the Legend Settings window.

Figure 3.45 Example of the Legend Settings Window Using Hybrid Fuel Economy.jmp



Title The name of the legend.

Check boxes Shows or hides items in the legend. JMP hides items that appear to be redundant, but you can make them appear by selecting them. If your legend has a large number of entries, some might be hidden in the Graph Builder window to ensure sufficient space for the graph. However, all of the entries appear here, so you can select the ones that you want to show. To select or deselect all entries, right-click and select **Toggle check box**.

Up and down arrows Changes the order of items in the legend.

Color Theme Select a different color theme. For more information about color themes, see *Using JMP*.

Title Position Places the legend title on top or to the left of the items in the legend.

Item Direction Displays the legend horizontally or vertically.

Item Wrap Sets the legend to be *n* items tall (if vertical) or *n* items wide (if horizontal).

Font Changes the font type, style, and size.

Preview Shows your changes to the legend.

OK Commits your changes to the legend.

Cancel Cancels your changes to the legend.

Help Opens the online Help.

Discrete or Continuous Legend Items

Legends can list discrete values or show a scale of continuous values:

- Discrete legends can contain the levels of a categorical variable, or a list of variables.
- Continuous legends describe a color intensity scale for a range of numerical values.

Right-click an item in either legend to see the following options:

Line Color (Discrete legends only.) Changes the color of the element associated with the item.

Marker Changes the type of marker associated with the item.

Marker Size Changes the size of the marker associated with the item.

Pattern or Style Changes the fill or pattern of the element associated with the item.

Line Width Changes the width of the line associated with the item.

Fill Color Assigns a color to the item.

Color by Theme Colors multiple contours by levels in a color theme. This option sets line colors and fill colors for density contours.

Transparency Changes the marker or label transparency. Enter the level of transparency to draw markers (points) on the graph. The degrees of opacity ranges from 0 (clear) to 1 (opaque).

Gradient (Continuous legends only.) Changes the gradient settings. See "Gradient Settings".

Density Gradient (Contour element only.) Changes the gradient colors:

Fade to White The highest density contour uses the fill color, and colors fade to white at lower densities.

Fade to Gray The highest density contour uses the fill color, and colors fade to gray at lower densities.

Full Color Density contours use a predefined or customized color theme with full color control.

Gradient Transparency (Contour element only.) Changes the transparency of the gradient:

None Density contours use one transparency level for all levels.

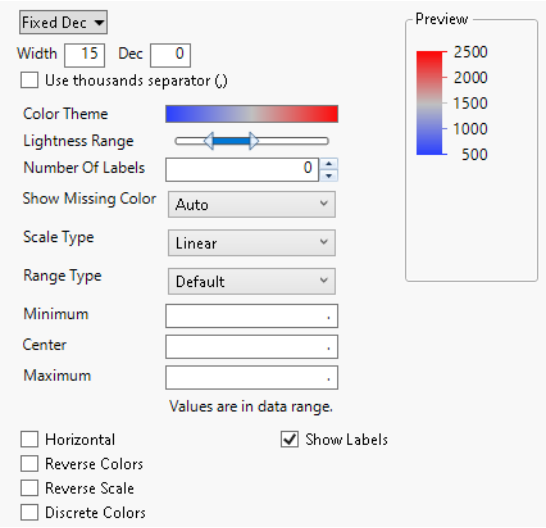
Linear Density contours get gradually more transparent at lower densities.

Revert Restores default settings.

Gradient Settings

You can change various aspects of the gradient legend in the Gradient Settings window.

Figure 3.46 Gradient Settings Window



To format the labels in the legend, use the menu, Width and Dec boxes, and the **Use thousands separator (,)** check box in the top left of the window.

Color Theme Change the color theme or define a custom color theme. For more information about color options, see *Using JMP*.

Lightness Range Set the range of intensities for the gradient coloring.

Number of Labels Specify the number of labels for your legend. The value of zero provides the default number of labels.

Show Missing Color Specifies whether to show missing color in the legend. Auto shows the color only when there are missing values, Off never shows the color, On always shows the color.

Scale Type Sets the scale for the gradient coloring. Look below the Maximum option for a description of the scale type.

Linear The scale is piecewise linear between the Minimum and Center values and between the Center and Maximum values. This is the default scaling.

Quantile The scale is piecewise linear for quantiles of the variable represented by the legend.

Standard Deviation The legend range is divided into offsets from the mean determined by standard deviation increments. The scale is piecewise linear between these offset settings. The number of offsets is determined by the Number of Labels.

Log The scale is linear for the logarithm of the values represented by the legend.

Log Offset The scale is linear for the offset of the base 10 logarithm of the values represented by the legend.

Range Type Sets the range of the legend values.

Default Unless you specify values for the Minimum and Maximum, the minimum and maximum values include the range of the data and are chosen to be appropriately rounded values.

Exact Data Range The minimum and maximum values are the exact minimum and maximum values for the data, or they are the values that you specify as Minimum and Maximum.

Middle 90% The minimum and maximum values are the 5^{th} and 95^{th} quantiles, or they are the values that you specify as Minimum and Maximum. This option is resistant to outliers.

Minimum, Center, Maximum Specify the smallest, middle, and largest values to use for your legend scaling.

Note: These values can override the Range Type values.

Horizontal Changes the orientation of the legend to horizontal.

Reverse Colors Reverses the colors in the color scheme.

Reverse Scale Reverses the scale for the color theme.

Discrete Colors Changes the color scheme from a continuous gradient to a stepped gradient with discrete colors.

Show Labels Shows or hides labels for the legend.

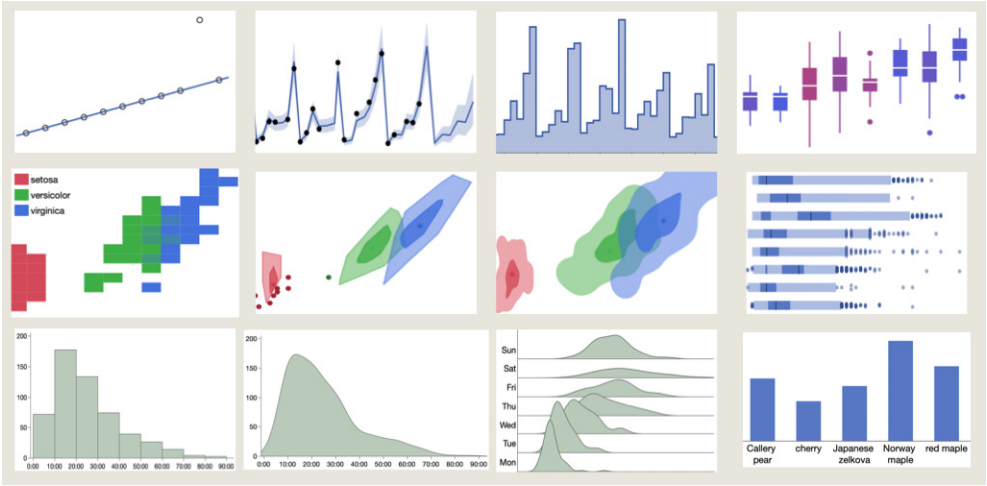
Chapter 4

Graph Builder Examples

Interactive Data Visualization

Graph Builder provides you with a variety of ways to visualize and explore your data. This chapter provides detailed examples of data exploration using bar charts, histograms, area plots, maps, and other types of plots.

Figure 4.1 Graphs Created in Graph Builder



Contents

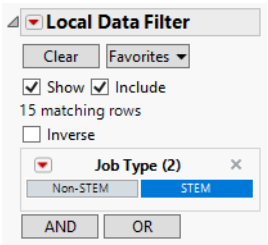
Example of an Ordered Bar Chart 123
Example of an Area and Line Chart 126
Example of a Map Chart 131
Example of a Butterfly Chart 134
Example of a Scatterplot 138
Example of a Packed Bar Chart 143
Example of an Overlaid Histogram and Ridgeline Chart 145
Example of a Bullet Bar Chart 147
Example of a Forecast Plot 149

Example of an Ordered Bar Chart

Data	This example uses data from college graduates in 2012. The table contains counts of degrees for graduates in 15 fields of study. Each degree is subdivided by the graduate's job type. There are categorical columns for degree, degree type, and job type (where type is STEM or non-STEM related). The continuous column Number contains the number of graduates from each unique combination of degree and job type. **Note:** These data come from the U.S. Census Bureau (2012).
Techniques	This example uses color, a local data filter, and sorting options.
Goal	The goal of this example is to visualize the percentage of graduates who are employed in STEM-related jobs, based on their degree type.

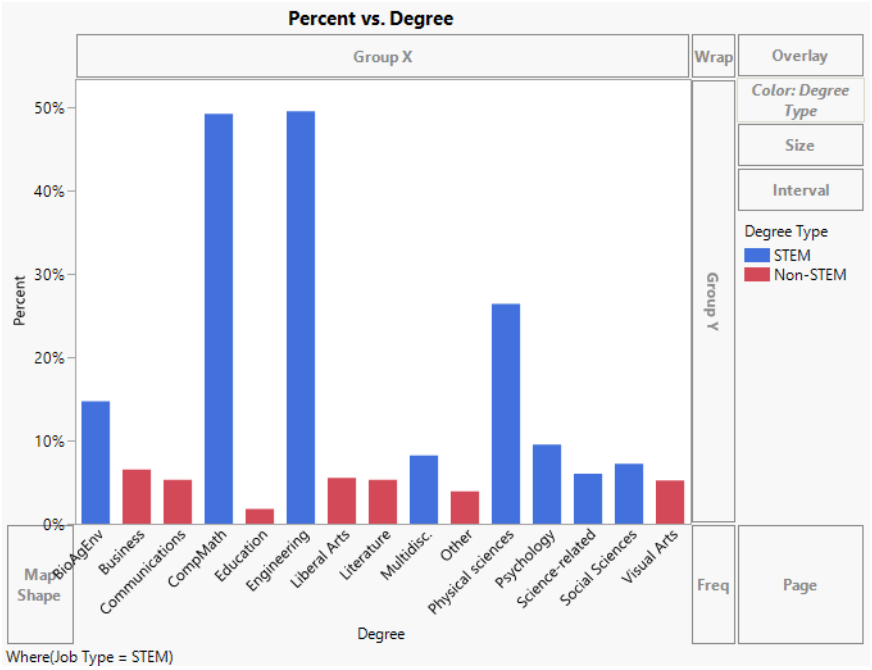
1. Select **Help** > **Sample Data Library** and open STEM Jobs.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select Degree and drag it to the **X** zone.
4. Select Percent and drag it to the **Y** zone.
5. Click the Bar element .
6. Click the Graph Builder red triangle and select **Local Data Filter**. In the Local Data Filter, perform these steps:
   a. Click Job Type and then click .
   b. Click **STEM**.

Figure 4.2 Selections in the Local Data Filter



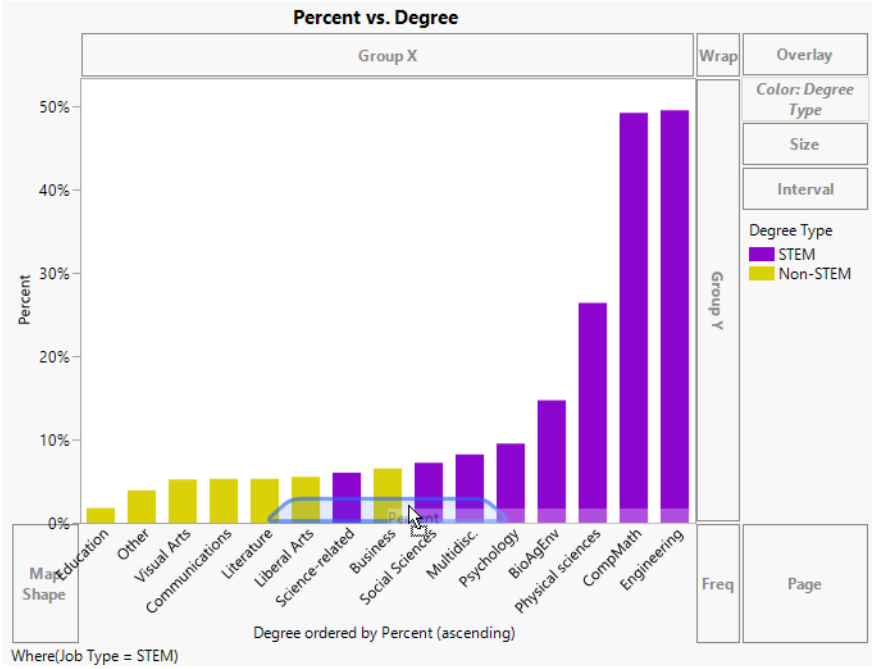
7. In the Graph Builder column list, select Degree Type and drag it to the **Color** zone.

Figure 4.3 Bar Chart Colored by Degree Type



8. In the legend, change the bar colors according to degree type:
   - Right-click **STEM** and change the fill color to purple.
   - Right-click **Non-STEM** and change the fill color to yellow.
9. In the Graph Builder column list, select Percent and drag it above the X axis. See Figure 4.4.

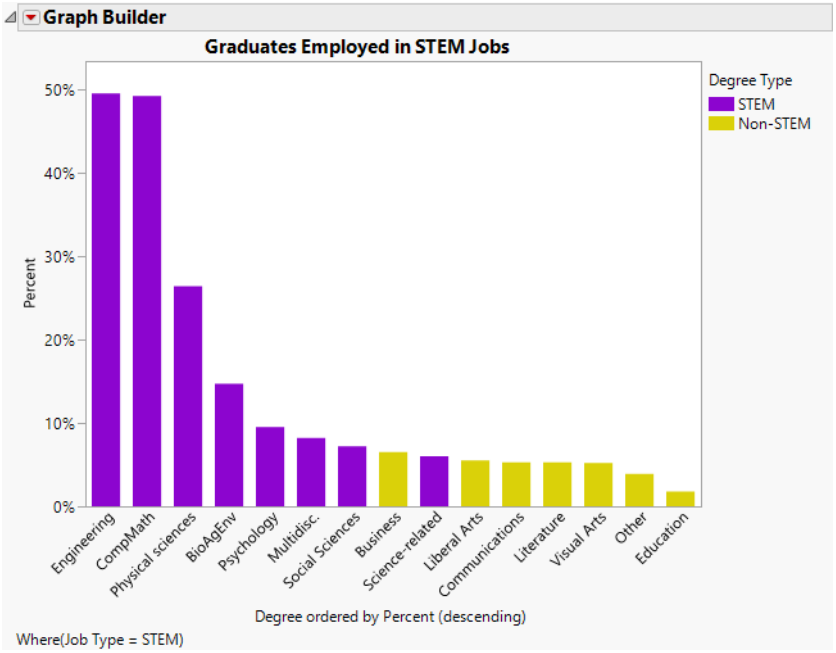
Figure 4.4 Ordering Degree by Percent



This orders the degree by percent in ascending order. However, you want it to be in descending order.

10. Right-click the X axis and select **Order By** > **Percent, descending**.
11. Click the graph title and type Graduates Employed in STEM Jobs and press Enter.
12. Click **Done**.

Figure 4.5 Completed Bar Chart



This graph displays the percent of graduates employed in STEM jobs by their degree type. Nearly 50% of Engineering and CompMath graduates are employed in STEM jobs, as compared to less than 2% of Education graduates.

Example of an Area and Line Chart

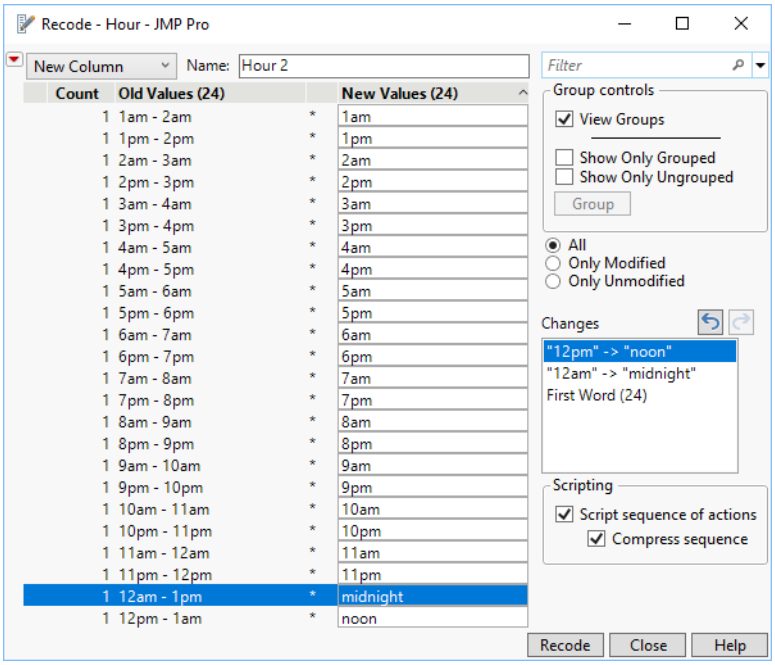
Data	This example uses data based on website traffic. It includes hourly measurements and historical range information.
Techniques	This example uses Recode to improve data labels, multiple graph elements (area and line), and legend customization. It also shows how to remove variables from a graphing element when multiple variables are plotted on a single axis.
Goal	The goal of this example is to display trend lines for website traffic within an overall range.

1. Select **Help** > **Sample Data Library** and open Bands.jmp.

2. Select the Hour column, and then select **Cols** > **Recode**.
3. In the Recode window, make these changes:
   a. Click the red triangle and select **First Word**.

      Notice that the New Values include only the first hour.
   b. Under New Values, click **12am** and enter **midnight**. Click **12pm** and enter **noon**.

Figure 4.6 Completed Recode Window



   c. Click **Recode**.

      A new column called Hour 2 is added to the data table.

Tip: Recode enables you to recode your data to a new column, as a formula in a new column, or to overwrite your current data. For more information about Recode, see *Using JMP*.

4. Select **Graph** > **Graph Builder**.
5. Select Hour 2 and drag it to the **X** zone.

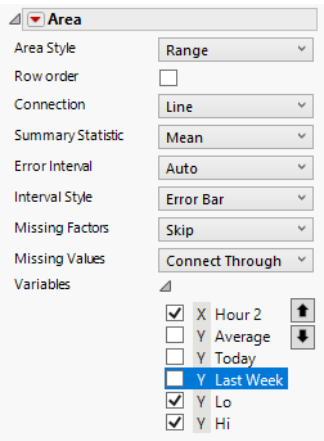
Note: The ordering of the X axis is logical, going from midnight to 11pm. This is because the Hour column has a value ordering column property that was copied to the Hour 2 column when you created it with Recode.

6. Select Average, Today, Last Week, Lo, and Hi and drag them to the **Y** zone.

7. Click the Area element .
8. In the Area properties panel, make these selections:
   a. Change the Area Style to **Range**.
   b. Open Variables and deselect Average, Today, and Last Week.

      Now the area element is applied only to the Lo and Hi variables.

Figure 4.7 Completed Area Properties Panel



9. Hold down the Shift key and click the Line element .

Tip: Holding down the Shift key and clicking adds additional elements to the graph, without replacing any existing elements.

10. In the Line properties panel, open Variables and deselect Lo and Hi.

    Now the line element is applied only to the Avg, Today, and Last Week Y variables.

Figure 4.8 Completed Line Properties Panel

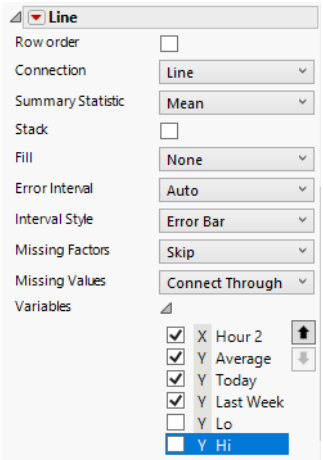
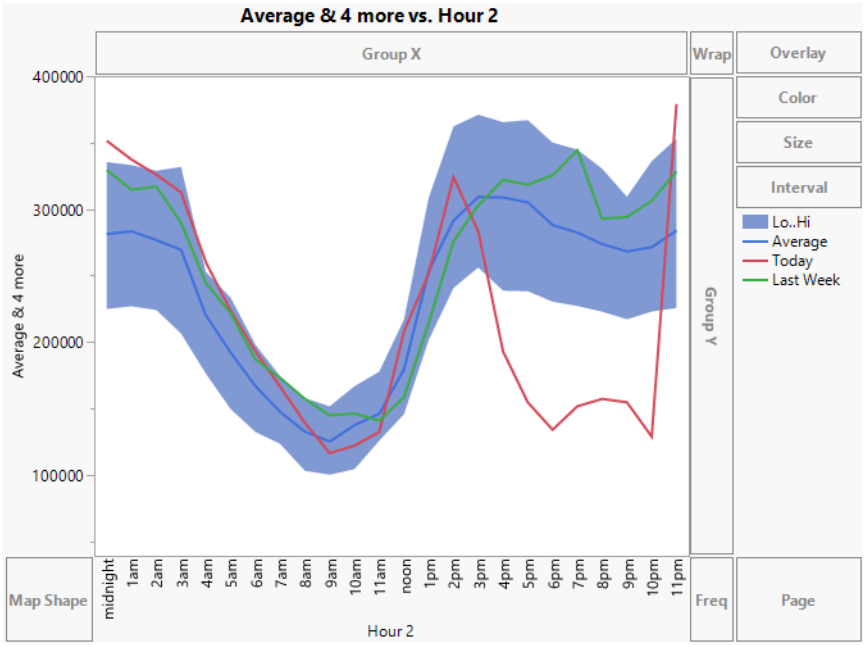


Figure 4.9 Area and Line Graphs

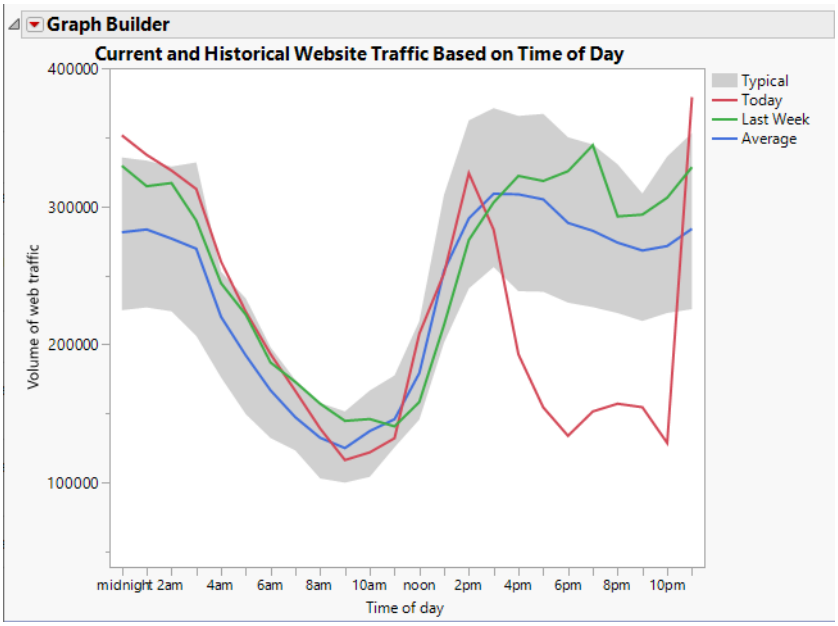


11. Customize the graph elements using the legend:
    a. In the legend at right, double-click **Lo..Hi**.
    b. In the Legend Settings window, double-click **Lo..Hi** and rename it to **Typical.**
    c. Highlight Today and Last Week, and click the up arrow to move them up one, so that the order is: Typical, Today, Last Week, and Average. Click **OK**.

d. Right-click Typical > **Fill Color** and select light gray.

12. Right-click the X axis > **Axis Settings**. For Label Orientation, click **Horizontal**, and then click **OK**.
13. Change the titles of the axes and graph:
    a. Click the Y-axis title and enter Volume of web traffic.
    b. Click the X-axis title and enter Time of day.
    c. Click the graph title and enter Current and Historical Website Traffic Based on Time of Day.
14. Click **Done**.

Figure 4.10 Completed Area and Line Graph



Tip: If an item in the legend is still highlighted, click it to deselect it.

The graph shows that the afternoon website traffic fell to unusually low levels today. The red trend line that represents today's traffic peaks at 1pm and then falls to outside of the gray band that represents the range of expected daily traffic for this website by about 3pm.

Example of a Map Chart

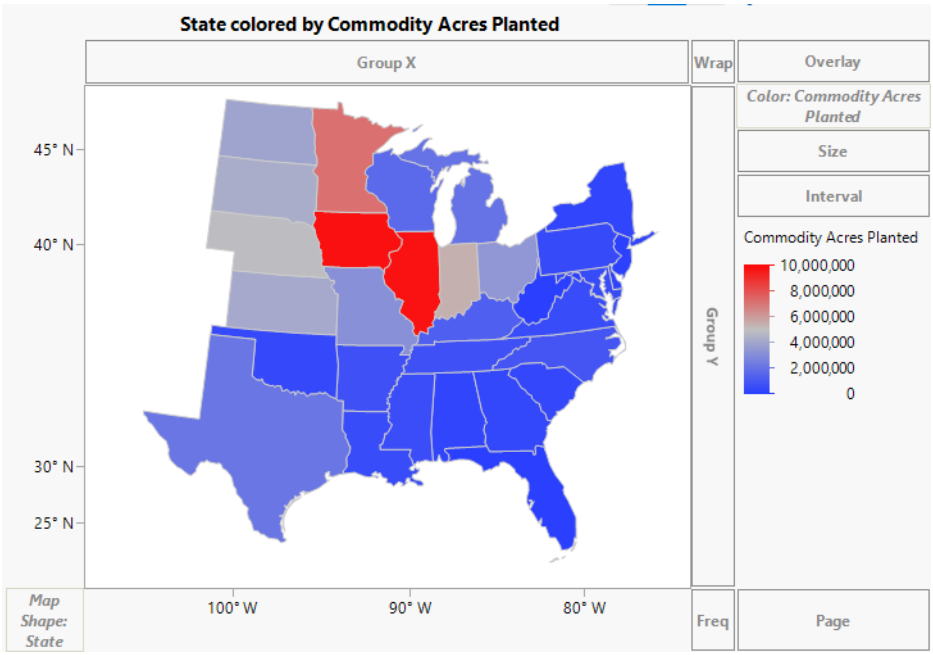
Data	This example uses data from 2017 of commodity acres planted in states within the US that grow corn, wheat, and soybeans. **Note:** These data come from the United States Department of Agriculture (2017).
Techniques	This example uses map shapes and legend customization.
Goal	The goal of this example is to show the median number of commodity acres that have been planted per state.

1. Select **Help** > **Sample Data Library** and open Corn Wheat Soybean Production.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select State and drag it to the **Map Shape** zone.

   Note: JMP recognizes US State names as map shapes. For more information, see "Map Shape".

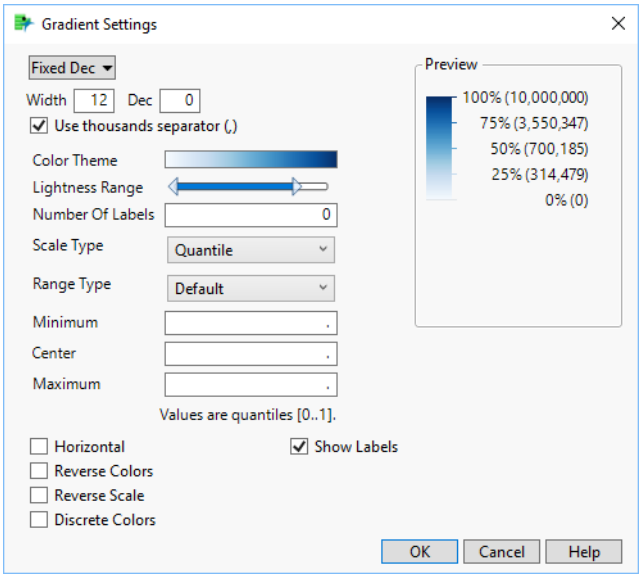
4. Select Commodity Acres Planted and drag it to the **Color** zone.
5. In the Map Shapes properties panel, change the Summary Statistic to **Median**.

Figure 4.11 Map Showing Median Commodity Acres Planted



6. Customize the graph elements in the legend:
   a. In the legend color bar, right-click and select **Gradient**.
   b. Next to Width, enter **12**.
   c. Next to Color Theme, click the color bar. Under Sequential, click the 2nd option (white to blue) and then click **OK**.
   d. From the Scale Type list, choose **Quantile**.

Figure 4.12 Completed Gradient Settings Window



e. Click **OK**.

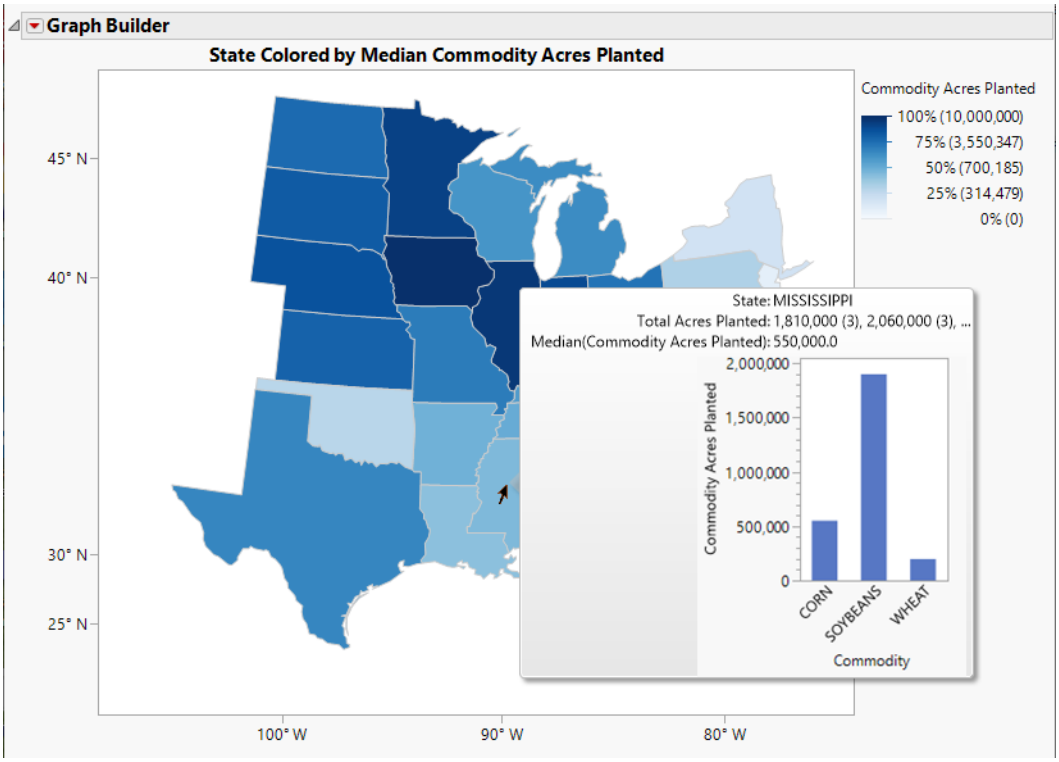
7. Click the title above the graph and type **State Colored by Median Commodity Acres Planted**.
8. Right-click in the graph and select **Hover Label > Bar**.

   This adds a bar chart hover graph. When you hover over a state, you can see the proportions of the commodities grown in that state in a bar chart.

   Note: For more information about Hover Labels, see *Using JMP*.

9. Click **Done**.
10. Hover over a state (in this case, Mississippi).

Figure 4.13 Completed Map Colored by Median Commodity Acres Planted



The coloring on the map highlights the states with the highest number of acres of wheat, corn, and soybeans. The dark blue states are known as America's Breadbasket. The hover graphs enable you to explore the proportions of the three crops grown in a state. This example shows that in Mississippi, soybeans are the prominent crop.

Tip: To see other examples using Graph Builder, run the additional scripts in the Corn Wheat Soybean Production.jmp data table.

Example of a Butterfly Chart

Data	This example uses summarized survey responses to 20 questions all using the same 5-point Likert scale.
Techniques	This example uses a bar chart with variable ordering and color customizations.

Goal	The goal of this example is to create a butterfly chart of the responses to 20 survey questions. This provides a summary visual of the results from a large number of survey questions in one chart.

1. Select **Help** > **Sample Data Library** and open Likert Survey.jmp.

   Notice the following about this data:

   – The data has a row for each question and columns with the counts of responders for each different response.

   – The order of the survey questions in the question column is not sequential. When you use this column in Graph Builder, JMP orders the rows sequentially.

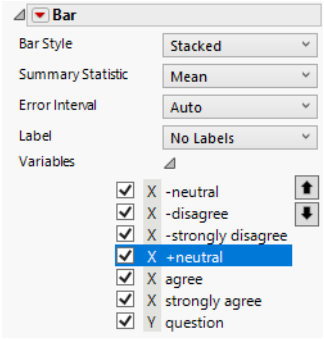
   – Formula columns are included in the table to create negative counts for negative responses. The neutral counts are split so that half contribute to a negative column and half contribute to a positive column.

2. Select **Graph** > **Graph Builder**.
3. Select question and drag it to the **Y** zone.
4. Select agree, strongly agree, -strongly disagree, -disagree, -neutral, and +neutral and drag them to the **X** zone.
5. Click the Bar element .
6. In the Bar properties panel, do the following:

   – Change the Bar Style to **Stacked**.

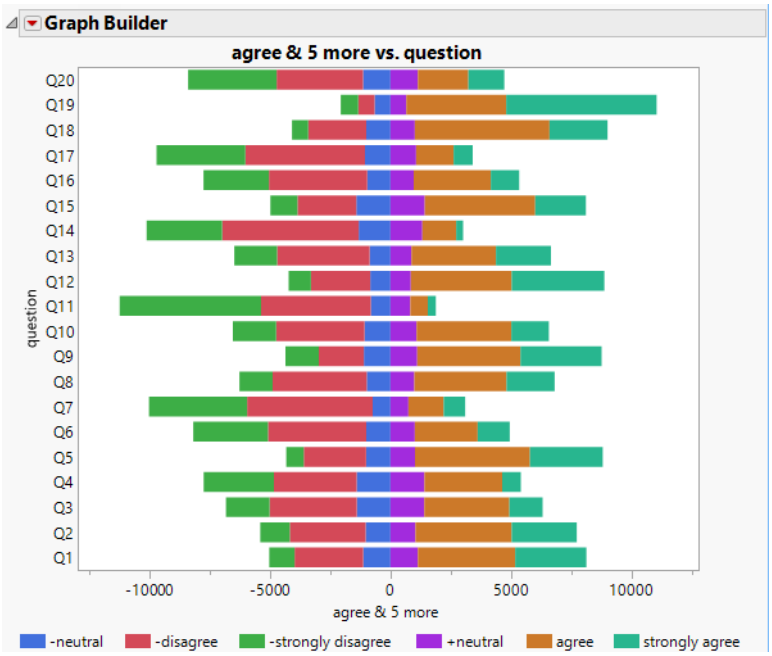
   – Open **Variables**, and then highlight a variable and use the arrow icons to move them into this order, from top to bottom: -neutral, -disagree, -strongly disagree, +neutral, agree, strongly agree.

Figure 4.14 Completed Bar Properties Panel



7. Click the Graph Builder red triangle and select **Legend Position** > **Bottom**.
8. Click **Done**.

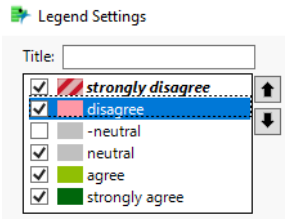
Figure 4.15 Graph Prior to Legend Customizations



Customize the Legend Colors and Ordering

1. Right-click on each color in the legend and change the Fill Color:
   - Light gray for +neutral and -neutral
   - Light pink for -disagree
   - Red for -strongly disagree
   - Light green for agree
   - Dark green for strongly agree
2. Hover over any item in the legend. The cursor should appear as a hand. Double-click to open the Legend Settings window.
3. Highlight a variable and use the arrow icons to change the order from top to bottom: -strongly disagree, -disagree, -neutral, +neutral, agree, strongly agree.
4. Double-click +neutral and rename it to **neutral**.
5. Deselect the box next to -neutral so that it no longer appears in the legend, because it is a duplicate.
6. Double-click -strongly disagree and -disagree and remove the - sign.

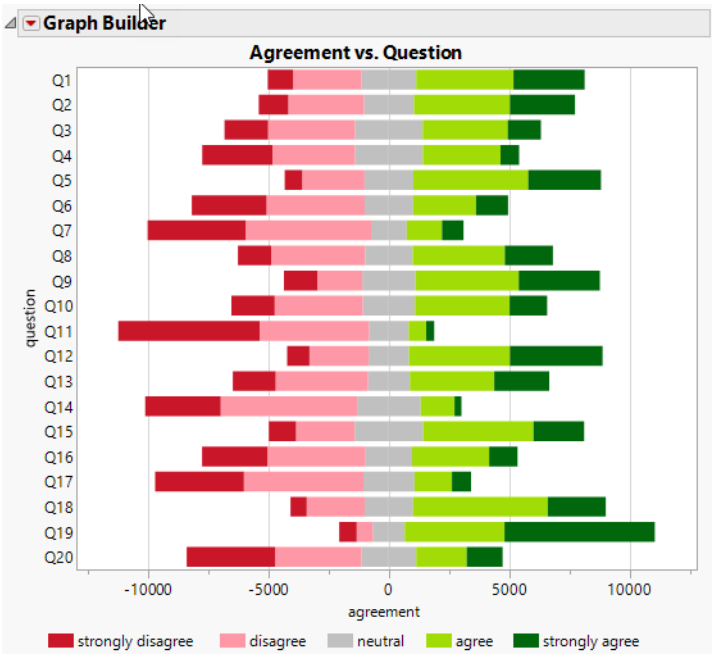
Figure 4.16 Completed Legend Settings



7. Click **OK**.
8. Change the titles of the X axis and graph:
   a. Click the graph title and enter Agreement vs. Question.
   b. Click the X-axis title and enter agreement.
9. Double-click in the Y axis and select the **Reverse Order** box, and then click **OK**.
10. Double-click in the X axis and select the **Major Grid Lines** box, and then click **OK**.

    The grid lines help you visually judge the differences between question results.

Figure 4.17 Final Likert Scale Graph



Tip: If an item in the legend is still highlighted, click it to deselect it.

This graph provides a visual summary of 20 survey questions, ignoring the non-responses. Note that the non-responses are less than 1% of the total responders for any question. Therefore, leaving them out of the visualization does not impact the conclusions. Question 19 has the most positive responses, and Question 11 has the least favorable responses.

Example of a Scatterplot

Data	This example uses data from 442 diabetic patients. The data includes baseline clinical and laboratory data as well as a binary measure of diabetes disease progression obtained one year after each patient's initial visit. This measure classifies disease progression as Low or High.
Techniques	This example uses two scatterplots with a shared X axis, axis customization, and annotation.
Goal	The goal of this example is to compare and understand the predicted probabilities of High disease progression from two classification models.

Run the Classification Models

You want to construct a classification model to predict the disease progression based on clinical and laboratory variables. You will build two different classification models, save prediction formulas, and compare their predicted classifications.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Click the green triangle next to the **Decision Tree of Y Binary** script to build a decision tree prediction model.
3. Click the Partition for Y Binary red triangle and select **Save Columns > Save Prediction Formula**. You can close this window.

   This saves the probability formulas to the data table.
4. In the data table, right-click the Prob(Y Binary= = High) column and select **Column Info**.
5. Change the column name to Partition Prob High and click **OK**.
6. Click the green triangle next to the **Neural of Y Binary** script to build a neural net prediction model.
7. Click the Model NTanH(3) red triangle and select **Save Profile Formulas**. You can close this window.

   This saves the probability formulas to the data table.
8. In the data table, right-click the Probability(Y Binary=High) column and select **Column Info**.

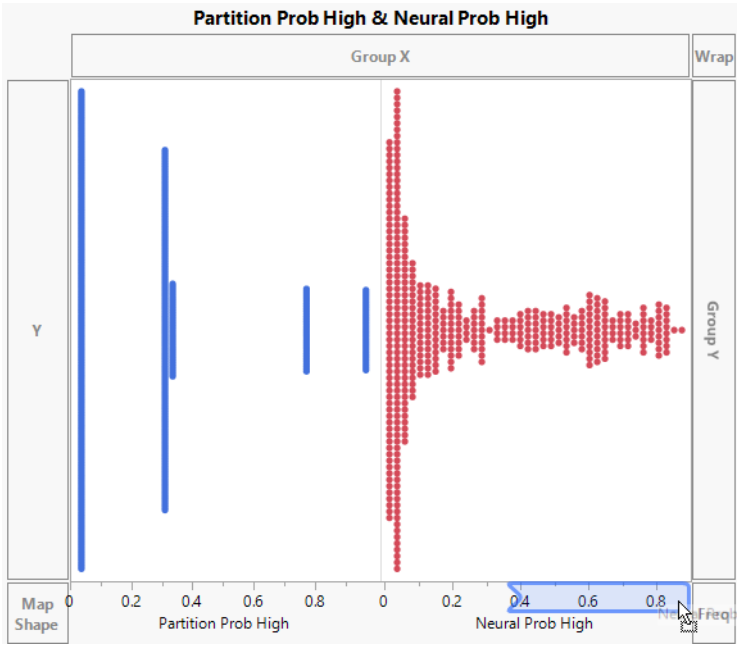
9. Change the column name to Neural Prob High and click **OK**.

Create the Initial Graph

You will compare the model predicted probabilities of a patient having a High disease progression.

1. Select **Graph > Graph Builder**.
2. Select Partition Prob High and drag it to the **X** zone.
3. Select Neural Prob High and drag it to the **X** zone, to the right of Partition Prob High. This creates a second X axis.

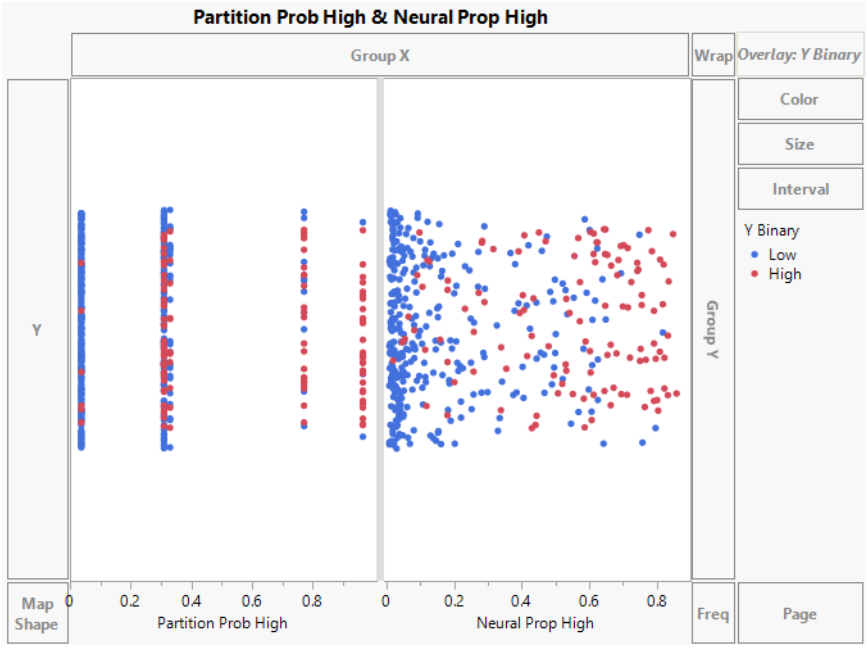
Figure 4.18 Drag Neural Prob High to the Right of Partition Prob High



4. Select Y Binary and drag it to the Overlay zone.
5. Click the Graph Builder red triangle and select **Graph Spacing**.
6. Type 6 next to Graph Spacing and click **OK**.

   This increases the spacing between the two X axes.

Figure 4.19 Initial Graph of Model Probabilities

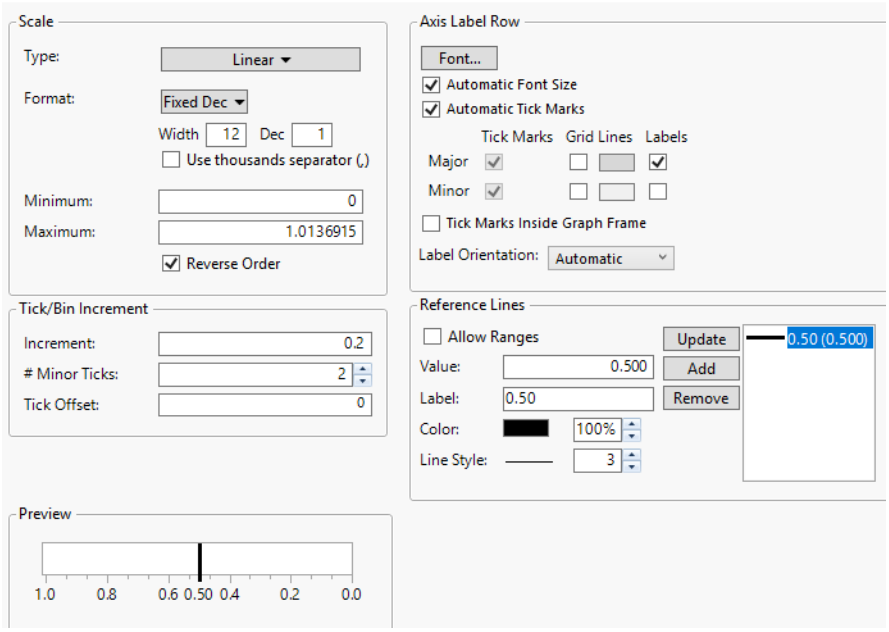


Customize the Graph

In both modeling platforms, the default threshold value is 0.50. This means that if a patient has a predicted probability greater than 0.50 of "High", the model predicts their classification as "High". Use reference lines to show the threshold value in the graph.

1. Right-click the Partition Prob High X axis and select **Axis Settings**.
2. In the Scale section, click the box next to **Reverse Order**.
3. In the Tick/Bin Increment section, set **# Minor Ticks** to 2.
4. In the Reference Lines section, type 0.50 in the boxes next to **Value** and **Label**.
5. In the box next to Line Style, enter 3. The 3 indicates the thickness of the reference line.
6. Click **Add**.

Figure 4.20 X Axis Settings

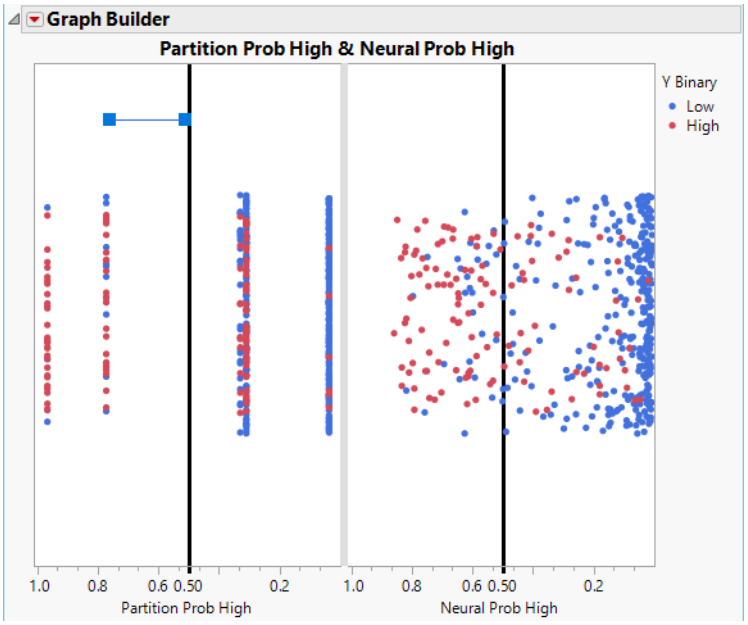


7. Click **OK**.
8. Right-click on the Partition Prob High X axis and select **Edit** > **Copy Axis Settings**. Right-click on the Neural Prob High X axis and select **Edit** > **Paste Axis Settings**.
9. Click **Done**.

Annotate the Graph

1. In the main menu, click **Tools** > **Line**. Draw a horizontal line to the left of the reference line in the Partition Prob High graph.

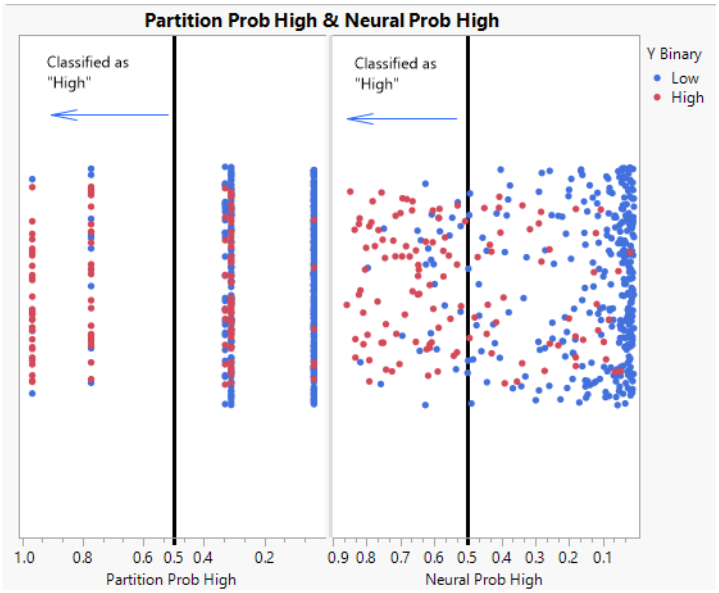
Figure 4.21 Line Drawn to the Left of the Partition Prob High Reference Line



2. Right-click the line and select **Point to**.

   The arrow should be pointing away from the reference line.
3. In the main menu, click **Tools** > **Annotate** and click on the graph above the arrow.
4. Type **Classified as "High"** in the text box. See Figure 4.22.
5. Repeat step 1 through step 4 in the Neural Prob High graph.

Figure 4.22 Distribution of Predicted Probabilities



The graph shows that the distribution of the predicted probabilities differs between the two models. The partition model has five predicted score levels while the neural model scores are dispersed across the score range. For both models, there is a cluster of blue data points on the far right of each plot. These are low subjects that both models classify correctly as they fall to the right of the threshold.

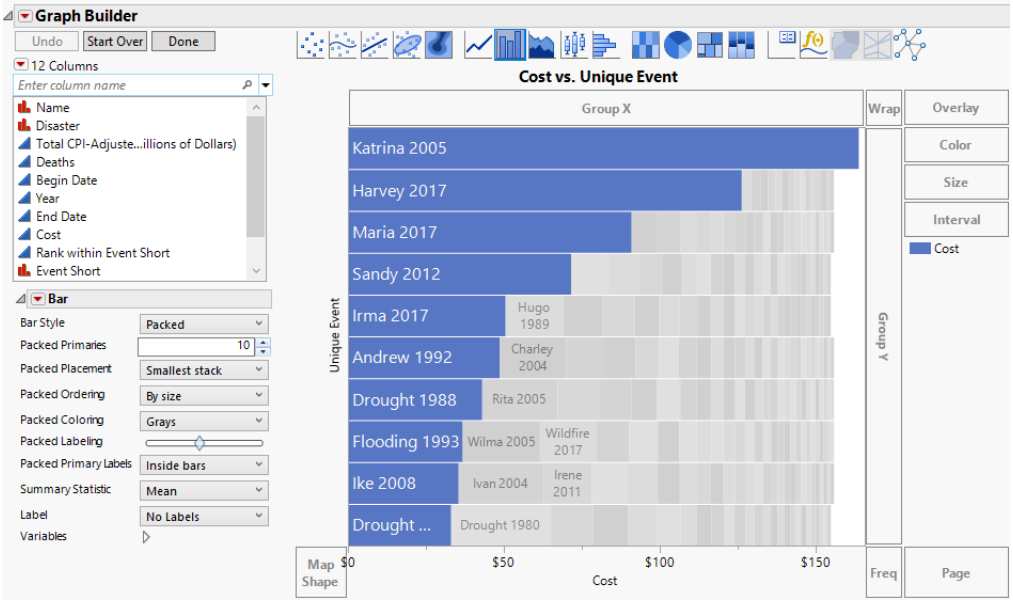
Example of a Packed Bar Chart

Data	This example uses data from the National Oceanic and Atmospheric Administration on the economic impact of weather and climate events in the U.S. during the years 1980 to 2018.
Techniques	This example uses a packed bar chart with 10 primary categories.
Goal	The goal of this example is to create a packed bar chart showing the events that had the greatest economic impact.

1. Select **Help > Sample Data Library** and open Billion Dollar Events.jmp.
2. Select **Graph > Graph Builder**.
3. Select Unique Event and drag it to the **Y** zone.

4. Select Cost and drag it to the **X** zone.
5. Select the Bar element .
6. In the Bar options panel:
   a. For the Bar Style, select **Packed**.
   b. Change Packed Primaries to 10.
   c. Move the Packed Labeling slider to the left until it is about halfway.

Figure 4.23 Weather Events With Large Economic Impacts in a Packed Bar Chart



The top 10 categories appear as a bar chart with blue bars. These are the “Packed Primaries” that you set to 10 in the options. Secondary categories are labeled and in gray. You can hover over any bar for more details.

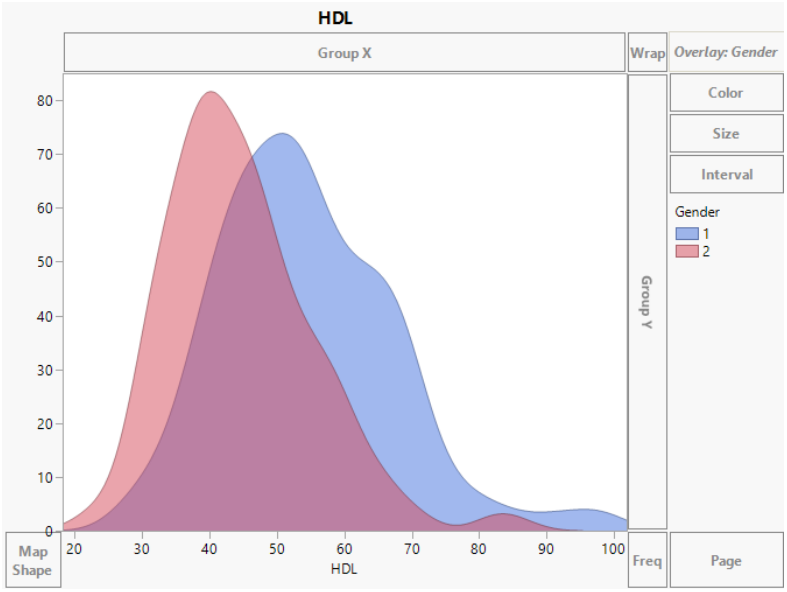
In this graph, you can clearly see that Katrina in 2005 had the largest financial impact of any storm that hit the US between 1980 and 2018. The 2017 hurricanes, Harvey and Maria, also had large financial impacts.

Example of an Overlaid Histogram and Ridgeline Chart

Data	This example uses data from 442 diabetic patients. The data include baseline clinical and laboratory data as well as a binary measure of diabetes disease progression obtained one year after each patient's initial visit. This measure classifies disease progression as Low or High.
Techniques	This example uses overlaid histograms and a ridgeline chart.
Goal	The goal of this example is to explore the distribution of a continuous variable grouped by categories. Specifically, you are interested in exploring how the distribution of HDL varies by gender and then by disease progression.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Graph > Graph Builder**.
3. Select HDL and drag it to the **X** zone.
4. Select Gender and drag it to the **Overlay** zone.
5. Click the **Histogram** element icon .
6. In the Histogram properties panel, from the Histogram Style list, select **Kernel Density**.
7. Hover over the Y axis near the top of the scale until the cursor turns into a horizontal hand. Click and drag until the maximum value is about 85.

Figure 4.24 Overlaid HDL Histograms by Gender

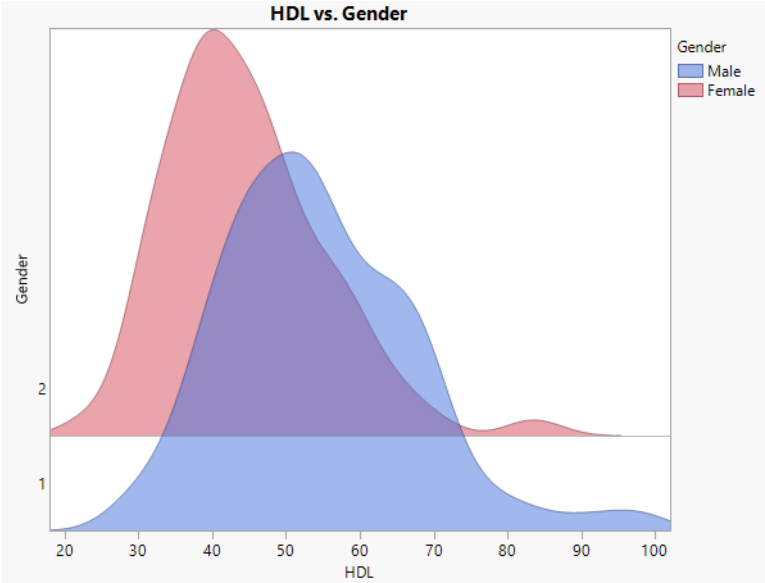


The histograms indicate that HDL levels are higher in males (blue) than females (pink). The blue distribution falls to the right, or higher on the HDL scale, than the pink distribution. Now change the graph to view the differences in gender in a ridgeline plot.

8. Select Gender and drag it to the Y zone.
9. In the Histogram properties panel, move the Overlap slider to overlap the histograms (until the slider is about 3/4 up).
10. Click **Done**.
11. Double-click in the legend. In the Legend Settings window, double-click 1 and type male. Double-click 2 and type female.
12. Click **OK**.

Tip: If an item in the legend is still highlighted, click it to deselect it.

Figure 4.25 Ridgeline Chart of HDL by Gender and Diabetes Level



The ridgeline chart shows the gender distributions offsets.

Example of a Bullet Bar Chart

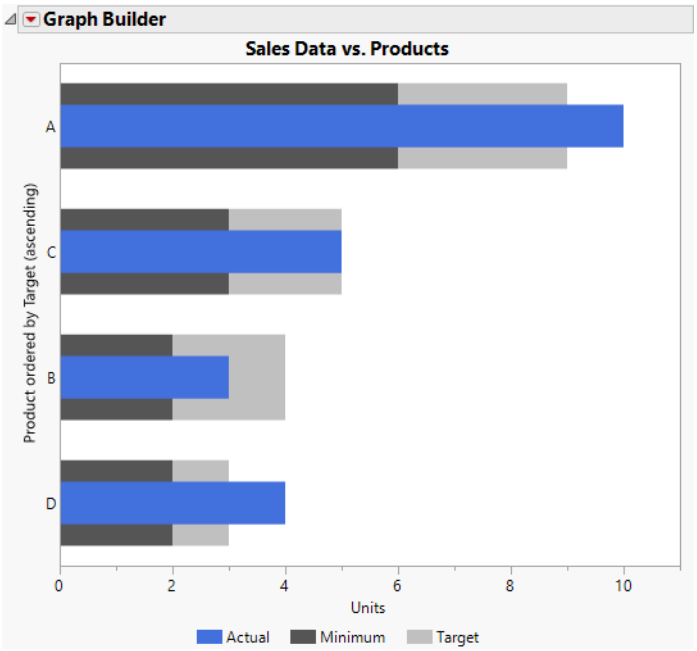
Data	This example uses fabricated sales data for four products.
Techniques	This example uses a bullet chart.
Goal	The goal of this example is to display actual units sold compared to a minimum and target number of units.

1. Select **Help** > **Sample Data Library** and open Bullet Measures.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select Actual, Target, and Minimum and drag them to the **X** zone.
4. Select Product and drag it to the **Y** zone.
5. Click the **Bar** element icon .
6. In the Bar properties panel, perform these actions:
   - From the Bar Style list, select **Bullet**.

- Open **Variables**, and then highlight Minimum and click the Up arrow once.

7. Click the Graph Builder red triangle and select **Legend Position** > **Bottom**.
8. Right-click on a color in the legend and change the Fill Color:
   - No change to Actual
   - Change Minimum to dark gray
   - Change Target to light gray
9. Right-click in the Y axis > **Order By** > **Target, ascending**.
10. Change the titles of the X axis and graph:
    a. Click the graph title and enter Sales Data vs. Products.
    b. Click the X-axis title and enter Units.
11. Click **Done**.

Figure 4.26 Bullet Chart of Actual, Target, and Minimum Units Sold



The products are ordered by the target number of units as shown by the light gray bars. Products A and D have actual sales that are above their targets; the blue bars extend beyond the light gray target bars. Sales for Product C are on target. Sales for Product B are below target but above the minimum number of units.

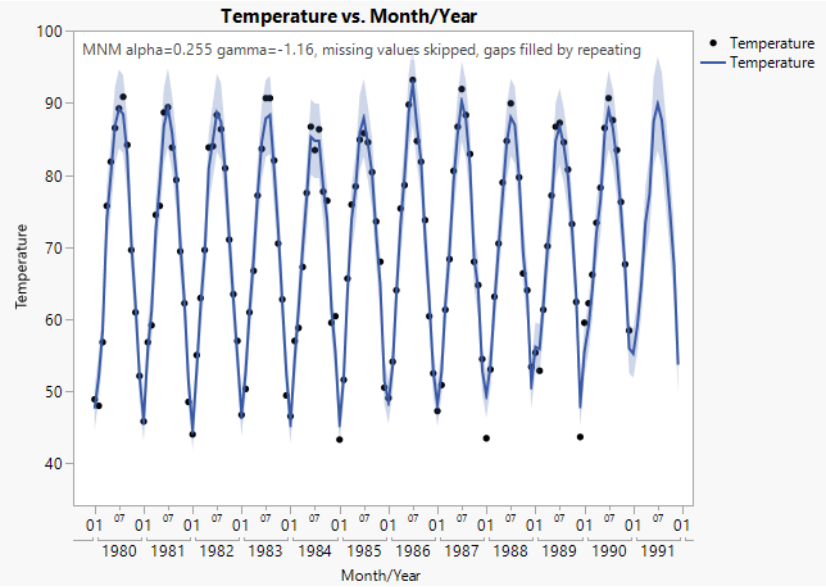
Example of a Forecast Plot

Data	This example uses maximum monthly temperature data from the US city of Raleigh, NC that were recorded from 1980 to 1990.
Techniques	This example uses a time series line of fit.
Goal	The goal of this example is to forecast the maximum monthly temperature for the following year.

1. Select **Help** > **Sample Data Library** and open Time Series/RaleighTemps.jmp.
2. Select **Graph** > **Graph Builder**.
3. Select Month/Year and drag it to the **X** zone.
4. Select Temperature and drag it to the **Y** zone.
5. Click the **Line of Fit** element icon .
6. In the Line of Fit properties panel, do these actions:
   - From the Fit list, select **Time Series**.
   - In the box next to Seasonal Period, type 12.
   - In the box next to Forecast Periods, type 12.

     This forecasts 12 months out. Since the data ends at 1990, it forecasts through 1991.
7. Right-click the X axis and select **Axis Settings**.
   - Under Tick/Bin Increment, change the Increment to 1.
   - Under Tick/Bin Increment, change the Label Row Nesting to 2.
   - Under Axis Label Rows, on the Label Row 1 tab, select **Labels** for the minor tick marks.
   - Click **OK**.
8. (Optional) To save the predicted forecast values and the lower and upper points of the forecast interval as new columns in the data table, click the Line of Fit red triangle and select **Save Formula**.
9. Click **Done**.

Figure 4.27 Forecast Plot of Maximum Monthly Temperature



The graph shows the data points used to estimate the time series function that extends through the forecast period. Note the following about this graph:

- The solid blue line represents the forecast model. The shaded blue area is the 95% confidence interval around the predicted values.
- The forecast period covers 1991, which is 12 months past the last data point. The forecast period is the right most portion of the plot that has no data points.
- The peak forecast high for 1991 is in July with an estimated value of 90°F. This estimate has a confidence interval of about 83°F to 96°F represented by the limits of the blue region above and below the estimate.
- The text at the top indicates the type of model that was fit, the parameter values for that model, and information about missing values. To remove this text from your graph, deselect the Forecast Model option in the Line of Fit properties panel.

Chapter 5

Bubble Plots

View Patterns in Multidimensional Data Using Bubble Plots

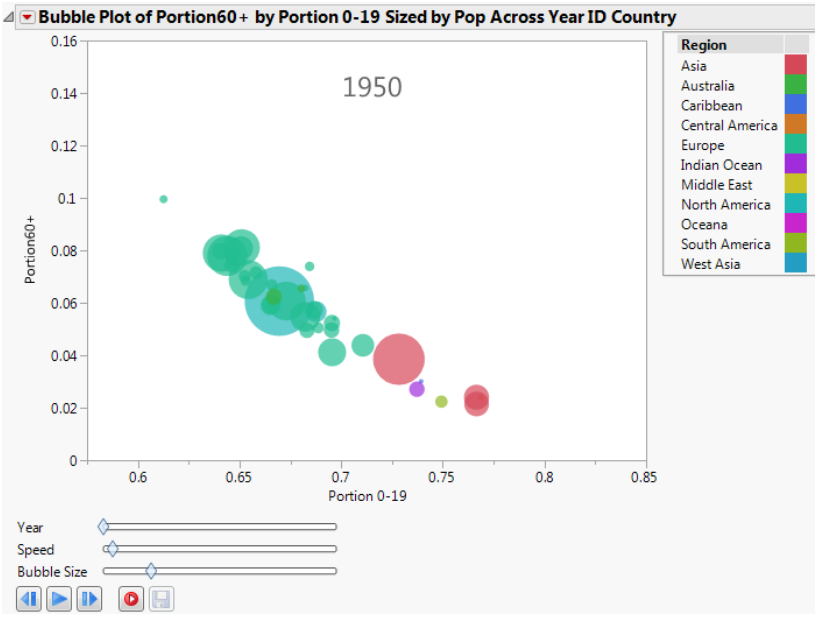
A bubble plot is a scatterplot that represents its points as circles, or bubbles. Bubble plots can be dynamic (animated over time) or static (fixed bubbles that do not move). Use bubble plots to:

- dynamically animate bubbles using a time variable, to see patterns and movement across time
- use size and color to clearly distinguish between different variables
- aggregate data (rows) into a single bubble, simplifying the bubble plot

Because you can see up to five dimensions at once (x position, y position, size, color, and time), bubble plots can produce dramatic visualizations and readily show patterns and trends.

Note: Dynamic bubble plots were pioneered by Hans Rosling, Professor of International Health, Karolinska Institutet, and the people involved in the Gapminder.org project.

Figure 5.1 Example of a Bubble Plot



Contents

Example of a Dynamic Bubble Plot 153
Launch the Bubble Plot Platform 155
Specifying Two ID Variables 156
Specifying a Time Variable 156
Interact with the Bubble Plot 157
Control Animation for Dynamic Bubble Plots 157
Select Bubbles 158
Use the Brush Tool 159
Bubble Plot Platform Options 159
Show Roles 162
Additional Examples of the Bubble Plot Platform 163
Example of Specifying Only a Time Variable 163
Example of Specifying Only ID Variables and Splitting a Bubble 165
Example of a Static Bubble Plot 167
Example of a Bubble Plot with a Categorical Y Variable 171

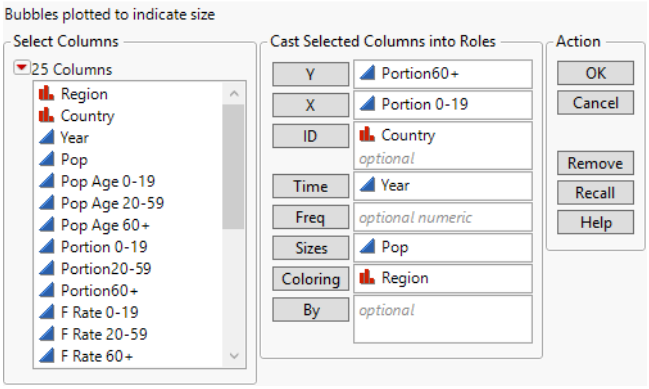
Example of a Dynamic Bubble Plot

This example uses the PopAgeGroup.jmp sample data table, which contains population data for countries and regions around the world. Examine the relationship between the proportion of younger and older people in the sample populations.

1. Select **Help > Sample Data Library** and open PopAgeGroup.jmp.
2. Select **Graph > Bubble Plot**.

   The launch window appears.

Figure 5.2 The Bubble Plot Launch Window



3. Select Portion60+ and click **Y**.

   The portion of the population that are 60 years or older becomes the *y* coordinate.
4. Select Portion 0-19 and click **X**.

   The portion of the population that are 0-19 years becomes the *x* coordinate.
5. Select Country and click **ID**.

   All the rows for each country are aggregated into a single bubble.
6. Select Year and click **Time**.

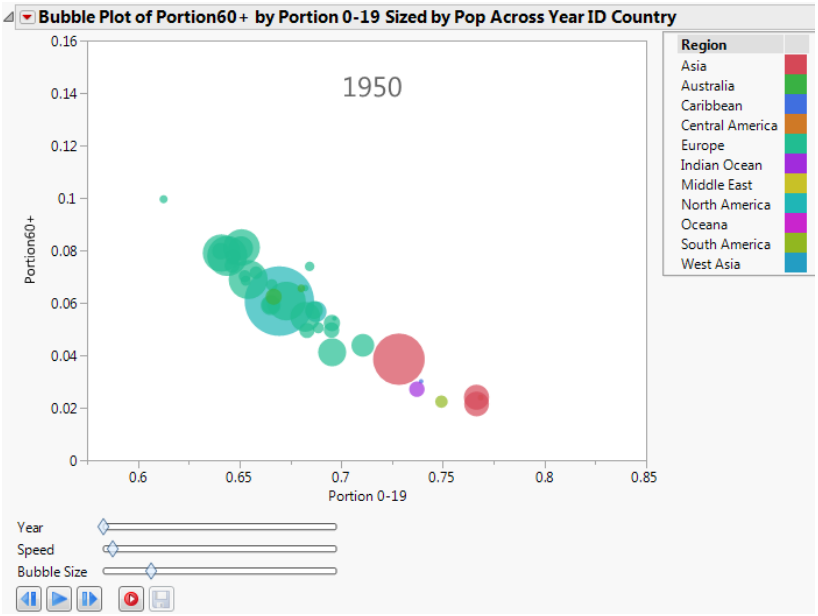
   The bubble plot shows a unique plot for each year's data.
7. Select Pop and click **Sizes**.

   The sizes of the bubbles reflect the overall population values.
8. Select Region and click **Coloring**.

   Bubbles for different regions are assigned different colors. See the Region legend in Figure 5.3. The colors shown in the plot are JMP default colors.
9. Click **OK**.

The report window appears.

Figure 5.3 The Bubble Plot Report Window



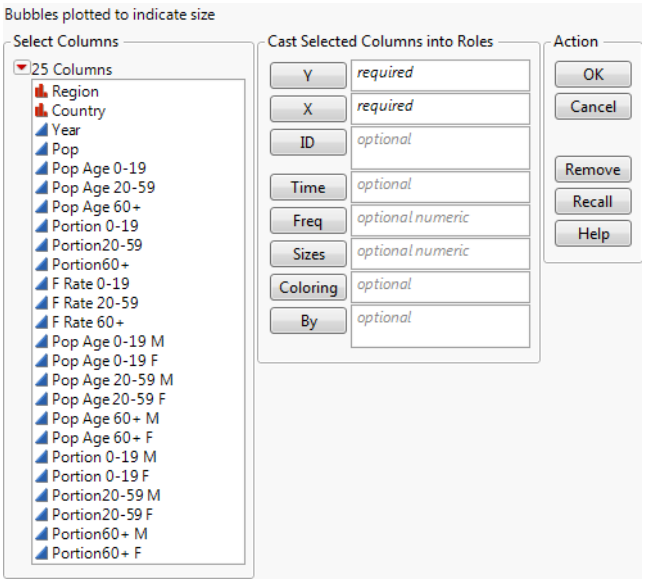
10. Click the play button to see the animated, dynamic report. Alternatively, you can click the previous button to move forward by one year.
11. (Optional) To view a legend that identifies each color with its region, click the Bubble Plot red triangle and select **Legend**.

As time progresses, you can see that the portion of the population that is 0-19 years decreases, and the portion of the population that is 60 years or more increases.

Launch the Bubble Plot Platform

Launch the Bubble Plot platform by selecting **Graph > Bubble Plot**.

Figure 5.4 The Bubble Plot Launch Window



Y, X The Y and X columns become the *y* and *x* coordinates of the bubbles in the plot. These values can be continuous or categorical (nominal or ordinal).

ID ID variables identify rows that should be aggregated and shown as a single bubble. The default coordinates of each bubble are the averaged *x* and *y* values, and the default size of each bubble is the sum of the sizes of all aggregated members. See "Specifying Two ID Variables".

Time Maintains separate coordinates, sizes, and colors for each unique time period. The bubble plot shows these values for a single time period. For example, if the **Time** column contains years, the bubble plot is updated to show data by each year. See "Specifying a Time Variable".

Freq Weights computations when aggregating bubbles using an ID variable.

Note: Negative frequency values are ignored.

Sizes Controls the size of the bubbles. The area of the bubbles is proportional to the **Size** value. There is a minimum bubble size, to keep bubbles visible, even if the size value is

zero. If **Size** is left blank, the default bubble size is proportional to the number of rows in that combination of **Time** and **ID**.

Coloring Colors the bubbles according to the selected variable. If the selected variable is categorical (nominal or ordinal), each category is assigned a unique color. If the selected variable is continuous, a gradient of colors is used. You can set a preference for a color theme by selecting File > Preferences > Graphs and making a selection in the Color Themes panel.

By Place a column here to produce a separate bubble plot for each level of the variable.

After you click **OK**, the Bubble Plot report window appears.

Specifying Two ID Variables

Specifying a second **ID** variable provides a hierarchy of categories, but the bubbles are not split by the second category until they are selected and split interactively. In the launch window, if you specify a second **ID** variable, **Split** and **Combine** buttons appear in the report window.

For example, you might specify a country as the first **ID** variable, resulting in a separate aggregated bubble for each country. A second **ID** variable, perhaps designating regions within each country, would further split each country when the interactive **Split** button under the graph is pressed.

Specifying a Time Variable

Maintains separate coordinates, sizes, and colors for each unique time period. The bubble plot shows these values for a single time period. For example, if the **Time** column contains years, the bubble plot is updated to show data by each year.

To move the time label on the plot, click and drag the label.

If data is missing within a time period, the value is linearly interpolated. If data is missing for the first or last time period, the value is not estimated, but left as missing.

Related Information

- "Control Animation for Dynamic Bubble Plots"
- "Example of Specifying Only a Time Variable"

Interact with the Bubble Plot

Note: If *all* of the rows used in constructing a bubble plot shape are hidden and not excluded, the corresponding bubble plot shape is not shown in the bubble plot. If rows are excluded in the data table, the bubble plot is constructed without the excluded rows.

Use the Bubble Plot platform in one of two modes:

- *Static mode*, where the bubbles are fixed and do not animate over time (no **Time** variable is specified). See "Example of a Static Bubble Plot".
- *Dynamic mode*, where the bubbles are animated over time (a **Time** variable is specified). See "Example of a Dynamic Bubble Plot".

You interact with both static and dynamic bubble plots in different ways.

Control Animation for Dynamic Bubble Plots

Use sliders and buttons to control the animation of dynamic bubble plots.

Figure 5.5 Animation Controls

Table 5.1 Descriptions of the Animation Controls

Slider or Button	Description
<Time variable>	Controls which time values appear in the bubble plot. You manually drag the slider to see a progression of time. Click and drag on the time variable in the bubble plot to move its position. Appears only if you specified a variable for **Time**.
Speed	Adjusts the speed of the animation. Appears only if you specified a variable for **Time**.
Bubble Size	Adjusts the size of the bubbles. The bubbles maintain their relative size, but their absolute size can be adjusted. Appears on all bubble plots.

Table 5.1 Descriptions of the Animation Controls *(Continued)*

Slider or Button	Description
	Adjusts the time value by one unit and shows the previous time value. Appears only if you specified a variable for **Time**.
 	Press Play to animate the bubble plot. Moves through all of the time values in order, and loops back to the beginning when the last time period is reached. Press Pause to stop the animation. Appears only if you specified a variable for **Time**.
	Adjusts the time value by one unit and shows the next time value. Appears only if you specified a variable for **Time**.
	Records the animation (Windows only).
	Saves the animation as an animated GIF file (Windows only).
Split	Separates the bubble represented by the first, larger **ID** variable into its smaller constituent parts, which are defined by the second, smaller **ID** column. Select the bubble and click **Split**. Appears only if you specified two **ID** variables.
Combine	Reverses the action of the **Split** button by recombining the smaller bubbles back into their original bubble. Select any of the smaller bubbles in the group and click **Combine**. Appears only if you specified two **ID** variables.

Select Bubbles

Click a bubble to select it. Note the following:

- Visually, selected bubbles become darker or brighter, and non-selected bubbles are more transparent.
- If the bubble was not filled initially, selection fills it.
- If no bubbles are selected, all of the bubbles are semi-transparent.

When you select a bubble, all of the rows in the data table that correspond to the selected bubble are highlighted. Note the following:

- If the bubble is an aggregate based on an **ID** column, all of the rows for that **ID** are highlighted. Otherwise, the one row represented by that bubble is highlighted.
- If you specify an **ID** and a **Time** variable, selecting a bubble highlights all of the rows for that **ID**, across all of the **Time** levels.

If you select a row from the data table, it is selected in the associated bubble plot. Note the following:

- If you have not specified a **Time** variable, selecting one row from the data table highlights the corresponding bubble in the plot.
- If you have specified a **Time** variable, selecting one row from the data table highlights the corresponding bubble for only that time period in the dynamic bubble plot.

Use the Brush Tool

Use the brush tool to temporarily select bubbles and obtain more information about the selected bubbles. When you select bubbles with the brush tool, the corresponding rows are highlighted in the associated data table.

Note: For a more granular examination of the highlighted rows, use the **Tables > Subset** command or the Row Editor. See *Using JMP*.

Bubble Plot Platform Options

The Bubble Plot red triangle menu provides the following options:

Draw Applies a fill or outline.

Filled Fills all of the bubbles.

Outlined Outlines all of the bubbles.

Filled and Outlined Fills and outlines all of the bubbles.

Set Shape Change the shape of the bubble.

You can create a custom shape using JSL. The **Custom** option opens the custom shape. If no custom shape has been created, the **Custom** option uses the default circle shape. For more information about creating custom shapes, see the *Scripting Guide*.

Orient Shapes Orients the shapes as they move in particular directions over time, following the shape of the data.

This option appears only if you have specified a variable for **Time**.

Trail Bubbles Shows the past history of bubbles as a semi-transparent trail. See "Example of Specifying Only a Time Variable".

Note the following:

- This option appears only if you have specified a variable for **Time**.
- If you do not want to see the bubble labels, select the **Label > None** option.

Trail Lines Shows the past history of bubbles as connected line segments. See "Example of Specifying Only a Time Variable".

Note the following:

- This option appears only if you have specified a variable for **Time**.
- If you do not want to see the bubble labels, select the **Label > None** option.

Label Changes the labels on the plot.

None Labels none of the bubbles in the plot.

All Labels all of the bubbles in the plot.

Selected Labels bubbles only when you select them.

Note: Click and drag on a label to move it.

Color Theme Change the colors representing the high, middle, and low values of the color variable.

This option appears only if you have specified a variable for **Coloring**.

Revert Color Theme Reverts back to the original color theme.

This option appears only if you have applied a color theme.

Legend Shows a legend that describes the colors in the bubble plot.

This option appears only if you have specified a variable for **Coloring**.

Selectable Across Gaps If a bubble is selected, this option keeps the bubble selected during time periods where data is missing. Otherwise, the bubble is not selected during time periods where data is missing.

Show Roles Shows the variables that are used in the bubble plot. You can change and delete the variables. See "Show Roles".

Split All Splits all bubbles into their constituent parts. Unlike the **Split** button, the bubbles do not have to be selected.

This option appears only if you have specified two **ID** variables.

Combine All Combines all constituent bubbles within a group into their larger bubble. Unlike the **Combine** button, the bubbles do not have to be selected.

This option appears only if you have specified two **ID** variables.

Show Time Annotation In bubble plots that contain a time variable, the Show Time Annotation option shows or hides the time that is displayed on the bubble plot.

Lock Scales Prevents axis scales and gradient legend scales from automatically adjusting in response to data or filtering changes.

Fit to Window Determines whether the plot is resized as you resize the JMP window. The default setting is Auto, which bases the scaling on the contents of the plot. For example, a plot with By variables does not stretch to fit the resized window; the plot extends beyond the viewing area. Change the setting to On to always fit the plot inside the window. Change the setting to Off to prevent the plot from resizing.

Aggregation Options Alters how the **X**, **Y**, and **Sizes** roles are computed. By default, the values are calculated using means for **X** and **Y**, and sums for **Sizes**.

X as Sum or Y as Sum Computes the **X** and **Y** values using sums.

Size as Sum Deselecting this option computes **Size** values using means.

Color as Sum Computes the sum of the data values and maps to a color. This option appears only for continuous variables.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

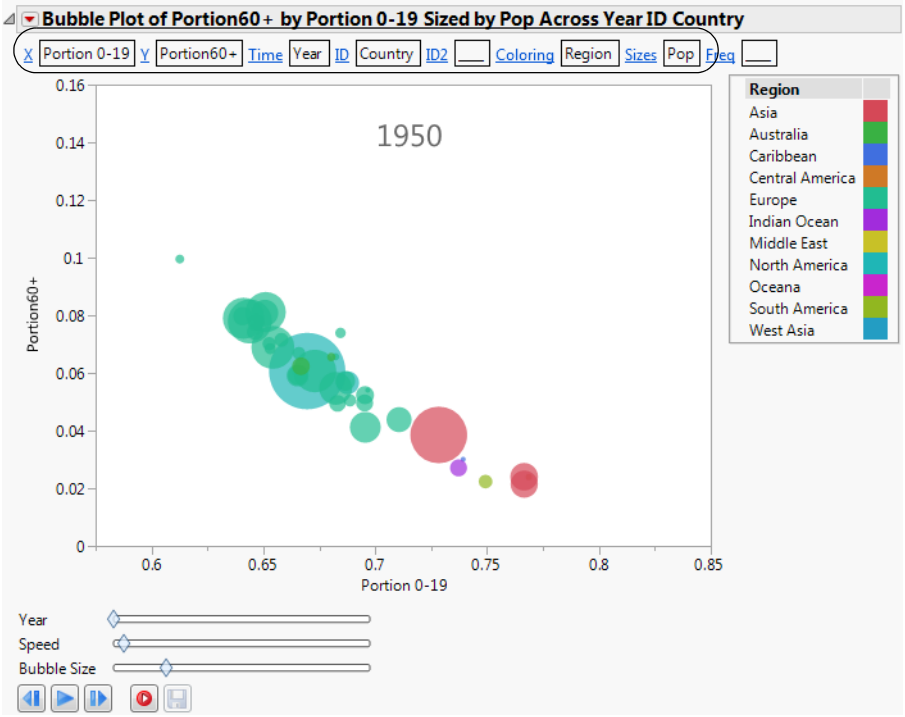
Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Show Roles

Using the **Show Roles** option in the red triangle menu, you can make changes to your existing variables without having to relaunch the platform and start your analysis over.

Follow the instructions in "Example of a Dynamic Bubble Plot" to produce the report window shown in Figure 5.6. (The colors are the JMP default colors.)

Figure 5.6 Example of Bubble Plot with Show Roles Selected



Change the Variable Assigned to a Role

To change the variable assigned to a role, click a blue underlined role name. For example, perform these steps to change the **Coloring** variable in Figure 5.6:

1. Click the **Coloring** link.

   The Select column for Coloring window appears.

2. Click Country.
3. Click **OK**.

   Country now replaces Region as the **Coloring** variable in the bubble plot.

Remove a Variable

To remove an existing variable from the bubble plot, make sure that nothing is selected in the Select column for <*Role*> window, and click **OK**. For example, perform these steps to remove the **Sizes** variable (Pop) in Figure 5.6:

1. Click the Sizes link.

   The Select column for Sizes window appears.
2. Ensure that nothing is selected. If a variable is selected, deselect it by holding down the Ctrl key and clicking on the variable.
3. Click **OK**.

   The Sizes role now appears with an empty box.

Note: The X and Y variables can be changed only and cannot be removed.

Add a Variable

Once you have removed an existing variable from the bubble plot, there are two ways to add a new variable:

- Click the blue underlined role name. See "Change the Variable Assigned to a Role".
- In the data table, click the variable in the column panel, and drag it into the empty role box.

Additional Examples of the Bubble Plot Platform

- "Example of Specifying Only a Time Variable"
- "Example of Specifying Only ID Variables and Splitting a Bubble"
- "Example of a Static Bubble Plot"
- "Example of a Bubble Plot with a Categorical Y Variable"

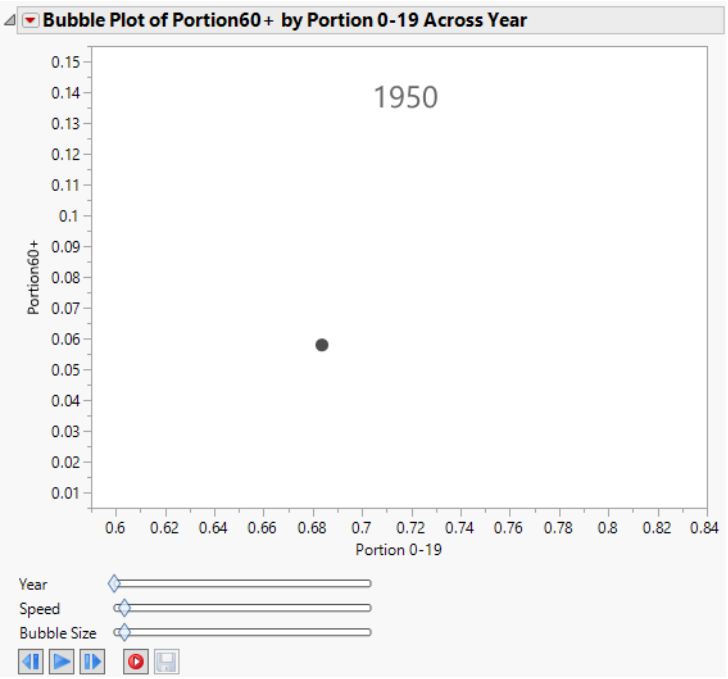
Example of Specifying Only a Time Variable

For dynamic bubble plots, you might specify only a **Time** variable and no **ID** variable. The resulting bubble plot contains a single moving bubble that tracks the series as the **Time** value changes.

1. Select **Help > Sample Data Library** and open PopAgeGroup.jmp.
2. Select **Graph > Bubble Plot**.

3. Select Portion60+ and click **Y**.
4. Select Portion 0-19 and click **X**.
5. Select Year and click **Time**.
6. Click **OK**.

Figure 5.7 The Initial Report Window with a Time Variable

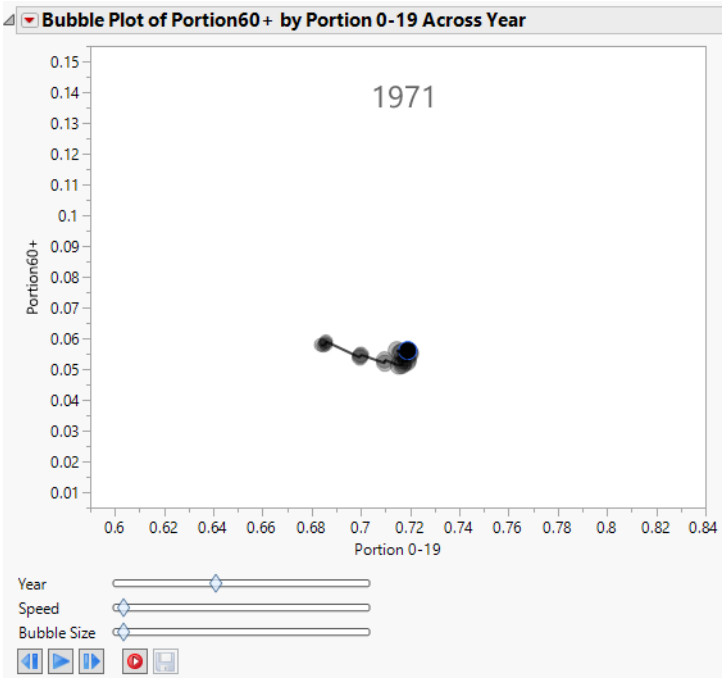


7. Click the bubble to select it.

   All rows in the data table are also highlighted.
8. Click the Bubble Plot red triangle and select **Trail Bubbles > All** and **Trail Lines > All**.
9. Click the play button.

   The bubble plot animates, showing a trail for the single bubble.

Figure 5.8 Animated Bubble Plot over Time



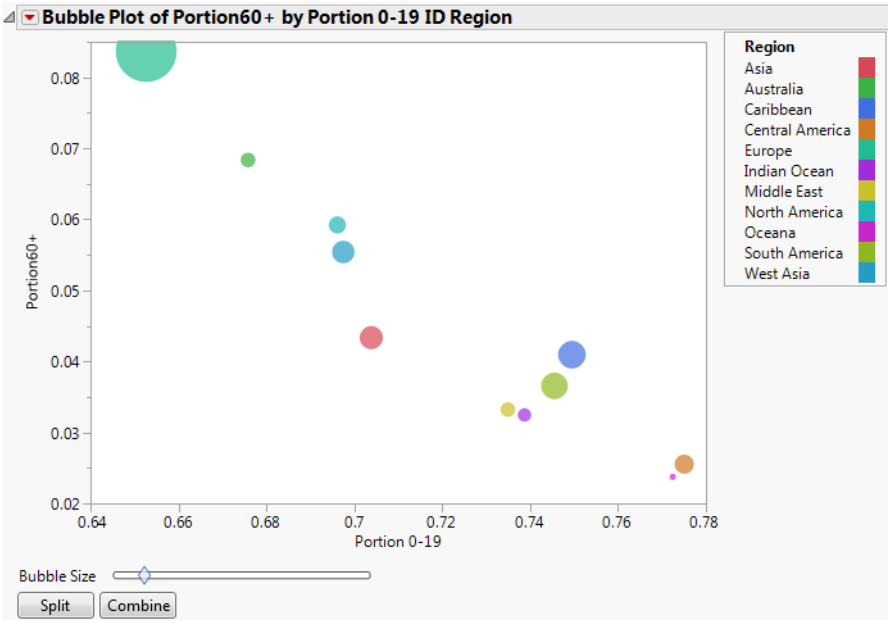
Example of Specifying Only ID Variables and Splitting a Bubble

For static bubble plots, you might specify one or two **ID** variables and no **Time** variable. The resulting bubble plot contains a bubble at each **ID** value. Note that although this bubble plot is static, you can perform splitting on bubbles.

1. Select **Help > Sample Data Library** and open PopAgeGroup.jmp.
2. Select **Graph > Bubble Plot**.
3. Select Portion60+ and click **Y**.
4. Select Portion 0-19 and click **X**.
5. Select Region and Country and click **ID**.
6. Select Region and click **Coloring**.
7. Click **OK**.

   The initial report window appears. (Figure 5.9 uses the default JMP colors.)

Figure 5.9 Initial Report Window with ID Variables

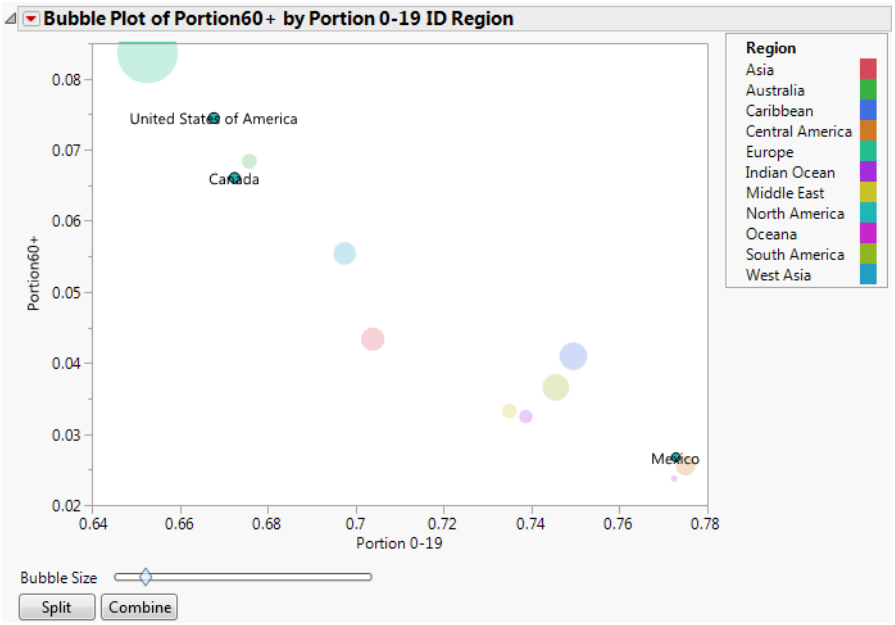


Split the bubble representing the region of North America into countries.

8. Click the bubble representing North America (hover over a bubble to see its label, or use the legend to find the color of North America.)
9. Click **Split**.

   You see that the North America bubble has split into three bubbles, representing the countries within the region of North America (the United States of America, Canada, and Mexico).

Figure 5.10 Splitting the North America Bubble



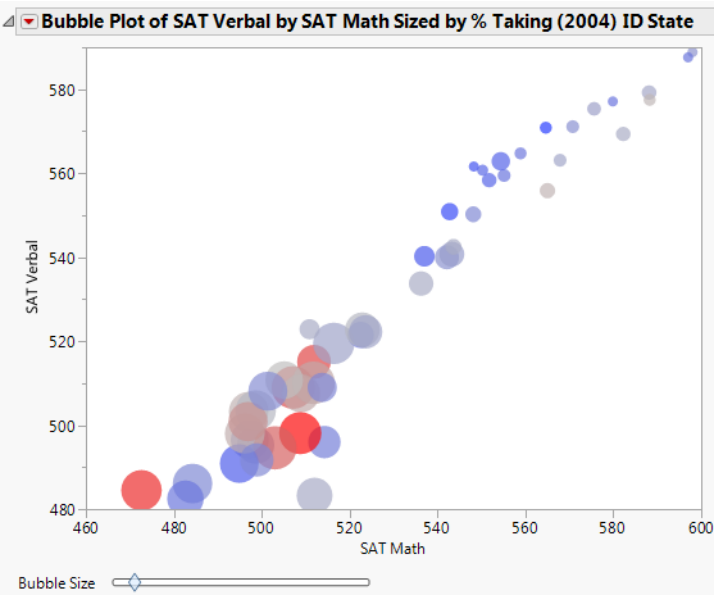
Example of a Static Bubble Plot

This example uses the SATByYear.jmp sample data table, which contains SAT verbal and math test scores for a selection of the US population in 2004.

1. Select **Help > Sample Data Library** and open SATByYear.jmp.
2. Select **Graph > Bubble Plot**.
3. Select SAT Verbal and click **Y**.
4. Select SAT Math and click **X**.
5. Select State and click **ID**.
6. Select % Taking (2004) and click **Sizes**.
7. Click **OK**.

   The report window appears. (Figure 5.11 uses the default JMP colors.)

Figure 5.11 The Static Bubble Plot Report Window



You draw the following conclusions:

- Higher verbal scores appear to be associated with higher math scores, since the two track very closely in the bubble plot. This signifies a correlation between verbal and math scores.
- The larger bubbles represent the US states that have a high percentage of individuals taking the SAT test in 2004. These larger bubbles are all grouped together in the lower left of the graph. This shows that when a state has a high percentage of individuals taking the test, both the math and verbal scores are low.

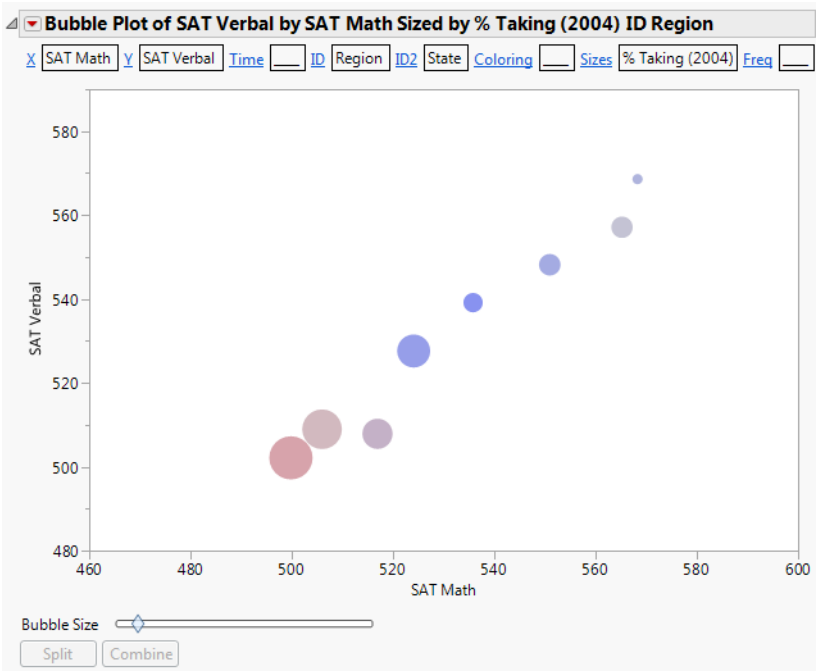
Instead of grouping the bubbles primarily by state, group the bubbles primarily by region:

1. Click the Bubble Plot red triangle and select **Show Roles**.
2. Click the **ID** link.
3. Select Region and click **OK**.

   Region is now the primary **ID** variable.
4. Click the **ID2** link.
5. Select State and click **OK**.

   State is now the secondary **ID** variable.

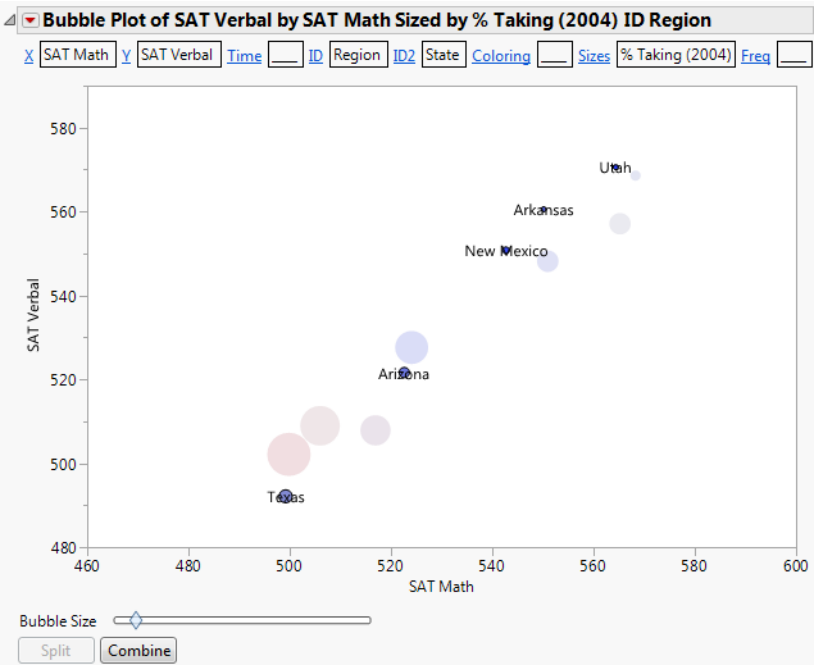
Figure 5.12 Example of Bubble Plot Grouped by Region and State



6. Click the bubble that represents the Southwest region (hover over a bubble or click it to see its label).
7. Click **Split**.

   Now the bubbles are split by the secondary **ID** variable, which is State. You now see each state within the Southwest region.

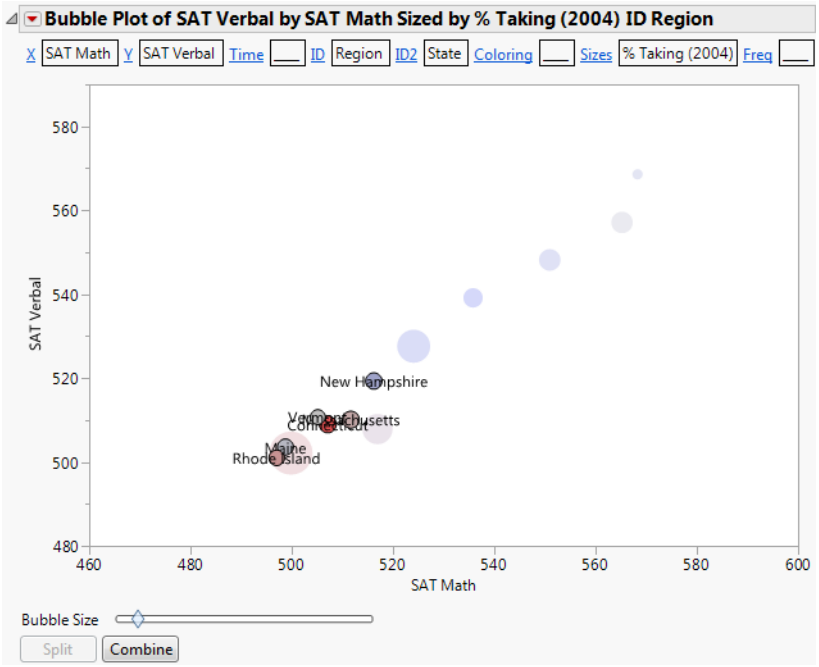
Figure 5.13 Example of Southwest Region Split by State



You see that there is significant variation between the scores from the Southwest states.

8. Click **Combine** to combine the southwest states again.
9. To do a comparison, click the New England bubble (hover over a bubble or click it to see its label).
10. Click **Split**.

Figure 5.14 Example of New England Region Split by State



You see that the New England states do not have as much variation as the Southwest states.

Example of a Bubble Plot with a Categorical Y Variable

All of the examples shown so far use continuous Y variables. If you use a categorical (nominal or ordinal) Y variable, the bubble plot appears differently.

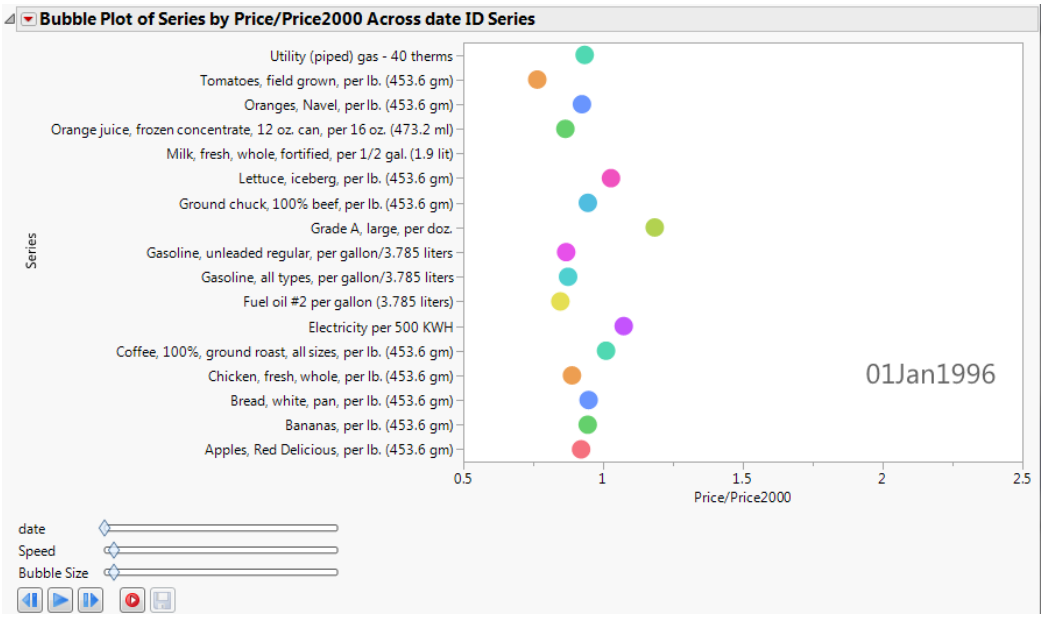
This example uses the blsPriceData.jmp sample data table, which shows the price of commodities over several years. Because the value of the US dollar changes over time, a column named Price/Price2000 shows the ratio of a commodity's price at any given time to the price in the year 2000.

1. Select **Help > Sample Data Library** and open blsPriceData.jmp.
2. Select **Graph > Bubble Plot**.
3. Select Series and click **Y**.
4. Select Price/Price2000 and click **X**.
5. Select date and click **Time**.
6. Click **OK**.

   The report window appears. (Figure 5.15 uses the JMP default colors.)

This produces a bubble plot that, when animated by clicking the play button, shows the price bubbles moving side to side according to their price ratio.

Figure 5.15 Static Example of Animated Bubbles



For easier readability, add grid lines:

7. Double-click the categorical axis.
8. In the Y Axis Settings window, select **Show Grid**.
9. Click **OK**.

To animate the bubble plot, click the play button. The price bubbles move side to side, according to their price ratio.

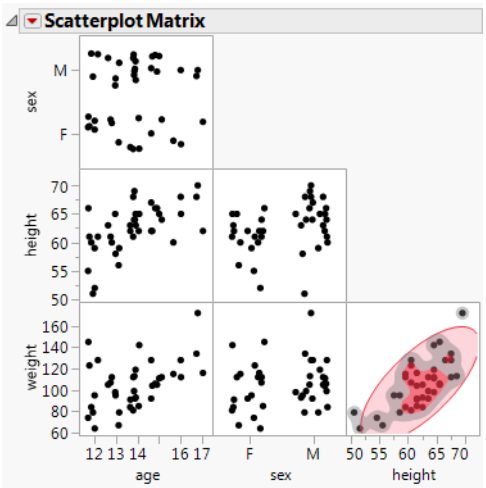
Chapter 6

Scatterplot Matrix

View Multiple Bivariate Relationships Simultaneously

Using the Scatterplot Matrix platform, you can assess the relationships between multiple variables simultaneously. A scatterplot matrix is an ordered collection of bivariate graphs. For further analysis, you can customize the scatterplots with density ellipses for all of your data, or for only groups of your data.

Figure 6.1 Example of a Scatterplot Matrix



Contents

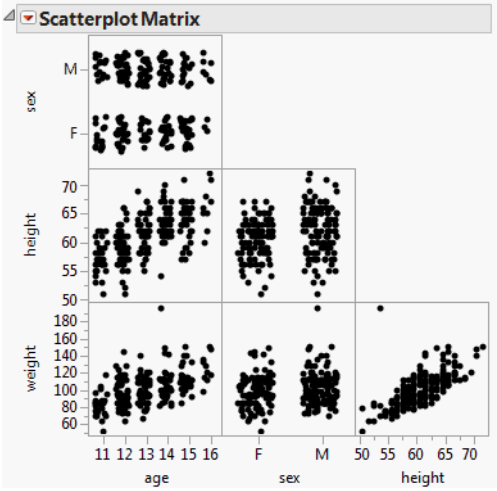
Example of a Scatterplot Matrix 175
Launch the Scatterplot Matrix Platform 176
Change the Matrix Format 178
The Scatterplot Matrix Window 179
Scatterplot Matrix Platform Options 179
Example Using a Grouping Variable 181
Create a Grouping Variable 182

Example of a Scatterplot Matrix

This example shows you how to create a scatterplot matrix.

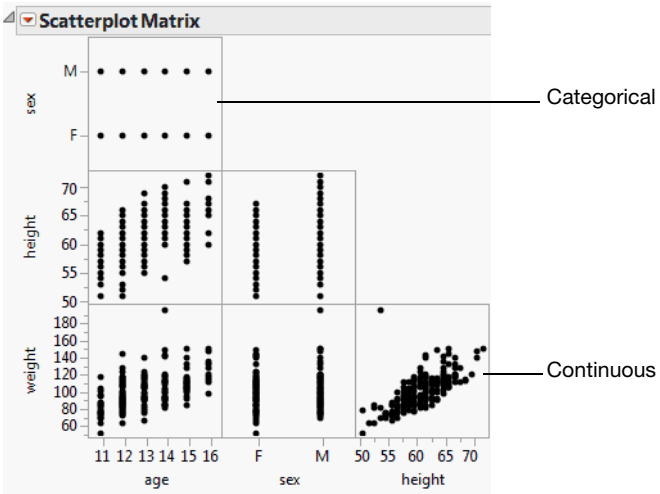
1. Select **Help > Sample Data Library** and open Students.jmp.
2. Select **Graph > Scatterplot Matrix**.
3. Select age, sex, height, and weight and click **Y, Columns**.
4. Click **OK**.

Figure 6.2 Example of a Scatterplot Matrix



In this example, you can see that the graph for weight versus height is different from the graph for sex versus age. If you turn off jitter by clicking on the red triangle menu and selecting **Points Jittered**, the difference becomes even more pronounced.

Figure 6.3 Example of a Scatterplot Matrix with No Jitter

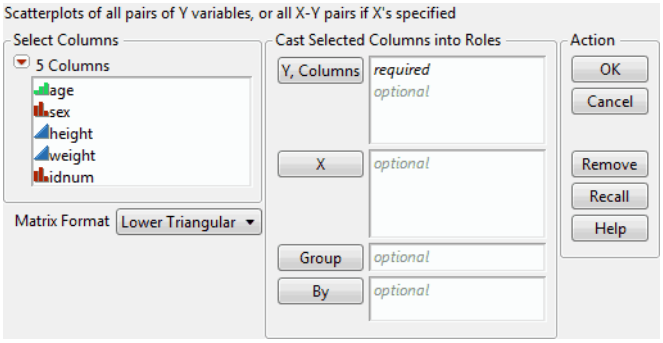


The weight versus height graph shows continuous data, and the sex versus age graph shows categorical data.

Launch the Scatterplot Matrix Platform

Launch the Scatterplot Matrix platform by selecting **Graph > Scatterplot Matrix**.

Figure 6.4 The Scatterplot Matrix Launch Window



Y, Columns, X Specify columns for *Y* and *X*.

- If you assign variables to the **Y, Columns** role only, they appear on both the horizontal and vertical axes.
- If you assign variables to both the **Y, Columns** and **X** role, then the **Y, Columns** variables appear on the vertical axis. The **X** variables appear on the horizontal axis. This

approach enables you to produce rectangular matrices, or matrices that have different, yet overlapping, sets of variables forming the axes of the matrix.

Group If you assign a variable to the **Group** role, you can add shaded density ellipses for each level of the **Group** variable. See "Example Using a Grouping Variable".

By This option produces a separate scatterplot matrix for each level of the **By** variable. If two **By** variables are assigned, a separate graph for each possible combination of the levels of both **By** variables is produced.

Matrix Format The **Matrix Format** can be one of three arrangements: **Upper Triangular**, **Lower Triangular**, or **Square**. See "Change the Matrix Format".

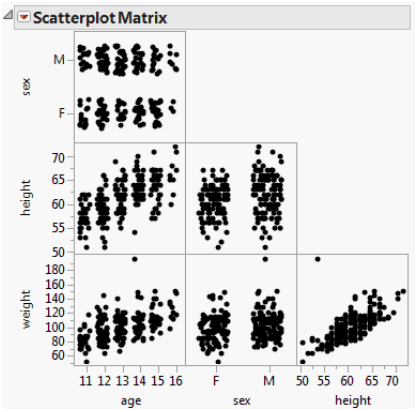
After you click **OK**, the Scatterplot Matrix window appears. See "The Scatterplot Matrix Window".

Change the Matrix Format

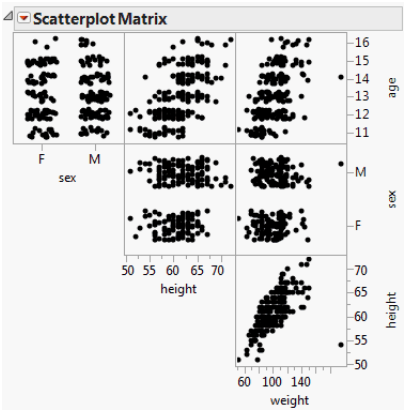
The **Matrix Format** can be one of three arrangements: **Upper Triangular**, **Lower Triangular**, or **Square**.

Figure 6.5 Examples of Matrix Formats

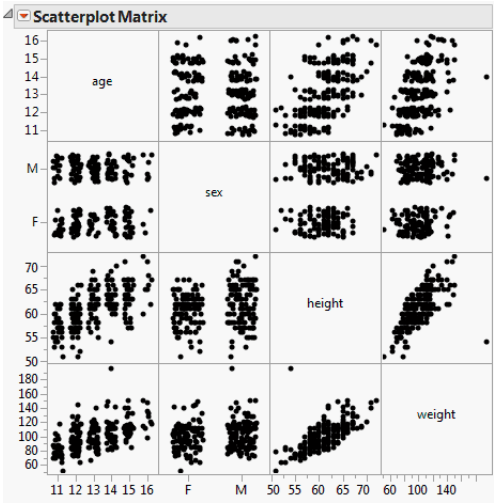
Lower Triangular



Upper Triangular



Square



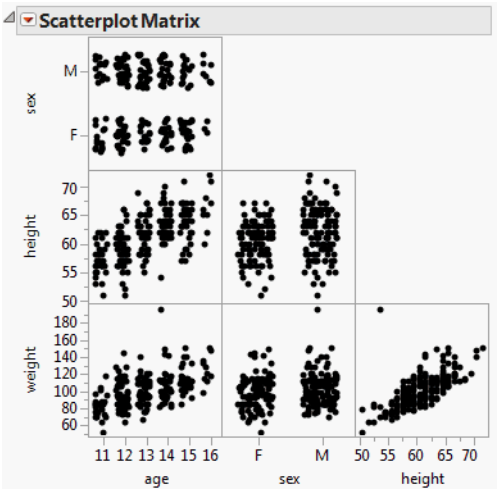
The Scatterplot Matrix Window

The Scatterplot Matrix window shows an ordered grouping of bivariate graphs. In each graph, you can examine the relationships between each pair of variables.

Follow the instructions in "Example of a Scatterplot Matrix" to produce the plot shown in Figure 6.6.

Note: For information about additional options, see "Scatterplot Matrix Platform Options".

Figure 6.6 Example of a Scatterplot Matrix Window



Replace variables in the plot by dragging and dropping a variable, in one of two ways: swap existing variables by dragging and dropping a variable from one axis to the other axis; or, click a variable in the Columns panel of the associated data table and drag it onto an axis. This feature is not available for matrices in the Square format.

Scatterplot Matrix Platform Options

The Scatterplot Matrix red triangle menu provides the following options:

Show Points Shows or hides the points in the scatterplots.

Points Jittered Turns the jittering of the points in the scatterplot on or off. This option is available when at least one variable is either ordinal or nominal.

Fit Line Fits a simple regression line and its mean confidence interval to the scatterplots.

Density Ellipses Shows or hides the outline and area of the density ellipses. See "Example Using a Grouping Variable".

Shaded Ellipses Colors the area within each ellipse. See "Example Using a Grouping Variable".

Ellipses Coverage Enables you to select an α-level for the ellipses to cover.

Ellipses Transparency Enables you to select the transparency of the shaded ellipses, where 0 is completely transparent and 1 is completely opaque.

Ellipses Color Enables you to select a color for the outline and the area within an ellipse.

Nonpar Density Shows or hides the nonparametric density, which represents the areas where the data points are the most dense. The nonparametric density estimation is helpful when you have a lot of points and the density of the points is difficult to see.

There are two quantile density contours. One contour includes 50% of the smoothed density, and the other contour includes 90% of the smoothed density. Since the percentage is based on the smoothed density, the percentage may not agree with the actual proportion of points within a contour.

Group By In the Group By window, you can perform the following actions:

- If you did not select a **Group** variable in the launch window, you can add one now.
- If you did select a **Group** variable in the launch window, you can remove the existing **Group** variable, or you can replace the **Group** variable.

See "Example Using a Grouping Variable".

Lock Scales Prevents axis scales and gradient legend scales from automatically adjusting in response to data or filtering changes.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

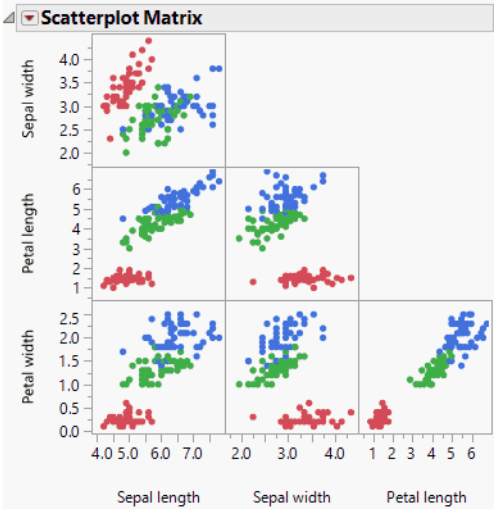
Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Example Using a Grouping Variable

This example shows you how to create a scatterplot matrix using a grouping variable.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Graph > Scatterplot Matrix**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Columns**.
4. Select Species and click **Group**.
5. Click **OK**.

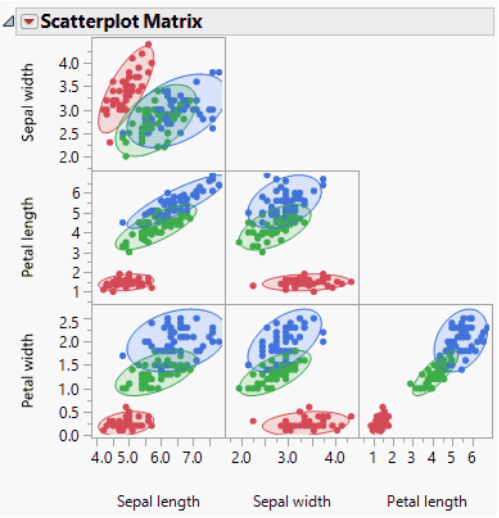
Figure 6.7 Initial Example Using a Grouping Variable



Make the groupings stand out by adding ellipses and shading:

6. Click the Scatterplot Matrix red triangle and select **Density Ellipses > Density Ellipses**.
7. Click the Scatterplot Matrix red triangle and select **Density Ellipses > Shaded Ellipses**.

Figure 6.8 Example of a Scatterplot Matrix with Ellipses



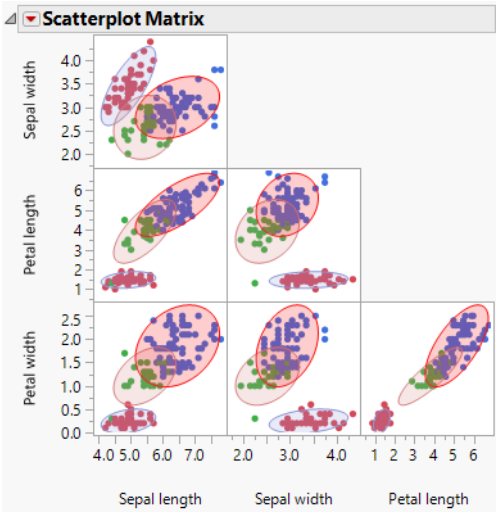
Create a Grouping Variable

If your data does not already have a grouping variable, you can create one using the Cluster platform. Using the Iris.jmp data, assume that the Species column does not exist. You know that the data comes from three species of Iris flowers, so you want to create three clusters within a group.

1. Using the Iris.jmp sample data table, select **Analyze > Clustering > Hierarchical Cluster**.
2. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Columns**.
3. Click **OK**.
4. Click the Hierarchical Clustering red triangle and select **Number of Clusters**.
5. Type 3 to represent the three different Iris species.
6. Click **OK**.
7. Click the Hierarchical Clustering red triangle and select **Save Clusters**.
8. Close the Hierarchical Cluster report window, and go back to the Iris.jmp data table.

   You can see that a Cluster column has been added to the Iris.jmp data table.
9. Perform the Scatterplot Matrix analysis. Follow the instructions in "Example Using a Grouping Variable", but use Cluster as the grouping variable.

Figure 6.9 Example of a Scatterplot Matrix Using a Cluster Variable



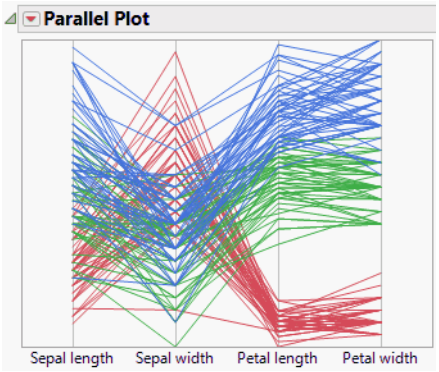
Chapter 7

Parallel Plots

View Patterns in Multidimensional Data by Plotting Parallel Coordinates

Using parallel plots, you can visualize each cell in a data table. Parallel plots draw connected line segments that represent each row in a data table. Parallel plots were initially developed by Inselberg (1985) and later popularized by Wegman (1990).

Figure 7.1 Example of a Parallel Plot



Contents

Example of a Parallel Plot 187
Launch the Parallel Plot Platform 189
The Parallel Plot 190
Interpreting Parallel Plots 190
Parallel Plot Platform Options 192
Additional Examples of the Parallel Plot Platform 193
Examine Iris Measurements 193
Examine Student Measurements 194

Example of a Parallel Plot

This example uses the Dogs.jmp sample data table, which contains histamine level measurements for 16 dogs that were given two different drugs. The histamine levels were taken at zero, one, three, and five minutes. Examine the variation in the histamine levels for each drug.

1. Select **Help > Sample Data Library** and open Dogs.jmp.

   To see the differences by drug, color the parallel plot lines by drug:

2. Select **Rows > Color or Mark by Column**.
3. Select drug.

   If the selection shown in your Colors menu is **JMP Default**, morphine is assigned the color red and trimeth is assigned the color blue.

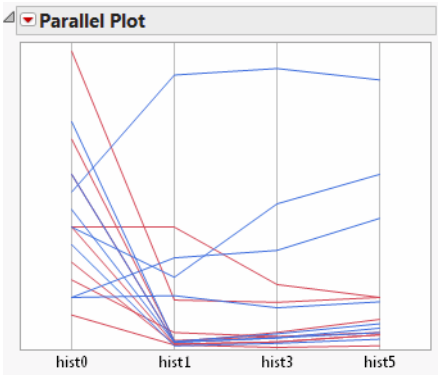
4. Click **OK**.

Create the parallel plot:

5. Select **Graph > Parallel Plot**.
6. Select hist0, hist1, hist3, and hist5 and click **Y, Response**.
7. Click **OK**.

   The report window appears.

Figure 7.2 Parallel Plot of Histamine Variables



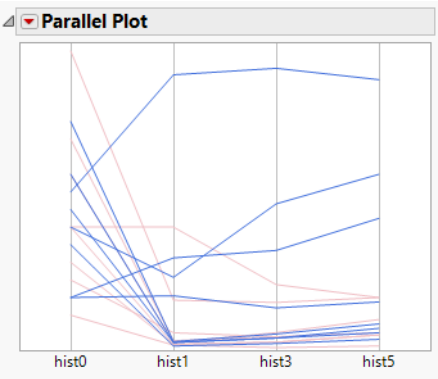
Each connected line segment represents a single observation. Click a line segment to see which observation (or row) it corresponds to in the data table.

For further exploration, isolate the trimeth values:

8. Select **Rows > Data Filter**.
9. Select drug and click Add .
10. Select trimeth.

    Only the trimeth values are highlighted in the parallel plot.

Figure 7.3 Trimeth Values Highlighted



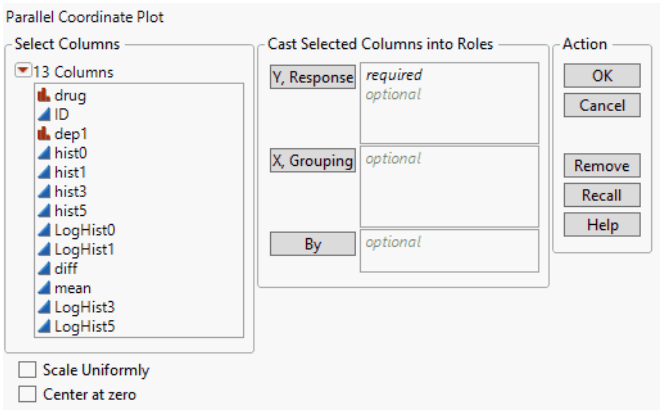
You observe the following about the histamine levels for dogs given trimeth:

- For most of the dogs, the histamine levels had a sharp drop at one minute.
- For four of the dogs, the histamine levels remained high, or rose higher. You might investigate this finding further, to determine why the histamine levels were different for these dogs.

Launch the Parallel Plot Platform

Launch the Parallel Plot platform by selecting **Graph > Parallel Plot**.

Figure 7.4 The Parallel Plot Launch Window



Y, Response Variables appear on the horizontal axis of the parallel plot. These values are plotted and connected in the parallel plot.

X, Grouping Produces a separate parallel plot for each level of the variable.

By Identifies a column that creates a report consisting of separate analyses for each level of the specified variable.

Scale Uniformly Represents all variables on the same scale, adding a *y*-axis to the plot. Without this option, each variable is on a different scale.

To allow for proper comparisons, select this option if your variables are measured on the same scale.

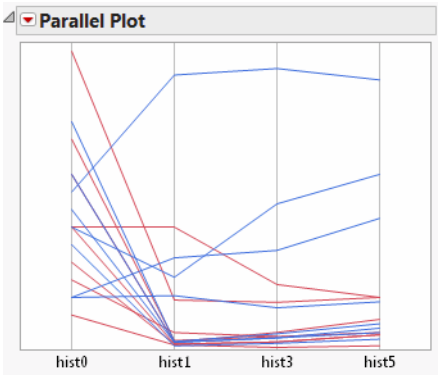
Center at zero Centers the parallel plot (not the variables) at zero.

After you click **OK**, the Parallel plot appears.

The Parallel Plot

To produce the plot shown in Figure 7.5, follow the instructions in "Example of a Parallel Plot".

Figure 7.5 The Parallel Plot Report



A parallel plot is one of the few types of coordinate plots that show any number of variables in one plot. However, the relationships between variables might be evident only in the following circumstances:

- when the variables are side by side
- if you assign a color to a level of a variable to track groups
- if you select lines to track groups

Tip: You can add reference lines for specification limits. For information, see *Using JMP*.

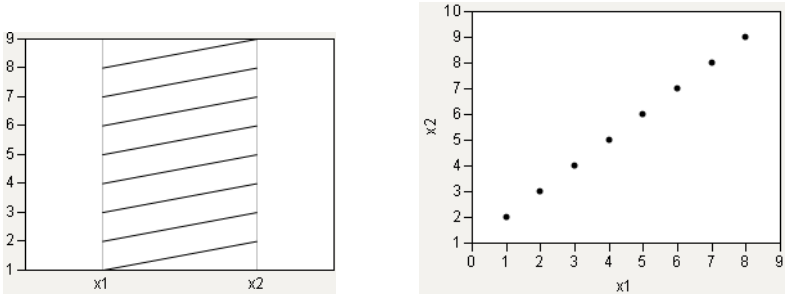
Interpreting Parallel Plots

To help you interpret parallel plots, compare the parallel plot with a scatterplot. In each of the following figures, the parallel plot appears on the left, and the scatterplot appears on the right.

Strong Positive Correlation

The following relationship shows a strong positive correlation. Notice the coherence of the lines in the parallel plot.

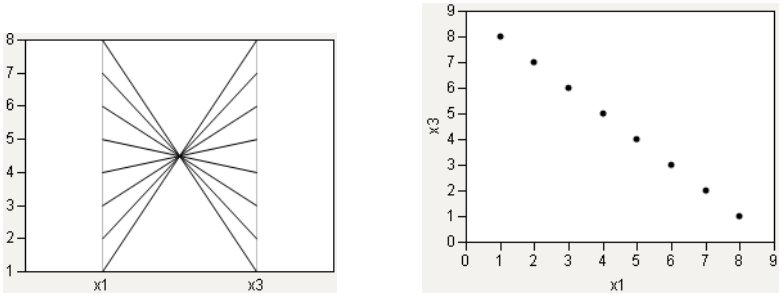
Figure 7.6 Strong Positive Correlation



Strong Negative Correlation

A strong negative correlation, by contrast, shows a narrow neck in the parallel plot.

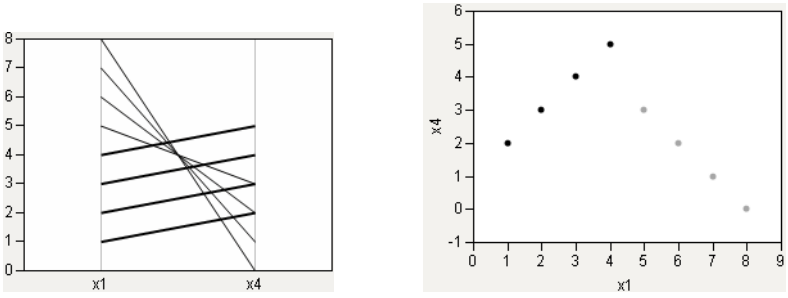
Figure 7.7 Strong Negative Correlation



Collinear Groups

Now, consider a case that encompasses both situations: two groups, both strongly collinear. One has a positive slope, the other has a negative slope. In Figure 7.8, the positively sloped group is highlighted.

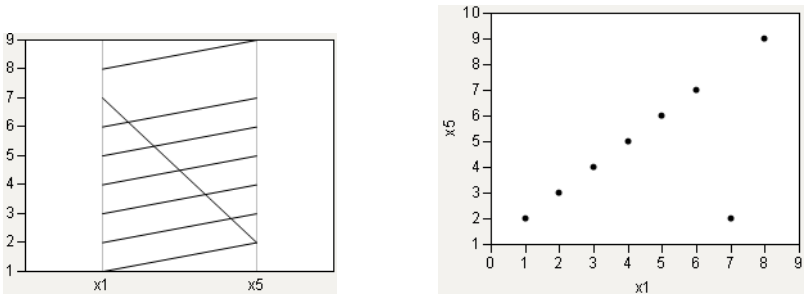
Figure 7.8 Collinear Groups: Parallel Plot and Scatterplot



Single Outlier

Finally, consider the case of a single outlier. The parallel plot shows a general coherence among the lines, with a noticeable exception.

Figure 7.9 Single Outlier: Parallel Plot and Scatterplot



Related Information

- "Additional Examples of the Parallel Plot Platform"

Parallel Plot Platform Options

The Parallel Plot red triangle menu provides the following options:

Show Reversing Checkboxes Reverses the scale for one or more variables.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

For more information about the context menu options that appear when you right-click a parallel plot, see *Using JMP*.

Additional Examples of the Parallel Plot Platform

- "Examine Iris Measurements"
- "Examine Student Measurements"

Examine Iris Measurements

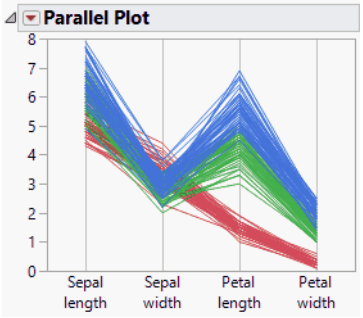
The following example uses the Fisher's Iris data set (Mardia, Kent, and Bibby 1979). The Iris.jmp sample data table contains measurements of the sepal length and width and petal length and width in centimeters for three species of Iris flowers: setosa, versicolor, and virginica. To find characteristics that differentiate the three species, examine these measurements.

Examine Three Species in One Parallel Plot

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Graph > Parallel Plot**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Response**.
4. Select the **Scale Uniformly** check box.
5. Click **OK**.

   The report window appears.

Figure 7.10 Three Species in One Parallel Plot



In this parallel plot, the three species are all represented in the same plot. The colors correspond to the three species:

- Blue corresponds to virginica.
- Green corresponds to versicolor.
- Red corresponds to setosa.

From the parallel plot, you observe the following:

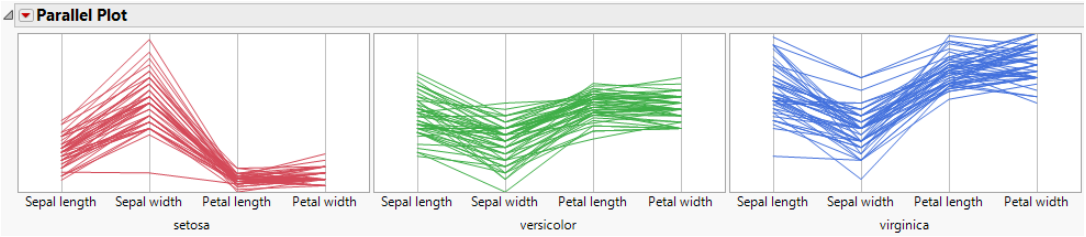
- For sepal width, the setosa values appear to be higher than the virginica and versicolor values.
- For petal width, the setosa values appear to be lower than the virginica and versicolor values.

Examine Three Species in Different Parallel Plots

1. From the Iris.jmp sample data table, select **Graph > Parallel Plot**.
2. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Response**.
3. Select Species and click **X, Grouping**.
4. Click **OK**.

   The report window appears.

Figure 7.11 Three Species in Different Parallel Plots



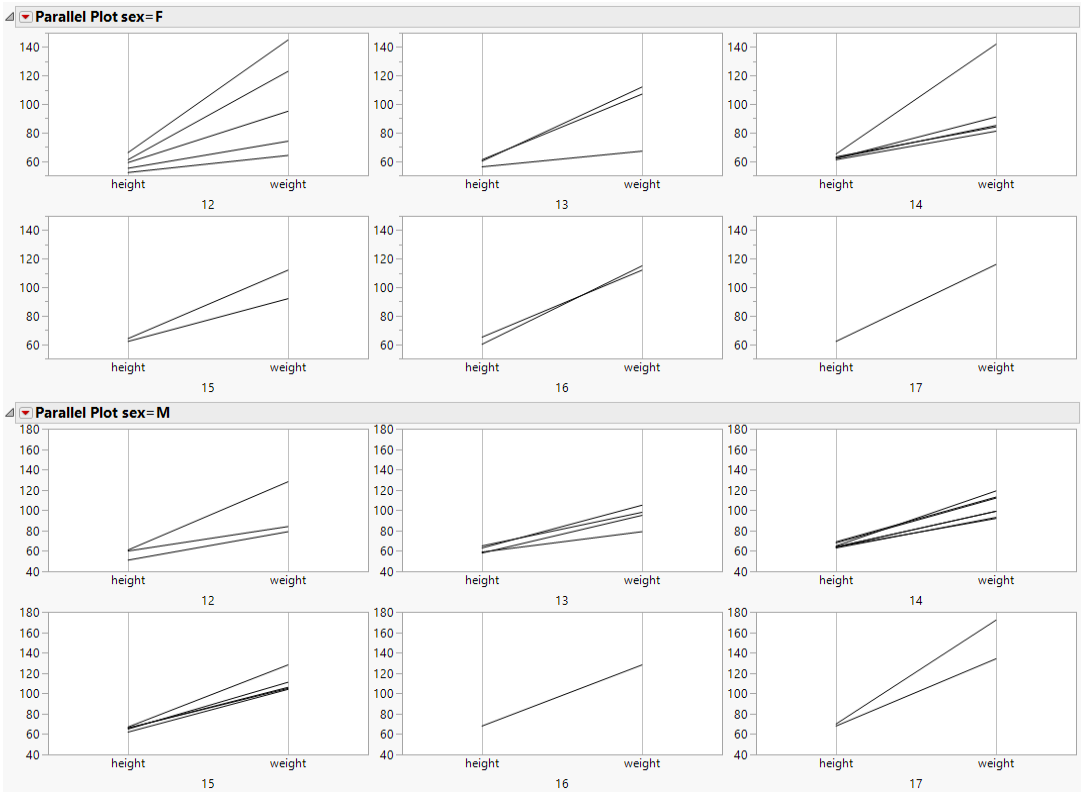
Each species is represented in a separate parallel plot.

Examine Student Measurements

The following example uses the Big Class.jmp sample data table, which contains data on age, sex, height, and weight for 40 students. Examine the relationships between different variables.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Graph > Parallel Plot**.
3. Select height and weight and click **Y, Response**.
4. Select age and click **X, Grouping**.
5. Select sex and click **By**.
6. Select the **Scale Uniformly** check box.
7. Click **OK**.

Figure 7.12 Height and Weight by Sex, Grouped by Age



You observe the following:

- Among the 13-year-old females, one female's weight is lower than the other females in her age group. If you click the line representing the lower weight, the respective individual (Susan) is highlighted in the data table.
- Among the 14-year-old females, one female's weight is higher than the other females in her age group. If you click the line representing the higher weight, the respective individual (Leslie) is highlighted in the data table.

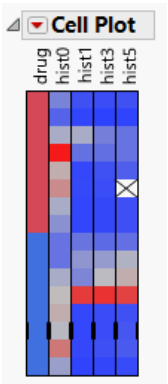
Chapter 8

Cell Plots

View Color-Intensity Plots of Variables

Using cell plots, you can visualize each cell in a data table. Cell plots are direct representations of a data table, since they draw a rectangular array of cells where each cell corresponds to a data table entry. Cell plots were popularized by genomics applications to browse large numbers of values for gene expression levels.

Figure 8.1 Example of a Cell Plot



Contents

Example of a Cell Plot 199
Launch the Cell Plot Platform 200
The Cell Plot 200
Cell Plot Platform Options 201
Right-Click Menu for Cell Plots 202
Additional Example of the Cell Plot Platform 203

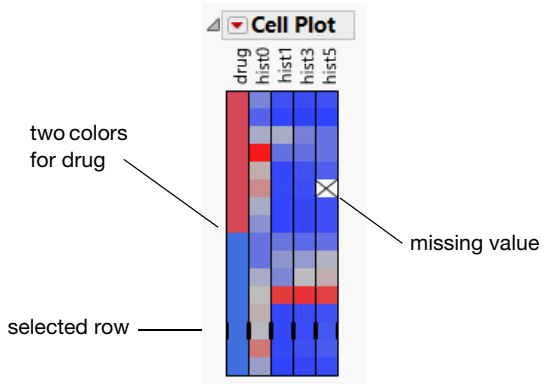
Example of a Cell Plot

Learn how to create and analyze a cell plot in JMP. This example uses the Dogs.jmp sample data table, which contains histamine level measurements for 16 dogs that were given two different drugs. The histamine levels were taken at zero, one, three, and five minutes. Examine the variation in the histamine levels for each drug.

1. Select **Help > Sample Data Library** and open Dogs.jmp.
2. Select the third row from the bottom (row 14).
3. Select **Graph > Cell Plot**.
4. Select drug, hist0, hist1, hist3, and hist5 and click **Y, Response**.
5. Click **OK**.

   The report window appears.

Figure 8.2 Dogs.jmp cell plot



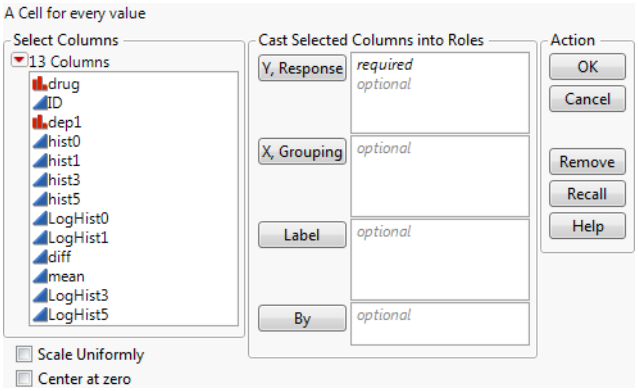
Notice the following:

- There are two types of drugs, represented by two distinct colors.
- Histamine levels are assigned colors from a gradient of blue to red.
- Any missing values are delineated by an X.
- The third row from the bottom is selected, and black lines appear next to the cells.

Launch the Cell Plot Platform

Launch the Cell Plot platform by selecting **Graph > Cell Plot**.

Figure 8.3 The Cell Plot Launch Window



Y, Response Variables appear on the horizontal axis of the cell plot. Each cell represents a value.

X, Grouping Produces a separate cell plot for each level of the variable.

Label Labels each row by the specified variable. See "Additional Example of the Cell Plot Platform".

By Identifies a column that creates a report consisting of separate analyses for each level of the variable.

Scale Uniformly Represents all variables on the same scale. Without this option, each variable is on a different scale.

Center at zero Centers the cell plot at zero.

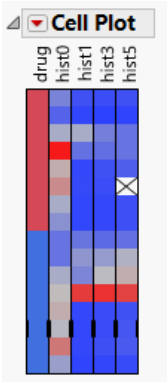
After you click **OK**, the Cell Plot window appears.

The Cell Plot

The cell plot appears with a one-to-one correspondence of a colored cell representing each data table entry. Colors are assigned to each cell based on the range and type of values found in the column.

Note: Any rows that are excluded in the data table are also hidden in the cell plot.

Figure 8.4 The Cell Plot



Note: To produce the plot shown in Figure 8.4, follow the instructions in "Example of a Cell Plot".

Note the following information about cell plots:

- Nominal variables use a distinct color for each level. You can customize nominal and ordinal colors using the **Value Colors** property of data columns, available through the **Column Info** command.
- Continuous variables are assigned a gradient of colors to show the smooth range of values in the variable.
- Ordinal variables are scaled like continuous variables in order.
- When some outliers are present, the scale uses all but the extreme categories for the 90% middle of the distribution, so that the outliers do not overly influence the scale.

Related Information

- "Additional Example of the Cell Plot Platform"

Cell Plot Platform Options

The Cell Plot red triangle menu provides the following options:

Legend Shows or hides a legend.

Arrange Plots Specifies how many plots to put on the same row before starting the next row of plots.

This option is available only if you specify an **X, Grouping** variable.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Right-Click Menu for Cell Plots

Right-click the cell plot to find the following options:

Graph Type Determines the appearance of the graph. See "Change the Graph Type".

Color Theme Shows a list of color themes that affect continuous variables in color maps. The default color theme is **Blue to Gray to Red** (corresponding to small values to middle values to large values). Use **White to Black** to create a gray-scale plot.

Note: To see custom colors, you must first create them. Select **File > Preferences > Graphs**. In the Color Themes area, click the type of color theme that you want to create, click **New**, and then change the colors. See *Using JMP* for more information about creating custom color themes.

Sort Ascending Sorts the rows of the plot from lowest to highest by the values of a column. To sort, right-click in the plot under a column and select **Sort Ascending**. The entire plot is rearranged to accommodate the sorting. See "Additional Example of the Cell Plot Platform".

Sort Descending Sorts the rows of the plot from highest to lowest by the values of a column. To sort, right-click in the plot under a column and select **Sort Descending**. The entire plot is rearranged to accommodate the sorting.

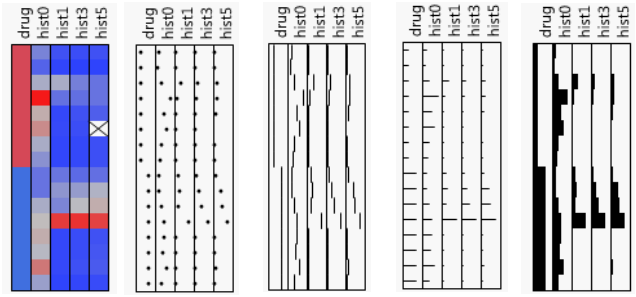
No Separator Lines Draws or removes lines separating the columns.

Note: For more information about the pop-up options that appear when you right-click labels, see *Using JMP*.

Change the Graph Type

Use the **Graph Type** option to change the appearance of the cell plot.

Figure 8.5 Graph Types



Additional Example of the Cell Plot Platform

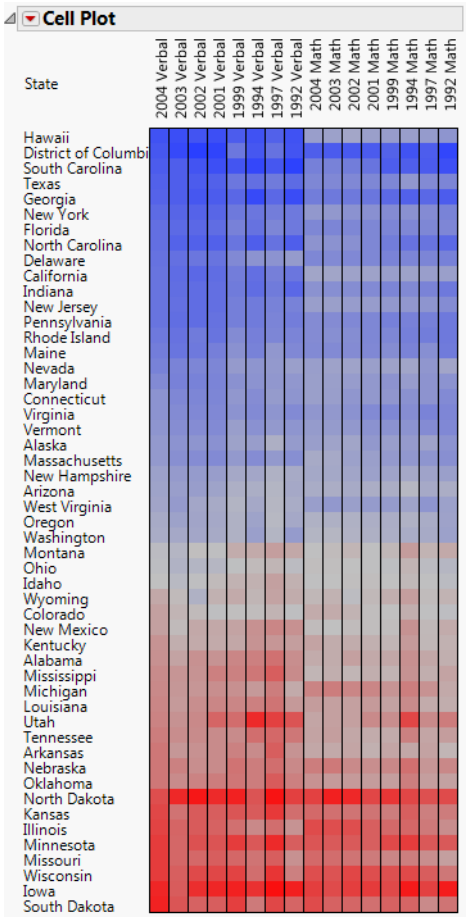
This example uses the SAT.jmp sample data table, which contains SAT test scores (divided into verbal and mathematics portions) for all 50 United States.

1. Select **Help > Sample Data Library** and open SAT.jmp.
2. Select **Graph > Cell Plot**.
3. Select all of the Verbal scores for all of the years, and click **Y, Response**.
4. Select all of the Math scores for all of the years, and click **Y, Response**.
5. Select State and click **Label**.
6. Click **OK**.

   The report window appears.
7. Right-click the plot under 2004 Verbal (the top left cell) and select **Sort Ascending**.

   This sorts the cell plot by the verbal scores for 2004.

Figure 8.6 Cell Plot for SAT Scores



You notice the following:

- Hawaii has the lowest verbal scores for 2004, and South Dakota has the highest verbal scores for 2004.
- There is a contrast between Hawaii's math and verbal scores. Hawaii has average math scores (represented by gray color values) but low verbal scores (represented by blue color values). Hawaii appears to be an outlier, since it has a strikingly different pattern for its math scores and its verbal scores.
- There is very little contrast between North Dakota's math and verbal scores. North Dakota's math and verbal scores are generally high (represented by red color values).

For a description of color themes, see "Right-Click Menu for Cell Plots".

Chapter 9

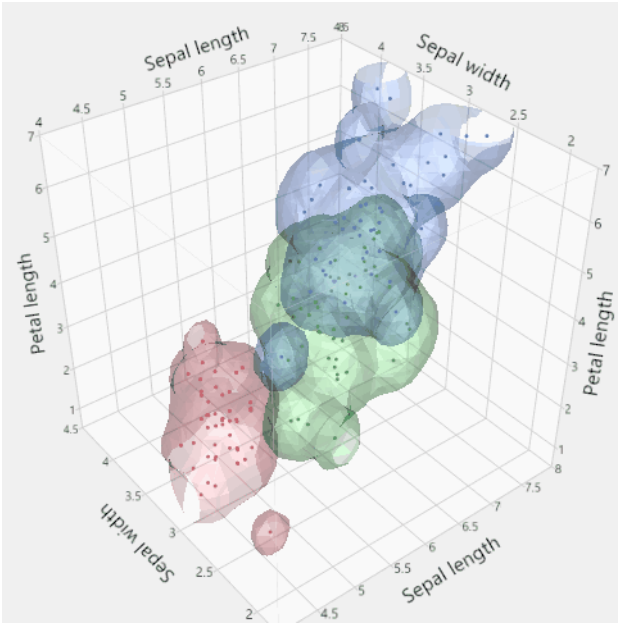
Scatterplot 3D

Create a Rotating Three-Dimensional View of Data

The Scatterplot 3D platform shows the values of numeric columns in the associated data table in a rotatable, three-dimensional view. Up to three columns that you select from the associated data table are displayed at one time.

To help visualize variation in higher dimensions, the 3D scatterplot can show a biplot representation of the points and variables when you request principal components. The most prominent directions of data are displayed on the 3D scatterplot report.

Figure 9.1 Example of a 3D Scatterplot



Contents

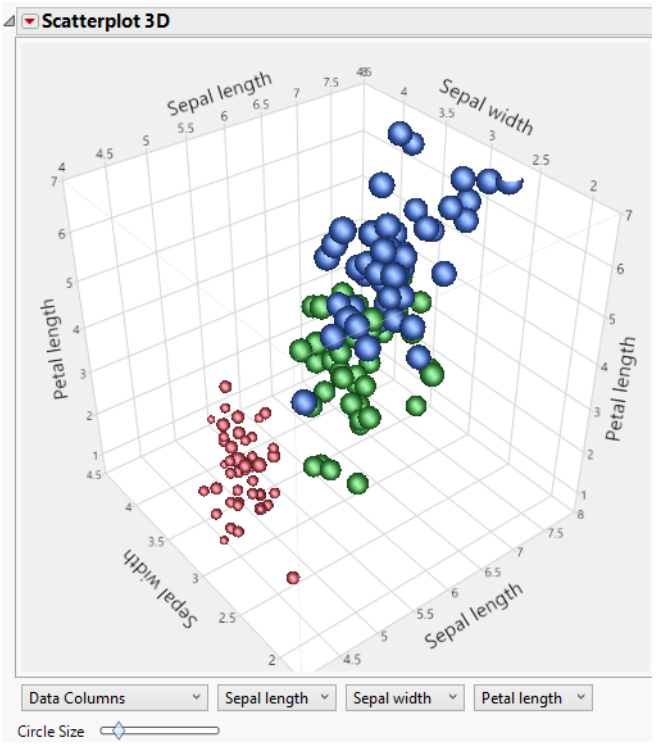
Example of a 3D Scatterplot 207
Launch the Scatterplot 3D Platform 208
The Scatterplot 3D Report 209
Spin the 3D Scatterplot 211
Change Variables on the Axes 212
Adjust the Axes 212
Assign Colors and Markers to Data Points 213
Assign Colors and Markers in the Data Table 213
Scatterplot 3D Platform Options 213
Normal Contour Ellipsoids 216
Nonparametric Density Contours 216
Pop-Up Menu 218
Additional Examples of the Scatterplot 3D Platform 221
Example of an Ungrouped Normal Contour Ellipsoid 221
Example of Grouped Normal Contour Ellipsoids 222
Example of a Grouped Nonparametric Density Contour 223

Example of a 3D Scatterplot

This example uses the Iris.jmp sample data table, which includes measurements of sepal length, sepal width, petal length, and petal width for three species of iris.

1. Open the Iris.jmp sample data table.
2. Select **Graph > Scatterplot 3D**.
3. Select Sepal length, Sepal width, and Petal length, and click **Y, Columns**.
4. Select Petal width and click **Weight**.
5. Click **OK**.

Figure 9.2 Example of an Initial 3D Scatterplot

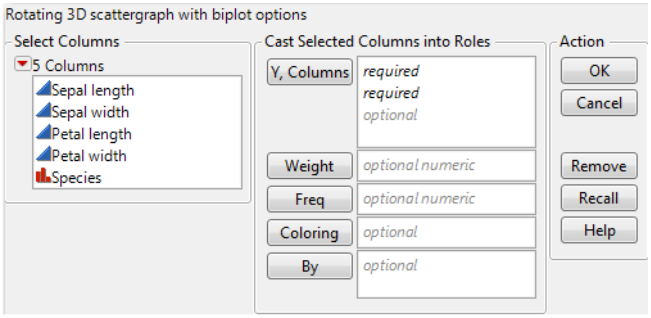


Now you can spin the 3D scatterplot to see the relationships between the variables. In this example, the data points are formatted in blue, red, and green. You might want to spin the scatterplot to see more clearly the relationships between the red and green points.

Launch the Scatterplot 3D Platform

Launch the Scatterplot 3D platform by selecting **Graph > Scatterplot 3D**.

Figure 9.3 The Scatterplot 3D Launch Window



Y, Columns Select the variables to plot on the 3D scatterplot. The order in which you select the variables determines where the data points appear on the axes:

- The first variable appears on the *x* axis.
- The second variable appears on the *y* axis.
- The third variable appears on the *z* axis.

You can assign the remaining variables interactively through the drop-down menus below the scatterplot.

Weight Use the **Weight** variable to:

- Assign a weight (importance or influence) to the data
- Visualize a fourth variable that sizes the points

Note: Red triangle options account for the **Weight** variable. If you do not want this variable accounted for in your analyses, remove it from the launch window.

When you specify a **Weight** variable, JMP draws the points as balls. The balls are scaled so that their volume represents the weight value. You click and drag the **Circle Size** slider below the scatterplot to resize the balls.

Freq Identifies the data table column whose variables assign a frequency to each row. This option is useful when a frequency is assigned to each row in summarized data.

Coloring Colors the points according to the selected variable. If the selected variable is categorical (nominal or ordinal), each category is colored distinctly. If the selected variable is continuous, a gradient of colors is used.

By Produces a separate 3D scatterplot for each By variable value. When two By variables are assigned, a separate graph is produced for each combination of both By variables.

After you click **OK**, the Scatterplot 3D report window appears.

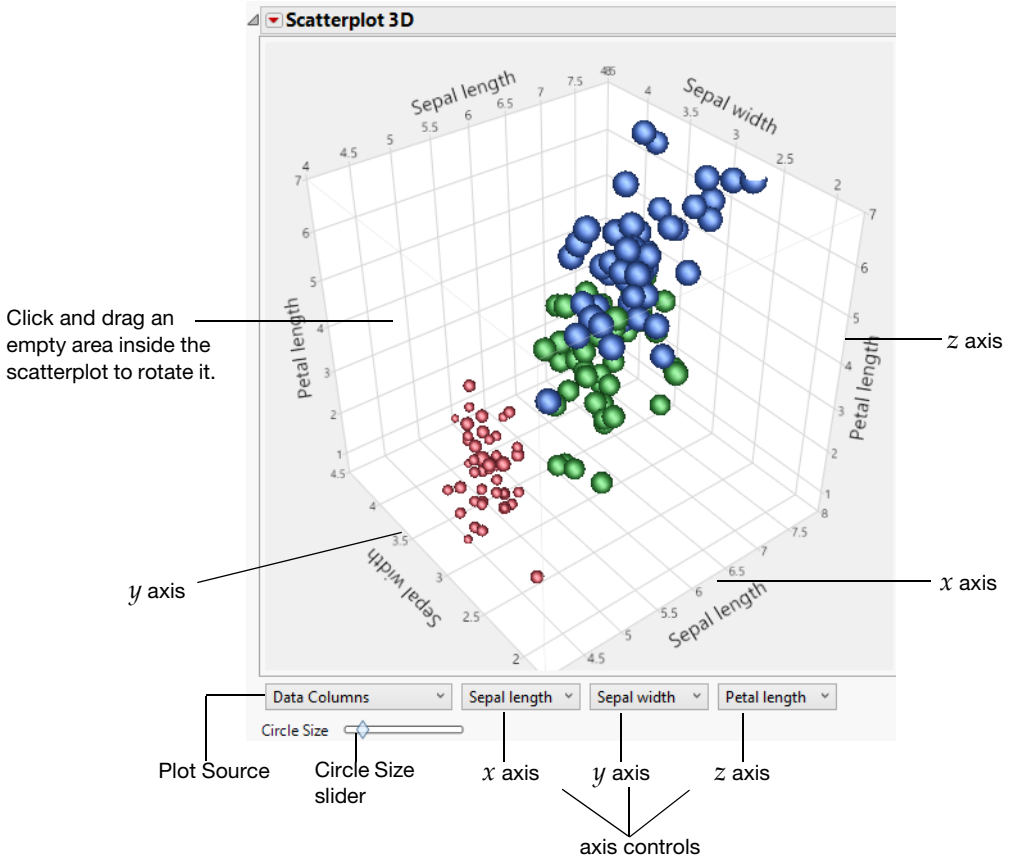
The Scatterplot 3D Report

To produce the 3D scatterplot shown in Figure 9.4, follow the instructions in "Example of a 3D Scatterplot".

The Scatterplot 3D report shows a three-dimensional spinnable view of your data. See Figure 9.4. In the launch window, you select the variables and then create the report. The variables are displayed on the 3D scatterplots' *x*, *y*, and *z* axes. Up to three variables can be displayed at a time.

Note: The Crosshairs tool is not supported in 3D scatterplots.

Figure 9.4 Example of Information Displayed on the Scatterplot 3D Report



Note: Any rows that are excluded in the data table are also hidden in the 3D scatterplot.

Plot source The plot source box indicates the source of the data in the plot.

Circle Size slider Click and drag the **Circle Size** slider to resize the balls while maintaining their relative sizes.

Note: The **Circle Size** slider appears only if you have specified a **Weight** variable.

Axis controls Select which variable appears on each axis. Choose the **Other** option to add a new variable.

Next Axis Set Cycles through the axis controls for any hidden variables. See "Change Variables on the Axes".

Note: The **Next Axis Set** button appears only if your analysis contains more than three variables.

After you create a 3D scatterplot, you can add features such as displaying ellipses around specific data points, showing separate principal components, rotating components, connecting points, and more. See "Scatterplot 3D Platform Options".

You can also assign colors and symbols (or markers) to data points either on the 3D scatterplot itself or in the associated data table. See "Assign Colors and Markers to Data Points" and "Assign Colors and Markers in the Data Table".

Spin the 3D Scatterplot

You spin the 3D scatterplot report in four ways:

- Click and drag an empty area on the 3D scatterplot. The 3D scatterplot spins in the direction you dragged the mouse.

  Note: Click and drag on an empty area on the 3D scatterplot, not on an axis or data point. Dragging the axis rescales the axis. Dragging a data point selects only the point.

- Slide the mouse wheel. The 3D scatterplot spins up and down only.
- Hold down an arrow key. (Before using an arrow on the number keypad, verify that Num Lock is turned off.)
- Hold down the Esc key. The 3D scatterplot spins left and right only.

In each case, the 3D scatterplot spins as long as you hold down the mouse button, arrow key, or Esc key. The spinning also continues as you slide the mouse wheel.

You can also spin the 3D scatterplot continuously in any of these ways:

- Click and drag: Hold down the Shift key, click and drag an empty area on the plot, and release Shift. The faster you drag the mouse, the faster the 3D scatterplot spins.
- Mouse wheel: Hold down the Shift key, slide the wheel, and release the wheel. The 3D scatterplot spins up and down only.
- Arrow keys: Hold down the Shift key and press the arrow key, and then release Shift.
- Esc key: Press Shift+Esc. The 3D scatterplot spins left and right only.

In addition to automatically spinning the plot, you can oscillate the plot. Press Shift+Ctrl and then click and drag the plot. The plot shakes up and down or left to right, depending on the direction in which you dragged the plot.

To stop the spinning or oscillating, click the plot or press the Esc key.

Change Variables on the Axes

The variables on each axis are determined by the order in which you select the variables in the launch window. For example, the first variable that you select is displayed on the *x* axis. The second variable is displayed on the *y* axis, and the third variable is displayed on the *z* axis.

After you create a 3D scatterplot, you can change the variable assigned to an axis, plot a different set of variables, or sequence through all combinations of the variables.

1. To change the variable on a specific axis, select the axis control drop-down menu and select a different variable.
2. To add a different variable, click an axis control drop-down menu, select **Other**, select the variable, and then click **OK**.
3. To sequence through combinations of all variables, click the **Next Axis Set** button until the variables that you want to plot are displayed.

Adjust the Axes

You can manually move or rescale the axis coordinates by clicking and dragging the axis. This option shows a different set of coordinates on the 3D scatterplot. It also lets you change the space displayed between the coordinates (or the coordinate scaling).

You can also specify axis properties by double-clicking the axis and modifying settings in the specifications window.

To Move the Coordinates on the Axis

1. Hover over the middle of the axis.
2. Click and drag the axis.

To Modify Coordinate Scaling

1. Hover over the end of the axis.
2. Click and drag the axis.

To Rescale an Axis Precisely

1. Hover over the middle of the axis (the axis, not the label).
2. Double-click the axis.
3. Change the minimum and maximum coordinate values.

For more information about updating axes, see *Using JMP*.

Assign Colors and Markers to Data Points

Each point in the 3D scatterplot corresponds to a row in the associated data table. To highlight points on the 3D scatterplot, you assign colors and markers to the points. The colors and markers are then displayed on the 3D scatterplot and in the data table.

When you click a point, the following items are selected:

- the point in the 3D scatterplot
- the corresponding row in the associated data table
- the point in any other opened graphs, if applicable

To select one point, click the point.

To select several points, double-click the 3D scatterplot and drag the cursor over the points. A box is displayed to indicate which points are selected.

To deselect points, double-click the 3D scatterplot.

Assign a color or marker to selected data points:

1. To assign a color to the selected point, select **Rows > Colors** and then select the color.
2. To assign a marker to the selected point, select **Rows > Markers** and then select the marker.

Assign Colors and Markers in the Data Table

You can assign colors and markers to rows in the data table. The colors and markers appear next to the row number in the data table and on the 3D scatterplot. This option distinguishes points for each variable, and you can save the settings in the data table. Assigning colors and markers to specific data points (as described in "Assign Colors and Markers to Data Points") only highlights them for the current open graphs.

See *Using JMP* for more information about assigning colors and markers in the data table. For more information about changing the size, quality, or transparency of markers, see "Scatterplot 3D Settings".

Scatterplot 3D Platform Options

The red triangle menu next to Scatterplot 3D contains options to customize the display and to compute, rotate, and save principal or rotated components.

Show Points Shows or hides the data points on the graph.

Show Controls Shows or hides the source and axis controls displayed beneath the 3D scatterplot (Figure 9.4).

Drop Lines Draws lines from each point to the plane created by the *x* and *z* variables that you selected on the launch window.

Connect Points Connects the points with a line. Points can be connected on the data as a whole or in groups. You can also group data by a specific variable.

Jitter Displays small spaces between the data points so that you can see each point more clearly.

Normal Contour Ellipsoids Draws one or more normal contour ellipsoids, that is, three-dimensional ellipses that encompass a specified portion of points. You specify whether you want an ellipsoid for all of the data or for each group. You can also control the size and transparency of the ellipsoids. See "Normal Contour Ellipsoids".

Ellipsoid Coverage Changes the size of normal contour ellipsoids. Type a value *between* 0 and 1, where the greater the value creates a bigger the ellipsoid. The actual values "0" and "1" produce no ellipsoid, so a warning appears if you try to use those values.

This option only appears after you add a normal contour ellipsoid to the 3D scatterplot.

Ellipsoid Transparency Changes the surface of normal contour ellipsoids. The greater the value, the more opaque the ellipsoid. This option only appears after you add a normal contour ellipsoid to the 3D scatterplot.

Nonpar Density Contour Draws nonparametric density contours, which approximately encompass a specified proportion of the points. You specify whether you want a density contour for all of the data or for each group. See "Nonparametric Density Contours".

Drop Line Thickness Changes the width of drop lines. This option only appears after you add drop lines to the 3D scatterplot.

Principal Components Calculates principal components on all variables. This changes the axes of the plot to have principal component scores.

Biplot rays are displayed by default. You can remove them by selecting **Biplot Rays** from the red triangle menu. For more information about principal components, see *Multivariate Methods*.

Std Prin Components Calculates principal components (as with the **Principal Components** option) but scales the principal component scores to have unit variance. If this option is not selected, the scores have variance equal to the corresponding eigenvalue.

With standardized principal components, the correlation between the variables and the principal component scores is equal to the values in the eigenvector. This helps you quickly assess the relative importance of the variables. See *Multivariate Methods*.

Select this option if you want GH' rather than JK' biplots. GH' biplots try to preserve relationships between variables; JK' biplots try to preserve relationships between

observations. The interpoint distance shown by GH' biplots is less meaningful, but the angles of the GH' biplot rays measure correlations better.

Rotated Components Specifies the number of factors that you want to rotate and the rotation method. You rotate components to better align the directions of the factors with the original variables so that the factors might be more interpretable. See *Multivariate Methods*.

Biplot Rays Shows or hides biplot rays that correspond to the principal components. You must have already selected Principal Components, Std Prin Components, or Rotated Components for this option to appear.

Show Ray Labels Shows or hides labels for the biplot rays. You must have already selected Biplot Rays for this option to appear.

Remove Prin Comp Removes principal components, standardized principal components, and rotated components from the scatterplot 3D report. The 3D scatterplot reverts to its original display before principal components were selected. This option, however, does not remove any saved principal components from the data table.

This option only appears after you add principal, standard, or rotated components to the 3D scatterplot.

Save Prin Components Saves the specified number of current principal component scores as new columns in the current data table. These columns also include the formulas used for the principal components. For *n* variables in the components list, *n* principal component columns are created and named Prin1, Prin2, ... Prinn.

This option only appears after you add principal, standard, or rotated components to the 3D scatterplot.

Save Rotated Components Saves all rotated component scores as columns in the current data table. These columns also include the formulas that were used. If you requested *n* rotated components, then *n* rotated component columns are created and named Rot1, Rot2, ... Rotn.

This option only appears after you add rotated components to the 3D scatterplot.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Normal Contour Ellipsoids

A normal contour ellipsoid is a 3-dimensional ellipse that encompasses a specified portion of points. The ellipsoid is computed from a contour of the multivariate normal distribution fit to the points. The ellipsoid is a function of the means, standard deviations, and correlations of variables on the plot. See *Multivariate Methods* for more information about multivariate normal distributions.

When you add an ellipsoid, two formatting options are available:

- *Coverage* changes the portion of data points covered by the ellipsoid. The larger the value, the bigger the ellipsoid.
- *Transparency* changes the surface of the ellipsoid from transparent to opaque. The larger the value, the more opaque the ellipsoid.

The coverage and transparency options also appear in the red triangle menu after you add the ellipsoid.

When you add normal contour ellipsoids to a 3D scatterplot, you specify whether you want an ellipsoid for all of the data or for a specific group of data. The ellipsoid for each set of grouped data is color-coded to differentiate one group from another.

You display and remove normal contour ellipsoids by selecting and deselecting **Normal Contour Ellipsoids** from the red triangle menu.

Related Information

- "Example of an Ungrouped Normal Contour Ellipsoid"
- "Example of Grouped Normal Contour Ellipsoids"

Nonparametric Density Contours

The nonparametric density contour shows contours that approximately encompass a specified proportion of the points. You add nonparametric density contours to see patterns in point density when the scatterplot is darkened by thousands of points.

This feature is particularly valuable when you have many points on a 3D scatterplot; the contours can be so dark that you cannot see the structure. In this situation, you remove the points so that only the contours are displayed. See "Optimizing a Dense Nonparametric Density Contour".

When you add nonparametric density contours to a 3D scatterplot, you specify whether you want a contour for all of the data or for a specific group of data. The contour for each set of grouped data is color-coded to differentiate one group from another.

You display and remove nonparametric density contours by selecting and deselecting **Nonpar Density Contours** from the red triangle menu.

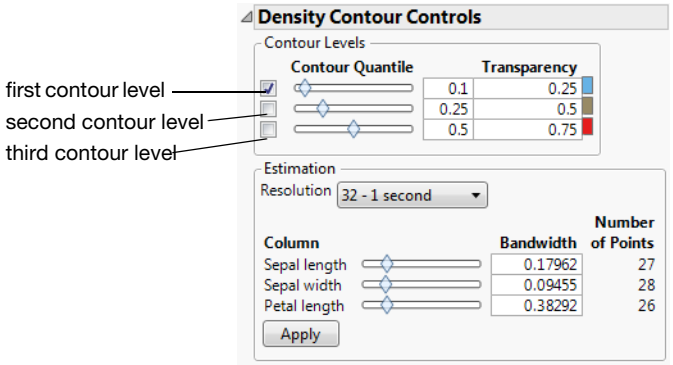
Related Information

- "Example of a Grouped Nonparametric Density Contour"

Density Contour Controls

The Density Contour Controls options are displayed below the 3D scatterplot. These options let you select additional contours and change each contour's formatting.

Figure 9.5 The Density Contour Controls Window



Contour Quantile Controls which contours are shown and lets you customize the contour formatting.

- *Density level* represents the volume and density of the points. As the contours go from smaller to larger values, the contours cover less volume but more dense areas. A 0.9 contour represents the 10% densest part of the total, where the points are closest together. Click and drag the slider below "Contour Quantile," or enter a value next to the slider.
- *Transparency* changes the surface of density contours. The greater the value, the more opaque the contour. Enter a value in the box.
- *Color* changes the color of the contour. Click the colored box and select a different color. (This option only appears for ungrouped density contours.)

Changes to these settings take effect immediately.

Resolution Changes the resolution of the contours. A higher resolution results in a less granular drawing of the contours but takes more time to display.

Column Bandwidth Changes the smoothness of the fitted density. A higher bandwidth results in a smoother fitted density.

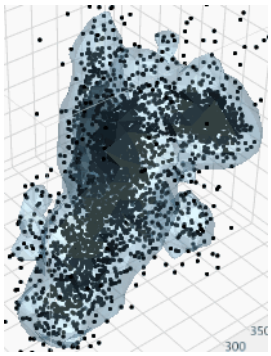
Type a new bandwidth for each variable, or click and drag the sliders. Click **Apply** to display your changes.

Optimizing a Dense Nonparametric Density Contour

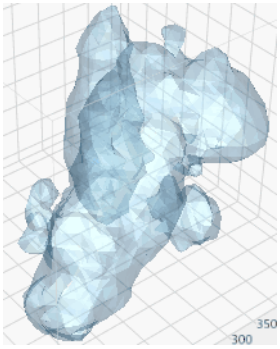
When you have many points on a 3D scatterplot, the contours can be so dark that you cannot see the structure. In this situation, you remove the points so that only the contours are displayed.

To remove points from a 3D scatterplot, select **Show Points** from the red triangle menu. You can further optimize the contours by changing their size, color, and transparency. See "Scatterplot 3D Platform Options".

Figure 9.6 Example of Optimizing a Dense Nonparametric Density Contour



3D scatterplot with density contour and points



3D scatterplot with density contour and no points

Pop-Up Menu

Right-click the 3D scatterplot to find the following options.

Show Legend Shows and hides the color legend for the 3D scatterplot.

Note: This option requires the **Coloring** role.

Reset Returns the orientation of the scatterplot to its original state.

Settings Provides options to change the appearance of the 3D scatterplot. See "Scatterplot 3D Settings".

Hide Lights Border Shows and hides a border that displays the lights. The lights highlight different portions of the 3D scatterplot.

Right-click a light to turn it on or off and to change the color.

Wall Color Changes the color of the 3D scatterplot.

Background Color Changes the color surrounding the 3D scatterplot.

Rows You can color, mark, exclude, hide, and label points that correspond to rows in the associated data table. You must select the points before selecting this option. See *Using JMP*.

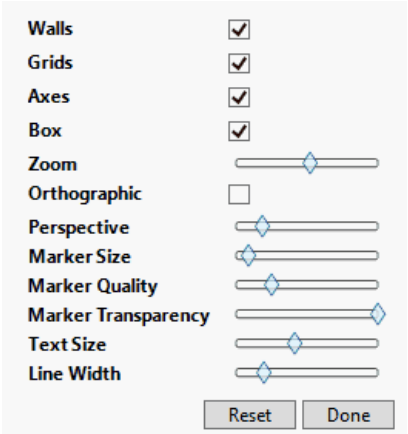
Use Hardware Acceleration Turns hardware acceleration on or off for machines that support acceleration. This option might display the scatterplot faster. If not, try updating your graphics drivers.

Show ArcBall Shows and hides a globe around the 3D scatterplot. This option helps you visualize the rotation of the scatterplot. Select whether you want the ArcBall to appear always, only when you drag the scatterplot, or never.

Scatterplot 3D Settings

To customize properties such as the marker size, text size, and grid lines, right-click the 3D scatterplot and select **Settings**. The Settings window appears. As you modify the settings, a preview appears on the 3D scatterplot.

Figure 9.7 The Scatterplot 3D Settings Window



Note the following:

- Move the sliders left to decrease the selected property or to the right to increase the selected property.
- To move the Settings window around the scatterplot, click and drag the top portion of the window.

The following options are available:

Reset Resets the default settings.

Done Closes the window.

Walls Adds or removes the 3D scatterplot walls. Without walls, the background color of the 3D scatterplot is displayed.

Grids Shows or hides the coordinate lines.

Axes Shows or hides the variable names that appear above each axis.

Box Shows or hides the box. Without the box, the 3D scatterplot is displayed as an open plot.

Zoom Enlarges or shrinks the 3D scatterplot.

Orthographic Changes the view of the scatterplot from 3-dimensional to an orthographic projection. In the orthographic view, the walls of the scatterplot do not converge to a vanishing point. This means that you can compare near and far distances and see the structure between data points.

Note: If you turn off orthographic view and completely decrease the perspective, the walls of the scatterplot do not converge. This is the same effect that you get when you turn on orthographic view.

Perspective Increases or decreases the perspective. Large values create a view that is unnaturally large and visually disorienting. In this case, you need to resize the scatterplot window to show the entire plot.

Marker Size Increases or decreases the size of the data point markers.

Marker Quality Increases and decreases the data marker quality. For example, when you increase the marker quality, some markers have an opaque center. Other symbol markers are formatted in bold. Increase the zoom to see these changes in quality.

Marker Transparency Increases or decreases the transparency of the data markers.

Text Size Increases or decreases the text size.

Line Width Changes the width of the coordinate and axes lines.

Additional Examples of the Scatterplot 3D Platform

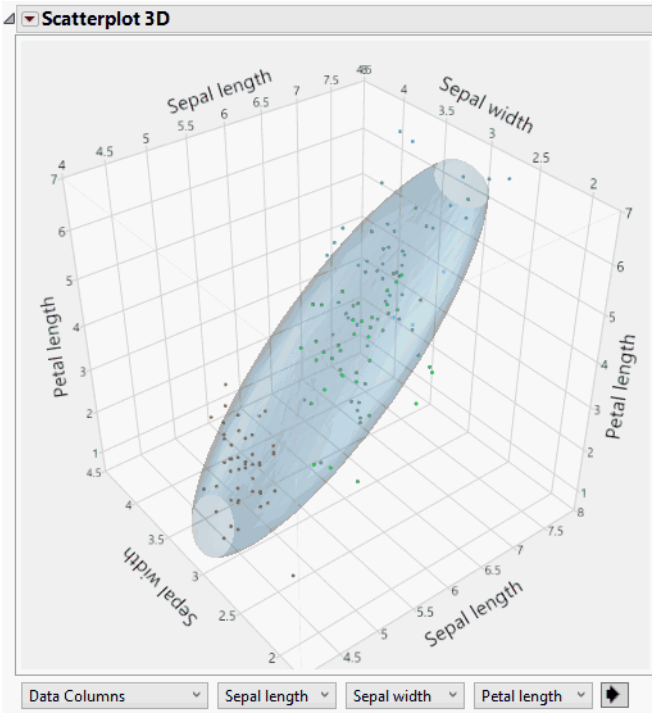
- "Example of an Ungrouped Normal Contour Ellipsoid"
- "Example of Grouped Normal Contour Ellipsoids"
- "Example of a Grouped Nonparametric Density Contour"

Example of an Ungrouped Normal Contour Ellipsoid

This example shows how to add a normal contour ellipsoid to more than 75% of the data points. The ellipsoid is 25% transparent.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Graph > Scatterplot 3D**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Columns**.
4. Click **OK**.
5. Click the Scatterplot 3D red triangle and select **Normal Contour Ellipsoids**. Notice that **Ungrouped** is already selected.
6. Type 0.75 next to Coverage.
7. Type 0.25 next to Transparency.
8. Click **OK**.

Figure 9.8 Example of an Ungrouped Normal Contour Ellipsoid

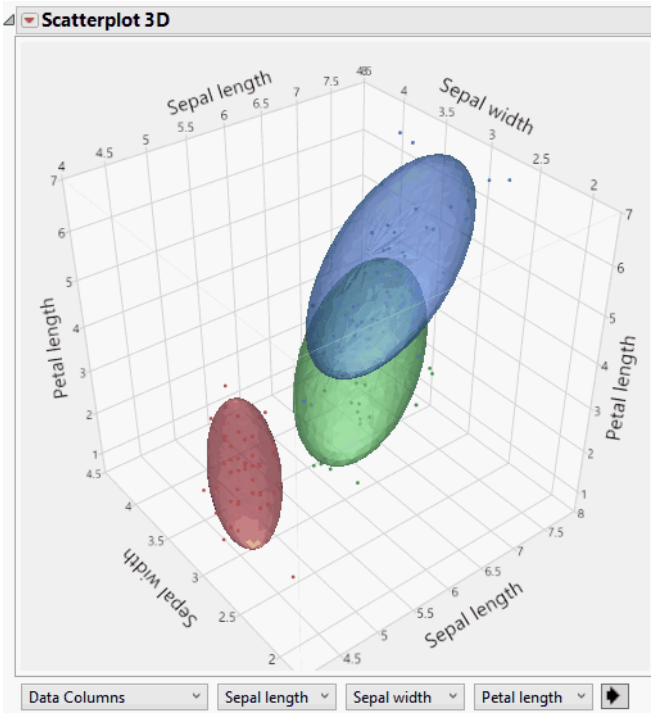


Example of Grouped Normal Contour Ellipsoids

This example shows how to group measurements by species and to format each group with a normal contour ellipsoid. The ellipsoids cover 75% of the data points and are 50% transparent. The contours are color-coded (using JMP default colors) based on species.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Graph > Scatterplot 3D**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Columns**.
4. Click **OK**.
5. Click the Scatterplot 3D red triangle and select **Normal Contour Ellipsoids**.
6. Select **Grouped by Column**.
7. Select **Species**.
8. Type 0.75 next to Coverage.
9. Type 0.5 next to Transparency.
10. Click **OK**.

Figure 9.9 Example of Grouped Normal Contour Ellipsoids

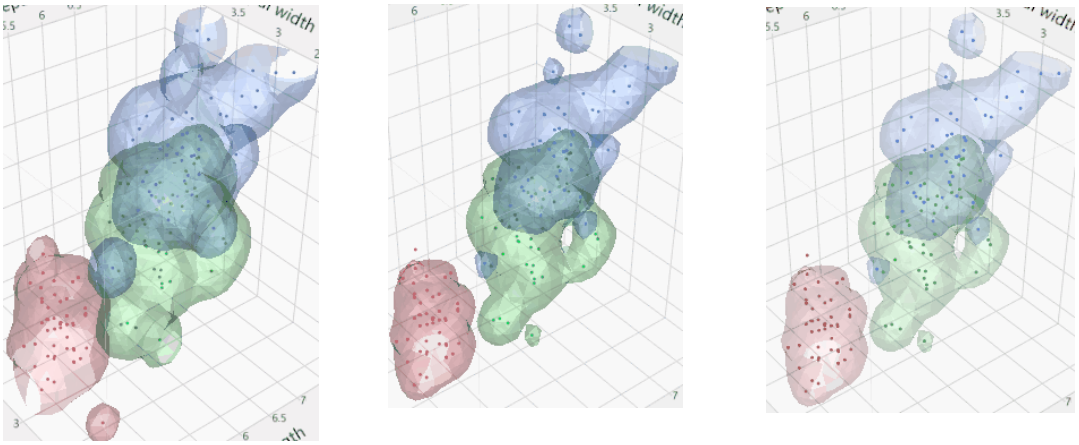


Example of a Grouped Nonparametric Density Contour

This example shows how to group data points and format each nonparametric density contour.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Graph > Scatterplot 3D**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Columns**.
4. Click **OK**.
5. Click the Scatterplot 3D red triangle and select **Nonpar Density Contour**.
6. Select **Grouped by Column**.
7. Select **Species** and click **OK**. A different colored contour is displayed for each of the three species.
8. Type 0.25 in the first Contour Quantile box. 25% of the data points appear outside the contour surfaces, which results in smaller contours.
9. Type 0.15 in the first Transparency box. The contours are 15% opaque.

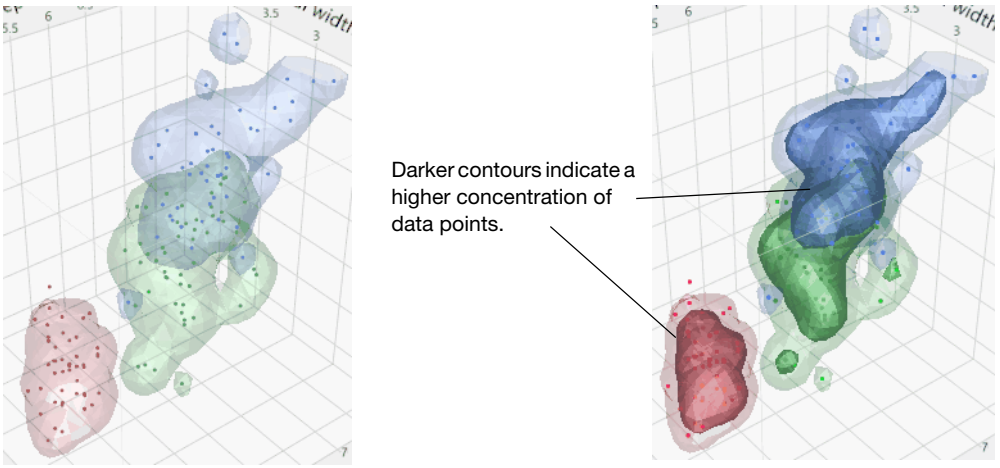
Figure 9.10 Changing the Nonparametric Density Contour Transparency and Density



original contours | decreased density | decreased transparency

10. Select the second check box. The contour quantiles are the same (.25), so the new contours overlap the first ones.
11. Type 0.5 in the second Contour Quantile box. 50% of the data points appear outside the contour surfaces. A second set of contours appears within the first, to further illustrate the density of the data points.

Figure 9.11 Adding a Second Nonparametric Density Contour



one level of contours | two levels of contours

You can now format the second levels of contours and turn on the third level of contours.

The options for formatting the grouped and ungrouped nonparametric density contours are similar. The only difference is that you cannot change the color of each *grouped* nonparametric density contour. See "Scatterplot 3D Platform Options" for options.

Chapter 10

Contour Plots

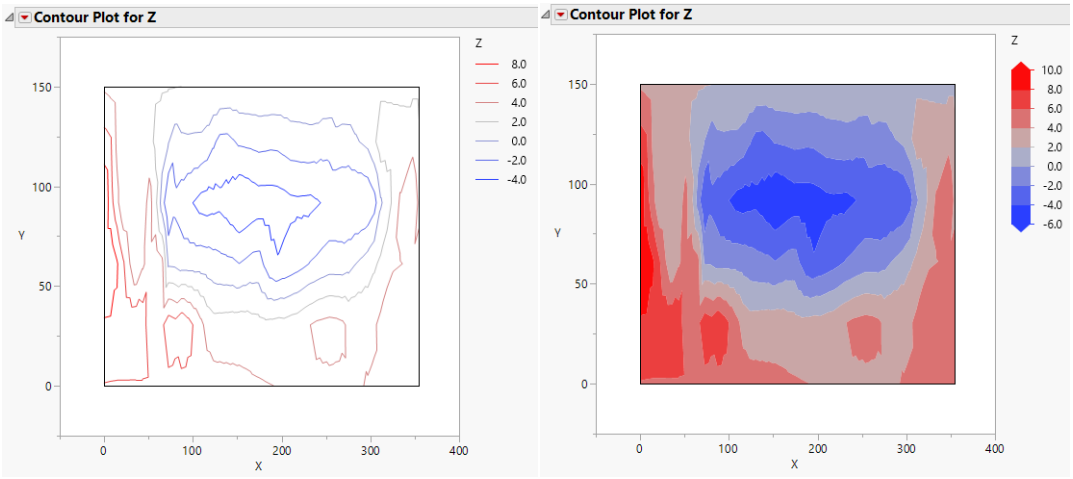
View Multidimensional Relationships in Two Dimensions

The **Contour Plot** command in the **Graph** menu constructs contours of a response in a rectangular coordinate system. A contour plot shows a three-dimensional surface in two dimensions. Contours delineate changes in the third dimension.

Here are some of the options available with the Contour platform:

- specify the number of contour levels
- choose to plot contour lines or filled contours
- show or hide data points
- label contours with response values
- define and use a custom coloring scheme

Figure 10.1 Examples of Contour Plots



Contents

Example of a Contour Plot 229
Example Using the Contour Platform 229
Example of a Contour Plot in Graph Builder 230
Launch the Contour Plot Platform 231
The Contour Plot 233
Contour Plot Platform Options 233
Fill Areas 235
Contour Specification 235
Contour Plot Save Options 239
Use Formulas for Specifying Contours 240
Additional Example of Contour Plots 241

Example of a Contour Plot

To create a contour plot, you can use either the Contour platform or Graph Builder. This section provides examples for both.

- To see the example using the Contour platform, see "Example Using the Contour Platform".
- To see the example using Graph Builder, see "Example of a Contour Plot in Graph Builder".

Example Using the Contour Platform

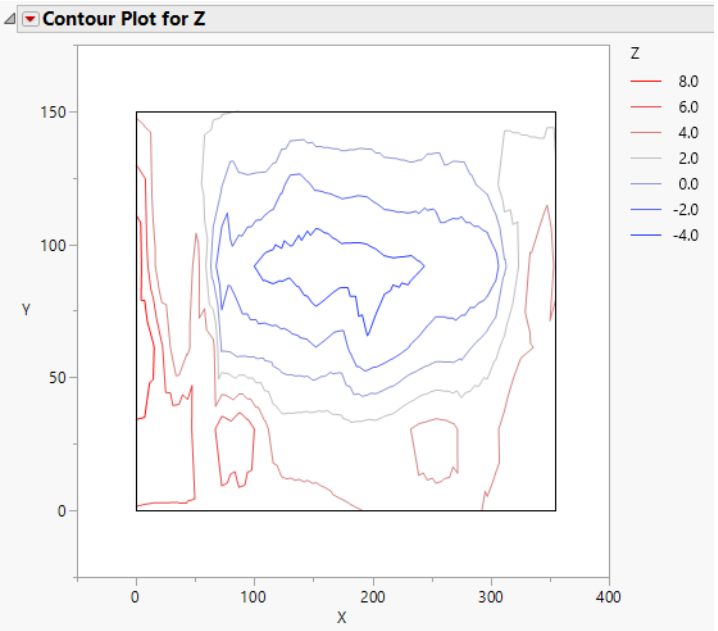
To create a contour plot, you need two variables for the *x*- and *y*-axes and at least one more variable for contours. You can also use several *y*-variables. This example uses the Little Pond.jmp sample data table. X and Y are coordinates of a pond. Z is the depth.

1. Select **Help > Sample Data Library** and open Little Pond.jmp.
2. Select **Graph > Contour Plot**.
3. Select the X and Y coordinates and click **X**.
4. Select the depth, Z, and click **Y**.

   Note: In a contour plot, the X1 and X2 roles are used for the X and Y axes.

5. Click **OK**.

Figure 10.2 Example of a Contour Plot with Legend



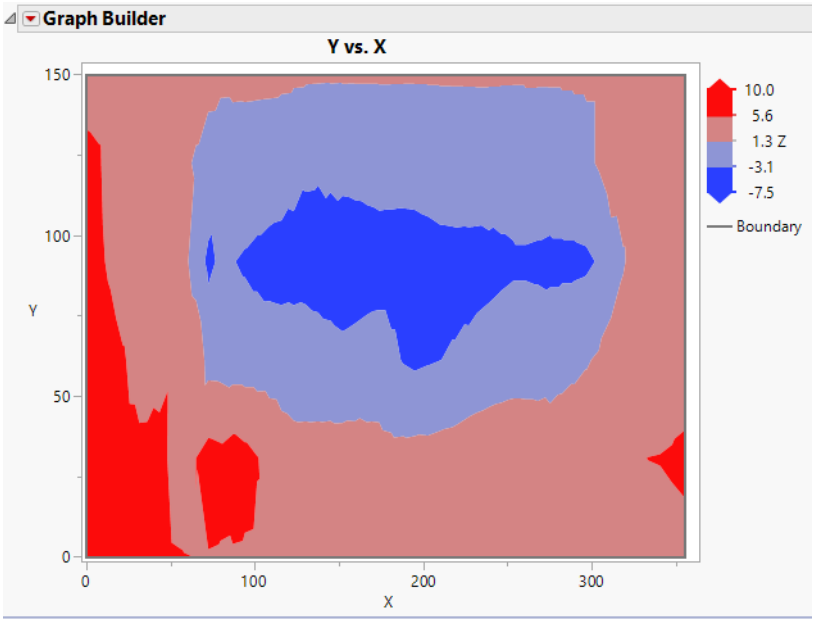
The *x*- and *y*-axes are coordinates and the contour lines are defined by the depth variable. This contour plot is essentially a map of a pond showing depth. To see the contours more clearly, click the red triangle next to Contour Plot for Z and select **Fill Areas**.

Example of a Contour Plot in Graph Builder

You can create the same contour plot shown in Figure 10.2 using Graph Builder.

1. Select **Help > Sample Data Library** and open Little Pond.jmp.
2. Select **Graph** > **Graph Builder**.
3. Click and drag the X coordinate to the **X** zone.
4. Click and drag the Y coordinate to the **Y** zone.
5. Click and drag depth, Z, to the **Color** zone.
6. Click the Contour icon.

Figure 10.3 Contour Plot in Graph Builder



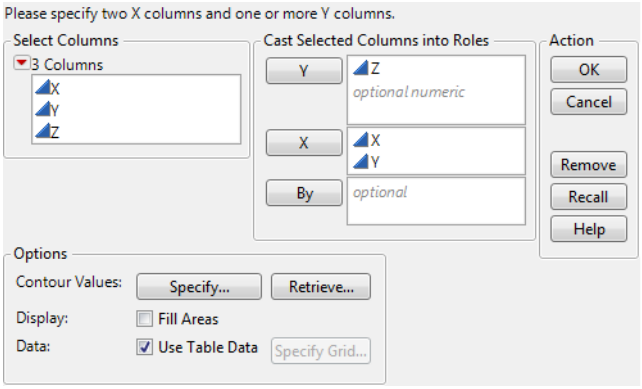
Notice the plot produced from the Contour platform looks slightly different than the plot produced using Graph Builder.

Launch the Contour Plot Platform

Launch the Contour Plot platform by selecting **Graph > Contour Plot**.

By default, the contour levels used in the plot are values computed from the data. You can specify your own number of levels and level increments in the Launch window before you create the plot. You can also do so in the red triangle menu for Contour Plot after you create the plot. You can use a column formula to compute the contour variable values.

Figure 10.4 The Contour Plot Launch Window



Y Columns assigned to the **Y** role are used as variables to determine the contours of the plot. You must specify at least one, and you can specify more than one.

You can also assign a column with a formula to this role. If you do so, the formula should be a function of exactly two variables. Those variables should be the *x* variables entered in the Launch window.

X Columns assigned to the **X** role are used as the variables for the *x*- and *y*-axes. You must specify exactly two columns for **X**.

By This option produces a separate graph for each level of the By variable. If two By variables are assigned, a separate graph for each possible combination of the levels of both By variables is produced.

Options

Contour Values Specify your own number of levels and level increments. See "Contour Specification".

Fill Areas Fill the areas between contour lines using the contour line colors.

Use Table Data and Specify Grid Most often, you construct a contour plot for a table of recorded response values. In that case, **Use Table Data** is selected and the **Specify Grid** button is unavailable.

However, if a column has a formula and you specify that column as the response (**Y**), the **Specify Grid** button becomes available. When you click **Specify Grid**, you can define the contour grid in any way, regardless of the rows in the existing data table. This feature is also available with table templates that have one or more columns defined by formulas but no rows. See "Use Formulas for Specifying Contours".

After you click **OK**, the Contour plot appears.

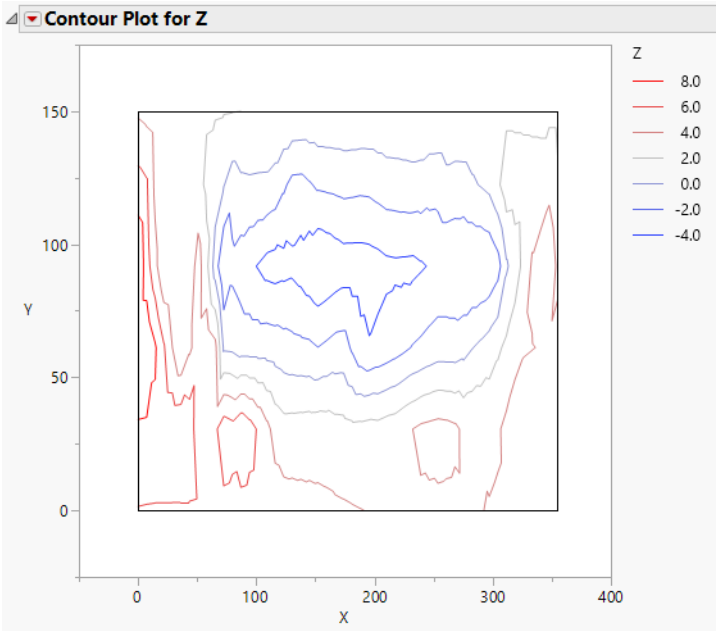
The Contour Plot

Follow the instructions in "Example of a Contour Plot" to produce the plot shown in Figure 10.5.

The legend for the plot shows individual markers and colors for the *Y* variable. Replace variables in the plot by dragging and dropping a variable, in one of two ways: swap existing variables by dragging and dropping a variable from one axis to the other axis; or, click a variable in the Columns panel of the associated data table and drag it onto an axis.

For information about additional options for the report, see "Contour Plot Platform Options".

Figure 10.5 The Contour Plot Report



Contour Plot Platform Options

Using the options in the red triangle menu next to Contour Plot, you can tailor the appearance of your contour plot and save information about its construction.

Show Data Points Shows or hides (*x*, *y*) points. The points are hidden by default.

Show Missing Data Points Shows or hides points with missing *y* values. Available only if **Show Data Points** is selected.

Show Contours Shows or hides the contour lines or fills. The contour lines are shown by default.

Show Boundary Shows or hides the boundary of the total contour area. The boundary is shown by default.

Show Control Panel Shows or hides the **Alpha** slider that allows you to change the **Alpha** shapes filter.

Transform If the contour plot includes a **Color** role, the **Transform** option is enabled. See "Additional Example of Contour Plots".

None The triangulation is computed without any scaling to coordinates using Delaunay triangulation. Delaunay triangles are computed to maximize the minimum angle of the triangles in the triangulation. This value is selected by default.

Range Normalized The X1/X2 values are both scaled to [0,1] prior to computing the triangulation. If the X1/X2 limits are different, then this is a non-uniform scale. This option may be more desirable in cases where the X1/X2 units are very different.

Fill Areas Fills the areas between the contours with a solid color. It is the same option that is available in the Launch window. If you leave it deselected in the Launch window, you can see the line contours before filling the areas. See "Fill Areas".

Label Contours Shows or hides the label (z-value) of the contour lines.

Fit to Window Specifies the stretching behavior of the report.

Change Contours Set your own number of levels and level increments. See "Contour Specification".

Save This menu has options to save information about contours, triangulation, and grid coordinates to data tables. See "Contour Plot Save Options".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

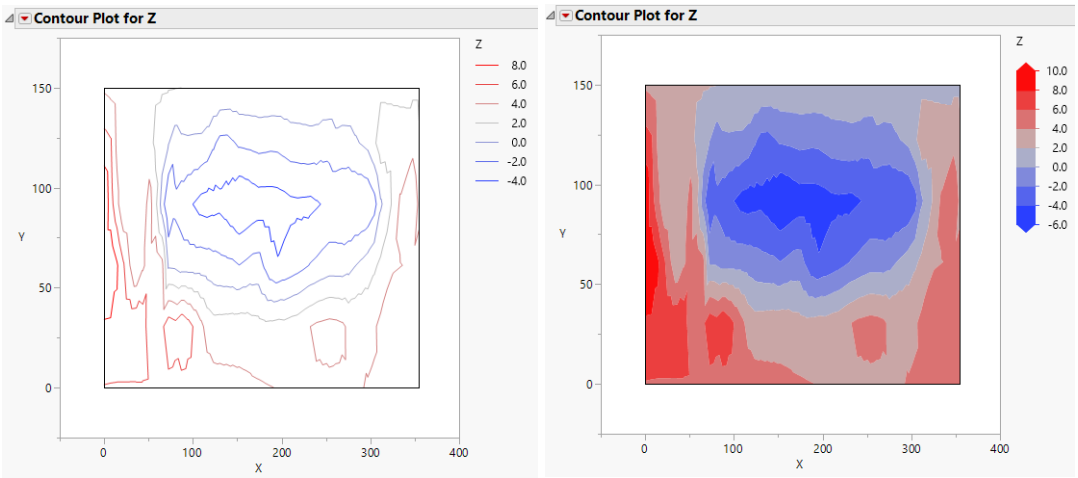
Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Fill Areas

If you select **Fill Areas**, the areas between contour lines are filled with the contour line colors. This option is available in the Launch window and in the red triangle menu for Contour Plot. Figure 10.6 shows a plot with contour lines on the left and a plot with the contour areas filled on the right.

Figure 10.6 Comparison of Contour Lines and Area Fills

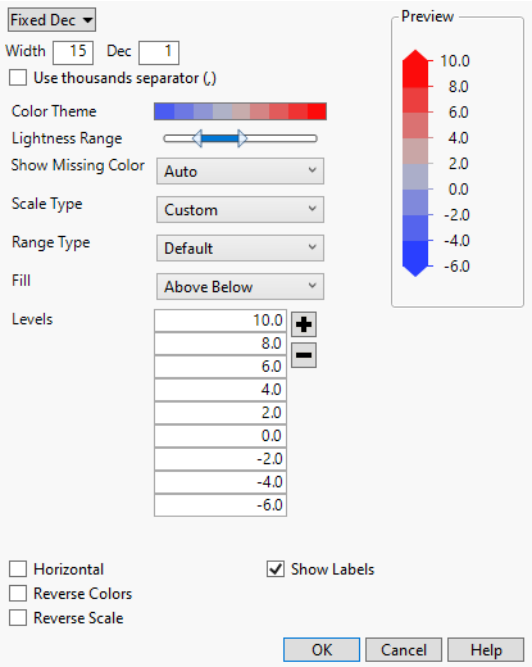


Areas are filled from low to high values on a color gradient. An additional color is added in the filled contour plot for the level above the last, and highest, contour line.

Contour Specification

If you do not select options in the Launch window, the default plot spaces the contour levels equally within the range of the *Y* variable. You can specify a color theme by selecting a Continuous Color Theme in **File > Preferences > Graphs**. You can customize colors for individual contours by right-clicking on the contour color in the Contour Plot legend. You can customize the entire contour color gradient by right-clicking on the Contour Plot legend and selecting Gradient to open the Gradient Settings.

Figure 10.7 Gradient Settings Window



To format the labels in the legend, use the format menu, Width and Dec boxes, and the Use thousands separator (,) check box in the top left of the window.

Color Theme Changes the color theme or defines a custom color theme. For more information about color options, see *Using JMP*.

Lightness Range Sets the range of intensities for the gradient coloring.

Number of Labels (Not available if the Scale Type is Custom.) Specifies the number of labels for your legend. The value of zero provides the default number of labels.

Show Missing Color Specifies whether to show missing color in the legend. Auto shows the color only when there are missing values, Off never shows the color, On always shows the color.

Scale Type Sets the scale for the gradient coloring. A description of the scale type is found below the Maximum value setting.

Linear The scale is piecewise linear between the Minimum and Center values and between the Center and Maximum values. This is the default scaling.

Quantile The scale is piecewise linear for quantiles of the gradient variable.

Standard Deviation The gradient variable range is divided into standard deviation offsets from the mean. The scale is piecewise linear between these offset settings. The number of offsets is determined by the Number of Labels.

Log The scale is linear for the logarithm of the gradient variable.

Log Offset The scale is linear for the offset of the base 10 logarithm of the gradient variable.

Custom The scale labels are specified by the user in the Levels setting.

Range Type Sets the range of the legend values.

Default Unless you specify values for the Minimum and Maximum, the minimum and maximum values include the range of the data and are chosen to be appropriately rounded values.

Exact Data Range The minimum and maximum values are the exact minimum and maximum values for the data, or they are the values that you specify as Minimum and Maximum.

Middle 90% The minimum and maximum values are the 5^{th} and 95^{th} quantiles, or they are the values that you specify as Minimum and Maximum. This option is resistant to outliers.

Fill Determines how the endpoints of the range are specified in the contours.

Between The contours stop at the endpoint and any data beyond the endpoint is not shown.

Above Values above the range are included in the top contour level.

Below Values below the range are included in the bottom contour level.

Above Below Values at either end are included in their respective contour levels.

Levels (Available only when the Custom Scale Type is specified.) Enables you to specify the number of labels and the label values for the legend.

Horizontal Sets the orientation of the legend to horizontal.

Reverse Colors Reverses the colors in the color scheme.

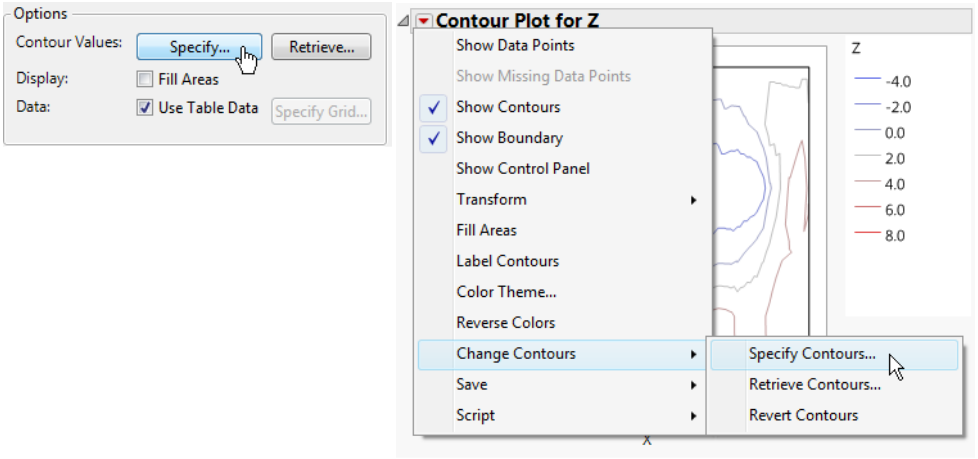
Reverse Scale Reverses the scale for the color theme.

Show Labels Shows or hides labels for the legend.

Specify

You can specify contour levels either in the Launch window (the **Specify** button) or in the report window from the red triangle menu for Contour Plot (the **Specify Contours** option). Selecting this option displays the Contour Specification window.

Figure 10.8 Example of Contour Specification: Launch Window (on the left) and Menu (on the right)

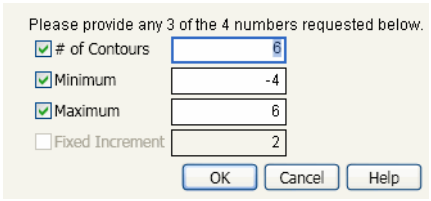


You can use the Contour Specification window to do the following:

- change the number of contours
- specify minimum and maximum values to define the range of the response to be used in the plot
- change the increment between contour values

You supply any three of the four values, and the remaining value is computed for you. Click the check box to deselect one of the numbers and automatically select the remaining check box.

Figure 10.9 The Contour Specification Window



Colors are automatically assigned and are determined by the number of levels in the plot. After the plot appears, you can right-click (press Control and click on macOS) on any contour in the plot legend and choose from the JMP color palette to change that contour color.

Retrieve

This option is both in the Launch window (the **Retrieve** button) and on the red triangle menu for Contour Plot (the **Retrieve Contours** option).

Note: Neither the button nor the menu option are active unless there is an open data table *in addition to* the table that has the contour plotting values. When you click **Retrieve** or select **Retrieve Contours**, a window with a list of open data tables appears.

Using this option, you can retrieve the following from an open JMP data table:

- the number of contours
- an exact value for each level
- a color for each level

From the list of open data tables, select the data table that contains the contour levels.

For level value specification, the Contour Plot platform looks for a numeric column with the same name as the response column that you specified in the Launch window. The number of rows in the data table defines the number of levels.

If there is a row state column with color information, those colors are used for the contour levels. Otherwise, the default platform colors are used.

Revert Contours

This option appears only on the red triangle menu for Contour Plot.

If you have specified your own contours, selecting this option reverts your Contour Plot back to the default contours.

Contour Plot Save Options

This menu has options to save information about contours, triangulation, and grid coordinates to data tables.

Save Contours Creates a new JMP data table with columns for the following:

- the x- and y-coordinate values generated by the Contour platform for each contour
- the response computed for each coordinate set
- the curve number for each coordinate set

The number of observations in this table depends on the number of contours you specified. You can use the coordinates and response values to look at the data with other JMP platforms. For example, you can use the Scatterplot 3D platform to get a three-dimensional view of the pond.

Generate Grid Displays a window that prompts you for the grid size that you want. When you click **OK**, the Contour platform creates a new JMP data table with the following:

- the number of grid coordinates you requested
- the contour values for the grid points computed from a linear interpolation

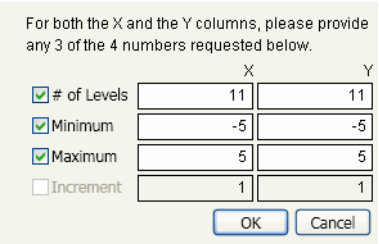
Save Triangulation Creates a new JMP data table that lists coordinates of each triangle used to construct the contours. By default, JMP uses Delaunay triangulation to connect the nearest data points to form triangles. The resulting set of triangles are calculated so that no other data points are inside a triangle's circumscribed circle, that is, the circle that passes through the three vertices of the triangle. To change the triangulation to a normalized scale, select **Transform > Range Normalized**.

Use Formulas for Specifying Contours

Most often you construct a contour plot for a table of recorded response values such as the Little Pond data table. In that case, in the launch window, **Use Table Data** is checked and the **Specify Grid** button is unavailable. However, if a column has a formula and you specify that column as the response (**Y**), the **Specify Grid** button becomes active.

When you click **Specify Grid**, the window shown in Figure 10.10 appears.

Figure 10.10 Example of the Contour Specification for Formula Column



You can complete the Specify Grid window and define the contour grid in any way, regardless of the rows in the existing data table. This feature is also available with table templates that have one or more columns defined by formulas but no rows.

Additional Example of Contour Plots

This example illustrates how to create a triangulation data table, to transform the triangulation to use Delaunay triangles, and to filter Alpha shapes of the triangles.

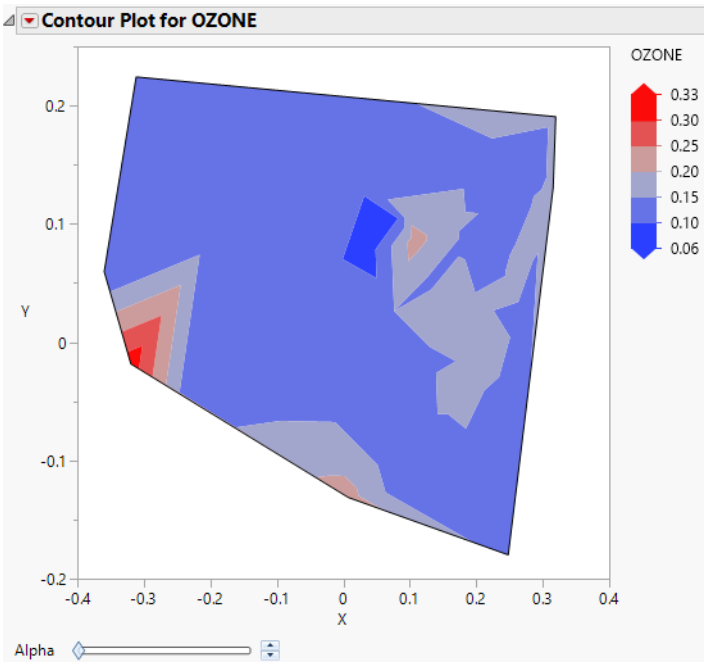
1. Select **Help > Sample Data Library** and open Cities.jmp.
2. Select **Graph > Contour Plot**.
3. Select OZONE and click **Y**.
4. Select X and Y and click **X**.
5. Select to **Fill Areas**.
6. Click **OK**.
7. Click the red triangle next to Contour Plot for OZONE and select the following options:
   a. **Transform** > **Range Normalized**

      Instead of Delaunay triangulation, this changes the method for calculating the triangulations to a normalized scale ([0,1]) in both X and Y.

   b. Show Control Panel

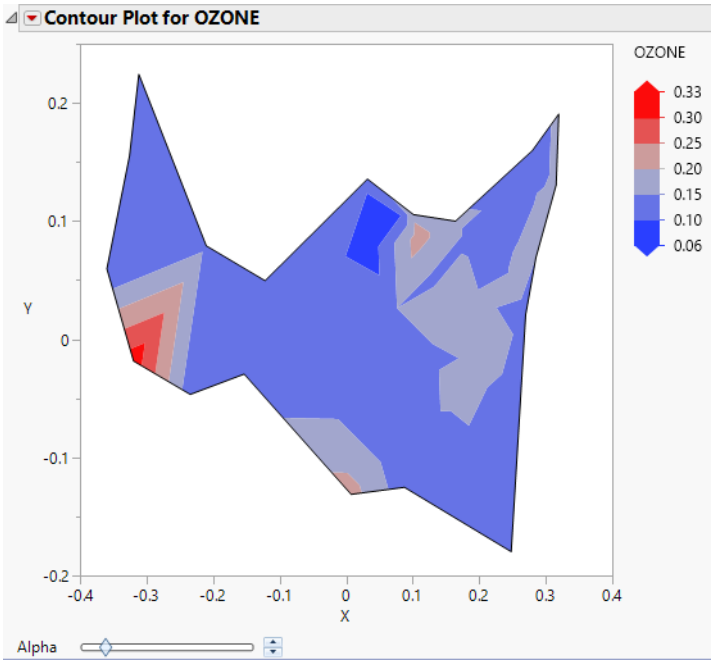
      The **Alpha** slider appears.

Figure 10.11 Contour Plot for OZONE



8. Click and move the **Alpha** slider to the right.

Figure 10.12 Alpha Shapes Filter



Using the Alpha slider filters out the larger Delaunay triangulation areas.

Chapter 11

Ternary Plots

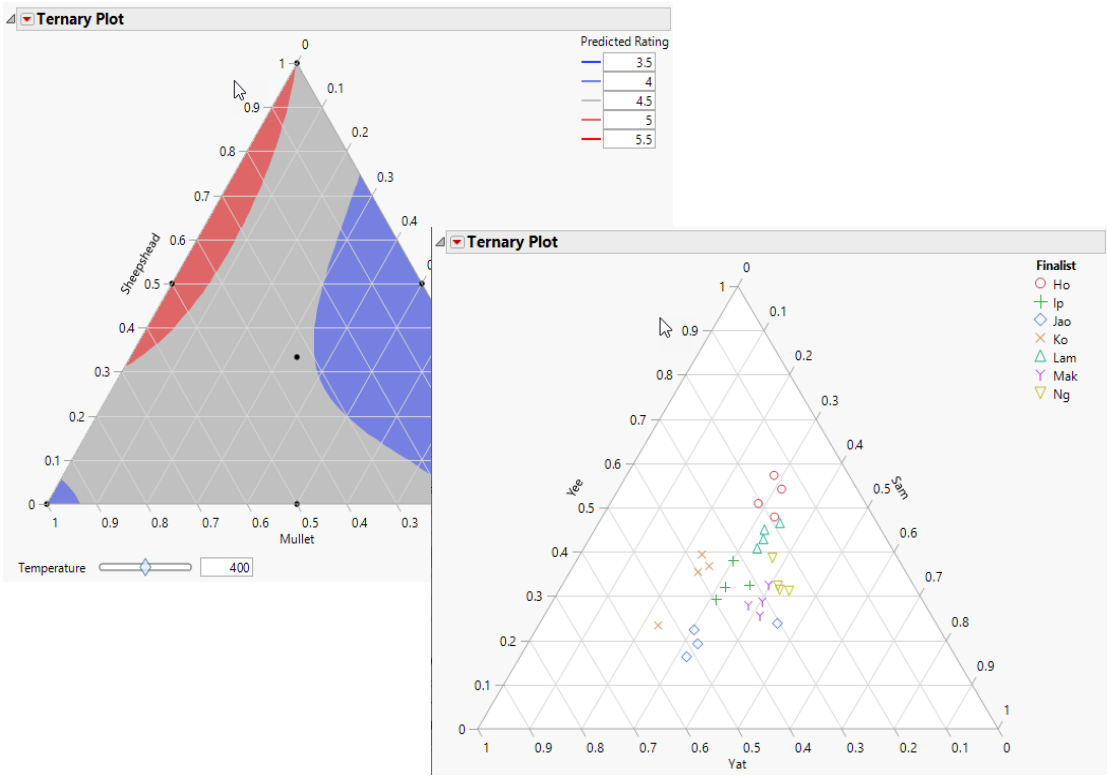
View Plots for Compositional or Mixture Data

The Ternary Plot command in the Graph menu produces a three-axis plot.

Ternary plots are a way of displaying the distribution and variability of three-part compositional data. (For example, the proportion of sand, silt, and clay in soil or the proportion of three chemical agents in a trial drug.) You can use data expressed in proportions or use absolute measures.

The ternary display is a triangle with sides scaled from 0 to 1. Each side represents one of the three components. A point is plotted so that a line drawn perpendicular from the point to each leg of the triangle intersect at the component values of the point.

Figure 11.1 Examples of Ternary Plots



Contents

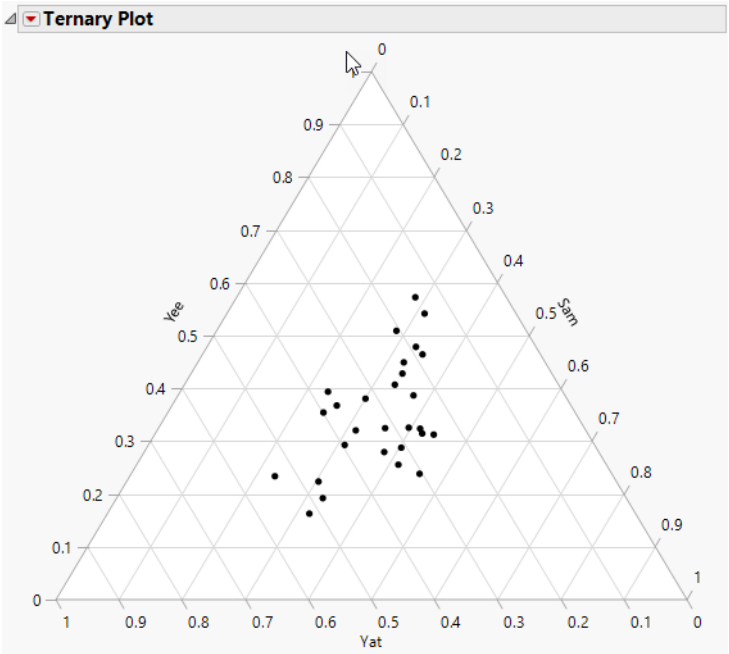
Example of a Ternary Plot 245
Launch the Ternary Plot Platform 247
The Ternary Plot 248
Mixtures and Constraints 249
Ternary Plot Platform Options 249
Additional Examples of the Ternary Plot Platform 250
Example Using Mixture Constraints 250
Example Using a Contour Function 251

Example of a Ternary Plot

This example uses the Pogo Jumps.jmp sample data table. The data, adapted from Aitchison (1986), show measurements for pogo jumps of seven finalists in the 1985 Hong Kong Pogo-Jump Championship. A single pogo jump is the total jump distance in three consecutive bounces, referred to as *yat*, *yee*, and *sam*.

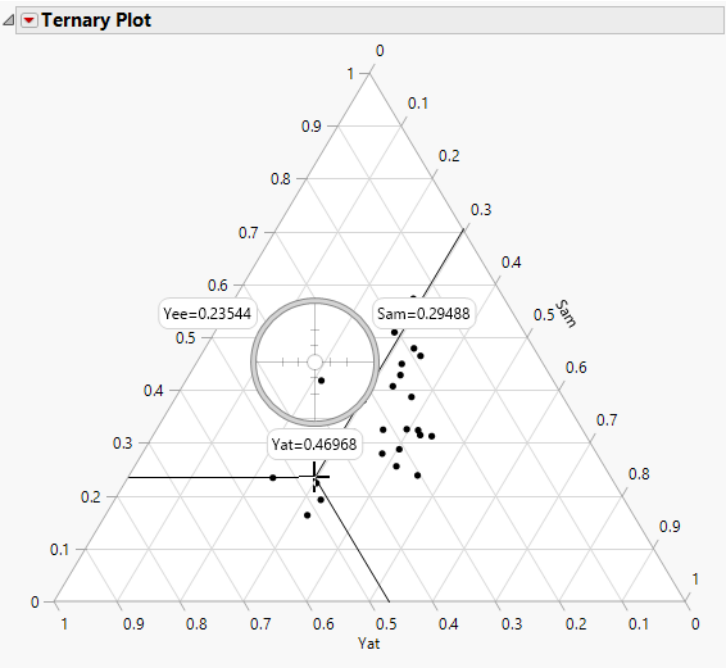
1. Select **Help > Sample Data Library** and open Pogo Jumps.jmp.
2. From the **Graph** menu, select **Ternary Plot**.
3. Select Yat, Yee, and Sam and click **X, Plotting**.
4. Click **OK**.

Figure 11.2 Example of a Ternary Plot



Use the crosshairs tool to determine exact coordinates of points within the plot.

Figure 11.3 Using the Crosshairs Tool

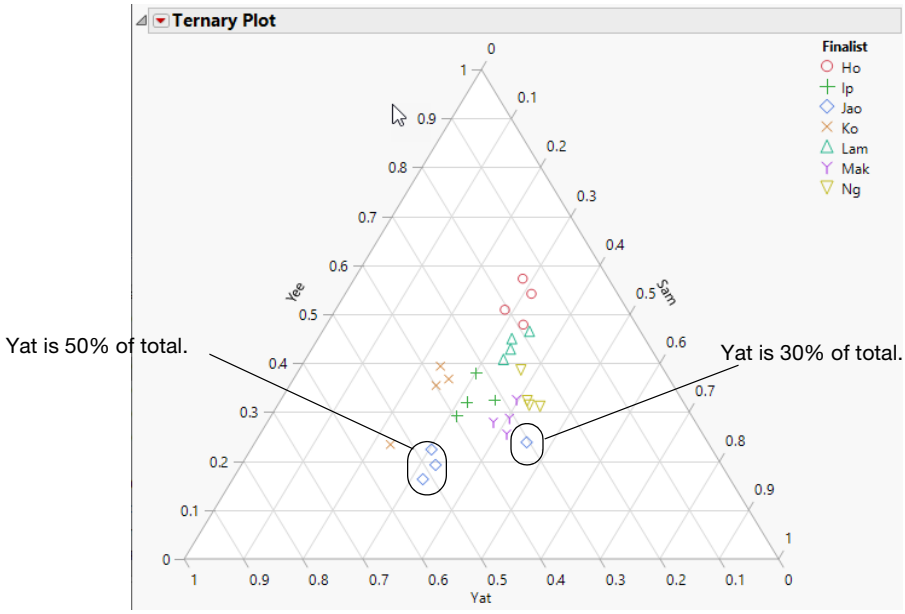


To get a better idea of how the three bounces contribute to total distance, assign each contestant's points a different color and marker.

1. Right-click the plot and select **Row Legend**.
2. Select Finalist in the column list box.

   Colors should be automatically set to **JMP Default**.
3. Select **Standard** from the **Markers** menu.
4. Click **OK**.

Figure 11.4 Pogo Data Colored by Finalist

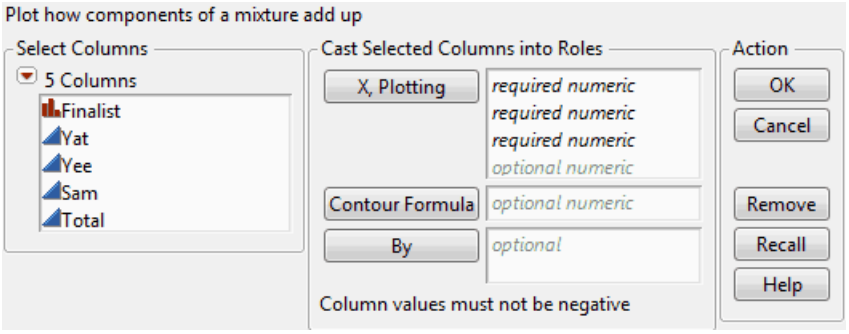


Note that most of the finalists are consistent in the composition of total distance. However, two finalists, Jao and Ko, both have one jump that is not consistent with their other jumps. For example, for three of Jao's jumps, the Yat composed about 50% of the total distance, but for the other jump, the Yat composed only 30% of the total distance. That jump is not consistent with the others. A similar observation can be made about Ko's jumps.

Launch the Ternary Plot Platform

Launch Ternary Plot by selecting **Graph > Ternary Plot**.

Figure 11.5 The Ternary Plot Launch Window



X, Plotting Assign three columns to generate a ternary plot.

If you assign more than 3 variables to the **X, Plotting** role, a matrix of ternary plots is created. A separate variable is assigned to the first two axes of a plot, with the third axis being the sum of the other variables. If necessary, the variables are scaled so they sum to 1.

Contour Formula To plot contours of a response surface, assign a column containing a formula to the **Contour Formula** role. If you have variables in a Contour formula that are not listed as **X, Plotting** variables, JMP appends sliders below the plot so that the values can be interactively adjusted. See "Example Using a Contour Function".

By This option produces a separate graph for each level of the By variable.

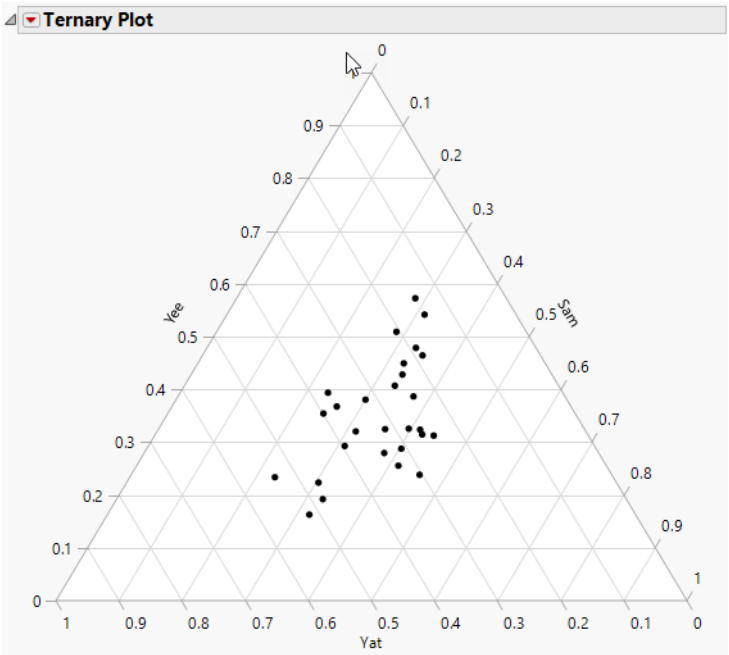
After you click **OK**, the Ternary Plot window appears.

The Ternary Plot

Follow the instructions in "Example of a Ternary Plot" to produce the plot shown in Figure 11.6.

Each of the three sides of a ternary plot represents a proportion of 0%, with the point of the triangle opposite that base representing a proportion of 100%. As a proportion increases in any one sample, the point representing that sample moves from the base to the opposite point of the triangle.

Figure 11.6 The Ternary Plot



Mixtures and Constraints

Ternary Plot uses the Mixture column property to shade the portion of the graph that is out of bounds. The Ternary plot recognizes the mixture sum, mixture bounds, and general linear constraints. For information about setting the Mixture column property in the Column Info window, see *Using JMP*.

Related Information

- "Example Using Mixture Constraints"

Ternary Plot Platform Options

The Ternary Plot red triangle menu contains options to modify the plot.

Note: To view more detailed options, right-click the plot.

Show Constraints Shows or hides the constraints on the plot. The default plot shows the constraints.

Contour Fill Allows filling of contours if a contour formula is specified in the plot. You can select Lines Only, Fill Above, or Fill Below. The default platform shows lines only.

Color Theme Allows you to select a color theme. The default plot shows the Blue to Gray to Red color theme.

Show Points Shows or hides the plotted points. The default plot shows the points.

Fit to Window Specifies whether the graph is resized as you resize the JMP window. The default setting is Auto, which is usually set to On except when the window is shared with other content or if there is a Page variable. To always fit the graph inside the window, keep the setting at On. You can also maintain the aspect ratio. To prevent the graph from resizing, change the setting to Off.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Examples of the Ternary Plot Platform

- "Example Using Mixture Constraints"
- "Example Using a Contour Function"

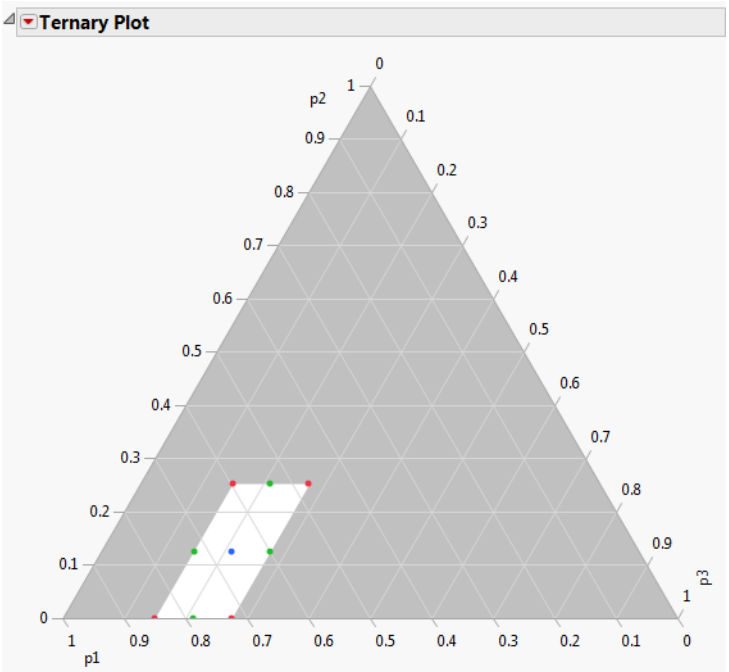
Example Using Mixture Constraints

1. Select **Help > Sample Data Library** and open Plasticizer.jmp.

   The p1, p2, and p3 columns all have Mixture Column Properties defined.
2. From the **Graph** menu, select **Ternary Plot**.

   Notice that p1, p2, and p3 appear next to **X, Plotting**.
3. Click **OK**.

Figure 11.7 Mixture Constraints in a Ternary Plot



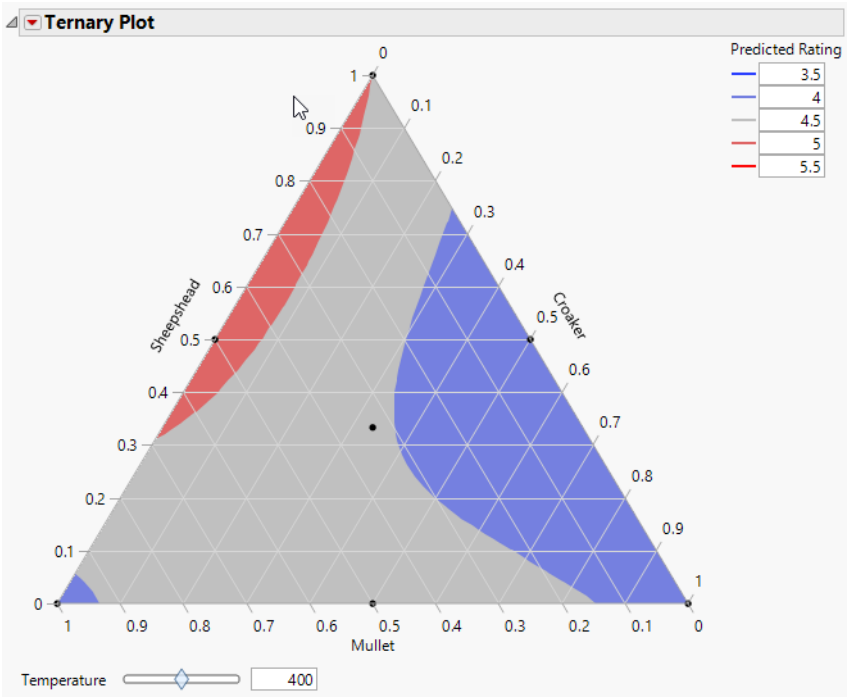
For more information about mixtures, see *Profilers*.

Example Using a Contour Function

The data in Fish Patty.jmp is adapted from Cornell (1990) and comes from an experiment to optimize the texture of fish patties. The columns Mullet, Sheepshead, and Croaker represent what proportion of the patty came from those fish types. The column Temperature represents the oven temperature used to bake the patties. The column Rating is the response and is a measure of texture acceptability, where higher is better. A response surface model was fit to the data and the prediction formula was stored in the column Predicted Rating. See *Profilers*.

1. Select **Help > Sample Data Library** and open Fish Patty.jmp.
2. From the **Graph** menu, select **Ternary Plot**.
3. Select Mullet, Sheepshead, and Croaker and click **X, Plotting**.
4. Select Predicted Rating and click **Contour Formula**.
5. Click **OK**.
6. Click the Ternary Plot red triangle and select **Contour Fill > Fill Above**.

Figure 11.8 Ternary Plot with a Contour Formula



The manufacturer wants the rating to be at least 5. You can drag the slider for Temperature and see the contours for the Predicted Rating change. Each point represents a mixture of the three fish. Any given mixture of fish types receives different ratings according to the temperature at which the patties are baked.

In this example, the red shaded area shows the mixture of fish that results in a rating of 5 to 5.5. Any purple areas show the mixture of fish that results in a rating of 4 and below. At 400 degrees, a mixture of mostly sheepshead and mullet with very little croaker results in a rating of 5 and above.

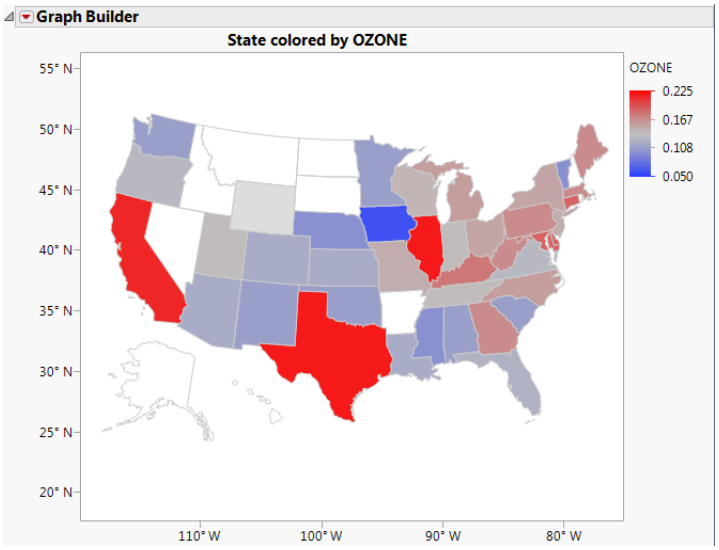
Chapter 12

Maps

Add Maps or Custom Shapes to Enhance Data Visualization

JMP transforms numbers and geographic data into compelling images, and turns simple tables of numbers into captivating pictures that bring the story in your data to life. JMP can help you display your data on geographical maps. Choose from built-in high-quality images. Select Street Map Service or Web Map Service to get custom map images from the Internet. JMP includes shape files for borders or many geographic regions and lets you add your own custom shapes, such as for a manufacturing plant or campus.

Figure 12.1 Example of a Map



Contents

Overview of Mapping 255
Example of Creating a Map in Graph Builder 256
Graph Builder 257
Map Shape 257
Color 259
Size 260
Custom Map Files 263
Background Maps 269
Images in Maps 271
Boundaries 276
Examples of Creating Maps 277
Louisiana Parishes Example 277
Hurricane Tracking Examples 285
Office Temperature Study 290
Boundary Map with Clipped Elements 295

Overview of Mapping

There are two types of map support in JMP: one where a map shows the data (Graph Builder) and one where a map provides context for the data (Background Maps). You can also create your own maps.

Graph Builder

You can interact with Graph Builder to create compelling visualizations of your data. JMP includes graphical support to display analyses using background maps and shape files. You can add color and geographical boundaries to maps through the following zones:

- The **Map Shape** zone assigns geographical boundaries to a map based on variables in the data table. The map shape value determines the x and y axes.

  Boundaries such as U.S. state names, Canadian provinces, and Japanese prefectures are installed with JMP. You can also create your own boundaries (geographical or otherwise) and specify them as a **Map Role** column property in the data table.
- The **Color** zone applies color based on a variable to geographical shapes.
- The **Size** element scales map shapes according to the size variable, minimizing distortion.

Background Maps

You can add background maps to any JMP graph through the Set Background Map window. You can use built-in background maps or connect to a Web Map Service (WMS) to display specialty maps like satellite images, radar images, or roadways. Right-click in a graph and select **Graph > Background Map** to choose from the following images and boundaries:

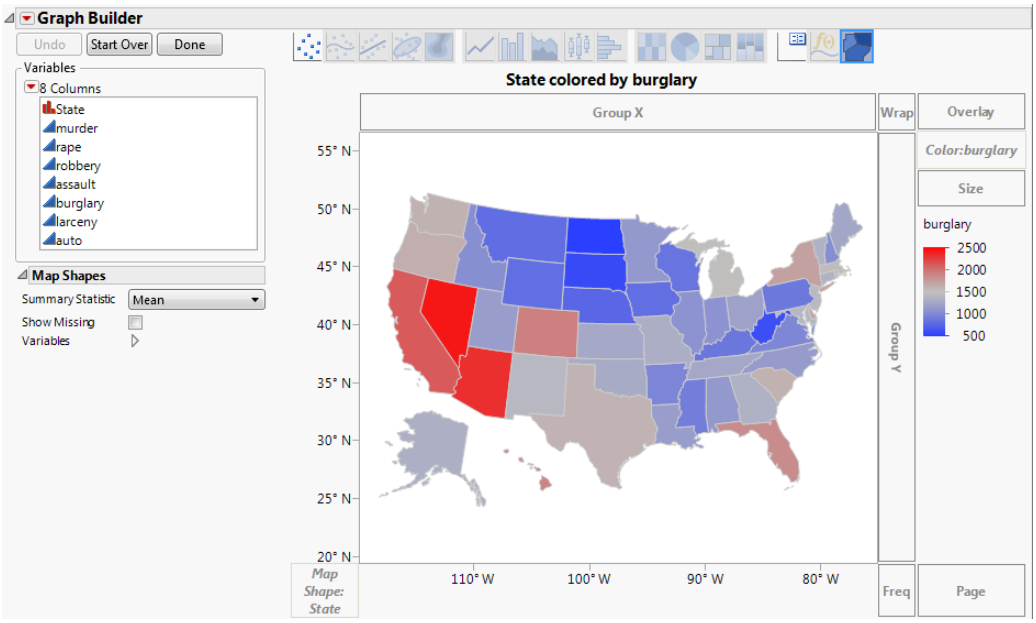
- **Simple Earth** and **Detailed Earth** maps are installed with JMP.
- **NASA server** provides maps using a WMS to show their most up-to-date maps.
- **Street Map Service** provides street maps. The OpenStreetMap and Open Database License links provide further information on the Street Map Service.
- **Web Map Service** lets you enter the URL for a website that provides maps using the WMS protocol. You can also specify the map layer.
- Boundaries for various regions.

Example of Creating a Map in Graph Builder

This example uses the Crime.jmp sample data table, which contains data on crime rates for each US state.

1. Select **Help > Sample Data Library** and open Crime.jmp.
2. Select **Graph > Graph Builder**.
3. Drag and drop State into the **Map Shape** zone.
4. Drag and drop Burglary into the **Color** zone.

Figure 12.2 Example of Burglary by State



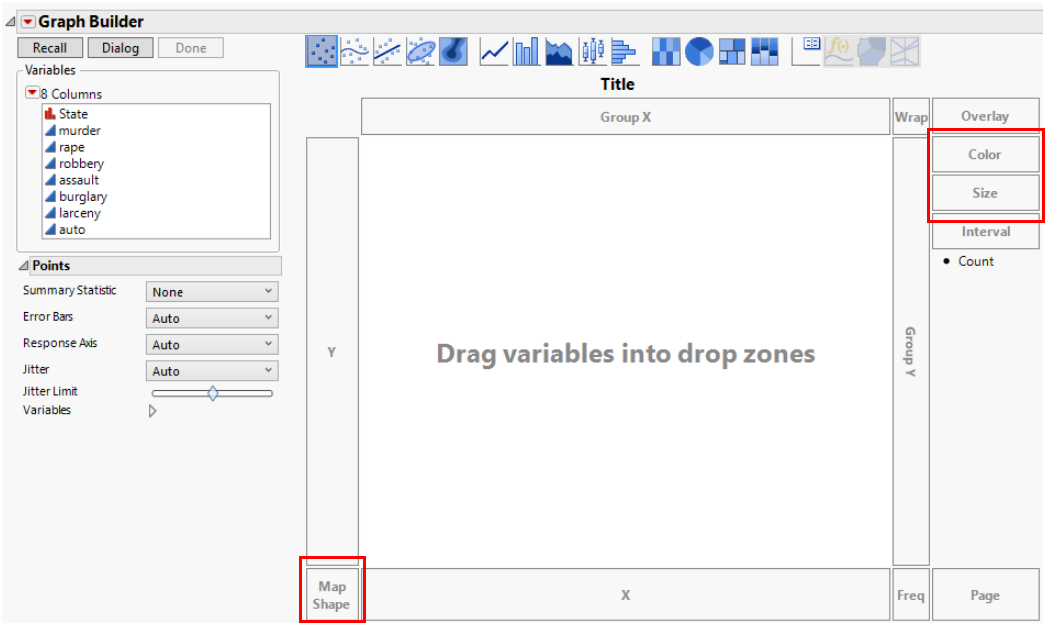
Note the following:

- The latitude and longitude appear on the Y and X axes.
- The legend shows the colors that correspond to the burglary rates. Since Burglary is a continuous variable, the colors are based on a continuous color theme. Figure 12.2 uses the JMP default continuous theme. You can change the theme under File > Preferences > Graphs.
- The map is projected so that relative areas are not distorted (the 49th parallel across the top of the US is not a straight line).

Graph Builder

Open a data table that contains geographic data. Launch Graph Builder by selecting **Graph** > **Graph Builder**. The primary element in the Graph Builder window is the graph area. The graph area contains drop zones (Map Shape, Color and Size), and you can drag and drop variables into the zones. From here you can map shapes for data tables that include place names.

Figure 12.3 The Graph Builder Window



Map Shape

When a column contains the names of geographical regions (such as countries, regions, states, provinces, counties), you can assign the column to the **Map Shape** zone. When a variable is dropped in **Map Shape**, Graph Builder looks for map shapes that correspond to the values of the variable and draws the corresponding map. The variable can have a column property that tells JMP where to find the map data. If not, JMP looks through all known map files. If you have a variable in the Map Shape zone, the X and Y zones disappear. The Map Shape zone is positional and influences the types of graph elements that are available.

Figure 12.4 Example of Cities.jmp After Dragging State to Map Shape

For each map there are two .jmp files; one for the name data (one row per entity) and one for coordinate data (many rows per entity). They are paired via a naming convention; xxx-Name.jmp and xxx-XY.jmp, where "xxx" is some common prefix.

Some examples of sample files that are shipped with JMP are:

- World-Name.jmp
- World-XY.jmp
- US-State-Name.jmp
- US-State-XY.jmp

Map Name Files

Each xxx-Name.jmp can contain any number of shape name columns, which are identified with a column property. Multiple name columns support localizations and alternate names styles (such as abbreviations), but a given graph usage uses only one column of names. The first column of the Name file contains unique Shape ID numbers in ascending order. JMP creates this columns for you. The column values match those in the -XY.jmp map data table Shape ID column.

Note: The Shape ID column can also be named Shape.

Figure 12.5 Example of US-State-Name.jmp

	Shape ID	Name	State FIPS	USPS Code	Abbr	Max Grow ...	Max Grow X Offset	Max Grow Y Offset
1	1	Alabama	01	AL	Ala.	1	0	0
2	2	Alaska	02	AK	Alaska	1	0	0
3	3	Arizona	04	AZ	Ariz.	1	0	0
4	4	Arkansas	05	AR	Ark.	1	0	0
5	5	California	06	CA	Calif.	3	-1.7	-3
6	6	Colorado	08	CO	Colo.	1	0	0
7	7	Connecticut	09	CT	Conn.	4	0.8	-1.1
8	8	Delaware	10	DE	Del.	3	0.5	-0.3
9	9	District of Columbia|Washi...	11	DC	D.C.	70	2	-2.5
10	10	Florida	12	FL	Fla.	4	2	-2
11	11	Georgia	13	GA	Ga.	1	0	0

Map XY Files

Each xxx-XY.jmp file has four columns. Each row is a coordinate in some shape. Each shape is made of one or more parts. Each part is a closed polygon. The first column is the same Shape ID as in the xxx-Name file. The second column is the Part ID. The next two columns are X and Y.

Figure 12.6 Example of US-State-XY.jmp

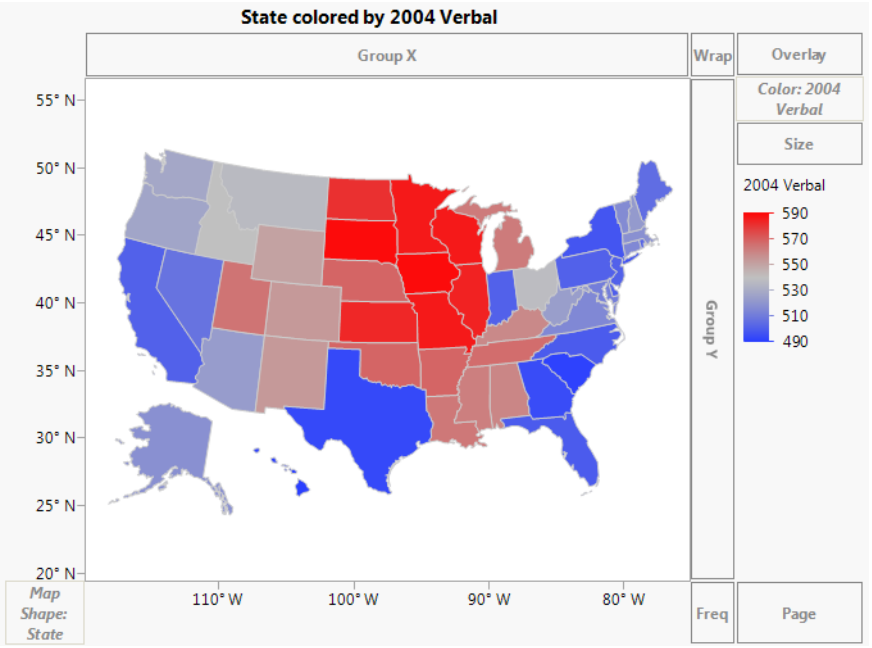
	Shape ID	Part ID	X	Y
1	1	1	85.6052° W	34.9847° N
2	1	1	85.1844° W	32.8613° N
3	1	1	84.9630° W	32.4242° N
4	1	1	85.0071° W	32.3284° N
5	1	1	84.8909° W	32.2615° N
6	1	1	85.0611° W	32.1341° N
7	1	1	85.1407° W	31.8575° N
8	1	1	85.0413° W	31.5410° N
9	1	1	85.1075° W	31.1865° N
10	1	1	85.0024° W	31.0007° N
11	1	1	87.5989° W	30.9975° N

Color

The Graph Builder platform lets you adds color to create choropleth maps. A choropleth map shows statistical differences in a geographic area while maintaining the proportion of the statistical variable.

Drag a column containing geographic place-names, like countries, regions, states, or provinces, into the **Map Shape** zone and create a map. Then drag a column to the **Color** zone to color the map by that column. The categorical or continuous color theme selected in your Preferences is applied to each shape.

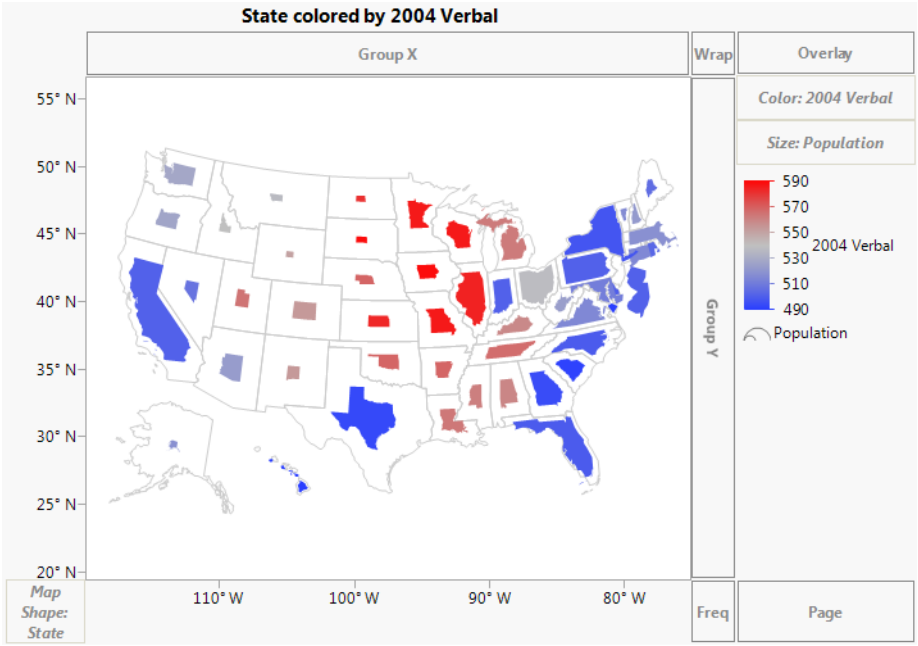
Figure 12.7 Example of SAT.jmp After Dragging 2004 Verbal to Color



Size

Use the **Size** element to scale map shapes according to the size variable, minimizing distortion.

Figure 12.8 Example of SAT.jmp After Dragging Population to Size



Customizing Graphs

To change colors and transparency for a map, right-click the color bar in the legend. The right-click options vary, depending on whether the color variable is continuous or categorical (nominal or ordinal). However, for both types of variables, you can change the transparency.

To change the transparency of a graph:

1. Right-click the color of the variable level on the color bar that you want to change and select **Transparency**.
2. Specify the transparency between 0 (clear) and 1 (opaque).
3. Click **OK**.

You can also change the transparency of images (for example, Simple Earth and Detailed Earth). To set the transparency, right-click over the graph and select **Customize...**. This brings up the Customize Graph window, where you can select the Background Map and assign a value for transparency. A valid value for transparency goes from 0.0 (completely transparent) to 1.0 (completely opaque).

Categorical (nominal or ordinal) variables use a singular coloring system, where each level of the variable is colored differently.

To change the color of one of the variable levels:

1. Right-click the color of the variable level that you want to change and select **Fill Color**.
2. Select the new color.

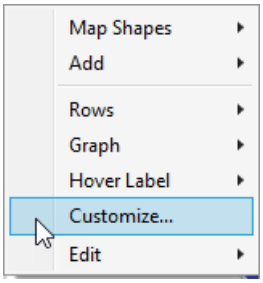
Continuous variables use a color gradient.

To change the color theme:

1. Right-click the color bar and select **Gradient**.
2. In the Gradient Settings window, select a different **Color Theme**.

Graphs consist of markers, lines, text, and other graphical elements that you can customize. If you right-click an image, there are several options for working with the graph. The options differ based on what you clicked. See "Gradient Settings" and *Using JMP*. Below are a few options.

Figure 12.9 Right-click Menu for Graphics



- Map Shapes:
  - Change To - Provides options for changing the map into another graph type.
  - Summary Statistics - Provides options for changing the statistic being plotted.
  - Show Missing Shapes - Shows or hides missing data from a map (turned off by default). Missing Shape means that there are some shape names that exist in the map file but not in the data table for analysis.
  - Remove - Removes the current map shape.
- Customize - You can change the properties of the graph such as contents, grid lines, or reference lines. The graphical elements that you can customize differ for each graph. Select **Background Map** to change the transparency of a background map or **Map Shape** to change the line color, line style and width, fill color, missing shape fill or missing value fill. Click **Help** in the Customize Graph window for a more detailed explanation of the customize options.

To change the shape file:

1. Right-click the Map Shapes zone and select **Set Shape File**.
2. Navigate to the shape file that contains the name data.
3. Click **Open**.
4. (Optional) To revert the changes, right-click the Map Shapes zone and select **Reset Shape File**.

Custom Map Files

You can create your own map files by following the same pattern as the built-in files. To add your own map files, you need two things: a series of XY coordinates for the vertices of the polygons that describe the shape, and a set of names for each polygon. Data and shape attributes are required to map custom shapes so that you can add your own shapes to JMP. There are two common sources for data like this: Esri shapefiles and SAS/GRAPH map data sets.

In order for JMP to automatically find your files, place them in the following directory:

- On Windows: C:/Users/<user name>/AppData/Roaming/SAS/JMP/Maps
- On macOS: /Users/<user name>/Library/Application Support/JMP/Maps

Note: On Windows, in JMP Pro, the "JMP" folder is named "JMPPro". Or, you can link the map files to your data files explicitly with the **Map Role** column property.

Note the following when creating map files:

- Each set of map files that you create must contain a -Name file and a -XY file.
- The first column in both files must be the ascending, numeric Shape ID variable. The -Name file can contain any other columns. The shapes are built by rows. The XY coordinates have to go around the shape rather than just define the convex hull of the shape.
- For the **Map Role** column property, columns that are marked with the **Shape Name Definition** are searched for shape identification and must contain unique values.
- If you import an Esri SHP file, it is opened in the correct format. -Name files commonly have a .dbf extension. See "Esri Shapefiles".
- SAS/GRAPH software includes a number of map data sets that can be used with JMP. See "SAS/GRAPH Map Data Sets".

You might want to create choropleth maps of other non-geographic regions (for example, a floor of an office building). Simply, add the two shape files for your non-geographic space. If you do not have XY coordinates, but you do have a graphic image of the space, you can use the Custom Map Creator add-in for JMP. With this add-in, you can trace the outlines of the space and JMP creates the -XY and -Name files for you. You can download this add-in from the JMP File Exchange page.

Map Role

You can specify the attributes and properties of a column in a data table within the Column Info window in Column Properties. The **Map Role** property is set for a column like other column properties in the Column Info window.

If you have created your own data table that contains boundary data (such as countries, regions, states, provinces, or counties) and you want to see a corresponding map in Graph Builder, use the **Map Role** property within Column Properties. Each pair of map files that you create must contain a -Name file and a -XY file.

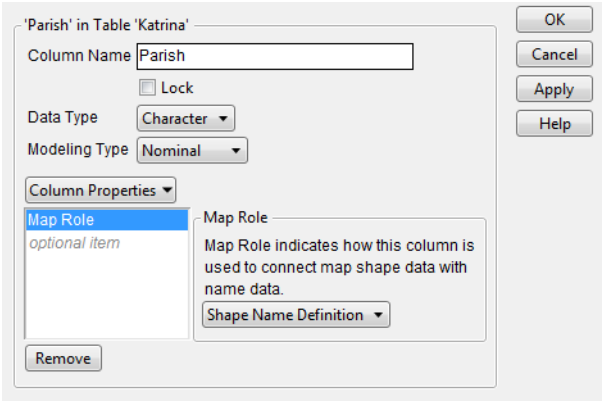
Note the following:

- If the custom boundary files reside in the default custom maps directory, then you need to specify only the **Map Role** property in the -Name file.
- If the custom boundary files reside in an alternate location, specify the **Map Role** property in the -Name file and in the data table that you are analyzing.
- The columns that contain the **Map Role** property must contain the same boundary names, but the column names can be different.

To add the Map Role property into the -Name data table:

1. Right-click the column containing the boundaries and select **Column Properties > Map Role**.
2. Select **Shape Name Definition** below **Map Role**.
3. Click **OK**.
4. Save the data table.

Figure 12.10 Shape Name Definition Example



To add the Map Role property into the data table that you are analyzing:

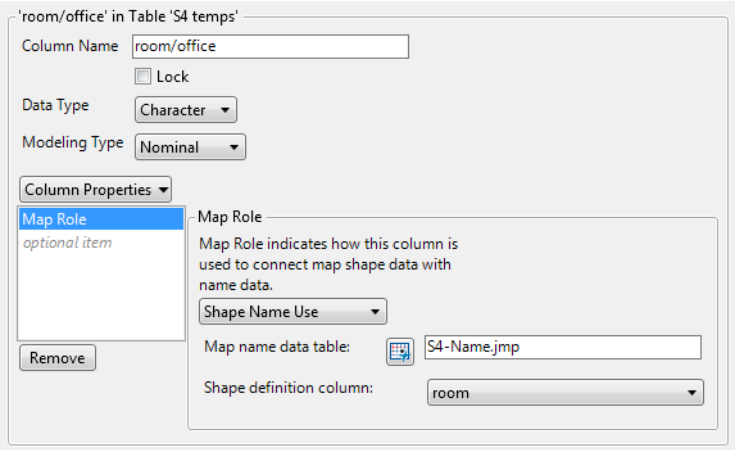
Note: Perform these steps only if your custom boundary files do not reside in the default custom maps directory.

1. Right-click the column containing the boundaries and select **Column Properties > Map Role**.
2. Select **Shape Name Use** below **Map Role**.
3. Next to **Map name data table**, click to browse to a -Name map data table. You can enter the relative or absolute path.

   If the map data table is in the same folder, enter only the filename. Quotation marks are not required when the path contains spaces.
4. From the **Shape definition column** list, select the column in the map data table whose values match those in the selected column.

   Figure 12.11 shows an example of the room/office column in the S4 Temps.jmp sample data table.

Figure 12.11 Shape Definition Column Example



5. Click **OK.**
6. Save the data table.

When you generate a graph in Graph Builder and assign the modified column to the **Map Shape** zone, your boundaries appear on the graph.

For numeric columns, the Format Menu appears in the Column Info window. Specify the format to tell JMP how to display numbers in the column. Latitude and Longitude for geographic maps are located under **Format > Geographic** when customizing axes and axes labels.

Geographic Shows latitude and longitude number formatting for geographic maps. Latitude and longitude options include the following:

- DDD (degrees)
- DMM (degrees and minutes)
- DMS (degrees, minutes, and seconds)

In each format, the last field can have a fraction part. You can specify the direction with either a signed degree field or a direction suffix. To show a signed degree field, such as *-59°00′00″*, deselect **Direction Indicator**. To show the direction suffix, such as *59°00′00″ S*, select **Direction Indicator**.

To use spaces as field separators, deselect **Field Punctuation**. To use degrees, minutes, and seconds symbols, select **Field Punctuation**.

Esri Shapefiles

The Esri shapefile is a vector data format that contains data about geographic features such as terrain and oceans. It is developed and regulated by Esri as a specification for geographic mapping software.

Each shapefile is a set of files with the same name and different extensions.

main file (.shp)

The .shp file contains sequences of points that make up polygons. When opened with JMP, a .shp file is imported as a JMP table.

- The Shape column is added during import to uniquely identify each geographic region. Each coordinate point is in a separate row.
- The Part column to indicate discontiguous regions, and the XY coordinates (in latitude and longitude degrees).

JMP supports two-dimensional .shp files (no elevation information).

dBase Table (.dbf)

You add a Shape ID column to the .dbf table, which maps to the Shape column in the .shp file. Add any number of columns that provide common names or values to refer to specific regions.

To convert an Esri shapefile to a JMP map file:

1. Open the .shp file in JMP.
2. Make sure that the Shape column is the first column in the .shp file. Add formatting and axis settings for the X and Y columns (optional). Graph Builder uses those settings for the X and Y axes.
3. Save the .shp file as a JMP data table to the Maps folder with a name that ends in -XY.jmp.
4. Open the .dbf file.
5. Add a Shape ID column as the first column in the table. This column should be the row numbers from 1 to n, the number of rows in the data table.

   Note: You can use **Cols > New Columns > Initialize Data > Sequence Data**) to fill the column with sequential numbers.

6. Assign the **Map Role** column property to any column that you use for place names in the Shape role of Graph Builder. To do this, right-click at the top of the column and select **Column Properties > Map Role**.
7. Select **Shape Name Definition** from the drop-down box in the property definition.

8. Save the table as a JMP data table with a name that matches the earlier table and that ends in -Name.jmp.

JMP looks for these files in two locations. One location is shared by all users on a machine. This location is:

- Windows: C:/Program Files/SAS/JMP/16/Maps/
- macOS: /Library/Application Support/JMP/16/Maps

The other location is specific for an individual user:

- On Windows: C:/Users/<user name>/AppData/Roaming/SAS/JMP/Maps
- On macOS: /Users/<user name>/Library/Application Support/JMP/Maps

Note: On Windows, in JMP Pro, the JMP folder is named JMPPro.

SAS/GRAPH Map Data Sets

SAS/GRAPH® software includes a number of map data sets that can be converted for use with JMP. The data sets are in the Maps library. The traditional map data sets contain the XY coordinate data and the feature table contains the common place names. You need to convert both of these files to JMP data tables for use with JMP.

Most of the traditional map data sets have unprojected latitude and longitude variables in radians. The data sets can be used with JMP once they have been converted to degrees and the longitude variable has been adjusted for projection. The following is a DATA step that shows the conversion process for the Belize data set.

```
data WORK.BELIZE;
keep id segment x y;
rename segment=Part;
set maps.belize;
if x NE .;
if y NE .;
y=lat*(180/constant('pi'));
x=-long*(180/constant('pi'));
run;
```

You can now import the converted file and save it as Belize-XY.jmp.

The next step is to import the matching feature data set (in this case: MAPS.BELIZE2). After importing the feature data set, move the ID column to the first position in the data table. Then assign the Map Role column property to the columns that you use for place names in the Shape role of Graph Builder. To do this, right-click the top of the column and select **Column Properties** > **Map Role**. Then select **Shape Name Definition** from the drop-down box in the property definition. For MAPS.BELIZE2, use the IDNAME column. Save the feature data table as Belize-Name.jmp.

To convert SAS maps, download the SAS to JMP Map Converter add-in from the JMP File Exchange page. For each map, the add-in reads the data from the two SAS map tables, rearranges and formats the data and then places it into the two JMP map tables.

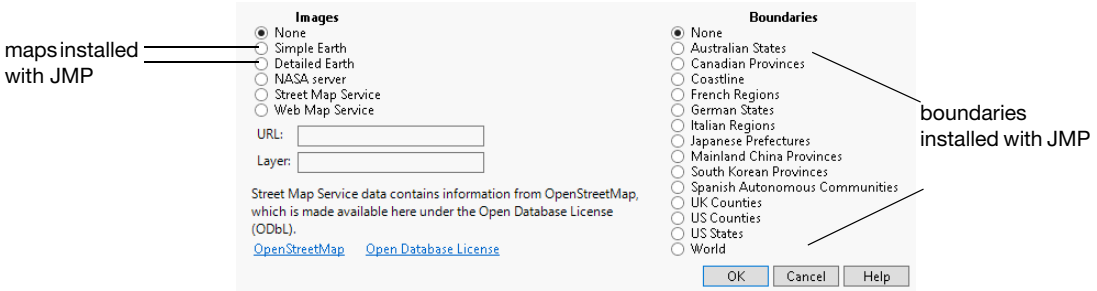
Background Maps

Adding map images and boundaries to graphs provides visual context to geospatial data. Affixing a background map generates an appealing map, providing your data a geographic context and giving you a whole new way to view your data. For example, you can add a map to a graph that displays an image of the U.S. Another option is displaying the boundaries for each state (when data includes the latitudes and longitudes for the U.S.). There are different types of background maps. Some maps are built into JMP and are delivered as part of the JMP install. Other maps are retrieved from an Internet source, and still other maps are user-defined.

The data should have latitudinal and longitudinal coordinates. Otherwise, the map has no meaning in the context of the data. The X and Y axes also have range requirements based on the type of map. These requirements are described in the following sections. Simply plot longitude and latitude on the X and Y axes, and then right-click within the graph and select **Graph** > **Background Map**.

The Background Map window shows two columns of choices: Images and Boundaries. On the left of the window that you can select from two built-in map images, or you can connect to a Web Map Service to retrieve a background image. On the right side of the window, you can select political boundaries for a number of regions.

Figure 12.12 Example of Background Map Options



The following Background Map options are available:

Images

None Removes the background map that you selected in the Images column.

Simple Earth Shows a map of basic terrain. For examples, run the associated scripts in these sample data tables: SAS Offices.jmp and Aircraft Incidents.jmp.

Detailed Earth Shows a high-resolution map with detailed terrain.

NASA Server Shows a map from the NASA server. Requires an Internet connection.

Street Map Service Shows a map with an appropriate amount of detail based on the display's zoom level. This enables you to zoom down to the street level. For example, run the associated script in the sample data table SAS Offices.jmp.

Web Map Service Shows a map from the Uniform Resource Locator (URL) and the layer that you specify. Requires an Internet connection.

Boundaries

None Removes the boundaries that you selected in the Boundaries column.

Boundaries for various regions Shows borders for the map regions, such as Canadian provinces, U.S. counties, U.S. States, and world countries. The list varies based on your location. The maps that you created from Esri shapefiles are also listed here.

Two tools are especially helpful when you are viewing a map:

- The grabber tool ( ) lets you scroll horizontally and vertically through a map.
- The magnifier tool ( ) lets you zoom in and out.

Images in Maps

Every flat map misrepresents the surface of the Earth in some way. Maps cannot match a globe in truly representing the surface of the entire Earth. A map projection is used to portray all or part of the round Earth on a flat surface. This cannot be done without some distortion. Every projection has its own set of advantages and disadvantages. A map can show one or more, but not all, of the following: true direction, distance, area, or shape. JMP uses a couple of projections (Albers Equal Area Conic and Kavrayskiy VII) for its maps. Within Images, you can select from two built-in map images, or you can connect to a Web Map Service to retrieve a background image.

Earth Images Installed with JMP

JMP provides two levels of earth imagery; simple and detailed. Both maps show features such as bodies of water and terrain. However, detailed maps show more precise terrain. And with detailed maps, you can zoom in farther, and the map features remain clear. Image maps are raster images. The maps wrap horizontally, so you continue to see map details as you scroll from left to right. The maps do not wrap vertically. Beyond the -90 and 90 y-axis range, a plain background appears instead of the map.

Figure 12.13 Examples of Simple and Detailed Maps

Simple Earth Image

Detailed Earth Image

As its name suggests, Simple Earth is a relatively unadorned image of the earth's geography. It does not show clouds or arctic ice, and it uses a green and brown color scheme for the land and a constant deep blue for water. Detailed Earth has a softer color scheme than Simple Earth, lighter greens and browns for the land, as well as variation in the blue for the water. Detailed Earth also has a slightly higher resolution than Simple Earth. The higher resolution lets you zoom into a graph further with Detailed Earth than with Simple Earth before the quality of the background image begins to blur.

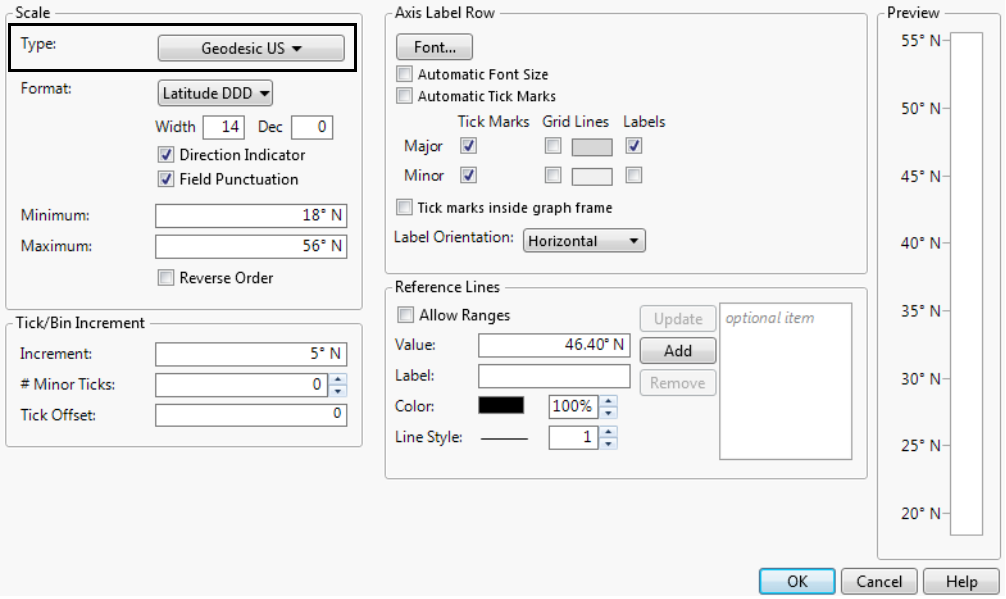
Another feature of Simple Earth and Detailed Earth is the ability to wrap. The Earth is round, and when you cross 180° longitude, the Earth does not end. The longitudinal value continues from -180° and increases. The map wraps continuously in the horizontal direction, much as the Earth does. The background map does not wrap in the vertical direction.

Simple Earth and Detailed Earth both support a geodesic scaling. In Figure 12.13, the Earth appears as a rectangle, where the width is twice as wide as the height. If we were to take this rectangle and roll it up, we would have a cylinder. In reality, we know that the Earth does not form a cylinder, but rather a sphere. You can use a geodesic scaling, which transforms the map to a more realistic representation of the Earth. To use the geodesic scaling, change the type of scale on the axes.

To change the axes scale:

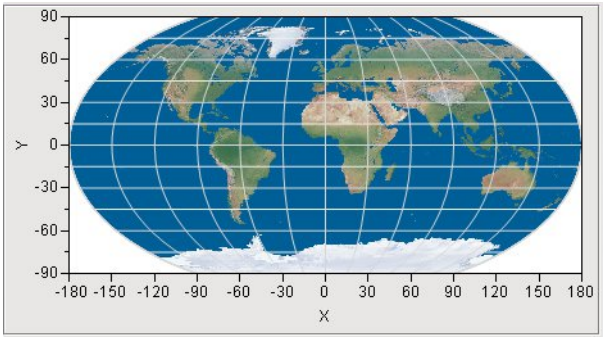
1. Right-click the X or Y axis and then select **Axis Settings.**
2. Change the Scale Type to **Geodesic** or **Geodesic US**.

Figure 12.14 Y Axis Setting Window



Both choices transform the map to a geodesic scaling. Use Geodesic US if you are viewing a map of the continental US and you want Alaska and Hawaii to be included in the map. It is important to note that you must set the scale to geodesic for both axes to get the transformation. You will not see a change in the map after setting only one of the axes. In the following figure, Simple Earth is used as the background map with the axes set to use a geodesic scale. The axes lines are turned on as well. Notice the longitudinal lines are now curved, instead of straight.

Figure 12.15 Simple Earth Background Map - Axes Set to Geodesic Scale

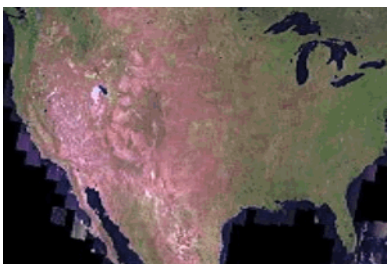


Since Detailed and Simple Earth are built into JMP, these options work anytime, without a network connection. However, these images might not be all that you want, or they might not be detailed at the resolution that you need. If this is the case, and if you have an Internet connection, you can connect to a Web Map Service to retrieve a map image that meets your needs.

Maps from the Internet

The National Aeronautics and Space Administration (NASA) and other organizations provide map image data using a protocol called Web Map Service (WMS). These maps have the advantage of showing the most up-to-date geographical information. However, the display of the maps can be slow depending on the response time of the server, and the sites can change or disappear at any time. An Internet connection is required to access the information.

Figure 12.16 Examples of NASA and WMS Maps

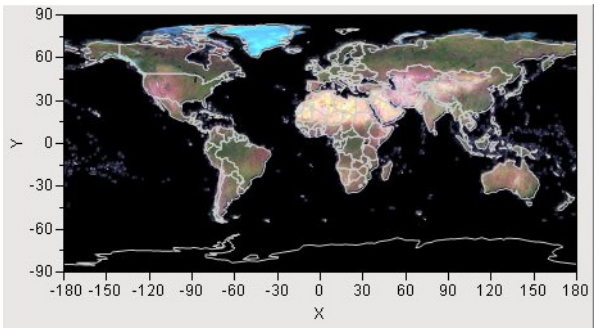


NASA image

WMS image from the MetaCarta™ Server

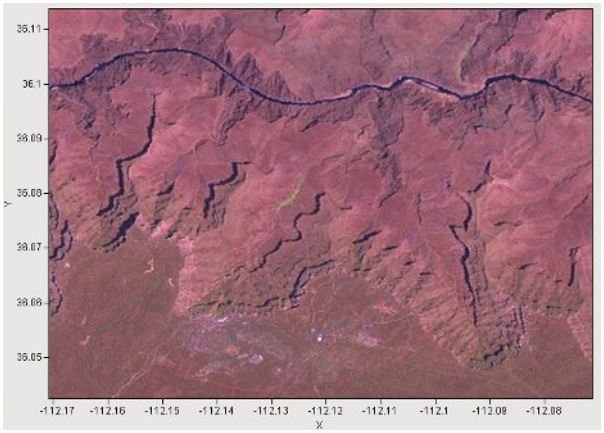
The NASA server provides maps for the entire Earth. The following figure displays the Earth using the NASA server as its source for the background map. The boundary map shows the outlines of the countries.

Figure 12.17 NASA Server Map Example



Not only does this server cover the entire Earth, but you can also zoom in on a much smaller area of the Earth and still get a reasonable map. The following figure displays the Colorado River running through the Grand Canyon in Arizona. The Grand Canyon Village is visible in the bottom of the map.

Figure 12.18 NASA Server Map Example - Zoom In on Colorado



If you look at the axes values, you can see that the area is less than 1/10° by 1/10°. The Simple Earth and Detailed Earth background maps do not display that type of resolution. The NASA server provides a fairly detailed view of any land mass on Earth. Water, however, is simply filled in as black. The NASA server is free to access, but it is also limited in availability. If the server is temporarily unavailable or becomes overloaded with requests, it delivers an error message instead of the requested map.

Another Internet-based option for background maps is a Web Map Service (WMS). The WMS option enables you to specify any server that supports the WMS interface. The NASA server is an example of a WMS server, but we have provided the URL and a layer name for you. With the WMS option, you must know the URL to the WMS server and a layer name supported by the server. Most WMS servers support multiple layers. For example, one layer can show terrain, another layer can show roads, and still another layer can include water, such as rivers and lakes. By specifying the URL for the server and the layer, JMP can make a request to the server and then display the map that is returned.

Unlike with simple and detailed maps, WMS maps do not wrap. You can scroll horizontally and vertically. However, beyond the -180 to 180 (x axis) and -90 to 90 (y axis) ranges, a plain background appears instead of the map. The limits of the axes are used to define the limits of the map that is displayed.

In order to use the WMS option for a background map, you need to decide which WMS server to use. There are many WMS servers freely available from the Internet. Most of them provide maps only for a particular area of the world, and each of them supports their own layers. So you have to search for the appropriate WMS server for your particular situation.

You can search for WMS servers on the Internet using your favorite search engine. Once you find one, you need to discover the layers that it supports. For this, you can use the WMS Explorer add-in. The WMS Explorer add-in generates a list of all the layers available on a server. You can select a layer from the list to see what it looks like. You can download the WMS Explorer add-in from the JMP File Exchange page.

Note: To use the WMS Explorer add-in and the WMS background map capabilities of JMP, your computer must be connected to the Internet.

To locate a server, launch the add-in through the menu items **Add-Ins > Map Images > WMS Explorer**. The add-in presents a text box for entering the url of a known WMS server. Alternatively, you can make a selection from a drop-down list of pre-discovered WMS servers (the list can be out of date). After specifying a WMS server, select **Get Layers**. Using Get Layers is not necessary if selecting from the drop-down list or if clicking **Enter** after entering a URL. This sends a request to the WMS server for a list of layers that the server supports. The returned list appears in the list box on the left, labeled **Layers**. A map of the world appears as an outline in the graph to the right. Selecting a layer makes a request to the WMS server to return a map, using the specified layer, that represents the entire earth. Selecting a different layer generates a different map.

The default maps do not cover the entire earth (for example, some WMS servers might provide mapping data for a particular county, within a state). In that case, it is likely that selecting a layer does not generate any visible map. You might have to zoom in on the appropriate area before any image map is visible. The standard JMP toolbar is available in the add-in window and the zoom tool works just like it does in any JMP window.

The graph is a typical graph in JMP, which means that all the regular JMP controls are available to you. You can adjust the axes or use the zoom tool (found on the menu bar) just as you would in JMP. You can also right-mouse-click to select **Size/Scale > Size to Isometric** to return the graph to a proper aspect ratio. You can also select Background Map, where you can adjust the boundary map.

Once a desirable map is determined, note the URL in the text box at the top and the selected layer in the Layers list. This is the information that you need to enter in the background map window when WMS is selected as the type of image background map.

Because requests are being made to a server across the Internet, there are a number of conditions that can generate an error. WMS servers often have limited availability and sometimes are not available at all. Occasionally a WMS server might return a name of a layer that it no longer supports. In these types of cases (and others), a server usually returns an error message in lieu of a map. If that happens, the error message is displayed below the Layers list in an area labeled **Errors**.

Boundaries

JMP can display boundaries (such as U.S. states or French region boundaries). These boundaries draw an outline around a defined area and can be displayed alone on a graph or combined with image data. Several boundaries are installed with JMP. Alternatively, you can create your own boundaries from Esri shapefiles or from scratch. Because of this, the list of Boundaries that you see in the Set Background Map window can be different.

When you add shape files to the built-in locations in JMP, they are available for the Graph Builder platform and for the Boundaries option in the Background Map window. In this way, you can add more political boundaries for use with background maps. Boundary-style maps are vector-based shapes.

Figure 12.19 Example of U.S. State Boundaries

Add a Background Map and Boundaries

To add a background map and boundaries:

1. Right-click a blank area on the graph and select **Background Map** (or select **Graph > Background Map** in Graph Builder).

   The Set Background Map window appears (Figure 12.12).
2. To display a background, do one of the following:
   - Select **Simple Earth**, **Detailed Earth**, **NASA server**, or **Street Map Service** in the Images column.
   - Select **Web Map Service** and paste a WMS URL next to URL. Enter the layer identifier next to **Layer**.
3. To display geographic borders on the map, select an option in the Boundaries column (If you installed your own boundary shapefiles, they are also listed in this column).
4. Click **OK**.

   If the NASA map, Street Map, or WMS map does not appear after you add it, the map server might not be available. View the error log to verify the problem.

Examples of Creating Maps

- "Louisiana Parishes Example"
- "Hurricane Tracking Examples"
- "Office Temperature Study"
- "Boundary Map with Clipped Elements"

Louisiana Parishes Example

In this example you work with custom map files and then create custom maps in two different ways:

- Set up custom map files initially and save them in the predetermined location. JMP finds and uses them in the future with any appropriate data.
- Point to specific predefined map files directly from your data. This step might be required each time you want to specify custom maps.

Set Up Automatic Custom Maps

Suppose that you have downloaded Esri shapefiles from the Internet and you want to use them as your map files in JMP. The shapefiles are named Parishes.shp and Parishes.dbf. These files contain coordinates and information about the parishes (or counties) of Louisiana.

Note: Pathnames in this section refer to the JMP folder. On Windows, in JMP Pro, the JMP folder is named JMPPro.

Save the .shp File

Save the .shp file with the appropriate name and in the correct directory.

1. In JMP, open the Parishes.shp file from the following default location:
   - On Windows: C:/Program Files/SAS/JMP/16/Samples/Import Data
   - On macOS: /Library/Application Support/JMP/16/Samples/Import Data

   Note: If you cannot see the file, you might need to change the file type to **All Files**.

   JMP opens the file as Parishes. The .shp file contains the x and y coordinates.
2. Save the Parishes file with the following name and extension: Parishes-XY.jmp. Save the file here:
   - On Windows: C:/Users/<user name>/AppData/Roaming/SAS/JMP/Maps
   - On macOS: /Users/<user name>/Library/Application Support/JMP/Maps
3. Close the Parishes-XY.jmp file.

Save the .dbf File

Perform the initial setup and save the .dbf file.

1. Open the Parishes.dbf file from the following default location:
   - On Windows: C:/Program Files/SAS/JMP/16/Samples/Import Data
   - On macOS: /Library/Application Support/JMP/16/Samples/Import Data

   Note: If you cannot see the file, you might need to change the file type to **All Files**.

   JMP opens the file as Parishes. The .dbf file contains identifying information.
2. Right-click the PARISH column and select **Column Info**.
3. Select **Column Properties > Map Role**.
4. Select **Shape Name Definition**.
5. Click **OK**.

6. Save the Parishes file with the following name and extension: Parishes-Name.jmp. Save the file here:
   - On Windows: C:/Users/<user name>/AppData/Roaming/SAS/JMP/Maps
   - On macOS: /Users/<user name>/Library/Application Support/JMP/Maps
7. Close the Parishes-Name.jmp file.

Create the Map in Graph Builder

Once the map files have been set up, you can use them. The Katrina.jmp data table contains data on Hurricane Katrina's impact by parish. You want to visually see how the population of the parishes changed after Hurricane Katrina.

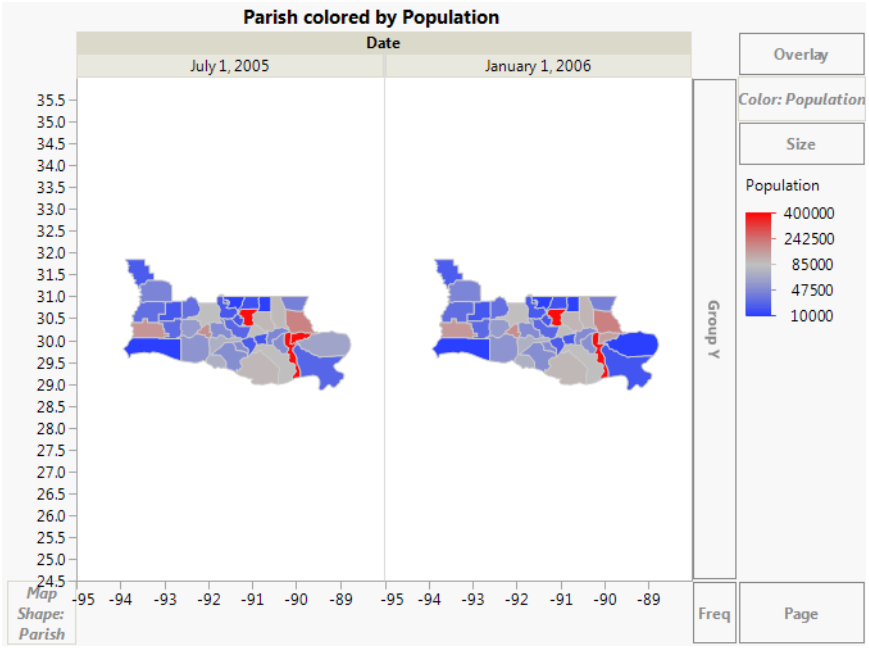
1. Select **Help > Sample Data Library** and open Katrina.jmp.
2. Right-click the Parish column and select **Column Properties > Map Role**.
3. Select **Shape Name Use**.
4. Click the **Map name data table** button and browse to select Parishes-Name.jmp, which you previously created.

   This tells JMP where the data tables containing the map information reside.
5. Select PARISH from the **Shape definition column** list.

   In Parishes-Name.jmp, the PARISH column has the **Shape Name Definition Map Role** property assigned. The column consists of map shape data for each parish.
6. Click **OK**.
7. Select **Graph > Graph Builder**.
8. Drag and drop Parish into the **Map Shape** zone.

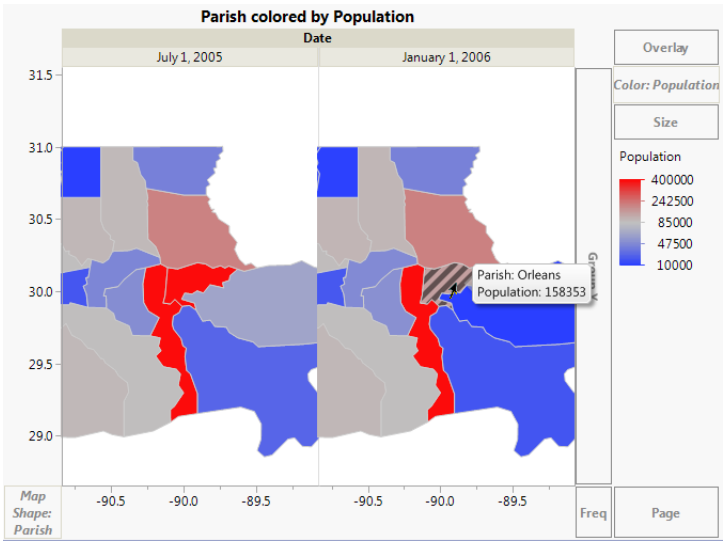
   The map appears automatically, since you defined the Parish column using the custom map files.
9. Drag and drop Population into the **Color** zone.
10. Drag and drop Date into the **Group X** zone.

Figure 12.20 Population of Parishes Before and After Katrina



11. Select the **Magnifier** tool to zoom in on the Orleans parish in both maps (Figure 12.21)

Figure 12.21 Orleans Parish



You can clearly see the drop in population as a result of Hurricane Katrina. The population of the Orleans parish went from 437,186 in July 2005 to 158,353 in January 2006.

Point to Existing Map Files Directly from Your Data

Suppose that you already have your custom map files and they are named appropriately. Your map files are US-MSA-Name.jmp and US-MSA-XY.jmp. They are saved in the sample data folder.

The PopulationByMSA.jmp data table contains population data from the years 2000 and 2010 for the metropolitan statistical areas (MSAs) of the United States. This example shows how the data table has been set up to create a map.

Add the Map Role Column Property

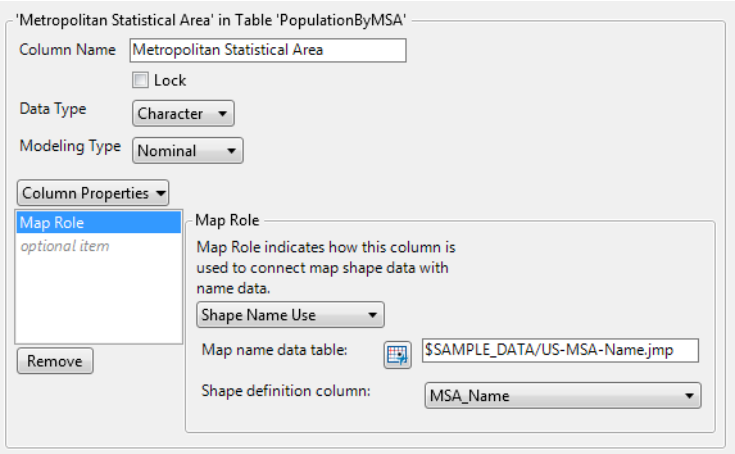
1. Select **Help > Sample Data Library** and open PopulationByMSA.jmp.
2. Right-click the **Metropolitan Statistical Area** column and select **Column Info**.
3. Select **Column Properties > Map Role**.
4. Select **Shape Name Use**.
5. Next to the **Map name data table**, type $SAMPLE_DATA/US-MSA-Name.jmp.

   This tells JMP where the data tables containing the map information reside.
6. Select MSA_Name from the **Shape definition column** list.

   MSA_Name is the specific column within the US-MSA-Name.jmp data table that contains the unique names for each metropolitan statistical area. Notice that the MSA_Name column has the **Shape Name Definition Map Role** property assigned, as part of correctly defining the map files.

Note: Remember, the Shape ID column in the -Name data table maps to the Shape ID column in the -XY data table. This means that indicating where the -Name data table resides links it to the -XY data table, so that JMP has everything that it needs to create the map.

Figure 12.22 Map Role Column Property



7. Click **OK**.

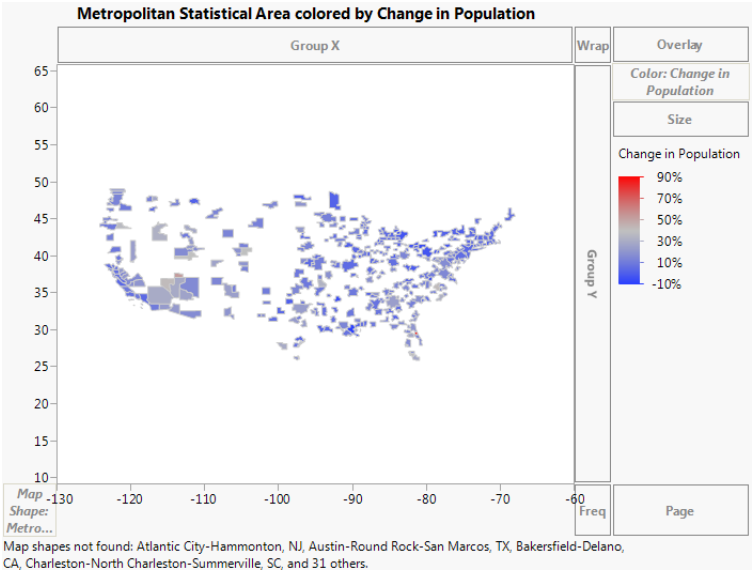
Create the Map in Graph Builder

Once the **Map Role** column property has been set up, you can perform your analysis. You want to visually see how the population has changed in the metropolitan statistical areas of the United States between the years 2000 and 2010.

1. Select **Graph > Graph Builder**.
2. Drag and drop **Metropolitan Statistical Area** into the **Map Shape** zone.

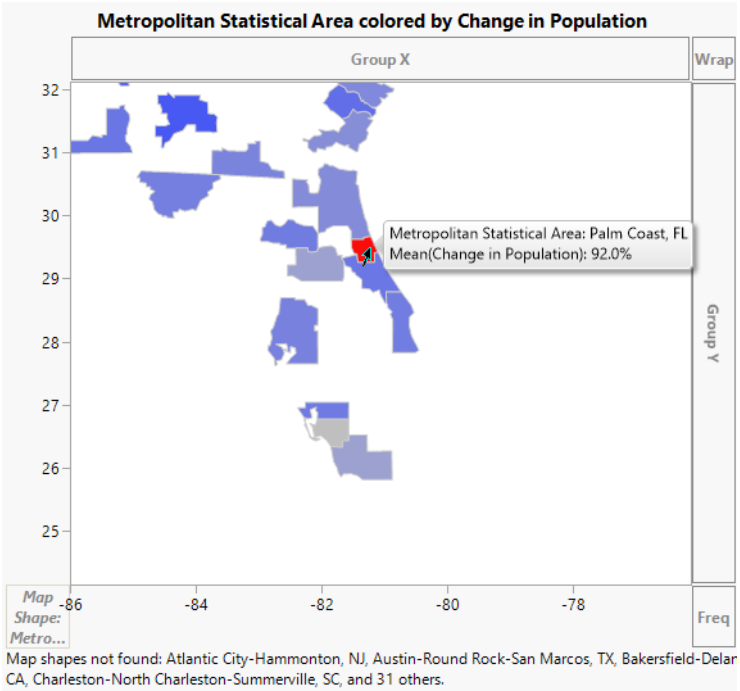
   Since you have defined the **Map Role** column property on this column, the map appears.
3. Drag and drop **Change in Population** to the **Color** zone.

Figure 12.23 Change in Population for Metropolitan Statistical Areas



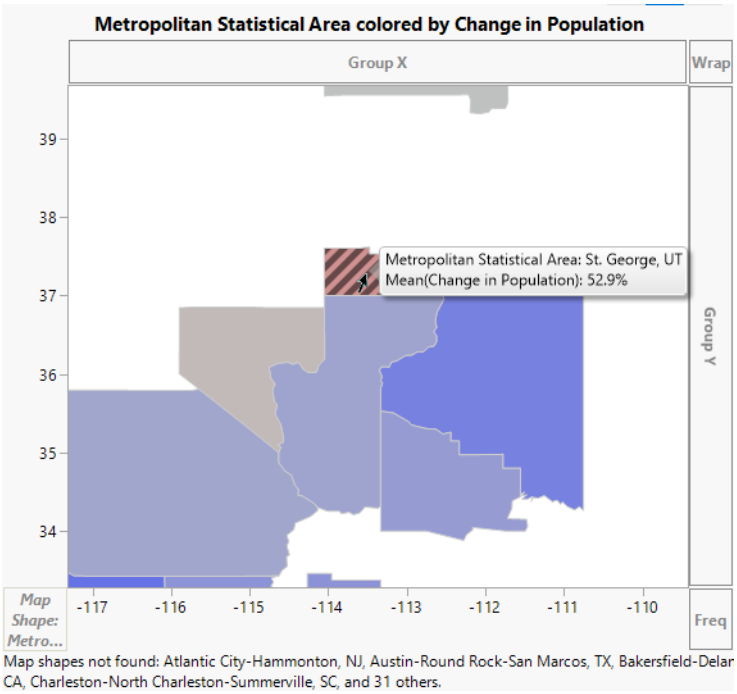
4. Select the **Magnifier** tool to zoom in on the state of Florida.
5. Select the **Arrow** tool and click the red area.

Figure 12.24 Population Change of Palm Coast, Florida



6. Select the **Magnifier** tool and hold down the Alt key while clicking on the map to zoom out.
7. Select the **Magnifier** tool and zoom in on the state of Utah.
8. Select the **Arrow** tool and click the area that is slightly red.

Figure 12.25 Population Change of St. George, Utah



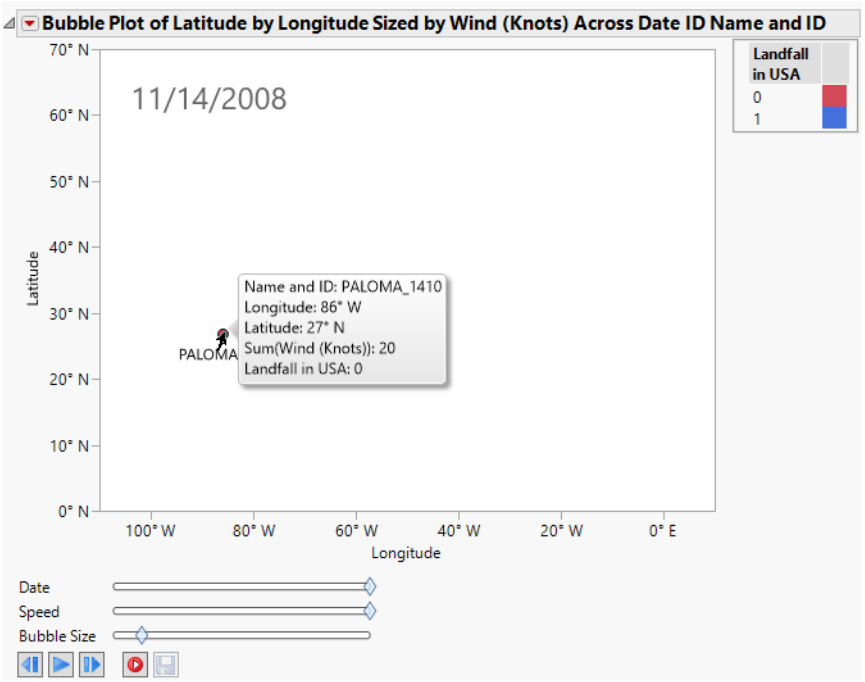
You can see that the areas of Palm Coast, Florida, and St. George, Utah had the most population change between 2000 and 2010. The Palm Coast area saw a population change of 92%, and the St. George area saw a population change of about 53%.

Hurricane Tracking Examples

This example uses the Hurricanes.jmp sample data table, which contains data on hurricanes that have affected the east coast of the United States. Adding a background map helps you see the areas the hurricanes affected. A script has been developed for this example and is part of the data table.

1. Select **Help > Sample Data Library** and open Hurricanes.jmp.
2. In the Table panel, click the green triangle next to the **Bubble Plot** script.
3. Drag the Date slider to the right as shown in Figure 12.26.
4. Click the red dot to display the name of the hurricane. The date appears in the upper left corner of the window. The red dot shows the location of Hurricane Paloma on November 14, 2008.

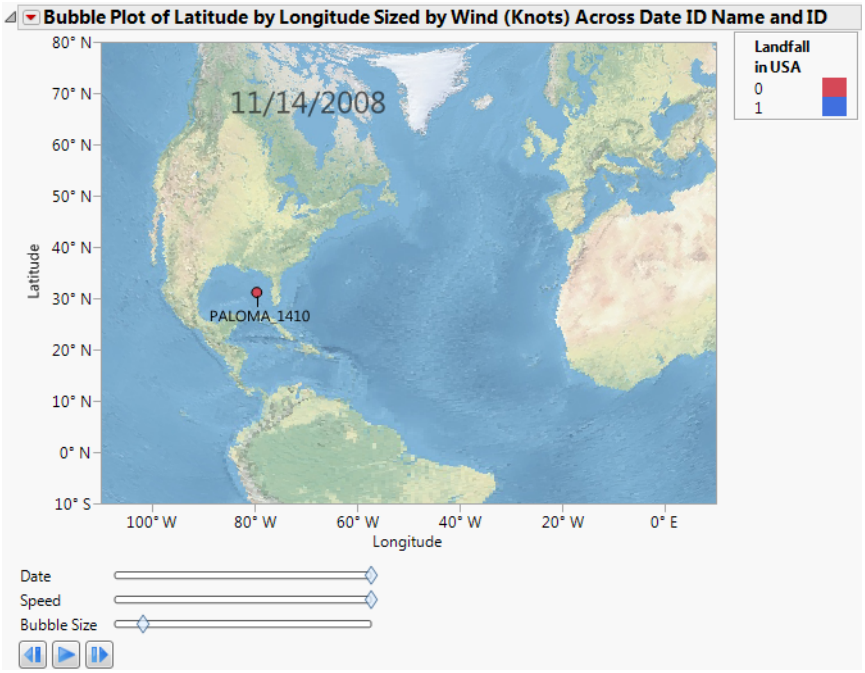
Figure 12.26 Bubble Plot of Hurricanes.jmp



Note that even though the location of the hurricane is plotted, it does not really tell us where it is. The axes information is there (27° North latitude and 86° West longitude), but we need a little more context. It is most likely over the middle of the Atlantic, but is it over a small island? This could make a big difference, especially for the inhabitants of the small island. Obviously, a map in the background of our graph would add a good deal of information.

5. Right-click the graph and select **Background Map**. The Set Background Map window appears (Figure 12.12).
6. Select **Detailed Earth** and click **OK**.

Figure 12.27 Bubble Plot of Hurricanes.jmp with Background Map

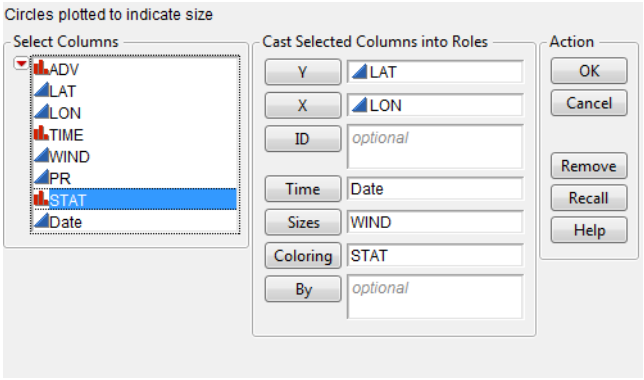


Now the coordinates make geographic sense. Click **Run** to view the animation of the hurricane data moving over the background map. Experiment with different options and view the displays. Adjust the axes or use the zoom tool to change what part of the world you are viewing. The map adjusts as the view does. You can also right-click the graph and select Size/Scale->Size to Isometric to get the aspect ratio of your graph to be proportional.

The next example uses the Katrina Data.jmp sample data table, which contains data on hurricane Katrina such as latitude, longitude, date, wind speed, pressure, and status. Adding a background map helps you see the path the hurricane took and impact on land based on size and strength. A script has been developed for this example and is part of the data table.

1. Select **Help > Sample Data Library** and open Katrina Data.jmp.
2. Select **Graph** > **Bubble Plot**.
3. Select LAT and click **Y**.
4. Select LON and click **X**.
5. Select Date and click **Time**.
6. Select WIND and click **Sizes**.
7. Select Stat and click **Coloring**.

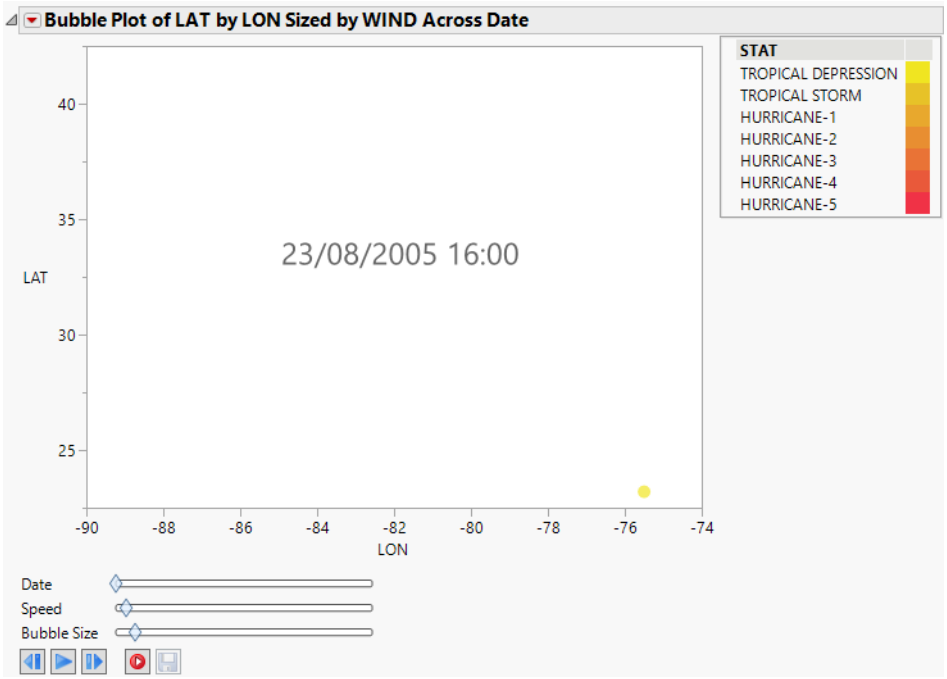
Figure 12.28 Bubble Plot Setup of Katrina Data.jmp



8. Click **OK**.

   The following image appears. The yellow dot shows the location of Tropical Depression Katrina on August 23, 2005.

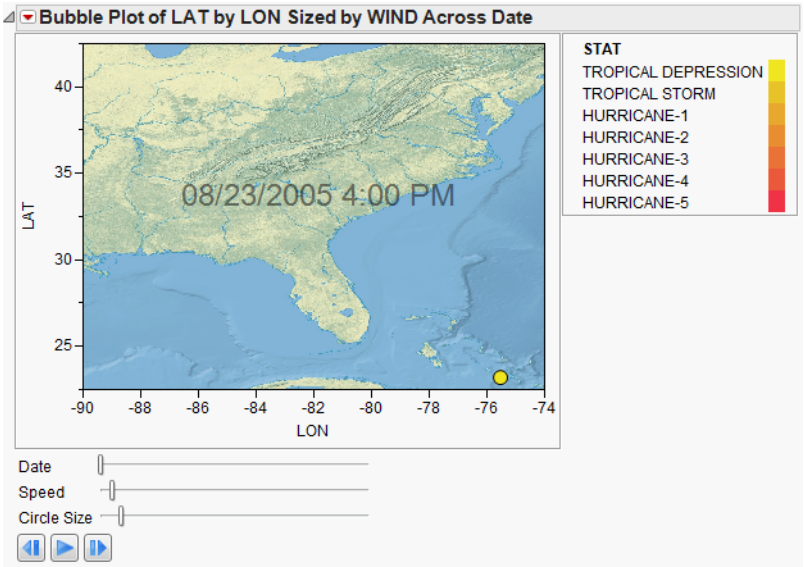
Figure 12.29 Bubble Plot of Katrina Data.jmp



Note that even though the location of the storm is plotted, it does not really tell us where it is. To add more context, add a map in the background.

9. Right-click the graph and select **Background Map**. The Set Background Map window appears.
10. Select **Detailed Earth** and click **OK**.

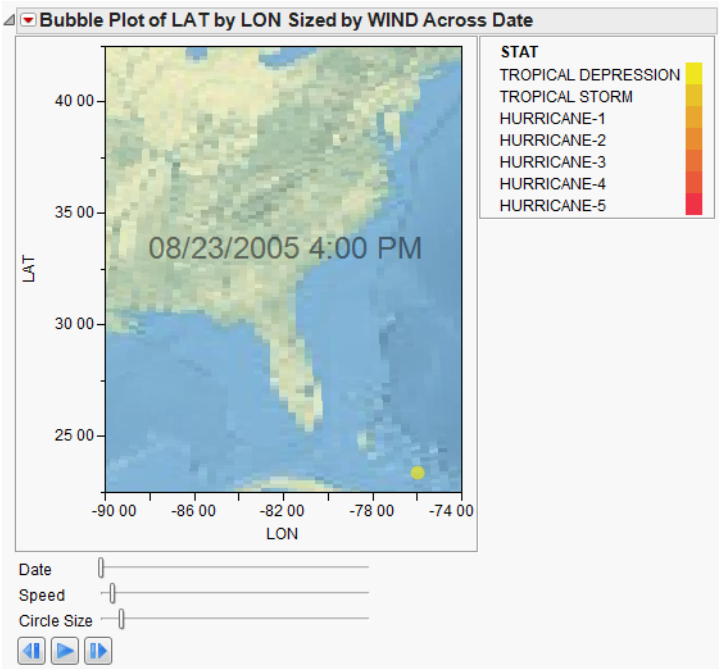
Figure 12.30 Bubble Plot of Katrina Data.jmp with Background Map



Now the coordinates make geographic sense. You can edit the axes and the size/scale to change how the graph appears.

11. Right-click the X axis (LON) and select **Axis Settings**. The X Axis Specification window appears.
12. Select **Scale** > **Geodesic US**.
13. Select **Format** > **Geographic** > **Longitude DMM**.
14. Click **OK**.
15. Repeat the same for the Y axis (LAT) except select **Format** > **Geographic** > **Latitude DMM**.
16. Right-click the map and select **Size/Scale** > **Size to Isometric.**

Figure 12.31 Bubble Plot of Katrina Data.jmp with Background Map



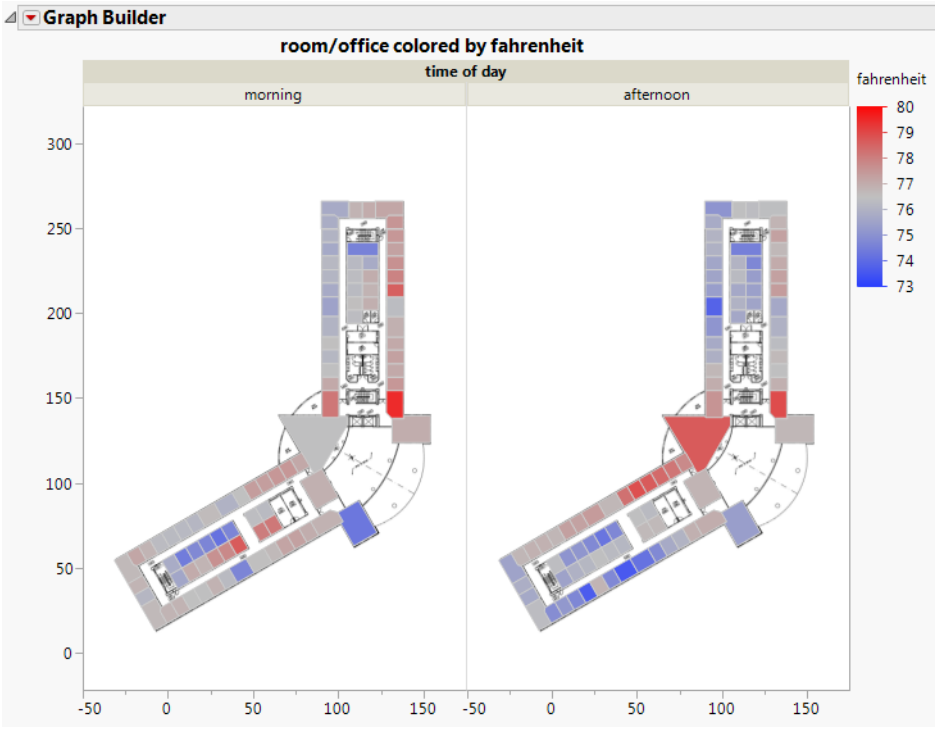
Click **Run** to view the animation of the hurricane data moving over the background map. You can manipulate the speed and the bubble size. Experiment with different options and view the displays. Adjust the axes or use the zoom tool to change what part of the area you are viewing. Add boundaries to the states. The map adjusts as the view does.

Office Temperature Study

This example demonstrates the creation of a custom background map for an office temperature study and how JMP was used to visualize the results. Data was collected concerning office temperatures for a floor within a building. A map was created for the floor using the Custom Map Creator add-in from the JMP File Exchange (https://community.jmp.com/docs/DOC-6218). Using Graph Builder, the office temperature results were then analyzed visually.

The map shown below is the floor, grouped by *time of day*. The color reflects the Fahrenheit value. Exploring data visually in this way can give hints as to what factors are affecting office temperature. Looking at this map, it appears the offices on the east side of the building are warmer in the mornings than they are in the afternoons. On the western side of the building, the opposite appears to be true. From this visualization, we might expect that both of these variables are affecting office temperatures, or perhaps that the interaction between these terms is significant. Such visuals help guide decision-making during the analysis.

Figure 12.32 Room/Office Colored by Fahrenheit and Grouped by Time of Day



First, data was collected and input into a data table (S4 Temps.jmp). Note the Room/Office column. It contains the unique names for each office and was assigned the Map Role to correctly define the map files.

Figure 12.33 Partial View of the S4 Temps.jmp Data Table

	day	time of day	thermometer	sector	room/office
1	1	morning	T4	sector 1	S4068
2	1	morning	T4	sector 1	S4069
3	1	morning	T4	sector 1	S4070
4	1	morning	T4	sector 1	S4072
5	1	morning	T4	sector 1	S4074
6	1	morning	T4	sector 1	S4076
7	1	morning	T4	sector 1	S4078
8	1	morning	T4	sector 1	S4082
9	1	morning	T4	sector 1	S4084
10	1	morning	T4	sector 1	S4086
11	1	morning	T4	sector 1	S4088
12	1	afternoon	T2	sector 1	S4068
13	1	afternoon	T2	sector 1	S4069

Then, a map of the floor was created using the Custom Map Creator add-in, which you can download from the JMP File Exchange at https://community.jmp.com/docs/DOC-6218. The add-in creates two tables to define the shapes; an XY table and a Name table. The instructions below describe how it was built.

Create a Map of the Floor

1. Launch the add-in through the menu items **Add-Ins > Map Shapes > Custom Map Creator**. Two tables open in the background followed by the Custom Map Creator Window.
2. Drag a background image into the graph frame. An image of the floor plan was available.
3. Perform any resizing on the background image and graph the frame.
4. Name the table (for example, S4).
5. Click **Next**.
6. Name the shape that you are about to define. For this example, each office was individually named for the map (for example, S4001).
7. Within the graph frame, use your mouse to click all of the boundaries of the shape that you want to define. A line appears that connects all of the boundary points.
8. As soon as you finish defining the boundaries of the shape, click **Next Shape**. Continue adding shapes until you have completed the floor plan. Note that you do not need to connect the final boundary point; the add-in automatically does that for you when you click **Next Shape**.
9. The line size and color can be changed. In addition, checking **Fill Shapes** fills each shape with a random color.
10. Click **Finish**.

The custom map files were created and named appropriately. The map files are S4-Name.jmp and S4-XY.jmp and have been saved in the JMP Samples/Data folder.

Add the Map Role Column Property

Note: Pathnames in this section refer to the JMP folder. On Windows, in JMP Pro, the JMP folder is named JMPPro.

The S4 Temps.jmp data table contains office data over a three-day period. Set up the **Map Role** column property in the data table:

1. Select **Help > Sample Data Library** and open S4 Temps.jmp.
2. Right-click the **Room/Office** column and select **Column Info**.
3. Select **Column Properties > Map Role**.
4. Select **Shape Name Use**.

5. Click the icon next to **Map name data table** and browse to the S4-Name.jmp file (located in the JMP Samples/Data folder).

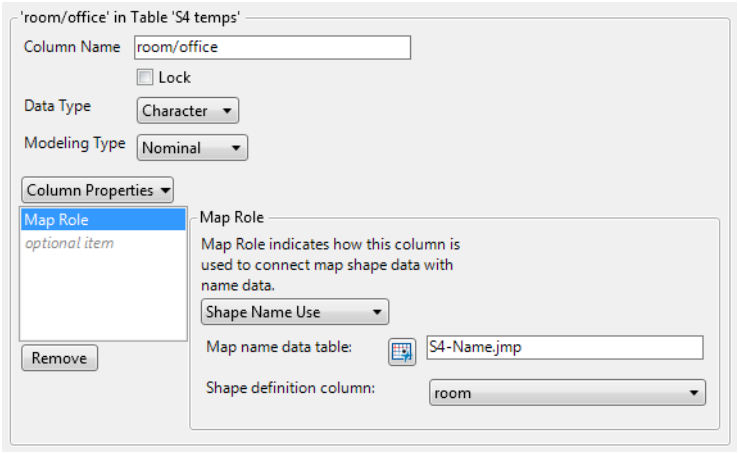
   This tells JMP where the data tables containing the map information reside.

6. Select room from the **Shape definition column** list.

   Room is the specific column within the S4-Name.jmp data table that contains the unique names for each office. Notice that the room column has the **Shape Name Definition Map Role** property assigned, as part of correctly defining the map files.

   Note: Remember, the Shape ID column in the -Name data table maps to the Shape ID column in the -XY data table. This means that indicating where the -Name data table resides links it to the -XY data table, so that JMP has everything that it needs to create the map.

Figure 12.34 Map Role Column Property



7. Click **OK**.

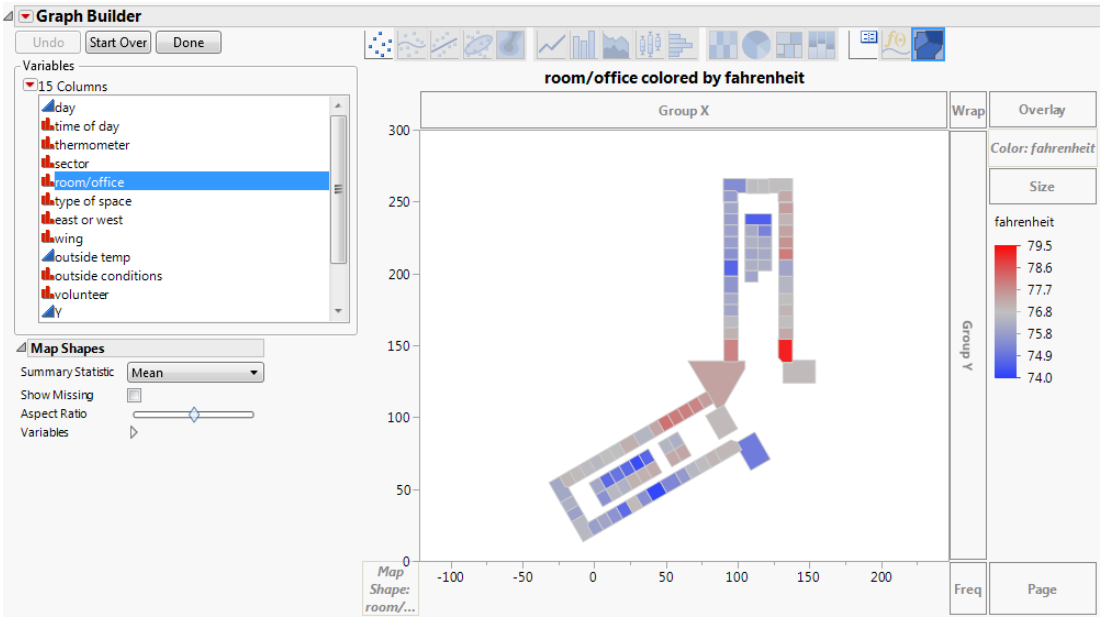
Once the **Map Role** column property has been set up, you can perform your analysis. You want to visually see the differences in office temperatures throughout the floor.

1. Select **Graph > Graph Builder**.
2. Drag and drop room/office into the **Map Shape** zone.

   Since you have defined the **Map Role** column property on this column, the map appears.

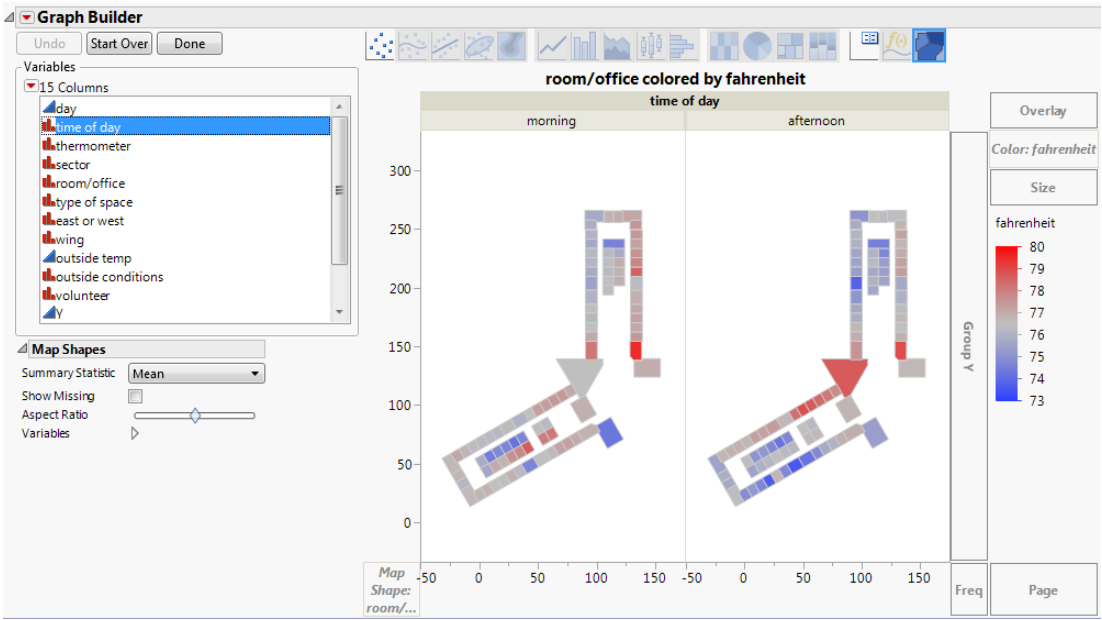
3. Drag and drop Fahrenheit to the **Color** zone.

Figure 12.35 Room/Office Colored by Fahrenheit



4. Drag and drop Time of Day onto Group X zone.

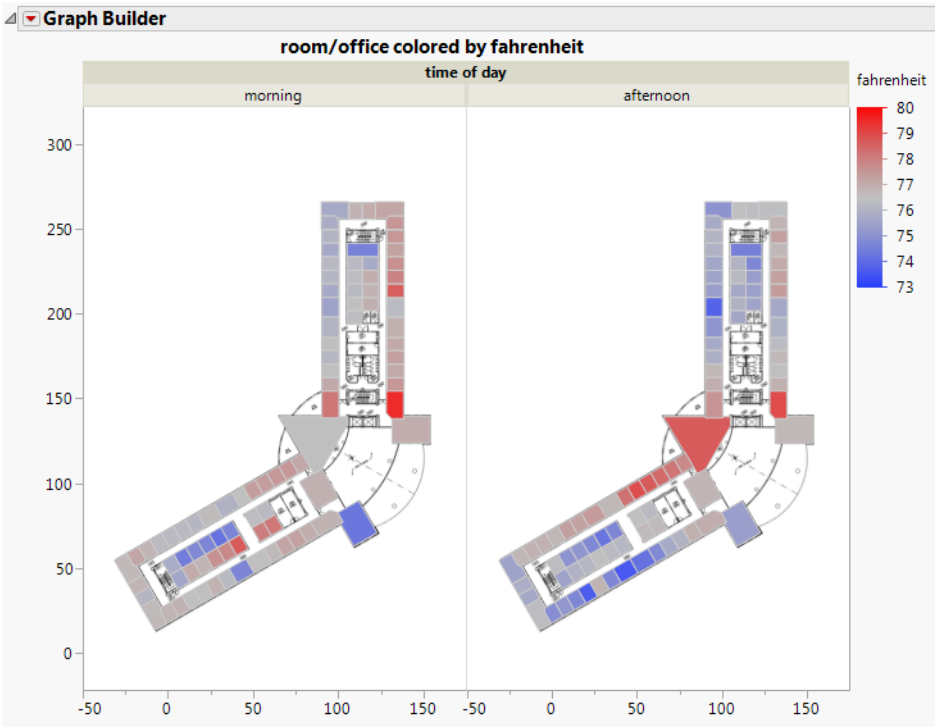
Figure 12.36 Room/Office Colored by Fahrenheit and Grouped by Time of Day



Note that only the offices that were part of the study and were created using the Custom Map Creator add-in are displayed. To add the entire floor plan image, the original floor plan graphic was dragged and dropped onto the Graph Builder window to create Figure 12.37.

To view Figure 12.37, select **Help > Sample Data Library** and open S4 Temps.jmp and run the **by Time of Day** script.

Figure 12.37 Room/Office Map with Original Floor Plan

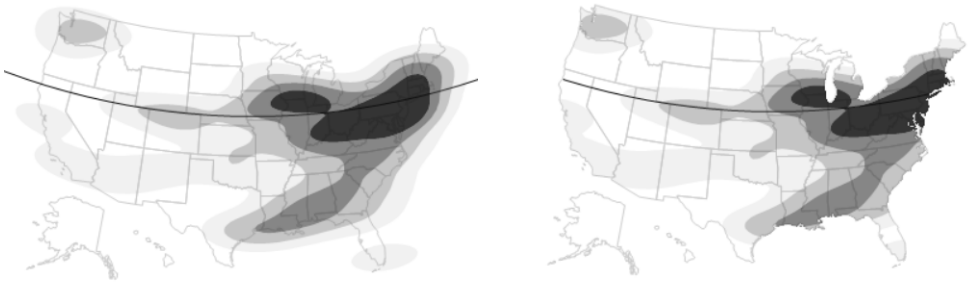


There are several scripts provided with the data table that you can run to view the various analysis and modeling that can be performed and visually displayed.

Boundary Map with Clipped Elements

Graphical elements (display segments such as reference, grid lines, and contours) can be clipped to conform to the boundaries of a geographical map. The Customize Graph window provides Clip Shape options, where you can select a boundary around which to clip the shape.

Figure 12.38 Unclipped (Left) and Clipped (Right) Reference Lines and Contours

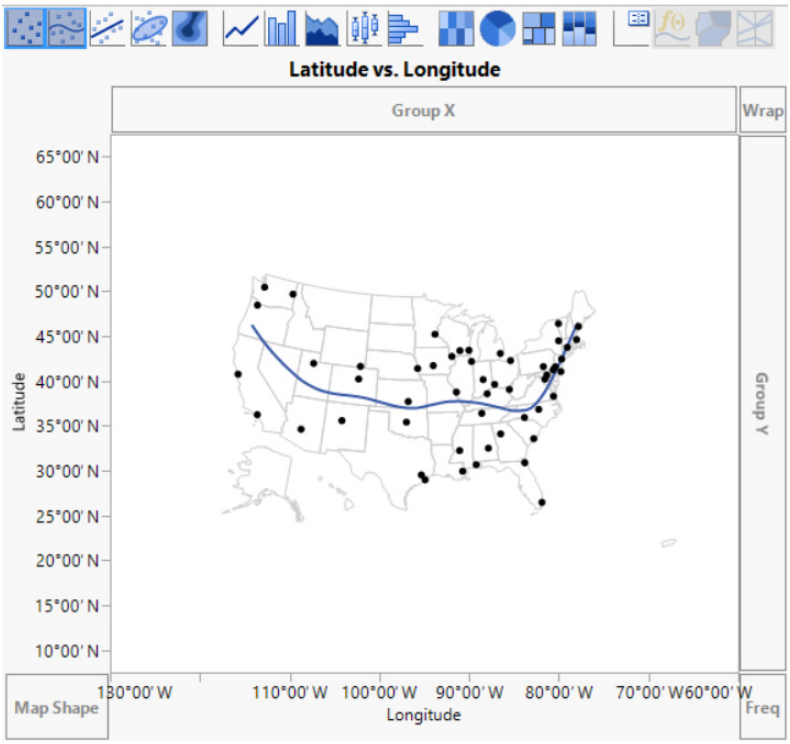


Create the Background Map

1. Select **Help > Sample Data Library** and open Cities.jmp.
2. Select **Graph > Graph Builder**.
3. Drag the Latitude column to the Y axis.
4. Drag the Longitude column to the X axis.
5. Right-click the graph and select **Graph > Background Map**.
6. In the Boundaries column, select **US States** and click **OK**.

   A map of the United States appears on the graph.

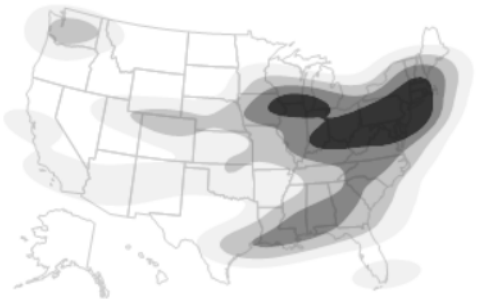
Figure 12.39 Background Map



7. Click the Points icon to remove the points.
8. Click the Smoother icon to remove the smoother.
9. Click the Contour icon .

   A contour plot appears on the map.

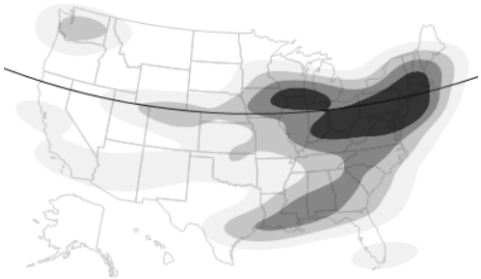
Figure 12.40 Unclipped Contours on Background Map



Add a Reference Line

1. On the Latitude axis, double-click **45°00’ N** to open the Y Axis Settings window.
2. In the Reference Lines pane, change the first number in the Value box to “40”.
   For example, in 44°40.11′ N, change “44” to “40”.
3. Click **Add** and then **OK**.

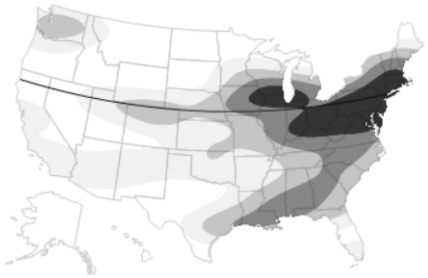
Figure 12.41 Unclipped Reference Line



Assign a Clip Shape

1. Right-click the graph and select **Customize**.
2. Select **Reference Lines** and select **US States** from the Clip Shape list.
3. Select **Contour**, select **US States** from the Clip Shape list, and click **OK**.

Figure 12.42 Clipped Reference Line and Contours



Tips:

- The boundaries that appear in the Background Map window are installed in the JMP Maps installation folder. See “Custom Map Files” for more information about adding your own boundaries.
- Scripting provides additional options, such as specifying the clipping path in a matrix or string. See the *Scripting Guide*.

Appendix A

Legacy Platforms

The following legacy platforms are deprecated. A deprecated platform is an older platform that contains features that can be accessed in a newer platform. The charts and plots in this appendix are all available in the Graph Builder platform. These legacy platforms might not be available in future releases of JMP.

Table A.1 Legacy Platforms

Platform	Graph Builder References
Treemap	• "Treemap"
Chart	• Bar chart examples in "Graph Builder Examples" • "Bar", "Line", or "Use a Heatmap to Apply Background Colors"
Overlay Plot	• "Example of an Overlaid Histogram and Ridgeline Chart" • "Graph Builder"

Treemaps

Treemaps are useful for observing patterns among groups that have many levels. Use treemaps when your data contains many categories, to visualize many groups.

Tip: To create a treemap, consider using Graph Builder. See "Treemap".

Launch the Treemap platform by selecting **Graph > Legacy > Treemap**.

When you click **OK** in the launch window, a treemap in Graph Builder is created by default. To produce a report in the legacy Treemap platform, you must first deselect the Use Graph Builder platform preference under Files > Preferences > Platforms > Treemap. For documentation on the legacy Treemap platform, see the Legacy Platforms section of the JMP 15.2 Help.

Charts

The Chart platform on the **Graph** menu charts continuous variables versus categorical variables. The continuous variables are summarized for each categorical level.

If you want to make a plot of individual data points (rather than summaries of data points), we recommend using an overlay plot instead.

Tip: To create a chart, consider using Graph Builder. See "Example of an Overlaid Histogram and Ridgeline Chart".

Launch the Chart platform by selecting **Graph > Legacy > Chart**.

When you click **OK** in the launch window, a chart in Graph Builder is created by default. To produce a report in the legacy Chart platform, you must first deselect the Use Graph Builder platform preference under Files > Preferences > Platforms > Chart. For documentation on the legacy Chart platform, see the Legacy Platforms section of the JMP 15.2 Help.

Overlay Plots

The Overlay Plot option produces plots of a single X column and one or more numeric Ys and does not accept non-numeric values for the *y*-axis.

Tip: To create an overlay plot, consider using the Overlay zone in Graph Builder. See "Example of an Overlaid Histogram and Ridgeline Chart".

Launch Overlay Plot by selecting **Graph > Legacy > Overlay Plot**.

When you click **OK** in the launch window, an overlay plot in Graph Builder is created by default. To produce a report in the legacy Overlay Plot platform, you must first deselect the Use Graph Builder platform preference under Files > Preferences > Platforms > Overlay Plot. For documentation on the legacy Overlay Plot platform, see the Legacy Platforms section of the JMP 15.2 Help.

Appendix B

References

Aitchison, J. (1986). *The Statistical Analysis of Compositional Data*. London: Chapman & Hall.

Box, G. E. P., Hunter, W. G., and Hunter, J. S. (1978). *Statistics for Experimenters*. New York: John Wiley & Sons.

Cornell, J. A. (1990). *Experiments with Mixtures*. 2nd ed. New York: John Wiley & Sons.

Few, Stephen (2008). "Dual-Scaled Axes in Graphs." *Visual Business Intelligence Newsletter*. http://www.perceptualedge.com/articles/visual_business_intelligence/dual-scaled_axes.pdf.

Hyndman, R. J. (1996). "Computing and Graphing High Density Regions."*The American Statistician* 50: 120 - 126.

Hyndman, R. J., Koehler, A. B., Ord, J. K., and Snyder, R. D. (2008). *Forecasting with Exponential Smoothing: The State Space Approach*. Berlin: Springer-Verlag.

Inselberg, A. (1985). "The Plane with Parallel Coordinates." *Visual Computing* 1: 69–91.

Mardia, K. V., Kent, J. T., and Bibby J. M. (1979). *Multivariate Analysis*. New York: Academic Press.

Rousseeuw, P. J., Ruts, I., and Tukey, J. W. (1999). "The Bagplot: A Bivariate Boxplot." *The American Statistician* 53: 382 - 387.

Wegman, E. J. (1990). "Hyperdimensional Data Analysis using Parallel Coordinates." *Journal of the American Statistical Association* 85: 664–675.

Appendix C

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

  - You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - You must cause any modified files to carry prominent notices stating that You changed the files; and
  - You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
  - You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Profilers

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. JMP® 16 *Profilers*. Cary, NC: SAS Institute Inc.

JMP® 16 Profilers

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Profilers

1 Learn about JMP 11
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 13
JMP Help 14
JMP Documentation Library 14
Additional Resources for Learning JMP 20
JMP Tutorials 20
Sample Data Tables 20
Learn about Statistical and JSL Terms 21
Learn JMP Tips and Tricks 21
JMP Tooltips 21
JMP User Community 22
Free Online Statistical Thinking Course 22
JMP New User Welcome Kit 22
Statistics Knowledge Portal 22
JMP Training 22
JMP Books by Users 23
The JMP Starter Window 23
JMP Technical Support 23

2 Introduction to Profilers 25
Visualize Response Surfaces and Optimize Processes
Overview of the Profiler Platform 27
Introduction to Profiling 28
Profiler Features in JMP 28
Profiler Launch Windows 30
Fit Group 31
Interpret the Profiles 32
Profiler Platform Options 35
Common Profiler Topics 36
Linear Constraints 36
Noise Factors 38

3 Profiler ... 41
Explore Cross Sections of Responses across Each Factor
Example of the Prediction Profiler ... 43
Launch the Prediction Profiler Platform ... 44
Prediction Profiler Options ... 44
Maximization Options Window ... 52
Desirability Profiling and Optimization ... 53
Construction of Desirability Functions ... 54
How to Use the Desirability Function ... 55
The Desirability Profile ... 57
Customized Desirability Functions ... 57
Assess Variable Importance ... 59
Bagging ... 63
Additional Examples of the Prediction Profiler ... 64
Example of Desirability Profiling for Multiple Responses ... 65
Example of a Noise Factor in the Prediction Profiler ... 66
Example of Variable Importance for One Response ... 72
Example of Variable Importance for Multiple Responses ... 74
Example of Bagging to Improve Prediction ... 76
Example of Bagging to Indicate the Accuracy of Predictions ... 79
Example of Confidence Intervals Using Bagging ... 82
Example of Confidence Intervals Using StdErr Pred Formula Column ... 84
Statistical Details for the Prediction Profiler ... 85
Assess Variable Importance ... 85
Propagation of Error Bars ... 87
Extrapolation Control Metrics ... 88

4 Contour Profiler ... 91
Explore Contours of Responses across Two Factors
Overview of the Contour Profiler ... 93
Example of the Contour Profiler ... 93
Launch the Contour Profiler Platform ... 95
The Contour Profiler Report ... 95
Factor Settings and Controls ... 96
Response Settings and Controls ... 96
Contour Profiler Platform Options ... 97
Constraint Shading Settings ... 99
Additional Example of the Contour Profiler ... 99
Explore Optimal Settings ... 99

5 Surface Plot ... 103
Explore Contours of Responses across Three Factors
Overview of the Surface Plot and Surface Profiler ... 105
Example of the Surface Plot Platform ... 105
Launch the Surface Plot Platform ... 108
The Surface Plot Report ... 110
The Surface Plot ... 110
Appearance Controls ... 111
Independent Variables ... 112
Dependent Variables ... 113
Surface Plot Platform Options ... 115
Pop-Up Menu Options ... 115
Dependent Variables Options ... 116
Sheet or Surface Properties ... 117
Keyboard Shortcuts ... 119
Additional Examples of the Surface Plot Platform ... 119
Construct a Surface Plot for a Single Mathematical Function ... 119
Plotting Isosurfaces ... 120

6 Mixture Profiler ... 123
Explore Factor Effects Using Ternary Plots
Overview of the Mixture Profiler ... 125
Ternary Plot Overview ... 125
Ternary Plots with More Than Three Components ... 126
Example of the Mixture Profiler ... 127
Launch the Mixture Profiler Platform ... 128
The Mixture Profiler Report ... 129
Factor Settings and Controls ... 129
Response Settings and Controls ... 130
Mixture Profiler Platform Options ... 130
Customizations for the Mixture Profiler ... 131
Linear Constraints ... 131
Additional Examples of the Mixture Profiler Platform ... 132
Mixture Variables with a Process Variable ... 132
Multiple Responses with Five Mixture Variables ... 135

7 Custom Profiler ... 141
Explore Response Surfaces Using a Numerical Calculator
Overview of the Custom Profiler ... 143
Example of the Custom Profiler ... 143
Launch the Custom Profiler Platform ... 145

The Custom Profiler Report ... 146
Factor Settings and Controls ... 146
Response Settings and Controls ... 147
Optimization Settings and Controls ... 147
Custom Profiler Platform Options ... 148

8 Simulator ... 149
Explore the Effects of Variation on Responses

Overview of the Simulator ... 151
Example of the Simulator ... 151
Launch the Simulator ... 154
Factors Simulation Settings ... 154
Response Simulation Settings ... 157
The Simulator Report ... 157
The Simulator Report Options ... 158
Defect Profiler ... 159
Introduction to Tolerance Design ... 159
Graph Scale ... 160
Defect Rates ... 160
Simulation Method and Details ... 161
Notes ... 161
Defect Parametric Profiler ... 162
Simulation Experiment ... 163
Specification Limits ... 163
Additional Examples of the Simulator ... 163
Example of the Defect Profiler ... 164
Example of Stochastic Optimization ... 167
Example of Simulating General Formulas ... 176
Statistical Details for the Simulator ... 179
Normal Weighted Distribution ... 179

9 Excel Profiler ... 181
Visualize Models Saved in Microsoft Excel

Overview of the Excel Profiler ... 183
Example of an Excel Model ... 183
Run the JMP Profiler ... 185
Use Linear Constraints ... 186
Resolution of Profile Lines ... 186
Use the Excel Profiler from JMP ... 187

A References 189
B Technology License Notices 191

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 13
JMP Help . . . 14
JMP Documentation Library . . . 14
Additional Resources for Learning JMP . . . 20
JMP Tutorials . . . 20
Sample Data Tables . . . 20
Learn about Statistical and JSL Terms . . . 21
Learn JMP Tips and Tricks . . . 21
JMP Tooltips . . . 21
JMP User Community . . . 22
Free Online Statistical Thinking Course . . . 22
JMP New User Welcome Kit . . . 22
Statistics Knowledge Portal . . . 22
JMP Training . . . 22
JMP Books by Users . . . 23
The JMP Starter Window . . . 23
JMP Technical Support . . . 23

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

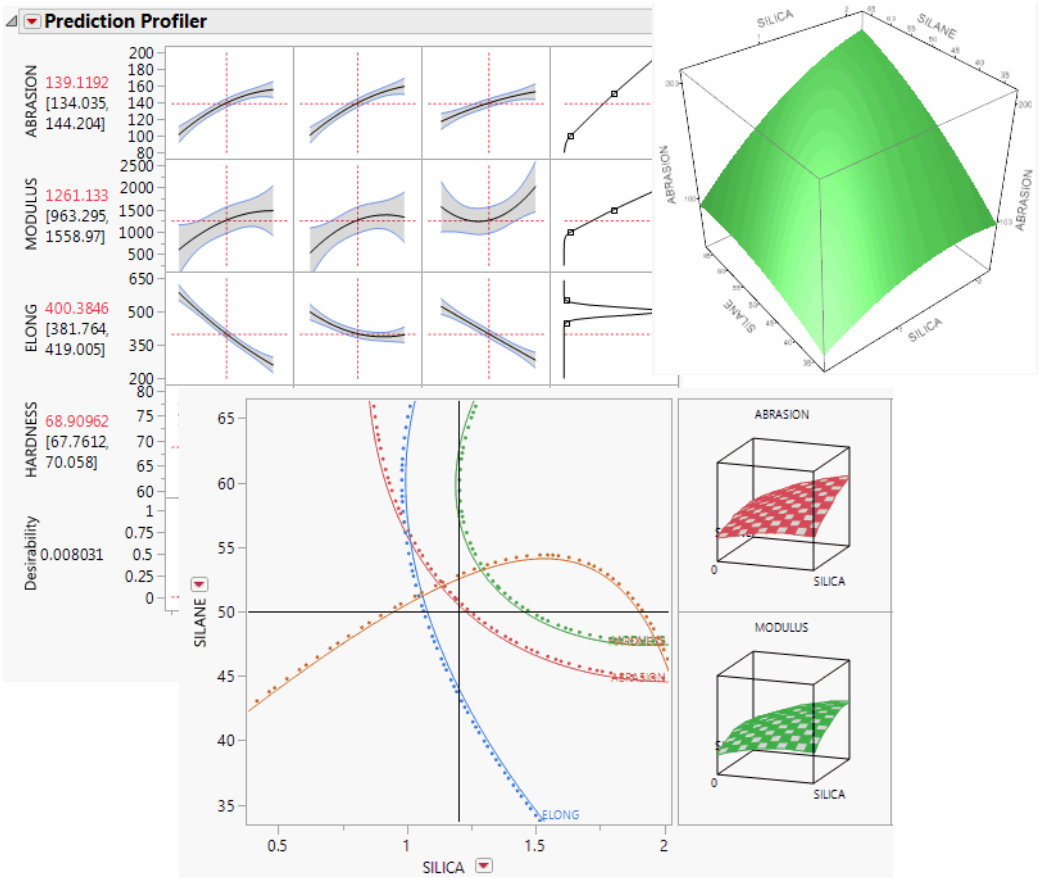
Chapter 2

Introduction to Profilers

Visualize Response Surfaces and Optimize Processes

Profiling is an approach to visualizing response surfaces by seeing what would happen if you change just one or two factors at a time. Essentially, a profile is a cross-section view. The interactive profilers in JMP promote exploring opportunity spaces. In fitting equations to data, the fitting is only the first step. Interpreting the fit, understanding the fitted response surface, and finding factor values to optimize the responses is desirable.

Figure 2.1 Examples of Profilers



Contents

Overview of the Profiler Platform 27
Introduction to Profiling 28
Profiler Features in JMP 28
Profiler Launch Windows 30
Fit Group 32
Interpret the Profiles 32
Profiler Platform Options 36
Common Profiler Topics 37
Linear Constraints 37
Noise Factors 39

Overview of the Profiler Platform

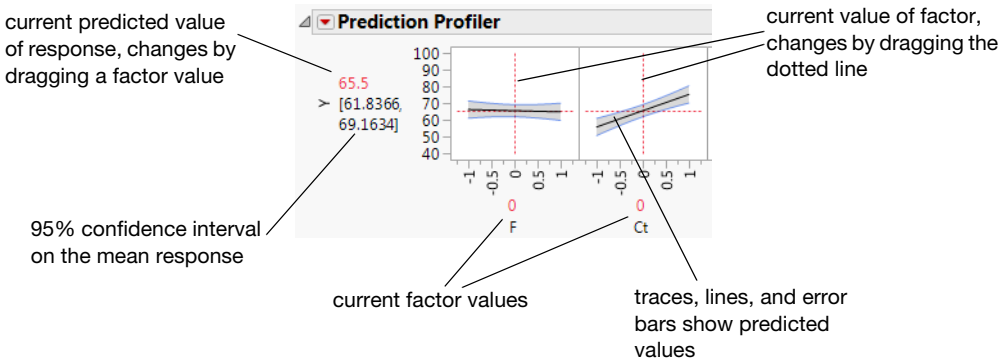
The Prediction Profiler displays profile traces (Figure 2.2) for each *X* variable. A *profile trace* is the predicted response as one variable is changed while the others are held constant at the current values. The Prediction Profiler recomputes the profiles and predicted responses (in real time) as you vary the value of an *X* variable.

- The vertical dotted line for each *X* variable shows its *current value* or *current setting*. If the variable is nominal, the X axis identifies categories.

  For each *X* variable, the value above the factor name is its current value. Change the current value by clicking in the graph or by dragging the dotted line where you want the new current value to be.
- The horizontal dotted line shows the *current predicted value* of each *Y* variable for the current values of the *X* variables.
- The black lines within the plots show how the predicted value changes when you change the current value of an individual *X* variable. In fitting platforms, the 95% confidence interval for the predicted values is shown by solid blue curves surrounding the prediction trace (for continuous variables) or the height of an error bar (for categorical variables). For continuous variables, the confidence interval region is shaded.

The Prediction Profiler is a way of changing one variable at a time and looking at the effect on the predicted response.

Figure 2.2 Illustration of Traces



The Prediction Profiler in some situations computes confidence intervals for each profiled column. If you have saved both a standard error formula and a prediction formula for the same column, the Prediction Profiler offers to use the standard errors to produce the confidence intervals rather than profiling them as a separate column.

Introduction to Profiling

It is easy to visualize a response surface with one input factor *X* and one output factor *Y*. It becomes harder as more factors and responses are added. The profilers in JMP provide a number of highly interactive cross-sectional views of any response surface. In this guide we use the following terms interchangeably.

- factor, input variable, *X* column, independent variable, setting, term
- response, output variable, *Y* column, dependent variable, outcome

Desirability profiling and optimization features are available to help find good factor settings and produce desirable responses. Most profilers also incorporate multithreading for faster computation. Simulation and defect profiling features are available for when you need to make responses that are robust and high-quality when the factors have variation.

- "Profiler Features in JMP"
- "Profiler Launch Windows"
- "Fit Group"

Profiler Features in JMP

There are several profiler facilities in JMP, accessible from a number of fitting platforms and the main menu under Graph. They are used to profile data column formulas.

Table 2.1 Profiler Features Summary

	Description	Features
Prediction Profiler	Shows vertical slices across each factor, holding other factors at current values	Desirability, Optimization, Simulator, Propagation of Error
Contour Profiler	Horizontal slices show contour lines for two factors at a time	Simulator

Table 2.1 Profiler Features Summary *(Continued)*

	Description	Features
Surface Profiler	3-D plots of responses for 2 factors at a time, or a contour surface plot for 3 factors at a time	Surface Visualization
Mixture Profiler	A contour profiler for mixture factors	Ternary Plot and Contours
Custom Profiler	A non-graphical profiler and numerical optimizer	General Optimization, Simulator
Excel Profiler	Visualize models (or formulas) stored in Excel worksheets.	Profile using Excel Models

Profiler availability is shown in Table 2.2. The Custom Profiler is available only through the Graph menu. (Model Comparison *does* have Custom Profiler available.)

Table 2.2 Where to Find JMP Profilers

Location	Profiler	Contour Profiler	Surface Profiler	Mixture Profiler
Graph Menu (as a Platform)	Yes	Yes	Yes	Yes
Fit Model: Least Squares	Yes	Yes	Yes	Yes
Fit Model: Generalized Regression	Yes			
Fit Model: Mixed Model	Yes	Yes	Yes	Yes
Fit Model: Logistic	Yes			
Fit Model: Loglinear Variance	Yes	Yes	Yes	
Fit Model: Generalized Linear Model	Yes	Yes	Yes	
Fit Model: Partial Least Squares	Yes			
Neural	Yes	Yes	Yes	
Model Comparison	Yes	Yes	Yes	Yes
Nonlinear: Factors and Response	Yes	Yes	Yes	
Nonlinear: Parameters and SSE	Yes	Yes	Yes	
Nonlinear: Fit Curve	Yes			

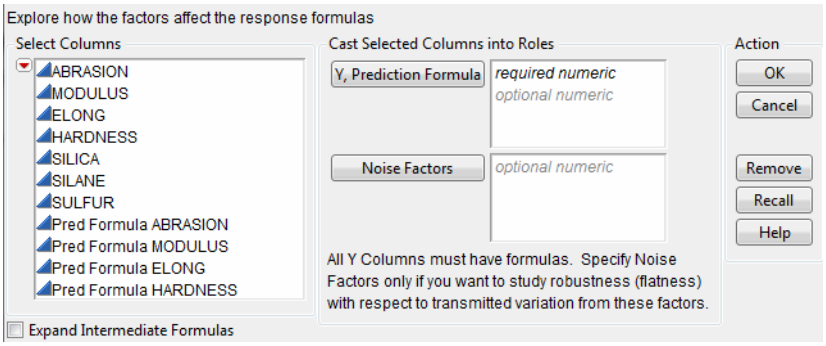
Table 2.2 Where to Find JMP Profilers *(Continued)*

Location	Profiler	Contour Profiler	Surface Profiler	Mixture Profiler
Gaussian Process	Yes	Yes	Yes	
Partial Least Squares	Yes			
Life Distribution	Yes			
Fit Life by X	Yes		Yes	
Recurrence Analysis	Yes			
Choice	Yes			
Custom Design Prediction Variance	Yes		Yes	

Profiler Launch Windows

When a profiler is invoked as a platform from the Graph menu, rather than through a fitting platform, you provide columns with formulas as the Y, Prediction Formula columns. These formulas could have been saved from the fitting platforms.

Figure 2.3 Profiler Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The columns referenced in the formulas become the *X* columns (unless the column is also a *Y*).

Y, Prediction Formula The response columns containing formulas.

Note: You can also add columns that contain the formula and the values for the standard errors of the corresponding response columns. These columns are used to create confidence intervals. The formula columns must be named PredSE <colname>. After you click **OK**, specify whether you want to use PredSE <colname> to construct confidence intervals for Pred Formula <colname>. Otherwise, JMP creates a separate profiler plot for PredSE <colname>.

Noise Factors Used only in special cases for modeling derivatives. For more information about noise factors, see "Noise Factors".

Expand Intermediate Formulas Tells JMP that if an ingredient column to a formula is itself a formula column that refers to other columns, to substitute the original columns into the inner formula. To prevent an ingredient column from expanding, add an Other column property, name it Expand Formula, and assign a value of 0. See "Expand Intermediate Formulas".

The Surface Plot platform is discussed in "Surface Plot". The Surface Profiler is very similar to the Surface Plot platform, except Surface Plot has more modes of operation. Neither the Surface Plot platform nor the Surface Profiler have some of the capabilities common to other profilers.

Expand Intermediate Formulas

The Profiler launch window has an Expand Intermediate Formulas check box. When this box is checked, the formula that is being profiled is handled differently. If the profiled formula contains another formula column that references other columns, then the original columns are substituted into the inner formula. Therefore, the formula being profiled is profiled with respect to the original columns instead of the intermediate column references. Expand Intermediate Formula also expands formula columns that have the Vector modeling type.

For example, when Fit Model fits a logistic regression for two levels (A and B), the end formulas (Prob[A] and Prob[B]) are functions of the Lin[x] column, which itself is a function of another column x. If Expand Intermediate Formulas is selected, then when Prob[A] is profiled, it is with reference to x, not Lin[x].

In addition, using the Expand Intermediate Formulas check box enables the Save Expanded Formulas command in the platform red triangle menu. This creates a new column with a formula, which is the formula being profiled as a function of the end columns, not the intermediate columns.

Fit Group

For the REML and Stepwise personalities of the Fit Model platform, if models are fit to multiple *Y*'s, the results are combined into a Fit Group report. This enables the different *Y*'s to be profiled in the same Profiler. The Fit Group red triangle menu has options for launching the joint Profiler. Profilers for the individual *Y*'s can still be used in the respective Fit Model reports.

Fit Group reports are also created when a By variable is specified for a Stepwise analysis. This allows for the separate models to be profiled in the same Profiler.

The Fit Group scripting command can be used to fit models in different platforms, and profile the individual models in the Profiler. See the *Scripting Guide*.

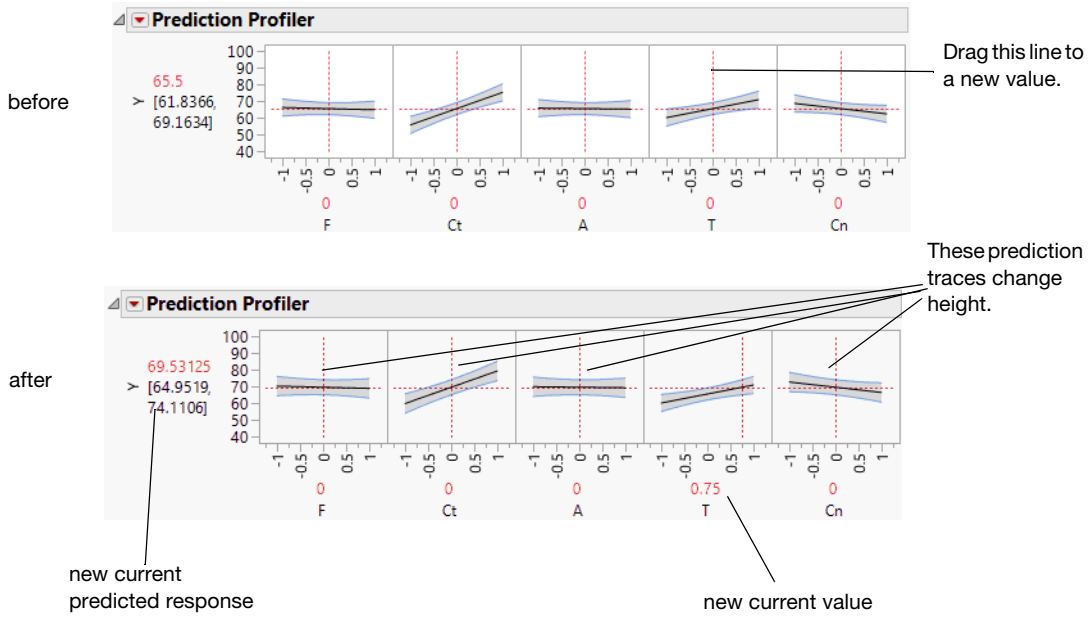
Interpret the Profiles

The illustration in Figure 2.4 describes how to use the components of the Prediction Profiler. There are several important points to note when interpreting a prediction profile:

- The importance of a factor can be assessed to some extent by the steepness of the prediction trace. If the model has curvature terms (such as squared terms), then the traces might be curved.
- If you change the value of a factor, the prediction trace for that factor is not affected, but the prediction traces of all the other factors can change. The *Y* response line must cross the intersection points of the prediction traces with their current value lines.

Note: If there are interaction effects or cross-product effects in the model, the prediction traces can shift their slope and curvature as you change current values of other terms. That is what interaction is all about. If there are no interaction effects, the traces change only in height, not slope or shape.

Figure 2.4 Changing One Factor from 0 to 0.75



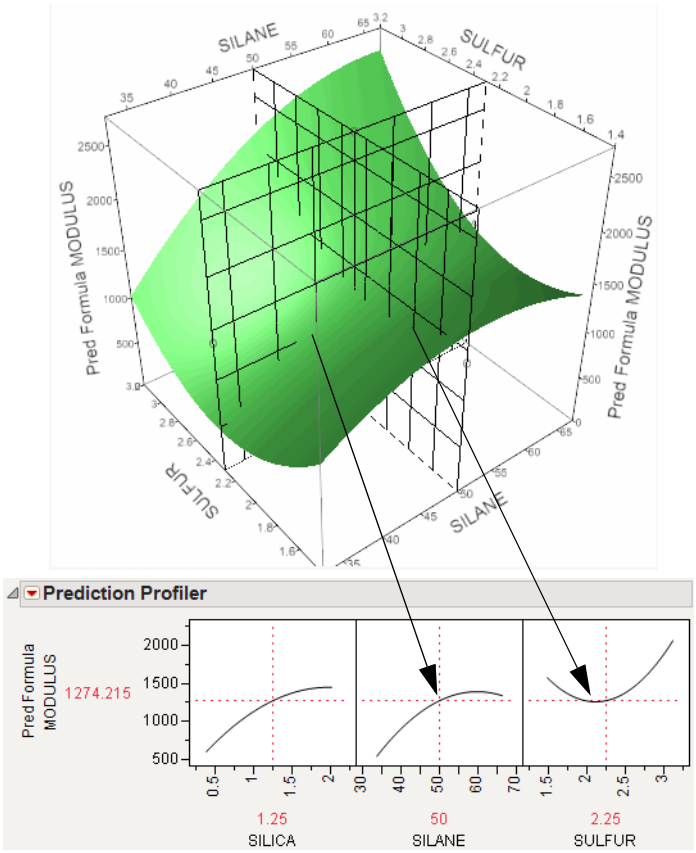
Prediction profiles are especially useful in multiple-response models to help judge which factor values can optimize a complex set of criteria.

Click a graph or drag the current value line right or left to change the factor's current value. The response values change as shown by a horizontal reference line in the body of the graph. Double-click in an axis to bring up a window that changes its settings.

The Profiler as a Cross-Section

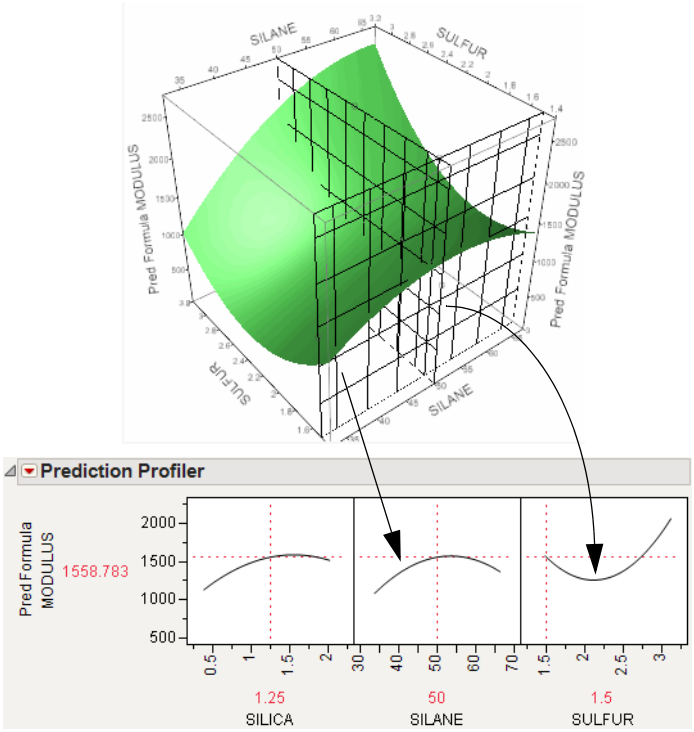
In the following example using Tiretread.jmp, look at the response surface of the expression for MODULUS as a function of SULFUR and SILANE (holding SILICA constant). Now look at how a grid that cuts across SILANE at the SULFUR value of 2.25. Note how the slice intersects the surface. If you transfer that down below, it becomes the profile for SILANE. Similarly, note the grid across SULFUR at the SILANE value of 50. The intersection when transferred down to the SULFUR graph becomes the profile for SULFUR.

Figure 2.5 Profiler as a Cross-Section



Now consider changing the current value of SULFUR from 2.25 to 1.5.

Figure 2.6 Profiler as a Cross-Section

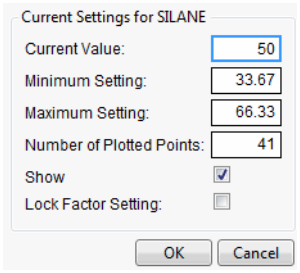


In the Prediction Profiler, note the new value just moves along the same curve for SULFUR, the SULFUR curve itself does not change. But the profile for SILANE is now taken at a different cut for SULFUR. The profile for SILANE is also a little higher and reaches its peak in the different place, closer to the current SILANE value of 50.

Set or Lock Factor Values

If you press Alt (Option on macOS) and click in a graph, a window prompts you to enter specific settings for the factor.

Figure 2.7 Continuous Factor Settings Window



For continuous variables, you can specify the following:

Current Value The value used to calculate displayed values in the profiler, equivalent to the red vertical line in the graph.

Minimum Setting The minimum value of the factor's axis.

Maximum Value The maximum value of the factor's axis.

Number of Plotted Points Specifies the number of points used in plotting the factor's prediction traces.

Show Show or hide the factor in the profiler.

Lock Factor Setting Locks the value of the factor at its current setting.

Profiler Platform Options

The Profiler red triangle menu contains the following options:

Profiler Shows or hides the Prediction Profiler.

Contour Profiler Shows or hides the Contour Profiler.

Custom Profiler Shows or hides the Custom Profiler.

Surface Profiler Shows or hides the Surface Profiler.

Mixture Profiler Shows or hides the Mixture Profiler.

Show Formulas Opens a JSL window showing all formulas being profiled.

Formulas for OPTMODEL Creates code for the OPTMODEL SAS procedure. Press Ctrl+Shift and then select **Formulas for OPTMODEL** from the red triangle menu.

The following options are available in many platforms. See *Using JMP*.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Common Profiler Topics

- "Linear Constraints"
- "Noise Factors"

Linear Constraints

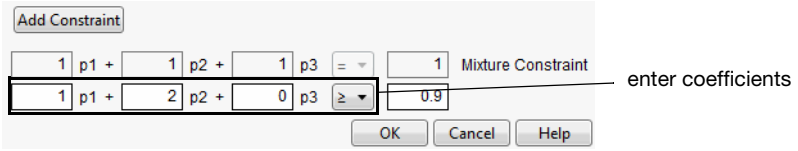
The Prediction Profiler, Custom Profiler, and Mixture Profiler can incorporate linear constraints into their operations. Linear constraints can be entered in two ways, described in the following sections.

Red Triangle Menu Item

To enter linear constraints via the red triangle menu, select **Alter Linear Constraints** from either the Prediction Profiler or Custom Profiler red triangle menu.

Choose **Add Constraint** from the resulting window, and enter the coefficients into the appropriate boxes. For example, to enter the constraint p1 + 2*p2 ≥ 0.9, enter the coefficients as shown in Figure 2.8. As shown, if you are profiling factors from a mixture design, the mixture constraint is present by default and cannot be modified.

Figure 2.8 Enter Coefficients



After you click **OK**, the Profiler updates the profile traces, and the constraint is incorporated into subsequent analyses and optimizations.

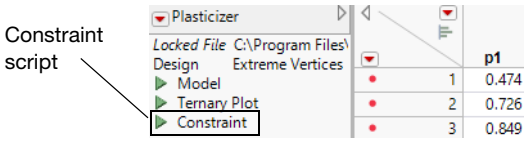
If you attempt to add a constraint for which there is no feasible solution, a message is written to the log and the constraint is not added. To delete a constraint, enter zeros for all the coefficients.

Constraints added in one profiler are not accessible by other profilers until saved. For example, if constraints are added under the Prediction Profiler, they are not accessible to the Custom Profiler. To use the constraint, you can either add it under the Custom Profiler red triangle menu, or use the **Save Linear Constraints** command described in the next section.

Constraint Table Property/Script

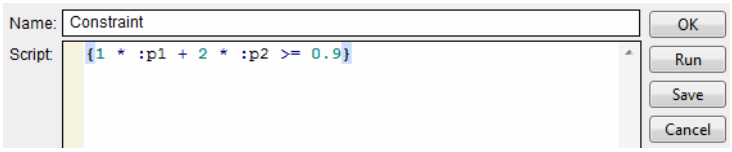
If you add constraints in one profiler and want to make them accessible by other profilers, use the **Save Linear Constraints** command, accessible through the platform red triangle menu. For example, if you created constraints in the Prediction Profiler, choose **Save Linear Constraints** under the Prediction Profiler red triangle menu. The Save Linear Constraints command creates or alters a Table Script called Constraint. An example of the Table Property is shown in Figure 2.9.

Figure 2.9 Constraint Table Script



The Constraint Table Property is a list of the constraints, and is editable. It is accessible to other profilers, and negates the need to enter the constraints in other profilers. To view or edit Constraint, right-click the green triangle and select **Edit**. The content of the constraint from Figure 2.8 is shown below in Figure 2.10.

Figure 2.10 Example Constraint



The Constraint Table Script can be created manually by choosing **New Script** from the red triangle menu beside a table name.

Note: When creating the Constraint Table Script manually, the spelling must be exactly "Constraint". Also, the constraint variables are case sensitive and must match the column name. For example, in Figure 2.10, the constraint variables are p1 and p2, not P1 and P2.

The Constraint Table Script is also created when specifying linear constraints when designing an experiment.

The Alter Linear Constraints and Save Linear Constraints commands are not available in the Mixture Profiler. To incorporate linear constraints into the operations of the Mixture Profiler, the Constraint Table Script must be created by one of the methods discussed in this section.

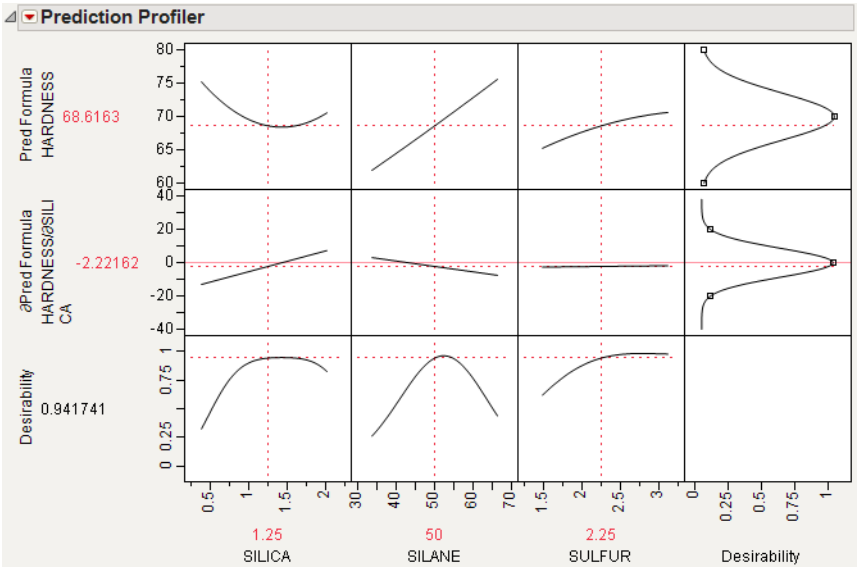
Noise Factors

Note: Noise factor optimization is also available in the Prediction Profiler, Contour Profiler, Custom Profiler, and Mixture Profiler.

Noise factors (robust process engineering) enables you to produce acceptable products reliably, despite variation in the process variables. Even when your experiment has controllable factors, there is a certain amount of uncontrollable variation in the factors that affects the response. This is called transmitted variation. Factors with this variation are called noise factors. Some factors you cannot control at all, like environmental noise factors. The mean for some factors can be controlled, but not their standard deviation. This is often the case for intermediate factors that are output from a different process or manufacturing step.

A good approach to making the process robust is to match the target at the flattest place of the noise response surface. Then, the noise has little influence on the process. Mathematically, this is the value where the first derivatives of each response with respect to each noise factor are zero. JMP computes the derivatives for you.

Figure 2.11 Noise Factor Example



To analyze a model with noise factors:

1. Fit the appropriate model (for example, using the Fit Model platform).
2. Save the model to the data table with the **Save > Prediction Formula** command.
3. Launch the **Profiler** (from the **Graph** menu).

4. Assign the prediction formula to the **Y, Prediction Formula** role and the noise factors to the **Noise Factor** role.
5. Click **OK**.

   The resulting profiler shows response functions and their appropriate derivatives with respect to the noise factors. The derivatives are set to have maximum desirability at zero.
6. Select **Optimization and Desirability > Maximize Desirability** from the **Profiler** menu.

   This finds the best settings of the factors, balanced with respect to minimizing transmitted variation from the noise factors.

Chapter 3

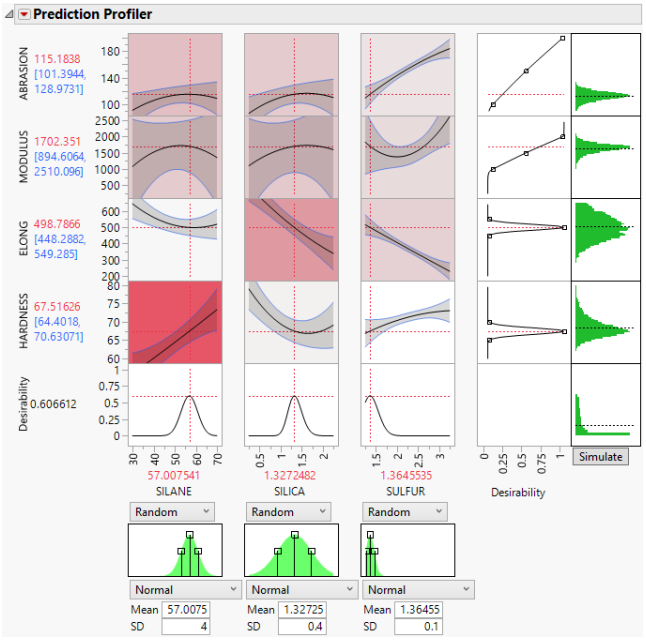
Profiler

Explore Cross Sections of Responses across Each Factor

The Prediction Profiler gives you a wealth of information about your model. Use the Prediction Profiler to do the following:

- See how your prediction model changes as you change settings of individual factors.
- Set desirability goals for your response or responses, and find optimal settings for your factors.
- Gauge your model's sensitivity to changes in the factors, where sensitivity is based on your predictive model.
- Assess the importance of your factors relative to model predictions, in a way that is independent of the model.
- Simulate your response distribution based on specified distributions for both factors and responses, and control various aspects of the appearance of the profiler.

Figure 3.1 Prediction Profiler for Four Responses with Simulator and Importance Coloring



Contents

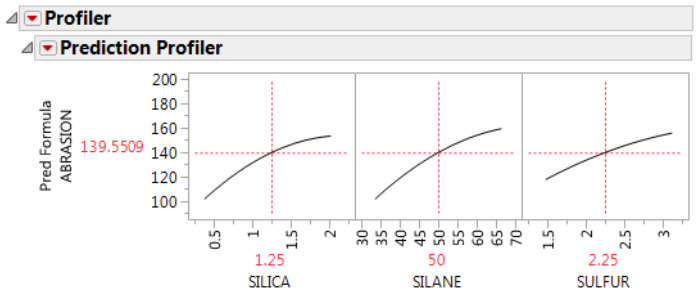
Example of the Prediction Profiler 43
Launch the Prediction Profiler Platform 44
Prediction Profiler Options 44
Maximization Options Window 52
Desirability Profiling and Optimization 53
Construction of Desirability Functions 54
How to Use the Desirability Function 55
The Desirability Profile 57
Customized Desirability Functions 57
Assess Variable Importance 59
Bagging 63
Additional Examples of the Prediction Profiler 64
Example of Desirability Profiling for Multiple Responses 65
Example of a Noise Factor in the Prediction Profiler 66
Example of Variable Importance for One Response 72
Example of Variable Importance for Multiple Responses 74
Example of Bagging to Improve Prediction 76
Example of Bagging to Indicate the Accuracy of Predictions 79
Example of Confidence Intervals Using Bagging 82
Example of Confidence Intervals Using StdErr Pred Formula Column 84
Statistical Details for the Prediction Profiler 85
Assess Variable Importance 85
Propagation of Error Bars 87
Extrapolation Control Metrics 88

Example of the Prediction Profiler

This example uses the Tiretread.jmp sample data table. There are three factors (SILICA, SILANE, and SULFUR) and four responses (ABRASION, MODULUS, ELONG, and HARDNESS). The goal is to find the optimal combination of the three factors in the development of a tire tread compound. See Derringer and Suich (1980).

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Graph > Profiler**.
3. Select Pred Formula Abrasion and click **Y, Prediction Formula**.
4. Click **OK**.

Figure 3.2 Prediction Profiler Report Window

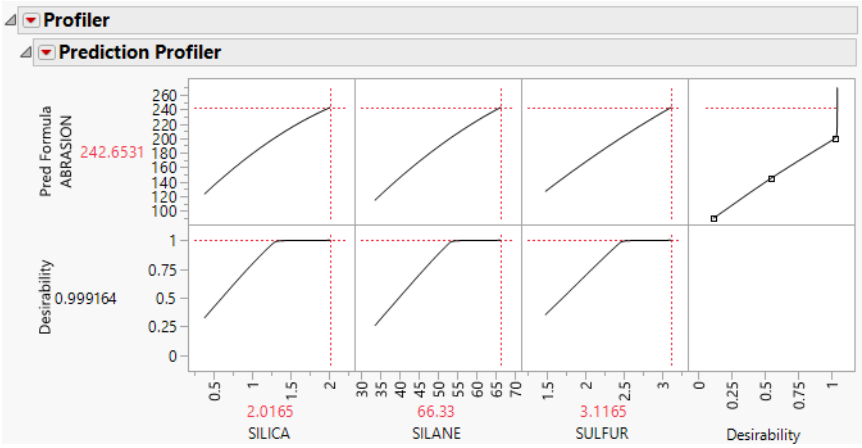


The profiler is interactive. The vertical red lines correspond to the current value of the factors. The current value of each factor is also shown in red below the horizontal axis. The red value on the vertical axis is the predicted response based on the current values of the factors. Click and drag the vertical red lines to change the current values of the factors. Click a factor's red value to set the factor to a specific value. Alternatively, you can animate the profiler to cycle through a variety of factor values.

5. Click the Prediction Profiler red triangle and select **Factor Settings > Animation**.
6. Click the play button to sequentially cycle through the factor values, one factor at a time.
7. Click the Prediction Profiler red triangle and select **Optimize and Desirability > Desirability Functions**.
8. Click the Prediction Profiler red triangle and select **Optimize and Desirability > Maximize Desirability.**

Tip: You might need to adjust the vertical axis of the prediction profiler to view the maximum value of ABRASION in the plot.

Figure 3.3 Maximize Desirability Report



By maximizing the desirability, you see that to maximize ABRASION, all factors are set to their highest levels.

Launch the Prediction Profiler Platform

The Prediction Profiler can be accessed in the following ways:

- The Prediction Profiler can be accessed directly from the Graph menu. When you access the Prediction Profiler in this way, the Prediction Profiler launch window appears. See "Profiler Launch Windows".
- The Prediction Profiler can be accessed as a red triangle menu option in many modeling platforms. See "Where to Find JMP Profilers" for more information about the availability of the Prediction Profiler in different platforms.
- The Prediction Profiler can be accessed from the Model Comparisons platform. Select Profiler from the Model Comparisons red triangle menu.
- JMP PRO The Prediction Profiler can be accessed from the Formula Depot platform. Select Profiler from the Formula Depot red triangle menu.

Prediction Profiler Options

The red triangle menu on the Prediction Profiler title bar has the following options:

Optimization and Desirability Submenu that consists of the following options:

Desirability Functions Shows or hides the desirability functions. Desirability is discussed in "Desirability Profiling and Optimization".

Maximize Desirability Sets the current factor values to maximize the desirability functions. Takes into account the response importance weights.

Note: In many situations, the settings that optimize the desirability function are not unique. The Maximize Desirability option gives one such setting. The Contour Profiler is a good tool for finding alternative factor combinations that optimize desirability. For an example, see "Explore Optimal Settings".

Note: If a factor has a Design Role column property value of Discrete Numeric, it is treated as continuous in the optimization of the desirability function. To account for the fact that the factor can assume only discrete levels, it is displayed in the profiler as a categorical term and an optimal allowable level is selected.

Maximize and Remember Maximizes the desirability functions and remembers the associated settings.

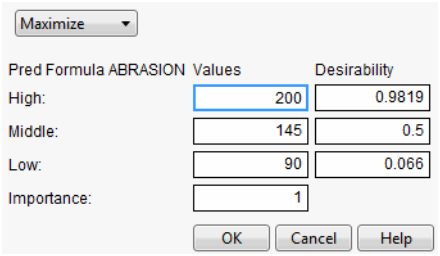
Maximization Options Opens the Maximization Options window where you can refine the optimization settings. See "Maximization Options Window".

Maximize for Each Grid Point Used only if one or more factors are locked. The ranges of the locked factors are divided into a grid, and the desirability is maximized at each grid point. This is useful if the model that you are profiling has categorical factors. Then the optimal condition can be found for each combination of the categorical factors.

Save Desirabilities Saves the three desirability function settings for each response, and the associated desirability values, as a Response Limits column property in the data table. These correspond to the coordinates of the handles in the desirability plots.

Set Desirabilities Opens the Response Goal window where you can set specific desirability values.

Figure 3.4 Response Goal Window



Save Desirability Formula Creates a column in the data table with a formula for Desirability. The formula uses the fitting formula when it can, or the response variables when it cannot access the fitting formula.

Assess Variable Importance Provides different approaches to calculating indices that measure the importance of factors to the model. These indices are independent of the model type and fitting method. See "Assess Variable Importance".

Save Bagged Predictions (Available only when the Prediction Profiler is embedded in select modeling platforms.) Launches the Bagging window. Bootstrap aggregating (bagging) enables you to create multiple training data sets by sampling with replacement from the original data. For each training set, a model is fit using the analysis platform, and predictions are made. The final prediction is a combination of the results from all of the models. This improves prediction performance by reducing the error from variance. See "Bagging".

Simulator Launches the Simulator. The Simulator enables you to create Monte Carlo simulations using random noise added to factors and predictions for the model. A typical use is to set fixed factors at their optimal settings, and uncontrolled factors and model noise to random values. You then find out the rate of responses outside the specification limits. See "Simulator".

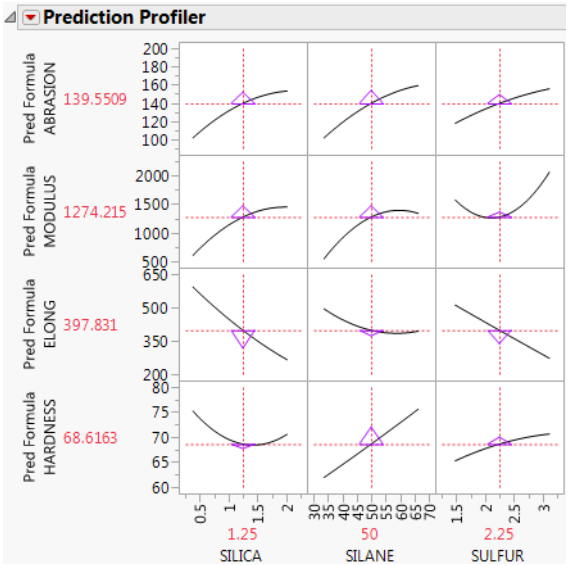
Interaction Profiler Shows or hides interaction plots that update as you update the factor values in the Prediction Profiler. Use this option to help visualize third degree interactions by seeing how the plot changes as current values for the factors change. The cells that change for a given factor are the cells that do not involve that factor directly.

Confidence Intervals Shows or hides confidence intervals in the Prediction Profiler plot. The intervals are drawn by bars for categorical factors, and curves for continuous factors. The interval values are also displayed on the vertical axis in blue. These are available when the profiler is used inside certain fitting platforms or when a standard error column of the form PredSE<colname> has been specified in the Y, Prediction Formula role of the Prediction Profiler launch dialog.

Prop of Error Bars (Appears when a Sigma column property exists in any of the factor and response variables.) Shows or hides the 3σ interval that is implied on the response due to the variation in the factor. The interval values are also displayed on the vertical axis in green. Propagation of error (POE) is important when attributing the variation of the response in terms of variation in the factor values when the factor values are not very controllable. See "Propagation of Error Bars".

Sensitivity Indicator Shows or hides a purple triangle whose height and direction correspond to the value of the partial derivative of the profile function at its current value. This is useful in large profiles to be able to quickly spot the sensitive cells.

Figure 3.5 Sensitivity Indicators



Profile at Boundary When analyzing a mixture design, JMP constrains the ranges of the factors so that settings outside the mixture constraints are not possible. This is why, in some mixture designs, the profile traces turn abruptly.

When there are mixture components that have constraints, other than the usual zero-to-one constraint, a new submenu, called Profile at Boundary, appears on the Prediction Profiler red triangle menu. It has the following two options:

Turn At Boundaries Lets the settings continue along the boundary of the restraint condition.

Stop At Boundaries Truncates the prediction traces to the region where strict proportionality is maintained.

JMP PRO **Extrapolation Control** Shows a submenu of options for extrapolation control. This feature helps identify possible extrapolated predictions. A prediction is considered an extrapolation when it is made using a combination of factor points that are not within the factor space of the original data. In the extrapolation control feature, the metric used to determine if a point is an extrapolation depends on the type of model fit. For models that are fit in the Standard Least Squares personality of the Fit Model platform, the leverage at the factor settings is used as the extrapolation metric. For all other models, the regularized Hotelling's T^2 value is used as the extrapolation metric. See "Extrapolation Control Metrics".

Extrapolation Control is available in profilers embedded in platforms and profilers launched from the Graph menu. The data used for the extrapolation control metrics depends on the type of profiler.

- When a model is built with validation, the embedded profiler and extrapolation control metrics are based on the training data.
- If you launch a profiler from the Graph menu the extrapolation control metrics are based on all data, unless you specifically exclude certain rows.
- When a model is built in a platform that ignores missing values during model fitting, those rows are excluded from the embedded profiler and extrapolation control metrics.
- When a model is built with Informative Missing, the embedded profiler and extrapolation control metrics reflect the informative missing.
- To include informative missing in the extrapolation control metrics when launching the profiler from the Graph menu use the Informative Missing column property.
- If you call Extrapolation Control from a profiler launched from the Graph menu, the regularized Hotelling's T^2 value is always used as the extrapolation metric, regardless of the type of model fit. Therefore, the extrapolation control results from a profiler embedded in the Standard Least Squares platform will not match those from the Graph menu profiler.

The extrapolation control red triangle menu includes options to either warn of possible extrapolation or to restrict the factor settings so that extrapolated predictions are not shown.

Off Turns off all extrapolation controls and warnings.

On Turns on extrapolation control. When this option is selected, it is indicated at the top of the profiler and the profile traces are restricted to factor combinations that do not lead to extrapolations.

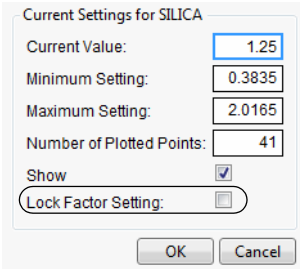
Warning On Turns on extrapolation warnings. When this option is selected, it is indicated at the top of the profiler. If a factor combination is selected that produces an extrapolation, an alert appears that reads --Possible Extrapolation--.

Extrapolation Details Shows or hides the extrapolation control details above the prediction profiler. The extrapolation control details include the value of the extrapolation metric at the current point, the value of the extrapolation threshold, the type of extrapolation metric, and the definition of the extrapolation threshold.

Set Threshold Criterion Opens a window that enables you to adjust the extrapolation threshold. When the extrapolation metric is the leverage at the factor settings, you can specify how the leverage is computed and the value of the corresponding multiplier. When the extrapolation metric is the regularized Hotelling's T^2 value, you can specify the multiplier. See "Extrapolation Control Metrics".

Reset Factor Grid Displays a window for each factor enabling you to enter a specific value for the factor's current setting, to lock that setting, and to control aspects of the grid. See the section "Set or Lock Factor Values".

Figure 3.6 Factor Settings Window



Factor Settings Submenu that consists of the following options:

Remember Settings Adds an outline node to the report that accumulates the values of the current settings each time the Remember Settings command is invoked. Each remembered setting is preceded by a radio button that is used to reset to those settings. There are options to remove selected settings or all settings in the Remember Settings red triangle menu.

Set To Data in Row Assigns the values of a data table row to the *X* variables in the Prediction Profiler.

Copy Settings Script Copies the current Prediction Profiler's settings to the clipboard.

Paste Settings Script Pastes the Prediction Profiler settings from the clipboard to a Prediction Profiler in another report.

Append Settings to Table Appends the current profiler's settings to the end of the data table. This is useful if you have a combination of settings in the Prediction Profiler that you want to add to an experiment in order to do another run.

Broadcast Factor Settings Sends the current profiler's factor settings to all other profilers, but does not link the profilers. A change in a factor in one profiler does not cause changes in any other profilers unless Broadcast Factor Settings is selected again.

Link Profilers Links all the profilers together. A change in a factor in one profiler causes that factor to change to that value in all other profilers, including Surface Plot. This is a global option, set, or unset for all profilers.

Set Script Sets a script that is called each time a factor changes. The set script receives a list of arguments of the form:

```
{factor1 = n1, factor2 = n2, ...}
```

For example, to write this list to the log, first define a function:

```
ProfileCallbackLog = Function({arg},show(arg));
```

Then enter `ProfileCallbackLog` in the Set Script dialog.

Similar functions convert the factor values to global values:

```
ProfileCallbackAssign = Function({arg},evalList(arg));
```

Or access the values one at a time:

```
ProfileCallbackAccess =
Function({arg},f1=arg["factor1"];f2=arg["factor2"]);
```

Unthreaded Enables you to change to an unthreaded analysis if multithreading does not work.

Animation Shows or hides animation controls that enable you to easily cycle through a variety of factor settings. See "Animation Controls".

Default N Levels Enables you to set the default number of levels for each continuous factor. This option is useful when the Prediction Profiler is especially large. When calculating the traces for the first time, JMP measures how long it takes. If this time is greater than three seconds, you are alerted that decreasing the Default N Levels speeds up the calculations.

Output Grid Table Produces a new data table with columns for the factors that contain grid values, columns for each of the responses with computed values at each grid point, and the desirability computation at each grid point.

If you have a lot of factors, it is impractical to use the Output Grid Table command, because it produces a large table. A memory allocation message might be displayed for large grid tables. In such cases, you should lock some of the factors, which are held at locked, constant values. To get the window to specify locked columns, ALT- or Option-click inside the profiler graph to get a window that has a Lock Factor Setting check box.

Output Random Table Prompts for a number of runs and creates an output table with that many rows, with random factor settings and predicted values over those settings. This is equivalent to (but much simpler than) opening the Simulator, resetting all the factors to a random uniform distribution, then simulating output. This command is similar to Output Grid Table, except it results in a random table rather than a sequenced one.

The prime reason to make uniform random factor tables is to explore the factor space in a multivariate way using graphical queries. This technique is called *Filtered Monte Carlo*.

Suppose you want to see the locus of all factor settings that produce a given range to desirable response settings. By selecting and hiding the points that do not qualify (using

graphical brushing or the Data Filter), you see the possibilities of what is left: the opportunity space yielding the result that you want.

Some rows might appear selected and marked with a red dot. These represent the points on the multivariate desirability Pareto Frontier - the points that are not dominated by other points with respect to the desirability of all the factors.

Alter Linear Constraints Enables you to add, change, or delete linear constraints. The constraints are incorporated into the operation of Prediction Profiler. See "Linear Constraints".

Save Linear Constraints Enables you to save existing linear constraints to a table script called Constraint. See "Linear Constraints".

Conditional Predictions Appears when random effects are included in the model. The random effects predictions are used in formulating the predicted value and profiles.

Appearance Submenu that consists of the following options:

Arrange in Rows Enter the number of plots that appear in a row. This option helps you view plots vertically rather than in one wide row.

Note: To set a default number of plots to appear in a row, go to File > Preferences > Platforms > Profiler and edit the Arrange in Rows preference.

Graph Spacing Opens a window that enables you to set the amount of horizontal space between graph panels.

Reorder X Variables Opens a window where you can reorder the model main effects by dragging them to the desired order.

Reorder Y Variables Opens a window where you can reorder the responses by dragging them to the desired order.

Adapt Y Axis Re-scales the vertical axis if the response is outside the axis range, so that the range of the response is included.

Show Creator Shows or hides the name of the platform that created the formula in the response column. The platform name appears on the vertical axis. (Available only if the response column contains a "Creator" named argument in the "Predicting" column property.)

Animation Controls

Figure 3.7 Animation Controls

Play/Pause Press play to animate the profiler. Moves through a cycle of factor settings and loops back to the beginning when the cycle is complete. Press pause to stop the animation.

Cycle Type Lists the types of cycles for factor settings.

Sequential Cycles through values for each factor, one factor after another. The name of the factor the animation is currently cycling through is displayed next to the speed slider bar.

Single Factor Cycles through values for the selected factor while all other factors are held constant. The name of the selected factor is displayed next to the speed slider bar.

Random Randomly cycles through different combinations of factor settings/values.

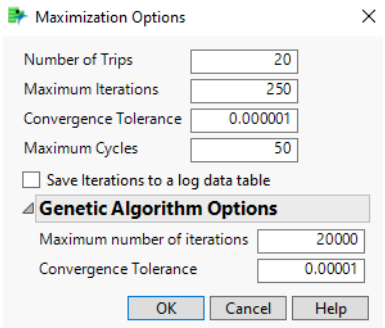
Data Sequential Sets the factor values to a row in the data table, one row at a time, starting with row 1. The row number of the current factor setting is displayed next to the speed slider bar.

Data Random Sets the factor values to a random row in the data table, one row at a time. The row number of the current factoring setting is displayed next to the speed slider bar.

Speed Slider Bar Use the slider bar to adjust the speed of the animation.

Maximization Options Window

Figure 3.8 Maximization Options Window



Number of Trips The number of random starts in the optimization algorithm. Each trip starts with a randomly chosen value for each of the factors in the model.

Maximum Iterations The maximum number of steps taken within each trip. If the algorithm does not converge before the maximum iterations is reached, a new trip is started.

Convergence Tolerance The upper limit for the convergence criterion for the optimization algorithm. If the convergence criterion is less than this value for two consecutive iterations, the algorithm stops.

Maximum Cycles (Not applicable if there are linear constraints.) The maximum number of cycles in the algorithm.

Genetic Algorithm Options (Available only if there is a disallowed combination constraint or if one of the extrapolation control features is turned on.) Provides additional options for the genetic algorithm. The genetic algorithm is used only if the solution from the regular optimization algorithm is not feasible.

Maximum number of iterations The maximum number of iterations for the genetic algorithm.

Convergence Tolerance The upper limit for the convergence criterion for the genetic algorithm.

Caution: The solution found by the genetic algorithm is not guaranteed to be the global optimum. This is true for all genetic algorithms. A genetic algorithm is needed for optimization problems that have more general constraints. For example, JMP uses the genetic algorithm when a disallowed combinations constraint is present because these types of constraints can involve categorical factors, be nonlinear, or be any general constraint entered via JSL.

Desirability Profiling and Optimization

You can define a desirability function for a single response variable or for several response variables. When you are optimizing relative to several responses, there can often be competing criteria. For example, you might want to maximize one response, minimize another, and keep a third response close to some target value.

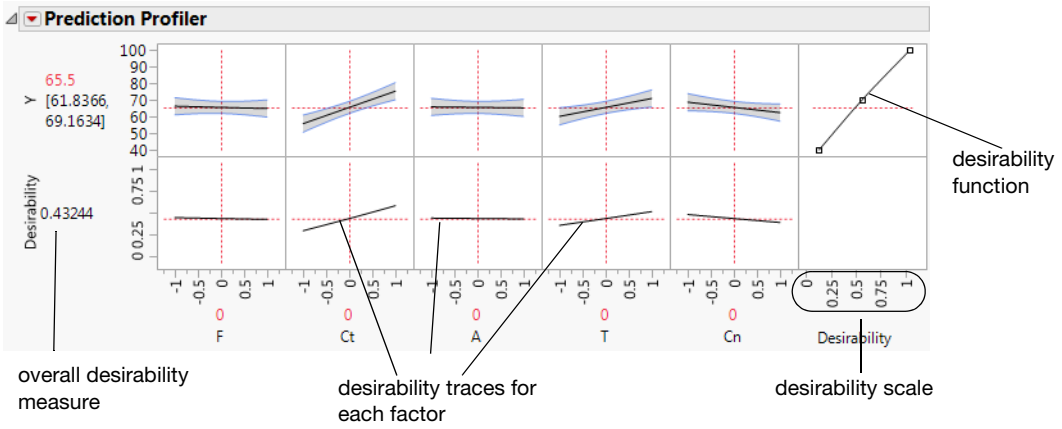
In desirability profiling, you specify a desirability function for each response. The overall desirability for all responses is defined as the geometric mean of the desirability functions for the individual responses. See Derringer and Suich (1980) for information about combining responses.

To use desirability profiling, select **Optimization and Desirability > Desirability Functions** from the Prediction Profiler red triangle menu.

Note: If the response column has a Response Limits property, desirability functions are turned on by default.

This command appends a new row to the bottom of the plot matrix, dedicated to graphing desirability. The row has a plot for each factor showing its *desirability trace*, as illustrated in Figure 3.9. It also adds a column that has an adjustable desirability function for each *Y* variable. The overall desirability measure shows on a scale of zero to one at the left of the row of desirability traces.

Figure 3.9 The Desirability Prediction Profiler



Construction of Desirability Functions

The individual desirability functions are smooth piecewise functions that pass through three defining points. These points are called control points (Low, Middle, High) and can be used to interactively control the shape of the desirability function.

- The Minimize and Maximize functions are three-part piecewise smooth functions that consist of interpolating cubics between the control points and exponentials in the tails.
- The Target function is a piecewise function that is a scale multiple of a normal density on either side of the Middle value (with different curves on each side), which is also piecewise smooth and fit to the control points. Exponential functions are fit to the tails.
- The None function enables you to specify an arbitrary desirability function. In particular, you can specify desirability to be lower at the Middle value than at the Low and High values. You can also construct custom desirability functions using formulas. See "Customized Desirability Functions".

The Low and High control points are not allowed to reach all the way to zero or one. This approach to constructing the desirability functions results in good behavior as the desirability values switch between the maximize, target, and minimize values.

Note: JMP does not use the Derringer and Suich (1980) functional forms. Because they are not smooth, they do not always work well with JMP's optimization algorithm.

Desirability Function for Multiple Optimization

When multiple responses are to be optimized, an overall desirability function is constructed and optimized. The overall desirability for all responses is defined as the geometric mean of the desirability functions for the individual responses.

Denote the individual desirability functions for k responses by $d_1, d_2,...,d_k$. Then the overall desirability function is the geometric mean of the individual desirability functions:

$$D = d_1^{1/k} d_2^{1/k} \ldots d_k^{1/k}$$

If Importance values are defined as part of the Response Limits column property or are defined in the Response Goal window, they are integrated into the overall desirability function. The Importance values are scaled so that they sum to 1. Denote the scaled importance values by $w_1, w_2,..., w_k$. Then the overall desirability is defined as a weighted geometric mean of the individual desirability functions:

$$D = d_1^{w_1} d_2^{w_2} \ldots d_k^{w_k}$$

Optimization Algorithm

The method used for optimization of the overall desirability function, or of the single desirability function if there is only one response, depends on the factor types.

- For categorical factors, a coordinate exchange algorithm is used.
- For continuous factors, a gradient descent algorithm is used.
- In the presence of constraints or mixture factors, a Wolfe reduced-gradient approach is used.
- To reduce the risk of finding local optima, JMP uses multiple random starts.

How to Use the Desirability Function

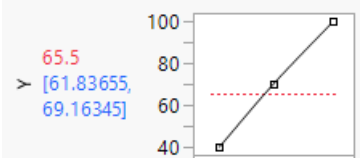
To use a variable's desirability function, drag the function handles to represent a response value.

As you drag these handles, the changing response value shows in the area labeled Desirability to the left of the plots. The dotted line is the response for the current factor settings. The overall desirability shows to the left of the row of desirability traces. Alternatively, you can select **Optimization and Desirability > Set Desirabilities** to enter specific values for the points.

Figure 3.10 shows steps to create desirability settings.

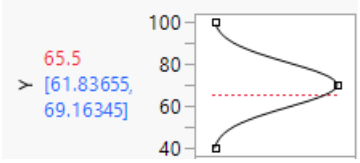
Maximize The default desirability function setting is maximize ("higher is better"). The High control point is positioned at the maximum Y value and aligned at the high desirability, close to 1. The Low control point is positioned at the minimum Y value and aligned at a low desirability, close to 0.

Figure 3.10 Maximizing Desirability



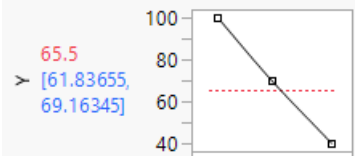
Target You can designate a target value as "best." In this example, the middle control point is positioned at Y = 70 and aligned with the maximum desirability of 1. Y becomes less desirable as its value approaches either 40 or 100. The High and Low control points at Y = 40 and Y = 100 are positioned at the minimum desirability close to 0.

Figure 3.11 Defining a Target Desirability



Minimize The minimize ("smaller is better") desirability function associates high response values with low desirability and low response values with high desirability. The curve is the maximization curve flipped around a horizontal line at the center of plot.

Figure 3.12 Minimizing Desirability



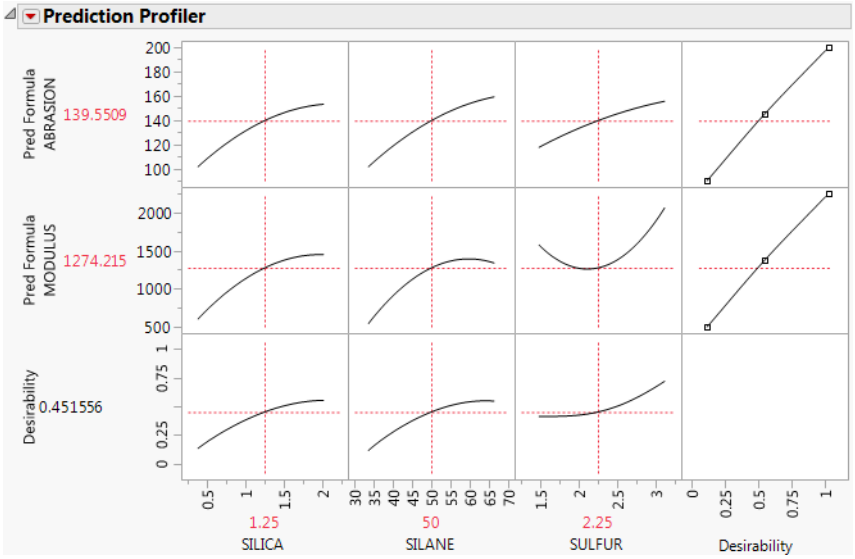
Note: Dragging the High or Low control point of a maximize or minimize desirability function across the *y*-value of the middle point results in the opposite point reflecting. A Minimize becomes a Maximize, and vice versa.

The Desirability Profile

The last row of plots shows the desirability trace for each factor. The numerical value beside the word Desirability on the vertical axis is the geometric mean of the desirability measures. This row of plots shows both the current desirability and the trace of desirabilities that result from changing one factor at a time.

For example, Figure 3.13 shows desirability functions for two responses. You want to maximize ABRASION and MODULUS. The desirability plots indicate that you could increase the desirability by increasing any of the factors.

Figure 3.13 Prediction Profile Plot with Adjusted Desirability and Factor Values



Customized Desirability Functions

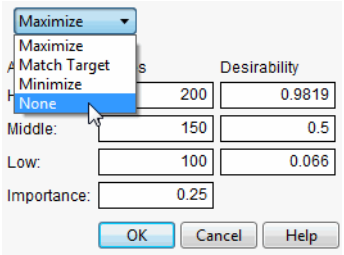
It is possible to use a customized desirability function. For example, suppose you want to maximize using the following function.

Figure 3.14 Maximizing Desirability Based on a Function

$$\frac{Pred\ Formula\ ABRASION}{96} + \frac{Pred\ Formula\ MODULUS}{700} + \frac{33}{(|Pred\ Formula\ ELONG - 450| + 1)} + \frac{2}{(|Pred\ Formula\ HARDNESS - 67| + 1)}$$

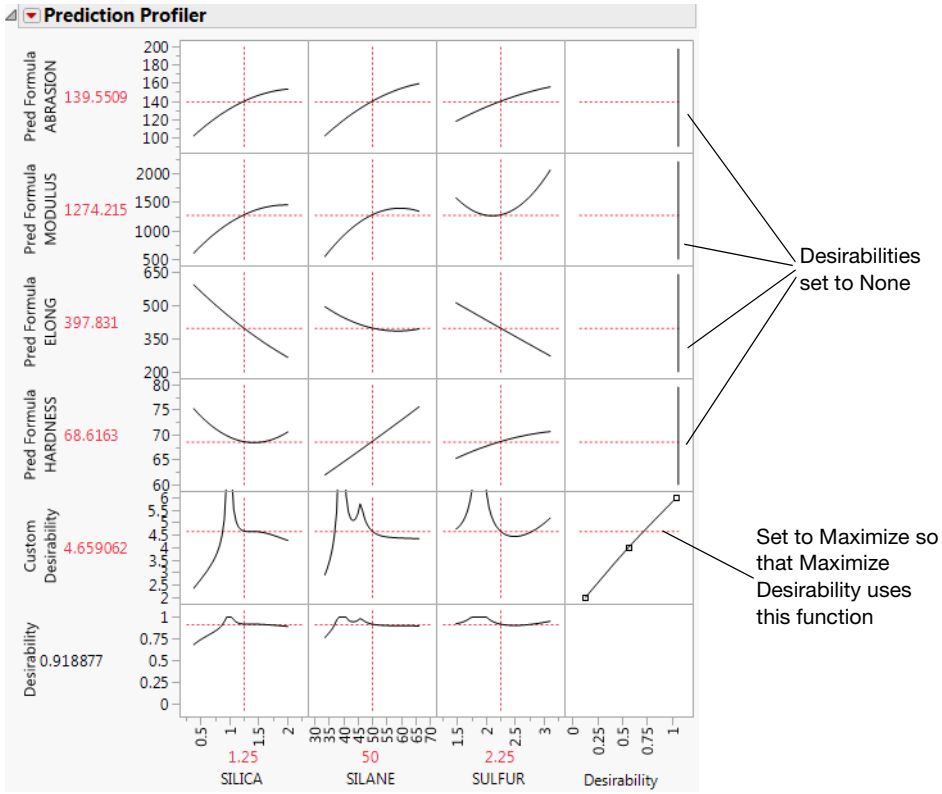
First, create a column called Custom Desirability that contains the above formula. Then, select **Graph > Profiler** to launch the platform. Select all the Pred Formula columns and the Custom Desirability column and select **Y, Prediction Formula**. Check the **Expand Intermediate Formula** option. Turn on the desirability functions by selecting **Optimization and Desirability > Desirability Functions** from the red triangle menu. All the desirability functions for the individual effects must be turned off. To do this, first double-click in a desirability plot window, and then select **None** in the window that appears. Set the desirability for Custom Desirability to be maximized.

Figure 3.15 Selecting No Desirability Goal



At this point, selecting **Optimization and Desirability > Maximize Desirability** uses only the custom Custom Desirability function.

Figure 3.16 Maximized Custom Desirability



Assess Variable Importance

The Variable Importance report calculates indices that measure the importance of factors in a model in a way that is independent of the model type and fitting method. The fitted model is used only in calculating predicted values. The method estimates the variability in the predicted response based on a range of variation for each factor. If variation in the factor causes high variability in the response, then that effect is important relative to the model.

Note: In some platforms, Assess Variable Importance is not available for categorical responses with more than two levels.

Assess Variable Importance can also be accessed in the Prediction Profiler that is obtained through the Graph menu.

For statistical details, see "Assess Variable Importance". See also Saltelli (2002).

Note: Assess Variable Importance requires that all columns reside in the same data table.

The Assess Variable Importance Report

The Assess Variable Importance menu has the following options that address the methodology used in constructing importance indices:

Independent Uniform Inputs For each factor, Monte Carlo samples are drawn from a uniform distribution defined by the minimum and maximum observed values. Use this option when you believe that your factors are uncorrelated and that their likely values are uniformly spread over the range represented in the study. This is the appropriate option for designed experiments that do not involve constraints or mixture factors.

Independent Resampled Inputs For each factor, Monte Carlo samples are obtained by resampling its set of observed values. Use this option when you believe that your factors are uncorrelated and that their likely values are not represented by a uniform distribution.

Dependent Resampled Inputs Factor values are constructed from observed combinations using a *k*-nearest neighbors approach, in order to account for correlation. This option treats observed variance and covariance as representative of the covariance structure for your factors. Use this option when you believe that your factors are correlated. Note that this option is sensitive to the number of rows in the data table. If used with a small number of rows, the results can be unreliable.

Note: The Independent Resampled Inputs and Dependent Resampled Inputs options are intended for observational studies. The Independent option is faster than the Dependent option, but the Dependent option handles multicollinearity better and does not extrapolate into regions far away from the data.

Linearly Constrained Inputs For each factor, Monte Carlo samples are drawn from a uniform distribution over a region defined by linear constraints. The linear constraints can be defined in the Prediction Profiler or constructed in connection with a designed experiment. In addition, the samples are restricted to fall within the minimum and maximum observed values. Use this option in the presence of linear constraints, when you believe that these constraints impact the distribution of the inputs.

The speed of these algorithms depends on the model evaluation speed. In general, the fastest option is Independent Uniform Inputs and the slowest is Dependent Resampled Inputs. You have the option to Accept Current Indices when the estimation process is unable to complete instantaneously.

Note: Variable importance indices are constructed using Monte Carlo sampling. For this reason, you can expect some variation in importance index values from one run to another.

Variable Importance Report

Each Assess Variable Importance option presents a Summary Report and Marginal Model Plots. When the Assess Variable Importance report opens, the factors in the Prediction Profiler are reordered according to their Total Effect importance indices. When there are multiple responses, the factors are reordered according to the Total Effect importance indices in the Overall report. When you run several Variable Importance reports, the factors in the Prediction Profiler are ordered according to their Total Effect indices in the most recent report.

Summary Report

For each response, a table displays the following elements:

Column The factor of interest.

Main Effect An importance index that reflects the relative contribution of that factor alone, not in combination with other factors.

Total Effect An importance index that reflects the relative contribution of that factor both alone and in combination with other factors. The Total Effect column is displayed as a bar chart. See "Weights".

Main Effect Std Error The Monte Carlo standard error of the Main Effect's importance index. This is a hidden column that you can access by right-clicking in the report and selecting **Columns > Main Effect Std Error**. By default, sampling continues until this error is less than 0.01. Details of the calculation are given in "Variable Importance Standard Errors". (Not available for Dependent Resampled Inputs option.)

Total Effect Std Error The Monte Carlo standard error of the Total Effect's importance index. This is a hidden column that you can access by right-clicking in the report and selecting **Columns > Total Effect Std Error**. By default, sampling continues until this error is less than 0.01. Details of the calculation are given in "Variable Importance Standard Errors". (Not available for Dependent Resampled Inputs option.)

Weights A plot that shows the Total Effect indices, located to the right of the final column. You can deselect or reselect this plot by right-clicking in the report and selecting **Columns > Weights**.

Proportion of function evaluations with missing values The proportion of Monte Carlo samples for which some combination of inputs results in an inestimable prediction. When the proportion is nonzero, this message appears as a note at the bottom of the table.

Note: When you have more than one response, the Summary Report presents an Overall table followed by tables for each response. The importance indices in the Overall report are the averages of the importance indices across all responses.

Marginal Model Plots

The Marginal Model Plots report (Figure 3.39) shows a matrix of plots, with a row for each response and columns for the factors. The factors are ordered according to the size of their overall Total Effect importance indices.

For a given response and factor, the plot shows the mean response for each factor value, where that mean is taken over all inputs to the calculation of importance indices. These plots differ from profiler plots, which show cross sections of the response. Marginal Model Plots are useful for assessing the main effects of factors.

Note that your choice of input methodology impacts the values plotted on marginal model plots. Also, because the plots are based on the generated input settings, the plotted mean responses might not follow a smooth curve.

The red triangle menu options enable you to show or hide the following aspects of the plots:

Estimate A smoothed estimate of the mean of the simulated values calculated as a function of the factor values.

> **Note:** The estimates of the mean are simulated, so the values change when you rerun the analysis.

Confidence Interval A 95% confidence band for the simulated means. This band is often narrow and might not be visible unless you expand the scale. Not available for Dependent Resampled Inputs.

> **Note:** The confidence bounds are simulated, so the bands change when you rerun the analysis.

Data The actual (unsimulated) values of the response plotted against the factor values.

Variable Importance Options

The Variable Importance red triangle menu contains the following options:

Reorder factors by main effect importance Reorders the cells in the Prediction Profiler in accordance with the importance indices for the main effects (Main Effect).

Reorder factors by total importance Reorders the cells in the Prediction Profiler in accordance with the total importance indices for the factors (Total Effect).

Colorize Profiler Colors cells in the profiler by Total Effect importance indices using a red to white intensity scale.

Note: You can click rows in the Summary Report to select columns in the data table. This can facilitate further analyses.

Bagging

Bootstrap aggregating (*bagging*) is a technique to improve predictive performance while also gaining insight into the reliability of predictions. Bagging is especially useful in unstable methods, including neural networks, classification trees, and regression trees.

Bagging creates *M* training data sets by sampling with replacement from the original data. All training data sets are of the same size as the original. For each training data set, a model is fit using the analysis platform, and predictions are made. Therefore, there are a total of *M* predictions for each observation in the original data set. The final prediction is the average of the *M* predictions.

Bagging is available in many analysis platforms. To use bagging, select **Save Bagged Predictions** from the Prediction Profiler red triangle menu. A window appears with the following options for Bagging:

Number of Bootstrap Samples Sets the number of times that you want to resample the data and build a model. A larger number results in more precise predictions. By default, the number of bootstrap samples is 100.

Random Seed Sets a random seed that you can re-enter in subsequent runs of the bagging analysis to duplicate your current results. By default, no seed is set.

Fractional Weights Performs a Bayesian bagging analysis. In each bootstrap iteration, each observation is assigned a nonzero weight. The model that makes the predictions uses the weighted observations. By default, the Fractional Weights option is not selected, and a simple bagging analysis is conducted.

Tip: Use the Fractional Weights option if the number of observations that are used in your analysis is small or if you are concerned about separation in a logistic regression setting.

Suppose that Fractional Weights is selected. For each bootstrap iteration, each observation that is used in the report is assigned a nonzero weight. These weights sum to *n,* the number of observations used in the model. For more information about how the weights are calculated and used, see *Basic Analysis*.

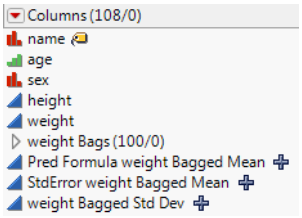
Save Prediction Formulas For each bagged prediction, this option saves the formula used to make that prediction in the column properties. This option is available in only a subset of the analysis platforms that offer bagging.

Note: If Save Prediction Formulas is not available, a note appears, stating that only the predicted values will be saved.

Bagging automatically creates new columns in the original data table. All M sets of bagged predictions are saved as hidden columns. The final prediction is saved in a column named "Pred Formula <colname> Bagged Mean". The standard deviation of the final prediction is saved in a column named "<colname> Bagged Std Dev". The standard error of the bagged mean is saved in a column named "StdError <colname> Bagged Mean." The standard error is the standard deviation divided by $\sqrt{M-1}$. Here, <colname> identifies the column in the report that was bagged.

The standard error gives insight about the precision of the prediction. A very small standard error indicates a precise prediction for that observation. For more information about bagging, see Hastie et al. (2009).

Figure 3.17 Bagging Columns



Additional Examples of the Prediction Profiler

This section contains additional examples illustrating various aspects of the Prediction Profiler platform and the Prediction Profiler embedded in many analysis platforms.

- "Example of Desirability Profiling for Multiple Responses"
- "Example of a Noise Factor in the Prediction Profiler"
- "Example of Variable Importance for One Response"
- "Example of Variable Importance for Multiple Responses"
- "Example of Bagging to Improve Prediction"
- "Example of Bagging to Indicate the Accuracy of Predictions"
- "Example of Confidence Intervals Using Bagging"
- "Example of Confidence Intervals Using StdErr Pred Formula Column"

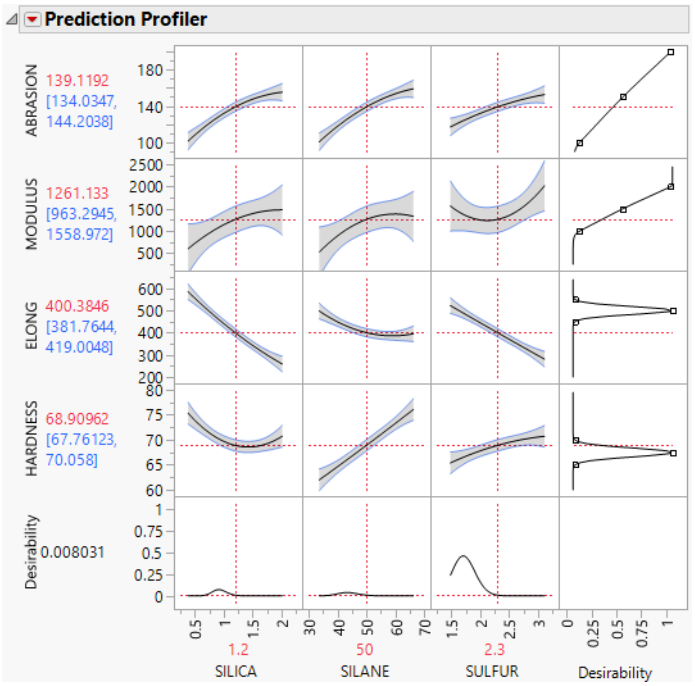
Example of Desirability Profiling for Multiple Responses

A desirability index becomes especially useful when there are multiple responses. The idea was pioneered by Derringer and Suich (1980), who give the following example. Suppose there are four responses, ABRASION, MODULUS, ELONG, and HARDNESS. Three factors, SILICA, SILANE, and SULFUR, were used in a central composite design.

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Click the green triangle next to RSM for 4 Responses to run the script.

   This script defines a model for the four responses with a full quadratic response surface. The summary tables and effect information appear for all the responses, followed by the prediction profiler shown in Figure 3.18.

Figure 3.18 Prediction Profiler for Multiple Responses before Optimization

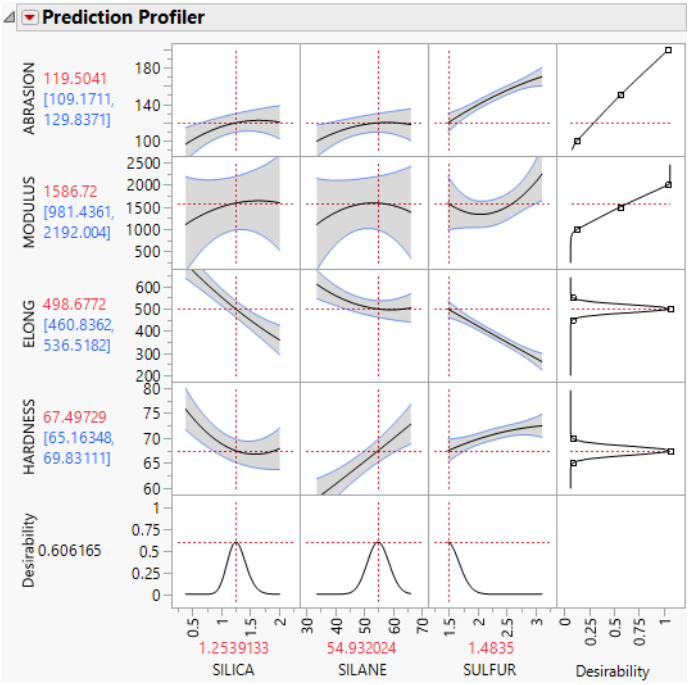


Note the following about the desirability functions:

- Maximum ABRASION and maximum MODULUS are most desirable.
- ELONG target of 500 is most desirable.
- HARDNESS target of 67.5 is most desirable.

3. Select **Optimization and Desirability > Maximize Desirability** from the Prediction Profiler red triangle menu to maximize desirability.

Figure 3.19 Prediction Profiler after Optimization



The desirability traces at the bottom decrease everywhere except the current values of the effects, which indicates that any further adjustment could decrease the overall desirability.

Example of a Noise Factor in the Prediction Profiler

This example uses the Tiretread.jmp sample data set. This data set shows the results of a tire manufacturer's experiment whose objective is to match a target value of HARDNESS= 70 based on three factors: SILICA, SILANE, and SULFUR content. Suppose the SILANE and SULFUR content are easily (and precisely) controllable, but SILICA is not easily controlled.

For comparison, first optimize the factors for hardness without considering variation from the noise factor, SILICA.

1. Select **Graph > Profiler** to launch the Prediction Profiler.
2. Assign Pred Formula HARDNESS to the **Y, Prediction Formula** role.
3. Click **OK**.
4. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Desirability Functions**.
5. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Set Desirabilities** to open the Response Goal window**.**

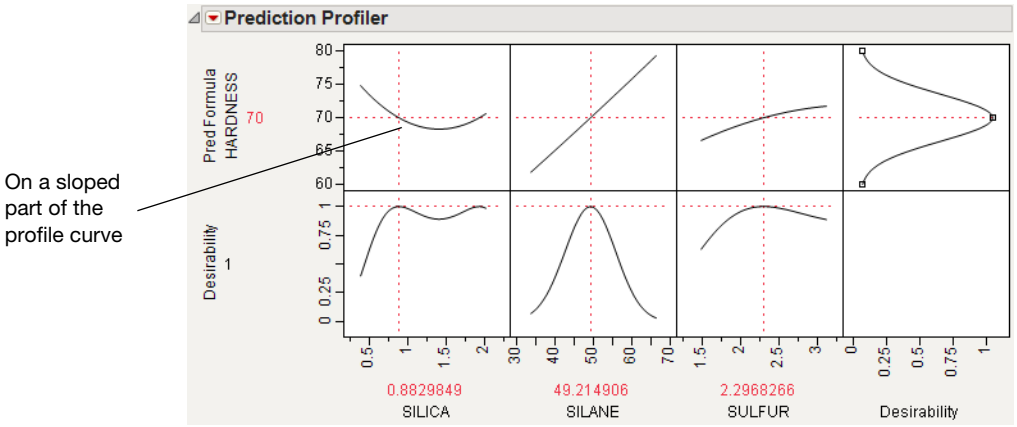
Tip: Alternatively, you can double-click in the Desirability plot to open the Response Goal window.

6. Select **Match Target** from the list and click **OK**.
7. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability** to find the optimum factor settings for the target value of HARDNESS.

   We get the following Prediction Profiler display. Notice that the SILICA factor's optimum value is on a sloped part of a profile curve. This means that variations in SILICA settings impact the response, HARDNESS.

Note: You might get different results from these because different combinations of factor values can all hit the target.

Figure 3.20 Maximizing Desirability for HARDNESS



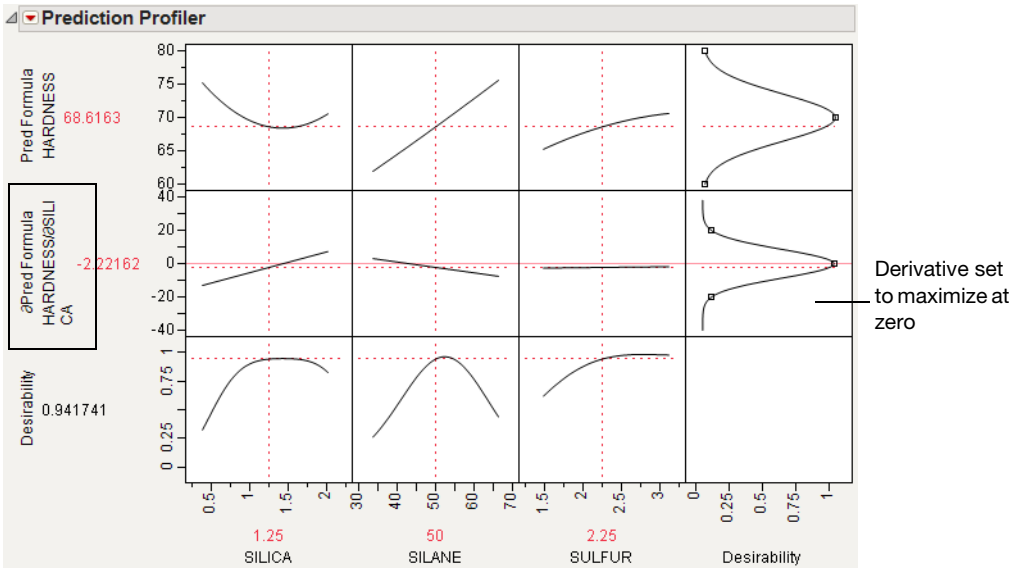
Now, we would like to not just optimize for a specific target value of HARDNESS, but also find a setting that is insensitive to the variability in the setting of Silica. Repeat the optimization process with SILICA as a noise factor.

1. Select **Graph > Profiler**.
2. Select Pred Formula HARDNESS and click **Y, Prediction Formula**.
3. Select SILICA and click **Noise Factors**.
4. Click **OK**.
5. Select **Optimization and Desirability > Desirability Functions** in the Prediction Profiler menu.

6. Double-click in the Desirability plot for the response Pred Formula HARDNESS to open the Response Goal window.
7. Select **Match Target** from the list and click **OK**.

   The resulting profiler has the appropriate derivative of the fitted model with respect to the noise factor, set to be maximized at zero, its flattest point.

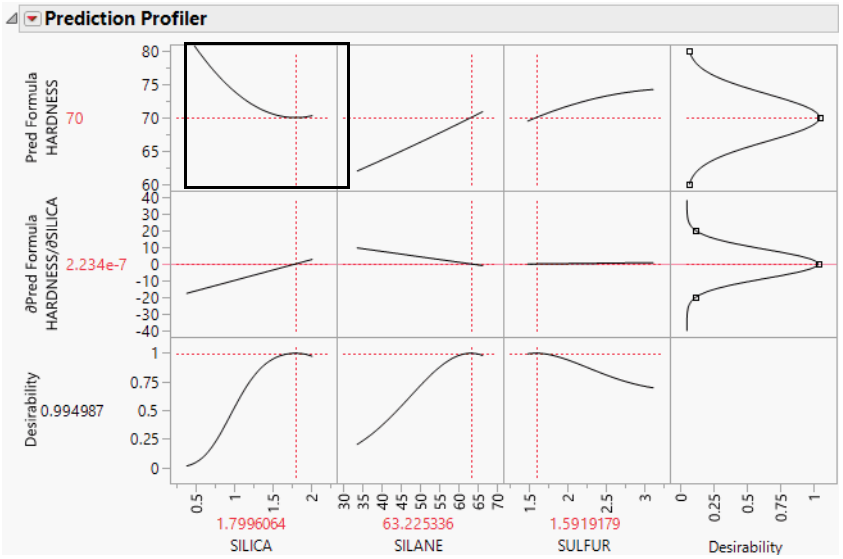
Figure 3.21 Derivative of the Prediction Formula with Respect to Silica



8. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability** to find the optimum values for the process factor, balancing for the noise factor.

   This time, we have also hit the targeted value of HARDNESS, but our value of SILICA is on its flatter region. This means variation in SILICA does not transmit as much variation to HARDNESS as when it was set at a value on the steep part of the SILICA curve.

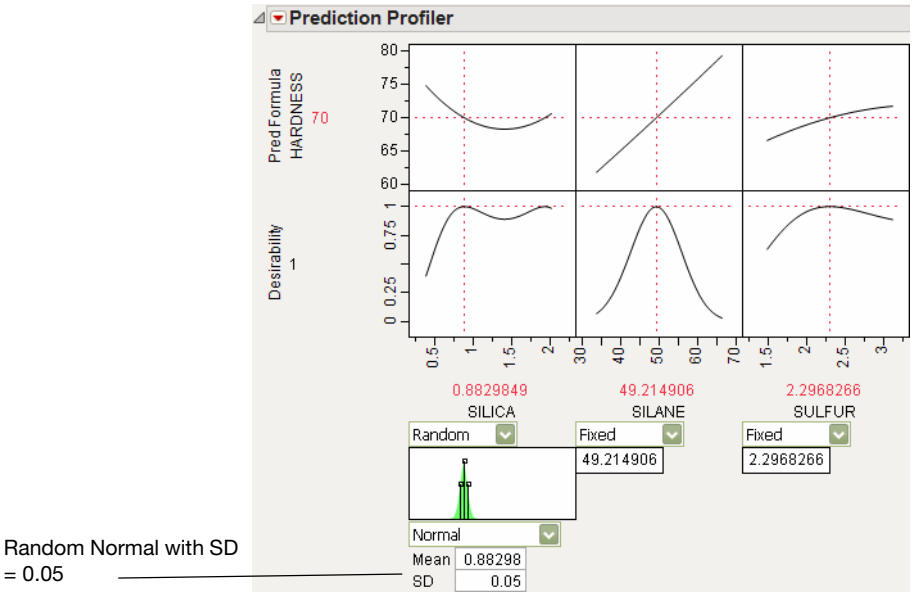
Figure 3.22 Maximize Desirability



You can see the effect of accounting for the SILICA variation on the variance of the predictions by following these steps for each profiler (one without the noise factor, and one with the noise factor):

1. In the Profiler without the noise factor, click the Prediction Profiler red triangle and select **Simulator**.
2. Assign SILICA to have a random Normal distribution with a standard deviation of 0.05. Use the default setting for the mean.

Figure 3.23 Setting a Random Normal Distribution



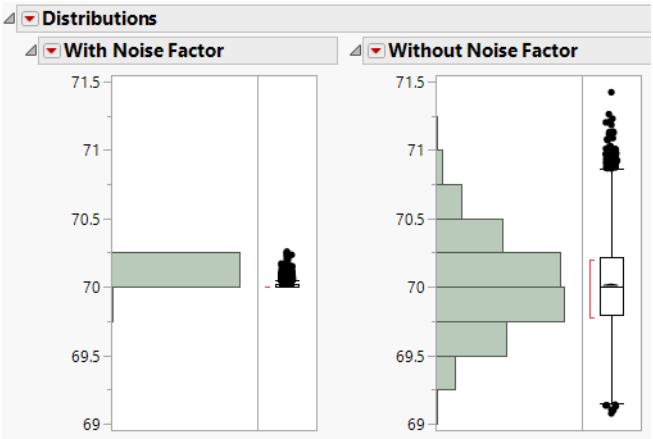
3. Click **Simulate**.
4. Click the Simulate to Table gray triangle and then click the **Make Table** button.
5. Rename the Pred Formula HARDNESS column to Without Noise Factor.
6. In the Profiler with the noise factor, click the Prediction Profiler red triangle and select **Simulator**.
7. Assign SILICA to have a random Normal distribution with a standard deviation of 0.05. Use the default setting for the mean.
8. Click **Simulate**.
9. Click the Simulate to Table gray triangle and then click the **Make Table** button.

Completing these steps for both the original and noise-factor-optimal simulations results in two similar data tables, one for each simulation. We want to compare the distributions of the predicted hardness from the two simulations. To facilitate the comparison we put the two prediction columns in a single data table.

1. Select the Pred Formula HARDNESS column in the current simulation table and select **Edit > Copy With Full Precision**.
2. In the other current simulation table, select **Cols > New Columns**.
3. Type With Noise Factor next to Column Name and click **OK**.
4. Select the With Noise Factor column and select **Edit > Paste**.
5. Select **Analyze > Distribution**.

6. Select With Noise Factor and Without Noise Factor and click **Y, Columns**.
7. Click **OK**.
8. Click the Distribution red triangle and select **Uniform Scaling**.

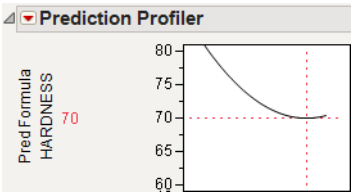
Figure 3.24 Comparison of Distributions with and without Noise Factors



The histograms show that there is much more variation in predicted Hardness when the noise factor was not included in the analysis.

It is also interesting to note the shape of the histogram when the noise factor was included. In the comparison histograms above, note that the With Noise Factor distribution has data trailing off in only one direction.The skewness in the With Noise Factor is clear if you deselect Uniform Scaling from the Distributions red triangle menu. The predictions are skewed because Hardness is at a minimum with respect to SILICA, as shown in Figure 3.25. Therefore, variation in SILICA can make only HARDNESS increase. When the non-robust solution is used, the variation could be transmitted either way.

Figure 3.25 Prediction Profiler Showing the Minima of HARDNESS by SILICA



Example of Variable Importance for One Response

The Boston Housing.jmp sample data table contains data on 13 factors that might relate to median home values. You fit a model using a neural network. Because neural networks do not accommodate formal hypothesis tests, these tests are not available to help assess which variables are important in predicting the response. However, for this purpose, you can use the Assess Variable Importance profiler option.

Note that your results might differ from, but should resemble, those shown here. There are two sources of random variability in this example. When you fit the neural network, *k*-fold cross validation is used. This partitions the data into training and validation sets at random. Also, Monte Carlo sampling is used to calculate the factor importance indices.

1. Select **Help > Sample Data Library** and open Boston Housing.jmp.
2. Select **Analyze > Predictive Modeling > Neural**.
3. Select mvalue from the Select Columns list and click **Y, Response**.
4. Select all other columns from the Select Columns list and click **X, Factor**.
5. Click **OK**.
6. In the Neural Model Launch panel, select **KFold** from the list under Validation Method.

   When you select KFold, the Number of Folds defaults to 5.
7. (Optional) Enter 123 next to Random Seed.

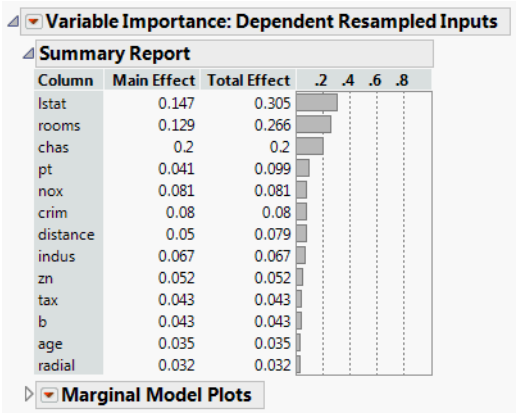
   Note: Results vary due to the random nature of choosing a validation set in the Neural Network model. Entering the seed above enables you to reproduce the results shown in this example.
8. Click **Go**.
9. Click the red triangle next to Model NTanH(3) and select **Profiler**.

   The Prediction Profiler is displayed at the very bottom of the report. Note the order of the factors for later comparison.

   Because the factors are correlated, you take this into account by choosing Dependent Resampled Inputs as the sampling method for assessing variable importance.
10. Click the Prediction Profiler red triangle and select **Assess Variable Importance > Dependent Resampled Inputs**.

    The Variable Importance: Dependent Resampled Inputs report appears. Check that the Prediction Profiler cells have been reordered by the magnitude of the Total Effect indices in the report. In Figure 3.26, check that the Total Effect importance indices identify rooms and lstat as the factors that have most impact on the predicted response.

Figure 3.26 Dependent Resampled Inputs Report

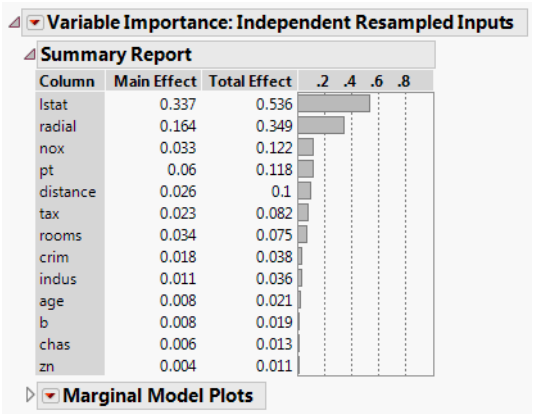


You might be interested in comparing the importance indices obtained assuming that the factors are correlated, with those obtained when the factors are assumed independent.

11. Click the Prediction Profiler red triangle and select **Assess Variable Importance > Independent Resampled Inputs**.

    The resampled inputs option makes sense in this example, because the distributions involved are not uniform. The Variable Importance: Independent Resampled Inputs report is shown in Figure 3.27. Check that the two factors identified as having the most impact on the predicted values are lstat and radial.

Figure 3.27 Independent Resampled Inputs Report



Example of Variable Importance for Multiple Responses

The data in the Tiretread.jmp sample data table are the result of a designed experiment where the factors are orthogonal. For this reason, you use importance estimates based on independent inputs. Suppose that you believe that, in practice, factor values vary throughout the design space, rather than assume only the settings defined in the experiment. Then you should choose Independent Uniform Inputs as the sampling scheme for your importance indices.

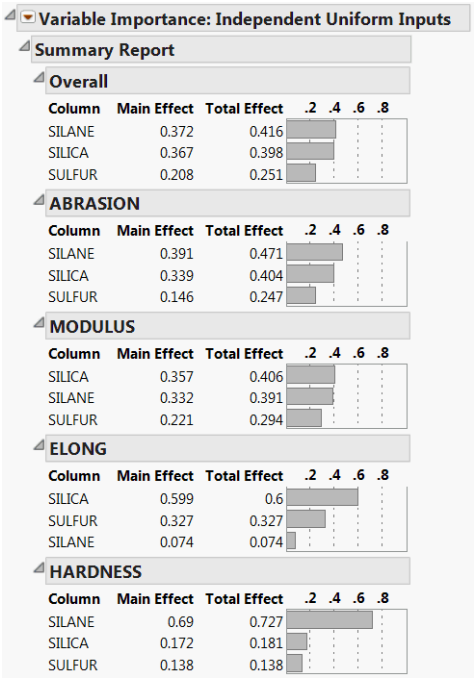
1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Run the script RSM for 4 Responses.

   The Prediction Profiler is displayed at the very bottom of the report.
3. Click the Prediction Profiler red triangle and select **Assess Variable Importance > Independent Uniform Inputs**.

   The Summary Report is shown in Figure 3.28. Because the importance indices are based on random sampling, your estimates might differ slightly from those shown in the figure.

   The report shows tables for each of the four responses. The Overall table averages the factor importance indices across responses. The factors in the Prediction Profiler (Figure 3.29) have been reordered to match their ordering on the Overall table's Total Effect importance.

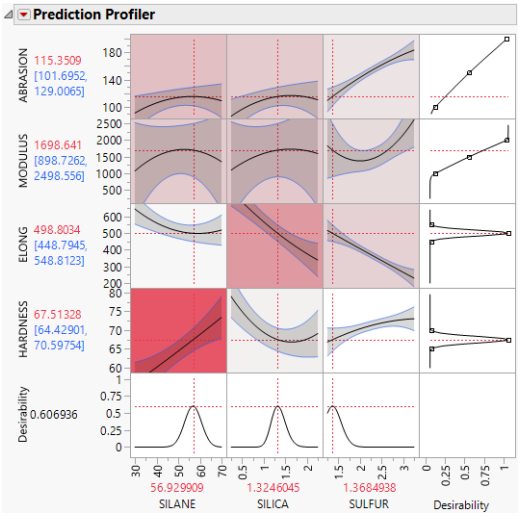
Figure 3.28 Summary Report for Four Responses



4. Click the red triangle next to Variable Importance: Independent Uniform Inputs and select **Colorize Profiler**.

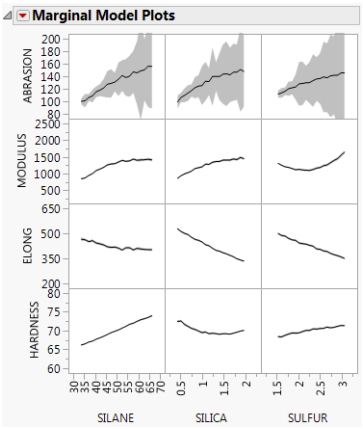
   Colors from a red to white intensity scale are overlaid on profiler panels to reflect Total Effect importance. For example, you easily see that the most important effect is that of Silane on Hardness.

Figure 3.29 Prediction Profiler for Four Responses



The Marginal Model Plots report shows mean responses for each factor across a uniform distribution of settings for the other two factors.

Figure 3.30 Marginal Model Plots for Four Responses



Example of Bagging to Improve Prediction

Bagging is used in a number of situations. One situation is to improve predictive power. Bagging is especially helpful for unstable models. This example uses the Tiretread.jmp sample data table. There are three factors (SILICA, SILANE, and SULFUR) and four responses (ABRASION, MODULUS, ELONG, and HARDNESS). First, you fit a neural network model to simultaneously predict the four response variables as a function of the three factors. Then, you perform bagging on the neural network model. Last, you compare the predictions to show the improvements obtained through bagging.

Fit Neural Network Model

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Analyze > Predictive Modeling > Neural**.
3. Select ABRASION, MODULUS, ELONG, and HARDNESS and click **Y, Response**.
4. Select SILICA, SILANE, and SULFUR and click **X, Factor**.
5. Click **OK**.
6. (Optional) Enter 2121 next to Random Seed.

   Note: Results vary due to the random nature of choosing a validation set in the Neural Network model. Entering the seed above enables you to reproduce the results shown in this example.

7. Click **Go**.
8. Click the red triangle next to Model NTanH(3) and select **Save Formulas**.

   Note: This option saves the predicted values for all response variables from the neural network model to the data table. Later, these values are compared to the predictions that are obtained from bagging.

Perform Bagging

Now that the initial model has been constructed, you can perform bagging using that model. Access the Bagging feature through the Prediction Profiler.

1. Click the red triangle next to Model NTanH(3) and select **Profiler**.

   The Prediction Profiler appears at the bottom of the report.

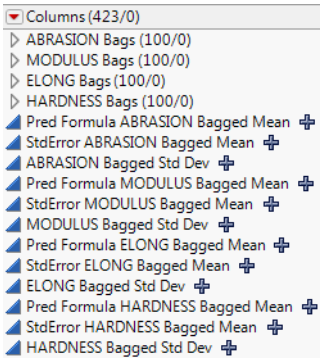
2. Click the Prediction Profiler red triangle and select **Save Bagged Predictions**.
3. Enter 100 next to Number of Bootstrap Samples.
4. (Optional) Enter 2121 next to Random Seed.

Note: Results vary due to the random nature of sampling with replacement. To reproduce the exact results in this example, set the Random Seed.

5. Click **OK**.

   Return to the data table. For each response variable, there are three new columns denoted as Pred Formula <colname> Bagged Mean, StdError <colname> Bagged Mean, <colname> Bagged Std Dev. The Pred Formula <colname> Bagged Mean columns are the final predictions.

Figure 3.31 Columns Added to Data Table After Bagging



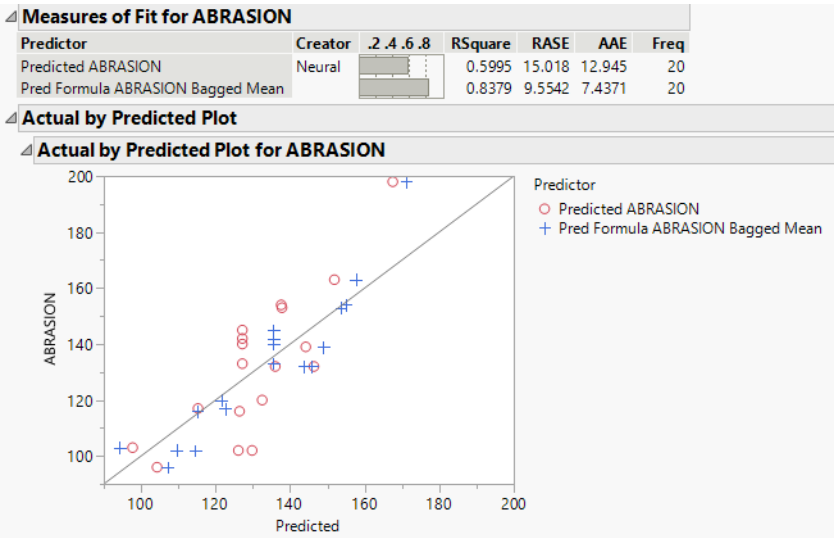
Compare the Predictions

To see how bagging improves predictive power, compare the predictions from the bagged model to the original model predictions. Use the Model Comparison platform to look at one response variable at a time.

1. Select **Analyze > Predictive Modeling > Model Comparison**.
2. Select Predicted ABRASION and click **Y, Predictors**.
3. Select Pred Formula ABRASION Bagged Mean and click **Y, Predictors.**
4. Click **OK**.

   A window that contains a list of columns appears.
5. Select ABRASION and click **OK**.
6. Click the Model Comparison red triangle and select **Plot Actual by Predicted**.

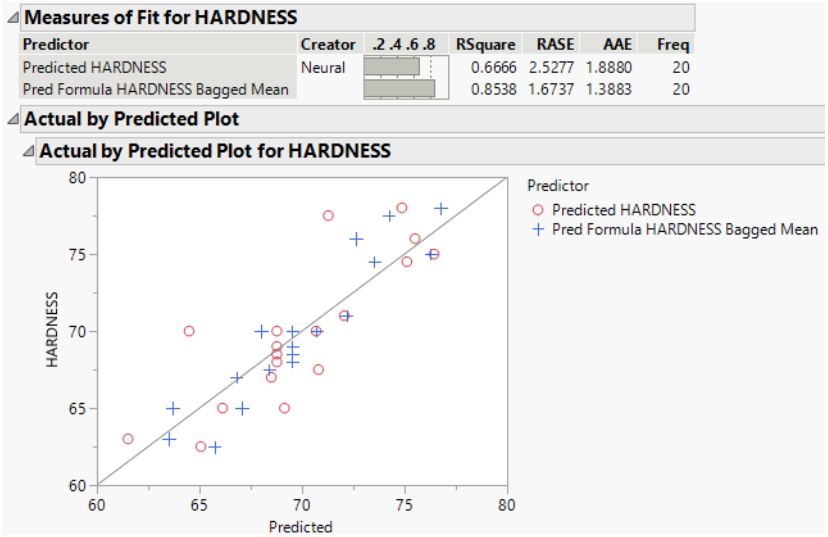
Figure 3.32 Comparison of Predictions for ABRASION



The Measures of Fit report and the Actual by Predicted Plot are shown in Figure 3.32. The predictions that were obtained from bagging are shown in blue. The predictions that were obtained from the original neural network model are shown in red. In general, the bagging predictions are closer to the line than the original model predictions. Because the bagging predictions are closer to the line, the RSquare value of 0.8379 for the bagged predictions is higher than the RSquare value for the original model predictions. You conclude that bagging improves predictions for ABRASION.

This example compared the predictions for ABRASION. To compare predictions for another response variable, follow step 2 through step 6, replacing ABRASION with the desired response variable. As another example, Figure 3.33 shows the Measures of Fit report for HARDNESS. The report shows similar findings as the Measures of Fit report for ABRASION. The RSquare value for the bagged predictions is slightly higher than the RSquare value for the original model predictions, which indicates a better fit and improved predictions.

Figure 3.33 Comparison of Predictions for HARDNESS



Example of Bagging to Indicate the Accuracy of Predictions

Bagging is also used to indicate the accuracy of the prediction through standard errors and other distributional measures. In platforms where the Save Predicted Formulas option is available in Bagging, you can make predictions on new observations and determine how accurate they are. The Save Predicted Formulas option is available in the Standard Least Squares, Generalized Regression, and Generalized Linear Models platforms.

In the Tiretread.jmp data table, suppose that you are interested in only predicting ABRASION as a function of the three factor variables. In this example, you fit a generalized regression model to predict ABRASION. Then, you perform bagging on that model. Last, you make a prediction for a new observation and investigate the accuracy of that prediction. This is done by obtaining a confidence interval for the prediction.

Fit a Generalized Regression Model

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Analyze > Fit Model**.
3. Select ABRASION and click **Y**.
4. Select **Generalized Regression** from the Personality list.
5. Select SILICA, SILANE, and SULFUR and click **Macros > Full Factorial**.

   This adds all terms, including interactions, to the model.
6. Click **Run**.

7. Click **Go**.

Figure 3.34 Parameter Estimates From Generalized Regression Report

Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	-46.05865	7.526913	37.44455	<.0001*	-60.81113	-31.30617
SILICA	32.987289	2.2710897	210.97194	<.0001*	28.536036	37.438543
SILANE	1.7880765	0.1194432	224.10381	<.0001*	1.5539721	2.0221809
(SILICA-1.2)*(SILANE-50)	1.025	0.2316574	19.577413	<.0001*	0.5709598	1.4790402
SULFUR	21.813077	1.4913761	213.92398	<.0001*	18.890033	24.73612
(SILICA-1.2)*(SULFUR-2.3)	28.5	4.6331484	37.838796	<.0001*	19.419196	37.580804
(SILANE-50)*(SULFUR-2.3)	1.575	0.2316574	46.224125	<.0001*	1.1209598	2.0290402
(SILICA-1.2)*(SILANE-50)*(SULFUR-2.3)	1.55	0.4633148	11.192085	0.0008*	0.6419196	2.4580804

Normal Distribution Parameters	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Scale	5.2681724	0.7759427	46.095727	<.0001*	3.7473527	6.7889921

Perform Bagging

1. Click the red triangle next to Normal Lasso with AICc Validation and select **Profilers > Profiler**.

   The Prediction Profiler appears at the bottom of the report.
2. Click the Prediction Profiler red triangle and select **Save Bagged Predictions**.
3. Enter 500 next to Number of Bootstrap Samples.
4. (Optional) Enter 4321 next to Random Seed.

   Note: Results vary due to the random nature of sampling with replacement. To reproduce the exact results in this example, set the Random Seed.
5. Confirm that **Save Prediction Formulas** is selected.
6. Click **OK**.

   Note: This might take longer to run than the "Example of Bagging to Improve Prediction". The larger number of samples gives a better estimate of the prediction distributions.

   Return to the data table. For each response variable, there are three new columns denoted as Pred Formula <colname> Bagged Mean, StdError <colname> Bagged Mean, and <colname> Bagged Std Dev. The Pred Formula ABRASION Bagged Mean column is the final prediction.

Prediction for a New Observation

You now have predictions for ABRASION for each observation in the data table, as well as the standard errors for those predictions. Suppose that you have an observation with new values of 0.9, 43, and 2 for SILICA, SILANE, and SULFUR, respectively. You can predict the ABRASION response and obtain a confidence interval for that prediction because the Save Prediction Formulas option saves the regression equation for each bagged model. Therefore, *M* predictions are made with the new factor values to create a distribution of possible predictions. The mean is the final prediction, but analyzing the distribution tells you how accurate the prediction is.

1. In the data table, select **Rows > Add Rows**.
2. Enter 1 in the **How many rows to add** box and click **OK**.
3. Under the SILICA column, type 0.9 in the box for the new row.
4. Under the SILANE column, type 43 in the box for the new row.
5. Under the SULFUR column, type 2 in the box for the new row.

   All of the prediction columns for the new row are automatically calculated.

Figure 3.35 Values for New Row

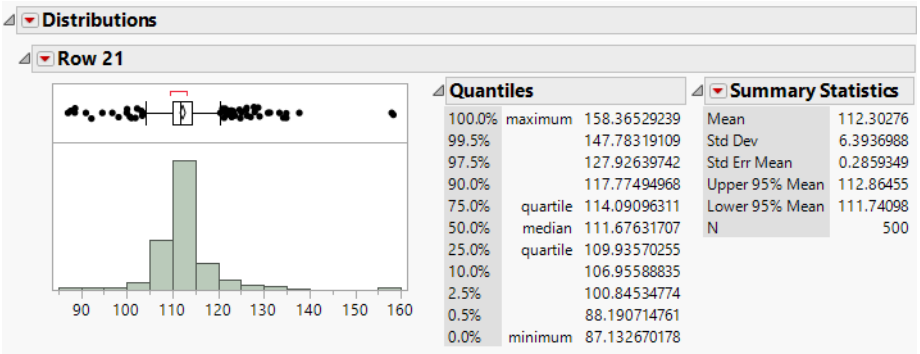
	ABRASION	MODULUS	ELONG	HARDNESS	SILICA	SILANE	SULFUR	Pred Formula ABRASION	Pred Formula MODULUS
1	102	900	470	67.5	0.7	40	2.8	96.7475395	747.638719
2	120	860	410	65	1.7	40	1.8	113.421752	1074.97186
3	117	800	570	77.5	0.7	60	1.8	114.695993	1011.176
4	198	2294	240	74.5	1.7	60	2.8	195.496359	2173.94727
5	103	490	640	62.5	0.7	40	1.8	104.934463	865.669653
6	132	1289	270	67	1.7	40	2.8	133.734829	1333.44092
7	132	1270	410	78	0.7	60	2.8	138.00907	1310.64507
8	139	1090	380	70	1.7	60	1.8	143.683282	1497.9782
9	102	770	590	76	0.3835	50	2.3	101.492758	600.398164
10	154	1690	260	70	2.0165	50	2.3	155.361002	1476.17966
11	96	700	520	63	1.2	33.67	2.3	100.727591	525.749229
12	163	1540	380	75	1.2	66.33	2.3	159.12617	1330.8286
13	116	2184	520	65	1.2	50	1.4835	117.116503	1564.51067
14	153	1784	290	71	1.2	50	3.1165	152.737257	2020.06715
15	133	1300	380	70	1.2	50	2.3	139.119239	1261.13314
16	133	1300	380	68.5	1.2	50	2.3	139.119239	1261.13314
17	140	1145	430	68	1.2	50	2.3	139.119239	1261.13314
18	142	1090	430	68	1.2	50	2.3	139.119239	1261.13314
19	145	1260	390	69	1.2	50	2.3	139.119239	1261.13314
20	142	1344	390	70	1.2	50	2.3	139.119239	1261.13314
21	•	•	•	•	0.9	43	2	114.489091	931.321632

6. Select **Tables > Transpose**.
7. Select ABRASION Bags (500/0) and click **Transpose Columns**.
8. Click **OK**.
9. Select **Analyze > Distribution**.
10. Select Row 21 and click **Y, Columns**.

Note: Row 21 corresponds to the predictions from the new observation.

11. Click **OK**.
12. Click the red triangle next to Row 21 and select **Display Options > Horizontal Layout**.

Figure 3.36 Distribution Report



Quantiles

100.0%	maximum	158.36529239
99.5%		147.78319109
97.5%		127.92639742
90.0%		117.77494968
75.0%	quartile	114.09096311
50.0%	median	111.67631707
25.0%	quartile	109.93570255
10.0%		106.95588835
2.5%		100.84534774
0.5%		88.190714761
0.0%	minimum	87.132670178

Summary Statistics

Mean	112.30276
Std Dev	6.3936988
Std Err Mean	0.2859349
Upper 95% Mean	112.86455
Lower 95% Mean	111.74098
N	500

The Distribution Report in Figure 3.36 contains information about the distribution of the predicted values of ABRASION from each bagged model. The final prediction of ABRASION for the new observation is 112.3, which is the mean of all the *M* bagged predictions. This prediction has a standard error of 6.39. You can also create confidence intervals for the new prediction using the quantiles. For example, a 95% confidence interval for the new prediction is 100.85 to 127.93.

Example of Confidence Intervals Using Bagging

Use the bagged standard errors to obtain confidence intervals around predictions for a Neural Network model.

Fit Neural Network

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Analyze > Predictive Modeling > Neural**.
3. Select ABRASION and click **Y, Response**.
4. Select SILICA, SILANE, and SULFUR and click **X, Factor**.
5. Click **OK**.
6. (Optional) Enter 123456 next to Random Seed.

Note: Results vary due to the random nature of choosing a validation set in the Neural Network model. To reproduce the results shown in this example, set the Random Seed.

7. Click **Go**.

Perform Bagging and Launch the Profiler

1. Click the red triangle next to Model NTanH(3) and select **Profiler**.
2. Click the Prediction Profiler red triangle and select **Save Bagged Predictions**.
3. Enter 100 next to Number of Bootstrap Samples.
4. (Optional) Enter 123456 next to Random Seed.

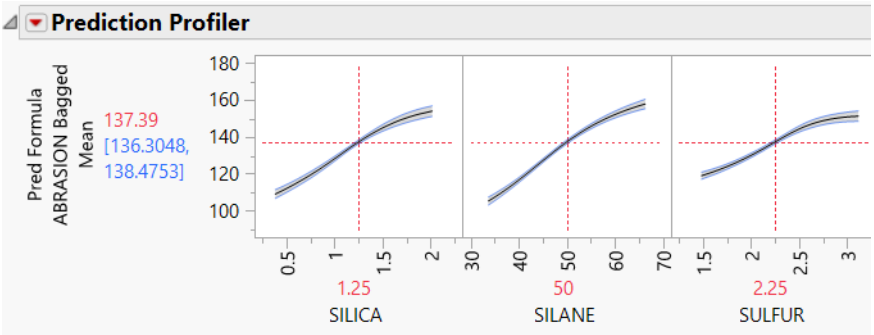
   Note: Results vary due to the random nature of sampling with replacement. To reproduce the results in this example, set the Random Seed.

5. Click **OK**.
6. Select **Graph > Profiler**.
7. Select Pred Formula ABRAISION Bagged Mean and StdError ABRASION Bagged Mean and click **Y, Prediction Formula**.
8. Select **Expand Intermediate Formulas**.
9. Click **OK**.

   A JMP Alert window appears asking if you want to use the StdError ABRASION Bagged Mean column to make intervals for the Pred Formula ABRAISION Bagged Mean column.

10. Click **Yes** in the JMP Alert window.

Figure 3.37 Profiler of Bagged Neural Predictions with Confidence Intervals



A profiler is created with confidence intervals using the values from the StdError ABRASION Bagged Mean column.

Example of Confidence Intervals Using StdErr Pred Formula Column

Use the saved Prediction Formula and StdErr Prediction Formula columns from a standard least squares model to create a prediction profiler with confidence intervals.

Fit the Model

Fit a standard least squares regression model using data on diabetes patients.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.

   By default, the Personality is set to Standard Least Squares.
4. Select Age through Total Cholesterol and click **Add**.
5. Select Validation and click **Validation**.
6. Click **Run**.
7. Click the Response Y red triangle and select **Save Columns > Prediction Formula**.
8. Click the Response Y red triangle and select **Save Columns > StdErr Pred Formula**.

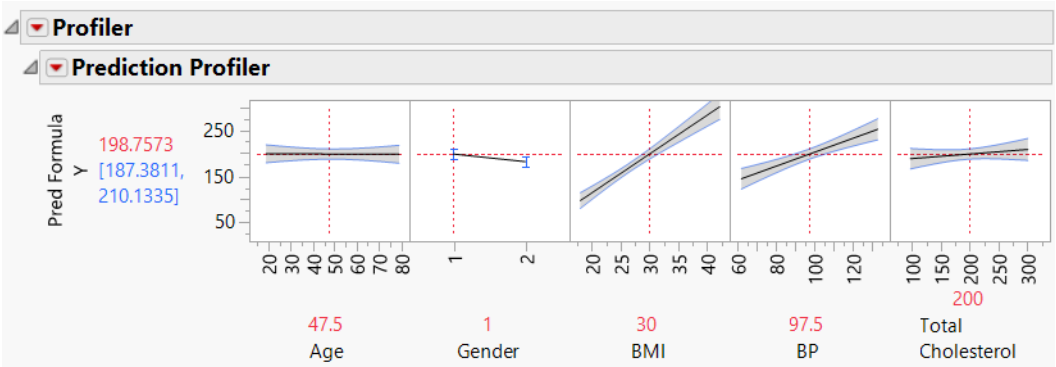
Launch the Profiler

The data table now contains two additional columns named Pred Formula Y and PredSE Y. Use these formula columns in the Profiler.

1. Select **Graph > Profiler**.
2. Select Pred Formula Y and PredSE Y and click **Y, Prediction Formula**.
3. Click **OK**.

   A JMP Alert window appears asking if you want to use the PredSE column to make intervals for the Pred Formula Y column.
4. Click **Yes** in the JMP Alert window.

Figure 3.38 Profiler with Confidence Intervals



A prediction profiler with confidence intervals constructed using the standard error of the predicted values is shown.

Statistical Details for the Prediction Profiler

- "Assess Variable Importance"
- "Propagation of Error Bars"
- "Extrapolation Control Metrics"

Assess Variable Importance

The details that follow relate to the how the variable importance indices are calculated.

Background

Denote the function that represents the predictive model by f, and suppose that $x_1, x_2, \ldots, x_n$ are the factors, or main effects, in the model. Let $y = f(x_1, x_2 \ldots, x_n)$.

- The expected value of y, $E(y)$, is defined by integrating y with respect to the joint distribution of $x_1, x_2, \ldots, x_n$.
- The variance of y, $Var(y)$, is defined by integrating $(y - E(y))^2$ with respect to the joint distribution of $x_1, x_2, \ldots, x_n$.

Main Effect

The impact of the main effect x_j on y can be described by $Var(E(y \mid x_j))$. Here the expectation is taken with respect to the conditional distribution of $x_1, x_2, \ldots, x_n$ given x_j, and the variance is taken over the distribution of x_j. In other words, $Var(E(y \mid x_j))$ measures the variation, over the distribution of x_j, in the mean of y when x_j is fixed.

It follows that the ratio $Var(E(y \mid x_j))/Var(y)$ gives a measure of the sensitivity of y to the factor x_j. The importance index in the Main Effect column in the Summary Report is an estimate of this ratio (see "Adjustment for Sampling Variation").

Total Effect

The Total Effect column represents the total contribution to the variance of $y = f(x_1, x_2 \ldots, x_n)$ from *all* terms that involve x_j. The calculation of Total Effect depends on the concept of functional decomposition. The function f is decomposed into the sum of a constant and functions that represent the effects of single variables, pairs of variables, and so on. These component functions are analogous to main effects, interaction effects, and higher-order effects. See Saltelli (2002); Sobol (1993).

Those component functions that include terms containing x_j are identified. For each of these, the variance of the conditional expected value is computed. These variances are summed. The sum represents the total contribution to $Var(y)$ due to terms that contain x_j. For each x_j, this sum is estimated using the selected methodology for generating inputs. The importance indices reported in the Total Effect column are these estimates (see "Adjustment for Sampling Variation").

Consider a simple example with two factors, x_1 and x_2. Then the Total Effect importance index for x_1 is an estimate of the following:

$$\frac{Var(E(y \mid x_1)) + Var(E(y \mid x_1, x_2))}{Var(y)}$$

Adjustment for Sampling Variation

Due to the fact that they are obtained using sampling methods, the Main Effect and Total Effect estimates shown in the Summary Table might have been adjusted. Specifically, if the Total Effect estimate is less than the Main Effect estimate, then the Total Effect importance index is set equal to the Main Effect estimate. If the sum of the Main Effect estimates exceeds one, then these estimates are normalized to sum to one.

Variable Importance Standard Errors

The standard errors that are provided for independent inputs measure the accuracy of the Monte Carlo replications. Importance indices are computed as follows:

- Latin hypercube sampling is used to generate a set of data values.
- For each set of data values, main and total effect importance estimates are calculated.
- This process is replicated until the estimated standard errors of the Main Effect and Total Effect importance indices for all factors fall below a threshold of 0.01.

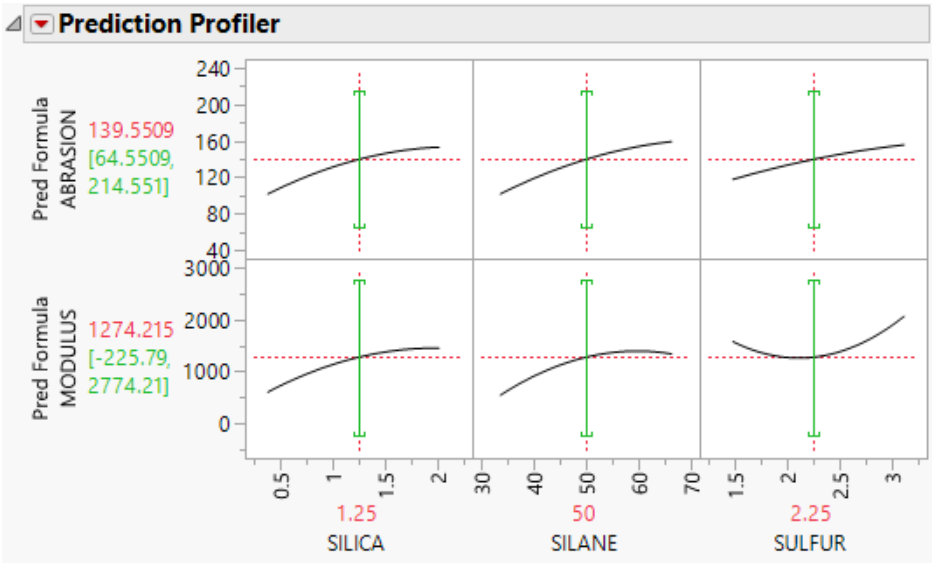
The standard errors that are reported are the standard error values in effect when the replications terminate.

Propagation of Error Bars

Propagation of error (POE) is important when attributing the variation of the response in terms of variation in the factor values when the factor values are not very controllable.

In JMP's implementation, the Prediction Profiler first looks at the factor and response variables to see whether there is a Sigma column property (a specification for the standard deviation of the column, accessed through the Cols > Column Info dialog box). If the property exists, then the Prop of Error Bars command becomes accessible in the Prediction Profiler drop-down menu. This displays the 3σ interval that is implied on the response due to the variation in the factor.

Figure 3.39 Green Propagation of Errors Bars in the Prediction Profiler



The POE is represented in the graph by a green bracket. The bracket indicates the prediction plus or minus three times the square root of the POE variance. The POE variance can be expressed as follows:

$$\sum_{i=1}^{N} \left(\sigma_{x_i}^2 \times \left(\frac{\partial f}{\partial x_i} \right)^2 \right) + \sigma_y^2$$

where σ_y is the user-specified sigma for the response column, and σ_x is the user-specified sigma for the factor column.

Currently, these partial derivatives are calculated by numerical derivatives:

centered, with δ=xrange/10000

POE limits increase dramatically in response surface models when you are over a more sloped part of the response surface. One of the goals of robust processes is to operate in flat areas of the response surface so that variations in the factors do not amplify in their effect on the response.

JMP PRO Extrapolation Control Metrics

Leverage

In models that are fit in the Standard Least Squares personality of the Fit Model platform, the leverage at the factor settings is used as the extrapolation metric.

The leverage of the i^{th} observation, h_{ii}, is the i^{th} diagonal entry of the matrix $\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$, sometimes called the hat matrix. The leverage for a new prediction point is calculated as $h_{pred} = \mathbf{x}'_{pred}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_{pred}$. The following two criteria can be used to determine if a prediction with leverage h_{pred} is an extrapolation:

- $h_{pred} > K \times \max(h_{ii})$, where K is a customizable multiplier
- $h_{pred} > L \times p/n$, where L is a customizable multiplier, p is the number of variables, n is the number of observations, and p/n is the average leverage

You can use the Set Threshold Criterion option to specify which criterion is used and the value of the multiplier. The default values of the multipliers are $K = 1$ and $L = 3$.

Note: Extrapolation control on profilers run from the graph menu using a saved least squares model do not implement the leverage methodology. Instead, the Regularized Hotelling's T^2 methodology is used.

Regularized Hotelling's T^2

In models other than least squares models, the Regularized Hotelling's T^2 value is used as the extrapolation metric. The T^2 value for the training data and T^2 values for the prediction points are calculated as follows:

$$T^2 = (x - \bar{x})' \hat{\Sigma}^{-1} (x - \bar{x})$$

$$T^2_{\text{pred}} = (x_{\text{pred}} - \bar{x})' \hat{\Sigma}^{-1} (x_{\text{pred}} - \bar{x})$$

where $\hat{\Sigma}$ is the Schafer and Strimmer regularized covariance matrix estimator estimated on the training data. The target matrix used for the Schafer Strimmer estimator is a diagonal covariance matrix. See Schafer and Strimmer (2005). In platforms that train models using observations with missing values, the covariance matrix is estimated with pairwise deletion.

Note: Categorical variables are converted to indicator variables for these calculations.

The calculation of the threshold depends on the number of nonmissing T^2 values computed on the training data.

- If there are ten or more nonmissing T^2 values, the threshold is set as follows:

  $$\overline{T^2} + K\hat{\sigma}_{T^2}$$

  where

  K is a customizable multiplier and is set to 3 by default

  $\hat{\sigma}_{T^2}$ is the standard deviation of the T^2 values.

- If there are less than ten nonmissing T^2 values, the threshold is set using an F distribution quantile equivalent to a $K\sigma$ limit.

  $$Q_{T^2}(q)$$

  where

  $q = \Phi(K)$

  $\Phi(\cdot)$ is the standard normal distribution

  K is a customizable multiplier and is set to 3 by default

$$T^2 \sim \frac{(n+1)(n-1)p}{n(n-p)} F(p, n-p)$$

p is the number of parameters

n is the number of nonmissing T^2 values

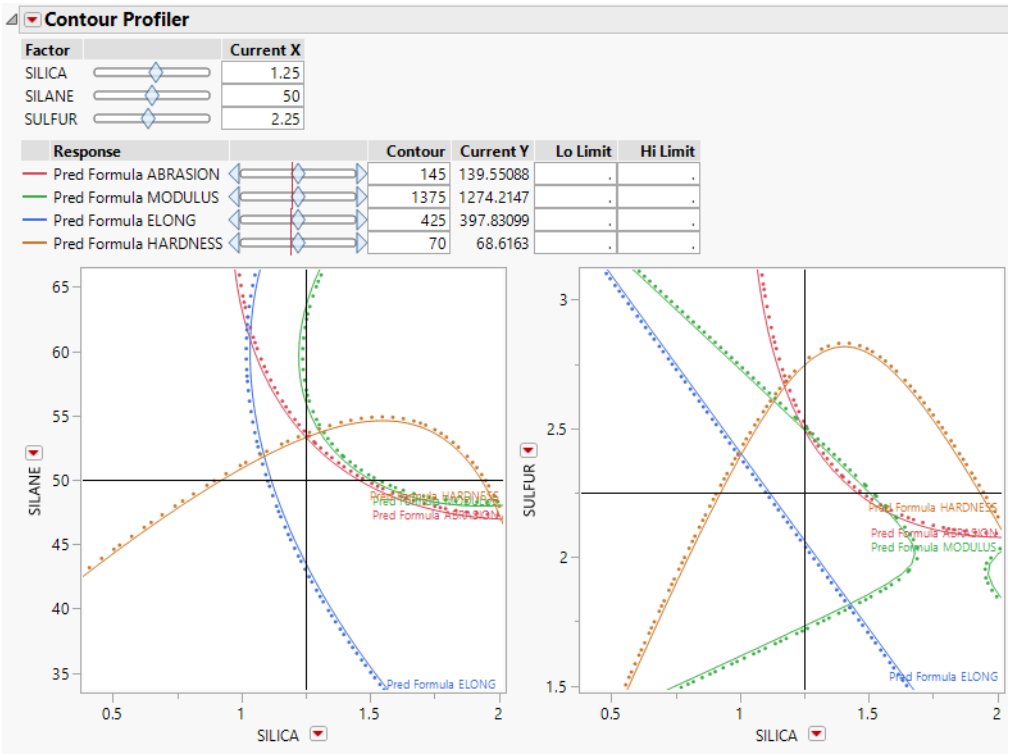
Chapter 4

Contour Profiler

Explore Contours of Responses across Two Factors

The Contour Profiler shows plots of response contours for two factors at a time. Multiple contour plots for different combinations of factors can be viewed at the same time. The interactive contour profiling facility is useful for optimizing response surfaces graphically.

Figure 4.1 Contour Profiler Example



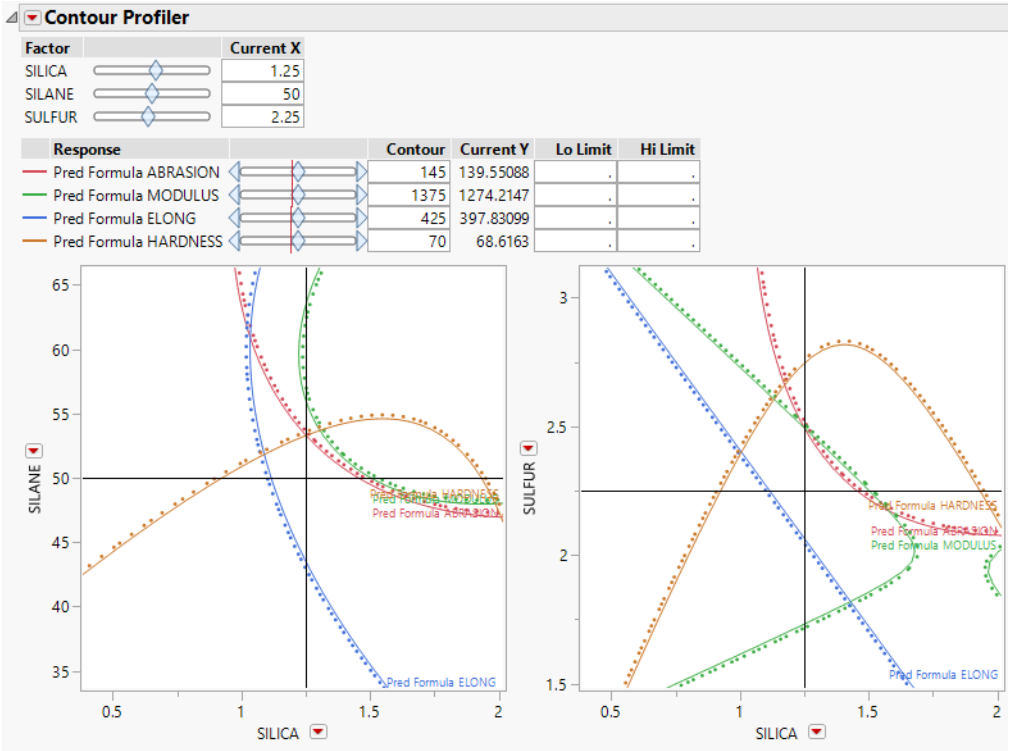
Contents

Overview of the Contour Profiler 93
Example of the Contour Profiler 93
Launch the Contour Profiler Platform 95
The Contour Profiler Report 95
Factor Settings and Controls 96
Response Settings and Controls 96
Contour Profiler Platform Options 97
Constraint Shading Settings 99
Additional Example of the Contour Profiler 99
Explore Optimal Settings 99

Overview of the Contour Profiler

The Contour Profiler shows plots of response contours for two factors at a time. Multiple contour plots for different combinations of factors can be viewed at the same time. Factors can be set to specific values to show how this affects the contours. The interactive contour profiling facility is useful for optimizing response surfaces graphically. Separate surface plots for each response are also available. Figure 4.2 shows an example of the Contour Profiler for the Tiretread.jmp sample data.

Figure 4.2 Contour Profiler



Example of the Contour Profiler

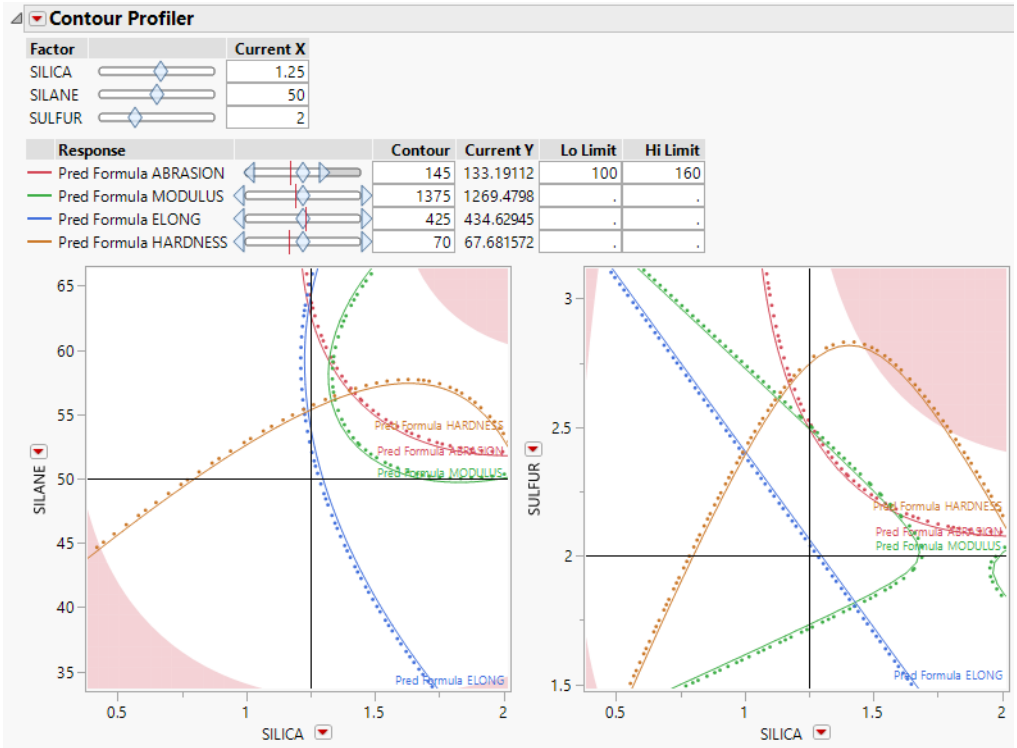
This example uses the Tiretread.jmp sample data table. There are three factors (SILICA, SILANE, and SULFUR) and four responses (ABRASION, MODULUS, ELONG, and HARDNESS). This data table already contains saved prediction formula columns for the four response variables.

1. Select **Help > Sample Data Library** and open Tiretread.jmp.

2. Select **Graph > Contour Profiler**.
3. Select Pred Formula ABRASION, Pred Formula MODULUS, Pred Formula ELONG, and Pred Formula HARDNESS and click **Y, Prediction Formula**.
4. Click **OK**.
5. Click the Contour Profiler red triangle and select **Multiple Contour Frames**.
6. Under Horizontal Factor, select SILICA and under Vertical Factor, select SULFUR.
7. Click **OK**.

   On both contour plots, the values for SILICA are on the horizontal axis. The values for SILANE and SULFUR are on the vertical axes. In the first contour plot, the value for SULFUR is fixed at 2.25. In the second contour plot, the value for SILANE is fixed at 50 (Figure 4.2).
8. Click the **Current X** box for SULFUR and type 2. Click anywhere outside the box to set the value.
9. In the **Lo Limit** box for Pred Formula ABRASION, type 100.
10. In the **Hi Limit** box for Pred Formula ABRASION, type 160.
11. Click anywhere outside the box to update the plot.

Figure 4.3 Contour Profiler for Tiretread.jmp



The contour lines in the first contour plot differ from those in Figure 4.2 because SULFUR is set at a different value. This can be seen in the placement of the horizontal line at 2 in the second contour plot. Move the slider bar for SULFUR to see how different values further change the contour lines in the first contour plot. Move the slider bar for SILANE to see how different values change the contour lines in the second contour plot. The shaded area on the contour plots represent the regions of the graphs that are not feasible, based on the Lo Limit and Hi Limit values for ABRASION.

Launch the Contour Profiler Platform

The Contour Profiler can be accessed in the following ways:

- The Contour Profiler can be accessed directly from the Graph menu. When you access the Contour Profiler in this way, the Contour Profiler launch window appears. See "Profiler Launch Windows".
- The Contour Profiler can be accessed as a red triangle menu option in many modeling platforms. See "Where to Find JMP Profilers" for more information about the availability of the Contour Profiler in different platforms.
- The Contour Profiler can be accessed from the Model Comparisons platform. Select Profiler from the Model Comparisons red triangle menu. Then, select Contour Profiler from the Profiler red triangle menu.
- JMP PRO The Contour Profiler can be accessed from the Formula Depot platform. Select Profiler from the Formula Depot red triangle menu and select the models to be profiled. Then, select Contour Profiler from the Profiler red triangle menu.

The Contour Profiler Report

The initial Contour Profiler report shows a contour profiler plot, factor settings and controls, and response settings and controls. The colored lines on the graph are the contours for the responses set by the Y slider controls or by entering values in the Contour column. There is a separately colored contour for each response. For each contour, there is a dotted line in the direction of higher response values, so that you get a sense of direction. To change the factor assignments to the horizontal and vertical axes of a contour plot, click the red triangle on the axis and select a factor. If you select the factor that is on the opposite axis, the factor assignments are interchanged.

- "Factor Settings and Controls"
- "Response Settings and Controls"

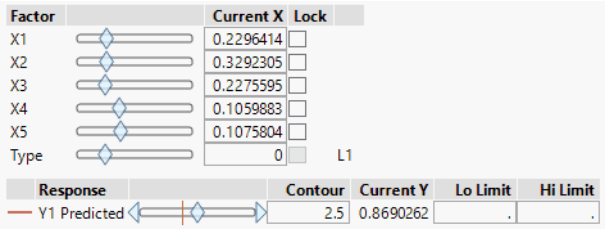
Factor Settings and Controls

Factor The list of factors.

Current X The current factor settings. Click in a box to change the value of a factor or use the slider controls. Right-click the slider control and select **Rescale Slider** to change the scale of the slider. When one value is changed, the values for the other factors remain the same, but the values in the response settings and controls report change accordingly. The Current X location is shown by the crosshair lines on the graph.

Lock A column that appears for mixture designs with more than three predictors. This column enables you to lock settings for mixture values so that they are not changed when the mixture needs to be adjusted due to other mixture effects being changed. When locked columns exist, the shaded area for a mixture recognizes the newly restricted area.

Figure 4.4 Lock Column



Note: If one or more of the factors have specification limits, a dotted rectangle is drawn on the contour plot to show the joint spec limit range.

Response Settings and Controls

Response The list of one or more responses, with a color legend. To change the color of a response, right-click the color legend for that response.

Contour The current value of the contour on the contour profiler. Click in the box to change the value. The slider control can also be used to change the value of the contour.

Current Y The predicted response based on the current X settings. The value is at the red line on the slider control. This value updates as the factor settings are changed.

Lo Limit Enables you to set a lower limit for your response. Click in a box to set the limit or click the left triangle of the slider control for that response. If a response column's Spec Limits column property has a value for Lower Spec Limit, that value is used as the initial value for Lo Limit.

High Limit Enables you to set an upper limit for your response. Click in a box to set the limit or click the right triangle of the slider control for that response. If a response column's Spec Limits column property has a value for Upper Spec Limit, that value is used as the initial value for Hi Limit.

When response limits are used, the profiler shows regions that are not feasible as shaded in the contour profiler plot.

Note: If your response has a Response Probability column property, the High Limit is automatically set to 0.5. In this case, the shaded region is the region where that response level is predicted.

Contour Profiler Platform Options

Grid Density Sets the density of the mesh plots in the surface plots.

Graph Updating Contains options to change the frequency of updates for the graph. (The difference might not be noticeable on a fast machine.) The following two options are available:

Per Mouse Move Updates the graph continuously as you drag the mouse. This is the default setting.

Per Mouse Up Updates the graph each time the mouse is released.

Surface Plot Shows or hides the mesh plots.

Contour Label Shows or hides a label for the contour lines. The label colors match the contour colors.

Contour Grid Draws contours on the Contour Profiler plot at intervals that you specify.

Remove Contour Grid Enables you to remove a contour grid once one is drawn.

Factor Settings Provides a submenu of commands that enables you to save and transfer the Contour Profiler's settings to other parts of JMP. See "Factor Settings".

Simulator Launches the Simulator. See "Simulator".

Up Dots Shows or hides dotted lines that correspond to each contour. The dotted lines show the direction of increasing response values.

Set Contours to Current Resets the contour lines to be where the current Y values are located. This means that all contour lines cross where the crosshairs are on the contour plot and the controls agree in the Y sliders.

Arrange X Controls Left Rearranges the X and Y controls horizontally with the X controls on the left or vertical with the X controls at the top.

Hide X Controls Shows or hides the X controls (Factor Settings and Controls).

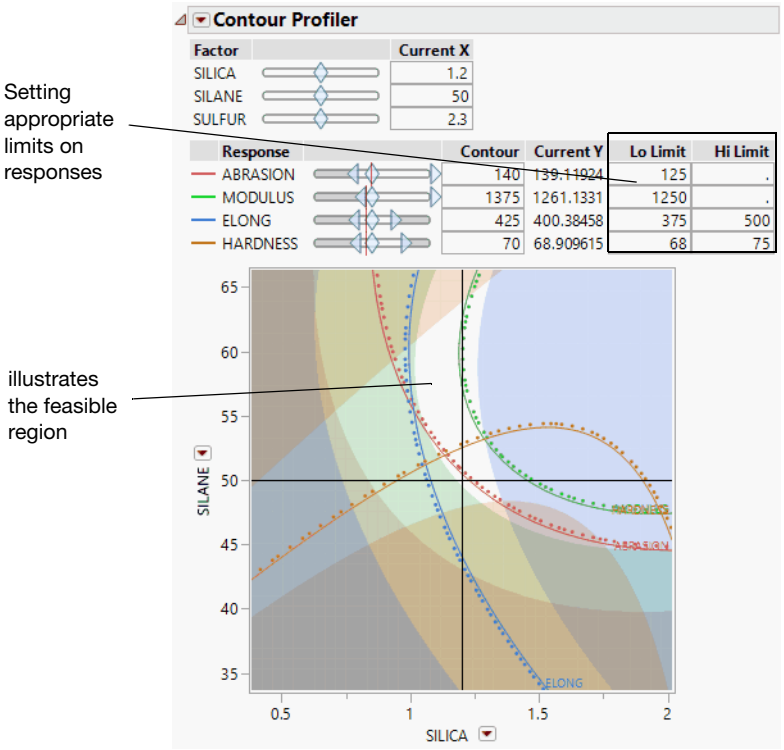
Hide Y Controls Shows or hides the Y controls (Response Settings and Controls).

Multiple Contour Frames Enables you to add one or more contour plots to the report that represent different combinations of factor settings. When you select this option, the Multiple Contour Frames dialog appears. Use the dialog to specify additional individual contour plots or a matrix of contour plots. You can specify a lower triangular, upper triangular, or full square matrix of contour frames. By default, the new contour plots will be added to the current contour plots in the report. Click the **Remove previous frames** check box to remove the current contour plots and replace them with your new selections.

Constraint Shading Settings

If you specify limits for the response columns, the areas of the plot that are outside of the feasible region are shaded. The unshaded white area represents the feasible region.

Figure 4.5 Settings for Contour Shading



Additional Example of the Contour Profiler

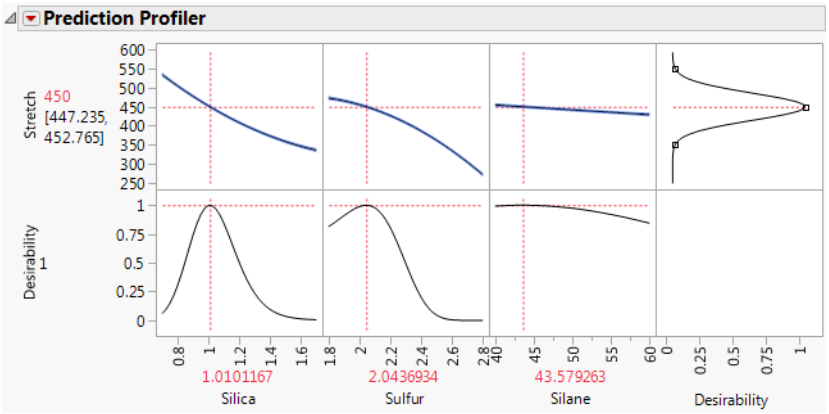
Explore Optimal Settings

Desirability functions are often used to determine the optimal settings for a response surface. The Maximize Desirability feature in the Prediction Profiler provides one combination of factor level settings that results in a predicted response that optimizes the desirability. However, there are often many factor level combinations that can optimize the desirability function. The Contour Profiler is a useful tool for finding alternative settings that optimize desirability.

This example uses the Bounce Data.jmp data table, which contains data for a tire tread experiment. The response, Stretch, is a function of three variables, Silica, Silane, and Sulfur. A model is built and the optimal settings are determined such that the value of Stretch maximizes the desirability function using the Maximize Desirability option in the Prediction Profiler. You use the Contour Profiler to explore other optimal settings for the specified value of Stretch.

1. Select **Help > Sample Data Library** and open Design Experiment/Bounce Data.jmp.
2. Click the green triangle next to the **Model** script, and then click **Run** in the launch window.
3. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 4.6 Maximize Desirability Settings

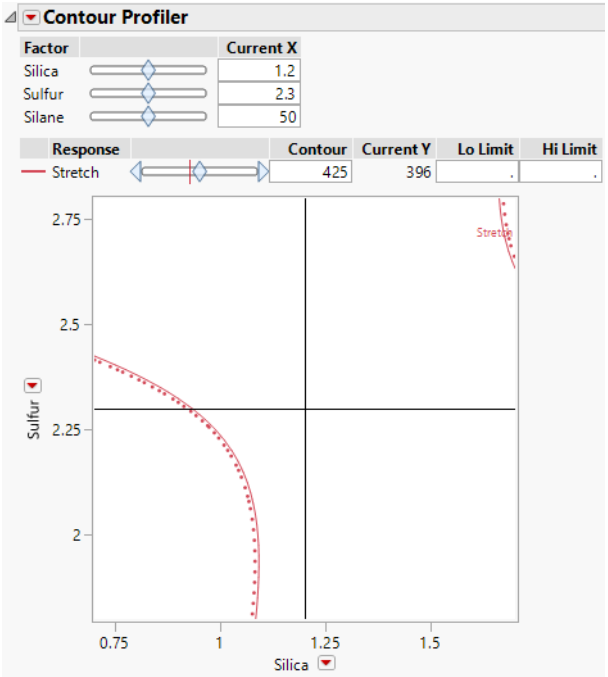


The value of Stretch that maximizes the desirability function is 450. One combination of factor level settings that results in a predicted Stretch of 450 is Silica = 1.0101, Sulfur = 2.0437, and Silane = 43.5792.

4. Click the Response Stretch red triangle and select **Factor Profiling > Contour Profiler**.

   To ensure that your new setting combinations maximize desirability, you want to make sure that the settings predict Stretch within 2 units of 450. In addition, you would like a high level of Silane, a fixed value of 60.

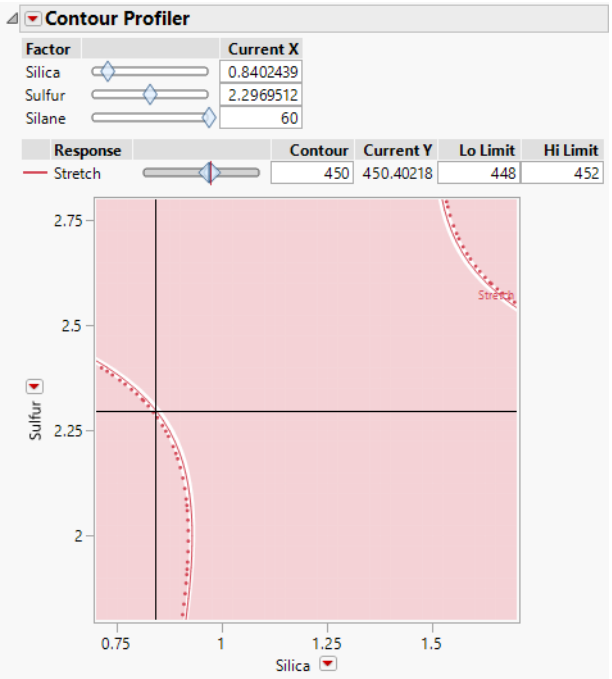
Figure 4.7 Contour Profiler for Bounce Data.jmp



The initial plot shows the contour values of Silica and Sulfur for Stretch at 425 and Silane at 50.

5. In the response controls, set the **Contour** value for Stretch to 450. Set the **Lo Limit** and **High Limit** for Stretch to 448 and 452, respectively. Press Enter.
6. Set the **Current X** for Silane to 60.

Figure 4.8 Contour Profiler for Optimal Settings



The values on the solid red curve are the Silica and Sulfur values that predict Stretch to be 450. The values within the unshaded band predict values of Stretch between 448 and 452. You can drag the crosshairs that appear on the plot to the unshaded sections to find specific settings for Silica and Sulfur. One such setting is seen in Figure 4.8.

Chapter 5

Surface Plot

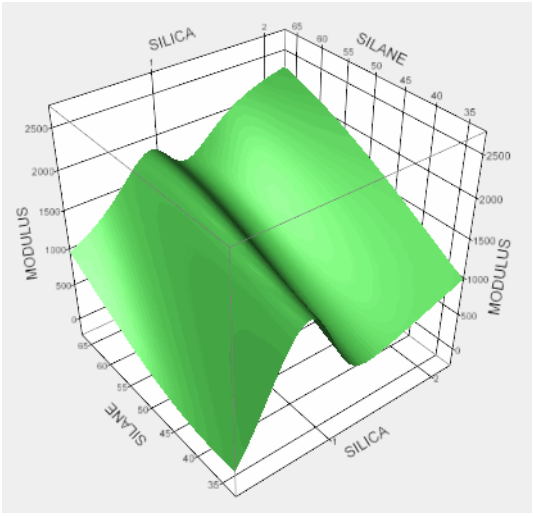
Explore Contours of Responses across Three Factors

A *surface plot* is a three-dimensional plot with one or more dependent variables represented by a smooth surface. You can display up to four dependent variables on the same surface plot.

JMP produces surface plots in the following two situations:

- The Surface Plot platform creates a stand-alone report that contains a surface plot for formulas. The formulas can be formula columns in your data table or mathematical formulas that do not involve any data points.
- The Surface Profiler option in many model fitting platforms produces a surface plot for the fitted model in the existing platform report.

Figure 5.1 Example of a Surface Plot



Contents

Overview of the Surface Plot and Surface Profiler . . . 105
Example of the Surface Plot Platform . . . 105
Launch the Surface Plot Platform . . . 108
The Surface Plot Report . . . 110
The Surface Plot . . . 110
Appearance Controls . . . 111
Independent Variables . . . 112
Dependent Variables . . . 113
Surface Plot Platform Options . . . 115
Pop-Up Menu Options . . . 115
Dependent Variables Options . . . 116
Sheet or Surface Properties . . . 117
Keyboard Shortcuts . . . 119
Additional Examples of the Surface Plot Platform . . . 119
Construct a Surface Plot for a Single Mathematical Function . . . 119
Plotting Isosurfaces . . . 120

Overview of the Surface Plot and Surface Profiler

The Surface Plot and Surface Profiler platforms are used to plot points and surfaces in three dimensions.

Surface plots are available as a separate platform (Graph > Surface Plot) and as options in many reports (known as the Surface Profiler). The surface plot functionality is similar in both situations.

The plots can be of points or surfaces or both. When the points are visible in the surface plot, the points are linked to the data table. You can click the points or use the Brush tool to select rows in the data table. The colors and markers assigned in the data table are also reflected in the surface plot.

Surfaces can be defined by a mathematical equation, or through a set of points defining a polygonal surface. Each surface can be displayed as a smooth surface or as a mesh, with or without contour lines. The labels, axes, and lighting for the surface plot are fully customizable.

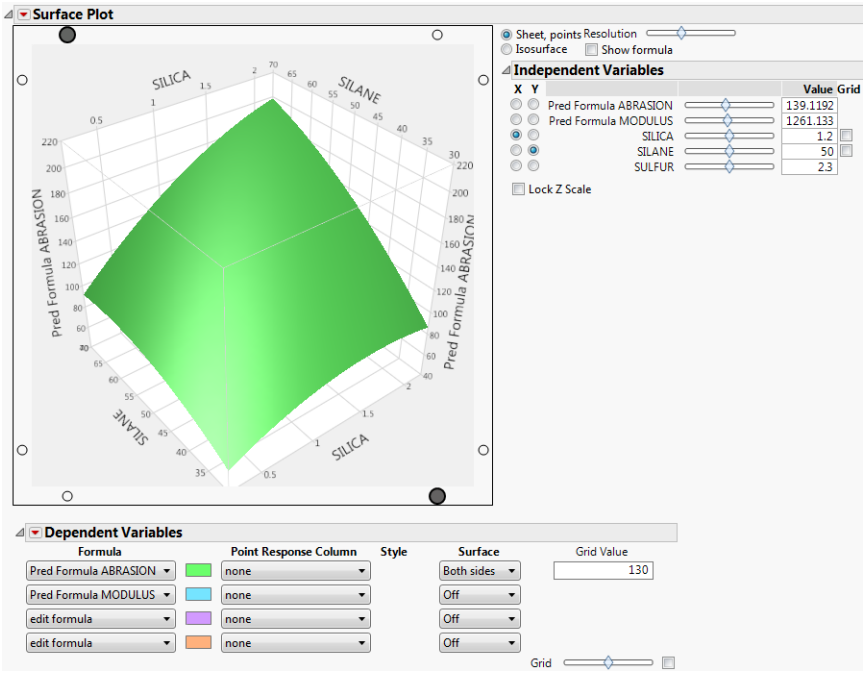
The Surface Plot is built using the 3D scene commands from the JMP Scripting Language (JSL). When you right-click in the surface plot, the pop-up menu contains Open GL Scene commands. For more information about the Open GL Scene commands, see the *Scripting Guide*.

Example of the Surface Plot Platform

This example uses the Tiretread.jmp sample data table. There are three factors (SILICA, SILANE, and SULFUR) and four responses (ABRASION, MODULUS, ELONG, and HARDNESS). This data table already contains saved prediction formula columns for the four response variables. Use these prediction formula columns to examine the surfaces of two of the response variables.

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Graph > Surface Plot**.
3. Select Pred Formula ABRASION and Pred Formula MODULUS, click **Columns**, and then click **OK**.

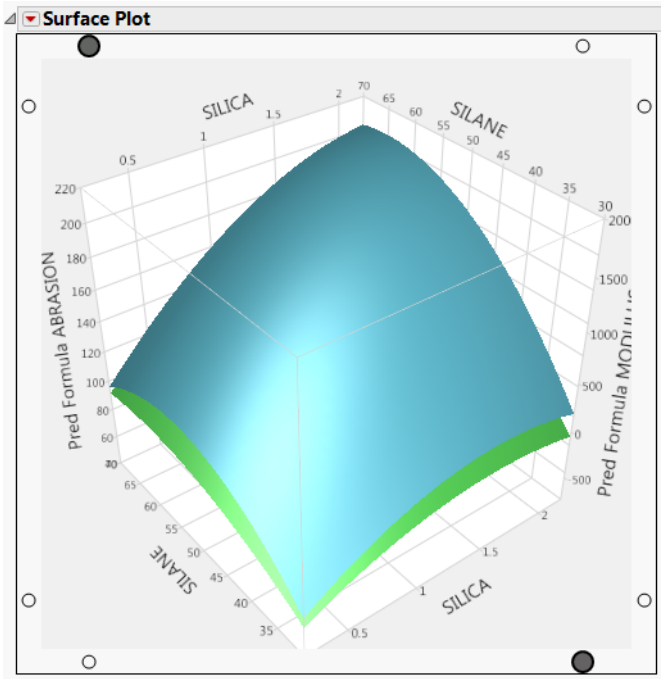
Figure 5.2 Initial Surface Plot Report



Initially, only the prediction surface for ABRASION appears in the plot. Note that in the Independent Variables report, there is a large difference between the values for ABRASION and MODULUS. The two responses are on different scales.

4. In the Dependent Variables report, select **Both Sides** from the Surface list for Pred Formula MODULUS.
5. Click the Surface Plot red triangle and select **Scale response axes independently**.

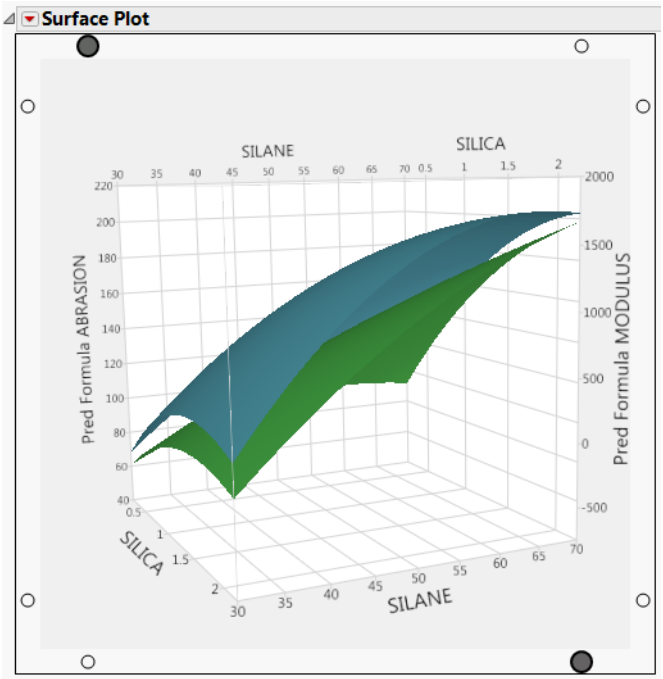
Figure 5.3 Prediction Surface for ABRASION and MODULUS



Pred Formula ABRASON is on one Z axis and Pred Formula MODULUS is on the other Z axis. Each response has its own scale. In this plot, the two factors considered are SILICA and SILANE. You can rotate the surface plot by clicking and dragging inside the graph. This enables you to view the prediction surfaces from different perspectives. For example, in Figure 5.4, you can see that predictions for ABRASION and MODULUS based on the two listed predictors have a similar surface shape, just on a different scale. Use the Independent Variables controls to see how different combinations of predictors affect the surface plot.

Tip: To see the original view after rotating, right-click anywhere on the surface plot and select **Reset**.

Figure 5.4 Rotated Prediction Surfaces

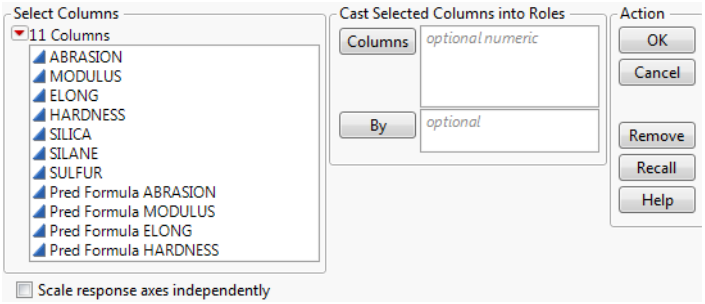


Launch the Surface Plot Platform

The Surface Plot Profiler can be accessed in the following ways:

- To launch the Surface Plot Profiler platform directly, select Surface Plot from the Graph menu. If there is a data table open, this displays the window in Figure 5.5. If you do not want to use a data table for drawing surface plots, click OK without specifying any columns. If there is no data table open, you are presented with the default surface plot shown in Figure 5.13.
- The Surface Plot Profiler can be accessed as a red triangle menu option in many modeling platforms. In the red triangle menus, it is referred to as Surface Profiler. See "Where to Find JMP Profilers" for more information about the availability of the Surface Profiler in different platforms.
- The Surface Plot Profiler can be accessed from the Model Comparisons platform. Select Profiler from the Model Comparisons red triangle menu. Then, select Surface Profiler from the Profiler red triangle menu.
- JMP PRO The Surface Plot Profiler can be accessed from the Formula Depot platform. Select Profiler from the Formula Depot red triangle menu and select the models to be profiled. Then, select Surface Profiler from the Profiler red triangle menu.

Figure 5.5 Surface Plot Launch Window



Columns Specifies the columns that you want to plot. Only numeric variables can be assigned to the Columns role.

By Specifies a variable to create separate surface plots for each level of the variable.

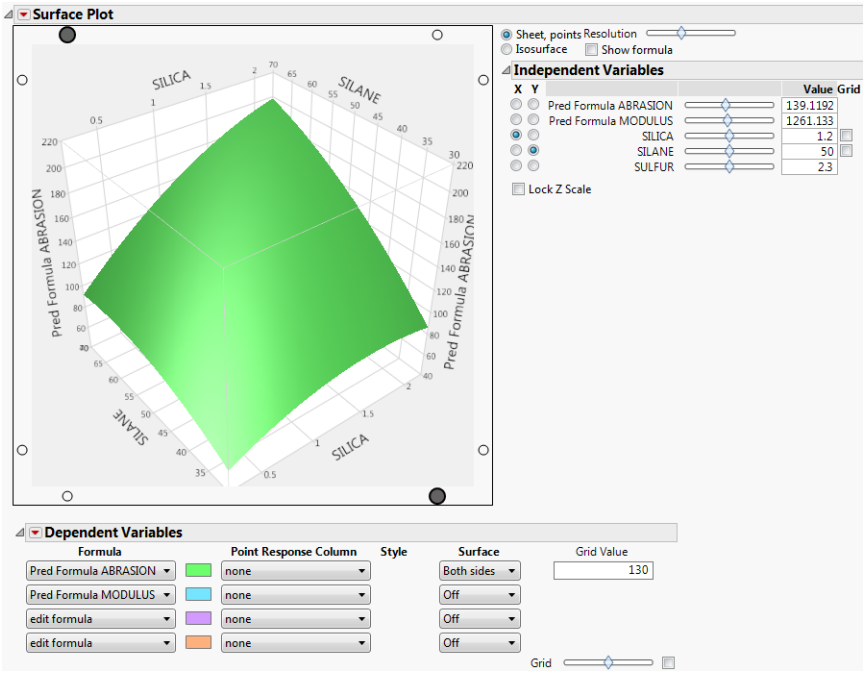
Scale response axes independently Assigns a separate scale to each response on the surface plot.

Note: When not selected, the axis scale for all responses is the same as the scale for the first item entered in the Columns role.

The Surface Plot Report

The initial Surface Plot report shows the surface plot, the appearance controls, the Independent Variables controls, and the Dependent Variables controls. If more than one prediction formula column is specified, only the first surface is displayed in the initial report. However, if more than one observed response column is specified, all points are displayed in the initial report.

Figure 5.6 Example of the Surface Plot Report



The Surface Plot

The surface plot has the following controls and settings:

Rotate Click anywhere that the cursor appears with a circular arrow icon on the surface plot and drag to rotate the plot in any direction.

Note: The Up, Down, Left, and Right arrow keys can also be used to rotate the plot.

Axis Settings Double-click an axis to reveal the axis settings window. The axis settings window enables you to change the scale, tick mark increment, and axis label settings. Like

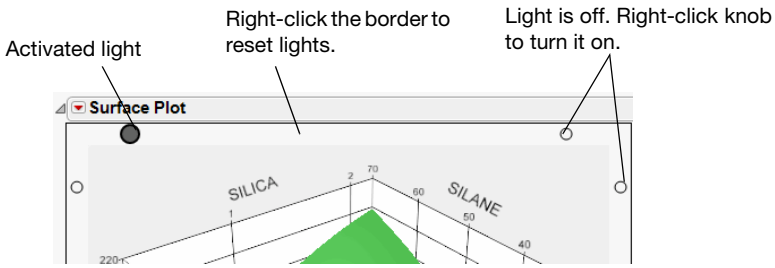
other JMP graphs, the axes can be adjusted, stretched, and compressed using the grabber tool. Hover over an axis to use the grabber tool.

Lights By default, the plot has lights shining on it. There are eight control knobs on the plot for changing the position and color of the lights. This is useful for highlighting different parts of a plot and creating contrast. Four of the eight knobs are shown in Figure 5.7.

You can perform the following actions to adjust the lights shining on the surface:

- Right-click a knob to turn that light on or off. More lights turned on brighten a plot, and fewer lights darken it.
- Drag a knob to change the position of a light.
- Change the color of a light by right-clicking on the knob. The default color is white.

Figure 5.7 Control Knobs for Lights



Appearance Controls

If you select the Surface Profiler option from the red triangle menu of a modeling platform, the Appearance Control options are different than the Appearance Controls in the stand-alone Surface Plot platform. See Figure 5.8 for an example of what the Appearance Controls look like in a modeling platform.

Surface Plot Appearance Controls

The set of controls in the top right of the report enables you to specify the overall appearance of the surface plot.

Sheet, points The setting for displaying sheets, points, and lines.

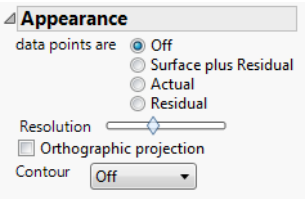
Isosurface Changes the display to show isosurfaces. See "Plotting Isosurfaces".

Show formula Shows the formula edit box, which enables you to edit the formula of the surface. The formula edit box appears only if you specified a column that contains a formula in the launch window.

Resolution A slider that affects how many points are evaluated for a formula. If the resolution is too coarse, a function with a sharp change might not be represented very well. However, if the resolution is set too high, evaluating and displaying the surface is slower than for lower resolutions.

Surface Profiler Appearance Controls

Figure 5.8 Appearance Controls



The Appearance Controls in modeling platforms contain the following controls:

data points are Contains the following options that control the appearance of the data points:

Off Turns the data points off.

Surface plus Residual Shows the difference between the predicted value and actual value on the surface.

Actual Shows the actual data points.

Residual Shows the residual values (if they are not off the plot).

Resolution A slider that affects how many points are evaluated for a formula. If the resolution is too coarse, a function with a sharp change might not be represented very well. However, if the resolution is set too high, evaluating and displaying the surface is slower than for lower resolutions.

Orthogonal Projection Changes the view of the graph, such that the axes form right angles.

Contour Determines the placement of the contour lines on the plot in relation to the surface. Options are off, below, above, and on surface.

Independent Variables

The Independent Variables report contains the following controls:

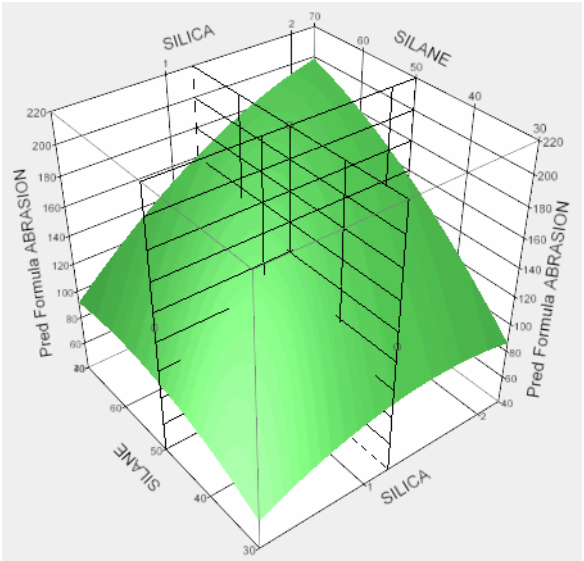
X, Y Radio buttons that enable you to select which independent variables are displayed on the *x*- and *y*-axes. This feature is useful when you have more than two independent variables.

Value The sliders and text boxes set the current values of each variable. These settings are most important for the variables that are not displayed on the axes. The plot shows the three-dimensional slice of the surface at the value shown in the text box. Move the slider to see different slices.

Lock Z Scale Locks the Z axis to its current values. This is useful when moving the sliders that are not on an axis.

Grid Selecting a check box activates a grid that is perpendicular to the selected variables axis. The sliders enable you to adjust the placement of each grid. The resolution of each grid can be controlled by adjusting axis settings. Figure 5.9 shows a surface with the X and Y grids activated.

Figure 5.9 Activated X and Y Grids



Dependent Variables

The Dependent Variables controls that are available depend on whether you have selected Sheet, points or Isosurface in the Appearance Controls.

Controls for Sheet and Points

The Dependent Variables controls that appear by default when Sheet, points is selected are shown in Figure 5.6.

Formula Enables you to select one or more formulas that are displayed as surfaces in the plot.

Point Response Column Enables you to select the column that contains values that are plotted as points.

Style (Menus appear only after you have selected a Point Response Column.) The Style menu enables you to choose how the points are displayed. The menu has the following options:

Points Shows individual points, which change according to the color and marker settings of the row in the data table.

Needles Draws lines from the *x*-*y* plane to the points, or, if a surface is also plotted, connects the surface to the points.

Mesh Connects the points into a triangular mesh.

Surface Overlays a smooth, reflective surface on the points.

Off Points are not shown at all.

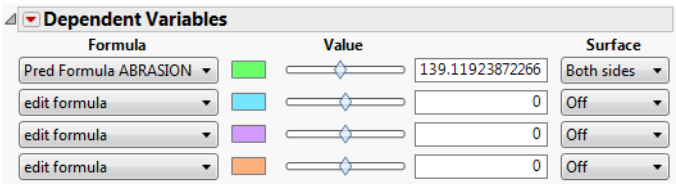
Surface Enables you to show or hide the top or bottom of a surface. If **Above only** or **Below only** is selected, the opposite side of the surface is darkened.

Grid, Grid Value Provides a slider and check box that activate a grid for the dependent variable. You can use the slider to adjust the value at which the grid is drawn. You can also enter the value into the Grid Value text box above the slider.

Controls for Isosurface

Most of the controls for Isosurface are identical to those of Sheet, points. Figure 5.10 shows the default controls, illustrating the slightly different presentation.

Figure 5.10 Dependent Variable Controls for Isosurfaces



Value Slider and text box that activate an isosurface for the dependent variable. You can use the slider to adjust the value at which the isosurface is drawn. You can also enter the value into the text box next to the slider.

Surface Plot Platform Options

The Surface Plot red triangle menu contains the following options:

Control Panel Shows or hides the Control Panel, which includes the controls for Appearance, Independent Variables, and Dependent Variables.

Scale response axes independently Scales response axes independently. See "Launch the Surface Plot Platform".

Fit to Window Determines whether the plot is resized as you resize the report window.

Auto (default) Bases the scaling on the contents of the plot. For example, a plot with By variables or used as a Surface Profiler in a Fit Model platform does not stretch to fit the resized window; the plot extends beyond the viewing area.

On Always fits the plot inside the window.

Off Prevent the plot from resizing.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Pop-Up Menu Options

You can right-click in the plot area to reveal the following pop-up menu options:

Sheet Properties (Option appears only if you right-click the sheet.) Displays the Sheet Properties window. Available only if Sheet, points is selected in the Appearance Controls. See "Sheet or Surface Properties".

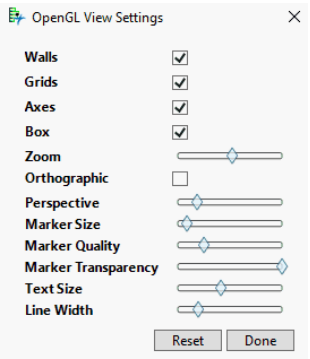
Surface Properties (Options appears only if you click the surface.) Displays the Surface Properties window. Available only if Isosurface is selected in the Appearance Controls. See "Sheet or Surface Properties".

Show Legend Shows or hides a legend.

Reset Resets the plot to the original viewpoint. Changes in wall and background color are not affected.

Settings Opens a window for changing many plot settings. For more information about the OpenGL Scene commands, see the *Scripting Guide*.

Figure 5.11 OpenGL View Settings



Hide Lights Border Shows or hides lighting controls.

Wall Color Enables you to change the plot wall color.

Background Color Enables you to change the plot background color.

Rows Enables you to change row colors or markers, and also exclude, hide, and label points.

Use Hardware Acceleration Provides for faster rendering of the display. For example, if the plot redraws slowly when rotating, this option can help it redraw faster.

Show ArcBall Provides options for using the ArcBall. The ArcBall is a sphere drawn around the plot to help visualize the directions of rotation.

Dependent Variables Options

The Dependent Variables red triangle menu provides the following options:

Formula Shows or hides the Formula column in the Dependent Variables controls.

Surface Shows or hides the Surface column in the Dependent Variables controls.

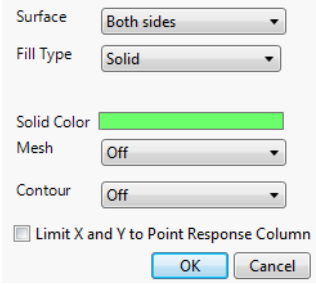
Points Shows or hides the Point Response Column column in the Dependent Variables controls.

Response Grid Shows or hides the Grid controls.

Sheet or Surface Properties

If you have selected the Sheet, points option in the Appearance Controls, you can right-click the sheet inside the plot and select **Sheet Properties** to reveal a window that enables you to change the sheet properties. If you are plotting an Isosurface, right-click the surface and select **Surface Properties** to reveal a similar window. The Sheet Properties and Surface Properties windows enable you to modify the appearance of the surface plot, including the surface color, opacity, and contours. You can also show or hide a mesh that appears on the surface plot.

Figure 5.12 Sheet Properties Window



Surface Enables you to show or hide the top or bottom of a surface. If Above only or Below only is selected, the opposite side of the surface is darkened.

Fill Type Enables you to color the surface using a solid color, or continuous or discrete gradients.

Solid Color (Available only when Solid is selected for Fill Type.) Enables you to choose a color for the surface.

Surface Fill (Available only when a gradient is selected for Fill Type.) Specifies the dependent variable's surface to which the fill type is applied. When Continuous Gradients is selected for Fill Type, you can also choose the Custom option for Surface Fill.

Custom (Available only when Continuous Gradients is selected for Fill Type and when Custom is selected for Surface Fill.) Enables you to specify an equation that defines a response value. This equation is specified using JSL and can refer to columns in the data table using column scoping.

Surface Color Theme (Available only when a gradient is selected for Fill Type.) Enables you to change the color theme for the surface or define a custom color theme. For more information about color themes, see *Using JMP*.

Gradient Levels (Available only when Discrete Gradients is selected for Fill Type.) Enables you to specify the number of different colors used on the surface plot.

Surface Color Range (Available only when a gradient is selected for Fill Type.) Enables you to choose the endpoints for the color gradient. If you choose Data, the endpoints of the color gradient are determined by the range of the data in the column selected for Surface. If you choose Axis, the endpoints of the color gradient are determined by the range of the axis for the dependent variable.

Caution: If you choose Axis for Surface Color Range and change the endpoints of the axis in the surface plot, the cutoff values for the color gradient change as well.

Lighting Enables you to change the lighting of the surface plot. Choose between None, Low Reflection, and Normal.

Mesh Enables you to turn on or off a surface mesh, for either the *X* or *Y* directions or both.

Mesh Color (Available only when a value other than Off is selected for Mesh.) Enables you to select the color for the surface mesh.

Contour Enables you to turn on or off a contour grid, either above, below, or on the surface. If turned on, the Contour Color option is revealed enabling you to change the color.

Contour Color (Available only when a value other than Off is selected for Contour.) Enables you to select the color for the contour grid.

Limit X and Y to Point Response Column Enables you to limit the range of the plot to the range of the data in the Point Response Column, if one is activated. If checked, this option essentially restricts the plot from extrapolating outside the range of the data in the Point Response Column.

Note: The equivalent JSL command for this option is `Clip Sheet( Boolean )`. You can send this message to a particular response column by appending the number of the response column. For example, `Clip Sheet2( 1 )` limits the range of the plot to the range of the data of the second response column. See the Scripting Index in the JMP Help menu for an example.

Keyboard Shortcuts

The following keyboard shortcuts can be used to manipulate the surface plot.

Table 5.1 Surface Plot Keyboard Shortcuts

Key	Function
left, right, up, and down arrows	spin
Home, End	diagonally spin
Enter (Return)	toggles ArcBall appearance
Delete	roll counterclockwise
Control	boost spin speed 10X
Shift	allows continual spinning

Additional Examples of the Surface Plot Platform

- "Construct a Surface Plot for a Single Mathematical Function"
- "Plotting Isosurfaces"

Construct a Surface Plot for a Single Mathematical Function

You can use the Surface Plot platform to produce the graph of a mathematical function without any data points.

1. Select **Graph > Surface Plot**.

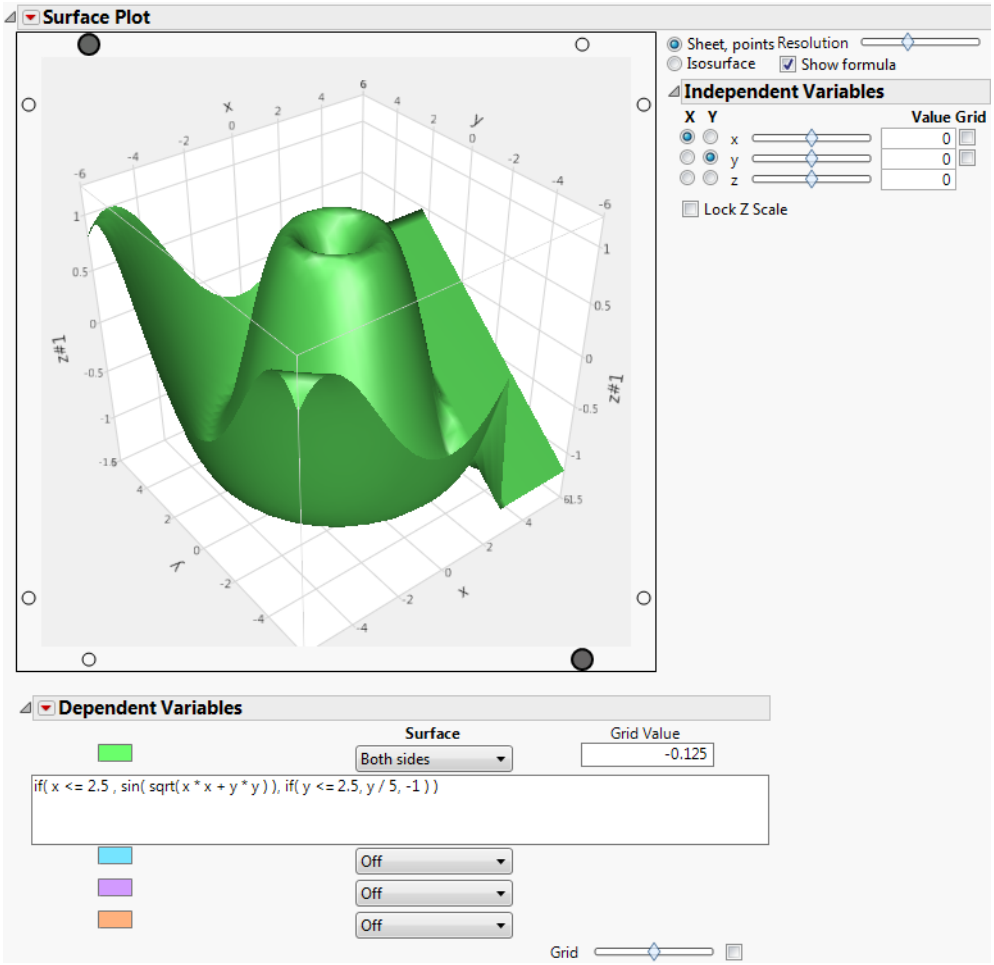
   Note: This can be done with or without a data table open.

2. Click **OK**.

   When you click OK in the Surface Plot launch window without selecting any columns, the default surface plot report appears. If no data tables are open, the launch window does not appear and this step can be skipped.

3. In the Appearance Controls, select the **Show Formula** check box to show the formula space.

Figure 5.13 Default Surface Plot with Formula



The default function shows in the box. To plot your own function, enter it in this box.

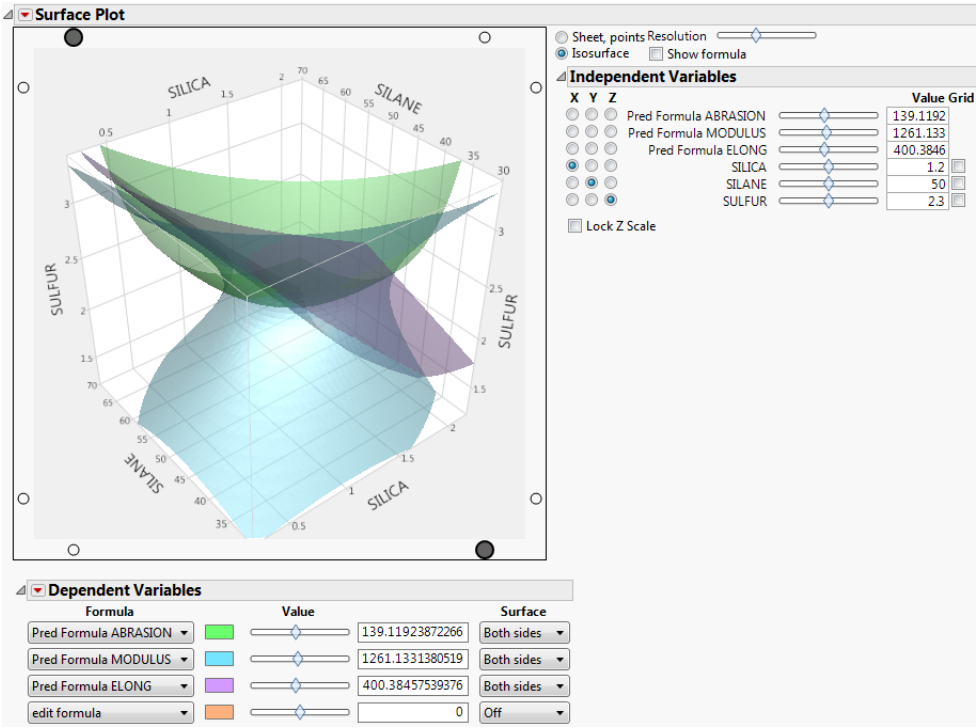
Plotting Isosurfaces

Isosurfaces are the 3-D equivalent to a 2-D contour plot. An isosurface requires a formula with three independent variables. The Resolution slider determines the $n \times n \times n$ cube of points over which the formula is evaluated. The Value slider in the Dependent Variable section selects the isosurface (that is, the contour level) value.

This example uses the Tiretread.jmp data table. The RSM for 4 Responses script produces a response surface model for dependent variables ABRASION, MODULUS, ELONG, and HARDNESS. The predicted formulas for these four responses are in the data table.

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Graph > Surface Plot**.
3. Select Pred Formula ABRASION, Pred FormulaMODULUS, and Pred FormulaELONG and click **Columns**.
4. Click **OK**.
5. Select the Isosurface radio button in the appearance controls.
6. In the Dependent Variables report, select **Both Sides** from the Surface menu for all three variables.

Figure 5.14 Isosurface of Three Variables



For the tire tread data, one might set the abrasion at a fixed minimum setting and the elongation at a fixed maximum setting. Use the MODULUS slider to see which values of MODULUS are inside the limits set by the other two surfaces.

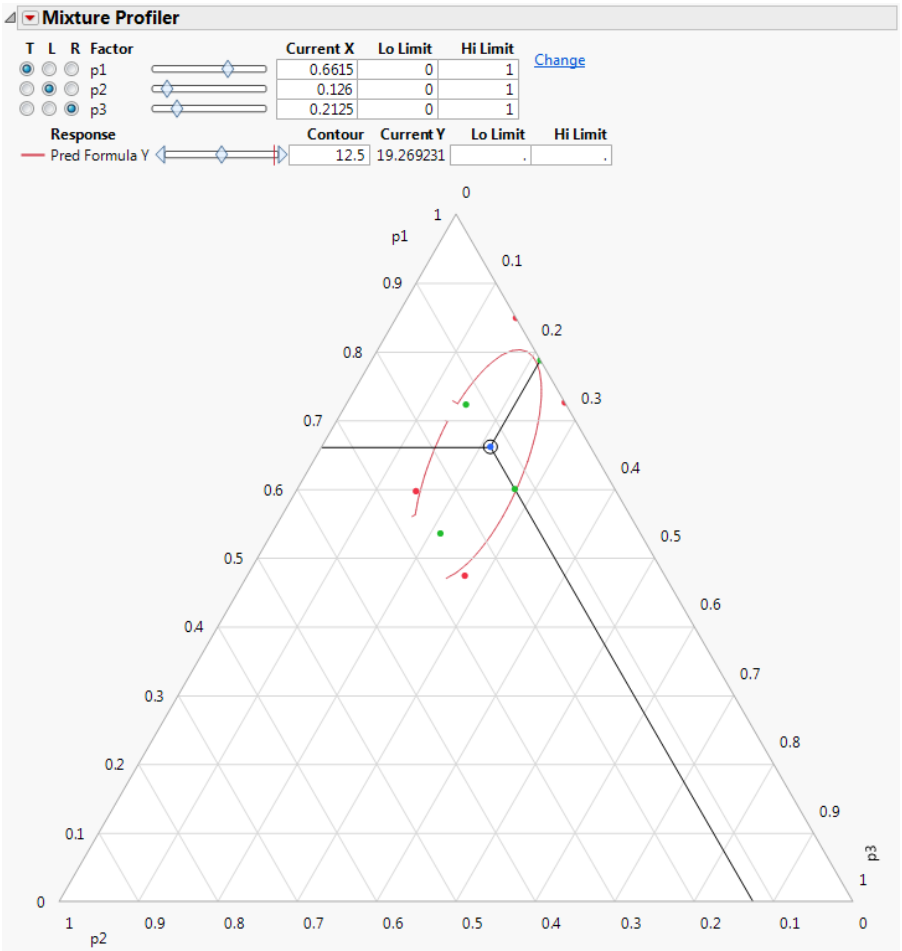
Chapter 6

Mixture Profiler

Explore Factor Effects Using Ternary Plots

The Mixture Profiler displays response contours on a ternary plot for mixture models, where three or more factors in the model are components (ingredients) in a mixture. The Mixture Profiler is useful for visualizing and optimizing response surfaces resulting from mixture experiments.

Figure 6.1 Mixture Profiler Example



Contents

Overview of the Mixture Profiler 125
Ternary Plot Overview 125
Ternary Plots with More Than Three Components 126
Example of the Mixture Profiler 127
Launch the Mixture Profiler Platform 128
The Mixture Profiler Report 129
Factor Settings and Controls 129
Response Settings and Controls 130
Mixture Profiler Platform Options 130
Customizations for the Mixture Profiler 131
Linear Constraints 131
Additional Examples of the Mixture Profiler Platform 132
Mixture Variables with a Process Variable 132
Multiple Responses with Five Mixture Variables 135

Overview of the Mixture Profiler

The Mixture Profiler shows response contours on a ternary plot for mixture models, where three or more factors in the model are components (ingredients) in a mixture. The Mixture Profiler is useful for visualizing and optimizing response surfaces resulting from mixture experiments.

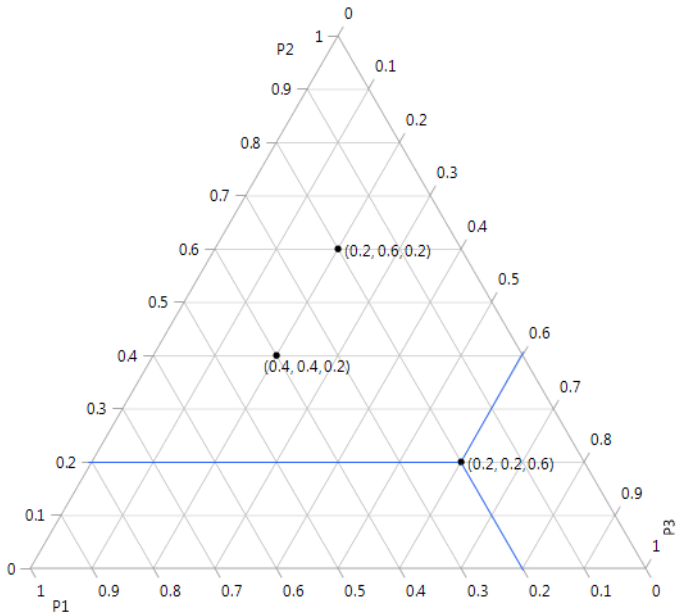
Many of the Mixture Profiler features are similar to those of the Contour Profiler as described in "Contour Profiler Platform Options". Unique to the Mixture Profiler is the use of a ternary plot.

Ternary Plot Overview

A ternary plot is a two-dimensional representation of three mixture components that sum to a constant. The plot is an equilateral triangle with an edge for each component. When unconstrained, each vertex of the triangle corresponds to a pure blend where one component is 1 (100%) and all other components are 0. When components are constrained, the feasible mixtures are represented by a portion of the ternary plot. Shading is used to exclude infeasible portions of the plot.

Figure 6.2 displays three components P1, P2, and P3. The three components are unconstrained with the range of each proportion ranging from 0 to 1 (100%). Three points are labeled with their coordinates (P1, P2, P3). One point (0.2, 0.2, 0.6) includes blue lines along the axis grid lines for each mixture component to guide you in reading the axes.

Figure 6.2 Ternary Plot Example



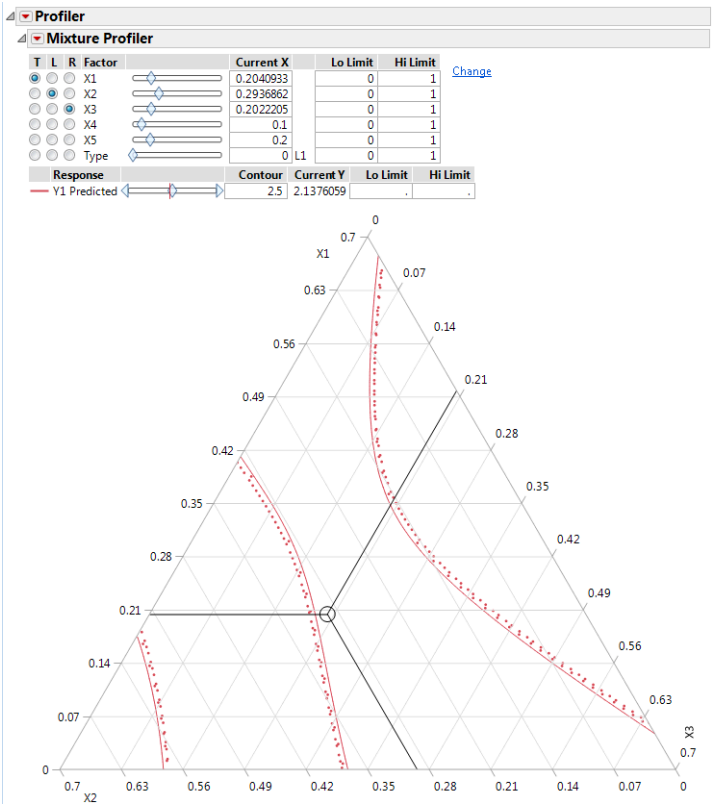
Ternary Plots with More Than Three Components

The ternary plot can display only three components at a time. For a model with more than three components, the total of the three on-axis (displayed) components is 1 minus the total of the off-axis (non-displayed) components. The plot axes are scaled such that the maximum value a component can attain is 1 minus the total for the off-axis components.

Figure 6.3 shows the Mixture Profiler Ternary Plot for an experiment with 5 factors. The Five Factor Mixture.jmp data table is being used, with the Y1 Predicted column as the formula. The on-axis factors are x1, x2, and x3, and the factors x4 and x5 are off-axis. There are no constraints on any of the mixture components. Note that all Lo Limit values are set to 0 and all Hi Limit values set to 1. The value for x4 is 0.1 and the value for x5 is 0.2, for a total of 0.3. The sum of x1, x2, and x3 is equal 1 – 0.3 = 0.7. Note that the maximum value for a plot axis is now 0.7, not 1.

If you change the value for either x4 or x5, then the values for x1, x2, and x3 change, maintaining their relative proportions, to accommodate the constraint that factor values sum to 1.

Figure 6.3 Ternary Plot with Scaled Axes to Account for Off-Axis Factors



Example of the Mixture Profiler

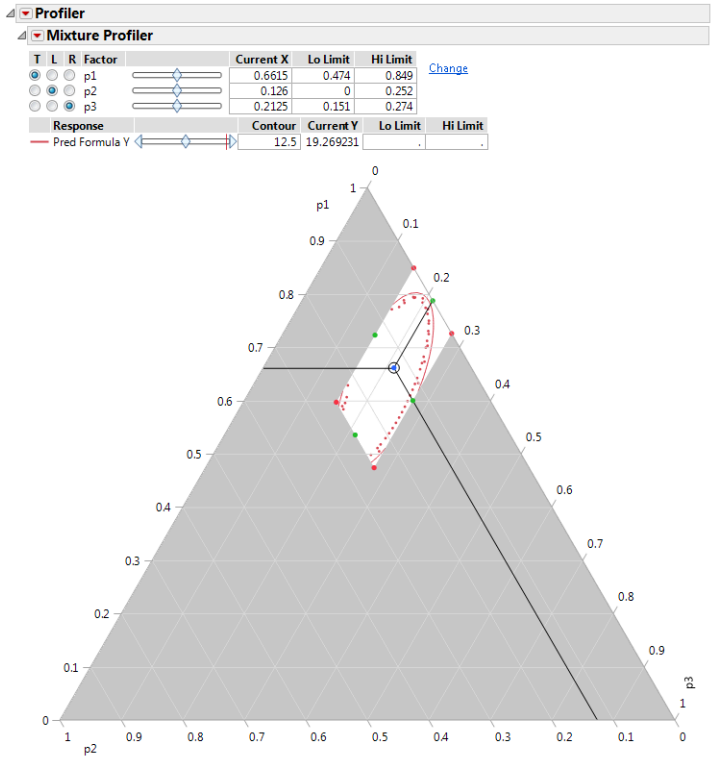
This example uses the Plasticizer.jmp sample data table There are three mixture components (p1, p2, and p3) and one response (Y). There are constraints on the component levels. This data table already contains a saved prediction formula column for the response.

1. Select **Help > Sample Data Library** and open Plasticizer.jmp.
2. Select **Graph > Mixture Profiler**.
3. Select Pred Formula Y and click **Y, Prediction Formula**.
4. Select Y from the Y, Prediction Formula role and click **Remove**.

Note: The Y column role is set to Y in the Columns list. The role automatically assigns columns in launch windows. In this case, the Y column was assigned to the Y role when the mixture profiler was launched. However, the Y column does not contain a formula, and the Mixture Profiler Y column requires a formula column.

5. Click **OK**.

Figure 6.4 Example of Mixture Profiler



Use the factor sliders to explore how changes impact the response. The unshaded region is the feasible region for this data set.

Launch the Mixture Profiler Platform

The Mixture Profiler can be accessed in the following ways:

- The Mixture Profiler can be accessed directly from the Graph menu. When you access the Mixture Profiler in this way, the Mixture Profiler launch window appears. See "Profiler Launch Windows".
- The Mixture Profiler can be accessed as a red triangle menu option in many modeling platforms. See "Where to Find JMP Profilers" for more information about the availability of the Mixture Profiler in different platforms.

- The Mixture Profiler can be accessed from the Model Comparisons platform. Select Profiler from the Model Comparisons red triangle menu. Then, select Mixture Profiler from the Profiler red triangle menu.
- JMP PRO The Mixture Profiler can be accessed from the Formula Depot platform. Select Profiler from the Formula Depot red triangle menu and select the models to be profiled. Then, select Mixture Profiler from the Profiler red triangle menu.

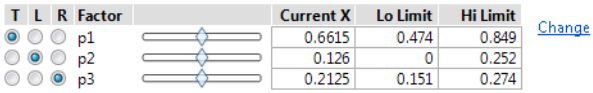
The Mixture Profiler Report

The initial Mixture Profiler report shows a mixture profiler plot, factor settings and controls, and response settings and controls.

- "Factor Settings and Controls"
- "Response Settings and Controls"

Factor Settings and Controls

Figure 6.5 Factor Settings and Controls



T, L, R Radio buttons to control factor assignments to the axes in the mixture profiler plot. **T** = top, **L** = left, and **R** = right.

Factor The list of factors.

Current X The current factor settings. Click in a box to change the value of a factor. The values for the other factors adjust proportionally to maintain the mixture sum. The slider controls can also be used to change factor settings.

Lo Limit The lower limit on each factor. Click in a box to change the value.

Hi Limit The upper limit on each factor. Click in a box to change the value.

Note: When factor limits are used, the profiler shows regions that are feasible as unshaded.

Change Opens the Factor Settings window. See "Mixture Profiler Platform Options".

Response Settings and Controls

Figure 6.6 Response Settings and Controls

Response The list of one or more responses.

Contour The current value of the contour on the mixture profiler. Click in the box to change the value. The slider control can also be used to change the value of the contour.

Current Y The predicted response based on the current X settings. The value is at the center of the cross hairs on the profiler plot. This value updates as the factor settings are changed.

Lo Limit Enables you to set a lower limit for your response.

High Limit Enables you to set an upper limit for your response.

Note: When response limits are used, the profiler shows regions that are feasible as unshaded.

Mixture Profiler Platform Options

Specify Factor Values Opens a dialog for specifying factor values. The entered values should sum to1. If needed, JMP adjusts the specified values to feasible settings. The option is also available from the **Change** button on the report window.

Show Points Shows or hides the individual points on the plot.

Show Current Value Shows or hides the three-way crosshairs on the plot. The intersection of the crosshairs represents the current factor values. The Current X values above the plot give the exact coordinates of the crosshairs.

Show Constraints Shows or hides the shading resulting from any constraints on the factors. Those constraints can be entered in the Lo Limits and Hi Limits columns above the plot, or in the Mixture Column Property for the factors.

Up Dots Shows or hides the dotted lines corresponding to each contour. The dotted lines show the direction of increasing response values, so that you get a sense of direction.

Contour Grid Draws contours on the plot at intervals that you specify.

Remove Contour Grid Removes the contour grid if one is on the plot.

Factor Settings A submenu of commands that enables you to save and transfer the Mixture Profiler settings to other parts of JMP. For more information on this submenu, see the discussion of the profiler in "Factor Settings".

Customizations for the Mixture Profiler

Customize the Mixture Profiler by right-clicking on the plot, and selecting **Customize**.

Contour Alter line color, fill color, and level of line transparency. If there are multiple contour lines on the plot, each appears individually on the list of options.

Component Constraints Alter component constraint display properties.

Linear Constraints Alter linear constraint display properties.

Grid Lines Use Axis Settings window to modify grid lines.

Reference Lines Use Axis Settings window to modify reference lines.

Double-click an axis on the plot to view the Axis Settings window. Here you can specify axis properties in detail, such as tick marks, grid lines and reference lines.

Crosshairs Alter the display properties of the crosshairs on the plot.

Marker Alter the display properties of the plot marker.

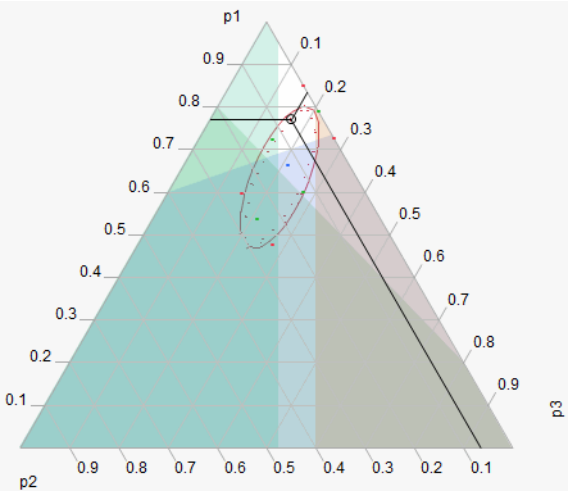
Linear Constraints

The Mixture Profiler can display linear constraints. To do this, a Constraint Table Script must be part of the data table. For more information about creating the Table Script, see "Linear Constraints".

When using constraints, unfeasible regions are shaded in the profiler. Figure 6.7 shows an example of a mixture profiler with shaded regions due to four constraints. The unshaded portion is the resulting feasible region. The constraints are below:

- $4*p2 + p3 \leq 0.8$
- $p2 + 1.5*p3 \leq 0.4$
- $p1 + 2*p2 \geq 0.8$
- $p1 + 2*p2 \leq 0.95$

Figure 6.7 Shaded Regions Due to Linear Constraints



Additional Examples of the Mixture Profiler Platform

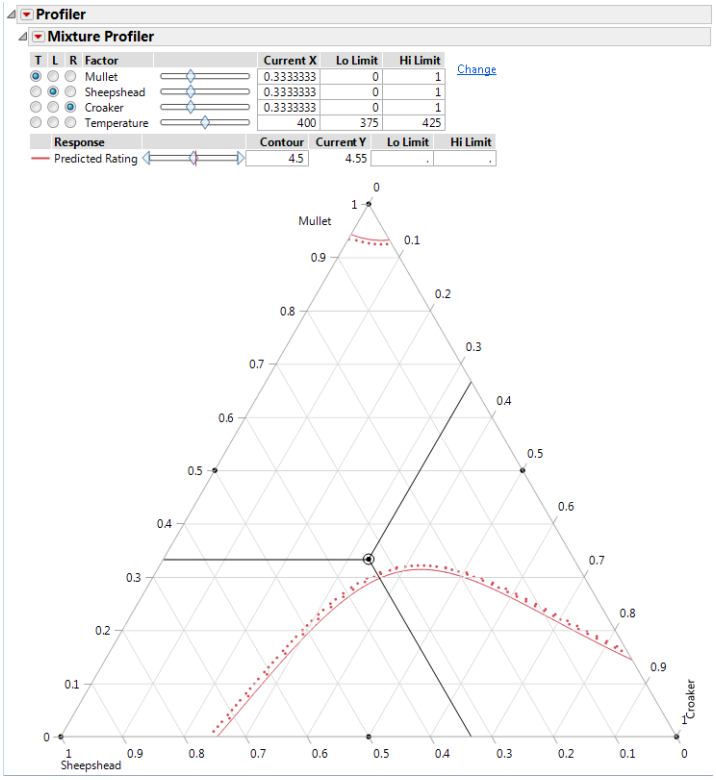
- "Mixture Variables with a Process Variable"
- "Multiple Responses with Five Mixture Variables"

Mixture Variables with a Process Variable

This example uses the Fish Patty.jmp sample data table. The data, adapted from Cornell (1990), comes from an experiment to optimize the texture of fish patties. The columns Mullet, Sheepshead, and Croaker are mixture components. Each column represents the proportion of a fish type in the fish patty. The Temperature column is a process variable. It is the oven temperature used to bake the patties. The column Rating is the response and is a measure of texture acceptability, where higher is better. A response surface model was fit to the data and the prediction formula was stored in the column Predicted Rating.

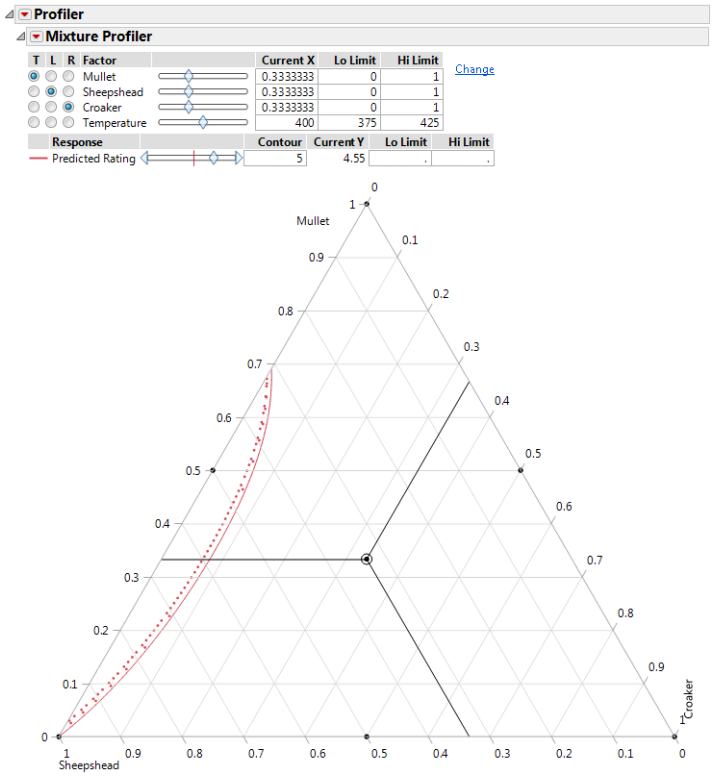
1. Select **Help > Sample Data Library** and open Fish Patty.jmp.
2. Select **Graph > Mixture Profiler**.
3. Select Predicted Rating and click **Y, Prediction Formula**.
4. Click **OK**.

Figure 6.8 Initial Output for Mixture Profiler



5. The manufacturer wants the rating to be at least 5. Use the slider control for Predicted Rating to move the contour close to 5. Alternatively, you can enter 5 in the Contour edit box to set the contour to 5.

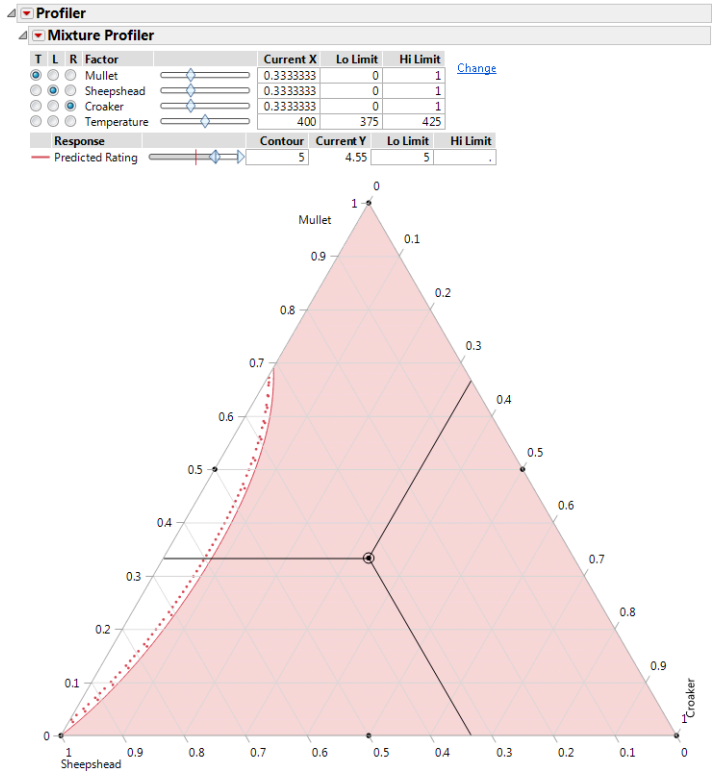
Figure 6.9 Contour Showing a Predicted Rating of 5



The Up Dots shown along the contour indicate the direction of increasing Predicted Rating.

6. Enter 5 in the Lo Limit edit box for the Predicted Rating. The resulting shaded region represents factor combinations that yield a rating less than 5. To produce patties with at least a rating of 5, the manufacturer can set the factors values anywhere in the feasible (unshaded) region. The region has small proportions of Croaker (<10%), mid to low proportions of Mullet (<70%) and mid to high proportions of Sheepshead (>30%). This region is for a baking temperature of 400 degrees.

Figure 6.10 Contour Shading Showing Predicted Rating of 5 or More.



7. Move the slide control for Temperature to observe how the feasible region changes for different temperature settings.

Additional analyses might include:

- Optimize the response across all four factors simultaneously. See "Custom Profiler" or "Desirability Profiling and Optimization".
- Simulate the response as a function of the random variation in the factors and model noise. See "Simulator".

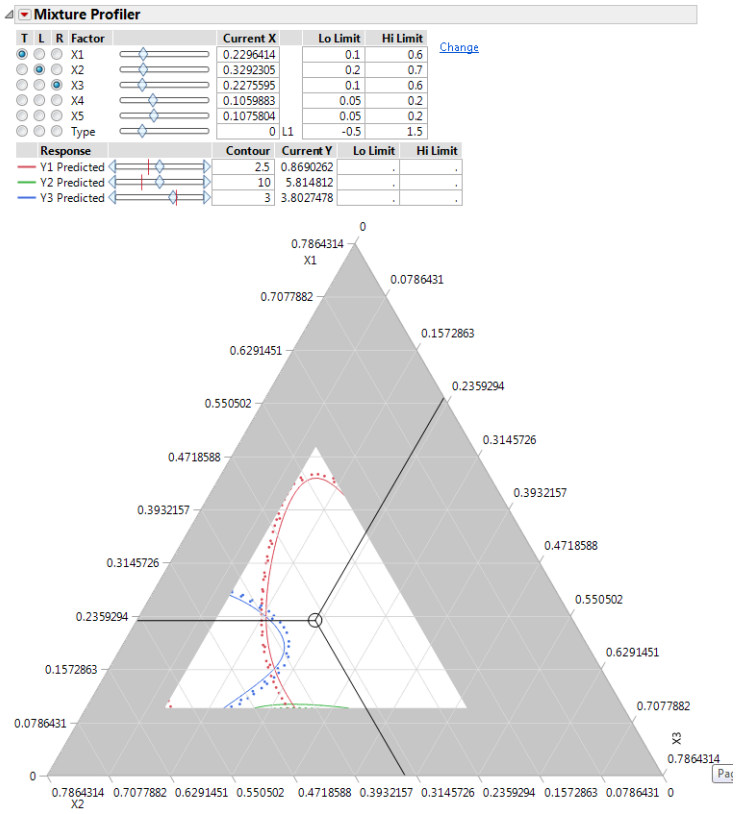
Multiple Responses with Five Mixture Variables

This example uses the Five Factor Mixture.jmp sample data table. There are five components (x1 to x5), one categorical process factor (Type), and three responses, Y1, Y2, and Y3. A response surface model is fit to each response and the prediction equations are saved in Y1 Predicted, Y2 Predicted, and Y3 Predicted.

1. Select **Help > Sample Data Library** and open Five Factor Mixture.jmp.
2. Select **Graph > Mixture Profiler**.

3. Select Y1 Predicted, Y2 Predicted, and Y3 Predicted and click **Y, Predicted**.
4. Click **OK**.
5. Enter 3 in the Contour edit box for Y3 Predicted so that the contour is visible on the plot.

Figure 6.11 Mixture Profiler with Y3 Predicted Contour Set to 3



A few items to note about the output.

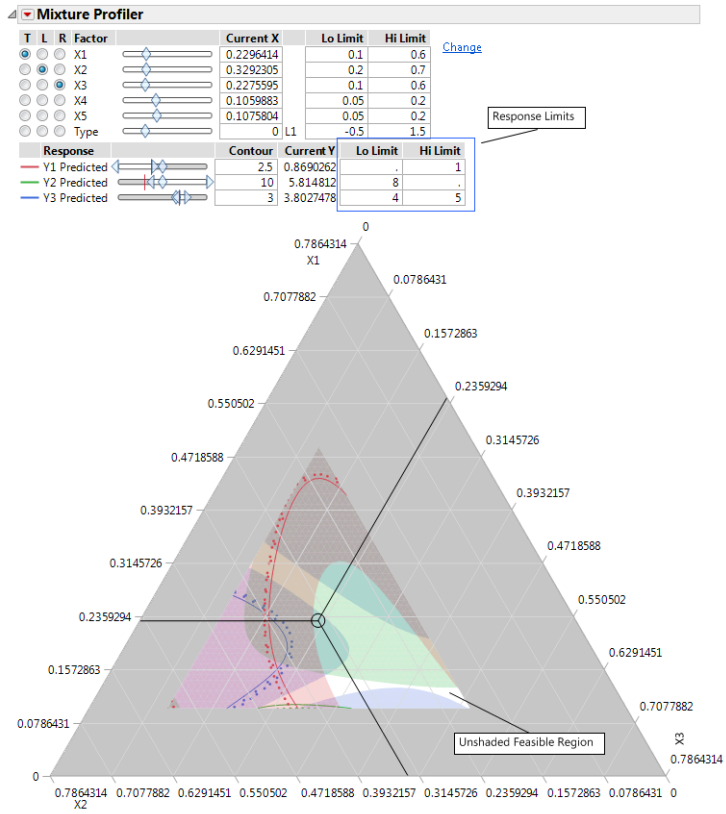
- The on axis factors are X1, X2, and X3 as indicated by the axes labels and the radio buttons in the factor settings. All of the factors have low and high limits, which were entered previously as Column Properties. See *Using JMP* for more information about entering column properties. Alternatively, you can enter the low and high limits directly by entering them in the Lo Limit and Hi Limit boxes.
- The white unshaded region is the feasible region. This is determined by the factor limits.
- The on-axis factors, x1, x2, and x3, radio buttons are selected.
- The categorical factor, Type, has a radio button, but it cannot be assigned to the plot. The current value for Type is L1, which is listed immediately to the right of the Current X box. The Current X box for Type uses a 0 to represent L1.

- All three prediction equations have contours on the plot and are differentiated by color.

Suppose the manufacturer desires to hold Y1 less than 1, Y2 greater than 8 and Y3 between 4 and 5, with a target of 4.5. The Mixture Profiler can help you investigate the response surface and find optimal factor settings.

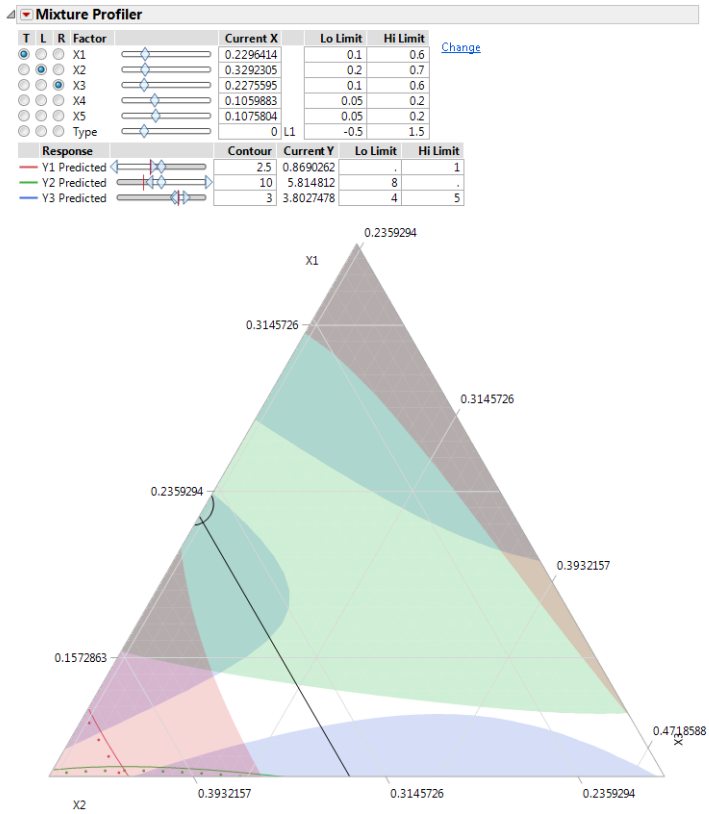
6. Enter 1 in the Y1 Predicted Hi Limit edit box. Enter 8 in the Y2 Predicted Lo Limit edit box. Enter 4 in theY3 Predicted Lo Limit edit box and 5 in the Y3 Predicted Hi Limit edit box.
7. The feasible region remains white (unshaded). Use the Response slider controls to position the contours in the feasible region. Alternatively, move the cross hair into the feasible region

Figure 6.12 Feasible Region After Setting Response Limits



8. Select the magnifier tool to zoom in on the feasible region.

Figure 6.13 Ternary Plot with Enlarged Feasible Region

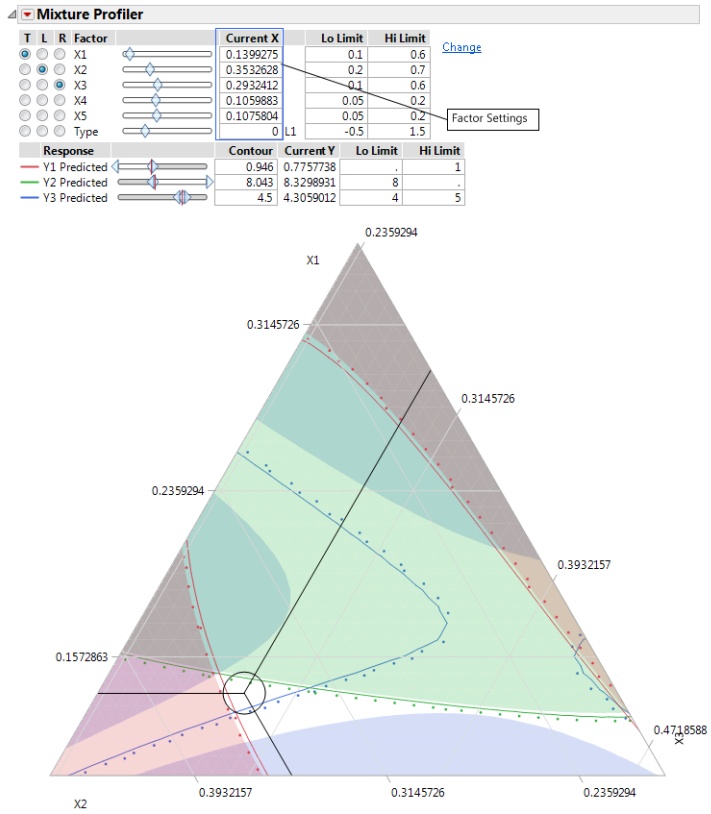


The manufacturer wants to maximize Y1, minimize Y2, and have Y3 at 4.5.

9. Use the slider controls or Contour edit boxes for Y1 Predicted to maximize the red contour within the feasible region. The Up Dots show direction of increasing predicted response.
10. Use the slider controls or Contour edit boxes for Y2 Predicted to minimize the green contour within the unshaded region.
11. Enter 4.5 in the Contour edit box for Y3 Predicted to set the blue contour to the target value.

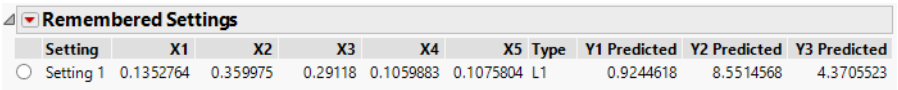
The resulting three contours do not all intersect at one spot. Y1 and Y2 cannot be optimized with Y3 on target, so you have to compromise. Position the three-way crosshairs in the middle of the contours to explore the factor levels that produce those response values. Note that these response contours are for the current settings of x4, x5, and Type.

Figure 6.14 Factor Settings for Optimal Results



12. Click the Mixture Profiler red triangle and select **Factor Settings > Remember Settings** to save the current settings. The settings are appended to the bottom of the report window.

Figure 6.15 Remembered Settings



Remembered Settings

Setting	X1	X2	X3	X4	X5	Type	Y1 Predicted	Y2 Predicted	Y3 Predicted
Setting 1	0.1352764	0.359975	0.29118	0.1059883	0.1075804	L1	0.9244618	8.5514568	4.3705523

With the current settings saved, you can now change the values of x4, x5 and Type to explore their impact on the feasible region. You can compare the factor settings and response values for each level of Type by referring to the Remembered Settings report.

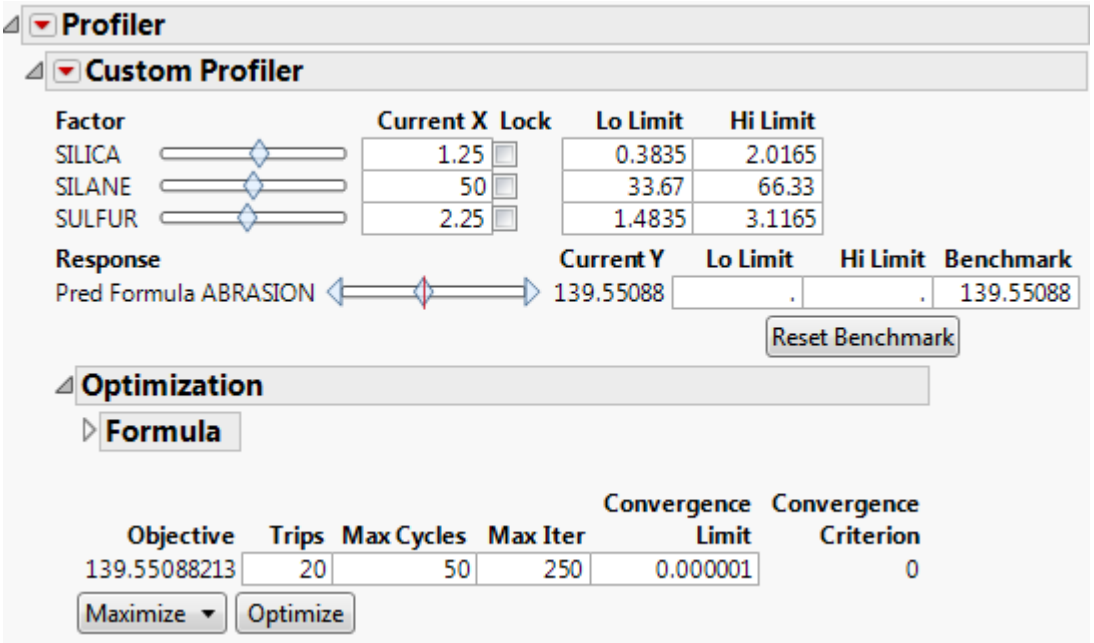
Chapter 7

Custom Profiler

Explore Response Surfaces Using a Numerical Calculator

The Custom Profiler enables you to optimize factor settings without graphical output. This is useful for large problems that would have too many graphs to visualize well.

Figure 7.1 Custom Profiler Example



Contents

Overview of the Custom Profiler 143
Example of the Custom Profiler 143
Launch the Custom Profiler Platform 145
The Custom Profiler Report 146
Factor Settings and Controls 146
Response Settings and Controls 147
Optimization Settings and Controls 147
Custom Profiler Platform Options 148

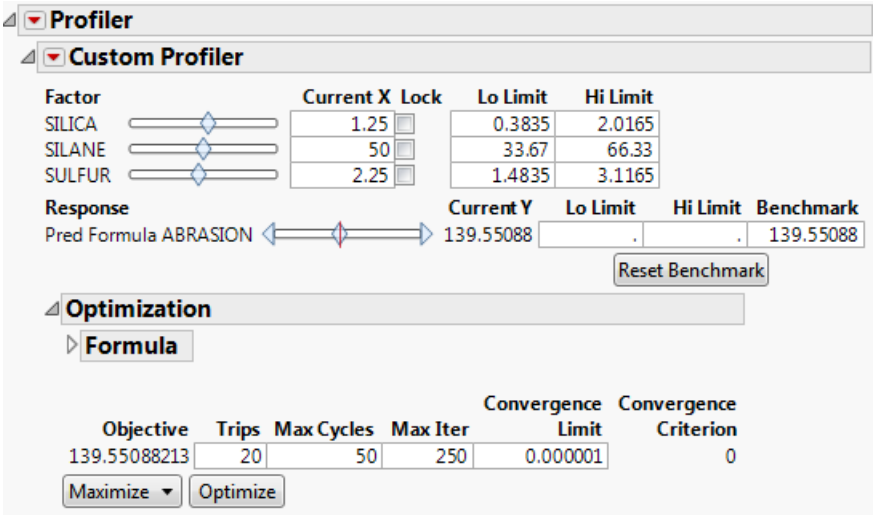
Overview of the Custom Profiler

The Custom Profiler enables you to optimize factor settings without graphical output. The Custom Profiler can be used for problems of any size. It is especially useful for large problems where the standard graphical profiler has too many graphs to visualize well.

The Custom Profiler report has many fields in common with other profilers. The Benchmark field holds a value, or benchmark, of the predicted response. You can compare new results to the benchmark value and update the value based on the current factor settings.

The Optimization report enables you to specify the formula to be optimized and specifications for the optimization.

Figure 7.2 Custom Profiler

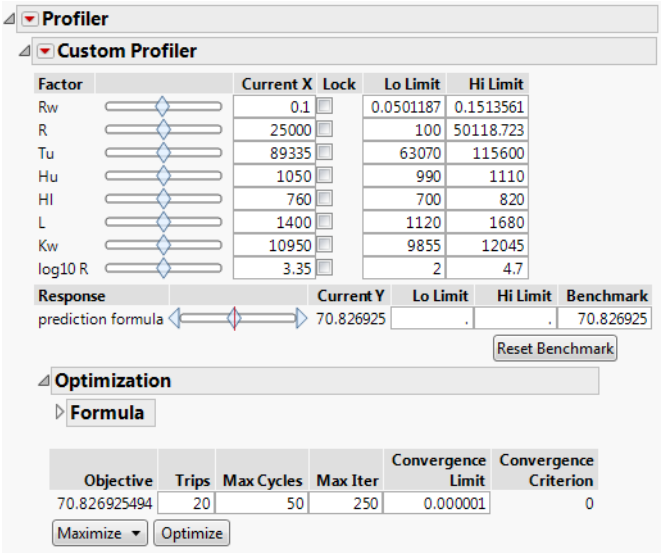


Example of the Custom Profiler

This example uses data that demonstrates the flow of water through a borehole that is drilled from the ground surface through two aquifers. You want to optimize the predicted value.

1. Select **Help > Sample Data Library > Design Experiment** and open Borehole Latin Hypercube.jmp
2. Select **Graph > Custom Profiler**.
3. Select prediction formula and click **Y, Prediction Formula.**
4. Click **OK**.

Figure 7.3 Custom Profiler Report



Note that the Benchmark value is 70.83. The Current Y value is also 70.83. This value is the predicted response with all factors set to their mean values.

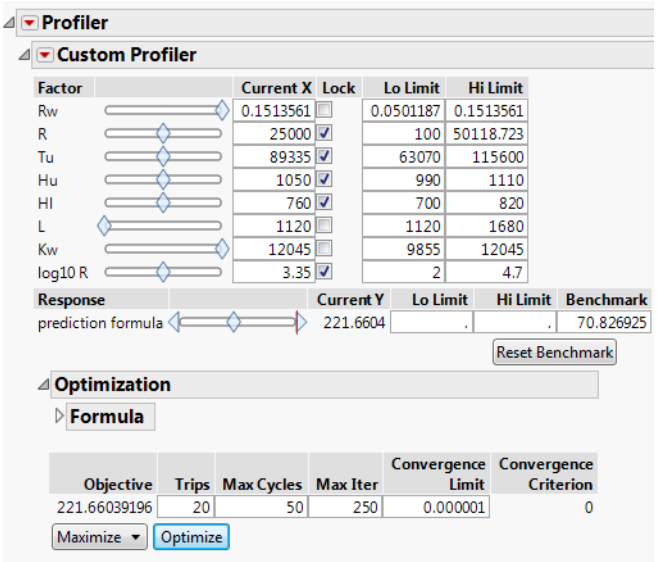
You would like to find the X values of the factors that optimize the predicted value. However, of the factors in the model, only those related to the borehole can be manipulated in practice. The other factors are out of human control. You can lock the uncontrollable factors at their mean values and optimize the factors that can be controlled.

Note: To view the description of a factor, right-click the corresponding column in the data table and select **Column Info**.

---

5. In the Custom Profiler Report, select the Lock check box next to R, Tu, Hu, HI, and log10 R.
6. In the Custom Profiler Report, click **Optimize**.

Figure 7.4 Optimized Custom Profiler Report



The optimization routine found an optimum predicted response at 221.66. In order to obtain the optimum of 221.66, Rw and Kw are set to their maximum values and L is set to its minimum value. The optimum is greater than the initial Benchmark value of 70.83.

Launch the Custom Profiler Platform

The Custom Profiler can be accessed in the following ways:

- The Custom Profiler can be accessed directly from the Graph menu. When you access the Custom Profiler in this way, the Custom Profiler launch window appears. See "Profiler Launch Windows".
- The Custom Profiler can be accessed as a red triangle menu option in many modeling platforms. See "Where to Find JMP Profilers" for more information about the availability of the Custom Profiler in different platforms.
- The Custom Profiler can be accessed from the Model Comparisons platform. Select Profiler from the Model Comparisons red triangle menu. Then, select Custom Profiler from the Profiler red triangle menu.
- JMP PRO The Custom Profiler can be accessed from the Formula Depot platform. Select Profiler from the Formula Depot red triangle menu and select the models to be profiled. Then, select Custom Profiler from the Profiler red triangle menu.

The Custom Profiler Report

The initial Custom Profiler report shows settings and controls for the factors, responses, and optimization.

- "Factor Settings and Controls"
- "Response Settings and Controls"
- "Optimization Settings and Controls"

Factor Settings and Controls

Figure 7.5 Factor Settings and Controls

Factor		Current X	Lock	Lo Limit	Hi Limit
Rw		0.1513561		0.0501187	0.1513561
R		100		100	50118.723
Tu		115600		63070	115600
Hu		1110		990	1110
HI		700		700	820
L		1120		1120	1680
Kw		12045		9855	12045
log10 R		2		2	4.7

Factor The list of model factors.

Current X The current value of each factor. Click in a box to change the value of a factor. The slider controls can also be used to change factor settings.

Lock Enables you to lock a factor so that it is fixed when the optimization is performed. You can change a locked factor using the slider or clicking in the box in the Current X column. The lock applies only to the optimization.

Nominal Column Unlabeled column to the right of the Lock column that lists the current value of nominal factors.

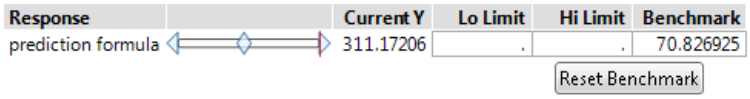
Note: The Current X column for nominal factors displays a coded (numeric) value for the current nominal factor.

Lo Limit The lower limit for each factor. Click in a box to change the value.

High Limit The upper limit for each factor. Click in a box to change the value.

Response Settings and Controls

Figure 7.6 Response Settings and Controls



Response The list of one or more responses.

Current Y The predicted response based on the current X settings. This value updates as the factor settings are changed.

Lo Limit Enables you to set a lower limit for your response.

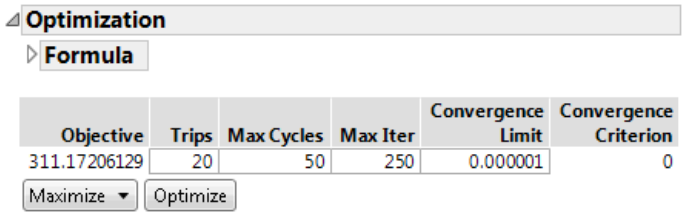
High Limit Enables you to set an upper limit for your response.

Benchmark A saved predicted value of the response. Initially, this value is set to the predicted value when all factors are at their mean value.

Reset Benchmark Updates the benchmark value to the current predicted value.

Optimization Settings and Controls

Figure 7.7 Optimization Settings and Controls



Formula The formula to be optimized. When a single response is used, the expression is the response column name. When multiple responses are used, the expression is a sum of desirability functions. You can edit the objective expression.

Objective The current value of the objective function. When a single response is used, the objective expression is the predicted response. When multiple responses are used, the objective expression is the desirability function. For more information about desirability functions, see "Desirability Profiling and Optimization".

Trips The number of random starts in the optimization algorithm. Each trip restarts the algorithm at a different starting point. This guards against finding local solutions.

Max Cycles The maximum number of cycles used in the optimization algorithm. Each cycle is single pass through the input parameters and optimizes each one individually.

Max Iter The maximum number of optimization iterations per cycle for each input parameter.

Convergence Limit The upper limit for the convergence criterion for the optimization algorithm. If the convergence criterion becomes less than this value, the algorithm stops.

Convergence Criterion The value of the convergence criterion for the optimization algorithm.

Maximize Enables you to choose to maximize or minimize the objective function.

Optimize Starts the optimization algorithm.

Custom Profiler Platform Options

Factor Settings Contains options identical to the Factor Settings submenu in the Prediction Profiler. See "Factor Settings".

Log Iterations Creates a data table that contains iterations of the optimization algorithm. The data table appears after the **Optimize** button is clicked.

Alter Linear Constraints Enables you to add, change, or delete linear constraints. The constraints are used in the Custom Profiler. See "Linear Constraints".

Save Linear Constraints Saves existing linear constraints as a data table script that is named Constraint. See "Linear Constraints".

Simulator Launches the Simulator. See "Simulator".

Chapter 8

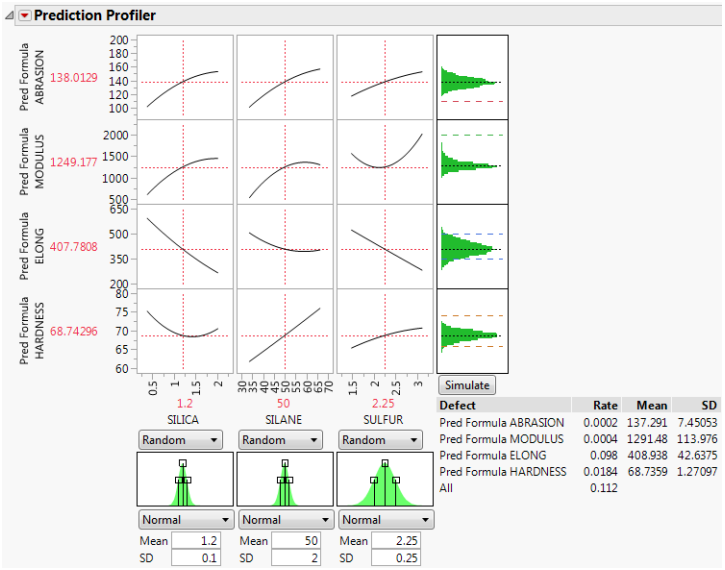
Simulator

Explore the Effects of Variation on Responses

Simulation enables you to discover the distribution of model outputs as a function of the random variation in the model inputs and noise. The Simulator in the profilers enables you to define random inputs, run simulations, and produce output tables of simulated values.

In the Prediction Profiler, the Simulator is integrated into the graphical layout. Factor specifications are displayed below each factor's profile. A simulation histogram is shown for each response.

Figure 8.1 Prediction Profiler with Simulator Example



Contents

Overview of the Simulator 151
Example of the Simulator 151
Launch the Simulator 154
Factors Simulation Settings 154
Response Simulation Settings 157
The Simulator Report 157
The Simulator Report Options 158
Defect Profiler 159
Introduction to Tolerance Design 159
Graph Scale 160
Defect Rates 160
Simulation Method and Details 161
Notes 161
Defect Parametric Profiler 162
Simulation Experiment 163
Specification Limits 163
Additional Examples of the Simulator 163
Example of the Defect Profiler 164
Example of Stochastic Optimization 167
Example of Simulating General Formulas 176
Statistical Details for the Simulator 179
Normal Weighted Distribution 179

Overview of the Simulator

Simulation enables you to discover the distribution of model outputs as a function of the random variation in the model inputs and noise. The Simulator in the profilers enables you to define random inputs, run simulations, and produce output tables of simulated values.

For example, you can use the simulator to model the defect rate of a process and to explore the robustness of the defect rate with respect to variation in the model factors. If specification limits have been set for the responses, they are displayed in the simulation output. The inclusion of specification limits enables you to perform a prospective capability analysis of the simulated model. For information about adding specification limits, see *Using JMP*.

You can access the Simulator through the Prediction, Contour, and Custom Profilers. The appearance of the Simulator depends on the profiler platform in which it appears.

Prediction Profiler

In the Prediction Profiler, the Simulator is integrated into the graphical layout. Factor specifications are displayed below each factor's profile. A simulation histogram is shown on the right for each response.

Contour Profiler and Custom Profiler

In the Contour and Custom profilers, the Simulator is not integrated into the graphical layout. There are no integrated histograms, and the interface is textual. However, the process is the same and the resulting output tables are the same as in other profilers.

Note: The Simulator supports mixture terms. The simulation assigns the random value to each mixture factor and then proportionally adjusts the values to maintain the factor constraints.

Example of the Simulator

Use sequencing to examine how the distribution of the response changes when the mean (sequencing location) and variability (sequencing spread) of the inputs change.

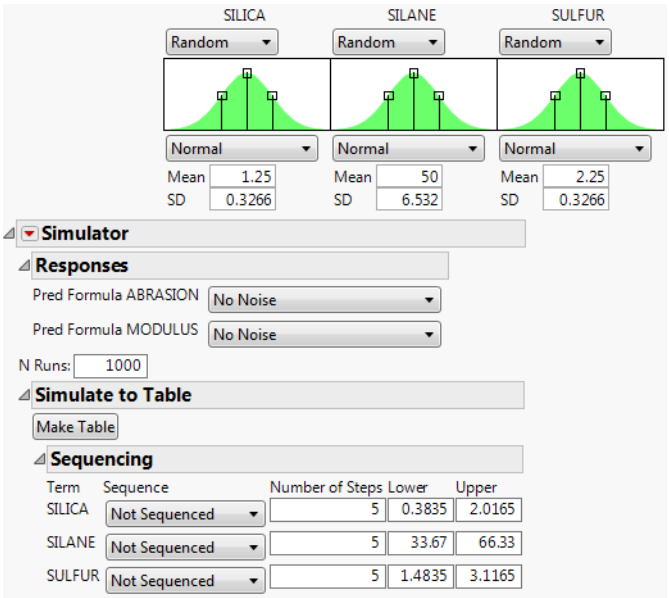
1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Graph > Profiler**.
3. Select Pred Formula ABRASION and Pred Formula MODULUS and click **Y, Prediction Formula**.
4. Click **OK**.

5. Click the Prediction Profiler red triangle and select **Simulator**.
6. Beneath each factor select **Random**.

Note: The default random distribution is a normal distribution.

7. Change the **N Runs** value to 1000.
8. Open Simulate to Table outline and then open the Sequencing outline.

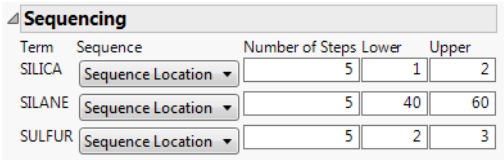
Figure 8.2 Simulator Settings



You want to examine how the responses change as the input factors change locations. To explore changes in factor mean, use sequence location. To instead explore changes in the variability of the factor, use sequence spread.

9. For SILICA, select **Sequence Location**. Leave the number of steps set to 5. Set the **Lower** limit to 1 and the **Upper** limit to 2.
10. For SILANE, select **Sequence Location**. Leave the number of steps set to 5. Set the **Lower** limit to 40 and the **Upper** limit to 60.
11. For SULFUR, select **Sequence Location**. Leave the number of steps set to 5. Set the values **Lower** limit to 2 and the **Upper** limit to 3.

Figure 8.3 Sequencing Settings



12. Click **Make Table**.

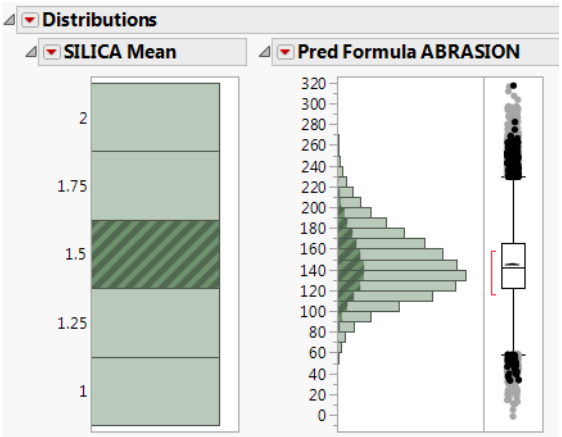
    The SILICA Mean, SILANE Mean, and SULFUR Mean columns contain five steps across their respective range of values. For example, the Silica Mean values are 1, 1.25, 1.5, 1.75, and 2. The SILICA, SILANE, and SULFUR columns are the simulated values from a normal distribution with a mean defined by the corresponding mean columns and a fixed standard deviation. Pred Formula ABRASION and Pred Formula MODULUS values are calculated for set of simulated values, so that you can explore how the responses change as the factor values change.

13. Select **Analyze > Distribution**.

14. Select SILICA Mean and Pred Formula ABRASION and click **Y, Columns**.

15. Click **OK**.

Figure 8.4 Distribution of SILICA Mean by Pred Formula ABRASION



Click a histogram bar that corresponds to a SILICA Mean to see how the predicted value for abrasion varies across the simulated silane and sulfur ranges given the selected value of silica.

Launch the Simulator

The Simulator can be accessed as a red triangle menu option in the Prediction Profiler, the Contour Profiler, and the Custom Profiler. When you launch the Simulator from the Prediction Profiler, factor specifications are displayed below each factor's profile. When you launch the Simulator from the Contour or Custom profilers, factor specifications appear in a Factors report, located below the respective profiler report. See "Factors Simulation Settings".

Once the Simulator is accessed, you can define simulation parameters for the responses in the Responses report. See "Response Simulation Settings". The Responses report also enables you to set the number of simulations.

A histogram is also added to the right of each response when the Simulator is accessed from the Prediction Profiler. Below the histograms is a **Simulate** button. Once you specify settings for the factors and responses, click **Simulate** to simulate response values. The histograms of the simulated responses are updated each time the simulate button is clicked. To simulate responses in the Contour and Custom profilers, open the Simulate to Table report and click **Make Table**. See "The Simulator Report".

Factors Simulation Settings

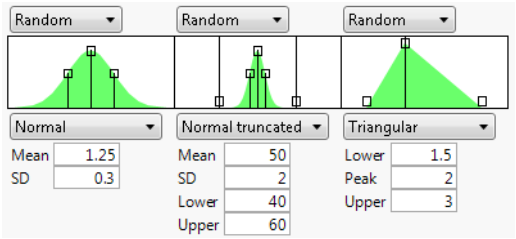
By default, factors (inputs) and responses (outputs) are defined by the functions that you select when the Profiler is launched. Simulation options enable you to modify how your factors and responses are simulated. There are controls for location, error, and noise.

Fixed Fixes the factor at the current value in the profiler for all simulation runs.

Random Draws a random value of the factor for each simulation run from the specified distribution and distributional parameters. When you select Random, distribution options appear. For more information about the options, see "Continuous Factors" and "Categorical Factors".

Note: The default random distribution is a normal distribution with a mean at the current factor setting and a standard deviation estimated by the range of the factor divided by 5. The mean of the normal distribution is automatically updated if you change the factor setting in the profiler. There are other random distributions that also update parameters automatically when the factor settings are changed. The Triangular distribution updates all three parameters, the Cauchy distribution updates the mode, and the Johnson Su distribution updates θ.

Figure 8.5 Example of Three Random Distributions



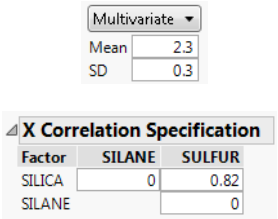
Expression Generates values for the factor based on a JSL expression. This gives you flexibility to use a random distribution of your choice. For example, you could create a censored normal distribution that guaranteed nonnegative values with an expression such as the following:

```
Max(0,RandomNormal(5,2))
```

After entering the expression, click the **Reset** button to submit the expression.

Multivariate Generates values for the factor based on a multivariate normal distribution. Specify the mean and standard deviation for each factor. The correlation matrix is defined separately in the X Correlation Specification report. You can use this option to accommodate correlated factors.

Figure 8.6 Defining a Correlation Matrix for the Multivariate Option



Continuous Factors

Many of the distributions available use standard random functions that are described in *Using JMP*. Descriptions of the specialized distributions available follow:

Normal weighted Generates values from a weighted normal distribution with the given mean and standard deviation. The weighting is a specific sampling scheme used to simulate rare events from the tails of the distribution. This is a good choice when you want to simulate very low defect rates. See "Statistical Details for the Simulator".

Normal truncated Generates values from a normal distribution limited by lower and upper limits. Any randomly generated value that exceeds a limit is discarded and another value is generated. This is a good choice when you want to simulate an inspection system where inputs that do not satisfy specification limits are discarded.

Normal censored Generates values from a normal distribution limited by lower and upper limits. Any randomly generated value that exceeds a limit is set to that limit, putting a density mass at the limits. This is a good choice when you want to simulate systems where inputs that do not satisfy specification limits are reworked until they are at the specification limit.

Sampled Generates factor values by selecting a value at random from that factor's column in the data table.

External Generates factor values by selecting a value at random from a column in another table. You are prompted to choose the table and column.

Note: The Aligned check box appears when you select Sampled or External. It is used when two or more factors are set to Sampled or External. When checked, the random draws come from the same row of the table. This maintains the correlation structure between two columns. If the Aligned option is used to associate two columns in different tables, the columns must have the same number of rows.

Categorical Factors

If a factor is categorical, then the random distribution is characterized by probabilities specified for each category. By default, the probabilities are set to the observed probabilities in the data table. You can change the probabilities using the handles in the plot or by changing the numerical values in the Prob column. The probabilities must sum to 1.

Note: If the probabilities that you select do not sum to 1, they are automatically regularized to sum to 1 once you click the Simulate button.

When the simulation for a factor is set to Random, a graphical representation of the random distribution density is shown. The graph shows the form of the density for the continuous distributions, and provides control points that can be dragged to change the distribution. The drag points for the Normal are the mean and the mean plus or minus one standard deviation. The Normal truncated and censored add points for the lower and upper limits. The Uniform and Triangular have limit control points, and the Triangular adds the mode.

Response Simulation Settings

Simulation settings for the responses enable you to simulate random noise in the responses. Often your factors explain part of the variation in the response with the rest of the variation of the response attributed to random noise. This is how the random noise is specified in your simulation.

No Noise Generates the response from the model based on the factor settings. No random noise added to the response.

Add Random Noise Adds a normal random value with mean zero and specified standard deviation to the generated response.

Add Random Weighted Noise Adds an error term to the generated response based on a weighted sampling scheme used to simulate rare events from the tails of the distribution.

Add Multivariate Noise Adds an error term to the generated response based on a multivariate normal distribution. Specify the standard deviation for the response. Specify the correlation structure between responses in the Y Correlation Specification report.

Random by Model (Available only within the Standard Least Squares and Generalized Regression personalities of the Fit Model platform.) Adds an error term to the generated response based on the model distribution specified in the Fit Model launch window.

The Simulator Report

The Simulator report contains the following items:

Simulate to Table Contains a Make Table button that saves the results of the simulation to a data table. If a response has specification limits, the table includes a Y In Spec column that indicates whether the simulated response is within the specification limits. The table also includes an Objective column, labeled Obj. If you added Desirability functions to your profiler prior to simulation, this column is the evaluation of the desirability function. If the Desirability functions were not present prior to simulation, the column contains all missing values.

Sequencing (Available only when the distribution is Normal, Uniform, or Triangular random.) Sequencing enables you to simulate across a range of locations or spreads for each factor. The simulations are saved to a data table. Select an option to sequence the location or spread of each factor. Specify the number of steps and the range for the sequencing. For each combination of the number of steps, the simulation runs N Runs. If you have two terms, each with 5 steps, and N Runs = 100, the resulting simulation table has 2,500 rows.

The Simulator Report Options

The Simulator red triangle menu contains the following options:

Automatic Histogram Update Toggles the histogram update. When this option is selected, the histograms update with new simulated values when you drag factor setting handles. By default, this option is not selected.

Defect Profiler (Available only when spec limits are specified.) Shows the defect rate as an isolated function of each factor. See "Defect Profiler".

Defect Parametric Profile (Available when the Defect Profiler is launched.) Shows the defect rate as an isolated function of the parameters of each factor's distribution. See "Defect Parametric Profiler".

N Strata (Appears only when you press Shift before clicking the Simulator red triangle menu.) Enables you to specify the number of strata for the Normal Weighted distribution. See "Statistical Details for the Simulator".

Set Random Seed (Appears only when you press Shift before clicking the Simulator red triangle menu.) Enables you to specify a seed for the simulation starting point. This enables the simulation results to be reproducible, unless the seed is set to zero. The seed is set to zero by default. If the seed is nonzero, then the latest simulation results are output when the Make Table button is clicked.

Simulation Experiment (Available for Normal, Uniform, and Triangular random distributions.) Runs a designed simulation experiment on the locations of the factor distributions. A window appears that enables you to specify the number of design points, the portion of the factor space to be used in the experiment, and which factors to vary in the experiment. Factors that are not selected for the experiment are set to their current Profiler value. See "Simulation Experiment".

Spec Limits Opens a table to set or edit specification limits for the responses. Click the **Save** button to save specification limits to the response column properties in the data table. See "Specification Limits".

Defect Profiler

To use the Defect Profiler, spec limits must be defined for at least one response. The Defect Profiler shows the probability of an out-of-spec output defect as a function of each factor while the other factors vary randomly. This is used to help visualize the changes in a factor's distribution that the process is most sensitive to, often as part of tolerance design.

Specification limits are used to define a defect, and the random distributions assigned to each factor are used to simulate responses.

Define at least one of the factors as Random in the simulation settings to obtain useful results. Otherwise, the simulation outputs are constant and the defect rate is zero if the outcome falls within spec or one if the outcome falls out of spec.

Tip: If you need to estimate very small defect rates, consider the Normal weighted random option. This random distribution estimates stable defect rates of just a few parts per million with only a few thousand simulation runs.

Introduction to Tolerance Design

Tolerance design is the investigation of how defect rates can be controlled by controlling variability in the input factors.

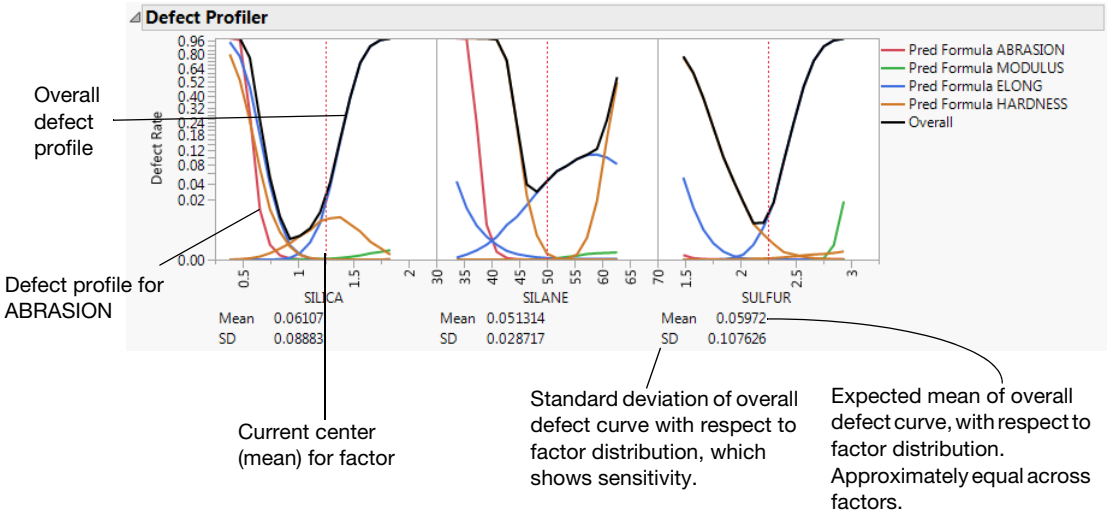
Input factors have variation. Specification limits are used to define acceptable ranges for each input. The variability in the input factors impacts the outputs. Specification limits are also used to define the acceptable range for output variables.

Sometimes, a tolerance design study shows that spec limits on an input are unnecessarily tight. Loosening such limits can result in reduced costs without sacrificing product quality. In these cases, tolerance design can save money.

In other cases, a tolerance design study might find that either tighter limits or different targets result in higher quality. In all cases, it is valuable to understand how the inputs and their variability impact that the defect rates.

The Defect Profiler shows the defect rate as a function of each factor fixed at the specified mean value while all other factors are varied according to their random specification. If there are multiple outputs with spec limits, each has a different colored defect rate curve. A black curve shows the overall defect rate.

Figure 8.7 Defect Profiler



Graph Scale

Defect rates are shown on a cubic root scale, so that both large and small defect rates are shown in some detail.

Defect Rates

The mean and standard deviation (SD) of the simulated overall defect rate are reported below each defect profile plot. This mean is calculated by integrating the defect profile curve with the specified factor distribution. Because of numerical error in the estimations, the overall mean defect rate reported under each factor can differ slightly.

The defect rates that are reported in the Defect Profiler are estimates of the simulated overall defect rate. This rate is also reported in the Rate column of the defect table that is in the Prediction Profiler outline.

Note: The defect table is added to the Prediction Profiler outline after you run a simulation and one or more of your responses has defined spec limits.

Because the rate estimates are obtained differently, by integration and simulation, they might differ slightly. If they are very different, you might consider increasing the number of simulation runs. In addition, verify that the range of the factor scale is wide enough so that the integration covers the distribution well.

The standard deviation is a measure of the sensitivity of the defect rate to the factor. It is small when either the factor profile is flat or the factor distribution has a small variance. The larger the standard deviation, the larger the impact that changes in that factor have on the defect rate variability. Comparing SDs across factors enables you to select factors for improvement to reduce defect rates.

The mean and SD values are updated when you change the factor distribution settings. This is one way to explore how to reduce defects as a function of one particular factor at a time. You can click and drag a handle point on the factor distribution and watch the mean and SD change as you drag. However, changes are not updated across all factors until you click the Rerun button to generate updated simulation runs.

Simulation Method and Details

Suppose you want a defect profile for factor X1, in the presence of random variation in X2 and X3. A series of n = N Runs simulation runs is done at each of k points in a grid of equally spaced values of X1. (By default, k is set at 17.) At each grid point, suppose that there are m defects due to the specification limits. At that grid point, the defect rate is m/n. These defect rates are connected and plotted as a continuous function of X1.

Notes

- The profile curve is not recalculated automatically when distributions change. Click **Rerun** to update the curve.
- Defect Profiling does not address the general optimization problem of optimizing quality against cost, given functions that represent all aspects of the problem. This more general problem would benefit from a surrogate model and space filling design.
- The defect profiles tend to be uneven when they are low. This is due, in part, to the use of the cubic scale where differences in low values are exaggerated. Jagged defect profiles could be due to limited simulation runs. If the overall defect curve (black line) is smooth, and the individual defect rates are fairly consistent, then your simulation probably has enough runs to provide a stable solution. If the overall defect rate curve is jagged, then consider increasing the number of runs. 20,000 runs is generally enough to stabilize the curves.

Defect Parametric Profiler

The Defect Parametric Profiler report shows the impact of process changes on the defect rate. The impacts are based on the simulation parameter settings for each factor. Four scenarios are considered.

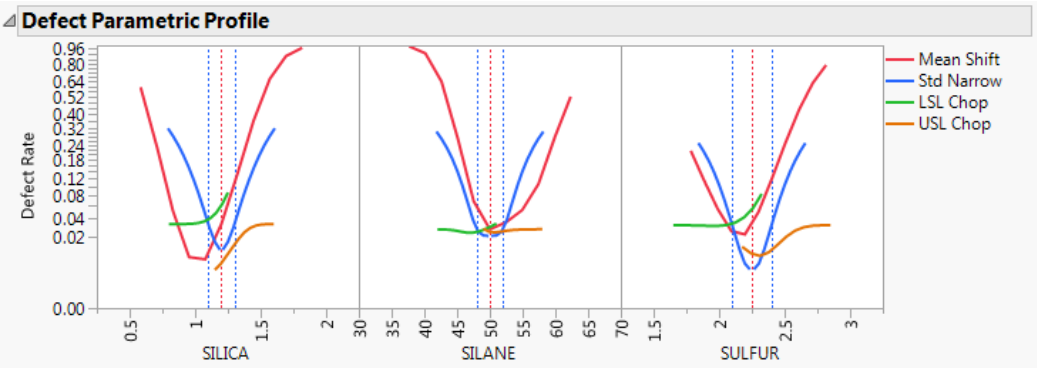
Mean Shift The impact of shifting the mean is shown by a red curve. The current mean is shown by a red dotted vertical line.

Std Narrow The impact of a reduction in variability is shown by a blue curve. The dotted blue vertical lines are set at the mean plus and minus one standard deviation. The minimum value on the curve corresponds to the defect rate when there is no variability.

LSL Chop The impact of inspection to remove all parts below the lower specification limit is shown by a green curve.

USL Chop The impact of inspection to remove all parts above the upper specification limit is shown by an orange curve.

Figure 8.8 Defect Parametric Profile



Simulation Experiment

Use the Simulation Experiment option to run a designed simulation experiment on the locations of the factor distributions. When you select Simulation Experiment, you to specify the number of design points, the portion of the factor space to be used in the experiment, and which factors to include in the experiment. For factors that are not included in the experiment, the current value set in the Profiler is used in the experiment.

The experimental design is a Latin Hypercube. At each design point, N Runs random draws are generated with the design point serving as the center of the random draws. The shape and variability come from the specified distributions. The output has one row for each design point. The responses include the defect rate for each response with spec limits and an overall defect rate. You can use a Gaussian Process model to model the overall defect rate from the simulation experiment. For more information about Gaussian Process models, see *Predictive and Specialized Modeling*.

Specification Limits

To add specification limits to a variable, right-click the desired column and select **Column Properties > Spec Limits.** In the Column Properties window, there are boxes to enter values for the Lower Spec Limit, Target, and Upper Spec Limit. Alternatively, you can define Specification Limits using the **Spec Limits** option in the Simulator red triangle menu.

The profilers support specification limits on the responses and provide a number of features:

- In the Profiler, if the Response Limits are not set up in the input data table to provide desirability coordinates, JMP looks for a Spec Limits column property and constructs desirability functions appropriate to those spec limits. For more information about Response Limits see *Using JMP*.
- If you use the Simulator to output simulation tables, JMP includes the Spec Limits column properties in the output data tables. This makes accounting for defect rates and capability indices easy.
- Adding spec limits enables the Defect Profiler.

Additional Examples of the Simulator

- "Example of the Defect Profiler"
- "Example of Stochastic Optimization"
- "Example of Simulating General Formulas"

Example of the Defect Profiler

To demonstrate a possible workflow with the Defect Profiler, we use Tiretread.jmp. The experimental data in the Tiretread.jmp sample data table comes from an experiment to study the effects of SILICA, SILANE, and SULFUR on four measures of tire tread performance.

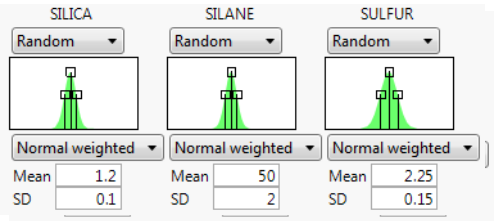
Add Specification Limits in the Simulator

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Graph > Profiler**.
3. Select Pred Formula ABRASION, Pred Formula MODULUS, Pred Formula ELONG, and Pred Formula HARDNESS and click **Y, Prediction Formula**.
4. Click **OK**.
5. Click the Prediction Profiler red triangle and select **Simulator**.
6. Click the Simulator red triangle and select **Spec Limits**.

   Note: If your columns have spec limits saved as a column property, those spec limits are shown in the Simulator Spec Limits table.

7. Set the spec limits:
   - For ABRASION set the LSL to 110.
   - For MODULUS set the USL to 2000.
   - For ELONG set the LSL to 350 and the USL to 550.
   - For HARDNESS set the LSL to 66 and the USL to 74.
8. Click **Save** to save the Spec Limits to the data table.
9. Beneath each factor, select **Random**.
10. For each factor, enter the random specifications as shown in Figure 8.9.

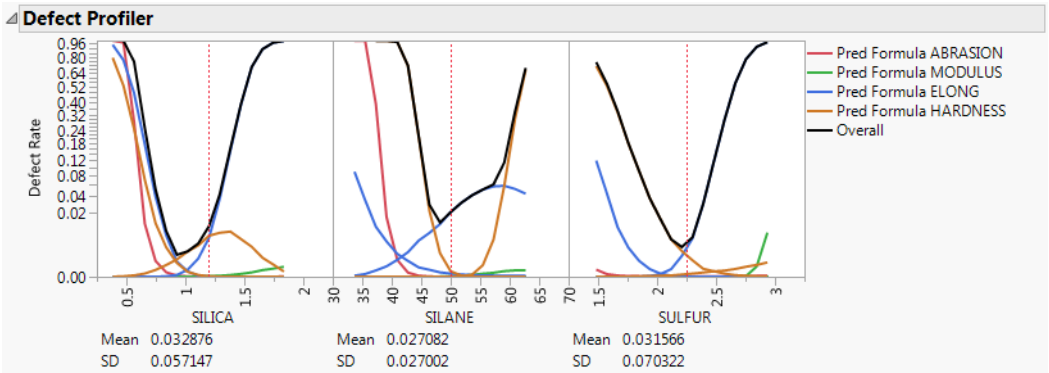
Figure 8.9 Profiler Random Specifications



Open the Defect Profiler and Defect Parametric Profile

1. Click the Simulator red triangle and select **Defect Profiler.**

Figure 8.10 Defect Profiler



Note: The defect profiler is based on simulation with random inputs. The values that you obtain might differ from those shown in Figure 8.10.

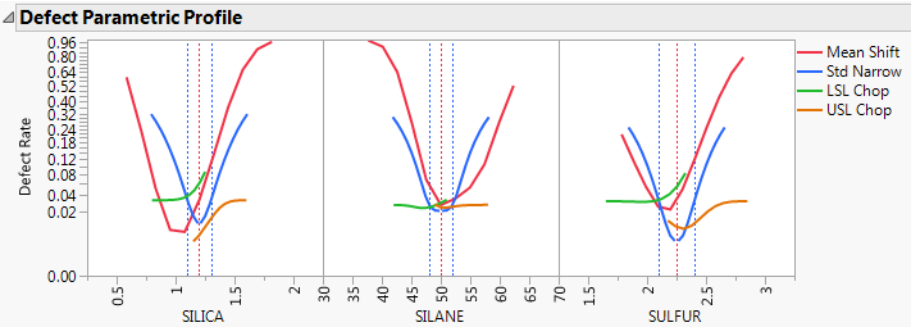
The black curves show the overall defect rate for each factor fixed at the X axis value while all other factors vary.

Consider the overall curve for SILICA. As silica varies, the defect rate goes from the lowest rate of 0.001 when silica is about 1 and increases quickly up to a defect rate of nearly 1 as silica increases or decreases from 1. However, SILICA is itself random. If you integrate the density curve of SILICA, you would estimate the average defect rate to be about 0.03, which is shown as the Mean for SILICA. This estimate of the overall defect rate estimated by the simulation is shown in the defect table under the simulation histograms. The Mean value for the overall defect rate for all factors are similar.

The Defect Profiler also includes an estimate of the standard deviation of the defect rate with respect to the variation in each factor. This value (labeled SD) is 0.057 for SILICA. The standard deviation is related to the sensitivity of the defect rate with respect to the distribution of that factor. Comparing the SD values across the three factors, the SD for SULFUR is higher than the SD values for SILICA and SILANE. This indicates that to improve the defect rate, shifting the distribution in SULFUR should have the greatest effect. A distribution can be shifted by changing its mean, changing its standard deviation, or by truncating the distribution by rejecting inputs that do not meet certain specification limits.

2. Click the Simulator red triangle and select **Defect Parametric Profile**.

Figure 8.11 Defect Parametric Profile



Consider SULFUR and note that the current defect rate (0.03) is represented in four ways corresponding to each of the four curves in the Parametric Profiler.

For the red curve, Mean Shift, the current rate is where the red curve intersects the vertical red dotted line. The Mean Shift curve represents the change in overall defect rate as the mean of SULFUR changes. One opportunity to reduce the defect rate is to shift the mean slightly to the left. If you use the crosshair tool on this plot, you see that a shift down in the mean reduces the defect rate to about 0.02.

For the blue curve, Std Narrow, the current defect rate is where the blue curve intersects the two dotted blue lines. The Std Narrow curves represent the change in defect rate as the standard deviation changes. The dotted blue lines represent one standard deviation below and above the current mean. The blue curve is drawn symmetrically around the center. At the center, the blue curve reaches a minimum, representing the defect rate for a standard deviation of zero. That is, if we totally eliminate variation in SULFUR, the defect rate is about 0.003. This is much lower than the current rate of 0.03. If you look at the other defect parametric profile curves, you can see that this is better than reducing variation in the other factors, something that we suspected by the SD value for SULFUR.

For the green curve, LSL Chop, there are no interesting opportunities for improvement in the defect rate. The green curve is above current defect rates for the entire range of the curve. This indicates that reducing the variation by rejecting parts with too-small values for SULFUR does not help reduce the defect rate.

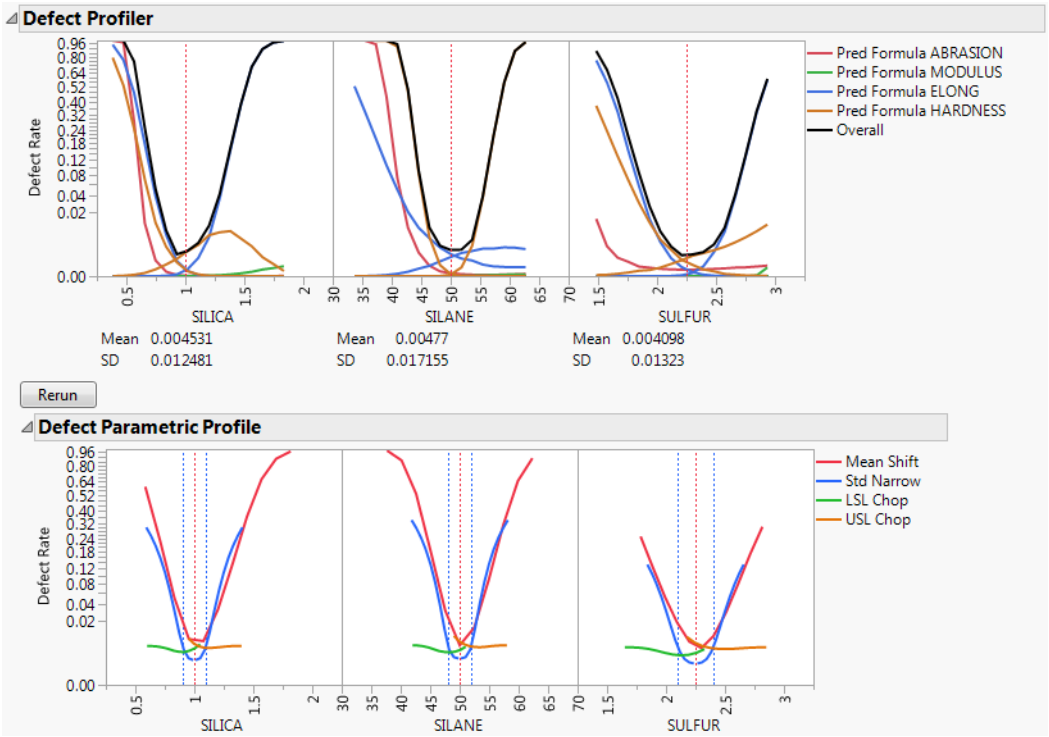
The orange curve, USL Chop, suggests a way to improve the defect rate. Reading the curve from the right, the curve starts out at the current defect rate (0.03). Then as you start rejecting more parts by decreasing the USL for SULFUR, the defect rate improves. However, moving a spec limit to the center of the distribution is equivalent to throwing away half the parts, which might not be a practical solution.

Looking at all the opportunities over all factors, it now looks like there are two good options for further investigation. You could shift the mean of SILICA to about 1 or reduce the variation in SULFUR. Because it is generally easier in practice to change a process mean than a process variation, the best first adjustment might be to shift the mean of SILICA to 1.

3. In the Prediction Profiler, adjust the Mean of SILICA from 1.2 to 1.0.
4. Below the Defect Profiler, click **Rerun**.

   The Defect Profiler updates based on the adjusted SILICA value.

Figure 8.12 Adjusted Defect Rates

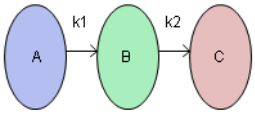


By shifting the mean of SILICA from 1.2 to 1.0, the defect rate has decreased from 0.03 to about 0.004, which is a good improvement. Investigating other defect reduction scenarios is easy to do by making changes to the distributions and rerunning the simulations.

Example of Stochastic Optimization

This example is adapted from Box and Draper (2007) and uses Stochastic Optimization.jmp. A chemical reaction converts chemical A into chemical B and B into C. The resulting amount of chemical B is a function of reaction time and reaction temperature. Use the profiler and simulator to explore factor settings to optimize the yield of B.

Figure 8.13 Chemical Reaction



For this reaction the yield of B can be computed using the Arrhenius laws. The column Yield contains the formula for yield. The formula is a function of Reaction Time (hours) and reaction rates k1 and k2. The reaction rates are a function of Reaction Temperature (degrees Kelvin) and known physical constants θ_1, θ_2, θ_3, θ_4. Therefore, Yield is a function of Reaction Time and Reaction Temperature.

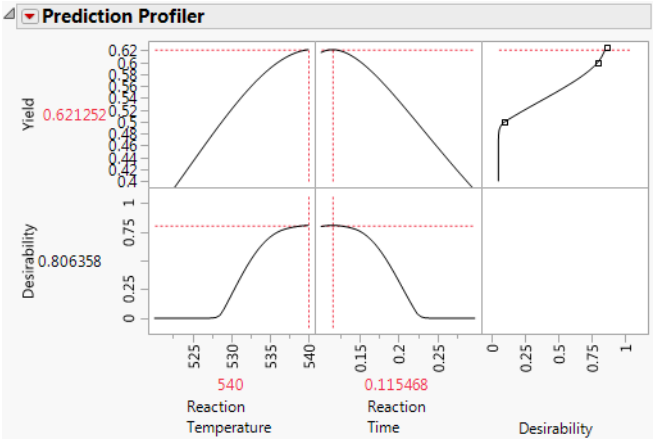
Maximize Yield Using Desirability Function

1. Select **Help > Sample Data Library** and open Stochastic Optimization.jmp.
2. Select **Graph > Profiler**.
3. Select Yield and click **Y, Prediction Formula**.
4. Click **Expand Intermediate Formulas** and then click **OK**.

   The Prediction Profiler with Desirability Functions enabled appears. For more information about Desirability Functions, see the "Desirability Profiling and Optimization".
5. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

   The Prediction Profiler maximizes Yield and sets the graphs to the optimum value of Reaction Time and Reaction Temperature.

Figure 8.14 Yield Maximum



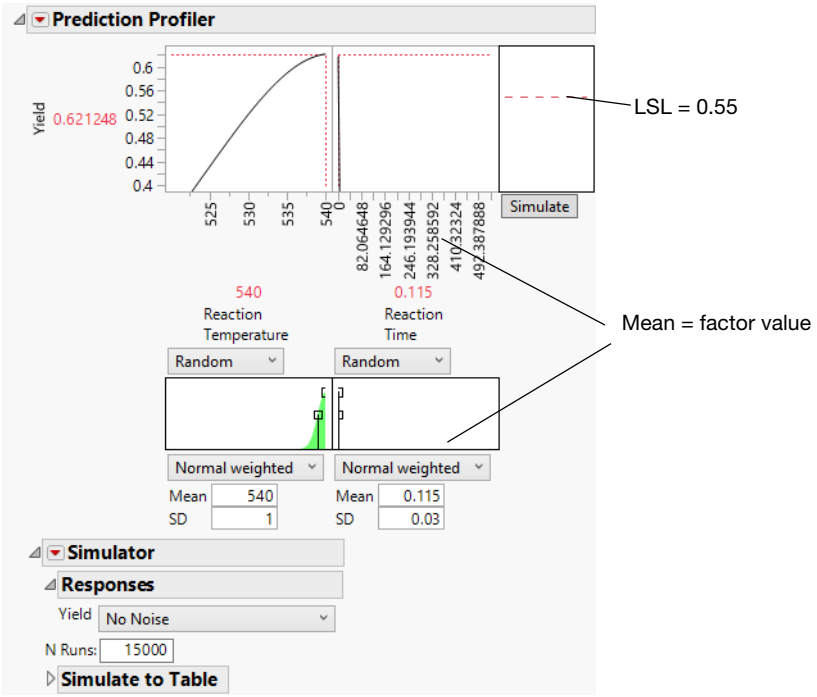
The maximum Yield is approximately 0.62 at a Reaction Time of 0.115 hours and Reaction Temperature of 540 degrees Kelvin, or hot and fast. (Your results might differ slightly due to random starting values in the optimization process.)

Simulate Defect Rate for Maximum Yield

In a production environment, process inputs cannot always be controlled exactly. What happens to Yield if the inputs (Reaction Time and Reaction Temperature) have random variation? Furthermore, if Yield has a spec limit, what percent of batches are out of spec? The Simulator can help you investigate the variation and defect rate for Yield, given variation in Reaction Time and Reaction Temperature.

1. Click the Prediction Profiler red triangle and deselect **Optimization and Desirability > Desirability Functions**.
2. Click the Prediction Profiler red triangle and select **Simulator**.
3. Set the simulation parameters for Reaction Temperature to **Random > Normal Weighted** with Mean = 540 and SD = 1.
4. Set the simulation parameters for Reaction Time to **Random > Normal Weighted** with Mean = 0.115 and SD = 0.03.
5. Set the **N Runs** value to 15,000.

Figure 8.15 Initial Simulator Settings

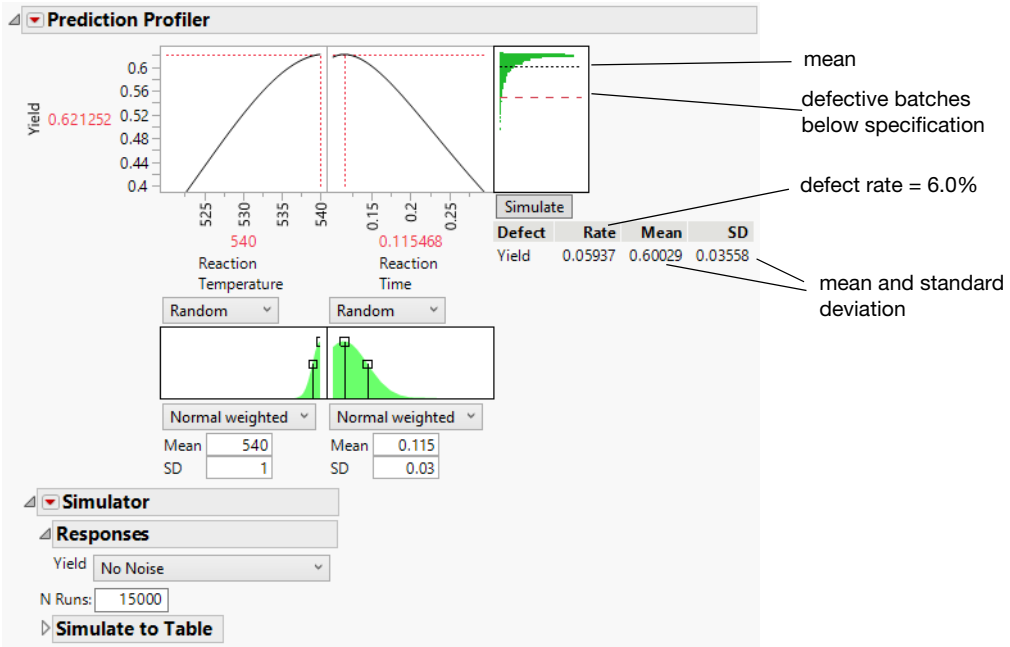


Yield has a lower spec limit of 0.55, set as a column property, and shows in Figure 8.15 as a dashed red line.

6. Click the **Simulate** button.

Note: Your numbers might differ from those shown in Figure 8.16 due to the random draws in the simulation.

Figure 8.16 Simulation Results



The predicted Yield is 0.62 for Reaction Temperature of 540 and Reaction Time of 0.115. The simulation estimates an average defect rate of about 6% with a standard deviation of 0.03 given the assumed variation in the temperature and time. The defect rate of 6.0%, indicates that about 6.0% of batches would be out of specification.

Use the simulator to explore other settings of Reaction Temperature and Reaction Time that might maintain a high Yield but with a lower defect rate. Before changing settings, save these factor settings that give us the maximum Yield for later use.

7. Click the Prediction Profiler red triangle and select **Factor Settings > Remember Settings**.
8. Type "Max Yield" and click **OK**.

The settings are appended to the report window.

Figure 8.17 Remembered Settings for Maximum Yield

Remembered Settings

	Setting	Reaction Temperature	Reaction Time	Yield
○	Max Yield	540	0.115468	0.6212525

9. Set the Mean value for Reaction Temperature to 535.

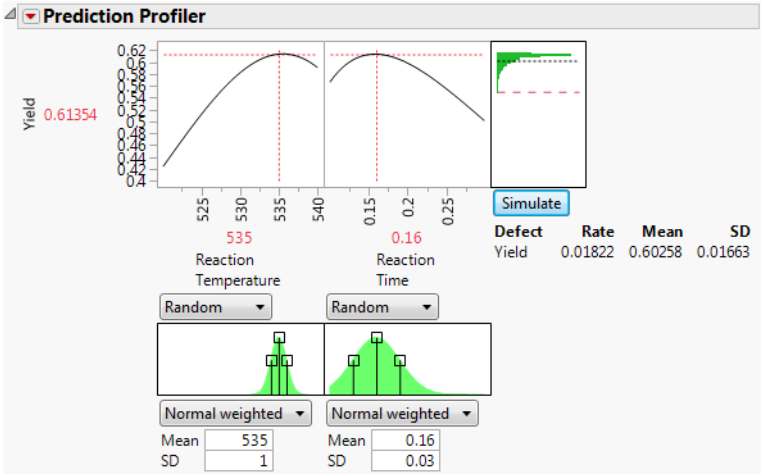
   Using the dashed red line in the Reaction Time plot, you can explore and determine the approximate value that maximized Yield. This value should be around 0.16.

10. Set the Mean value of Reaction Time to 0.16.

11. Click **Simulate**.

Note: Your numbers might differ from those shown in Figure 8.18 due to the random draws in the simulation.

Figure 8.18 Defect Rate for Temperature of 535



By making slight changes to the input factor settings, the defect rate decreases to about 1.8% with a decrease of less than 0.01 in the predicted yield. The fixed (no variability) settings that maximize Yield are not the same settings that minimize the defect rate in the presence of factor variation.

Simulation Experiment

You can run a simulation experiment to find the settings of Reaction Temperature and Reaction Time that minimize the defect rate. To do this, simulate the defect rate at each point in an experimental design for Reaction Temperature and Reaction Time. Then fit a predictive model for the defect rate and find factor settings to minimize the defect rate.

1. Click the Simulator red triangle and select **Simulation Experiment**.
2. Set the number of experimental runs to 80 and the portion of the design space to 1 to use the whole factor space in the experiment.
3. Click **OK**.

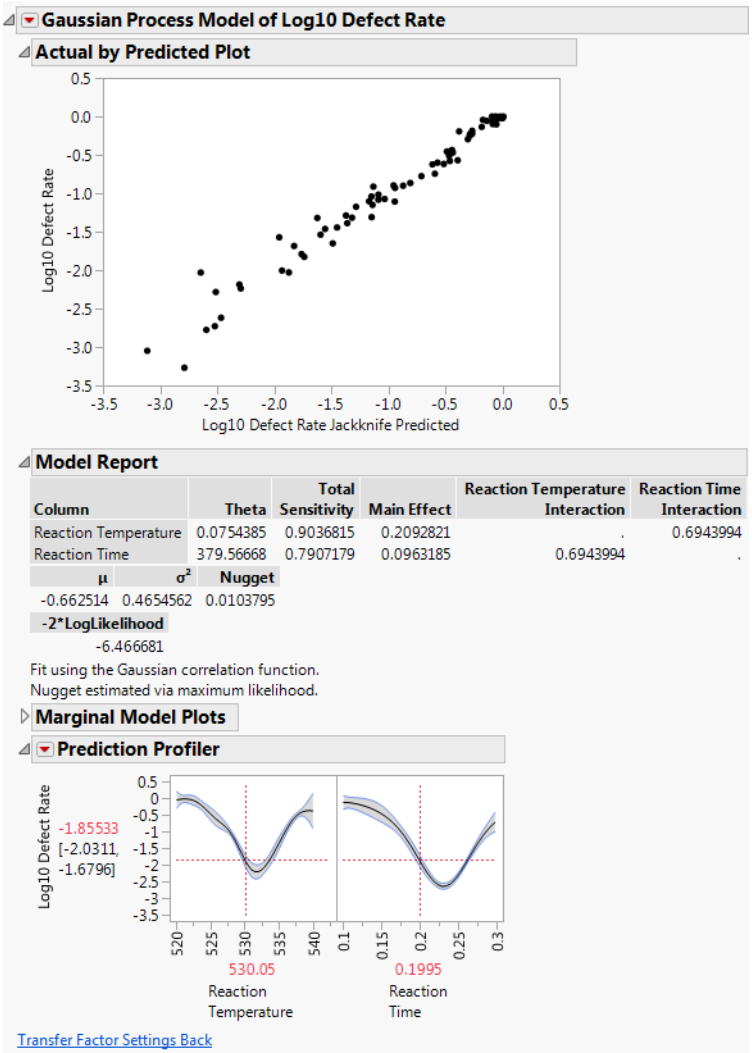
   A Latin Hypercube design with 80 design points is generated within the specified factor space, and N Runs random draws are taken at each of the design points. The design points are the center of the random draws, and the shape and variance of the random draws come from the factor distribution settings.

   A data table is created with the results of the experiment. The Overall Defect Rate is given at each design point. You can now fit a model that predicts the defect rate as a function of Reaction Temperature and Reaction Time.

   Note: Do not close the Stochastic Optimization Profiler window. You come back to it later.

4. From the new data table, click the green triangle next to the Gaussian Process script.

Figure 8.19 Results of Gaussian Process Model Fit

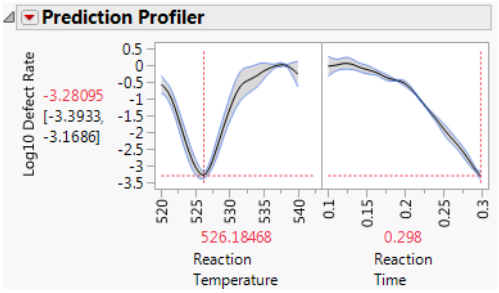


Note: Your results might be slightly different due to the random draws in the simulation.

5. To find the settings of Reaction Temperature and Reaction Time that minimizes the defect rate, click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

   The desirability function is already set up to minimize the defect rate.

Figure 8.20 Settings for Minimum Defect Rate



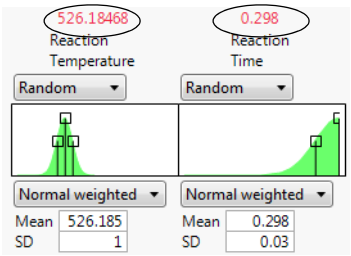
The settings that minimize the defect rate are approximately Reaction Temperature = 526 and Reaction Time = 0.3.

6. Click the **Transfer Factor Settings Back** button.

   This updates the original Profiler report to use the setting for Reaction Temperature and Reaction Time that minimize the defect rate.

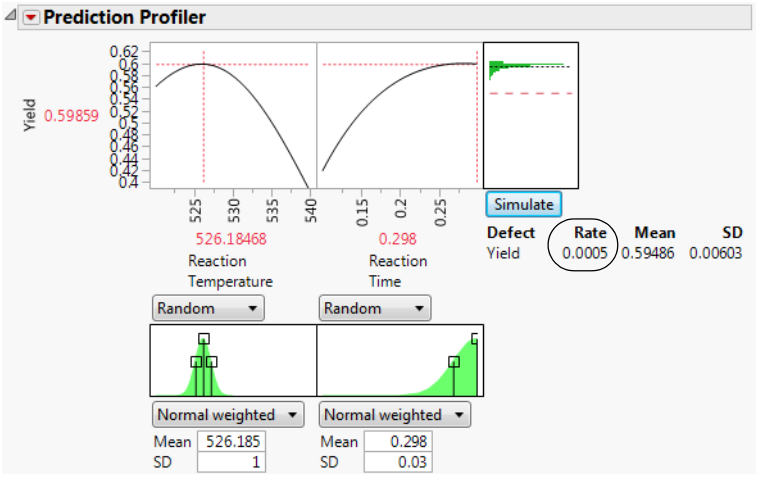
7. Return to the original Profiler report window.
8. Click the Prediction Profiler red triangle and select **Factor Settings > Remember Settings**.
9. Type "Min Defect" and click **OK**.

Figure 8.21 Minimum Defect Settings



10. With the new settings in place, click the **Simulate** button to estimate the defect rate at the new settings.

Figure 8.22 Lower Defect Rate

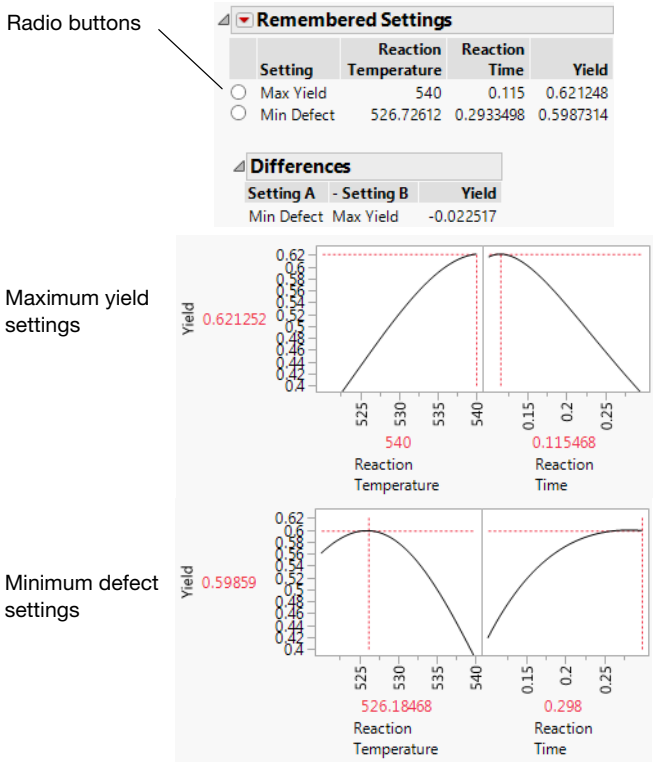


At the new settings the defect rate is 0.05%. This is much lower than the 6.0% for the settings that maximize Yield. That is a reduction of about 120x. Recall that the average Yield from the first settings is 0.62, and the new average is 0.59. The decrease in average Yield of 0.03 is the trade off for lowering the defect rate by 120x.

Because you saved the settings using Remember Settings, you can easily compare the old and new settings. The Differences report summarizes the difference.

11. Click the **Remembered Settings** radio buttons to view the profiler for each setting.

Figure 8.23 Settings Comparison



The chemist now knows what settings to use for a quality process. If the factors have no variation, the settings for maximum Yield are hot and fast. But, if the process inputs have variation similar to what we have simulated, the settings for maximum Yield produce a high defect rate. Therefore, to minimize the defect rate in the presence of factor variation, the settings should be cool and slow.

Example of Simulating General Formulas

Though the profiler and simulator are designed to work from formulas stored from a model fit, they work for any formula that is stored in a column. A typical application of simulation is to exercise financial models under certain probability scenarios to obtain the distribution of the objectives. This can be done in JMP. The key is to store the formulas into columns, set up ranges, and then conduct the simulation.

1. In the JMP Home Window, select **File > New > Script**. This opens a new script window.
2. Copy and paste the following JSL script into the new script window.

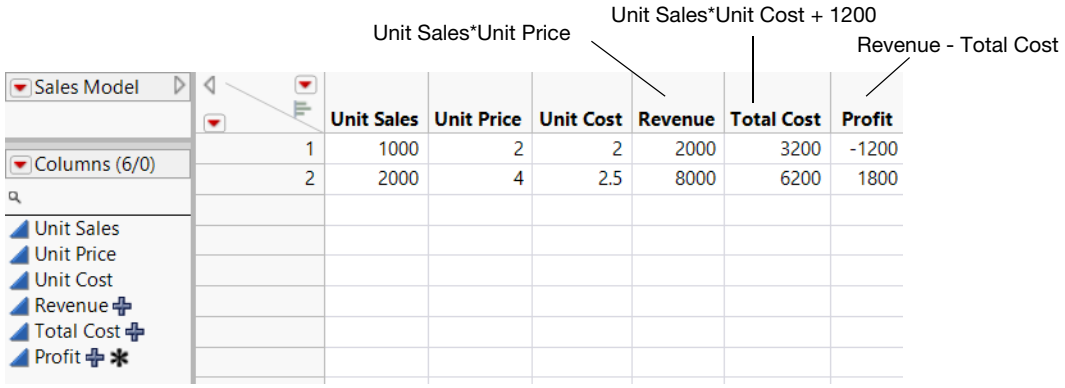
```
dt = New Table( "Sales Model" );
dt << New Column( "Unit Sales", Values( {1000, 2000} ) );
```

```
dt << New Column( "Unit Price", Values( {2, 4} ) );
dt << New Column( "Unit Cost", Values( {2, 2.5} ) );
dt << New Column( "Revenue",
   Formula( :Unit Sales * :Unit Price )
);
dt << New Column( "Total Cost",
   Formula( :Unit Sales * :Unit Cost + 1200 )
);
dt << New Column( "Profit",
   Formula( :Revenue - :Total Cost ),
   Set Property( "Spec Limits", {LSL( 0 )} )
);
Profiler(
   Y( :Revenue, :Total Cost, :Profit ),
   Objective Formula( Profit )
);
```

3. Click the Run Script icon to run the script. Alternatively, you can select Ctrl-R.

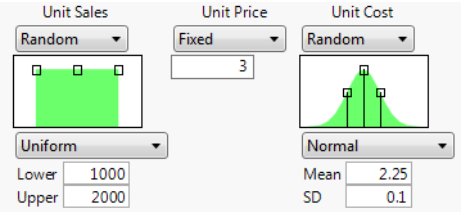
   The script creates the data table in Figure 8.24 with some initial scaling data and stores formulas into the output variables. It also launches the Prediction Profiler.

Figure 8.24 Data Table Created from Script



4. Click the Prediction Profiler red triangle and select the **Simulator**.
5. In the menus beneath Unit Sales and Unit Cost, select **Random**.
6. Fill in the factor parameters:
   - Unit Sales is **Uniform** with Lower limit 1000 and Upper limit 2000.
   - Unit Price is **Fixed** at 3.
   - Unit Cost is **Normal** with mean of 2.25 and standard deviation of 0.1.

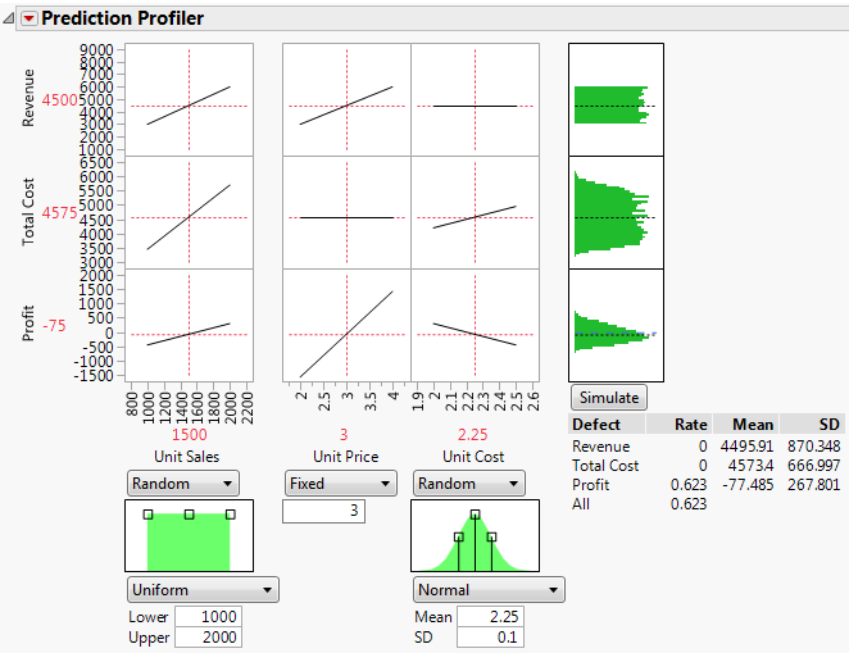
Figure 8.25 Specifications for Prediction Profiler



7. Click the **Simulate** button.

> **Note:** Your numbers might differ from those shown in Figure 8.26 due to the random draws in the simulation.

Figure 8.26 Simulator



It appears that the models are unlikely to be profitable. By putting a lower specification limit of zero on Profit, the defect report tells you that the probability of being unprofitable is 62%.

8. Change the fixed value of Unit Price to 3.25.
9. Click the **Simulate** button.

Figure 8.27 Results

Defect	Rate	Mean	SD
Revenue	0	4870.57	942.877
Total Cost	0	4573.4	666.997
Profit	0.2162	297.174	330.16
All	0.2162		

Now the probability of being unprofitable is down to about 21%.

If unit price cannot be raised anymore, you should now investigate lowering your cost or increasing sales, if you want to further decrease the probability of being unprofitable.

Statistical Details for the Simulator

Normal Weighted Distribution

JMP uses the *multivariate radial strata* method for each factor that uses the Normal Weighted distribution. This seems to work better than a number of importance sampling methods and is accurate at estimating in the extreme tails.

First, define the number of strata. The strata are a net of hyperspheres that are centered around 0. For d random factors, the strata are defined by their radial intervals. See Table 8.1.

Table 8.1 Strata Intervals

Strata Number	Inside Distance	Outside Distance
0	0	$\sqrt{d}$
1	$\sqrt{d}$	$\sqrt{d+\sqrt{2d}}$
2	$\sqrt{d+\sqrt{2d}}$	$\sqrt{d+2\sqrt{2d}}$
i	$\sqrt{d+(i-1)\sqrt{2d}}$	$\sqrt{d+i\sqrt{2d}}$
$N_{\text{Strata}}-1$	$\sqrt{d+\{(N_{\text{Strata}}-1)-1\}\sqrt{2d}}$	$\sqrt{d+(N_{\text{Strata}}-1)\sqrt{2d}}$

The default number of strata is 16. To change the number of strata, a hidden command N Strata is available if you hold the Shift key down while clicking on the red triangle next to Simulator. Increase the sample size as needed to maintain an even number of strata.

For each simulation run, the following is done:

1. Select a strata as mod($i - 1$, N_{Strata}) for run i.
2. Determine a random n-dimensional direction by scaling multivariate Normal (0,1) deviates to unit norm.
3. Determine a random distance using a chi-square quantile appropriate for the strata of a random uniform argument.
4. Scale the variates so that the norm is the random distance.
5. Scale and re-center the variates individually to be as specified for each factor.

The resulting factor distributions are multivariate normal with the appropriate means and standard deviations when estimated with the right weights. Note that you cannot use the Distribution standard deviation with weights, because it does not estimate the desired value. However, multiplying the weight by a large value, like 10^{12}, and using that as a Freq value results in the correct standard deviation.

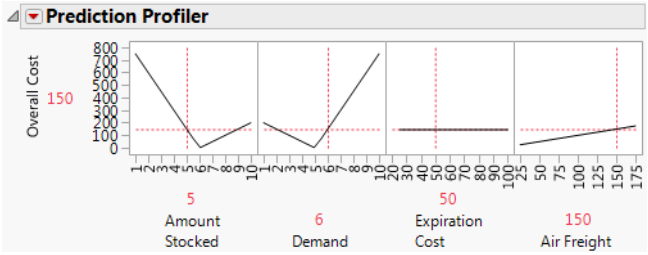
Chapter 9

Excel Profiler

Visualize Models Saved in Microsoft Excel

The JMP Add-In for Excel uses the JMP Profiler to visualize models (or formulas) stored in Excel worksheets. You can install the Excel add-in when you install JMP.

Figure 9.1 Example of a Prediction Profiler Using an Excel Model



Contents

Overview of the Excel Profiler 183
Example of an Excel Model 183
Run the JMP Profiler 185
Use Linear Constraints 186
Resolution of Profile Lines 186
Use the Excel Profiler from JMP 187

Overview of the Excel Profiler

The JMP Add-In for Excel uses the JMP Profiler to visualize models (or formulas) stored in Excel worksheets. You can choose to install the Excel Add-In when you install JMP. Profiling in the Excel Add-In is a two-step process:

1. Click the **Create/Edit Model** button (Excel 2010 through 2016) to enter information about the model that JMP needs. This needs to be done only once per model. For more information, click **Help** in the Create/Edit Model window.
2. Click the **Run Model** button (Excel 2010 through 2016) to launch the JMP Profiler and run the Excel model. See "Run the JMP Profiler".

Notes:

- The Preferences, Data Table, Graph Builder, Distribution, Fit Y by X, Fit Model, Time Series, and Control Chart buttons are not needed to profile an Excel model. For more information about these features, see *Using JMP*.
- A JMP ribbon is added to Microsoft Excel when the add-in is installed. If there is no JMP ribbon, install the add-in by double-clicking the JMP setup.exe file, selecting **Modify**, selecting **Excel Add-In**, and clicking **Next**.

Example of an Excel Model

An Excel model consists of one or more Excel formulas. Each formula must be a function of one or more other cells. This example uses the Demand.xls sample import data located in the Samples/Import Data folder.

Figure 9.2 Demand Model in Excel

Clipboard | Font | Alignment

JMP_Output_5cfcff4... | =IF(B3<=B4,(B4-B3) *B5,(B3-B4) *B6)

	A	B	C	D	E	F	G
1	JMP Profiler						
2	Label	Inputs	Min	Max	Mean		
3	Amount Stocked	5	1	10	5		
4	Demand	6	1	10	1		
5	Air Freight	150	25	175	150		
6	Expiration Cost	50	25	100	50		
7	Label	Outputs	Min	Max	Mean		
8	Overall Cost	150	0	1000	100		
9							
10							

About the Demand.xls Sample Import Data

The formula in cell B8 is a calculation of the Overall Cost associated with having different amounts of product in stock. The formula, which is shown in the Formula Bar, is a function of four cells:

- Amount Stocked is the amount of product in stock.
- Demand is the customer demand for the product.
- Air Freight is the cost per unit to ship additional product by air when the demand exceeds the amount in stock.
- Expiration Cost is the cost per unit of disposing of unused product when the demand is less than the amount in stock.

The calculations of the formula depend on the relationship between Amount Stocked and Demand.

- If Amount Stocked is less than Demand, then the company has to ship additional units, at a cost of (Demand-Amount Stocked) × Air Freight. For example, if the demand is 8, but the company has only 6 in stock, then it has to ship 8 - 6 = 2 units at a cost of 2 × 150 = 300.
- If Amount Stocked is greater than Demand, then the company has to dispose of unused product, at a cost of (Amount Stocked - Demand) × Expiration Cost. For example, if the demand is 5, but the company has 8 in stock, then it has to dispose of 8 - 5 = 3 units at a cost of 3 × 50 = 150.
- If Amount Stocked is equal to Demand, then there is no shipping cost or disposal cost.
- There is never both a shipping cost and a disposal cost at the same time.

Create the Model in Excel

1. Select **Help > Sample Data Library** and navigate up one level to the Samples/Import Data folder.
2. Double-click Demand.xls to open the file in Microsoft Excel.
3. In Microsoft Excel, click the JMP ribbon.
4. Click the **Create/Edit Model** button .

   The name of the workbook is displayed in the Model and Model Name fields.

   The Inputs and Outputs fields are populated with data from the worksheet.
5. Enter Customer Demand in the Model Name field and click **Apply**.

   The Model field is updated.
6. Select **Air Freight** in the Inputs box and then click the down arrow button .

   Air Freight is moved to the bottom of the list because you want it displayed last in the Profiler.

7. Click **OK**.

   The Excel model is saved to the worksheet.

Notes:

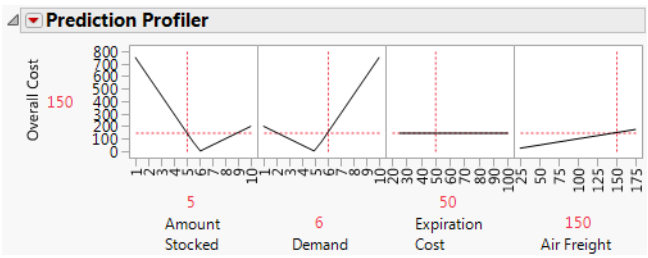
- If the fields in the Create/Edit Model window are not populated when you set up your model, click **Choose** and select the cell that contains the input or output name.
  - For inputs, specify the values and click **Apply**. Inputs must be values, not formulas.
  - For outputs, the specified cell must be a formula containing only information from the Input cells.
- You can create more than one model for a worksheet. In the Create/Edit Model window, click the plus button + next to the Model name. In the Model Name field, change the name if necessary and click the **Apply** button. You can then change the inputs and outputs as necessary and click **OK**.
- You must define the entire model on one worksheet. A model cannot reference cells on another worksheet.

Run the JMP Profiler

Once you create the model using the Excel Add-In, you can run it in the JMP Profiler. From the Excel Add-In, perform the following actions:

1. In Microsoft Excel, click the JMP ribbon.
2. Click the **Run Model** button.
3. Select the model that you want to run.
4. Click **Profile in JMP**.
5. Use the JMP Profiler to simultaneously see the effect of all inputs on the output. You can also simulate a range of input combinations to see the resulting range of output values.

Figure 9.3 Example of the Prediction Profiler Using Excel Models



Note: To ensure that your original Excel worksheet is not altered, JMP runs a hidden copy of Excel in the background that controls all of the Profiler calculations. Since the Excel worksheet and the Profiler are not dynamically linked, after you update a formula in the worksheet, you must relaunch the JMP Profiler to reflect the changes.

Use Linear Constraints

Within the JMP Profiler, you can alter the linear constraints in order to restrict the model input values. You are prompted to save the constraints to the Excel workbook. After constraints are saved to the Excel workbook, whenever the model is profiled from the Excel Add-In, the constraints are incorporated.

1. Click the Prediction Profiler red triangle and select **Alter Linear Constraints**.
2. Click **Add Constraint**.
3. Type in the constraining values.
4. Click **OK**.
5. Click the Prediction Profiler red triangle and select **Save Linear Constraints**.

   You are prompted to save the constraints to the Excel workbook.
6. Click **Yes**.

Note: When you save the .xls file, you might see a compatibility error. If so, click **Continue** to save the file.

The workbook opens in Excel. When you run the model, the constraints are reflected in the JMP Profiler. For more information about linear constraints, see "Linear Constraints".

Tip: To delete a linear constraint, set all constraint values to zero.

Resolution of Profile Lines

The Default N Levels option on the Prediction Profiler red triangle menu affects the resolution of the profile lines. Note the following information:

- This option defaults to 17 when the Profiler runs a model stored in Excel.
- This option defaults to 41 when the model is stored directly in JMP.

If the same model is stored in both Excel and JMP, then the profile lines can appear differently when the models are profiled. Increasing this value causes the Excel Profiler to run slower.

Use the Excel Profiler from JMP

After you have defined model input and outputs in an Excel file, you can profile the model from within JMP.

1. Select **Graph** > **Excel Profiler**.
2. Locate the Excel file containing the model and then click **Open**.
3. If the Excel file contains multiple models, you are prompted to select the model that you want to profile.

Use the following code to script the Excel Profiler:

```
Excel Profiler( "path to workbook", <"model name"> ) ;
```

If more than one model exists, and no model is specified, a window with the list of available models appears. For more information about scripting the Excel Profiler, see the *Scripting Guide*.

Appendix A

References

Box, G. E. P., and Draper, N. R. (1987). *Empirical Model–Building and Response Surfaces*. New York: John Wiley & Sons.

Box, G. E. P., and Draper, N. R. (2007). *Response Surfaces, Mixtures, and Ridge Analyses*. 2nd ed. New York: John Wiley & Sons.

Cornell, J. A. (1990). *Experiments with Mixtures*. 2nd ed. New York: John Wiley & Sons.

Derringer, G., and Suich, R. (1980). "Simultaneous Optimization of Several Response Variables." *Journal of Quality Technology* 12.4:214–219.

Hastie, T., Tibshirani, R., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. New York: Springer-Verlag.

Saltelli, A. (2002). "Making best use of model evaluations to compute sensitivity indices." *Computer Physics Communications* 145:280–297.

Schafer, J., and Strimmer, K. (2005). "A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics." *Statistical Applications in Genetics and Molecular Biology* 4:1–30.

Sobol, I. M. (1993). "Sensitivity Estimates for Nonlinear Mathematical Models." *MMCE* 1.4:407–414.

Appendix B

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

  - You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - You must cause any modified files to carry prominent notices stating that You changed the files; and
  - You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
  - You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS”, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Design of Experiments Guide

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.2

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Design of Experiments Guide*. Cary, NC: SAS Institute Inc.

JMP® 16 Design of Experiments Guide

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

December 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Design of Experiments Guide

1 Learn about JMP 21
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 23
JMP Help 24
JMP Documentation Library 24
Additional Resources for Learning JMP 30
JMP Tutorials 30
Sample Data Tables 30
Learn about Statistical and JSL Terms 31
Learn JMP Tips and Tricks 31
JMP Tooltips 31
JMP User Community 32
Free Online Statistical Thinking Course 32
JMP New User Welcome Kit 32
Statistics Knowledge Portal 32
JMP Training 32
JMP Books by Users 33
The JMP Starter Window 33
JMP Technical Support 33

2 Introduction to DOE 35
Overview of Design of Experiment Platforms

3 Starting Out with DOE 41
Example and Key Concepts
Overview of Experimental Design and the DOE Workflow 43
The Coffee Strength Experiment 44
Define the Study and Goals 45
Create the Design 47
Run the Experiment 52
Analyze the Data 53
The DOE Workflow: Describe, Specify, Design 59
Define Responses and Factors 60

Specify the Model ... 61
Generate the Design ... 62
Evaluate the Design ... 63
Make the Table ... 63
Principles and Guidelines for Experimental Design ... 64
Effect Hierarchy ... 64
Effect Heredity ... 64
Effect Sparsity ... 65
Center Points, Replicate Runs, and Testing ... 65

4 Custom Designs ... 67
Construct Designs That Meet Your Needs

Overview of Custom Design ... 69
Example of a Custom Design ... 71
Build a Custom Design ... 88
Responses ... 88
Factors ... 90
Define Factor Constraints ... 97
Alias Terms ... 102
Design Generation ... 103
Design ... 106
Design Evaluation ... 106
Output Options ... 107
Custom Design Options ... 109
Simulate Responses ... 113
Save X Matrix ... 115
Number of Starts ... 117
Design Search Time ... 118
Technical Details for Custom Designs ... 119
Designs with Randomization Restrictions ... 119
Covariates with Hard-to-Change Levels ... 128
Numbers of Whole Plots and Subplots ... 128
Optimality Criteria ... 129
D-Efficiency ... 134
Coordinate-Exchange Algorithm ... 134

5 Examples of Custom Designs ... 135
Perform Experiments That Meet Your Needs

Screening Experiments ... 137
Design That Estimates Main Effects Only ... 137
Design That Estimates All Two-Factor Interactions ... 140

Design That Avoids Aliasing of Main Effects and Two-Factor Interactions 143
Supersaturated Screening Designs 146
Design for Fixed Blocks 154
Response Surface Experiments 159
Response Surface Design 159
Response Surface Design with Flexible Blocking 165
Comparison of a D-Optimal and an I-Optimal Response Surface Design 168
Response Surface Design With Constraints and Categorical Factor 172
Mixture Experiments 178
Mixture Design with Nonmixture Factors 178
Mixture of Mixtures Design 182
Experiments with Covariates 187
Design with Fixed Covariates 187
Design with Hard-to-Change Covariates 190
Design with a Linear Time Trend 194
Experiments with Randomization Restrictions 197
Split-Plot Experiment 197
Two-Way Split-Plot Experiment 202
Experiments for Robust Process and Product Design 210

6 Augment Designs 213
Add Additional Runs to an Existing Design

Example Using the Augment Design Platform 215
Launch the Augment Design Platform 221
Augment Design Window 222
Factors 222
Define Factor Constraints 223
Augmentation Choices 226
Augment Design Platform Options 227
Additional Examples of Augmentation Choices 231
Replicate a Design 231
Add Center Points to a Design 233
Create a Foldover Design 234
Add Axial Points to a Design 235
Space Filling Augmentation 236
Statistical Details 237

7 Definitive Screening Designs 239

Overview of Definitive Screening Design 241
Examples of Definitive Screening Designs 243

Definitive Screening Design ... 244
Comparison with a Fractional Factorial Design ... 246
Definitive Screening Design with Blocking ... 248
Comparison of a Definitive Screening Design with a Plackett-Burman Design ... 252
Definitive Screening Design Window ... 255
Responses ... 256
Factors ... 258
Design Options ... 260
Design ... 262
Design Evaluation ... 262
Output Options ... 263
Definitive Screening Design Options ... 265
Simulate Responses ... 267
Technical Details ... 270
Structure of Definitive Screening Designs ... 270
Analysis of Experimental Data ... 272

8 The Fit Definitive Screening Platform ... 275
Analyze Data from Definitive Screening Experiments

Overview of the Fit Definitive Screening Platform ... 277
Identification of Active Effects in DSDs ... 277
Effective Model Selection for DSDs ... 277
Example of the Fit Definitive Screening Platform ... 278
Fit the Model ... 278
Examine Results ... 279
Reduce the Model ... 280
Launch the Fit Definitive Screening Platform ... 282
Fit Definitive Screening Report ... 282
Stage 1 - Main Effect Estimates ... 283
Stage 2 - Even Order Effect Estimates ... 284
Combined Model Parameter Estimates ... 285
Main Effects Residual Plots ... 285
Prediction Profiler ... 286
Fit Definitive Screening Platform Options ... 286
Technical Details for the Fit Definitive Screening Platform ... 286
The Effective Model Selection for DSDs Algorithm ... 286

9 Screening Designs ... 291

Overview of Screening Designs ... 293
Underlying Principles ... 293

Analysis of Screening Design Results ... 294
Examples of Screening Designs ... 294
Compare a Fractional Factorial Design and a Main Effects Screening Design ... 294
Main Effects Screening Design where No Standard Design Exists ... 302
Screening Design Window ... 305
Responses ... 306
Factors ... 308
Choose Screening Type ... 310
Choose from a List of Fractional Factorial Designs ... 311
Design Type ... 311
Resolution as a Measure of Confounding ... 314
Display and Modify Design ... 315
Main Effects Screening Designs ... 318
Design Generation ... 319
Design ... 320
Design Evaluation ... 320
Output Options ... 321
Make Table ... 322
Screening Design Options ... 323
Additional Examples of Screening Designs ... 325
Modify Generating Rules in a Fractional Factorial Design ... 326
Plackett-Burman Design ... 331

10 The Fit Two Level Screening Platform ... 337
Analyze Data from Screening Experiments

Overview of the Fit Two Level Screening Platform ... 339
Example of Fit Two Level Screening Platform ... 339
Launch the Fit Two Level Screening Platform ... 342
The Screening Report ... 343
Contrasts ... 344
Half Normal Plot ... 344
Make or Run Model ... 345
Additional Fit Two Level Screening Analysis Examples ... 345
Analyze a Plackett-Burman Design ... 345
Effect Heredity Example ... 346
Analyze a Supersaturated Design ... 348
Technical Details for the Fit Two Level Screening Platform ... 350
Order of Effect Entry ... 350
Fit Two Level Screening as an Orthogonal Rotation ... 350
Lenth's Pseudo-Standard Error ... 351

Lenth t-Ratios 351

11 Response Surface Designs 355

Overview of Response Surface Designs 357
Example of a Response Surface Design 359
Construct a Box-Behnken Design 360
Analyze the Experimental Data 361
Explore Optimal Settings 363
Response Surface Design Window 366
Responses 367
Factors 369
Choose a Design 371
Specify Output Options 372
Make Table 373
Response Surface Design Options 374

12 Full Factorial Designs 377

Overview of Full Factorial Design 379
Example of a Full Factorial Design 379
Construct the Design 379
Analyze the Experimental Data 381
Build a Full Factorial Design 385
Responses 385
Factors 388
Select Output Options 389
Make Table 391
Full Factorial Design Options 392

13 Mixture Designs 395

Designs for Formulations

Overview of Mixture Designs 397
Mixture Design Window 397
Responses 398
Factors 400
Ternary Plot Overview 401
Examples of Mixture Design Types 402
Optimal Mixture Design 402
Simplex Centroid Design 404
Simplex Lattice Design 406

Extreme Vertices Design . . . 407
ABCD Design . . . 411
Space Filling Design . . . 412
Fitting Mixture Designs . . . 414
Whole Model Tests and Analysis of Variance Reports . . . 415
Understanding Response Surface Reports . . . 415
A Chemical Mixture Example . . . 416
Create the Design . . . 416
Analyze the Mixture Model . . . 418
The Prediction Profiler . . . 419
The Mixture Profiler . . . 420

14 Taguchi Designs . . . 423
Designs with Noise Factors

Overview of Taguchi Designs . . . 425
Example of a Taguchi Design . . . 425
Build a Taguchi Design . . . 429
Responses . . . 430
Factors . . . 430
Choose Taguchi Inner and Outer Array Designs . . . 431
Display Coded Design . . . 431
Make the Design Table . . . 432
Taguchi Design Options . . . 432

15 Evaluate Designs . . . 435
Explore Properties of Your Design

Overview of the Evaluate Design Platform . . . 437
Example of Evaluate Design . . . 438
Assessing the Impact of Lost Runs . . . 438
Evaluating Power Relative to a Specified Model . . . 445
Evaluate Design Launch Window . . . 447
Evaluate Design Window . . . 447
Factors . . . 449
Model . . . 449
Alias Terms . . . 450
Design . . . 451
Design Evaluation . . . 451
Power Analysis . . . 451
Prediction Variance Profile . . . 458
Fraction of Design Space Plot . . . 459
Prediction Variance Surface . . . 460

Estimation Efficiency ... 462
Alias Matrix ... 464
Color Map on Correlations ... 465
Design Diagnostics ... 467
Evaluate Design Options ... 469

16 Compare Designs ... 471
Compare and Evaluate Designs Simultaneously

Overview of the Compare Designs Platform ... 473
Examples of Comparing Designs ... 473
Designs of Same Run Size ... 473
Designs of Different Run Sizes ... 478
Split Plot Designs with Different Numbers of Whole Plots ... 482
Launch the Compare Designs Platform ... 486
Compare Designs Window: Specify Model and Alias Terms ... 488
Reference Design ... 488
Factors ... 488
Model ... 488
Alias Terms ... 489
Compare Designs Window: Design Evaluation ... 489
Power Analysis ... 490
Prediction Variance Profile ... 493
Fraction of Design Space Plot ... 495
Relative Estimation Efficiency ... 496
Alias Matrix Summary ... 498
Absolute Correlations ... 500
Design Diagnostics ... 502
Compare Designs Options ... 504

17 Prospective Sample Size and Power ... 505

Launch the Sample Size and Power Platform ... 508
One Sample Mean Calculator ... 509
Examples of the One Sample Mean Calculator ... 510
One Sample Mean Calculator Fields ... 514
Statistical Details for the One Sample Mean Calculator ... 515
Two Sample Means Calculator ... 516
Example of the Two Sample Means Calculator ... 516
Two Sample Means Calculator Fields ... 517
Statistical Details for the Two Sample Means Calculator ... 518
k Sample Means Calculator ... 519

Example of the k Sample Means Calculator 519
k Sample Means Calculator Fields 521
Statistical Details for the k Sample Means Calculator 522
One Sample Standard Deviation Calculator 523
Example of the One Sample Standard Deviation Calculator 523
One Sample Standard Deviation Calculator Fields 524
Statistical Details for the One Sample Standard Deviation Calculator 525
One Sample Proportion Calculator 526
Example of the One Sample Proportion Calculator 527
One Sample Proportion Calculator Fields 528
Statistical Details for the One Proportion Calculator 529
Two Sample Proportions Calculator 530
Example of the Two Sample Proportions Calculator 530
Two Sample Proportions Calculator Fields 531
Statistical Details for the Two Proportions Calculator 532
Counts per Unit Calculator 533
Example of the Counts per Unit Calculator 533
Counts per Unit Calculator Fields 534
Statistical Details for the Counts per Unit Calculator 535
Sigma Quality Level Calculator 536
Example of the Sigma Quality Level Calculator 536
Example Computing the Number of Defects 536
Sigma Quality Level Calculator Fields 537
Statistical Details for the Sigma Quality Calculator 537
Reliability Test Plan Calculator 538
Example of the Reliability Test Plan Calculator 538
Reliability Test Plan Calculator Fields 540
Statistical Details for the Reliability Test Plan Calculator 541
Reliability Demonstration Calculator 542
Example of the Reliability Demonstration Calculator 542
Reliability Demonstration Calculator Fields 544
Statistical Details for the Reliability Demonstration Calculator 545

18 Choice Designs 547
Create a Design for Selecting Preferred Product Profiles

Overview of Choice Designs 549
Choice Design Terminology 549
Bayesian D-Optimality 550
Example of a Choice Design 550
Example of a Choice Design with Analysis 553

Create a Choice Design for a Pilot Study ... 553
Analyze the Pilot Study Data ... 555
Design the Final Choice Experiment Using Prior Information ... 556
Run the Design and Analyze the Results ... 558
Choice Design Window ... 561
Attributes ... 561
Model ... 562
Design Generation ... 563
Design ... 564
Make Table ... 566
Choice Design Options ... 566
Technical Details ... 567
Bayesian D-Optimality and Design Construction ... 567
Utility-Neutral and Local D-Optimal Designs ... 568

19 MaxDiff Design ... 569
Create a Design for Selecting Best and Worst Items

Overview of the MaxDiff Design Platform ... 571
Example of a MaxDiff Design ... 571
MaxDiff Design Launch Window ... 575
MaxDiff Window ... 575
Design Options Outline ... 575
Design Outline ... 576
Pairwise Incidence Matrix ... 576
Make Table ... 576
MaxDiff Options ... 576

20 Covering Arrays ... 579
Detecting Component Interaction Failures

Overview of Covering Arrays ... 581
Example of a Covering Array with No Factor Level Restrictions ... 583
Create the Design ... 583
Analyze the Experimental Data ... 585
Example of a Covering Array with Factor Level Restrictions ... 587
Create the Design ... 588
Analyze the Experimental Data ... 592
Covering Array Window ... 593
Factors ... 594
Restrict Factor Level Combinations ... 595
Design ... 597
Metrics ... 600

Output Options 600
The Covering Array Data Table 601
Covering Array Options 602
Technical Details for Covering Arrays 603
Algorithm for Optimize 604
Formulas for Metrics 604

21 Space-Filling Designs 607

Overview of Space-Filling Designs 609
Build a Space Filling Design 610
Responses 610
Factors 612
Define Factor Constraints 614
Space Filling Design Methods 618
Design 618
Design Diagnostics 618
Design Table 619
Space Filling Design Options 619
Sphere-Packing Designs 621
Creating a Sphere-Packing Design 621
Visualizing the Sphere-Packing Design 623
Latin Hypercube Designs 624
Creating a Latin Hypercube Design 624
Visualizing the Latin Hypercube Design 626
Uniform Designs 628
Comparing Sphere-Packing, Latin Hypercube, and Uniform Methods 630
Minimum Potential Designs 632
Maximum Entropy Designs 634
Gaussian Process IMSE Optimal Designs 636
Fast Flexible Filling Designs 637
FFF Optimality Criterion 637
Set Average Cluster Size for FFF Designs 639
Constraints 639
Creating and Viewing a Constrained Fast Flexible Filling Design 640
Creating a Space-Filling Design for a Map Shape 641
Example of a Sphere-Packing Design 646
Create the Sphere-Packing Design for the Borehole Data 646
Guidelines for the Analysis of Deterministic Data 647

22 Accelerated Life Test Designs 651
Designing Experiments for Accelerated Life Tests
Overview of Accelerated Life Test Designs 653
Example of an Accelerated Life Test Design 653
Example of Augmenting an Accelerated Life Test Design 660
Build an Accelerated Life Test Design 665
Specify the Design Structure 666
Specify Acceleration Factors 666
Specify Design Details 667
Review and Update Specifications for ALT Plans 669
Create and Assess the Optimal 671
Create Design Tables 673
Accelerated Life Test Options 673
Statistical Details for the ALT Design Platform 674
Failure Distributions for ALT Designs 674
R Precision Factor (for a 95% CI) 676
Optimality Criterion for Accelerated Life Tests 677

23 Nonlinear Designs 679

Overview of Nonlinear Designs 681
Examples of Nonlinear Designs 682
Create a Nonlinear Design with No Prior Data 682
Augment a Design Using Prior Data 687
Create a Design for a Binomial Response 691
Nonlinear Design Launch Window 694
Nonlinear Design Window 695
Factors 695
Parameters 696
Design Generation 696
Design 696
Make Table or Augment Table 697
Nonlinear Design Options 697
Statistical Details for Nonlinear Designs 698
Nonlinear Models 698
Radial-Spherical Integration of the Optimality Criterion 699
Finding the Optimal Design 699

24 Balanced Incomplete Block Designs 701

Example of Balanced Incomplete Block Designs 703

Build a Balanced Incomplete Block Design ... 705
Specify Treatments, Blocks, and Make Design ... 705
Verify Design and Make Table ... 706
Balanced Incomplete Block Design Options ... 707
Statistical Details for Balanced Incomplete Block Designs ... 708

25 MSA Designs ... 709
Create a Design for a Measurement Systems Analysis Experiment
Overview of MSA Designs ... 711
Example of MSA Design ... 711
Build an MSA Design ... 714
Responses ... 714
Factors ... 717
MSA Design ... 718
MSA Design Diagnostics ... 718
MSA Design Table ... 719
MSA Design Options ... 719
Statistical Details for the MSA Design Platform ... 720

26 Group Orthogonal Supersaturated Designs ... 723
Create Group Orthogonal Supersaturated Designs
Overview of Group Orthogonal Supersaturated Designs ... 725
Example of Group Orthogonal Supersaturated Designs ... 725
Build a Group Orthogonal Supersaturated Design ... 727
Specify Factors, Structure, and Make Design ... 727
Verify Design and Make Table ... 728
Group Orthogonal Supersaturated Design Options ... 729
Statistical Details for Group Orthogonal Supersaturated Designs ... 730

27 The Fit Group Orthogonal Supersaturated Platform ... 731
Analyze Data from Group Orthogonal Supersaturated Designs
Overview of the Fit Group Orthogonal Supersaturated Platform ... 733
Example of Fit Group Orthogonal Supersaturated ... 733
Launch the Fit Group Orthogonal Supersaturated Platform ... 736
The Fit Group Orthogonal Supersaturated Report ... 737
Fit Group Orthogonal Supersaturated Platform Options ... 738

28 Sample Size Explorers ... 739
Sample Size and Power Calculators
Overview of the Sample Size Explorers Platform ... 741
Example of Sample Size Explorers ... 741
Launch Sample Size Explorers ... 742

Power Explorers for Hypothesis Tests ... 743
Power for One Sample Mean ... 743
Power for One Sample Proportion ... 746
Power for One Sample Variance ... 748
Power for One Sample Equivalence ... 750
Power for Two Independent Sample Means ... 752
Power for Two Independent Sample Proportions ... 756
Power for Two Independent Sample Variances ... 758
Power for Two Independent Sample Equivalence ... 761
Power for ANOVA ... 764
Confidence Interval Calculators ... 765
Margin of Error for One Sample Mean ... 766
Margin of Error for One Sample Proportion ... 769
Margin of Error for One Sample Variance ... 770
Margin of Error for Two Independent Sample Means ... 771
Margin of Error for Two Independent Sample Proportions ... 774
Margin of Error for Two Independent Sample Variances ... 776
Reliability Demonstration Calculators ... 778
Parametric Reliability Demonstration ... 778
Nonparametric Reliability Demonstration ... 780
Saved Settings in the Sample Size Explorers ... 781

A Column Properties ... 783
Understanding Column Properties Assigned by DOE

Adding and Viewing Column Properties ... 785
Response Limits ... 786
Response Limits Example ... 788
Editing Response Limits ... 791
Detection Limits ... 794
Functional Response ... 794
Design Role ... 794
Design Role Example ... 795
Coding ... 797
Low and High Values ... 797
Coding Column Property and Center Polynomials ... 798
Coding Example ... 798
Assigning Coding ... 802
Mixture ... 805
Mixture Example ... 807
Factor Changes ... 810

Factor Changes Example ... 811
Value Order ... 813
Value Order Example ... 815
Assigning Value Order ... 816
Value Labels ... 817
Value Labels Example ... 817
RunsPerBlock ... 819
RunsPerBlock Example ... 819
ConstraintState ... 820
ConstraintState Example ... 821

B Technical Details ... 823

The Alias Matrix ... 825
Designs with Hard or Very Hard Factor Changes ... 825
Designs with If Possible Effects ... 826
Power Calculations ... 826
Power for a Single Parameter ... 826
Power for a Categorical Effect ... 828
Relative Prediction Variance ... 829

C References ... 831

D Technology License Notices ... 837

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 23
JMP Help . . . 24
JMP Documentation Library . . . 24
Additional Resources for Learning JMP . . . 30
JMP Tutorials . . . 30
Sample Data Tables . . . 30
Learn about Statistical and JSL Terms . . . 31
Learn JMP Tips and Tricks . . . 31
JMP Tooltips . . . 31
JMP User Community . . . 32
Free Online Statistical Thinking Course . . . 32
JMP New User Welcome Kit . . . 32
Statistics Knowledge Portal . . . 32
JMP Training . . . 32
JMP Books by Users . . . 33
The JMP Starter Window . . . 33
JMP Technical Support . . . 33

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction to DOE

Overview of Design of Experiment Platforms

The JMP DOE platforms help you to design, evaluate, and analyze experiments. Most of the platforms focus on constructing designs. Other platforms support the design effort. This section provides a quick overview of each of the platforms found under the DOE menu.

Design Construction Platforms

Custom Design Constructs designs that fit a wide variety of settings. Custom designs tend to be more cost effective and flexible than approaches based exclusively on classical designs.

Custom designs accommodate various types of factors, constraints, and disallowed combinations. You can specify which effects are necessary to estimate and which are desirable to estimate, given the number of runs. You can specify a number of runs that matches the budget for your experimental situation. Custom designs also support hard-to-change and very-hard-to-change factors, allowing you to construct split-plot and related designs.

The Custom Design platform constructs many special design types:

- screening
- response surface
- mixture
- random block
- split-plot
- split-split-plot
- two-way split-plot

You can construct classical screening, response surface, and mixture designs using other platforms. However, the Custom Design platform gives you flexibility that is not available in the other platforms. Constructing designs for split-plot situations can be done only using the Custom Design platform.

Definitive Screening Design Constructs screening designs for continuous and two-level categorical factors. Definitive screening designs are useful if you suspect active interactions or curvature. Definitive screening designs enable you to identify the source of

strong nonlinear effects while avoiding complete confounding between any effects up through the second order.

Definitive screening designs are most appropriate for experimentation with four or more factors. Definitive screening designs support grouping runs into blocks. The number of blocks is user-specified.

Screening Design Constructs screening designs for continuous, discrete numeric, and categorical factors with an arbitrary number of levels. When standard designs exist, you have two options:

- Choose from a list of classical screening designs. These designs allow two-level continuous factors or two- or three-level categorical or discrete continuous factors.
- Generate a design that is orthogonal or nearly orthogonal for main effects. *Near-orthogonal designs* allow for categorical and discrete numeric factors with any number of levels, as well as two-level continuous factors. These designs focus on estimating main effects in the presence of negligible interactions.

For many screening situations, standard designs are not available. In these situations, you can construct near-orthogonal screening designs.

Response Surface Design Constructs designs that model a quadratic function of continuous factors. To fit the quadratic effects, response surface designs require three settings for each factor. JMP provides response surface designs for up to eight factors.

You can choose from a list of Central Composite or Box-Behnken designs. When appropriate, Central Composite designs that block orthogonally are included in the list. Various modifications to Central Composite designs are supported.

Full Factorial Design Constructs full factorial designs for any number of continuous or categorical factors, both with arbitrarily many levels. A full factorial design has a run at every combination of settings of the factors. Full factorial designs tend to be large. The number of runs equals the product of the numbers of factor levels.

Mixture Design Constructs designs that you use when factors are ingredients in a mixture. In a mixture experiment, a change in the proportion of one ingredient requires that one or more of the remaining ingredients change to maintain the sum. Choose from among several design types, including some classical mixture design approaches: optimal, simplex centroid, simplex lattice, extreme vertices, ABCD, and space filling. For optimal, extreme vertices, and space filling mixture designs, you can specify linear inequality constraints to limit the design space.

Taguchi Arrays Constructs designs that you use for signal-to-noise analysis. The designs are based on Taguchi's inner and outer array approach. Control factor settings constitute the

inner array and noise factor settings form the outer array. The mean and signal-to-noise ratio are the responses of interest.

An alternative to using a Taguchi array is to construct a custom design that includes control factors, noise factors, and control-by-noise interactions. Such designs, called *combined arrays*, are generally more cost-effective and informative than Taguchi arrays.

Choice Design Constructs designs that you can use to compare prospective products. The factors in a choice design are product attributes. The design arranges product profiles, which are combinations of various attributes, in pairs or in groups of three or four. The experiment consists of having respondents indicate which profile in a pair of profiles that they prefer. You can generate a choice design that reflects prior information about the product attributes.

MaxDiff Constructs a design consisting of choice sets that can be presented to respondents as part of a MaxDiff study. Respondents report only the most and least preferred options from among a small set of choices. This forces respondents to rank options in terms of preference, which often results in rankings that are more definitive than rankings obtained using standard preference scales.

JMP PRO **Covering Array** Constructs combinatorial designs that you can use to test software, networks, and other systems. A strength t covering array has the property that every combination of levels of every t factors appears in at least one run. Covering arrays allow for any number of categorical factors, each with an arbitrary number of levels. Disallowed combinations can be specified.

Space Filling Design Constructs designs for situations where the system of interest is deterministic or near-deterministic. A standard application involves creating a simpler surrogate model of a highly complex deterministic computer simulation model.

In a deterministic system, there is no variation. The goal is to minimize the difference between the fitted model and the true model (bias). Space-filling designs attempt to meet this goal either by spreading the design points out as far from each other as possible or by spacing the points evenly over the design region.

JMP provides seven space-filling design approaches. One of these approaches, the fast flexible filling design, accommodates categorical factors with any number of levels and supports linear constraints.

Accelerated Life Test Design Constructs and augments designs useful for testing products at extreme conditions which are intended to accelerate failure time. Use experimental results to predict reliability under normal operating conditions.

The life distribution can be lognormal or Weibull. Designs can include one or two accelerating factors. If there are two accelerating factors, you can choose to include their interaction. You can specify prior distributions for the acceleration model parameters. D-optimal and two types of I-optimal designs are available.

Nonlinear Design Constructs and augments designs that you use to fit models that are nonlinear in their parameters. You can construct a design using estimates from a model fit to existing data. You can also construct a design by applying prior knowledge if you do not have model-based estimates.

Balanced Incomplete Block Design Constructs design for testing a treatments in b blocks where only k treatments ($k < a$) can be run in any one block

Group Orthogonal Supersaturated Design Constructs supersaturated screening designs. They are appropriate in early stage work when the number of factors to be investigated is larger than the number of feasible runs. A group orthogonal supersaturated design is a special class of two-level supersaturated designs with properties that are desirable for model selection.

Supporting Platforms

Augment Design Adds runs to existing designs in such a way that the resulting design is optimal. Augment Design enables you to conduct experiments in an iterative fashion. You can replicate the design, add center points, create a fold-over design, add axial points, add points to create a space-filling design, or augment the design with a specified number of runs. You can group runs into blocks to distinguish the original runs from the augmented runs. You can add model effects that were not in the original model and specify requirements for these effects.

Fit Definitive Screening Design Analyzes definitive screening designs using a methodology called *Effective Model Selection for DSDs*. This methodology takes advantage of the special structure of definitive screening designs.

Fit Group Orthogonal Supersaturated Analyzes group orthogonal supersaturated designs. This analysis technique takes advantage of the group orthogonal structure of group orthogonal supersaturated designs.

Evaluate Design Provides diagnostics for an existing experimental design. The Evaluate Design platform provides various ways for you to assess the strengths and limitations of your design. The platform can be used with any data table, not only designs created using JMP.

Several diagnostics are provided:

- power analysis
- prediction variance plots
- estimation efficiency for parameters
- the alias matrix, showing the bias structure for model effects
- a color map showing absolute correlations among effects
- design efficiency values

Compare Designs Compares up to four designs to a reference design. Use to explore, evaluate, and compare design performance. Diagnostics show how the designs perform relative to each other and how they perform in an absolute sense.

Sample Size and Power Provides sample size and power calculations for a variety of testing situations: one or more sample means, a standard deviation, one or two proportions, counts per unit (Poisson mean), and sigma quality level. For these options, you specify two of three quantities to compute the third. These three quantities are the difference you want to detect, the sample size, and the power. If you supply only one of these values, a plot of the relationship between the other two values is provided.

You can compute the sample size required for a reliability test plan, where your goal is to estimate failure probabilities. You can also compute the sample size required for a reliability demonstration, where your goal is to demonstrate that a product meets or exceeds a specified standard.

Chapter 3

Starting Out with DOE

Example and Key Concepts

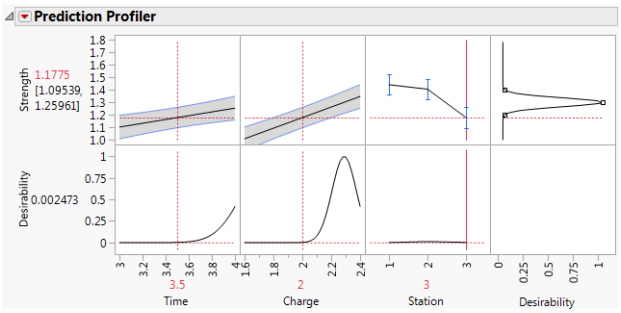
A designed experiment is a controlled set of tests designed to model and explore the relationship between factors and one or more responses. JMP includes a variety of tools that enable you to create efficient experimental designs that work for your situation. In particular, these classes of designs are available:

- The Custom Design platform customizes a design for your unique situation. It constructs designs that accommodate any number of factors of any type and factors that are difficult to change (split plot situations). You control the number of runs.
- The Definitive Screening Design platform constructs an innovative class of screening designs where main effects are not aliased with each other or with two-way interactions. These designs also allow estimation of quadratic terms.
- The Screening Design, Response Surface Design, Full Factorial Design, Mixture Design, and Taguchi Arrays platforms construct traditional experimental designs.
- The Covering Array, Choice Design, Space Filling Design, Accelerated Life Test Design, and Nonlinear Design platforms construct specialized design types.

The Evaluate Design and Augment Design platforms provide tools for evaluating and augmenting existing design. The Sample Size and Power platform addresses sample size and power calculations for specialized situations.

This chapter presents an example that illustrates the JMP approach to DOE. This chapter also discusses the framework for DOE, the workflow that supports design creation, and principles that are fundamental to DOE.

Figure 3.1 Example of a Profiler Plot



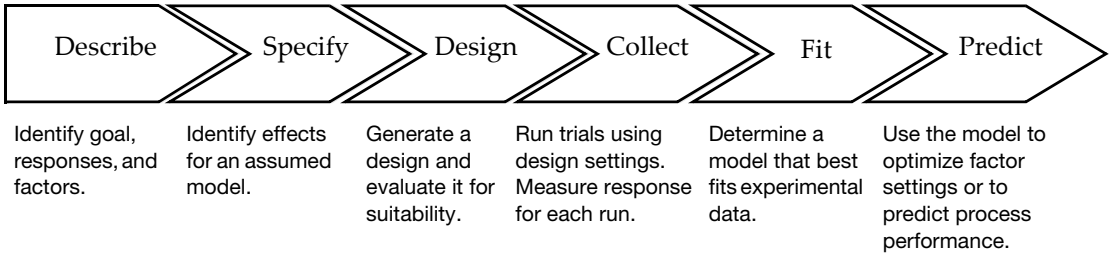
Contents

Overview of Experimental Design and the DOE Workflow 43
The Coffee Strength Experiment 44
Define the Study and Goals 45
Create the Design 47
Run the Experiment 52
Analyze the Data 53
The DOE Workflow: Describe, Specify, Design 59
Define Responses and Factors 60
Specify the Model 61
Generate the Design 62
Evaluate the Design 63
Make the Table 63
Principles and Guidelines for Experimental Design 64
Effect Hierarchy 64
Effect Heredity 64
Effect Sparsity 65
Center Points, Replicate Runs, and Testing 65

Overview of Experimental Design and the DOE Workflow

A six-step framework provides the structure for designing an experiment, running the experimental trials, and analyzing the results. Sound engineering and process knowledge is critical to all of these steps.

Figure 3.2 Framework for Experimental Design



You perform the first three steps in the DOE platforms. The end result is a design that can be run in your work environment. For a detailed description of the workflow for these three steps, see "The DOE Workflow: Describe, Specify, Design".

Describe Determine the goal of your experiment. Identify responses and factors.

Your goal might be to identify active factors, to find optimal factor settings, or to build a predictive model.

Specify Determine or specify an assumed model that you believe adequately describes the physical situation.

Your *assumed* model is an initial model that ideally contains all the effects that you want to estimate. In some platforms, you can explicitly build the model of interest. In others, the model is implicit in the choices that you make. For example, in the Screening Design platform, you might select a model with a given resolution. The resolution of the design determines which effects are confounded. Confounding of effects potentially leads to ambiguity about which effect is truly active.

Design Generate a design that is consistent with your assumed model. Evaluate this design to understand its strengths and limitations, and to ensure that it provides the information that you need, given your model and goals.

The Design Evaluation or Design Diagnostics outline in the design generation platform give you insight about the properties of your design.

The next step is the data collection phase, where the experiment is run under controlled conditions.

Collect Conduct each of the trials and record the response values.

After you run your experiment, scripts in the generated data table help you fit a model using platforms such as Fit Model and Screening. Depending on your goal, the model can help you identify active effects or find optimal settings.

Fit Fit your assumed model to the experimental data.

Use the JMP modeling platforms to fit and refine your model. In some situations, you might need to augment the design and perform additional runs to resolve model ambiguity.

Predict Use your refined model to address your experimental goals.

Determine which effects are active, find factor levels to optimize responses, or build a predictive model.

Designed experiments are typically used sequentially to construct process knowledge. A design strategy often begins with a screening design to narrow the list of potentially active factors. Then the identified factors are studied in designs that focus on building a better understanding of interactions and quadratic effects. Sometimes there is a need to augment a design to resolve ambiguities relating to the factors responsible for effects. The steps outlined in this section relate to conducting and analyzing a single experiment. However, you may require a sequence of experiments to achieve your goals.

The example in "The Coffee Strength Experiment" explicitly illustrates the steps in the DOE workflow process. It also shows how to use a data table script to analyze your experimental data. Many examples in the *Design of Experiments Guide* illustrate both the workflow that supports a good design and the analysis of the experimental data from the study.

The Coffee Strength Experiment

- "Define the Study and Goals"
- "Create the Design"
- "Run the Experiment"
- "Analyze the Data"

Define the Study and Goals

Your employer is a local mid-size coffee roaster. You need to address the strength of individually brewed twelve ounce cups of coffee. Your goal is to determine which factors have an effect on coffee strength and to find optimal settings for those factors.

Response

The response is coffee Strength. It is measured as total dissolved solids, using a refractometer. The coffee is brewed using a single cup coffee dripper and measured five minutes after the liquid is released from the grounds.

Previous studies indicate that a strength reading of 1.3 is most desirable, though the strength is still acceptable if it falls between 1.2 and 1.4.

Factors

Four factors are identified for the study: Grind, Temperature, Time, and Charge. Coffee is brewed at three stations in the work area. To account for variation due to brewing location, Station is included in the study as a blocking factor. The following describes the factors:

- Grind is the coarseness of the grind. Grind is set at two levels, Medium and Coarse.
- Temperature is the temperature in degrees Fahrenheit of the water measured immediately before pouring it over the grounds. Temperature is set at 195 and 205 degrees Fahrenheit.
- Time is the brewing time in minutes. Time is set at 3 or 4 minutes.
- Charge is the amount of coffee placed in the cone filter, measured in grams of coffee beans per ounce of water. Charge is set at 1.6 and 2.4.
- Station is the location where the coffee is brewed. The three stations are labeled as 1, 2, and 3.

Table 3.1 summarizes information about the factors and their settings. The factors and levels are also given in the Coffee Factors.jmp sample data table, located in the Design Experiment folder.

Table 3.1 Factors and Range of Settings for Coffee Experiment

Factor	Role	Range of Settings
Grind	Categorical	Medium, Coarse
Temperature	Continuous	195 - 205
Time	Continuous	3 - 4
Charge	Continuous	1.6 - 2.4

Table 3.1 Factors and Range of Settings for Coffee Experiment *(Continued)*

Factor	Role	Range of Settings
Station	Blocking	1, 2, 3

Note the following:

- Grind is categorical with two levels.
- Temperature, Time, and Charge are continuous.
- Station is a blocking factor with three levels.

All factors can be varied and reset for each run. There are no hard-to-change factors for this experiment.

The apparatus used in running the coffee experiment is shown in Figure 3.3. This is the setup at one of the three brewing stations. The two other stations have the same type of equipment.

Figure 3.3 Coffee Experiment Apparatus

Number of Runs

Based on the resources and time available, you determine that you can conduct 12 runs in all. Since there are three stations, you conduct 4 runs at each station.

Create the Design

Create the design following the steps in the design workflow process outlined in "The DOE Workflow: Describe, Specify, Design":

- "Define Responses and Factors"
- "Specify the Model"
- "Generate the Design"
- "Evaluate the Design"
- "Make the Table"

Define Responses and Factors

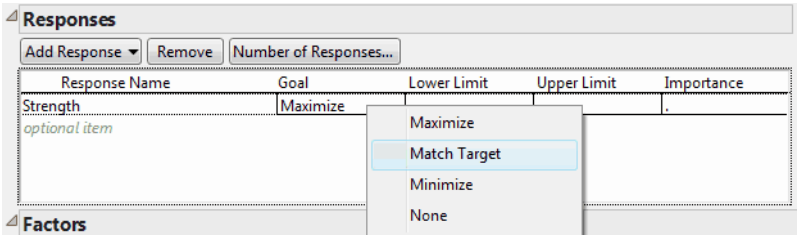
In the first outlines that appear, enter information about your response and factors.

Responses

1. Select **DOE > Custom Design**.
2. Double-click Y under Response Name and type Strength.

   Note that the default Goal is **Maximize**. Your goal is to find factor settings that enable you to brew coffee with a target strength of 1.3, within limits of 1.2 and 1.4.
3. Click the default Goal of **Maximize** and change it to **Match Target**.

Figure 3.4 Selection of Match Target as the Goal



4. Click under **Lower Limit** and type 1.2.
5. Click under **Upper Limit** and type 1.4.
6. Leave the area under **Importance** blank.

   Because there is only one response, that response is given Importance 1 by default.

The completed Responses outline appears in Figure 3.5.

Factors

Enter factors either manually or from a pre-existing table that contains the factors and settings. If you are designing a new experiment, you must first enter the factors manually. Once you have saved the factors to a data table using the Save Factors option, you can load them using the saved table.

For this example, you can choose either option. See "Entering Factors Manually" or see "Entering Factors Using Load Factors".

Entering Factors Manually

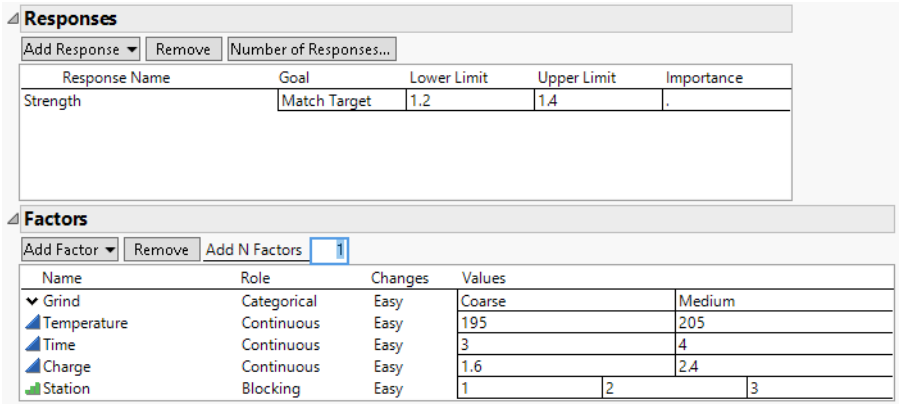
1. Click **Add Factor > Categorical > 2 Level**.
2. Type Grind over the default Name of X1.

   Note that Role is set to Categorical, as requested. The Changes attribute is set to Easy by default, indicating that Grind settings can be reset for every run.
3. Click the default Values, L1 and L2, and change them to Coarse and Medium.
4. Type 3 next to **Add N Factors.** Then click **Add Factor > Continuous**.
5. Type the factor names and values over the default entries:
   - Temperature (195 and 205)
   - Time (3 and 4)
   - Charge (1.6 and 2.4)
6. Click **Add Factor > Blocking > 4 runs per block.**

   Recall that your run budget allows for 12 runs. You want to balance these runs among the three stations.
7. Type Station over the default Name of X5.

   Notice that Role is set to Blocking and that only one setting for Values appears. This is because JMP cannot determine the number of blocks until the desired number of runs is specified. Once you specify the Number of Runs in the Design Generation outline, JMP updates the number of levels for Station to what is required.

Figure 3.5 Completed Responses and Factors Outlines



8. Click **Continue**.

   The following outlines are added to the Custom Design window:

   – Define Factor Constraints (not used in this example)
   – Model
   – Alias Terms
   – Design Generation

Entering Factors Using Load Factors

Enter factors using a table containing factor information:

1. Click the Custom Design red triangle and select **Load Factors**.
2. Select **Help > Sample Data Library** and open Design Experiment/Coffee Factors.jmp.

   After loading the factors, the Custom Design window is updated with additional outlines:

   – Define Factor Constraints (not used in this example)
   – Model
   – Alias Terms
   – Design Generation

Define Factor Constraints

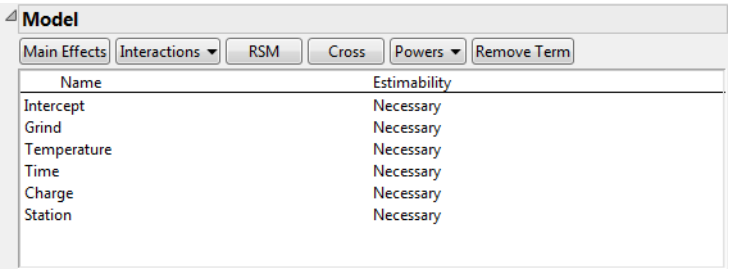
The Define Factor Constraints outline appears once you have entered your factors manually and clicked Continue, or once you have loaded the factors from the factor table. Adding factor constraints, if you have any, is part of the Responses and Factors step. Since there are no constraints on factor settings for this design, leave this outline unchanged.

Specify the Model

Model Outline

Figure 3.7 shows the Model outline. The Model outline is where you specify your assumed model, which contains the effects that you want to estimate. See "Specify". The list that appears by default shows all main effects as **Necessary**, indicating that the design is capable of estimating all main effects. Because your main interest at this point is in the main effects of the factors, you do not add any effects to the Model outline.

Figure 3.6 Model Outline with Main Effects Only



Steps to Duplicate Results (Optional)

Because the Custom Design algorithm begins with a random starting design, your design might differ from the one shown in Figure 3.8. To obtain a design with exactly the same runs, set the random seed and number of starts before generating your design:

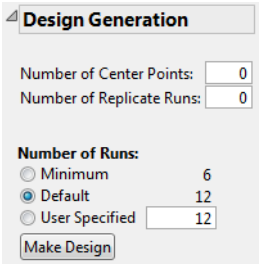
1. Click the Custom Design red triangle and select **Set Random Seed**.
2. Type 569534903.
3. Click **OK**.
4. Click the Custom Design red triangle and select **Number of Starts**.
5. Type 100.
6. Click **OK**.

Note: Setting the Random Seed and Number of Starts reproduces the exact design shown in this example. However, the rows in the design table might be in a different order. In constructing a design on your own, these steps are not necessary.

Generate the Design

In the Design Generation outline, you can enter additional details about the structure and size of your design. The Default design is shown as having 12 runs. Recall that your design budget allows for 12 runs ("Number of Runs").

Figure 3.7 Design Generation Outline



1. Click **Make Design**.

   The Design and Design Evaluation outlines are added to the Custom Design window. The Output Options panel also appears.

   The Design outline shows the design (Figure 3.8). If you did not set the random seed and number of starts as described in "Steps to Duplicate Results (Optional)", your design might be different from the one in Figure 3.8. This is because the algorithm used to generate the design begins with a random starting design.

Figure 3.8 Design for Coffee Experiment

Design

Run	Grind	Temperature	Time	Charge	Station
1	Coarse	195	3	1.6	1
2	Medium	195	4	2.4	2
3	Medium	195	4	1.6	3
4	Medium	205	4	2.4	1
5	Coarse	205	3	2.4	2
6	Medium	205	3	2.4	3
7	Medium	205	3	1.6	1
8	Coarse	205	4	1.6	2
9	Coarse	205	4	1.6	3
10	Coarse	195	4	2.4	1
11	Medium	195	3	1.6	2
12	Coarse	195	3	2.4	3

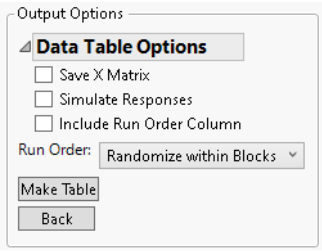
Evaluate the Design

The Design Evaluation outline provides various ways to evaluate your design. This is an important topic, but for simplicity, it is not covered in the context of this example. See "Evaluate Designs".

Make the Table

Specify the order of runs in your data table using the Output Options panel. The default selection, **Randomize within Blocks**, is appropriate. This selection arranges the runs in a random order for each Station.

Figure 3.9 Output Options



1. Click **Make Table**.

   The data table shown in Figure 3.10 opens. Keep in mind that, if you did not set the random seed and number of starts as described in "Steps to Duplicate Results (Optional)", your design table might be different. Your design table represents an alternative optimal design.

Figure 3.10 Custom Design Table

Custom Design

Design	Custom Design
Criterion	D Optimal

- Model
- Evaluate Design
- Generalized Regression
- DOE Dialog

Columns (6/0)

- Grind ✱
- Temperature ✱
- Time ✱
- Charge ✱
- Station ✱
- Strength ✱

	Grind	Temperature	Time	Charge	Station	Strength
1	Coarse	195	4	2.4	1	•
2	Medium	205	3	1.6	1	•
3	Medium	205	4	2.4	1	•
4	Coarse	195	3	1.6	1	•
5	Medium	195	4	2.4	2	•
6	Medium	195	3	1.6	2	•
7	Coarse	205	4	1.6	2	•
8	Coarse	205	3	2.4	2	•
9	Medium	195	4	1.6	3	•
10	Medium	205	3	2.4	3	•
11	Coarse	205	4	1.6	3	•
12	Coarse	195	3	2.4	3	•

Note the asterisks in the Columns panel to the right of the factors and response. These indicate column properties that have been saved to the columns in the data table. These column properties are used in the analysis of the data. See "Factors" and "Factor Column Properties".

Run the Experiment

At this point, you perform the experiment. At each Station, four runs are conducted in the order shown in the design table. Equipment and material are reset between runs. For example, if two consecutive runs require water at 195 degrees, separate 12-ounce batches of water are heated to 195 degrees after the heating container cools. The Strength measurements are recorded.

Your design and the experimental results for Strength are given in the Coffee Data.jmp sample data table (Figure 3.11), located in the Design Experiment folder.

Figure 3.11 Coffee Design with Strength Results

	Grind	Temperature	Time	Charge	Station	Strength
1	Mediu...	205	4	2.4	1	1.78
2	Coarse	195	3	1.6	1	1.25
3	Mediu...	205	3	1.6	1	1.10
4	Coarse	195	4	2.4	1	1.63
5	Coarse	205	4	1.6	2	1.26
6	Mediu...	195	4	2.4	2	1.63
7	Mediu...	195	3	1.6	2	1.22
8	Coarse	205	3	2.4	2	1.51
9	Coarse	205	4	1.6	3	1.07
10	Coarse	195	3	2.4	3	1.26
11	Mediu...	195	4	1.6	3	1.13
12	Mediu...	205	3	2.4	3	1.25

Analyze the Data

The Custom Design platform facilitates the task of data analysis by saving a Model script to the design table that it creates (Figure 3.10). Run this script after you conduct your experiment and enter your data. The script opens a Fit Model window containing the effects that you specified in the Model outline of the Custom Design window.

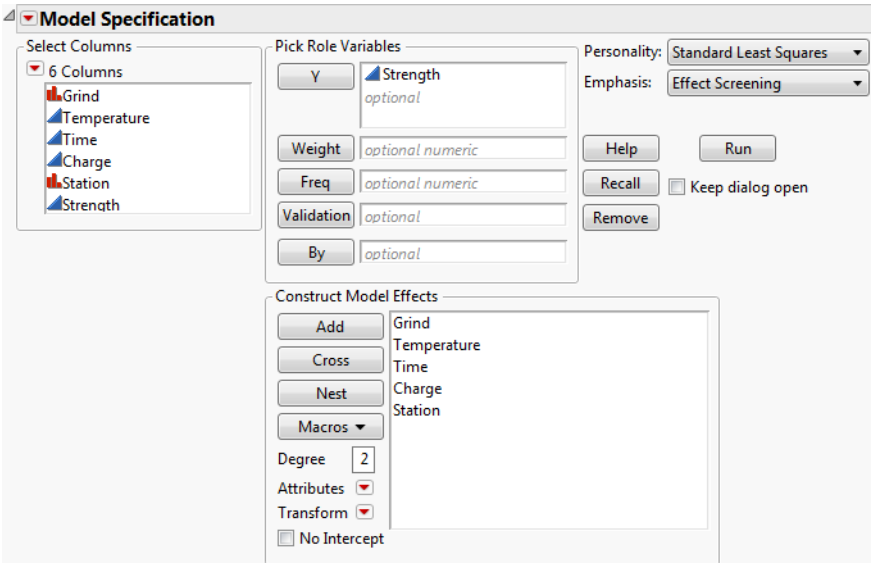
Fit the Model

1. Select **Help > Sample Data Library** and open Design Experiment/Coffee Data.jmp.

   In the Table panel, notice the Model script created by Custom Design.
2. Click the green triangle next to the Model script.

   The Model Specification window shows the effects that you specified in the Model outline.

Figure 3.12 Model Specification Window

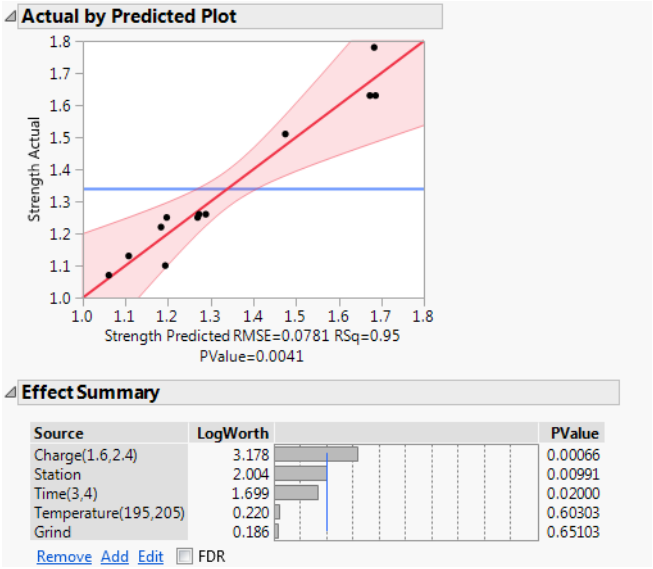


3. Select the **Keep dialog open** option.
4. Click **Run**.

Analyze the Model

The Effect Summary and Actual by Predicted Plot reports give high-level information about the model.

Figure 3.13 Effect Summary and Actual by Predicted Plot for Full Model



Note the following:

- The Actual by Predicted Plot shows no evidence of lack of fit.
- The model is significant, as indicated by the Actual by Predicted Plot. The notation **P = 0.0041**, shown below the plot, gives the significance level of the overall model test.
- The Effect Summary report shows that Charge, Station, and Time are significant at the 0.05 level.
- The Effect Summary report also shows that Temperature and Grind are not significant.

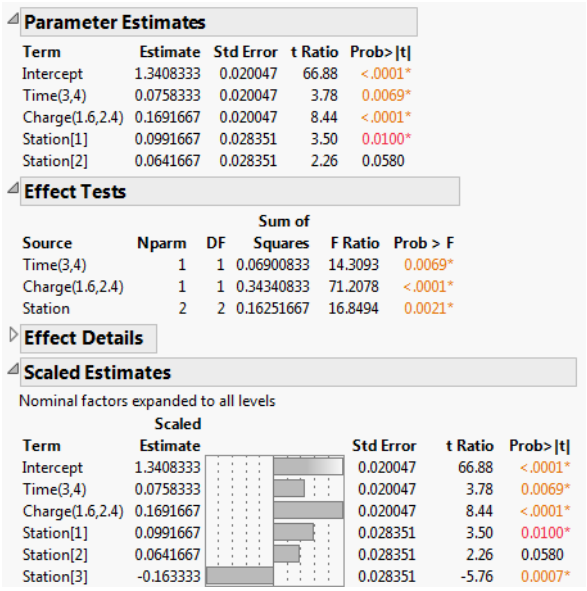
Reduce the Model

Because Temperature and Grind appear not to be active, they contribute random noise to the model. Refit the model without these effects to obtain more precise estimates of the model parameters associated with the active effects.

1. In the Model Specification window, select Temperature and Grind in the Construct Model Effects list.
2. Click **Remove**.
3. Confirm that the model **Emphasis** is set to **Effect Screening**.

   The Effect Screening emphasis presents reports (such as the Prediction Profiler) that are useful for analyzing experimental designs.
4. Click **Run**.

Figure 3.14 Partial Output for the Reduced Model



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	1.3408333	0.020047	66.88	<.0001*
Time(3,4)	0.0758333	0.020047	3.78	0.0069*
Charge(1.6,2.4)	0.1691667	0.020047	8.44	<.0001*
Station[1]	0.0991667	0.028351	3.50	0.0100*
Station[2]	0.0641667	0.028351	2.26	0.0580

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F
Time(3,4)	1	1	0.06900833	14.3093	0.0069*
Charge(1.6,2.4)	1	1	0.34340833	71.2078	<.0001*
Station	2	2	0.16251667	16.8494	0.0021*

Effect Details

Scaled Estimates

Nominal factors expanded to all levels

Term	Scaled Estimate	Std Error	t Ratio	Prob>|t|
Intercept	1.3408333	0.020047	66.88	<.0001*
Time(3,4)	0.0758333	0.020047	3.78	0.0069*
Charge(1.6,2.4)	0.1691667	0.020047	8.44	<.0001*
Station[1]	0.0991667	0.028351	3.50	0.0100*
Station[2]	0.0641667	0.028351	2.26	0.0580
Station[3]	-0.163333	0.028351	-5.76	0.0007*

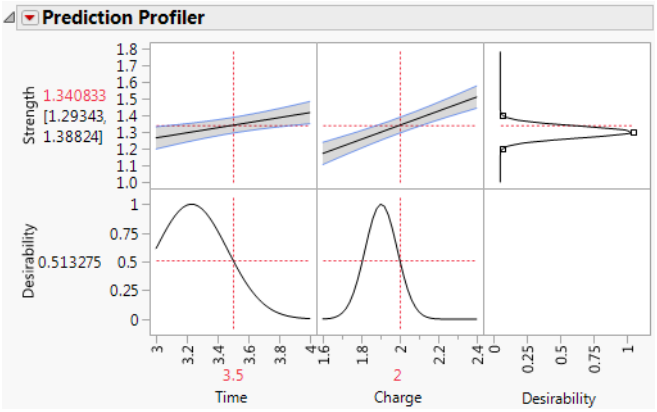
Note the following:

- The Effect Tests report shows that all three effects remain significant.
- The Scaled Estimates report further indicates that the Station[1]and Station[3] means differ significantly from the average response of Strength.
- Note that the Estimates that appear in the Parameter Estimates report are identical to their counterparts in the Scaled Estimates report. This is because the effects are coded. See "Coding".
- The estimate of the Station[3] effect only appears in the Scaled Estimates report, where nominal factors are expanded to show estimates for all their levels.
- The Parameter Estimates report gives estimates for the model coefficients where the model is specified in terms of the coded effects.

Explore the Model

The Prediction Profiler appears at the bottom of the report.

Figure 3.15 Prediction Profiler



Recall that, in designing your experiment, you set a response Goal of Match Target with limits of 1.2 and 1.4. JMP uses this information to construct a desirability function to reflect your specifications. See "Factors".

Note the following in Figure 3.15:

- The first two plots in the top row of the graph show how Strength varies for one of the factors, given the setting of the other factor. For example, when Charge is 2, the line in the plot for Time shows how predicted Strength changes with Time.
- The values to the left of the top row of plots give the Predicted Strength (in red) and a confidence interval for the mean Strength for the selected factor settings.
- The right-most plot in the top row shows the desirability function for Strength. The desirability function indicates that the target of 1.3 is most desirable. Desirability decreases as you move away from that target. Desirability is close to 0 at the limits of 1.2 and 1.4.
- The plots in the bottom row show the desirability trace for each factor at the setting of the other factor.
- The value to the left of the bottom row of plots gives the Desirability of the response value for the selected factor settings.

Explore various factor settings by dragging the red dashed vertical lines in the columns for Time and Charge. Since there are no interactions in the model, the profiler indicates that increasing Charge increases Strength. Also, Strength seems to be more sensitive to changes in Charge than to changes in Time.

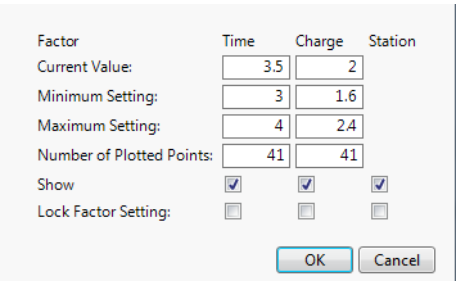
Since Station is a blocking factor, it does not appear in the Prediction Profiler. However, you might like to see how predicted Strength varies by Station. Include Station in the Prediction Profiler:

1. Click the Prediction Profiler red triangle and select **Reset Factor Grid**.

A Factor Settings window appears with columns for Time, Charge, and Station. Under Station, notice that the box corresponding to **Show** is not selected. This indicates that Station is not shown in the Prediction Profiler.

2. Select the box under Station in the row corresponding to **Show**.
3. Deselect the box under Station in the row corresponding to **Lock Factor Setting**.

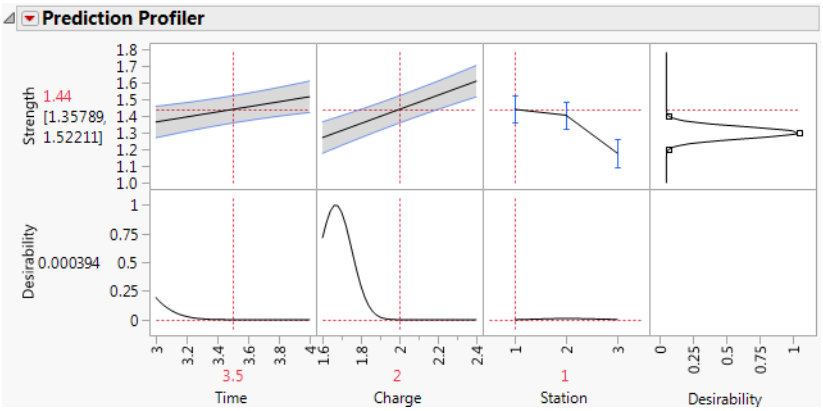
Figure 3.16 Factor Settings Window



4. Click **OK**.

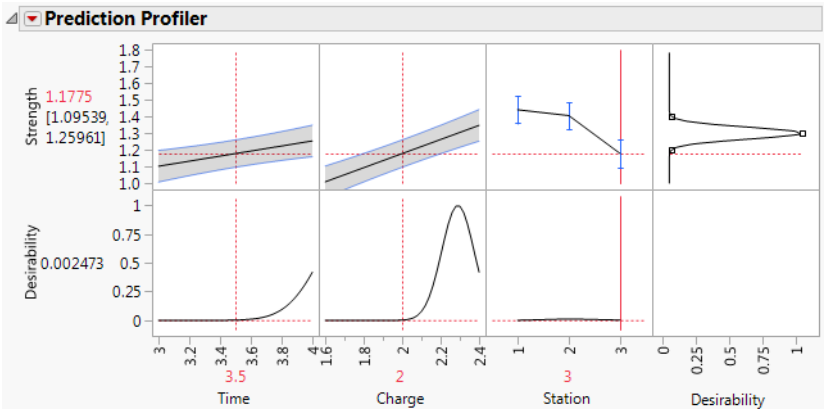
   Plots for Station appear in the Prediction Profiler.
5. Click in either plot above Station to insert a dashed red vertical line.
6. Move the dashed red vertical line to Station 1.

Figure 3.17 Prediction Profiler Showing Results for Station 1



7. Move the dashed red vertical line to Station 3.

Figure 3.18 Prediction Profiler Showing Results for Station 3



The predicted Strength in the center of the design region for Station 1 is 1.44. For Station 3, the predicted Strength is about 1.18. The magnitude of the difference indicates that you need to address Station variability. Better control of Station variation should lead to more consistent Strength. Once Station consistency is achieved, you can determine common optimal settings for Time and Charge.

The process that you used to construct the design for the coffee experiment followed the steps in the DOE workflow. The next section describes the DOE workflow in more detail.

The DOE Workflow: Describe, Specify, Design

The DOE platforms are structured as a series of steps that present the workflow that is intrinsic to designing experiments. Once you complete each step, you click **Continue** to move to the next step. The elements described in this section are common to nine of the design of experiments platforms. These are the platforms that are addressed in this section:

- Custom Design
- Definitive Screening Design
- Screening Design
- Response Surface Design
- Full Factorial Design
- Mixture Design
- JMP PRO Covering Array
- Space Filling Design
- Taguchi Arrays

Three special-purpose platforms differ substantially: Choice Designs, Accelerated Life Test Design, and Nonlinear Design. These three platforms are not addressed in this section.

This section describes the steps in the DOE workflow. It also discusses their implementation in the various design platforms.

Define Responses and Factors

In the Describe step of the experimental design framework:

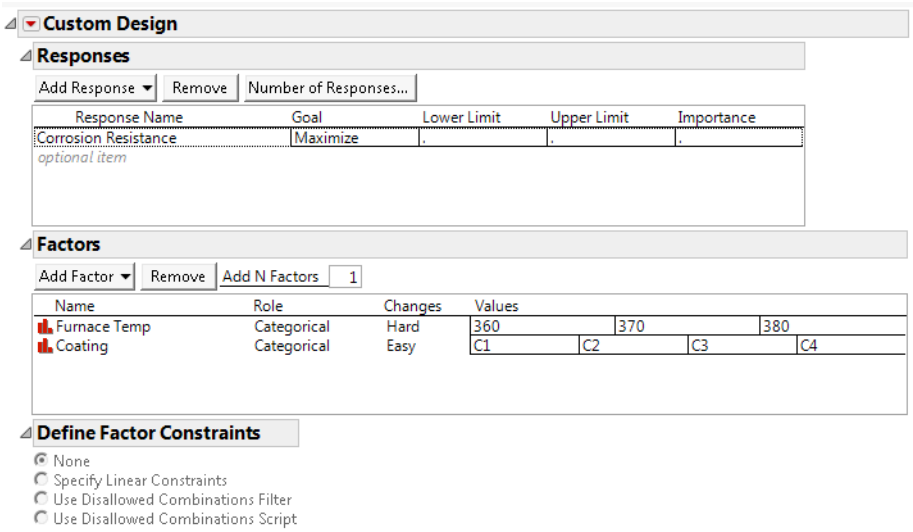
- You identify the responses and factors of interest.
- You determine your goals for the experiment. Do you want to maximize the response, or hit a target? What is that target? Or do you simply want to identify which factors have an effect on the response?
- You identify factor settings that describe your experimental range or design space.

When they open, most of the JMP DOE platforms display outlines where you can list your responses and your factors. The Responses outline is common across platforms. There you insert your responses and additional information, such as the response goal, lower limit, upper limit, and importance.

The Factors outline varies across platforms. This is to accommodate the types of factors and specific design situations that each platform addresses. In certain platforms, once responses and factors are entered, a Define Factor Constraints outline appears after you click Continue. In this outline, you can constrain the values of the factors that are available for the design.

Figure 3.19 shows the Responses and Factors outline using the Custom Design platform for constructing the design in the Box Corrosion Split-Plot.jmp sample data table, located in the Design Experiment folder. Also shown is the Define Factor Constraints outline, which appears once you click Continue. The Define Constraints outline enables you to specify restrictions that your factor settings must satisfy.

Figure 3.19 Responses and Factors for Box Corrosion Split-Plot Experiment



Specify the Model

Once you have completed filling in the Responses and Factors outlines, click the **Continue** button. This brings you to the next phase of design construction, where you either explicitly or implicitly choose an assumed model.

The Custom Design platform enables you to explicitly specify the model that you want to fit. The design that is generated is optimal for this model. The other design platforms do not allow you to explicitly specify your model. For example, in the screening platform, one option enables you to choose from a list of full factorial, fractional factorial, and Plackett-Burman designs. The aliasing relationships in these designs implicitly define the models that you can fit.

In Custom Design, when you click Continue after filling in the Responses and Factors, you see the Model outline. An example, for the design used in the Box Corrosion Split-Plot.jmp sample data table, is shown in Figure 3.20. The assumed model requires that the Furnace Temp and Coating main effects, and their interaction, be estimable. The design that is generated guarantees estimability of these effects.

In most other platforms, clicking Continue gives you a collection of designs to choose from. In Full Factorial, Continue takes you directly to Output Options, since the design is determined once the Factors outline is completed.

Figure 3.20 Model Outline for Box Corrosion Split-Plot Experiment

Model

Main Effects | Interactions | RSM | Cross | Powers | Remove Term

Name	Estimability
Intercept	Necessary
Furnace Temp	Necessary
Coating	Necessary
Furnace Temp*Coating	Necessary

Generate the Design

Most of the DOE platforms give you some control over the size of the final design. In Custom Design, you can specify the number of runs and, when appropriate, the number of center points and replicate runs. In other platforms, you have various degrees of flexibility. Often you can specify the number of center points, replicate runs, or replicates of the design.

Once you have specified your options in terms of the number of runs, click Make Design. The DOE window is updated to show your design in a Design outline.

The Design outline for a 24-run custom design for the Box Corrosion Split-Plot.jmp experiment is shown in Figure 3.21. Because **Changes** for Furnace Temp was specified as **Hard**, a Whole Plots factor is constructed to represent the random blocks of settings for Furnace Temp.

Figure 3.21 Design Outline for Box Corrosion Split-Plot Experiment

Design

Run	Whole Plots	Furnace Temp	Coating
1	1	370	C2
2	1	370	C3
3	1	370	C1
4	1	370	C4
5	2	370	C3
6	2	370	C4
7	2	370	C2
8	2	370	C1
9	3	360	C3
10	3	360	C2
11	3	360	C4
12	3	360	C1
13	4	380	C4
14	4	380	C3
15	4	380	C2
16	4	380	C1
17	5	380	C4
18	5	380	C2
19	5	380	C1
20	5	380	C3
21	6	360	C3
22	6	360	C1
23	6	360	C4
24	6	360	C2

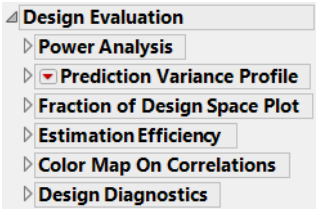
Note: For a custom design, once you have created your design, the controls in the Model and Alias Outlines are disabled. Use the Back button to change model or alias terms.

Evaluate the Design

When you click Make Design, in most platforms, a Design Evaluation outline appears. Here you can explore the design that you created in terms of its power to detect effects, its prediction variance, its estimation efficiency, its aliasing relationships, the correlations between effects, and other design efficiency measures. The Design Evaluation outline for a Custom Design is shown in Figure 3.22. Design Evaluation is covered in "Evaluate Designs".

For some platforms, other types of design diagnostics are appropriate. For example, Space Filling Design provides a Design Diagnostics outline with metrics specific to space-filling designs. Covering Array provides a Metrics outline with measures that are specific to coverage.

Figure 3.22 Design Evaluation Outline in Custom Design

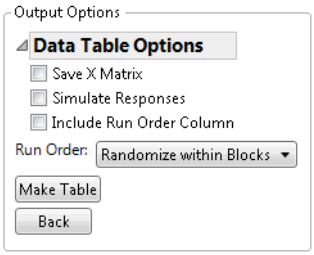


Make the Table

Most platforms provide an Output Options node or panel. Depending on the platform and the design, you can use the Output Options panel to specify additional design structure. For example, you can specify the number of runs, center points, replicates, or the order in which you want the design runs to appear in the generated data table.

The Output Options panel shown in Figure 3.23 is for the experiment in the Wine Data.jmp sample data table, located in the Design Experiment folder. In this example, you can choose various Run Order options and construct the design data table. Or, you can choose to go Back and restructure your design.

Figure 3.23 Output Options Panel for Wine Experiment



Principles and Guidelines for Experimental Design

Certain principles underlie the design of experiments and the analysis of experimental data. The principles of effect hierarchy, effect heredity, and effect sparsity relate primarily to model selection. These principles help you reduce the set of possible models in searching for a best model. See Hamada and Wu (1992), Wu and Hamada (2009), and Goos and Jones (2011).

- "Effect Hierarchy"
- "Effect Heredity"
- "Effect Sparsity"
- "Center Points, Replicate Runs, and Testing"

Effect Hierarchy

In regression modeling, the principle of *effect hierarchy* maintains that main (first-order) effects tend to account for the largest amounts of variation in the response. Second-order effects, that is, interaction effects and quadratic terms, are next in terms of accounting for variation. Then come higher-order terms, in hierarchical order.

Here are the implications for modeling: main effects are more likely to be important than second-order effects; second-order effects are more likely to be important than third-order effects; and so on, for higher-order terms.

Effect Heredity

The principle of *effect heredity* relates to the inclusion in the model of lower-order components of higher-order effects. The motivation for this principle is observational evidence that factors with small main effects tend not to have significant interaction effects.

Strong effect heredity requires that all lower-order components of a model effect be included in the model. Suppose that a three-way interaction (ABC) is in the model. Then all of its component main effects and two-way interactions (A, B, C, AB, AC, BC) must also be in the model.

Weak effect heredity requires that only a sequence of lower-order components of a model effect be included. If a three-way interaction is in the model, then the model must contain one of the factors involved and one two-way interaction involving that factor. Suppose that the three-way interaction ABC is in the model. Then if B and BC are also in the model, the model satisfies weak effect heredity.

For continuous factors, effect heredity ensures that the model is invariant to changes in the location and scale of the factors.

Effect Sparsity

The principle of *effect sparsity* asserts that most of the variation in the response is explained by a relatively small number of effects. Screening designs, where many effects are studied, rely heavily on effect sparsity. Experience shows that the number of runs used in a screening design should be at least twice the number of effects that are likely to be significant.

Center Points, Replicate Runs, and Testing

Several DOE platforms enable you to add center points (for continuous factors), replicate runs, or full replicates of the design, to your design. Here is some background relative to adding design points.

Adding Center Points

Center points for continuous factors enable you to test for lack of fit due to nonlinear effects. Testing for lack of fit helps you determine whether the error variance estimate has been inflated due to a missing model term. This can be a wise investment of runs.

You can replicate runs solely at center points or you can replicate other design runs. JMP uses replicate runs to construct a model-independent error estimate (pure-error estimate).This pure-error estimate enables you to test for lack of fit.

Be aware that center points do not help you obtain more precise estimates of model effects. They enable you to test for evidence of curvature, but do not identify the responsible nonlinear effects.

To identify the source of curvature, you must set continuous factors at a minimum of three levels. Definitive screening designs are three-level designs with the ability to detect and identify any factors causing strong nonlinear effects on the response. See “Definitive Screening Designs”.

Adding Replicate Runs

If your run budget allows, you can either replicate runs or distribute new runs optimally within the design space. Adding replicate runs adds precision for some estimates and improves the power of the lack of fit test. However, for a given run budget, adding replicate runs generally lowers the ability of the design to estimate model effects. You are not able to estimate as many terms as you could by distributing the runs optimally within the design space.

Testing for Lack of Fit

Designed experiments are typically constructed to require as few runs as possible, consistent with the goals of the experiment. With too few runs, only extremely large effects can be detected. For example, for a given effect, the *t* test statistic is the ratio of the change in response means to their standard error. If there is only one error degree of freedom (df), then the critical value of the test exceeds 12. So, for such a nearly saturated design to detect an effect, it has to be very large.

A similar observation applies to the lack-of-fit test. The power of this test to detect lack-of-fit depends on the numbers of degrees of freedom in the numerator and denominator. If you have only 1 df of each kind, you need an *F* value that exceeds 150 to declare significance at the 0.05 level. If you have 2 df of each kind, then the *F* value must exceed 19. In order for the test to be significant in this second case, the lack-of-fit mean square must be 19 times larger than the pure error mean square. It is also true that the lack-of-fit test is sensitive to outliers.

For more information about the Lack of Fit test, see *Fitting Linear Models*.

Determining the Number of Runs

In industrial applications, each run is often very costly, so there is incentive to minimize the number of runs. To estimate the fixed effects of interest, you need only as many runs as there are terms in the model. To determine whether the effects are active, you need a reasonable estimate of the error variance. Unless you already have a good estimate of this variance, consider adding at least 4 runs to the number required to estimate the model terms.

Chapter 4

Custom Designs

Construct Designs That Meet Your Needs

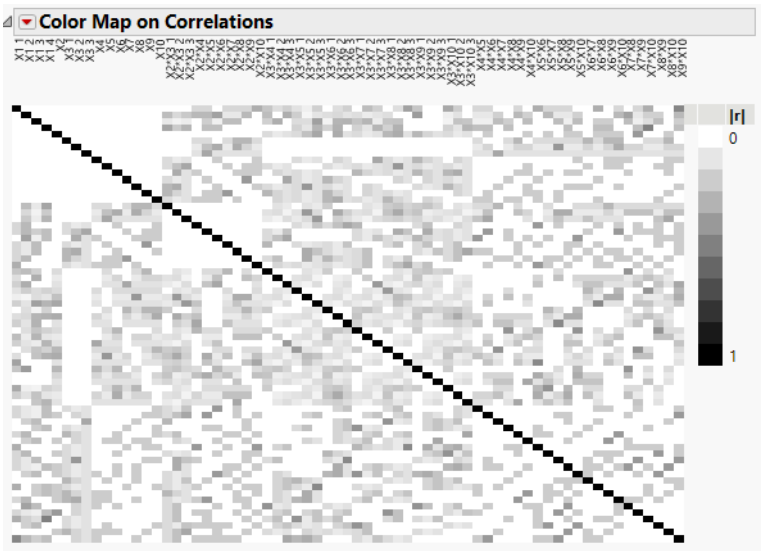
Use the Custom Design platform to construct optimal designs that are custom built for your specific experimental setting. Generally, a custom design is more cost-effective than a design obtained using alternative methods. You can perform the following tasks:

- Enter factors of many different types.
- Specify constraints on the design space.
- Indicate which effects are necessary to estimate and which are desirable to estimate, if possible, given the number of runs.
- Specify a number of experimental runs that matches your budget.

The Custom Design platform constructs a wide variety of designs, including these special design types: Screening, Response Surface, Mixture, Random Block, Split Plot, Split-Split-Plot, and Two-Way Split Plot.

This chapter contains a detailed example of how to use the Custom Design platform, followed by information about the platform. See also "Examples of Custom Designs".

Figure 4.1 Color Map for Absolute Correlations



Contents

Overview of Custom Design 69
Example of a Custom Design 71
Build a Custom Design 88
Responses 88
Factors 90
Define Factor Constraints 97
Alias Terms 102
Design Generation 103
Design 106
Design Evaluation 106
Output Options 107
Custom Design Options 109
Simulate Responses 113
Save X Matrix 115
Number of Starts 117
Design Search Time 118
Technical Details for Custom Designs 119
Designs with Randomization Restrictions 119
Covariates with Hard-to-Change Levels 128
Numbers of Whole Plots and Subplots 128
Optimality Criteria 129
D-Efficiency 134
Coordinate-Exchange Algorithm 134

Overview of Custom Design

Use the Custom Design platform to construct an optimal design custom built for your specific experimental needs.

You can include a wide range of factor types, including the following:

- Continuous
- Discrete numeric (with any number of levels)
- Categorical (with any number of levels)
- Blocking (with a specified number of runs per block)
- Covariate
- Mixture
- Constant
- Uncontrolled

Specify the Region of Operability

You can restrict your experimental region to reflect your operating conditions using linear factor constraints or disallowed combinations. In particular, restrictions can be specified for categorical, continuous, and discrete numeric factors. See "Define Factor Constraints".

Specify Factors with Hard-to-Change Levels

For continuous, discrete numeric, categorical, and mixture factors, you can indicate two levels of difficult-to-change factors. These difficulty levels are represented by whole plots or whole plots and split plots. You can also specify hard-to-change covariates.

Specify the Effects of Primary Interest

You can explicitly specify your *assumed* model. Your assumed model is an initial model that ideally contains all the effects that you want to estimate. Your model can contain any combination of main effects, interactions, response surface effects, and polynomial effects (up to the fifth power). You can specify the effects for which estimability is necessary and those for which estimability is desired. Custom Design uses a Bayesian optimality approach to estimate effects whose estimability is desired, subject to the number of runs. See "Model".

Specify the Number of Runs

The Custom Design platform enables you to specify the number of runs that matches the budget for your experimental situation. The platform indicates the minimum number of runs that can be used to estimate the required effects and provides a default number of runs. These values can serve as a guide for determining a feasible number of runs. See "Design Generation".

Construct the Appropriate Design Type

Custom Design can construct a wide variety of design types. These include classical designs and random block designs. For examples of different design types, see "Examples of Custom Designs".

Construct an Optimal Design

Given your specific requirements, the Custom Design platform constructs a design that is optimal. The algorithm supports several optimality criteria:

- D optimality
- I optimality
- Bayesian D and I optimality (using If Possible effects)
- A optimality
- Alias optimality

See "Optimality Criteria".

Designs are constructed using the coordinate-exchange algorithm (Meyer and Nachtsheim, 1995). See "Coordinate-Exchange Algorithm".

Example of a Custom Design

Use the custom designer to design an experiment to investigate the effect of wine processing factors on wine taste. Your employer grows two varieties of Pinot Noir grapes that can be processed in different ways. Your goal is to determine which factors affect the taste of Pinot Noir wine. Before the grapes are processed, you set up your experimental design. Once processed, the wine samples are aged for 12 months, then filtered and bottled. At this point, the wine samples are rated for quality by expert wine tasters.

Response

Most of your vineyard's product is sold to five large wine distributors. You arrange for a wine-tasting expert from each distributor to evaluate the wine samples for quality. To maximize the number of factors that you can study, you decide that each expert must rate eight different samples. This means that your design needs to have 40 wine samples, or runs.

The ratings follow a 0 – 20 scale, where 0 is the worst and 20 is the best. Rating, the variable consisting of the experts' ratings, is the response of interest. You want to identify the wine-related factors that maximize the response.

Blocking Factor

A blocking factor is used to account for variation that is not necessarily of direct interest. A blocking factor is particularly effective when observations taken at one factor level are expected to be more similar than observations at different levels. In your experiment, ratings by one expert are likely to have similar characteristics and to differ from ratings by a different expert. Yet, you are interested in which properties of the wine lead to high ratings by all experts.

Because each rater tastes eight wines, Rater is a blocking factor with eight runs per block. For this experiment, only these five raters are of concern. You are not interested in generalizing to a larger population of raters.

Process Factors

You have identified nine process factors for the study. These include the grape variety, the field on which the grapes were grown, and seven other factors related to processing. You can experiment with any combination of these factors. Also, the factors can be varied at will as part of the experiment. Relative to the experiment, these factors are all "Easy" to change. For information about specifying factor changes, see "Changes and Random Blocks".

The factors and their levels appear in Table 4.1. Note that all of these factors are categorical. The factors and their levels are also given in the factor table Wine Factors.jmp in the Design Experiment folder of Sample Data.

To experiment with all possible combinations of these factors would require a staggering $4 \times 2^8 = 1024$ runs. However, in this example, you are able to construct a compelling design in only 40 runs.

Table 4.1 Process Factors and Levels for Wine Tasting Experiment

Factor	Levels
Variety	Bernard, Dijon
Field	1, 2, 3, 4
De-Stem	No, Yes
Yeast	Cultured, Wild
Temperature	High, Low
Press	Hard, Soft
Barrel Age	New, 2 Years
Barrel Seasoning	Air, Kiln
Filtering	No, Yes

Now that the experimental goals, factors, and responses have been defined, you can build your custom design.

Add Responses

For this custom design example add your response, the response Goal, and, if appropriate, the Lower Limit, Upper Limit, and Importance. Here, the response is Rating.

1. Select **DOE > Custom Design**.
2. Double-click Y under Response Name and type Rating.

   Note that the default Goal is Maximize. Because you want to maximize the taste rating, do not change the goal.
3. Click under Lower Limit and type 0.

   The least desirable rating is 0.
4. Click under Upper Limit and type 20.

   The most desirable rating is 20.
5. Leave the area under Importance blank.

   Because there is only one response, that response is given Importance 1 by default.

Figure 4.2 shows the completed Responses outline.

Add Factors Manually or Automatically

For this custom design example, enter factors either manually or automatically using a pre-existing table that contains the factors and settings.

- If you are designing a new experiment, you must first enter the factors manually. See "Add Factors Manually".
- Once you have saved the factors using the Save Factors option, you can load them automatically using the saved table. See "Add Factors Automatically Using Load Factors".

Both methods add these four outlines to the Custom Design window: Define Factor Constraints, Model, Alias Terms, and Design Generation.

Add Factors Manually

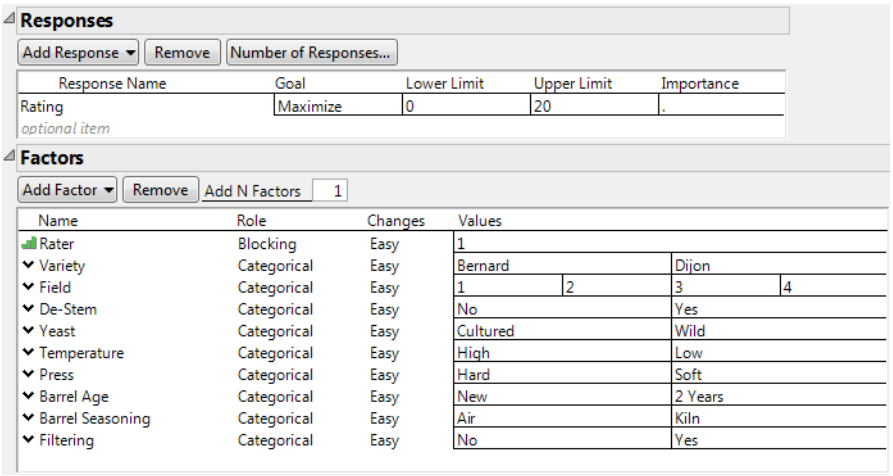
1. First, add the blocking factor, Rater. Click **Add Factor > Blocking > 8 runs per block**.
2. Type Rater over the default Name of X1.

   Note that Role is set to Blocking. Note also that only one setting for Values appears. This is because the number of blocks cannot be determined until the desired number of runs is specified. Once you specify the Number of Runs in the Design Generation outline, the number of levels for Rater updates to what is required.
3. Click **Add Factor > Categorical > 2 Level**.
4. Type Variety over the default Name of X2.

   Note that Role is set to Categorical, as requested, and that Changes is set to Easy by default.
5. Click L1 and L2 and change them to Bernard and Dijon.
6. Click **Add Factor > Categorical > 4 Level**.
7. Type Field over the default Name of X3.
8. Click L1, L2, L3, and L4, and change them to 1, 2, 3, and 4.
9. Click **Add Factor > Categorical > 2 Level**.
10. Type De-Stem over the default Name of X4.
11. Click L1 and L2 and change them to No and Yes.
12. Type 6 next to **Add N Factors**, and then click **Add Factor > Categorical > 2 Level**. This adds six categorical two-level factors to your design.
13. Change the default factor names and values:
    - Yeast (Cultured and Wild)

- Temperature (High and Low)
- Press (Hard and Soft)
- Barrel Age (New and Two Years)
- Barrel Seasoning (Air and Kiln)
- Filtering (No and Yes)

Figure 4.2 Completed Responses and Factors Outlines



14. Click **Continue**.

The following outlines are added to the Custom Design window:

- Define Factor Constraints (not used in this example)
- Model
- Alias Terms
- Design Generation

Add Factors Automatically Using Load Factors

Enter factors using a table containing factor information:

1. Select **Help > Sample Data Library** and open Design Experiment/Wine Factors.jmp.
2. Click the Custom Design red triangle and select **Load Factors**.

After loading the factors, the Custom Design window automatically updates. The following outlines are added to the Custom Design window:

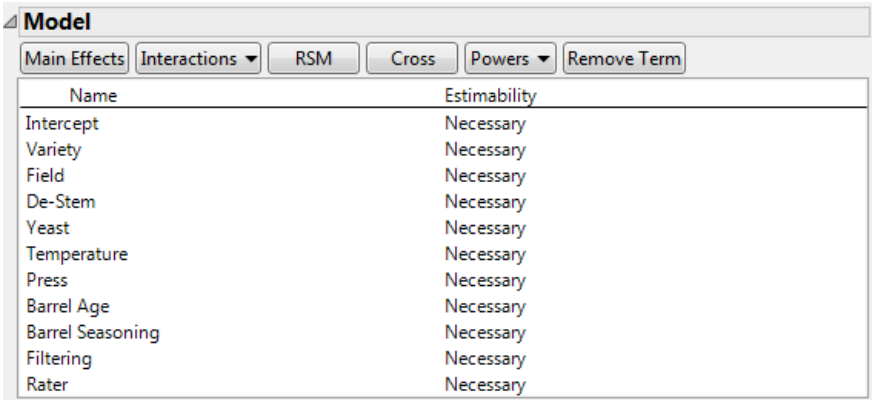
- Define Factor Constraints (not used in this example)
- Model

- Alias Terms
- Design Generation

Define the Model

For this custom design example, the Model outline shows all main effects as Necessary, indicating that the design needs to be capable of estimating all main effects. Your assumed model reflects your interest in main effects only. However, if you wanted to estimate other effects, you could add them to the Model outline. For more information about models, see "Model".

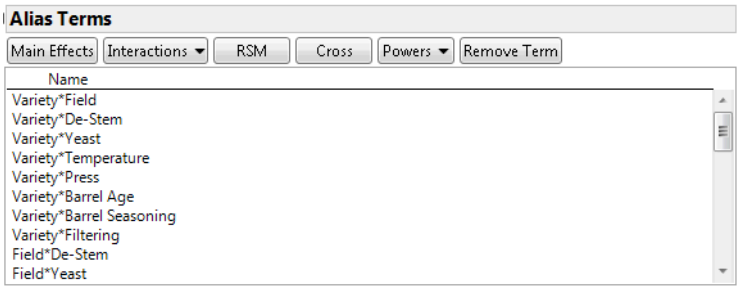
Figure 4.3 Model Outline



Define Alias Terms

The Alias Terms outline specifies the effects to be shown in the Alias Matrix, which appears later. The Alias Matrix shows the aliasing relationships between the Model terms and the effects listed in the Alias Terms outline. Open the Alias Terms outline node to verify that all two-factor interactions are listed. For more information about the alias matrix, see "Alias Matrix".

Figure 4.4 Partial View of the Alias Terms Outline



Set Random Seed to Duplicate Design

The Custom Design algorithm begins with a random starting design. To duplicate a design such as in this example, or in a teaching setting, set the random seed used to define the staring design. If you want to obtain a design with exactly the same runs and run order as the one shown in Figure 4.5, set the random seed and number of starts:

1. Click the Custom Design red triangle and select **Set Random Seed**.
2. Type 100526291 (the random seed).
3. Click **OK**.
4. Click the Custom Design red triangle and select **Number of Starts**.
5. Type 2.
6. Click **OK**.

Note: Setting the Random Seed and Number of Starts reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

Generate the Design

In the Design Generation outline, you can enter additional details about the structure and size of your design. In this example, the Default design shows 16 runs. But you have five raters, each of whom can sample eight wines. This means that you want a design with 40 runs. Change the number of design runs:

1. Under **Number of Runs**, type **40** in the **User Specified** box.

   Because you do not want to replicate runs, leave the **Number of Replicate Runs** set to **0**.
2. Click **Make Design**.

The Design and Design Evaluation outlines are added to the Custom Design window. The Output Options panel also appears.

Verify the Design

The Design outline shows the runs in the custom design that you have constructed. Later, you are able to randomize the order under Output Options. For now, verify that this design is appropriate for your experiment. For example, check that each of five Raters evaluates eight wines, that all necessary factors are shown, and that none of the settings represent infeasible combinations.

Figure 4.5 Design for Wine Experiment

Design

Run	Rater	Variety	Field	De-Stem	Yeast	Temperature	Press	Barrel Age	Barrel Seasoning	Filtering
1	1	Dijon	3	No	Wild	High	Soft	New	Kiln	Yes
2	2	Bernard	4	Yes	Cultured	Low	Hard	New	Kiln	No
3	3	Bernard	2	Yes	Wild	High	Hard	New	Air	No
4	4	Bernard	1	Yes	Wild	Low	Hard	2 Years	Air	Yes
5	5	Dijon	1	Yes	Wild	Low	Soft	New	Kiln	No
6	1	Dijon	2	Yes	Cultured	Low	Soft	2 Years	Air	Yes
7	2	Dijon	1	No	Cultured	High	Hard	New	Air	No
8	3	Dijon	4	No	Cultured	High	Soft	New	Air	Yes
9	4	Dijon	4	Yes	Wild	High	Hard	New	Kiln	Yes
10	5	Bernard	2	No	Wild	Low	Hard	New	Air	Yes
11	1	Bernard	3	Yes	Wild	High	Hard	New	Air	No
12	2	Dijon	1	Yes	Wild	High	Soft	2 Years	Air	Yes
13	3	Bernard	4	Yes	Wild	Low	Soft	2 Years	Kiln	No
14	4	Bernard	3	No	Cultured	Low	Soft	New	Air	No
15	5	Dijon	2	No	Cultured	High	Soft	2 Years	Kiln	No
16	1	Dijon	1	Yes	Cultured	Low	Hard	2 Years	Kiln	Yes
17	2	Bernard	2	No	Cultured	Low	Hard	2 Years	Air	No
18	3	Dijon	2	Yes	Cultured	High	Hard	New	Kiln	No
19	4	Bernard	1	No	Cultured	Low	Hard	New	Kiln	Yes
20	5	Bernard	3	Yes	Cultured	High	Hard	2 Years	Air	Yes
21	1	Bernard	4	No	Wild	High	Hard	2 Years	Kiln	No
22	2	Bernard	4	Yes	Cultured	High	Soft	2 Years	Kiln	Yes
23	3	Dijon	1	No	Wild	Low	Hard	2 Years	Kiln	No
24	4	Dijon	2	Yes	Wild	High	Soft	2 Years	Air	No
25	5	Bernard	1	No	Cultured	High	Soft	New	Kiln	No
26	1	Bernard	2	No	Wild	Low	Soft	New	Kiln	Yes
27	2	Dijon	3	No	Wild	High	Hard	New	Kiln	Yes
28	3	Bernard	1	No	Wild	High	Soft	2 Years	Air	Yes
29	4	Bernard	3	Yes	Cultured	High	Soft	2 Years	Kiln	No
30	5	Dijon	3	Yes	Wild	Low	Hard	2 Years	Kiln	No
31	1	Dijon	4	No	Cultured	Low	Hard	2 Years	Air	No
32	2	Bernard	2	Yes	Wild	Low	Soft	New	Kiln	Yes
33	3	Bernard	3	No	Cultured	Low	Soft	2 Years	Kiln	Yes
34	4	Dijon	2	No	Cultured	High	Hard	2 Years	Kiln	Yes
35	5	Dijon	4	Yes	Cultured	Low	Soft	New	Air	Yes
36	1	Bernard	1	Yes	Cultured	High	Soft	New	Air	No
37	2	Dijon	3	No	Wild	Low	Soft	2 Years	Air	No
38	3	Dijon	3	Yes	Cultured	Low	Hard	New	Air	Yes
39	4	Dijon	4	No	Wild	Low	Soft	New	Air	No
40	5	Bernard	4	No	Wild	High	Hard	2 Years	Air	Yes

Evaluate the Design

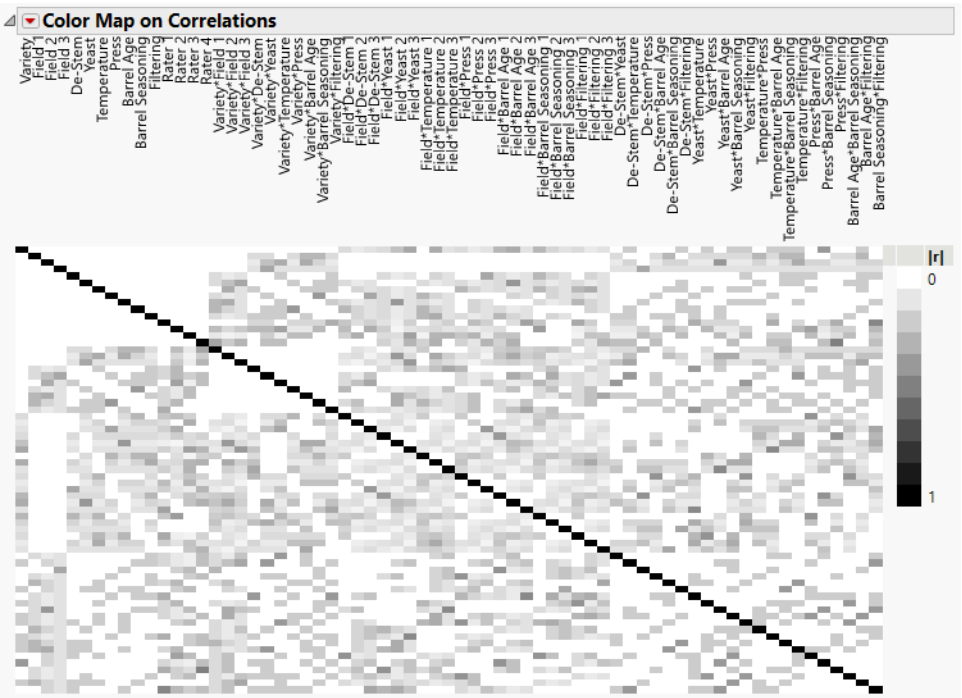
The Design Evaluation outline provides different ways to evaluate your custom design. For this example, open the Design Evaluation outline, and examine the Color Map on Correlations, the Alias Matrix, and Design Diagnostics.

Note: For more information about the Design Evaluation outline, see "Evaluate Designs".

Color Map on Correlations

The Color Map on Correlations shows the absolute value of the correlation between any two effects that appear in either the Model or the Alias Terms outline.

Figure 4.6 Color Map on Correlations



The main effects are represented by the 15 terms in the upper left corner of the map. The white corresponding to the correlations of main effects with other main effects indicate correlations of 0. This means that all main effects are orthogonal and can be estimated independently.

The only black in Figure 4.6 is on the main diagonal. Black indicates absolute correlations of one, reflecting that each term is perfectly correlated with itself. It follows that no main effect is completely confounded with any two-way interaction. In fact, the absolute values of the correlations of main effects with two-way interactions are fairly low. This means that estimates of main effects might be only slightly biased by the presence of active two-way interactions.

Tip: Hover over cells in the color map to see the absolute correlations between effects. Right-click below the legend to save the correlations to a data table.

Alias Matrix

In the Alias Matrix, model effects are listed in the column on the left. For a given model effect, a column entry indicates the degree to which the column effect (if active) biases the estimate of the model effect.

Figure 4.7 Partial View of Alias Matrix

Alias Matrix

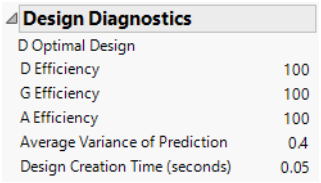
Effect	Variety*Field 1	Variety*Field 2	Variety*Field 3	Variety*De-Stem	Variety*Yeast	Variety*Temperature	Variety*Press	Variety*Barrel Age
Intercept	0	0	0	0	0	0	0	0
Variety	0	0	0	0	0	0	0	0
Field 1	0	0	0	0.141	0.141	0	-0.14	0.283
Field 2	0	0	0	0.082	-0.24	-0.33	0.082	0.327
Field 3	0	0	0	-0.12	0.346	0.115	-0.12	-0.12
De-Stem	0.141	0.082	-0.12	0	0	-0.2	0	0
Yeast	0.141	-0.24	0.346	0	0	0	-0.2	0
Temperature	0	-0.33	0.115	-0.2	0	0	0	-0.2
Press	-0.14	0.082	-0.12	0	-0.2	0	0	0
Barrel Age	0.283	0.327	-0.12	0	0	-0.2	0	0
Barrel Seasoning	0.283	0.163	0.115	0	-0.2	0.2	0.4	0
Filtering	-0.14	-0.08	0.115	-0.2	0.2	0.2	0	0.2
Rater 1	0	0	0	-0.16	-0.16	0	-0.16	0.158
Rater 2	-0.26	0.149	-0.21	-0.27	0.274	-0.37	-0.09	-0.09
Rater 3	0.091	-0.05	0.075	0	-0.19	-0.06	-0.26	-0.26
Rater 4	0.354	-0.2	0.289	0	0.25	-0.25	0	0

For example, consider the model effect Barrel Seasoning. If Variety*Press is active, then the expected value of the estimate for the Barrel Seasoning effect differs from an unbiased estimate of that effect. The amount by which it differs is equal to 0.4 times the effect of Variety*Press. Therefore, what appears to be a significant Barrel Seasoning estimated effect could in reality be a significant Variety*Press effect.

Design Diagnostics

The Design Diagnostics outline provides information about the efficiency of the design. Efficiency measures compare your design to a theoretically optimal design, which might not exist. The efficiency values are ratios, expressed as percents, of the efficiency of your design to the efficiency of this optimal design. For more information about the efficiency measures, see "Estimation Efficiency".

Figure 4.8 Design Diagnostics Outline



Notice that the D-, G-, and A-efficiency values are all 100%. Because your design is orthogonal for main effects, the design is optimal for the main effects model relative to all three efficiency criteria.

The first line in the Design Diagnostics outline indicates that your design was constructed to optimize the D-efficiency criterion. See the Optimality Criterion description in "Custom Design Options". In this case, your design has D Efficiency of 100%.

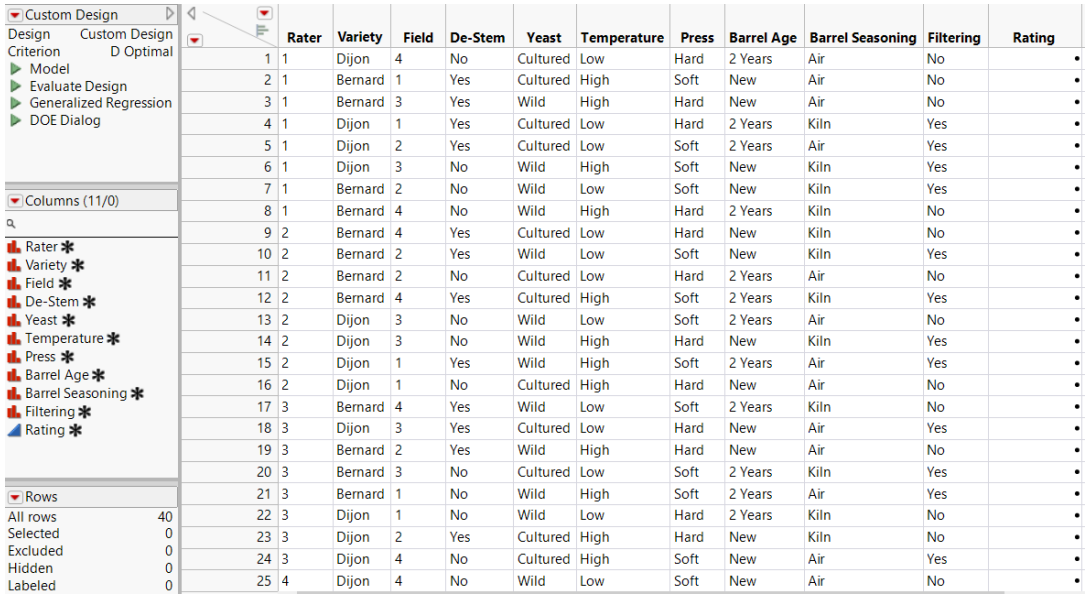
Specify Design Table Options

The final steps of your custom design generation are to specify options for the design table and to make the table. Specify the order of runs in your data table using the Output Options panel. The default selection, **Randomize within Blocks**, is appropriate for this example. Simply click **Make Table**.

A Custom Design table is created and opens, similar to the one in Figure 4.9.

Note: Your table might look different because the algorithm that creates it uses a random starting design. To obtain the precise table shown in Figure 4.9, set the random seed and number of starts as described in "Set Random Seed to Duplicate Design".

Figure 4.9 Custom Design Table



	Rater	Variety	Field	De-Stem	Yeast	Temperature	Press	Barrel Age	Barrel Seasoning	Filtering	Rating
1	1	Dijon	4	No	Cultured	Low	Hard	2 Years	Air	No	•
2	1	Bernard	1	Yes	Cultured	High	Soft	New	Air	No	•
3	1	Bernard	3	Yes	Wild	High	Hard	New	Air	No	•
4	1	Dijon	1	Yes	Cultured	Low	Hard	2 Years	Kiln	Yes	•
5	1	Dijon	2	Yes	Cultured	Low	Soft	2 Years	Air	Yes	•
6	1	Dijon	3	No	Wild	High	Soft	New	Kiln	Yes	•
7	1	Bernard	2	No	Wild	Low	Soft	New	Kiln	Yes	•
8	1	Bernard	4	No	Wild	High	Hard	2 Years	Kiln	No	•
9	2	Bernard	4	Yes	Cultured	Low	Hard	New	Kiln	No	•
10	2	Bernard	2	Yes	Wild	Low	Soft	New	Kiln	Yes	•
11	2	Bernard	2	No	Cultured	Low	Hard	2 Years	Air	No	•
12	2	Bernard	4	Yes	Cultured	High	Soft	2 Years	Kiln	Yes	•
13	2	Dijon	3	No	Wild	Low	Soft	2 Years	Air	No	•
14	2	Dijon	3	No	Wild	High	Hard	New	Kiln	Yes	•
15	2	Dijon	1	Yes	Wild	High	Soft	2 Years	Air	Yes	•
16	2	Dijon	1	No	Cultured	High	Hard	New	Air	No	•
17	3	Bernard	4	Yes	Wild	Low	Soft	2 Years	Kiln	No	•
18	3	Dijon	3	Yes	Cultured	Low	Hard	New	Air	Yes	•
19	3	Bernard	2	Yes	Wild	High	Hard	New	Air	No	•
20	3	Bernard	3	No	Cultured	Low	Soft	2 Years	Kiln	Yes	•
21	3	Bernard	1	No	Wild	High	Soft	2 Years	Air	Yes	•
22	3	Dijon	1	No	Wild	Low	Hard	2 Years	Kiln	No	•
23	3	Dijon	2	Yes	Cultured	High	Hard	New	Kiln	No	•
24	3	Dijon	4	No	Cultured	High	Soft	New	Air	Yes	•
25	4	Dijon	4	No	Wild	Low	Soft	New	Air	No	•

Note the following:

- In the Table panel, the Model, Evaluate Design, and DOE Dialog scripts are added during the design creation process. The Model script opens a Fit Model window containing the effects that you specified as Necessary in the Custom Design dialog window. The DOE Dialog script re-creates the window used to generate the design table.
- In the Columns panel, the asterisks to the right of the factors and response indicate column properties that have been saved to the columns in the data table. These column properties are used in the analysis of the data. For more information about column properties, see "Factors" and "Factor Column Properties".

Analyze the Custom Design

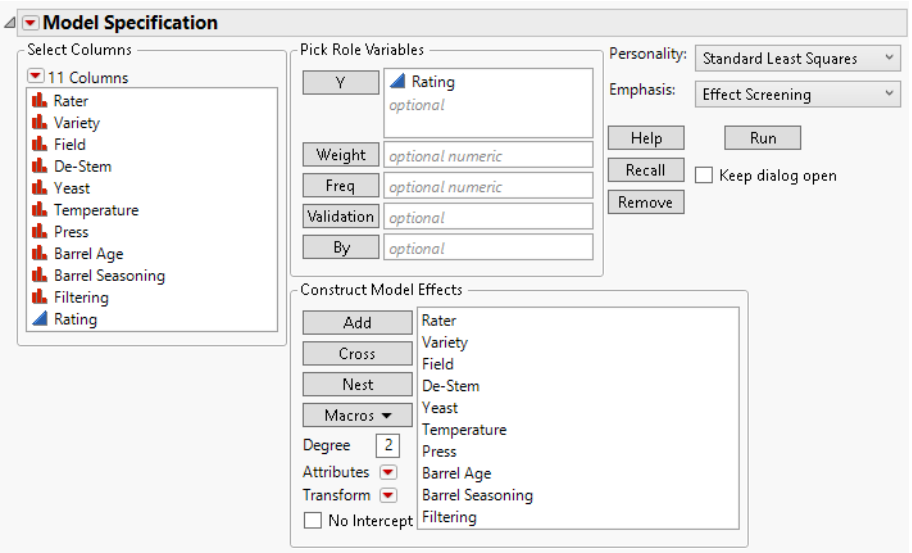
After running an experiment and compiling the data you are ready to analyze the results. Use the Rating column of your Custom Design table to record the Rating data. .

1. Select **Help > Sample Data Library** and open Design Experiment/Wine Data.jmp.

   The Wine Data.jmp table is exactly the same as the Custom Design table shown in Figure 4.9, except that it contains your recorded experimental results.

2. In the Table panel, click the green triangle next to the **Model** script.

Figure 4.10 Model Specification Window for the Wine Experiment



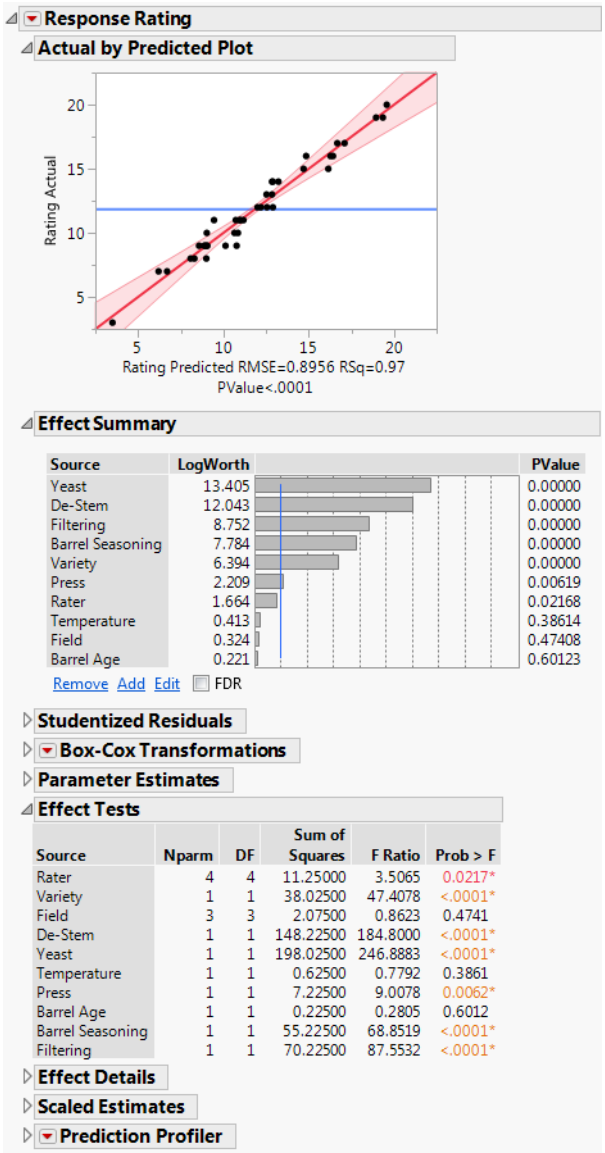
Notice that Rater, the blocking factor, is added as a fixed effect, rather than as a random block effect. This is appropriate because the five raters were specifically chosen and are not a random sample from a larger population.

3. Click **Run**.

Interpret the Full Model Results

The model output from our custom design example is shown below.

Figure 4.11 Partial Model Fit Results



Note the following:

- The Actual by Predicted Plot shows no obvious evidence of lack of fit.
- The model is significant, as indicated by the Actual by Predicted Plot and by the *p*-value beneath it.
- The Effect Tests report indicates that seven of the model terms are significant at the 0.05 level. Field, Temperature, and Barrel Age are not significant.

- The Effect Summary report lists these effects in decreasing order of significance. Larger LogWorth values correspond to smaller PValues and greater significance.

Reduce the Model

Reduce the model for the custom design results by removing the effects that you identified as inactive:

1. In the Effect Summary report, press Control and select Temperature, Field, and Barrel Age.
2. Click **Remove**.

   The report updates to show the model fit with these three effects removed.

Interpret the Model Results with the Profiler

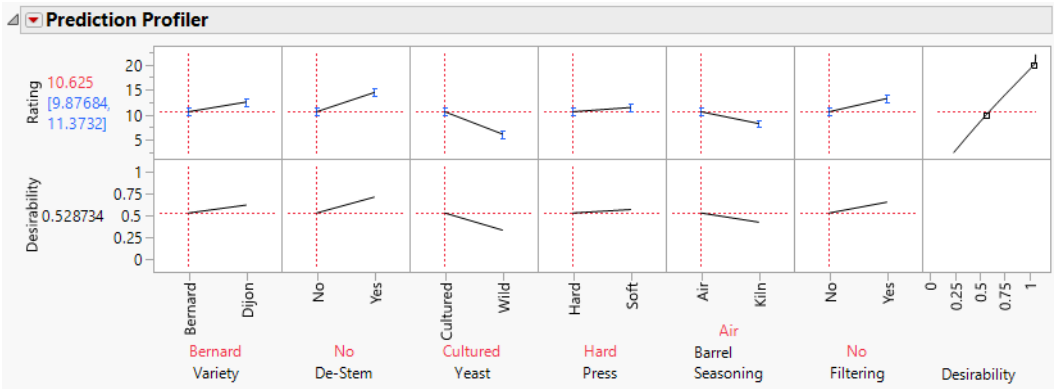
The Actual by Predicted Plot for the reduced model shows no lack of fit issues. The Effect Summary and the Effect Test report show that the remaining seven terms are significant at the 0.05 level. Use the prediction profiler to further explore your reduced model.

Figure 4.12 shows the Prediction Profiler. Recall that you specified a response goal of Maximize, with lower and upper limits of 0 and 20. Setting these limits caused a Response Limits column property to be saved to the Rating column in the Custom Design table. The Prediction Profiler uses the Response Limits information to construct a Desirability function, which appears in the right-most plot in the top row in Figure 4.12. The bottom row displays Desirability traces.

The first six plots in the top row show traces of the predicted model. For each factor, the line in the plot shows how Rating varies when all other factors are set at the values defined by the red dashed vertical lines. By default, the profiler appears with categorical factors set at their low settings. By varying the settings for the factors, you can see how the predicted Rating for wines changes. Notice that a confidence interval is given for the mean predicted Rating.

Observe that Rater is not included among the factors shown in the profiler. This is because Rater is a block variable. You included Rater to explain variation, but Rater is not of direct interest in terms of optimizing process factor settings. The predicted Rating for a wine with the given settings is the average of the predicted ratings for that wine by all raters.

Figure 4.12 Profiler for Reduced Model



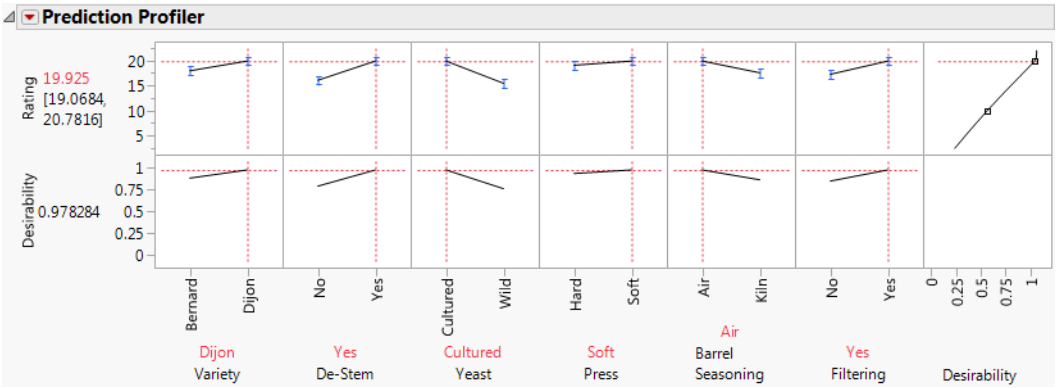
Optimize Factor Settings

Use the prediction profiler to identify optimal settings based on your custom design results. You would like to identify settings that maximize Rating across raters.

1. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

   The red dashed vertical lines in the Prediction Profiler update to show optimal settings for each factor. The optimal settings result in a predicted rating of 19.925. In general, there can be different sets of factor settings that result in the same optimal value.

Figure 4.13 Prediction Profiler with Factor Settings Optimized



2. To see predicted ratings for all runs, save the Prediction Formula. Click the Response Rating red triangle and select **Save Columns > Prediction Formula**.

   A column called Pred Formula Rating is added to the data table. Note that one of the runs, row 33, was given the maximum rating of 20 by Rater 5. The predicted rating for that run by Rater 5 is 19.550. But the row 33 trial was run at the optimal settings. The predicted

value of 19.925 given for these settings in the Prediction Profiler is obtained by averaging the predicted ratings for that run over all five raters.

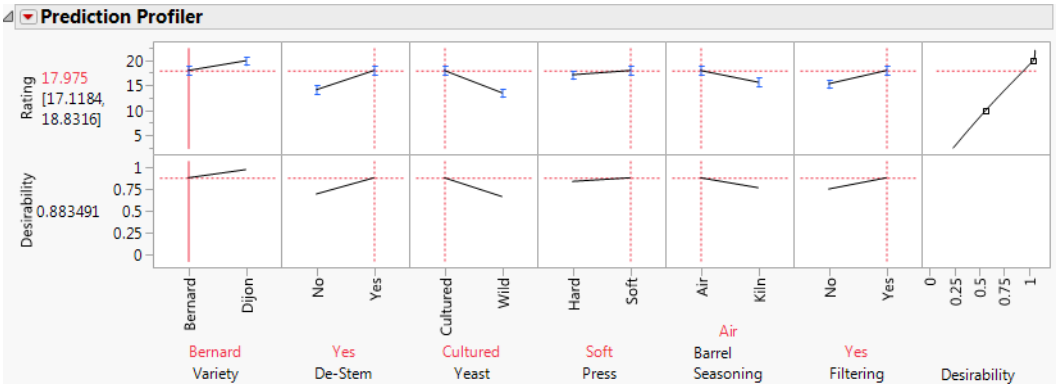
Lock a Factor Level

You can lock factors in the profiler to explore optimization under specific conditions. When you maximized your design response , you learned that the optimal rating is achieved with the Dijon variety of grapes (Figure 4.13). Your manager points out that it would be cost-prohibitive to replant the fields that are growing Bernard grapes with young Dijon vines. Therefore, you need to find optimal process settings and the predicted rating for Bernard grapes.

1. In the Variety plot of the Prediction Profiler, drag the red dashed vertical line to Bernard.
2. Press Control and click in one of the Variety plots.

   The Factor Settings window appears.
3. Select **Lock Factor Setting** and click **OK**.
4. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 4.14 Prediction Profiler with Optimal Settings for Bernard Variety



The optimal settings are unchanged because the model contains no interaction terms. The predicted rating at these settings is 17.975.

Add the Rater to the Profiler

If you want to see the Profiler traces for the levels of Rater, perform the following steps:

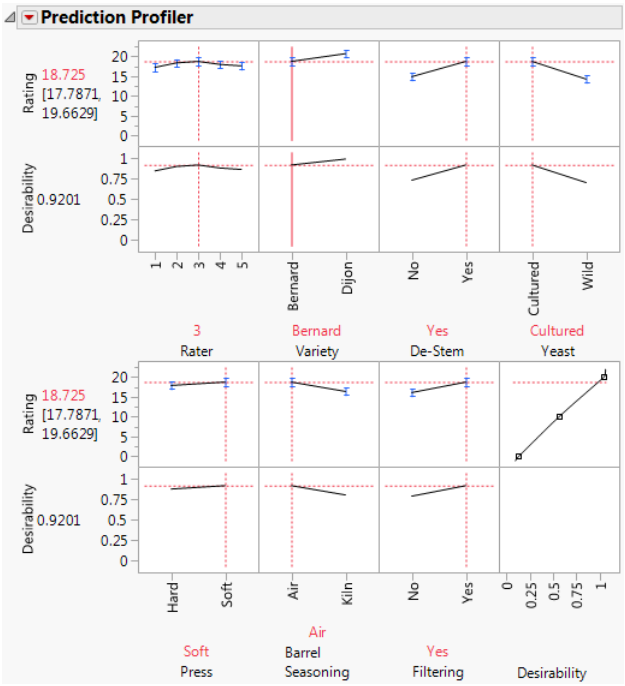
1. Click the Prediction Profiler red triangle and select **Reset Factor Grid**.

A Factor Settings window appears with columns for all of the factors, including **Rater**. The box under Rater and next to **Show** is not checked. This indicates that Rater is not shown in the Prediction Profiler.

2. Check the box under Rater in the row corresponding to **Show**.
3. Deselect the box under Rater in the row corresponding to **Lock Factor Setting**.
4. Click **OK**.

   The Profiler updates to show a plot for Rater.
5. Click in either plot above Rater.

Figure 4.15 Profiler for Reduced Model Showing Rater



A dashed vertical red line appears. Drag this line to see the traces for each of the raters. Keep in mind that Variety is still locked at Bernard. To unlock Variety, press Control and click in one of the Variety plots. In the Factor Settings window that appears, deselect **Lock Factor Setting**.

Summary of Custom Design Example

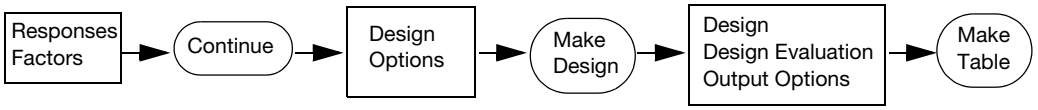
In your wine tasting experiment, using only 40 runs, you have identified six (out of nine) factors that have an effect on ratings for Pinot Noir grapes. You found that you could achieve a predicted rating of 19.925 (out of a possible 20) at the optimal settings for those factors. You also identified optimal settings for both varieties of grapes.

In this section, you constructed a design using the outlines in the Custom Design window. The next section explains each outline and the design steps in more detail.

Build a Custom Design

Build a Custom Design by selecting **DOE > Custom Design**. First define your responses and factors. Next, continue to the design options, generation, and evaluation. The design process follows the flow in Figure 4.16.

Figure 4.16 Custom Design Flow

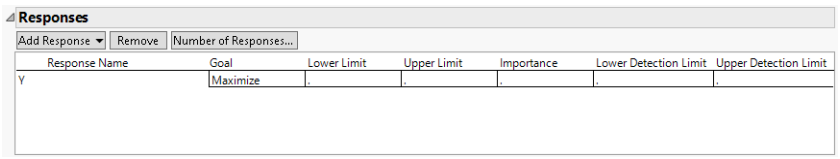


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 4.17 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.
- A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for

the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

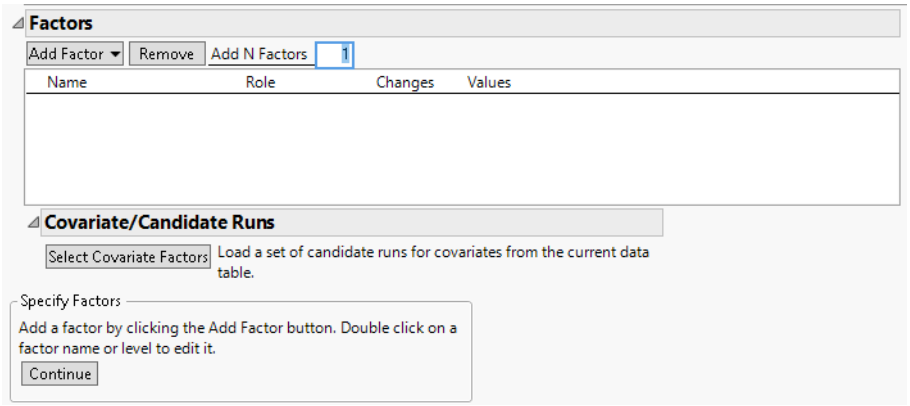
If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

Factors

Add factors to study in a custom design in the Factors outline.

Tip: When you have completed the Factors outline, consider selecting **Save Factors** from the red triangle menu. This saves the factor names, roles, changes, and values in a data table that you can later reload.

Figure 4.18 Factors Outline



Add Factor Select the factor type. See "Factor Types".

Remove Removes the selected factors.

> **Note:** If you attempt to remove all factors after clicking the Continue or Back button, one continuous factor remains. You can delete it after you add new factors.

Add N Factors Adds multiple factors. Enter the number of factors to add, click **Add Factor,** and then select the factor type. Repeat **Add N Factors** to add multiple factors of different types.

Factors Outline

The Factors outline contains the following columns:

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role Specifies the Design Role of the factor. The Design Role column property for the factor is saved to the data table. This property ensures that the factor type is modeled appropriately.

Changes Indicates whether the factor levels are Easy, Hard, or Very Hard to change. Click the default value of Easy to change it. When you specify factors as Hard or Very Hard to change, your design reflects these restrictions on randomization. A factor cannot be designated as Very Hard unless the Factors list contains a factor designated as Hard. The Factor Changes column property is saved to the data table. See "Changes and Random Blocks".

Values The experimental settings for the factors. To insert Values, click the default values and enter the desired values.

Editing the Factors Outline

In the Factors outline, note the following:

- To edit a factor name, double-click the factor name.
- Categorical factors have a down arrow to the left of the factor name. Click the arrow to add a level.
- To remove a factor level, click the value, click **Delete**, and click outside the text box.
- To modify the entry under Changes, click the value in the Changes column and select the appropriate entry.
- To edit a value, click the value in the Values column.

Factor Types

To choose a factor type, click **Add Factor** in Custom Design.

Note: A Design Role column property containing each factor's role is added to that factor's column in the design table that is generated. The Design Role column property ensures that the factor is modeled correctly.

Continuous Numeric data types only. A continuous factor is a factor that you can conceptually set to any value between the lower and upper limits you supply, given the limitations of your process and measurement system.

Discrete Numeric Numeric data types only. A discrete numeric factor can assume only a discrete number of values. These values have an implied order.

The default values for a discrete numeric factor with k levels, where $k > 2$, are the integers 1, 2, ..., k. The default values for a discrete numeric factor with $k = 2$ levels are -1 and 1. Replace the default values with the settings that you plan to use in your experiment.

Note: Not all levels of a discrete numeric factor appear in the design. The levels that appear are determined by your specifications in the Model outline. If you need all levels to appear in your design, consider using the Screening Design platform.

In the assumed model, the effects for a discrete numeric factor with k levels include polynomial terms in that effect through order $k-1$. For k greater than 6, powers up to the 5th level are included. The Estimability for polynomial effects (powers of two or higher) is set to If Possible. This allows the algorithm to use the multiple levels as permitted by the run size. If the polynomial terms are not included, then a main effects only design is

created. For more information about how discrete numeric factors are treated in the assumed model, see "Model".

Fit Model treats a discrete numeric factor as a continuous predictor. The Model script that is saved to the design table does not contain any polynomial terms of order greater than two.

Categorical Either numeric or character data types. The data type in the resulting data table is categorical. The value order of the levels is the order of the values, as entered from left to right. This ordering is saved in the Value Order column property after the design data table is created.

Blocking Either numeric or character data types. A blocking factor is a special type of categorical factor that can enter the model only as a main effect. When you define a blocking factor, you specify the number of runs per block. The RunsPerBlock column property is saved to the design table. The Default run size always assumes that there are at least two blocks. If you specify a run size that is not an integer multiple of the number of runs per block, JMP tries to balance the design to the extent possible. In balancing the design, JMP ensures that there are at least two runs per block.

Covariate Either numeric or character data types. Use the mixture column property to include mixture covariates in your design. The values of a covariate factor are measurements on experimental units that are known in advance of an experiment. If the covariates are mixture variables, they must represent all of the mixture variables in the design.

Covariate values are selected to ensure the optimality of the resulting design relative to the optimality criterion. See "Changes and Random Blocks" and "Covariates with Hard-to-Change Levels".

JMP obtains the covariate factors and their values from an open data table that contains the measured covariates for the available experimental units. Make this data table your current data table. Use **Add Factors** or click S**elect Covariate Factors** in the Covariate/Candidate Runs outline to add covariates. When you add a covariate, a list of columns in the current data table opens, and you select the columns containing covariates from this list. The addition of covariate factors loads a table with the candidate rows into the Covariate/Candidate Runs outline. Once loaded, you can select rows to force into the design. To force selected covariate rows into your design, use the Design Generation option **Include all selected covariate rows in the design**.

If the design has fewer runs than the number of covariate rows, then the design table includes a Covariate Row Index column. This column indicates the row from the covariate table that corresponds to each experimental run.

In some situations, you might want to select a small set of design points from a larger set of candidate settings. For example, you might have multiple measurement columns (factors) for a large batch of units. You want to treat the measurements for each unit as a

candidate run. From these candidate runs, you want to select a small but optimal collection for which you measure a response. In this case, make the data table of all candidate runs the active table, select Add Factor > Covariate, and enter all of your measurement columns as covariates. Specify your desired run size. The Custom Design platform identifies an optimal collection of design settings.

Note: When you have covariates, the Design Generation outline contains two options. The first option allows you to specify that selected rows in your covariate table are included in the design. The second option allows covariate rows to be repeated in the design.

Mixture Continuous factors that represent ingredients in a mixture. The values for a mixture factor must sum to a constant. By default, the values for all mixture factors sum to one. To set the sum of the mixture components to some other positive value, select **Advanced Options > Mixture Sum** from the red triangle menu. The Mixture column property is saved to the data table.

Constant Either numeric or character data types. A constant factor is a factor whose values are fixed during an experiment. Constant factors are not included in the Model outline or in the Model script that is saved to the data table.

Uncontrolled Either numeric or character data types. An uncontrolled factor is one whose values cannot be controlled during production, but it is a factor that you want to include in the model. It is assumed that you can record the factor's value for each experimental run.

An empty column with a Continuous Modeling Type is created in the design table. You can change the column's Data Type and Modeling Type in the Column Info window if required. Enter your data in this column. Uncontrolled factors are included in the Model outline and the Model script that is saved to the data table.

Changes and Random Blocks

Specifying the relative difficulty of changing a factor from run to run is useful in industrial experimentation. It is often convenient to make several runs while keeping factors that are hard-to-change fixed at some setting. A Changes value of Hard results in a split-plot design. A Changes value of Very Hard results in a split-split-plot design or a two-way split-plot design.

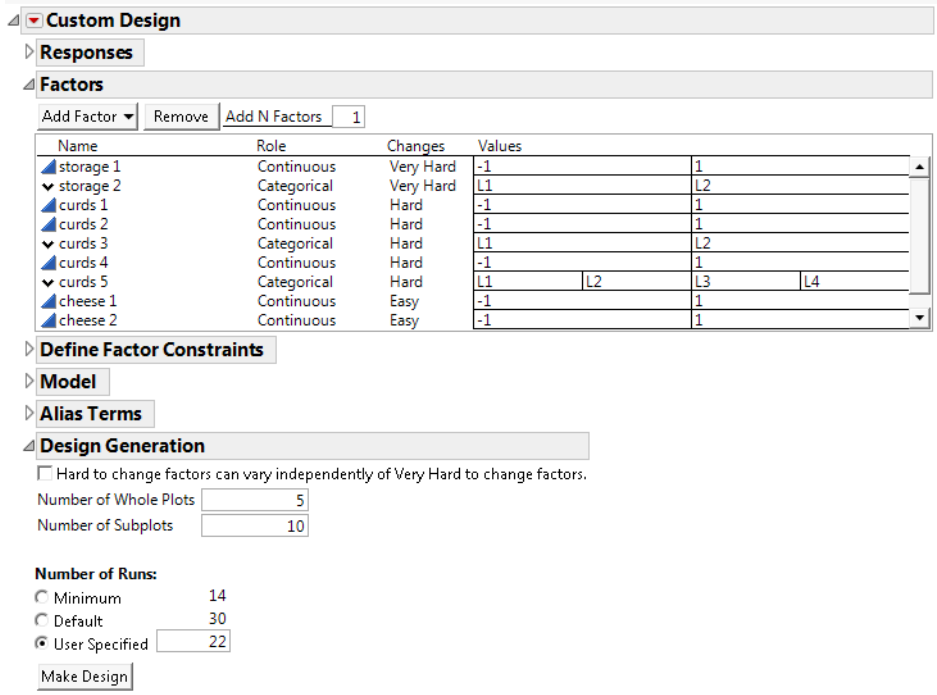
You can set Changes for Continuous, Discrete Numeric, Categorical, and Mixture factors to Hard and Very Hard. To set a factor to Very Hard, the list must contain another factor that is set to Hard.

You can set Changes for a Covariate factor to Hard. In this case, all other covariates are also set to Hard and the remaining factors are set to Easy. The algorithm requires a combination of row exchange and coordinate exchange. For this reason, even moderately sized designs might take some time to generate.

For designs with Hard or Very Hard to change factors, Custom Design strives to find a design that is optimal, given your specified optimality criterion. See "Optimality Criteria". For more information about the methodology used to generate split-plot designs, see Jones and Goos (2007). For more information about designs with hard-to-change covariates, see Jones and Goos (2015).

Figure 4.19 shows a split-split-plot scenario, using the factors from the Cheese Factors.jmp sample data table (located in the Design Experiment folder).

Figure 4.19 Factors and Design Generation Outline for a Split-Split-Plot Design



If you assign Changes as Hard for one or more factors, but no factors are assigned Changes that are Very Hard, a categorical factor called Whole Plots is added to the design. This situation results in a split-plot design:

- Each level of Whole Plots corresponds to a block of constant settings of the hard-to-change factors.
- The Model script in the design table applies the Random Effect attribute to the factor Whole Plots.
- The factor Whole Plots is assigned the Design Role column property with a value of Random Block.

When you designate Changes as both Hard and Very Hard, categorical factors called Subplots and Whole Plots are added to the design. This situation results in a split-split-plot design:

- Each level of Subplots corresponds to a block of constant settings of the hard-to-change factors.
- Each level of Whole Plots corresponds to a block of constant settings of the very-hard-to-change factors.
- The Model script in the design table applies the Random Effect attribute to the Whole Plots and Subplots effects.
- The levels of the hard-to-change factor are assumed to be nested within the levels of the very-hard-to-change factor by default.
- In the design table, both of the factors Whole Plots and Subplots are assigned the Design Role column property with a value of Random Block.

To construct a two-way split-plot design, select the **Hard to change factors can vary independently of Very Hard to change factors** option under Design Generation. The option crosses the levels of the hard-to-change factor with the levels of the very-hard-to-change factor. See "Two-Way Split-Plot Designs".

Use the Number of Whole Plots and Number of Subplots text boxes to specify values for the numbers of whole plots or subplots. These boxes are initialized to suggested numbers of whole plots and subplots. For information about how these values are obtained, see "Numbers of Whole Plots and Subplots".

For more information and scenarios that illustrate random block split-plot, split-split-plot, and two-way split-plot designs, see "Designs with Randomization Restrictions". For more information about designs with hard-to-change covariates, see "Covariates with Hard-to-Change Levels".

Factor Column Properties

For each factor, various column properties are saved to the data table. You can find more information about these column properties and related examples in "Column Properties".

Design Role Each factor is given the Design Role column property. The Role that you specify in defining the factor determines the value of its Design Role column property. When you add a random block under Design Generation, that factor is assigned the Random Block value. The Design Role property reflects how the factor is intended to be used in modeling the experimental data. Design Role values are used in the Augment Design platform. See "Design Role".

Factor Changes Each factor is assigned the Factor Changes column property. The value that you specify under Changes determines the value of its Factor Changes column property. The Factor Changes property reflects how the factor is used in modeling the experimental

data. Factor Changes values are used in the Augment Design and Evaluate Design platforms. See "Factor Changes".

Coding If the Role is Continuous, Discrete Numeric, a continuous Covariate, or Uncontrolled, the Coding column property for the factor is saved. This property transforms the factor values so that the low and high values correspond to –1 and +1, respectively. See "Coding".

Value Order If the Role is Categorical or Blocking, the Value Order column property for the factor is saved. This property determines the order in which levels of the factor appear. See "Value Order".

Mixture If the Role is Mixture, the Mixture column property for the factor is saved. This property indicates the limits for the factor and the mixture sum. It also enables you to choose the coding for the mixture factors. See "Mixture".

RunsPerBlock For a blocking factor, indicates the maximum allowable number of runs in each block. When a Blocking factor is specified in the Factors outline, the RunsPerBlock column property is saved for that factor. See "RunsPerBlock".

Define Factor Constraints

Use Define Factor Constraints to restrict the design space. Unless you have loaded a constraint or included one as part of a script, the **None** option is selected. To specify constraints, select one of the other options:

Specify Linear Constraints Specifies inequality constraints on linear combinations of factors. Available only for factors with a Role of Continuous or Mixture. See "Specify Linear Constraints".

Note: When you save a script for a design that involves a linear constraint, the script expresses the linear constraint as a *less than or equal to* inequality (≤).

Use Disallowed Combinations Filter Defines sets of constraints based on restricting values of individual factors. You can define both AND and OR constraints. See "Use Disallowed Combinations Filter".

Use Disallowed Combinations Script Defines disallowed combinations and other constraints as Boolean JSL expressions in a script editor box. See "Use Disallowed Combinations Script".

Note: When you analyze a design that has factor constraints, the model profiler honors the constraints.

Specify Linear Constraints

In cases where it is impossible to vary continuous factors independently over the design space, you can specify linear inequality constraints. Linear inequalities describe factor level settings that are allowed. For an example, see "Mixture of Mixtures Design".

Click **Add** to enter one or more linear inequality constraints.

Add Adds a template for a linear expression involving all the continuous factors in your design. Enter coefficient values for the factors and select the direction of the inequality to reflect your linear constraint. Specify the constraining value in the box to the right of the inequality. To add more constraints, click **Add** again.

Note: The Add option is disabled if you have already constrained the design region by specifying a Sphere Radius.

Remove Last Constraint Removes the last constraint.

Check Constraints Checks the constraints for consistency. This option removes redundant constraints and conducts feasibility checks. A JMP alert appears if there is a problem. If constraints are equivalent to bounds on the factors, a JMP alert indicates that the bounds in the Factors outline have been updated.

Use Disallowed Combinations Filter

This option uses an adaptation of the Data Filter to facilitate specifying disallowed combinations. For detailed information, see *Using JMP*.

Select factors from the Add Filter Factors list and click **Add**. Then specify the disallowed combinations by using the slider (for continuous factors) or by selecting levels (for categorical factors). For an example, see "Response Surface Design With Constraints and Categorical Factor".

The red triangle options for the Add Filter Factors menu are those found in the Select Columns panel of many platform launch windows. See *Using JMP*.

When you click Add, the Disallowed Combinations control panel shows the selected factors and provides options for further control. Factors are represented as follows, based on their modeling types:

Continuous Factors For a continuous factor, a double-ended slider that spans the range of factor settings appears. You can specify disallowed settings by dragging the slider ends or by setting the endpoints by clicking on the text value under the slider. In the slider, a solid blue highlight represents the disallowed values.

Categorical Factor For a categorical factor, the possible levels are displayed either as labeled blocks or, when the number of levels is large, as list entries. Select a level to disallow it. To

select multiple levels, hold the Control key. The block or list entries are highlighted to indicate the levels that have been disallowed. When you add a categorical factor to the Disallowed Combinations panel, the number of levels of the categorical factor is given in parentheses following the factor name.

Disallowed Combinations Options

The control panel has the following controls:

Clear Clears all disallowed factor level settings that you have specified. This does not clear the selected factors.

Start Over Removes all selected factors and returns you to the initial list of factors.

AND Opens the Add Filter Factors list. Selected factors become an AND group. Any combination of factor levels specified within an AND group is disallowed.

To add a factor to an AND group later on, click the group's outline to see a highlighted rectangle. Select AND and add the factor.

To remove a single factor, select **Delete** from its red triangle menu.

OR Opens the Add Filter Factors list. Selected factors become a separate AND group. For AND groups separated by OR, a combination is disallowed if it is specified in at least one AND group.

Red Triangle Options for Factors

A factor can appear in several OR groups. An occurrence of the factor in a specific OR group is referred to as an *instance* of the factor.

Delete Removes the selected instance of the factor from the Disallowed Combinations panel.

Clear Selection Clears any selection for that instance of the factor.

Invert Selection Deselects the selected values and selects the values not previously selected for that instance of the factor.

Display Options Available only for categorical factors. Changes the appearance of the display. Options include showing each level as a block, list, or single category, or adding a check box next to each value.

Find Available only for categorical factors. Provides a text box beneath the factor name where you can enter a search string for levels of the factor. Press the Enter key or click outside the text box to perform the search. Once **Find** is selected, Find options appear in the red triangle menu, such as Does not contain, Match Case, Starts with or Ends with, or Clear Find.

Use Disallowed Combinations Script

Use this option to disallow particular combinations of factor levels using a JSL script. This option can be used with continuous factors or mixed continuous and categorical factors.

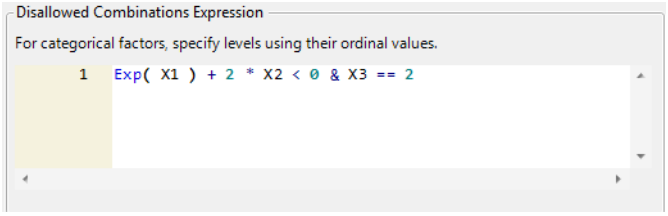
This option opens a script window where you insert a script that identifies the combinations that you want to disallow. The script must evaluate as a Boolean expression. When the expression evaluates as true, the specified combination is disallowed.

When forming the expression for a categorical factor, use the ordinal value of the level or the name of the level. If a factor's levels are high, medium, and low, specified in that order in the Factors outline, their associated ordinal values are 1, 2, and 3. For example, suppose that you have two continuous factors, X1 and X2, and a categorical factor X3 with three levels: L1, L2, and L3, in order. You want to disallow levels where the following is true:

$$e^{X_1} + 2X_2 < 0 \text{ and } X_3 = L2$$

Enter the expression `(Exp(X1) + 2*X2 < 0) & (X3 == 2)` into the script window.

Figure 4.20 Expression in Script Editor



(In the figure, unnecessary parentheses were removed by parsing.) Notice that functions can be entered as part of the Boolean expression. The expression
`(Exp(X1) + 2*X2 < 0) & (X3 == "L2")` is also valid.

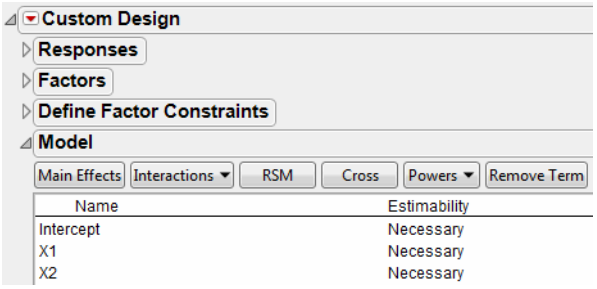
Model

Specify your assumed model (which contains all the effects that you want to estimate) in the Model outline. For each effect that you specify, you can designate that effect's Estimability. The Estimability value indicates whether it is Necessary to estimate that effect, or if you are content to estimate that effect If Possible.

The initial Model outline includes the main effects for all factors. If you have entered a discrete numeric factor with three or more levels, polynomial terms are also included in the initial model. The Estimability of second-and higher-order terms is set to If Possible. If you want to ensure that these terms are estimable, change their Estimability to Necessary.

Note: You can ensure that the estimability of discrete numeric polynomial terms is always set to Necessary. Select **File > Preferences > Platforms > DOE**. Check Discrete Numeric Powers Set to Necessary.

Figure 4.21 Model Outline



When you construct your design table, JMP saves a Model script to the data table. Except for discrete numeric factors, the Model script contains the effects shown in the Model outline. For a discrete numeric factor, the Model script contains only its main effect and quadratic term.

The Model outline contains the following buttons and fields:

Main Effects Adds main effects for all factors in the model, and polynomial terms for discrete numeric factors.

Interactions Adds interaction effects. If no factors are selected in the Factors outline, select 2nd, 3rd, 4th, or 5th to add all appropriate interactions up to that order. Add interactions up to a given order for specific factors by selecting the factor names in the Factors outline, selecting Interactions, and then specifying the appropriate order. Interactions between non-mixture and mixture factors, and interactions with blocking and constant factors, are not added.

RSM Adds interaction and quadratic terms up to the second order (response surface model terms) for all continuous factors. Categorical factors are not included in RSM terms. Main effects for non-mixture factors that interact with all the mixture factors are removed.

Cross Adds specific interaction terms. Select factor names in the Factors outline and effect names in the Model outline. Click Cross to add the crossed terms to the Model outline.

Powers Adds polynomial terms. If no factor names are selected in the Factors outline, adds polynomial terms for all continuous factors. If factor names are selected in the Factors outline, adds polynomial terms for only those factors. Select 2nd, 3rd, 4th, or 5th to add polynomial terms of that order.

Scheffé Cubic (Available for mixture factors.) Adds Scheffé cubic terms for all mixture factors. These terms are used to specify a mixture model with third-degree polynomial terms.

Remove Term Removes selected effects.

Name The name of the effect.

Estimability A designation of your need to estimate the effect.

- If Estimability is set to Necessary, the algorithm ensures that the effect is estimable.
- If Estimability is set to If Possible, the algorithm attempts to make that effect estimable, as permitted by the number of runs that you specify.

Except for polynomial terms for discrete numeric factors, all effects are specified as Necessary by default. Click an effect's Estimability value to change it.

Bayesian D-Optimality and Estimation of If Possible Effects

The Bayesian D-Optimal design approach obtains precise estimation of all Necessary terms while providing omnibus detectability (and some estimability) for If Possible terms. For more detail, see "Response Surface Experiments" and "Bayesian D-Optimality". If any main effect model term is set to If Possible, the Bayesian Information matrix is used for design diagnostic calculations.

Alias Terms

In the Alias Terms outline, add potentially active effects that are not in your assumed model but might bias the estimates of model terms. It is possible that effects *not* included in your assumed model are active. Once you generate your design, the Alias Matrix outline appears under Design Evaluation. The Alias Matrix entries represent the degree of bias imparted to model parameters by the effects that you specified in the Alias Terms outline. See the "The Alias Matrix".

By default, the Alias Terms outline includes all two-way interaction effects that are not in your Model outline (with the exception of terms involving blocking factors). Add terms using the buttons. For a description of how to use these buttons to add effects to the Alias Terms table, see "Model".

Note: For interaction terms, only interactions of the selected order are added. This differs from the Model outline where interactions of the selected order and all lower orders are added to the model.

For example, suppose that you specify a design with three continuous factors. Your assumed model, specified in the Model outline, contains only those three main effects. You can afford only six runs. You want to see how estimates of the main effects might be biased by active two-way interactions and the three-way interaction.

The Alias Terms table includes all two-way interactions by default. You can add the three-way interaction by selecting **Interactions > 3rd**.

Figure 4.22 Alias Terms Outline

Once you specify six runs in the Design Generation outline and click **Make Design**, the Design Evaluation outline appears. Open the Design Evaluation outline and the Alias Matrix outline.

Figure 4.23 Alias Matrix

Effect	X1*X2	X1*X3	X2*X3	X1*X2*X3
Intercept	0	0	0.5	-0.5
X1	0	0	-0.5	0.5
X2	0.5	-0.5	0	0
X3	-0.5	0.5	0	0

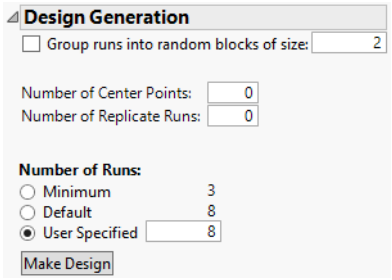
The Alias Matrix indicates that each main effect is partially aliased with two of the interactions. See "Alias Matrix" and "The Alias Matrix".

Design Generation

The Design Generation outline gives you choices relating to blocking, center points, replication, and the number of runs in your design. Typically, the input area has two parts:

- Design structure options
- Number of runs options

Figure 4.24 Design Generation Outline



Design Structure Options

Group runs into random blocks of size (Not available if a blocking factor is specified.) To construct a random block design, enter the number of runs that you want in each block. When you specify the sample size, a factor called Random Block is created. Its levels define blocks of a size that is consistent with the block size that you entered, given the specified number of runs. If the number of runs is an integer multiple of the block size, the block sizes equal your specified value.

Number of Whole Plots Appears when you specify a hard or very-hard-to-change factor. The factor Whole Plots corresponds to the very-hard-to-change factors (split-split-plot design), if there are any, otherwise to the hard-to-change factors (split-plot design). JMP suggests a value for the number of whole plots that maximizes the information about the coefficients in the model. Or, you can enter a value for the number of whole plots. See "Numbers of Whole Plots and Subplots".

Number of Subplots Appears when you specify a very-hard-to-change factor. The factor Subplots corresponds to the hard-to-change factors in the split-split-plot design. JMP suggests values for the number of whole plots and subplots that maximize the information about the coefficients in the model. Or, you can enter a value for the number of subplots. See "Numbers of Whole Plots and Subplots".

Hard to change factors can vary independently of Very Hard to change factors Select this option to create a strip-plot (also known as two-way split-plot or split block) design. This option creates a design where the hard-to-change factors are randomized within the levels of the very-hard-to-change factors. They are *not* nested within the very-hard-to-change factors.

Number of Center Points Appears only if the design contains factors with a Continuous or Mixture factor type. Specify how many additional runs you want to add as center points to

the design. A center point is a run whose setting for each continuous factor is midway between the high and low settings. See "Center Points, Replicate Runs, and Testing".

If a design contains both continuous and other types of factors, center points might not be balanced relative to the levels of the other factors. Custom Design chooses the center points to maximize the D-, I-, or alias efficiency of the design.

Number of Replicate Runs Specify the number of replicate trials that you want to add to the design. This does not replicate the entire design, but chooses the optimal design points to replicate. See "Center Points, Replicate Runs, and Testing".

Number of Runs Options

Minimum A lower bound on the number of runs necessary to avoid failures in design generation. When you select Minimum, the resulting design is saturated. There are no degrees of freedom for error.

Note: If you select the Minimum number of runs, there will be no error term for testing. You will not be able to test parameter estimates. This choice is appropriate only when the cost of additional runs is prohibitive.

Default Suggests the number of runs. This value is based on heuristics for creating a balanced design with at least four runs more than the Minimum number of runs.

User Specified Specify the number of runs that you want. Enter that value into the Number of Runs text box. This option enables you to balance the cost of additional runs against the potential gain in information.

Number of Runs This is the only option that appears when a covariate factor with Changes set to Easy is specified. The number of runs shown is the number of rows in the data table associated with your covariate or covariates. You can specify a smaller number of runs. In that case, the covariate runs that are selected optimize the design criterion.

Make Design

Once you have completed the Design Generation outline, click **Make Design**. Custom Design generates the design, presents it in the Design outline, and provides evaluation information in the Design Evaluation outline. Notice the controls in the Model and Alias Outlines are disabled. Use the Back button to change model or alias terms. The Output Options panel also appears, enabling you to create the design table.

Note: Sometimes several designs can optimize the optimality criterion. When this is the case, the design algorithm might generate different designs when you click the **Back** and **Make Design** buttons repeatedly.

Design

The Design outline shows the runs for the custom design. The design is optimal, given the conditions that you have specified. The runs might not appear to be randomized. You can select Run Order options in the Output Options panel before generating your design table.

Design Evaluation

The Design Evaluation red triangle menu and outlines provide a number of ways to evaluate the properties of the generated custom design.

Use Bayesian Information Select to use the Bayesian Information matrix in the design diagnostic calculations for designs with *If Possible* model effects that are not estimable. For more information about the Bayesian Information adjustments see "Bayesian D-Optimality".

Note: If all model effects are estimable, the Design Diagnostics are presented for all effects without making the Bayesian Information adjustment. If some terms are inestimable, only the *Necessary* model terms are presented unless the **Use Bayesian Information** option is selected.

Power Analysis Enables you to explore your ability to detect effects of given sizes.

Prediction Variance Profile Shows the prediction variance over the range of factor settings.

Fraction of Design Space Plot Shows how much of the model prediction variance lies below (or above) a given value.

Prediction Variance Surface Shows a surface plot of the prediction variance for any two continuous factors.

Estimation Efficiency For each parameter, gives the fractional increase in the length of a confidence interval compared to that of an idealized (orthogonal) design, which might not exist. Also gives the relative standard error of the parameters.

Alias Matrix Gives coefficients that indicate the degree by which the model parameters are biased by effects that are potentially active, but not in the model. You specify the terms representing potentially active effects in the Alias Terms table. See "The Alias Matrix".

Color Map on Correlations Shows the absolute correlation between effects on a plot using an intensity scale.

Note: The default intensity scale is a gray scale. To change the color scale, click the Color Map on Correlations red triangle and select **Blue to Gray to Red**. For a custom color scale, right-click in the plot and select **Color Theme**.

Design Diagnostics Indicates the optimality criterion used to construct the design. Also gives efficiency measures for your design. See Optimality Criterion in "Custom Design Options" and "Optimality Criteria".

Note: The Design Diagnostics outline does not provide the following statistics when the model includes factors with Changes set to Hard or Very Hard or with Estimability set to If Possible: D Efficiency, G Efficiency, A Efficiency.

For more information about the Design Evaluation outline, see "Design Evaluation".

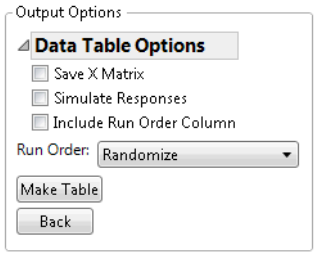
Output Options

Use the Output Options panel to save your custom design X Matrix, simulate responses, or customize your design data table. You can perform the following tasks:

- specify how you want the custom design data table to appear
- construct the design table
- return to a previous point in the Custom Design window

In most cases, the Output Options panel appears as shown in Figure 4.25.

Figure 4.25 Output Options Panel



The Output Options panel contains these options:

- "Data Table Options"
- "Run Order"
- "Make Table"
- "Back"

Data Table Options

The **Data Table Options** includes the **Save X Matrix** option ("Save X Matrix"), the **Simulate Response** option ("Simulate Responses") as well as an **Include Run Order Column** option.

Include Run Order Column Adds a column to the design data table that corresponds to the design order in the Design outline.

Tip: Use the Include Run Order Column when you save your data with a sorted run order. This provides a column of the randomized run order from the design outline.

Run Order

The **Run Order** options determine the order of the runs in the design table. Choices include the following:

Keep the Same Rows in the design table are in the same order as in the Design outline.

Sort Left to Right Columns in the design table are sorted from left to right.

Randomize Rows in the design table are in random order.

Sort Right to Left Columns in the design table are sorted from right to left.

Randomize within Blocks Rows in the design table are in random order within the blocks.

Make Table

Click **Make Table** to construct the custom design data table. In the Custom Design table, the Table panel (in the upper left) can contain scripts, as appropriate given your design. The Model, Evaluate Design, and DOE Dialog scripts are always provided. To run a script, click the green triangle next to the script name. If your design includes covariates, the design table includes a covariate row index column unless all covariate rows are used in the design.

Figure 4.26 Custom Design Table Showing Scripts

Custom Design

Design: Custom Design
Criterion: I Optimal

- Model
- Evaluate Design
- Generalized Regression
- Constraint
- DOE Dialog

	X1	X2	X3	X4	X5	X6	X7	X8	Y
1	-1	1	-1	-1	1	-1	1	1	•
2	1	-1	1	-1	-1	-1	1	-1	•
3	-1	-1	1	-1	-1	1	-1	1	•
4	1	-1	-1	1	-1	-1	-1	1	•
5	1	-1	1	1	1	1	1	1	•
6	1	1	1	-1	-1	1	-1	1	•
7	-1	1	1	1	1	-1	-1	-1	•
8	-1	1	-1	1	-1	1	1	-1	•
9	1	-1	-1	-1	1	1	-1	-1	•

Possible scripts include the following:

Model Runs the **Analyze** > **Fit Model** platform. The model described by the script is determined by your choices in the Model outline and by the type of design.

Evaluate Design Runs the **DOE > Design Diagnostics > Evaluate Design** platform. The model described by the script is determined by your choices in the Model outline and by the type of design.

Constraint Shows model constraints that you entered in the Define Factor Constraints outline using the **Specify Linear Constraints** option.

Disallowed Combinations Shows model constraints that you entered in the Define Factor Constraints outline using the **Use Disallowed Combinations Filter** or the **Use Disallowed Combinations Script** options.

DOE Dialog Re-creates the Custom Design window that you used to generate the design table. The script also contains the random seed used to generate your design.

Back

The Back button takes you back to where you were before clicking Make Design. You can make changes to the previous outlines and regenerate the design.

Note: If you attempt to remove all factors after clicking the Back button, one continuous factor remains. You can delete the continuous factor after new factors are added.

Custom Design Options

The Custom Design red triangle menu contains options for design setup and generation.

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a

new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Save X Matrix Saves the Moments Matrix and Model Matrix to table scripts in the design data table. These scripts contain the moments and design matrices. See "Save X Matrix".

Caution: For a design with nominal factors, the Model Matrix saved by the Save X Matrix option is *not* the coding matrix used in fitting the linear model. You can obtain the coding matrix used for fitting the model by selecting the option Save Columns > Save Coding Table in the Fit Model report that you obtain when you run the Model script.

Optimality Criterion Changes the design optimality criterion. The default criterion, **Recommended**, specifies D-optimality for all design types, unless you added quadratic effects using the RSM button in the Model outline. For more information about the D-, I-, and alias-optimal designs, see "Optimality Criteria".

Note: You can set a preference to always use a given optimality criterion. Select File > Preferences > Platforms > DOE. Check Optimality Criterion and select your preferred criterion.

Number of Starts Enables you to specify the number of random starts used in constructing the design. See "Number of Starts".

Design Search Time Maximum number of seconds spent searching for a design. The default search time is based on the complexity of the design. See "Design Search Time" and "Number of Starts".

If the iterations of the algorithm require more than a few seconds, a Computing Design progress window appears. If you click **Cancel** in the progress window, the calculation stops and gives the best design found at that point. The progress window also displays D-efficiency for D-optimal designs that do not include factors with Changes set to Hard or Very Hard or with Estimability set to If Possible.

Note: You can set a preference for Design Search Time. Select File > Preferences > Platforms > DOE. Check Design Search Time and enter the maximum number of seconds. In certain situations where more time is required, JMP extends the search time.

Sphere Radius Constrains the continuous factors in a design to a hypersphere. Specify the radius and click **OK**. Design points are chosen so that their distance from 0 equals the Sphere Radius. Select this option before you click Make Design.

Note: Sphere Radius constraints cannot be combined with constraints added using the Specify Linear Constraints option. Also, the option is not available when hard-to-change factors are included (split-plot designs).

Advanced Options > Mixture Sum Set the sum of the mixture factors to any positive value. Use this option to keep a component of a mixture constant throughout an experiment.

Advanced Options > Split Plot Variance Ratio Specify the ratio of the variance of the random whole plot and the subplot variance (if present) to the error variance. Before setting this value, you must define a hard-to-change factor for your split-plot design, or hard and very-hard-to-change factors for your split-split-plot design. Then you can enter one or two positive numbers for the variance ratios, depending on whether you have specified a split-plot or a split-split-plot design.

Advanced Options > Prior Parameter Variance (Available only when the Model outline is available.) Specify the weights that are used for factors whose Estimability is set to If Possible. The option updates to show the default weights when you click Make Design. Enter a positive number for each of the terms for which you want to specify a weight. The value that you enter is the square root of the reciprocal of the prior variance. A larger value represents a smaller variance and therefore more prior information that the effect is not active.

Bayesian D- or I-optimality is used in constructing designs with If Possible factors. The default values used in the algorithm are 0 for Necessary terms, 4 for interactions involving If Possible terms, and 1 for If Possible terms. See "The Alias Matrix" and "Optimality Criteria".

Advanced Options > A- Optimality Parameter Weights (Use for A-Optimal designs.) Specify weights for the model parameters. This enables you to place more weight on the variance of the main effects over say 2nd order effects. For more information about parameter weights see Morgan and Stallings (2017).

Advanced Options > D Efficiency Weight Specify the relative importance of D-efficiency to alias optimality in constructing the design. Select this option to balance reducing the variance of the coefficients with obtaining a desirable alias structure. Values should be between 0 and 1. Larger values give more weight to D-Efficiency. The default value is 0.5. This option has an effect only when you select Make Alias Optimal Design as your Optimality Criterion.

For the definition of D-efficiency, see "Optimality Criteria". For more information about alias optimality, see "Alias Optimality".

Save Script to Script Window Creates the script for the design that you specified in the Custom Design window and places it in an open script window.

Simulate Responses

Use simulated responses for design exploration prior to data collection or in a classroom setting. When simulate responses is selected and you click Make Table to create your design table, the Simulate Responses option does the following for each response:

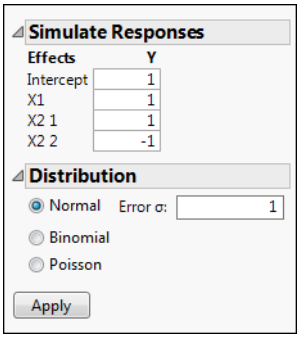
- It adds random response values to the response column in your design table.
- It adds a new a column containing a simulation model formula to the design table. The formula and values are based on the model that is specified in the design window.

A Model window appears where you can specify parameter values and select a response distribution for simulation. When you click Apply in the Model window, each column containing a simulation model formula is updated.

Control Window

Figure 4.27 shows the Model window for a design with one continuous factor (X1) and one three-level categorical factor (X2). Notice that X2 is represented by two model terms.

Figure 4.27 Simulate Responses Control Window



The initial window shows values for the coefficients of either 1 or -1, and a Normal distribution with error standard deviation equal to 1. If you have set Anticipated Coefficients as part of Power Analysis under Design Evaluation in the DOE window, then the default values in the Simulate Responses outline are the values that you specified as Anticipated Coefficients and Anticipated RMSE (Error Std) in the Power Analysis outline. If it is not possible to fit the model specified in the data table's Model script, the intercept and coefficients have default values of 0.

Simulate Responses

To specify a model for simulated values, do the following:

1. For each term in the list of Effects, enter coefficients for the linear model used to simulate the response values. These define a linear function, $L(\mathbf{x}, \beta) = \mathbf{x}'\beta$. See the Simulate Responses outline in Figure 4.27:
   - The vector **x** consists of the terms that define the effects listed under Effects.
   - The vector β is the vector of model coefficients that you specify under Y.
2. Under Distribution, select a response distribution.
3. Click **Apply**. A <Y> Simulated column containing simulated values and their formula is added to the design table, where Y is the name of the response column.

Distribution

Choose from one of the available distributions in the Simulate Responses window:

Normal Simulates values from a normal distribution. Enter a value for Error σ, the standard deviation of the normal error distribution. If you have designated factors to have Changes of Hard in the Factors outline, you can enter a value for Whole Plots σ, the whole plot error. If you have designated factors to have Changes of Hard and Very Hard, you can enter values for both the subplot and whole plot errors. When you click Apply, random values and a formula containing a random response vector based on the model are entered in the column <Y> Simulated.

Binomial Simulates values from a binomial distribution. Enter a value for N, the number of trials. Random integer values are generated according to a binomial distribution based on N trials with probability of success $1/(1 + exp(-L(\mathbf{x}, \beta))$. When you click Apply, random values and their formula are entered in the column <Y> Simulated. A column called N Trials that contains the value N is also added to the data table.

Poisson Simulates random integer values according to a Poisson distribution with parameter $exp((L(\mathbf{x}, \beta))$. When you click Apply, random values and their formula are entered in the column <Y> Simulated.

Note: You can set a preference to simulate responses every time you click Make Table. To do so, select **File > Preferences > Platforms > DOE**. Select **Simulate Responses**.

Save X Matrix

This option saves scripts called Moments Matrix and Model Matrix that contain the moments matrix and the model matrix to the design data table. The moments matrix and the model matrix are used to calculate the Average Variance of Prediction, which appears in the Design Diagnostics section of the Design Evaluation outline. See Goos and Jones (2011). If the design is a split-plot design, a V Inverse script is also saved. The V Inverse script contains the inverse of the covariance matrix of the responses.

Caution: For a design with nominal factors, the Model Matrix saved by the Save X Matrix option is *not* the coding matrix used in fitting the linear model. You can obtain the coding matrix used for fitting the model by selecting the option Save Columns > Save Coding Table in the Fit Model report that you obtain when you run the Model script.

Note: You can set a preference to always save the matrix script. Select **File > Preferences > Platforms > DOE**. Check Save X Matrix.

Model Matrix

The *model matrix* describes the design for the experiment. The model matrix has a row for each run and a column for each term of the model specified in the Model outline. For each run, the corresponding row of the model matrix contains the coded values of the model terms:

- Continuous terms are coded to span the range from -1 to 1.
- Nominal terms are coded by applying the Gram-Schmidt orthogonalization procedure to the coding vectors used in fitting linear models.

For more information about the Gram-Schmidt orthogonalization procedure, see Horn and Johnson (2012).

Note: Coding for power analysis matches that of fitting linear models. For more information about the coding used for nominal terms in fitting linear models, see *Fitting Linear Models*.

Moments Matrix

The *moments matrix* is dependent upon the model effects but is independent of the design. It is defined as follows:

$$\mathbf{M} = \int_R f(\mathbf{x}) f(\mathbf{x})' d\mathbf{x}$$

where *f(x)* denotes the model effects corresponding to factor combinations of the vector of factors, *x*, and *R* denotes the design space. For more information about moments and design matrices, see Goos and Jones (2011) and Myers et al. (2009). Note that the moments matrix is called a matrix of region moments in Myers et al. (2009).

Scripts

From the Custom Design red triangle menu, select **Save X Matrix**. After the design and the table are created, in the Custom Design table, the Moments Matrix and Model Matrix scripts, and if the design is a split plot, the V Inverse script, are saved as table scripts.

- Right-click and Select **Edit** from either the Moments Matrix, Model Matrix, or V Inverse script. The script shows the corresponding matrix. You can copy this matrix into scripts that you write.
- When you run the Moments Matrix script, the log shows the number of rows in the moments matrix, called Moments.
- When you run the script Model Matrix, the log displays the number of rows in the model matrix, called X.
- When you run the script V Inverse, the log displays the number of rows in the inverse covariance matrix, called V Inverse.

Example

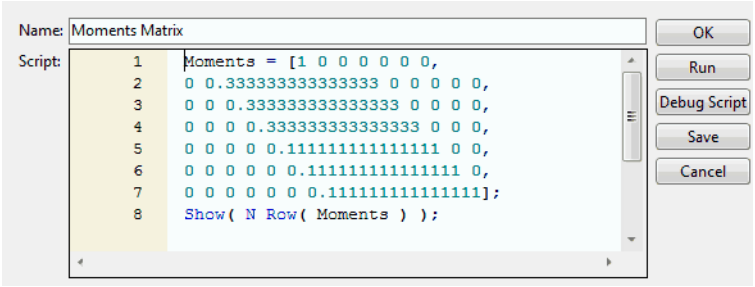
This example illustrates use of the model matrix script:

Tip: To see the log, select **View > Log** (**Window > Log** on macOS).

1. Select **DOE > Custom Design**.
2. Add 3 continuous factors and click **Continue**.
3. Click **Interactions > 2nd**.
4. Click the Custom Design red triangle and select **Save X Matrix**.
5. Using the Default Number of Runs (12), click **Make Design** and then **Make Table**.
6. In the Table panel, right-click the Moments Matrix script and select **Edit**.

   The script appears in a script window. The script shows the moments matrix, which is called **Moments**.

Figure 4.28 Moments Matrix Script



7. If it is not already open, select **View > Log** (**Window > Log** on the macOS).
8. In the Table panel, click the green triangle next to the Moments Matrix script.

   The number of rows appear in the log as `N Row(Moments)=7`.
9. In the Table panel, right-click the Model Matrix script and select **Edit**.

   The script appears in a script window. The script shows the model matrix, which is called **X**.
10. Click **Run**.

    The number of rows appears in the log as `N Row(X)=12`.
11. To view the Model Matrix as a data table, add these lines to the script:

```
dt = New Table( "Model Matrix" );
dt << Set Matrix( X );
```

12. Click **Run**.

Number of Starts

The number of starts is the number of times that the coordinate-exchange algorithm initiates with a new design. See "Coordinate-Exchange Algorithm". You can specify your own value using the **Number of Starts** option. Increasing the number of random starts tends to improve the optimality of the resulting design.

Unless you specify a value for Number of Starts and click OK, the number of starts is controlled by Design Search Time. To see how many starts were used to construct a design, click Make Design. Then select Number of Starts. The value in the text box is the number of starts used to construct the specific design.

In certain special cases, the globally optimal design is known from theory. If the coordinate-exchange algorithm detects that it has found an optimal design, it stops searching and returns that design.

Tip: To reproduce a specific design, you need to specify the Number of Starts and the Random Seed originally used to produce the design. Obtain these values from the red triangle options after you click Make Design.

In examples of custom designs in the documentation, the random seed and number of starts are often provided so that you can reconstruct the exact design being discussed.

Design Search Time

Design Search Time is the amount of time allocated to finding an optimal design. Custom Design's coordinate-exchange algorithm consists of finding near-optimal designs based on random starting designs. See "Coordinate-Exchange Algorithm". The Design Search Time determines how many designs are constructed based on random starting designs.

You can specify your own value using the **Design Search Time** option. Increasing the search time tends to improve the optimality of the resulting design.

Keep in mind that designs produced by rerunning the algorithm can differ. Even with the same random seed, the numbers of starting designs used to construct the final design might differ because of variations in computing speed and other factors.

Note: The number of starting designs is given by the value in the Number of Starts text box. However, this value is not updated until after you construct your design by clicking Make Design.

In certain special cases, the globally D-optimal design is known from theory. These cases include:

- Two-level fractional factorial designs or nonregular orthogonal arrays. These are globally D-optimal for all main effect and two-factor interaction models.
- Latin-square designs. These are D-optimal for main effect models assuming the right sample size and numbers of levels of the factors.
- Plackett-Burman designs. These are D-optimal for main effect models.

If the coordinate-exchange algorithm detects that it has found an optimal design, it stops searching and returns that design.

Technical Details for Custom Designs

- "Designs with Randomization Restrictions"
- "Covariates with Hard-to-Change Levels"
- "Numbers of Whole Plots and Subplots"
- "Optimality Criteria"
- "D-Efficiency"
- "Coordinate-Exchange Algorithm"

Designs with Randomization Restrictions

It is not always feasible to run a completely randomized design. Logistical or physical restrictions to randomization have to be considered when designing an experiment. This section describes how the Custom Design platform handles various types of designs where random assignment of experimental units to factor level settings is restricted. Random block designs and various types of split-plot designs are included.

Random Block Designs

A random block design groups the runs of an experiment into blocks that are considered to be randomly chosen from a larger population. Runs within a block of runs are usually more homogeneous than runs in different blocks. In these instances, you are often better able to discern other effects if you account for the variation explained by the blocking variables.

Scenario for a Random Block Design

Goos (2002) presents an example involving a pastry dough mixing experiment. The purpose of the experiment is to understand how certain properties of the dough depend on three factors: feed flow rate, initial moisture content, and rotational screw speed. Since it was possible to conduct only four runs a day, the experiment required several days to run. It is likely that random day-to-day differences in environmental variables have some effect on all of the runs that are performed on a given day. To account for the day-to-day variation, the runs were grouped into blocks of size four so that this variation would not compromise the information about the three factors.

The blocking factor, Day, consists of each day's runs. The days on which the trials were conducted are representative of a large population of days with different environmental conditions. Day is a random blocking factor.

Setup for a Random Block Design

To create a random block design, use the Custom Design platform to enter responses and factors and define your model as usual. In the Design Generation outline, select the **Group runs into random blocks of size** option and enter the number of runs you want in each block. See "Design Structure Options".

Note: To define a fixed blocking factor, enter a blocking factor in the Factors outline. To define a random blocking factor, do not enter a blocking factor in the Factors outline. Instead, select the **Group runs into random blocks of size** option under Design Generation.

Split-Plot Designs

Split-plot designs are used in situations where the settings of certain factors are held constant for groups of runs. In industry, these are usually factors that are difficult or expensive to change from run to run. Factors whose settings need to be held constant for groups of runs are classified as *hard-to-change* in JMP.

Because certain factors are hard-to-change, it is not practical to randomly allocate them to experimental units. Instead, they are allocated to groups of units. This imposes a restriction on randomization that must be considered in generating a design and in analyzing the results.

Scenario for a Split-Plot Design

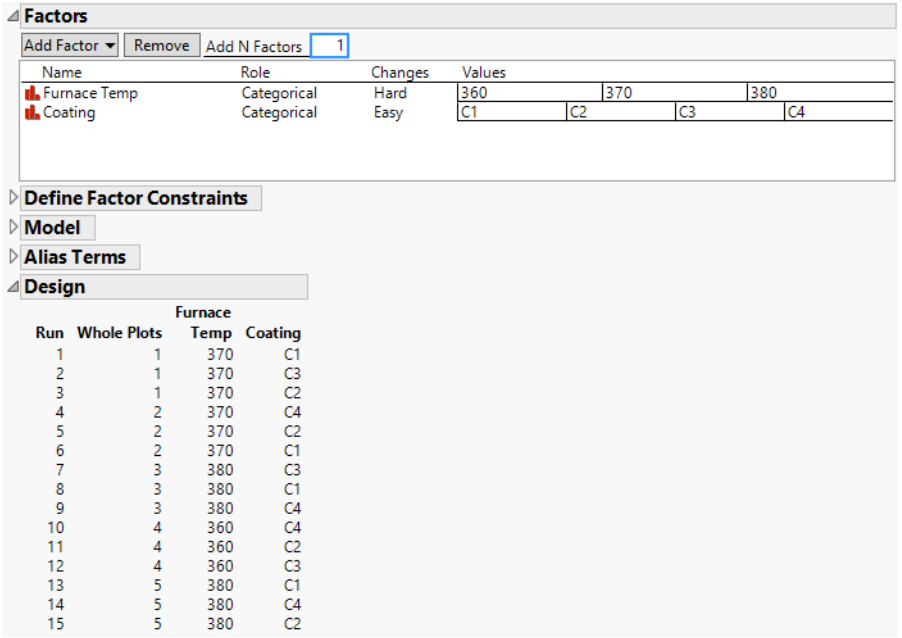
Box et al. (2005) presents an experiment to study the corrosion resistance of steel bars. The bars are placed in a furnace for curing. Afterward, a coating is applied to increase resistance to corrosion. The two factors of interest are:

- Furnace Temp in degrees centigrade, with levels 360, 370, and 380
- Coating, with levels C1, C2, C3, and C4 depicting four different types of coating

Furnace Temp is a hard-to-change factor, due to the time it takes to reset the temperature in the furnace. For this reason, four bars are processed for each setting of furnace temperature. At a later stage, the four coatings are randomly assigned to the four bars.

The experimental units are the bars. Furnace Temp is a hard-to-change factor whose levels define whole plots. Within each whole plot, the Coating factor is randomly assigned to the experimental units to which the whole plot factor was applied.

Figure 4.29 Factors and Design Outlines for Split-Plot Design



Run	Whole Plots	Furnace Temp	Coating
1	1	370	C1
2	1	370	C3
3	1	370	C2
4	2	370	C4
5	2	370	C2
6	2	370	C1
7	3	380	C3
8	3	380	C1
9	3	380	C4
10	4	360	C4
11	4	360	C2
12	4	360	C3
13	5	380	C1
14	5	380	C4
15	5	380	C2

The Factors outline for the corrosion experiment has Changes set to Hard for Furnace Temp and Easy for Coating. The 15-run design consists of five whole plots, within which the settings of Temperature are held constant.

Setup for a Split-Plot Design

In general, several factors can be applied to a processing step where settings are hard-to-change. In the furnace example, you might consider a furnace location factor, as well as temperature. In the Factors outline, under the Changes column, you would specify a Changes value of Hard for such factors.

When a custom design involves only easy-to-change and hard-to-change factors, the runs of the hard-to-change factors are grouped using a new factor called Whole Plots. The values of Whole Plots designate blocks of runs with identical settings for the hard-to-change factors. The Model script that is saved to the design table treats Whole Plots as a random effect. See "Changes" and "Design Structure Options".

For an example of creating a split-plot design and analyzing the experimental data, see "Split-Plot Experiment".

Split-Split-Plot Designs

A split-split-plot design is used when there are two levels of factors that are hard-to-change. In industry, such designs often occur when batches of material or experimental units from one processing stage pass to a second processing stage. Factors are applied to batches of material at the first stage. Then those batches are divided for second-stage processing, where additional factors are studied. The first stage factors are considered *very-hard*-to-change, and the second-stage factors are considered *hard*-to-change. Additional factors can be applied to experimental units after the second processing stage. These factors are considered *easy*-to-change.

In a split-split-plot design, the batches are considered to be random blocks. Since the batches are divided for second-stage processing, the second-stage factors are nested within the first-stage factors.

Scenario for a Split-Split-Plot Design

Schoen (1999) presents an example of a split-split-plot design that relates to cheese quality. The factors are given in the Cheese Factors.jmp data table found in the Design Experiment folder. The experiment consists of three stages of processing:

- Milk is received from farmers and stored in a large tank.
- Milk from this tank is distributed to smaller tanks used for curd processing.
- The curds from each tank are transported to presses for processing individual cheeses.

The experiment consists of testing:

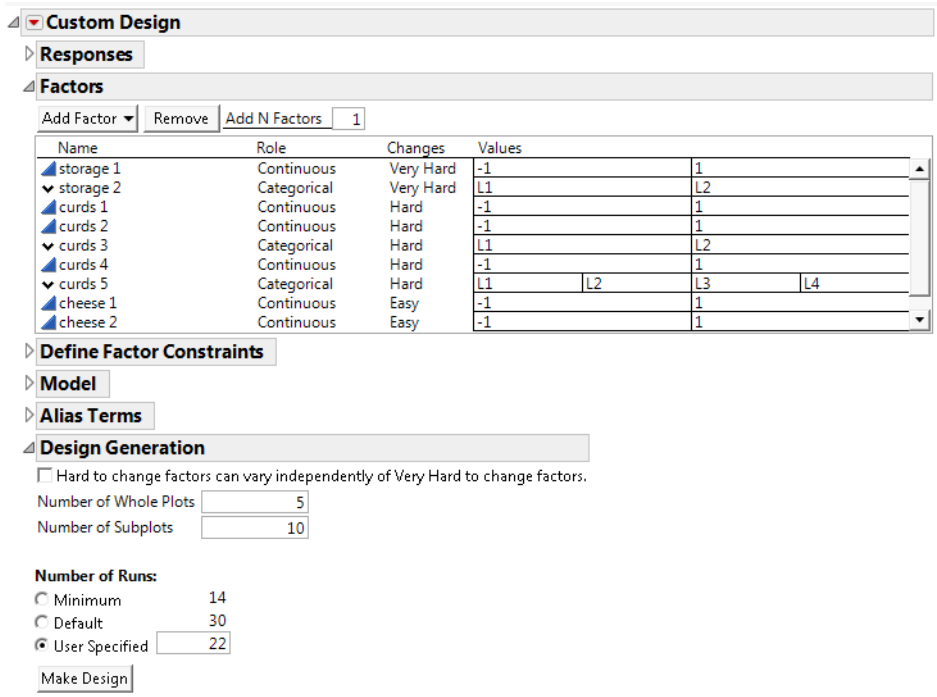
- Two factors that are applied when the milk is in the large storage tank.
- Five factors that are applied to the smaller curd processing tanks.
- Three factors that are applied to the individual cheeses from a curds processing tank.

Notice that the levels of factors applied to the curd processing tanks (subplots) are nested within the levels of factors applied to the milk storage tank (whole plots).

The Factors outline have Changes set as Very Hard, Hard, or Easy:

- Very Hard for the two storage tank factors.
- Hard for the five curd processing tank factors.
- Easy for the three factors that can be randomly assigned to cheeses.

Figure 4.30 Factors and Design Generation Outline for Split-Split-Plot Design



The default number of whole plots is 5 and the default number of subplots is 10.The number of runs is set to 22. If only 10 subplots are used the design does not have enough whole plots to estimate the subplot variance. Change the number of subplots to 11 and click **Make Design** to see a 22-run design.

Figure 4.31 Split-Split-Plot Design for Cheese Scenario

Design

Run	Whole Plots	Subplots	storage 1	storage 2	curds 1	curds 2	curds 3	curds 4	curds 5	cheese 1	cheese 2	cheese 3
1	1	1	-1	L2	1	1	L1	-1	L2	1	-1	L3
2	1	1	-1	L2	1	1	L1	-1	L2	-1	1	L1
3	1	2	-1	L2	-1	-1	L2	-1	L3	1	1	L1
4	1	2	-1	L2	-1	-1	L2	-1	L3	-1	-1	L3
5	1	3	-1	L2	1	-1	L2	1	L1	1	1	L3
6	2	4	1	L1	-1	1	L1	-1	L1	1	-1	L2
7	2	4	1	L1	-1	1	L1	-1	L1	1	1	L3
8	2	4	1	L1	-1	1	L1	-1	L1	-1	1	L1
9	2	5	1	L1	1	1	L1	1	L3	-1	-1	L1
10	2	5	1	L1	1	1	L1	1	L3	1	1	L2
11	3	6	1	L2	1	-1	L2	-1	L2	1	-1	L1
12	3	6	1	L2	1	-1	L2	-1	L2	-1	1	L2
13	3	7	1	L2	-1	1	L1	1	L4	1	1	L2
14	3	7	1	L2	-1	1	L1	1	L4	-1	-1	L3
15	4	8	1	L1	-1	1	L2	1	L2	1	1	L3
16	4	8	1	L1	-1	1	L2	1	L2	-1	-1	L1
17	4	9	1	L1	1	-1	L1	-1	L4	1	-1	L1
18	4	9	1	L1	1	-1	L1	-1	L4	-1	1	L2
19	5	10	-1	L1	1	1	L2	-1	L4	-1	-1	L2
20	5	10	-1	L1	1	1	L2	-1	L4	1	1	L1
21	5	11	-1	L1	-1	-1	L1	1	L2	-1	1	L3
22	5	11	-1	L1	-1	-1	L1	1	L2	1	-1	L2

The five whole plots correspond to the storage factors; storage 1 and storage 2. The settings of the storage factors are constant within a whole plot. If consecutive whole plots have the same setting for a whole plot factor, the factor should be reset between the plots. For example, you should reset the level for storage 1 between runs 10 and 11 and between runs 14 and 15, and you should reset the level for storage 2 between runs 18 and 19. Resetting the factor between whole plots, even when the specified settings are the same, is required in order to capture whole plot variation.

The 11 subplots correspond to the curds factors. Within a subplot, the settings of the curds factors are constant. Each level of Subplots appears only within one level of Whole Plots, indicating that the levels of Subplots are nested within the levels of Whole Plots.

Levels of the cheese factors vary randomly from run to run.

Setup for a Split-Split-Plot Design

In a split-split-plot design, the Factors outline contains factors with Changes set to Very Hard and Hard. The design can also contain factors with Changes set to Easy. Two factors are created:

- A factor called Whole Plots represents the blocks of constant levels of the factors with Changes set to Very Hard.
- A factor called Subplots represents the blocks of constant levels of the factors with Changes set to Hard.
- The factor Subplots reflects the nesting of the levels of the factors with Changes set to Hard within the levels of the factors with Changes set to Very Hard.

- The levels of factors with Changes set to Easy are randomly assigned to units within subplots.
- The factors Whole Plots and Subplots are treated as random effects in the Model script that is saved to the design table.

See the Changes description under "Factors Outline" and "Design Structure Options".

Two-Way Split-Plot Designs

A two-way split-plot (also known as strip plot or split block) design consists of two split-plot components. In industry, these designs arise when batches of material or experimental units from one processing stage pass to a second processing stage. But, after the first processing stage, it is possible to divide the batches into sub-batches. The second-stage processing factors are applied randomly to these sub-batches. For a specific second-stage experimental setting, all of the sub-batches assigned to that setting can be processed simultaneously. Additional factors can be applied to experimental units after the second processing stage.

In contrast to a split-split-plot design, the second-stage factors are *not nested* within the first-stage factors. After the first stage, the batches are subdivided and formed into new batches. Therefore, both the first- and second-stage factors are applied to whole batches.

Although factors at both stages might be equally hard-to-change, to distinguish these factors, JMP denotes the first stage factors as *very-hard*-to-change, and the second-stage factors as *hard*-to-change. Additional factors applied to experimental units after the second processing stage are considered *easy*-to-change.

Scenario for a Two-Way Split-Plot Design

Vivacqua and Bisgaard (2004) describe an experiment to improve the open circuit voltage in battery cells. Two stages of processing are of interest:

- First stage: A continuous assembly process
- Second stage: A curing process with a 5-day cycle time

The engineers want to study six two-level factors:

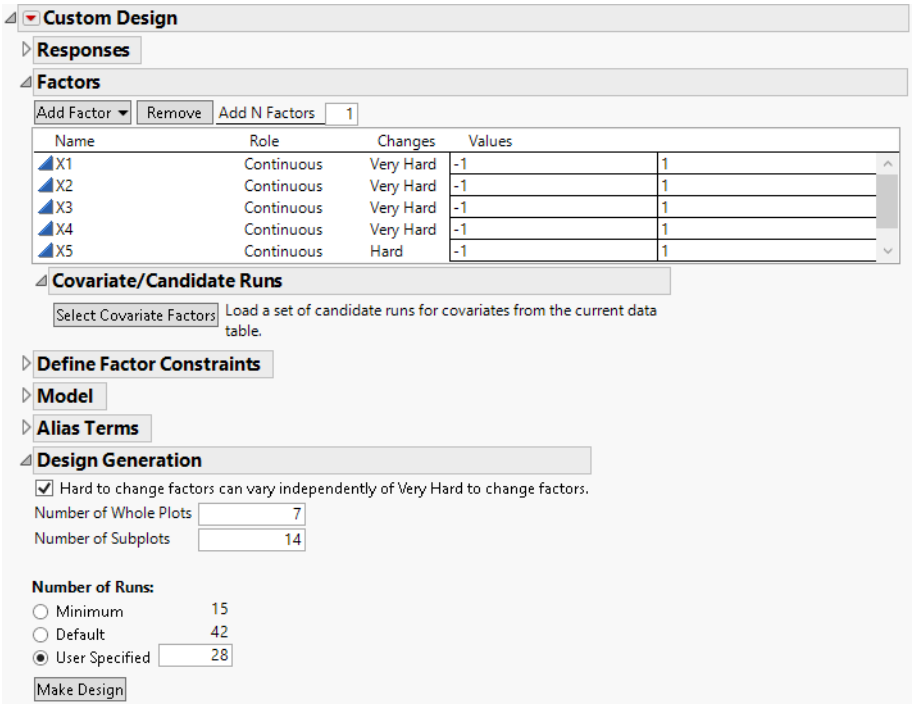
- Four factors, X1, X2, X3, and X4, that are applied to the assembly process
- Two factors, X5 and X6, that are applied to the curing process

A full factorial design with all factors at two levels would require $2^6 = 64$ runs, and would require a prohibitive 64*5 = 320 days. Also, it is not practical to vary assembly conditions for individual batteries. However, assembly conditions can be changed for large batches, such as batches of 2000 batteries.

Both the first- and second-stage factors are hard-to-change. In a sense, there are two split-plot designs. However, the batches of 2,000 batteries from the first-stage experiment can be divided into four sub-batches of 500 batteries each. These sub-batches can be randomly assigned to the four settings of the two second-stage factors. All of the batches assigned to a given set of curing conditions can be processed simultaneously. In other words, the first- and second-stage factors are *crossed*.

To distinguish between the first- and second-stage factors, you designate the Changes for the first-stage factors as Very Hard, and the Changes for the second-stage factors as Hard (Figure 4.32). Also, under Design Generation, note the following option: **Hard to change factors can vary independently of Very Hard to change factors**. If this is not checked, the design is treated as a split-split-plot design, with nesting of factors at the two levels. Check this option to create a two-way split-plot design.

Figure 4.32 Factors and Design Generation Outline for Two-Way Split Plot Design



The default number of whole plots is 7; the default number of subplots is 14. Click **Make Design** to see the 28-run design.

Figure 4.33 Two-Way Split-Plot Design for Battery Cells

Design

Run	Whole Plots	Subplots	X1	X2	X3	X4	X5	X6
1	1	1	1	1	1	-1	1	-1
2	1	2	1	1	1	-1	-1	1
3	1	3	1	1	1	-1	-1	-1
4	1	4	1	1	1	-1	1	1
5	2	5	-1	-1	-1	-1	-1	1
6	2	6	-1	-1	-1	-1	1	-1
7	2	7	-1	-1	-1	-1	-1	-1
8	2	8	-1	-1	-1	-1	1	1
9	3	9	1	1	-1	-1	-1	1
10	3	10	1	1	-1	-1	1	-1
11	3	11	1	1	-1	-1	-1	-1
12	3	12	1	1	-1	-1	1	1
13	4	13	1	-1	-1	1	-1	-1
14	4	14	1	-1	-1	1	1	1
15	4	1	1	-1	-1	1	1	-1
16	4	2	1	-1	-1	1	-1	1
17	5	3	-1	1	-1	1	-1	-1
18	5	4	-1	1	-1	1	1	1
19	5	5	-1	1	-1	1	-1	1
20	5	6	-1	1	-1	1	1	-1
21	6	7	1	-1	1	1	-1	-1
22	6	8	1	-1	1	1	1	1
23	6	9	1	-1	1	1	-1	1
24	6	10	1	-1	1	1	1	-1
25	7	11	-1	-1	1	-1	-1	-1
26	7	12	-1	-1	1	-1	1	1
27	7	13	-1	-1	1	-1	-1	-1
28	7	14	-1	-1	1	-1	1	1

The seven whole plots correspond to the first-stage factors, X1, X2, X3, and X4. The settings of these factors are constant within a whole plot. The 14 subplots correspond to the second-stage factors, X5 and X6. For example, the sub-batches for runs 1 and 15 (from different whole plots) are subject to the same subplot treatment, where X5 is set at 1 and X6 at -1.

Setup for a Two-Way Split-Plot Design

A two-way split-plot design requires factors with Changes set to Very Hard and to Hard. As described in "Setup for a Split-Split-Plot Design", factors called Whole Plots and Subplots are created. However, in a two-way split-plot design, Subplots does not nest the levels of factors with Changes set to Hard within the levels of factors with Changes set to Very Hard. Both Whole Plots and Subplots are treated as random effects in the Model script that is saved to the design table.

You need to ensure that the factor Subplots is *not nested* within the factor Whole Plots. Select the option **Hard to change factors can vary independently of Very Hard to change factor** in the Design Generation outline (Figure 4.32). See "Changes" and "Design Structure Options".

For an example of creating a split-plot design and analyzing the experimental data, see "Two-Way Split-Plot Experiment".

Covariates with Hard-to-Change Levels

The Custom Design platform enables you to designate covariates as hard-to-change. Suppose that you have measurements on batches of material that are available for use in testing experimental factors. Or suppose that you have measurements on individuals who might be selected to participate in testing experimental factors. These measurements are known in advance of the experiment and are considered to be covariates.

The batches or individuals correspond to whole plots. You might want to use only some of these whole plots in your experiment. Because information about the whole plots in the form of covariates is available, the design should choose the whole plots in an optimal fashion.

The model, as given by the terms that you include in the Model outline, can include interactions and powers constructed using covariates and experimental factors.

Note: When you set Changes for a Covariate factor to Hard, all other covariates are also set to Hard The remaining factors must be set to Easy. Because the algorithm requires a combination of row exchange and coordinate exchange, even moderately sized designs might take some time to generate.

Scenario for an Experiment with a Hard-to-Change Covariate

An experiment involving batches of polypropylene plates is discussed in Goos and Jones (2011, Chapter 9) and Jones and Goos (2015). Large batches of polypropylene plates are produced according to various formulations determined by several variables. Some plates are used immediately, and the remainder are stored for future experimental purposes. The compositions of these stored batches are known.

A customer has certain requirements regarding the plate formulation. Future experiments involve customizing the gas plasma treatment to the types of formulations required by the customer. The composition variables are considered hard-to-change covariates. Gas plasma treatment factors can be applied to sub-batches of plates with a given formulation.

The optimal design identifies the batches (defined by the covariates) to use, determines the number of plates from each batch to use, and provides settings for the gas plasma levels. Note that the optimal number of batches and plates from a given batch depend on the covariates.

An example is provided in “Examples of Custom Designs”.

Numbers of Whole Plots and Subplots

JMP suggests default values for the Number of Whole Plots and Number of Subplots. These values are based on heuristics for creating a balanced design that allows for estimation of the effects specified in the Model outline.

If you enter missing values for Number of Whole Plots or Number of Subplots, JMP chooses values that maximize the D-efficiency of the design. The algorithm uses the values specified in the Split Plot Variance Ratio option. See "Advanced Options > Split Plot Variance Ratio". The D-efficiency is given by the determinant of $\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}$, where $\mathbf{V}^{-1}$is the inverse of the variance matrix of the responses. See Goos (2002).

If you enter values for the Number of Whole Plots and Number of Subplots, Custom Design attempts to maximize the optimality of the resulting design. For more information about split-plot designs, see Jones and Goos (2007). For more information about designs with hard-to-change covariates, see Jones and Goos (2015).

Optimality Criteria

Custom designs are created using search routines that depend on an optimality criteria. This section provides information about the optimality options for creating a custom design:

- "D-Optimality"
- "Bayesian D-Optimality"
- "I-Optimality"
- "Bayesian I-Optimality"
- "A-Optimality"
- "Alias Optimality"

D-Optimality

By default, the Custom Design platform optimizes the D-optimality criterion except when a full quadratic model is created using the RSM button. In that case, an I-optimal design is constructed.

The D-optimality criterion minimizes the determinant of the covariance matrix of the model coefficient estimates. It follows that D-optimality focuses on precise estimates of the effects. This criterion is desirable in the following cases:

- screening designs
- experiments that focus on estimating effects or testing for significance
- designs where identifying the active factors is the experimental goal

The D-optimality criterion is dependent on the assumed model. This is a limitation because often the form of the true model is not known in advance. The runs of a D-optimal design optimize the precision of the coefficients of the assumed model. In the extreme, a D-optimal design might be saturated, with the same number of runs as parameters and no degrees of freedom for lack of fit.

Specifically, a D-optimal design maximizes *D*, where *D* is defined as follows:

$$D = \det[\mathbf{X}'\mathbf{X}]$$

and where **X** is the model matrix as defined in "Simulate Responses".

D-optimal split-plot designs maximize *D*, where *D* is defined as follows:

$$D = \det[\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}]$$

and $\mathbf{V}^{-1}$ is the block diagonal covariance matrix of the responses (Goos 2002).

Since a D-optimal design focuses on minimizing the standard errors of coefficients, it might not allow for checking that the model is correct. For example, a D-optimal design does not include center points for a first-order model. When there are potentially active terms that are not included in the assumed model, a better approach is to specify If Possible terms and to use a Bayesian D-optimal design.

Bayesian D-Optimality

Bayesian D-optimality is a modification of the D-optimality criterion. The Bayesian D-optimality criterion is useful when there are potentially active interactions or non-linear effects. See DuMouchel and Jones (1994) and Jones et al (2008).

Bayesian D-optimality estimates a specified set of model parameters precisely. These are the effects whose Estimability you designate as Necessary in the Model outline. But at the same time, Bayesian D-optimality has the ability to estimate other, typically higher-order effects, as allowed by the run size. These are the effects whose Estimability you designate as If Possible in the Model outline. To the extent possible given the run size restriction, a Bayesian D-optimal design allows for detecting inadequacy in a model that contains only the Necessary effects.

The Bayesian D-optimality criterion is most effective when the number of runs is larger than the number of Necessary terms, but smaller than the sum of the Necessary and If Possible terms. When this is the case, the number of runs is smaller than the number of parameters that you would like to estimate. Using prior information in the Bayesian setting allows for precise estimation of all of the Necessary terms while providing the ability to detect and estimate some If Possible terms.

To allow for a meaningful prior distribution to apply to the parameters of the model, responses and factors are scaled to have certain properties (DuMouchel and Jones, 1994, Section 2.2).

Consider the following notation:

- **X** is the model matrix as defined in "Simulate Responses"
- **K** is a diagonal matrix with values as follows:

- $k = 0$ for Necessary terms
- $k = 1$ for If Possible main effects, powers, and interactions involving a categorical factor with more than two levels
- $k = 4$ for all other If Possible terms

The prior distribution imposed on the vector of If Possible parameters is multivariate normal, with mean vector **0** and diagonal covariance matrix with diagonal entries $1/k^2$. Therefore, a value k^2 is the reciprocal of the prior variance of the corresponding parameter.

The values for k are empirically determined. If Possible main effects, powers, and interactions with more than one degree of freedom have a prior variance of 1. Other If Possible terms have a prior variance of 1/16. In the notation of DuMouchel and Jones (1994) $k = 1/\tau$.

To control the weights for If Possible terms, select **Advanced Options > Prior Parameter Variance** from the red triangle menu. See "Advanced Options > Prior Parameter Variance".

The posterior distribution for the parameters has the covariance matrix $(\mathbf{X}'\mathbf{X} + \mathbf{K}^2)^{-1}$. The Bayesian D-optimal design is obtained by maximizing the determinant of the inverse of the posterior covariance matrix:

$$\det(\mathbf{X}'\mathbf{X} + \mathbf{K}^2)$$

I-Optimality

I-optimal designs minimize the average variance of prediction over the design space. The I-optimality criterion is more appropriate than D-optimality if your primary experimental goal is not to estimate coefficients, but rather to do the following:

- predict a response
- determine optimum operating conditions
- determine regions in the design space where the response falls within an acceptable range

In these cases, precise prediction of the response takes precedence over precise estimation of the parameters.

The prediction variance relative to the unknown error variance at a point x_0 in the design space can be calculated as follows:

$$\text{var}(\hat{\mathbf{Y}}|\mathbf{x}_0) = f(\mathbf{x}_0)'(\mathbf{X}'\mathbf{X})^{-1}f(\mathbf{x}_0)$$

where $\mathbf{X}$ is the model matrix as defined in "Simulate Responses".

I-optimal designs minimize the integral I of the prediction variance over the entire design space, where I is given as follows:

$$I = \int_R f(\mathbf{x})'(\mathbf{X}'\mathbf{X})^{-1} f(\mathbf{x}) d\mathbf{x} = \text{trace}[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{M}]$$

Here $\mathbf{M}$ is the moments matrix:

$$\mathbf{M} = \int_R f(\mathbf{x}) f(\mathbf{x})' d\mathbf{x}$$

See "Simulate Responses".

The moments matrix does not depend on the design and can be computed in advance. The row vector $f(\mathbf{x})'$ consists of a 1 followed by the effects corresponding to the assumed model. For example, for a full quadratic model in two continuous factors, $f(\mathbf{x})'$ is defined as follows:

$$f(\mathbf{x})' = (1, x_1, x_2, x_1 x_2, x_1^2, x_2^2)$$

A-Optimality

Use an A-optimal design when you want to put emphasis on certain model effects. An A-optimal design enables you to assign weights to your model parameters. These weights are used in the A-optimality criterion to select the design. The resulting design will place more emphasis on factor combinations that lower the variance of the estimates for highly weighted terms than for terms with lower weights. For example, you may have a group of interactions that are more important to estimate than other interactions. You can increase the weights on the key interactions to decrease the variance of their estimates as compared to the other interactions.

The A-optimality criterion minimizes the trace of the covariance matrix of the model coefficient estimates. The trace is the sum of the main diagonal elements of a matrix. An A-optimal design minimizes the sum of the variances of the regression coefficients.

Bayesian I-Optimality

The Bayesian I-optimal design minimizes the average prediction variance over the design region for Necessary and If Possible terms.

Consider the following notation:

- $\mathbf{X}$ is the model matrix, defined in "Simulate Responses"
- $\mathbf{K}$ is a diagonal matrix with values as follows:

- $k = 0$ for Necessary terms
- $k = 1$ for If Possible main effects, powers, and interactions involving a categorical factor with more than two levels
- $k = 4$ for all other If Possible terms

The prior distribution imposed on the vector of If Possible parameters is multivariate normal, with mean vector **0** and diagonal covariance matrix with diagonal entries $1/k^2$. (For more information about the values k, see "Bayesian D-Optimality" .)

The posterior variance of the predicted value at a point $\mathbf{x}_0$ is as follows:

$$\text{var}(\hat{\mathbf{Y}}|\mathbf{x}_0) = f(\mathbf{x}_0)'(\mathbf{X}'\mathbf{X} + \mathbf{K}^2)^{-1} f(\mathbf{x}_0)$$

The Bayesian I-optimal design minimizes the average prediction variance over the design region, as follows:

$$I_B = \text{trace}[(\mathbf{X}'\mathbf{X} + \mathbf{K}^2)^{-1}\mathbf{M}]$$

where **M** is the moments matrix. See "Simulate Responses".

Alias Optimality

Alias optimality seeks to minimize the aliasing between effects that are in the assumed model and effects that are not in the model but are potentially active. Effects that are not in the model but that are of potential interest are called *alias effects*. For more information about alias-optimal designs, see Jones and Nachtsheim (2011b).

Specifically, let $\mathbf{X}_1$ be the model matrix corresponding to the terms in the assumed model, as defined in "Simulate Responses". The design defines the model that corresponds to the alias effects. Denote the matrix of model terms for the alias effects by $\mathbf{X}_2$.

The *alias matrix* is the matrix **A**, defined as follows:

$$\mathbf{A} = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2$$

The entries in the alias matrix represent the degree of bias associated with the estimates of model terms. See "The Alias Matrix" for the derivation of the alias matrix.

The sum of squares of the entries in **A** provides a summary measure of bias. This sum of squares can be represented in terms of a trace as follows:

$$\text{trace}(\mathbf{A}'\mathbf{A})$$

Designs that reduce the trace criterion generally have lower D-efficiency than the D-optimal design. Consequently, alias optimality seeks to minimize the trace of $\mathbf{A}'\mathbf{A}$ subject to a lower bound on D-efficiency. For the definition of D-efficiency, see "Optimality Criteria". The lower bound on D-efficiency is given by the D-efficiency weight, which you can specify under Advanced Options. See "Advanced Options > D Efficiency Weight".

D-Efficiency

Let $\mathbf{X}$ denote the design, or model, matrix for a given assumed model with p parameters. For the definition of the model matrix, see "Simulate Responses". Let $\mathbf{X}_D$ denote the model matrix for a D-optimal design for the assumed model. Then the D-efficiency of the design given by $\mathbf{X}$ is as follows:

$$\text{D-Efficiency} = \left[\frac{\det(\mathbf{X}'\mathbf{X})}{\det(\mathbf{X}_D'\mathbf{X}_D)}\right]^{1/p}$$

Coordinate-Exchange Algorithm

The Custom Designer constructs a design that seeks to optimize one of several optimality criteria. (See "Optimality Criteria".) To optimize the criterion, Custom Design uses the *coordinate-exchange algorithm* (Meyer and Nachtsheim, 1995). The algorithm begins by randomly selecting values within the specified design region for each factor and each run to construct a starting design.

Suppose your study requires continuous factors, no factor constraints, and a main-effects model. An iteration consists of testing each value of the model matrix, as follows:

- The current value of each factor is replaced by its two most extreme values.
- The optimality criterion is computed for both of these replacements.
- If one of the values increases the optimality criterion, this value replaces the old value.

The process continues until no replacement occurs for an entire iteration.

Appropriate adjustments are made to the algorithm to account for polynomial terms, nominal factors, and factor constraints.

The design obtained using this process is optimal in a large class of neighboring designs. But it is only *locally* optimal. To improve the likelihood of finding a globally optimal design, the coordinate-exchange algorithm is repeated a large number of times. Goos and Jones (2011, p. 36) recommend using at least 1,000 random starts for all but the most trivial design situations. The number of starting designs is controlled by the Number of Starts option. See "Number of Starts". Custom Design provides the design that maximizes the optimality criterion among all the constructed designs.

Chapter 5

Examples of Custom Designs

Perform Experiments That Meet Your Needs

Use the Custom Design platform as your primary tool for constructing a wide range of experimental designs. You can construct a variety of design types and fine tune them to your specific experimental needs and resource budget.

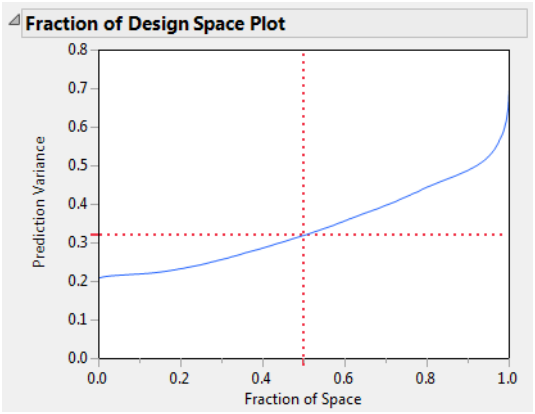
Custom Design provides more options and control than the Screening, Response Surface, Full Factorial, and Mixture Design platforms. The designs that you construct are created specifically to meet your goals. This eliminates the struggle to find a classical design that only comes close to meeting your goals.

The flexible special-purpose designs that you can construct using Custom Design include:

- Screening designs, including supersaturated screening designs
- Response surface designs, including those with categorical factors
- Mixture designs, including those with process factors, and mixture of mixture designs
- Designs that include covariates or that are robust to linear time trends
- Fixed and random block designs
- Split-plot, split-split-plot, and two-way split-plot (strip-plot) designs

In this chapter you construct most of these design types within the Custom Design platform. In many cases, you also analyze the experimental results. For help with using the Custom Design platform, see "Custom Designs".

Figure 5.1 Fraction of Design Space Plot



Contents

Screening Experiments . . . 137
Design That Estimates Main Effects Only . . . 137
Design That Estimates All Two-Factor Interactions . . . 140
Design That Avoids Aliasing of Main Effects and Two-Factor Interactions . . . 143
Supersaturated Screening Designs . . . 146
Design for Fixed Blocks . . . 154
Response Surface Experiments . . . 159
Response Surface Design . . . 159
Response Surface Design with Flexible Blocking . . . 165
Comparison of a D-Optimal and an I-Optimal Response Surface Design . . . 168
Response Surface Design With Constraints and Categorical Factor . . . 172
Mixture Experiments . . . 178
Mixture Design with Nonmixture Factors . . . 178
Mixture of Mixtures Design . . . 182
Experiments with Covariates . . . 187
Design with Fixed Covariates . . . 187
Design with Hard-to-Change Covariates . . . 190
Design with a Linear Time Trend . . . 194
Experiments with Randomization Restrictions . . . 197
Split-Plot Experiment . . . 197
Two-Way Split-Plot Experiment . . . 202
Experiments for Robust Process and Product Design . . . 210

Screening Experiments

In the early stages of studying a process, you identify a list of factors that potentially affect your response or responses. You are interested in identifying the *active* factors, that is, the factors that actually do affect your response or responses. A *screening design* helps you determine which factors are likely to be active. Once the active factors are identified, you can construct more sophisticated designs, such as response surface designs, to model interactions and curvature.

Screening designs constructed using the Custom Design platform are often equivalent to the classical designs provided in the Screening Design platform. However, Custom Design can construct designs for cases where classical screening designs are not available.

The Custom Design platform constructs screening designs using either the D-optimality or Bayesian D-optimality criterion. The D-optimality criterion minimizes the determinant of the covariance matrix of the model coefficient estimates. It follows that D-optimality focuses on precise estimates of the effects. See "Optimality Criteria".

- "Design That Estimates Main Effects Only"
- "Design That Estimates All Two-Factor Interactions"
- "Design That Avoids Aliasing of Main Effects and Two-Factor Interactions"
- "Supersaturated Screening Designs"
- "Design for Fixed Blocks"

Design That Estimates Main Effects Only

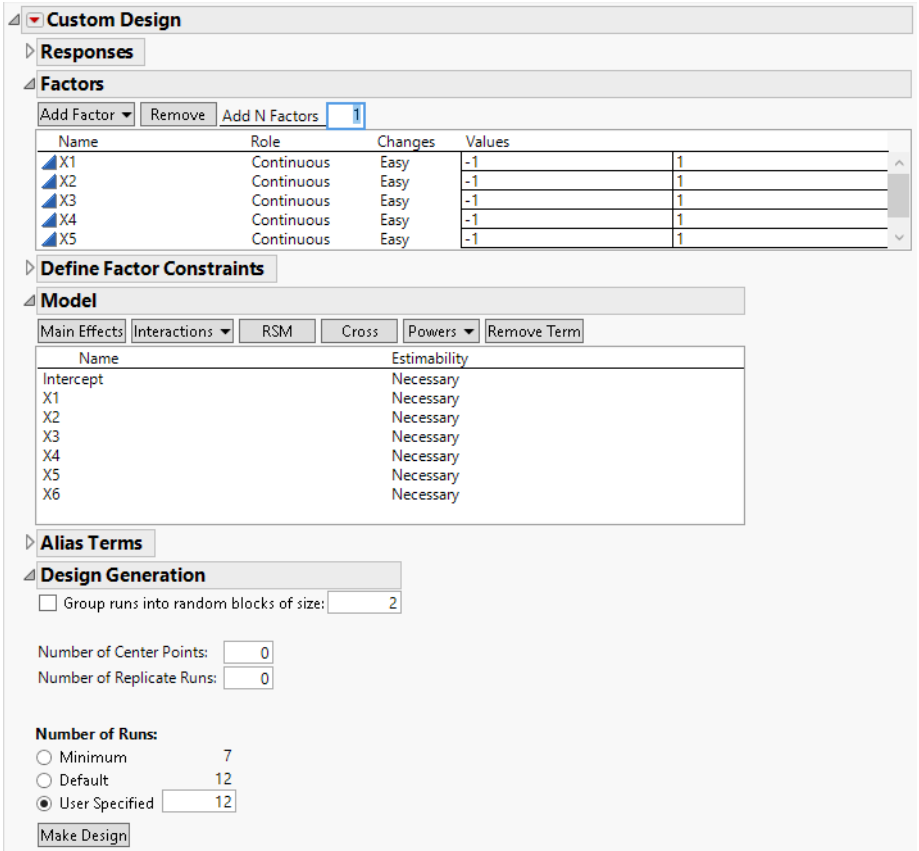
Note: For more information about main effects only screening designs, see "Screening Designs".

In this example, you are interested in studying the main effects of six factors. You construct a screening design where all of the main effects are orthogonal. However, the main effects are aliased with two-factor interactions.

1. Select **DOE > Custom Design**.
2. In the Factors outline, type 6 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Click **Continue**.

   The Model outline appears. It includes only main effects with Estimability designated as Necessary. This means that all main effects are estimable in the design that is generated.

Figure 5.2 Custom Design Window Showing Model Outline



Keep the default of 12 runs.

Note: Setting the Random Seed in step 5 and Number of Starts in step 6 reproduces the exact results shown in this example. When constructing a design on your own, these steps are not necessary.

5. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
6. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 1, and click **OK**.
7. Click **Make Design**.

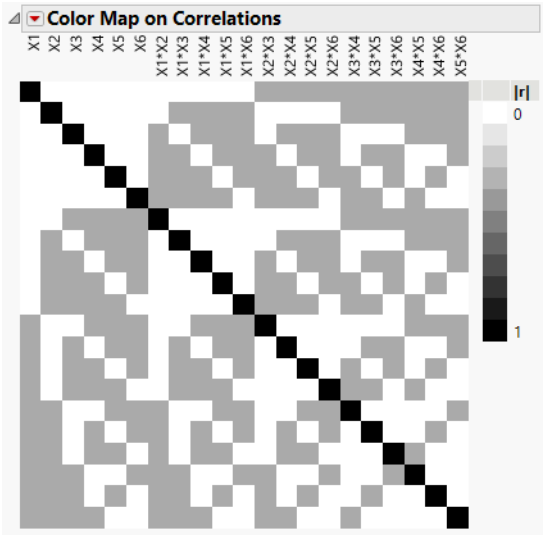
Figure 5.3 Design for Main Effects Only

Design

Run	X1	X2	X3	X4	X5	X6
1	-1	-1	-1	-1	-1	-1
2	1	-1	1	-1	-1	-1
3	1	1	-1	1	-1	-1
4	-1	1	1	-1	1	-1
5	1	-1	1	1	-1	1
6	1	1	-1	1	1	-1
7	1	1	1	-1	1	1
8	-1	1	1	1	-1	1
9	-1	-1	1	1	1	-1
10	-1	-1	-1	1	1	1
11	1	-1	-1	-1	1	1
12	-1	1	-1	-1	-1	1

8. Open the **Design Evaluation > Color Map on Correlations** outline.

Figure 5.4 Color Map on Correlations



Notice the following:

- The main effects are represented by the six terms in the upper left corner of the map.
- The white coloring corresponding to the correlations of the six main effects with other main effects indicates correlations of 0. This means that all main effects are orthogonal and can be estimated independently of each other.
- The light gray color in the squares corresponding to some two-way interactions indicates that the corresponding effects are correlated. This means that these effects cannot be estimated independently of other effects. Hover over these squares to see the exact correlation.

– Notice that no effects are completely confounded with each other. The black squares, indicating absolute correlations of 1, are only on the main diagonal.

9. Open the **Design Evaluation > Alias Matrix** outline.

Figure 5.5 Alias Matrix

Alias Matrix

Effect	X1*X2	X1*X3	X1*X4	X1*X5	X1*X6	X2*X3	X2*X4	X2*X5	X2*X6	X3*X4	X3*X5	X3*X6	X4*X5	X4*X6	X5*X6
Intercept	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X1	0	0	0	0	0	-0.33	0.333	0.333	-0.33	-0.33	-0.33	0.333	-0.33	-0.33	0.333
X2	0	-0.33	0.333	0.333	-0.33	0	0	0	0	-0.33	0.333	0.333	-0.33	-0.33	-0.33
X3	-0.33	0	-0.33	-0.33	0.333	0	-0.33	0.333	0.333	0	0	0	-0.33	0.333	-0.33
X4	0.333	-0.33	0	-0.33	-0.33	-0.33	0	-0.33	-0.33	0	-0.33	0.333	0	0	-0.33
X5	0.333	-0.33	-0.33	0	0.333	0.333	-0.33	0	-0.33	-0.33	0	-0.33	0	-0.33	0
X6	-0.33	0.333	-0.33	0.333	0	0.333	-0.33	-0.33	0	0.333	-0.33	0	-0.33	0	0

The Alias Matrix shows how the coefficients of the main effect terms in the model are biased by potentially active two-factor interaction effects. The column labels identify interactions. For example, in the X1 row, the column X2*X3 has a value of -0.33 and the column X2*X4 has a value of 0.333. This means that the expected value of the main effect of X1 is the sum of the main effect of X1 plus -0.33 times the effect of X2*X3, plus 0.333 times the effect of X2*X4, and so on, for the rest of the X1 row. In order for the estimate of the main effect of X1 to be meaningful, you must assume that these interactions are negligible in size compared to the effect of X1.

Tip: The Alias Matrix is a generalization of the confounding pattern in fractional factorial designs.

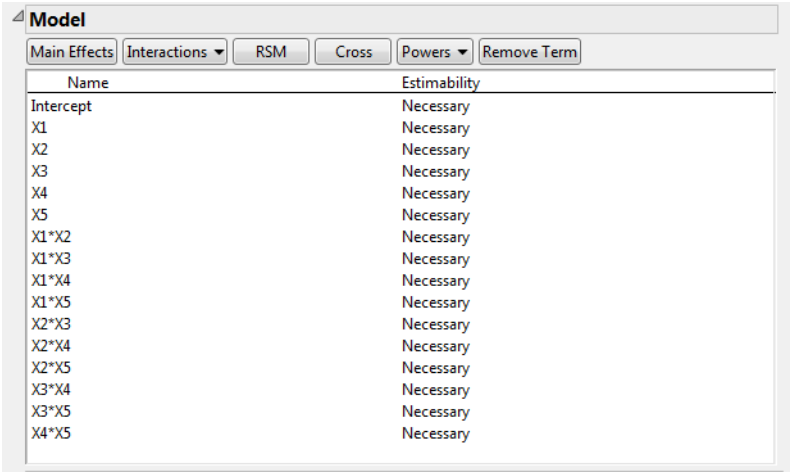
Design That Estimates All Two-Factor Interactions

The Alias Matrix in Figure 5.5 shows partial aliasing of effects. In other cases, main effects might be fully aliased, or *confounded*, with two-factor interactions. In both of these cases, strong two-factor interactions can confuse the results of main effects only experiments. To avoid this risk, create a design that resolves all two-factor interactions.

In this example, you create a resolution V screening design. Two-factor interactions are orthogonal, but they are confounded with three-factor interactions.

1. Select **DOE > Custom Design**.
2. Type 5 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Click **Continue**.
5. In the Model outline, select **Interactions > 2nd**.

Figure 5.6 Model Outline Showing Interactions



6. Click **Minimum** to accept 16 for the number of runs.

   Note: Setting the Random Seed in step 7 and Number of Starts in step 8 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

7. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
8. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 1, and click **OK**.
9. Click **Make Design**.

   Figure 5.7 shows the runs of the design. All main effects and two-factor interactions are estimable because their Estimability was designated as Necessary (by default) in the Model outline.

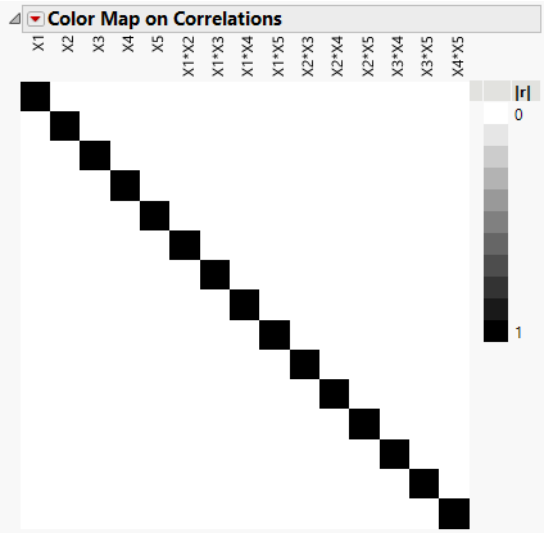
Figure 5.7 Design to Estimate All Two-Factor Interactions

Design

Run	X1	X2	X3	X4	X5
1	-1	-1	1	-1	-1
2	-1	-1	-1	1	-1
3	1	1	1	1	1
4	-1	-1	1	1	1
5	1	-1	1	-1	1
6	1	1	-1	1	-1
7	1	-1	-1	-1	-1
8	1	-1	1	1	-1
9	-1	1	1	-1	1
10	-1	-1	-1	-1	1
11	-1	1	-1	1	1
12	1	1	-1	-1	1
13	1	1	1	-1	-1
14	-1	1	1	1	-1
15	1	-1	-1	1	1
16	-1	1	-1	-1	-1

10. Open the **Design Evaluation > Color Map on Correlations** outline.

Figure 5.8 Color Map on Correlations



The Color Map indicates that the five main effects and the ten two-way interactions are all mutually orthogonal.

Design That Avoids Aliasing of Main Effects and Two-Factor Interactions

Suppose that your primary interest is in estimating the main effects of six continuous factors. However, you want to do this in a way that minimizes aliasing of main effects with potentially active two-factor interactions.

Your budget allows for only 16 runs. With six factors, there are 15 possible two-factor interactions. The minimum number of runs required to fit the constant, the six main effects, and the 15 two-factor interactions is 22.

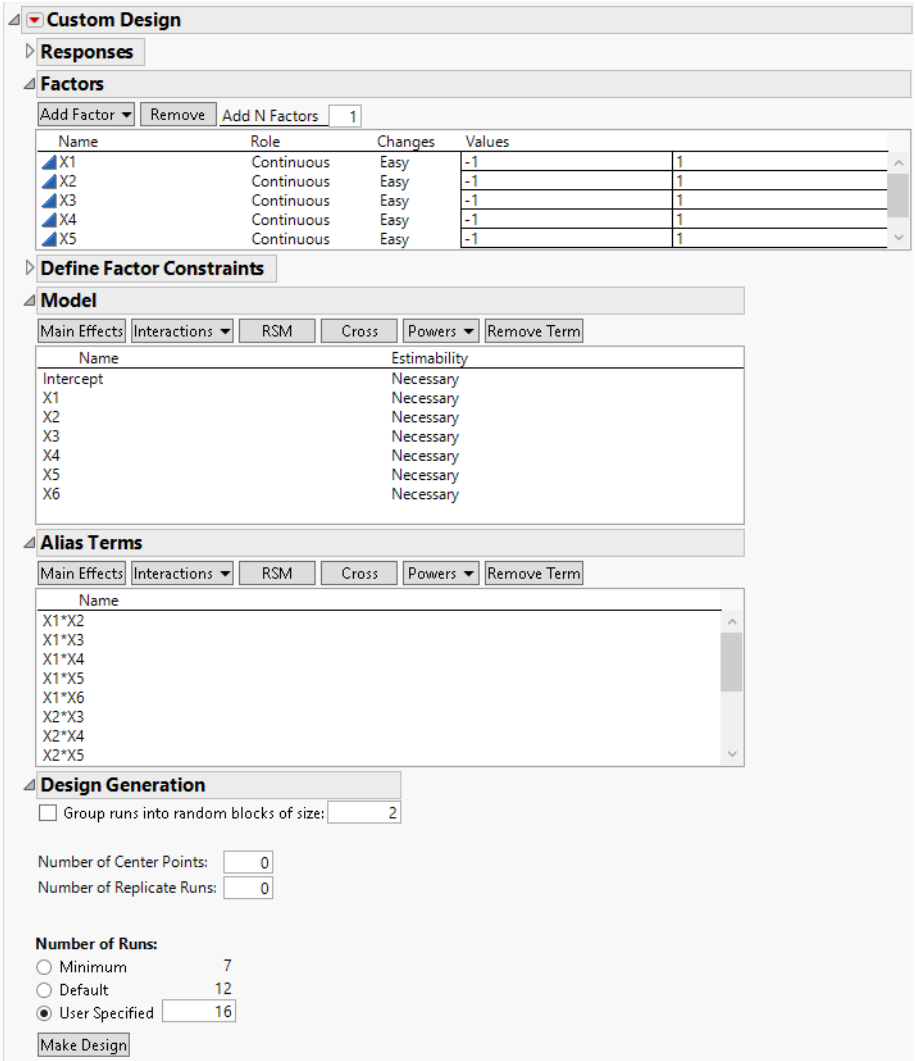
In this example, you find a compromise between an 8-run main effects only design (see "Design That Estimates Main Effects Only") and a 22-run design capable of fitting all the two-factor interactions. You use Alias Optimality as the optimality criterion to achieve your goal.

1. Select **DOE > Custom Design**.
2. Type 6 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Click **Continue**.

   The model includes the main effect terms by default. The default estimability of these terms is Necessary. In the Alias Terms outline, notice that second-order interactions are added. By default, all two-way interactions not included in the assumed model are added to the Alias Terms list.
5. Click the Custom Design red triangle and select **Optimality Criterion > Make Alias Optimal Design**.

   The Make Alias Optimal Design selection tells JMP to generate a design that balances reduction in aliasing with D-efficiency. See "Alias Optimality".
6. Click **User Specified** and change the number of runs to 16.

Figure 5.9 Factors, Model, Alias Terms, and Number of Runs



Note: Setting the Random Seed in step 7 and Number of Starts in step 8 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

7. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
8. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 10, and click **OK**.
9. Click **Make Design**.

10. Open the **Design Evaluation > Alias Matrix** outline.

Figure 5.10 Alias Matrix

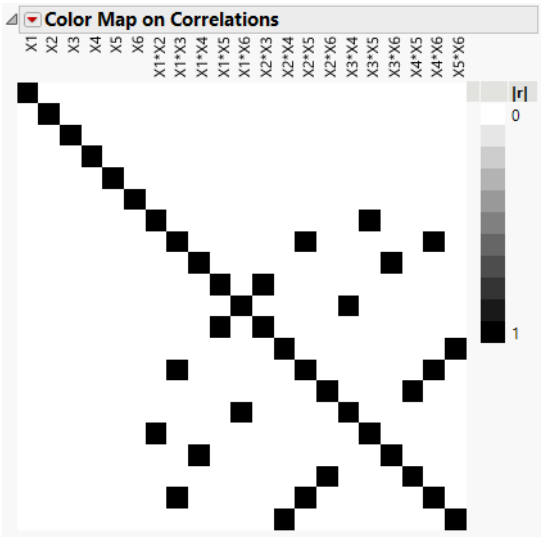
Alias Matrix

Effect	X1*X2	X1*X3	X1*X4	X1*X5	X1*X6	X2*X3	X2*X4	X2*X5	X2*X6	X3*X4	X3*X5	X3*X6	X4*X5	X4*X6	X5*X6
Intercept	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

All rows contain only zeros, which means that the Intercept and main effect terms are not biased by any two-factor interactions.

11. Open the **Design Evaluation > Color Map on Correlations** outline.

Figure 5.11 Color Map on Correlations



The Color Map on Correlations shows that main effects can be estimated independently of two-way interactions. However, some two-way interactions are fully aliased with other two-way interactions. Hover over the off-diagonal red squares to see which two-way interactions are confounded.

It turns out that this particular design is a resolution IV orthogonal screening design. Main effects are not aliased with each other or with two-way interactions. But two-way interactions are fully aliased with other two-way interactions.

Supersaturated Screening Designs

It is common for brainstorming sessions to identify dozens of potentially active factors. Rather than reduce the list without the benefit of data, you can use a supersaturated design.

In a *saturated* design, the number of runs equals the number of model terms. In a *supersaturated* design, the number of model terms exceeds the number of runs (Lin, 1993). A supersaturated design can examine dozens of factors using fewer than half as many runs as factors. This makes it an attractive choice for factor screening when there are many factors and experimental runs are expensive.

Alternatively, use a group orthogonal supersaturated design for improved active effect identification over traditional supersaturated designs. See "Group Orthogonal Supersaturated Designs".

Limitations of Supersaturated Designs

There are drawbacks to supersaturated designs:

- If the number of *active* factors is more than half the number of runs in the experiment, then it is likely that these factors will be impossible to identify. A general rule is that the number of runs should be at least four times larger than the number of active factors. In other words, if you expect that there might be as many as five active factors, you should plan on at least 20 runs.
- Analysis of supersaturated designs cannot yet be reduced to an automatic procedure. However, using forward stepwise regression is reasonable. In addition, the Screening platform (**DOE > Classical > Two Level Screening > Fit Two Level Screening**) offers a streamlined analysis.

Generate a Supersaturated Design

In this example, you want to construct a supersaturated design to study 12 factors in 8 runs. To create a supersaturated design, you set the Estimability of all model terms (except the intercept) to If Possible.

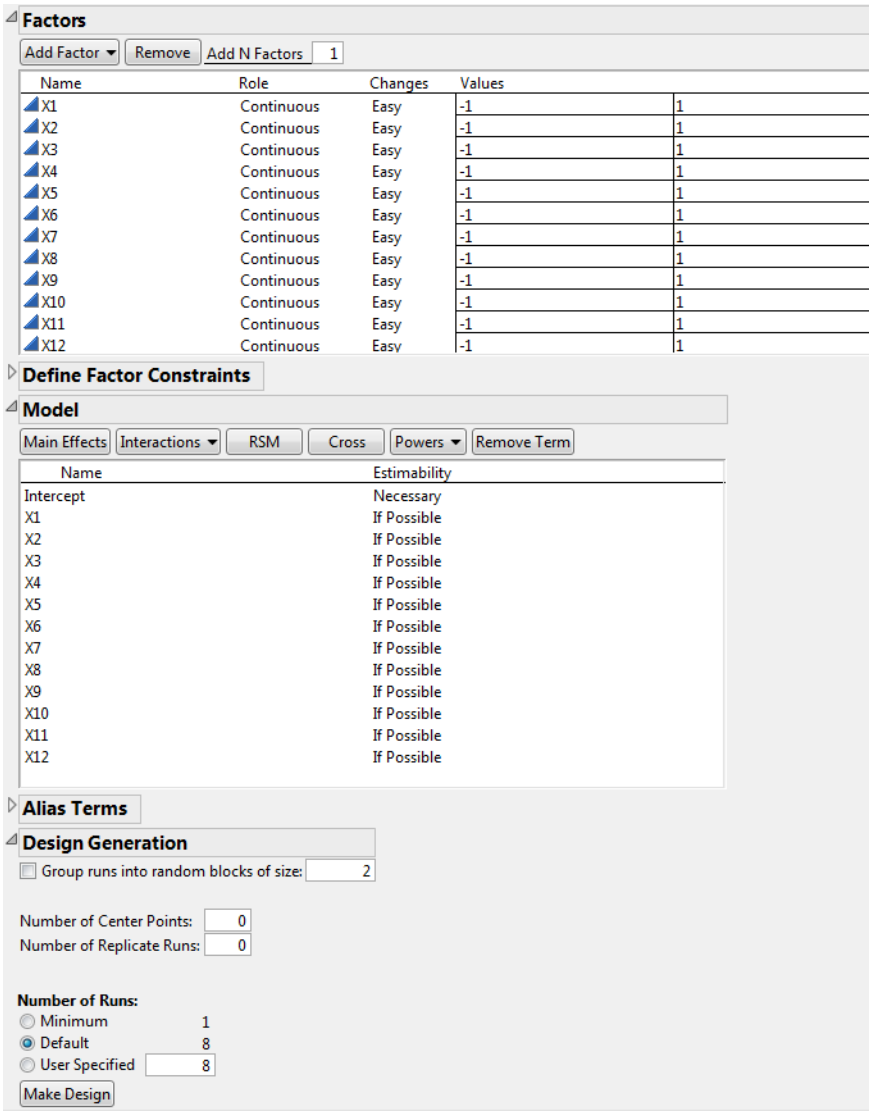
Note: This example is for illustration only. You should have at least 14 runs in any supersaturated design. If there are as many as four active factors, it is very difficult to interpret the results of an 8-run design. See "Limitations of Supersaturated Designs".

1. Select **DOE > Custom Design**.
2. Type 12 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Click **Continue**.

5. In the Model outline, select all terms except the Intercept.
6. Click Necessary next to any effect and change it to If Possible.

   Setting the effects to If Possible ensures that JMP uses the Bayesian D-optimality criterion to obtain the design.

Figure 5.12 Factors, Model, and Number of Runs



7. In the Alias Terms outline, select all effects and click **Remove Term**.

   This ensures that only the main effects appear in the Color Map on Correlations. This plot is constructed once the design is created.

8. Click the Custom Design red triangle and select **Simulate Responses**.

   This option generates random responses that appear in your design table. You will use these responses to see how to analyze experimental data.

   Keep the Number of Runs set to the Default of 8.

   Note: Setting the Random Seed in step 9 and Number of Starts in step 10 reproduces the design shown in this example. In constructing a design on your own, these steps are not necessary. Your Y Simulated values may not be the same as those shown in Figure 5.13.

9. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
10. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 5, and click **OK**.
11. Click **Make Design**.
12. Click **Make Table**.

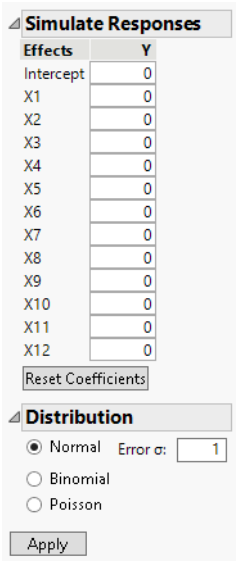
    Do not close your Custom Design window. You return to it later in this example.

Figure 5.13 Design Table with Simulated Responses

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	Y	Y Simulated
1	1	-1	1	-1	-1	1	-1	-1	-1	1	1	-1	1.0254449856	1.02544499
2	1	-1	-1	1	-1	-1	-1	1	1	1	-1	-1	-0.379735885	-0.3797359
3	-1	1	-1	-1	1	-1	-1	-1	1	1	1	1	0.9400841447	0.94008414
4	1	1	1	-1	1	-1	-1	1	-1	-1	-1	-1	-0.838928467	-0.8389285
5	-1	-1	-1	1	1	1	-1	1	-1	-1	1	1	-1.213695159	-1.2136952
6	-1	-1	1	1	1	-1	1	-1	-1	1	-1	-1	0.0143435941	0.01434359
7	1	-1	-1	-1	1	1	1	-1	1	-1	-1	-1	-2.100004766	-2.1000048
8	1	1	-1	1	-1	-1	1	-1	-1	-1	1	1	0.3350112573	0.33501126

The response columns, Y and Y Simulated, are initiated with the same simulated values. The values are random values from a *N(0, s)* distribution. where *s* is the RMSE from the power analysis dialog with a default of 1. The Y Simulated values are updated with randomly generated values using the model defined by the parameter values in the Simulate Responses window. The Y column is intended for your true responses after the experiment has be run.

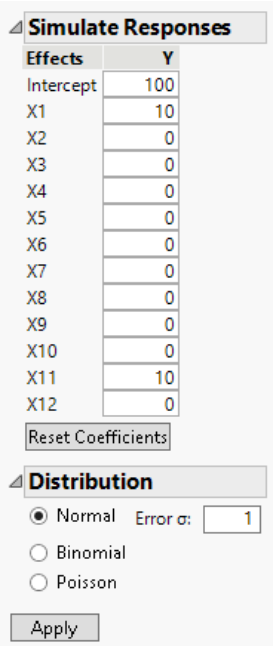
Figure 5.14 Simulate Responses Window



The Simulate Responses window shows default coefficients of 0 for all terms, a Normal distribution selection and Error σ of 1. The values in the Y and Y Simulated column currently reflect only random variation.

13. Change the values of the coefficients in the Simulate Responses window as shown in Figure 5.15.

Figure 5.15 Parameter Values for Simulated Responses



14. Click **Apply**.

The response values in the Y Simulated column change.

Note: The response values are randomly generated. Your values will not match those in Figure 5.16 exactly.

Figure 5.16 Y Simulated Response Column with X1 and X11 Active

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	Y	Y Simulated
1	1	-1	1	-1	-1	1	-1	-1	-1	1	1	-1	-1.663039788	118.978167
2	1	-1	-1	1	-1	-1	-1	1	1	1	-1	-1	0.0739968896	99.9295038
3	-1	1	-1	-1	1	-1	-1	-1	1	1	1	1	0.7497579146	99.5953464
4	1	1	1	-1	1	-1	-1	1	-1	-1	-1	-1	0.2104890444	98.3301026
5	-1	-1	-1	1	1	1	-1	1	-1	-1	1	1	-0.604530523	99.5705285
6	-1	-1	1	1	1	-1	1	-1	-1	1	-1	-1	-0.37357894	79.2516034
7	1	-1	-1	-1	1	1	1	-1	1	-1	-1	-1	-0.033505062	99.3275879
8	1	1	-1	1	-1	-1	1	-1	-1	-1	1	1	1.9285750628	120.254567

In your simulation, you specified X1 and X11 as active factors with large effects relative to the error variation. For this reason, your analysis of the data should identify these two factors as active.

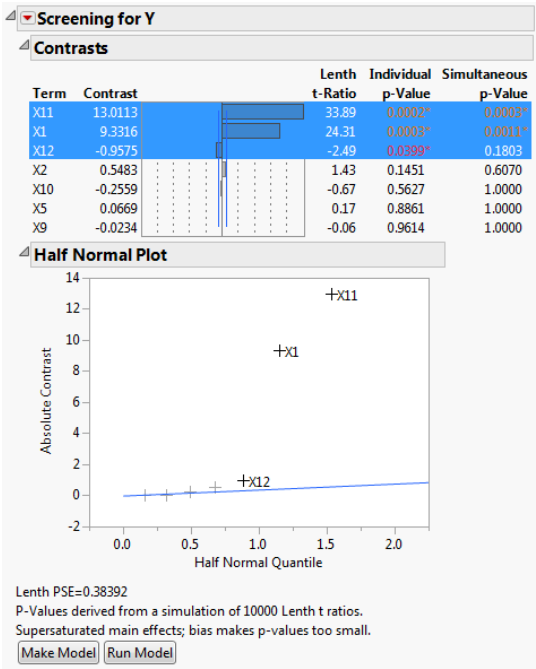
Analyze a Supersaturated Design Using the Screening Platform

The Screening platform provides a way to identify active factors. Use the screening platform to analyze the Y Simulated values in the design table (Figure 5.16). The Screening platform is located under the **DOE > Classical** menu.

Note: Your data table has slightly different Y Simulated values than those in this example. The exact values in your reports will differ from the results shown here.

1. Select **DOE > Classical > Two Level Screening > Fit Two Level Screening**.
2. Select Y Simulated and click **Y**.
3. Select X1 through X12 ,click **X** and click **OK**.

Figure 5.17 Screening Report for Supersaturated Design



The factors X1 and X11 have large contrast and Lenth t-Ratio values. Also, their Simultaneous p-Values are small. In the Half Normal Plot, both X1 and X11 fall far from the line. The Contrasts and the Half Normal Plot reports indicate that X1 and X11 are active. Although X12 has an Individual p-Value less than 0.05, its effect is much smaller than that of X1 and X11.

Because the design is supersaturated, *p*-values might be smaller than they would be in a model where all effects are estimable. This is because effect estimates are biased by other

potentially active main effects. In Figure 5.17, a note directly above the Make Model button warns you of this possibility.

You might also want to check whether the effects that appear active could be highly correlated with other effects. When this occurs, one effect can mask the true significance of another effect. The Color Map in Figure 5.19 displays absolute correlations between effects.

4. Click **Make Model**.

   The constructed model contains only the effects X1, X11, and X12.

5. Click **Run** in the Model Specification window.

Figure 5.18 Parameter Estimates for Model

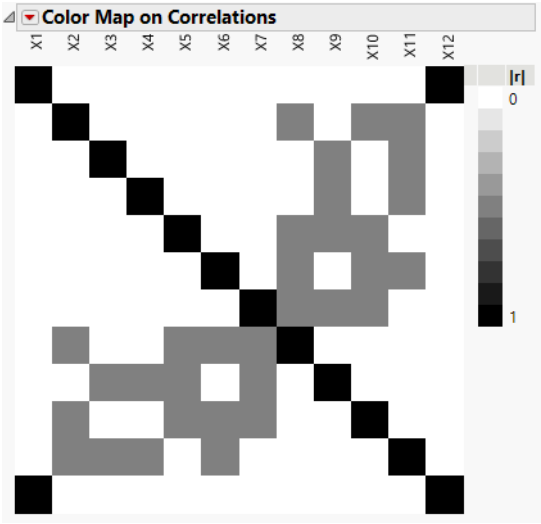
Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	100.12645	0.319502	313.38	<.0001*
X11	11.347787	0.360447	31.48	<.0001*
X1	9.8209584	0.319502	30.74	<.0001*
X12	-1.1329	0.360447	-3.14	0.0347*

Note that the parameter estimates for X11 and X1 are close to the theoretical values that you used to simulate the model. See Figure 5.15, where you specified a model with X1 = 10 and X11 = 10. The significance of the factor X12 is an example of a false positive.

6. In your Custom Design window, open the **Design Evaluation > Color Map on Correlations** outline.

Figure 5.19 Color Map on Correlations Outline



Hover over cells to see the absolute correlations. Notice that X1 has correlations as high as 0.5 with other main effects (X4, X5, X7). (Figure 5.19 uses JMP default colors.)

Analyze a Supersaturated Design Using Stepwise Regression

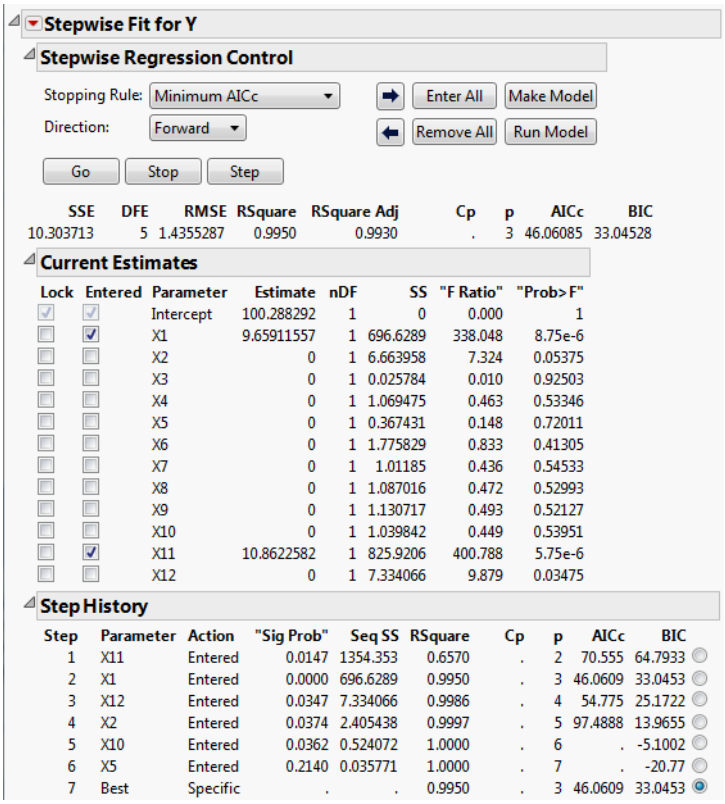
Stepwise regression is another way to identify active factors. The design table in Figure 5.16 contains three scripts. The Model script analyzes your data using stepwise regression in the Fit Model platform.

Note: Your data table has slightly different Y Simulated values than those in this example. The exact values in your reports will differ from the results shown here.

1. In the Table panel of the design table, click the green triangle next to the **Model** script.
2. Change the **Personality** from **Standard Least Squares** to **Stepwise**.
3. Click **Run**.
4. In the Stepwise Fit for Y report, change the **Stopping Rule** to **Minimum AICc**.

   For designed experiments, BIC is typically a more lenient stopping rule than AICc as it tends to allow inactive effects into the model.
5. Click **Go**.

Figure 5.20 Stepwise Regression for Supersaturated Design



Stepwise Fit for Y

Stepwise Regression Control

Stopping Rule: Minimum AICc | Enter All | Make Model
Direction: Forward | Remove All | Run Model
Go | Stop | Step

SSE	DFE	RMSE	RSquare	RSquare Adj	Cp	p	AICc	BIC
10.303713	5	1.4355287	0.9950	0.9930	.	3	46.06085	33.04528

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	100.288292	1	0	0.000	1
☐	☑	X1	9.65911557	1	696.6289	338.048	8.75e-6
☐	☐	X2	0	1	6.663958	7.324	0.05375
☐	☐	X3	0	1	0.025784	0.010	0.92503
☐	☐	X4	0	1	1.069475	0.463	0.53346
☐	☐	X5	0	1	0.367431	0.148	0.72011
☐	☐	X6	0	1	1.775829	0.833	0.41305
☐	☐	X7	0	1	1.01185	0.436	0.54533
☐	☐	X8	0	1	1.087016	0.472	0.52993
☐	☐	X9	0	1	1.130717	0.493	0.52127
☐	☐	X10	0	1	1.039842	0.449	0.53951
☐	☑	X11	10.8622582	1	825.9206	400.788	5.75e-6
☐	☐	X12	0	1	7.334066	9.879	0.03475

Step History

Step	Parameter	Action	"Sig Prob"	Seq SS	RSquare	Cp	p	AICc	BIC
1	X11	Entered	0.0147	1354.353	0.6570	.	2	70.555	64.7933
2	X1	Entered	0.0000	696.6289	0.9950	.	3	46.0609	33.0453
3	X12	Entered	0.0347	7.334066	0.9986	.	4	54.775	25.1722
4	X2	Entered	0.0374	2.405438	0.9997	.	5	97.4888	13.9655
5	X10	Entered	0.0362	0.524072	1.0000	.	6	.	-5.1002
6	X5	Entered	0.2140	0.035771	1.0000	.	7	.	-20.77
7	Best	Specific	.	.	0.9950	.	3	46.0609	33.0453

Figure 5.20 shows that the selected model consists of the two active factors, X1 and X11. The step history appears in the bottom part of the report. Keep in mind that correlations between X1 and X11 and other factors could mask the effects of other active factors (Figure 5.19).

Note: This example defines two large main effects and sets the rest to zero. Real-world situations can be less likely to have such clearly differentiated effects.

Design for Fixed Blocks

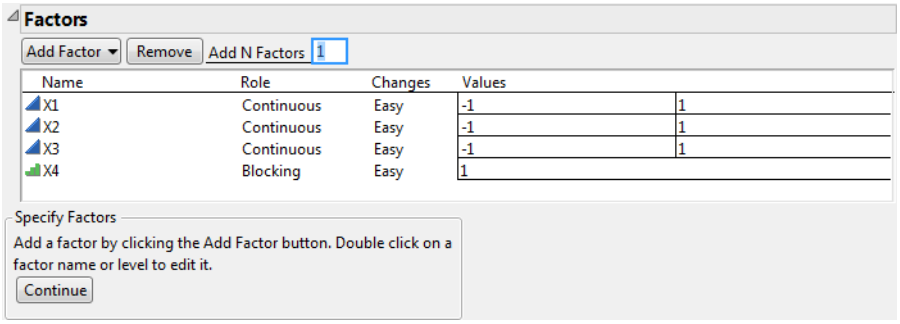
Traditional screening designs require block sizes to be a power of two. However, the Custom Design platform can create designs with fixed blocks of any size.

Suppose that you want to study three factors. You can run only three trials per day and you expect substantial day-to-day variation. Consequently, you need to block your design over multiple days. Also, in this study, you are interested in estimating all two-factor interactions. In this example, you construct a design with three runs per block.

1. Select **DOE > Custom Design**.
2. In the Factors outline, type 3 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Click **Add Factor > Blocking > 3 runs per block**.

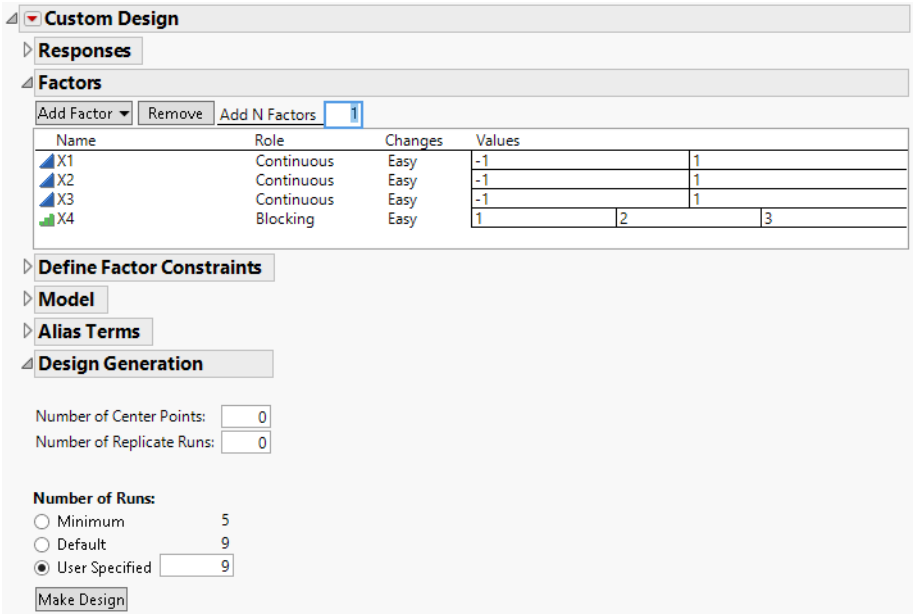
   The blocking factor X4 shows only one level under Values. This is because the run size is unknown at this point.

Figure 5.21 Factors Outline Showing One Block for X4



5. Click **Continue**.

Figure 5.22 Factors Outline Showing Three Blocks for X4

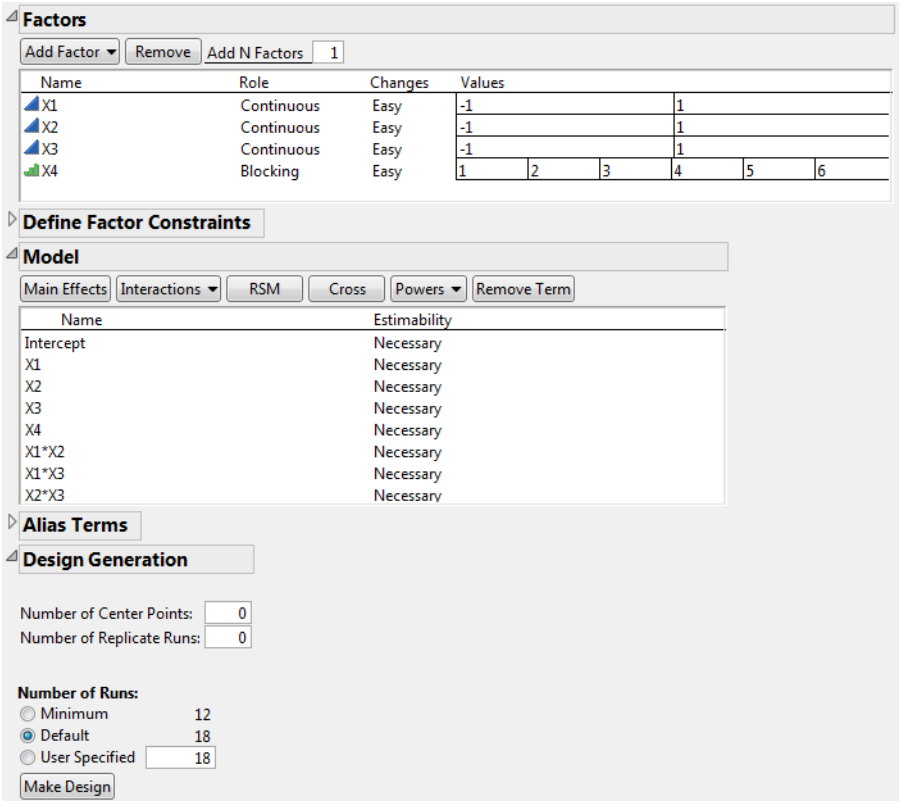


The Factors outline now shows an appropriate number of blocks, calculated as the Default run size divided by the number of runs per block. For this example, the default sample size of 9 requires three blocks. The Factors outline now shows that X4 has three values, indicating the three blocks.

Note: If you specify a different number of runs, the Factors outline updates to show the appropriate number of values for the blocking factor.

6. Select the three continuous factors, X1, X2, and X3, in the Factors outline.
7. In the Model outline, click I**nteractions > 2nd**.

Figure 5.23 Factors Outline Showing Six Blocks for X4



The Number of Runs panel now shows that 18 is the Default run size. The Factors outline now shows six values for X4, indicating six blocks.

Note: Setting the Random Seed in step 8 and Number of Starts in step 9 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

8. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
9. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 5, and click **OK**.
10. Click **Make Design.**

Figure 5.24 Fixed Block Design

Design

Run	X1	X2	X3	X4
1	1	-1	-1	1
2	-1	-1	-1	2
3	1	1	-1	3
4	-1	-1	-1	4
5	1	1	1	5
6	-1	-1	1	6
7	-1	1	-1	1
8	1	1	-1	2
9	-1	1	1	3
10	1	-1	1	4
11	-1	-1	1	5
12	-1	1	-1	6
13	-1	-1	1	1
14	-1	1	1	2
15	1	-1	1	3
16	-1	1	1	4
17	1	-1	-1	5
18	1	1	1	6

In the design, look at the blocking factor, X4. The six blocks are represented. When you conduct your experiment, each day you will run three trials, where X4 = 1 on the first day, X4 = 2 on the second day, and so on. So you want the design table to randomize the trials within blocks. In the Output Options panel, the Randomize within Blocks option is already selected for Run Order.

11. Click **Make Table**.

Figure 5.25 Design Table for Fixed Block Design

	X1	X2	X3	X4	Y
1	1	-1	-1	1	•
2	-1	1	-1	1	•
3	-1	-1	1	1	•
4	-1	1	1	2	•
5	1	1	-1	2	•
6	-1	-1	-1	2	•
7	1	-1	1	3	•
8	-1	1	1	3	•
9	1	1	-1	3	•
10	-1	1	1	4	•
11	1	-1	1	4	•
12	-1	-1	-1	4	•
13	1	-1	-1	5	•
14	-1	-1	1	5	•
15	1	1	1	5	•
16	-1	-1	1	6	•
17	-1	1	-1	6	•
18	1	1	1	6	•

The rows in the design table are grouped by each day's runs. This design enables you to estimate the block effect, all main effects, and two-factor interactions.

Response Surface Experiments

Response surface experiments typically involve a small number (generally 2 to 8) of continuous factors that have been identified as active. The main goal of a response surface experiment is to develop a predictive model of the relationship between the factors and the response. Often, you use the predictive model to find better operating settings for your process. For this reason, your assumed model for a response surface experiment is usually quadratic.

Because a screening design is focused on identifying active effects, a measure of its quality is the size of the relative variance of the coefficients. You want these relative variances to be small. D-optimality addresses these relative variances.

In response surface experiments, the prediction variance over the range of the factors is more important than the variance of the coefficients. The prediction variance over the design space is addressed by I-optimality. An I-optimal design tends to place fewer runs at the extremes of the design space than does a D-optimal design. For more information about D- and I-optimality, see "Optimality Criteria".

By default, Custom Design uses the Recommended option for the Optimality Criterion. Custom Design uses the I-optimality criterion as the Recommended criterion whenever you add quadratic effects using the RSM button in the Model outline. Otherwise, Custom Design uses the D-optimality criterion as the Recommended criterion. See "Optimality Criteria".

- "Response Surface Design"
- "Response Surface Design with Flexible Blocking"
- "Comparison of a D-Optimal and an I-Optimal Response Surface Design"
- "Response Surface Design With Constraints and Categorical Factor"

Response Surface Design

This example contains these sections:

- "Construct a Response Surface Design"
- "Analyze the Experimental Results"

Construct a Response Surface Design

Construct a response surface design for three continuous factors that you have identified as active. You want to find process settings to maintain your response(Y) within specifications. The lower and upper specification limits for Y are 54 and 56, respectively, with a target of 55.

1. Select **DOE > Custom Design**.

2. In the Responses outline, click **Maximize** and select **Match Target**.
3. Type 54 as the **Lower Limit** and 56 as the **Upper Limit**.
4. Leave **Importance** blank.

   Because there is only one response, the Importance value is set to 1 by default.
5. Type 3 next to **Add N Factors**.
6. Click **Add Factor > Continuous**.

   This adds three continuous factors: X1, X2, and X3.
7. Click **Continue**.
8. In the Model outline, click the **RSM** button.

   This adds quadratic and interaction terms to the model. It also sets the value of the Recommended optimality criterion to I-optimality. You can verify this in the Design Diagnostics outline once you click Make Design.

   Leave the Default Number of Runs set to 16.

   Note: Setting the Random Seed in step 9 and Number of Starts in step 10 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.
9. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 929281409, and click **OK**.
10. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 40, and click **OK**.
11. Click **Make Design**.

    The Design outline shows the design.

Figure 5.26 RSM Design

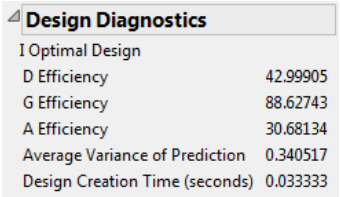
Design

Run	X1	X2	X3
1	0	-1	0
2	1	-1	1
3	0	0	0
4	-1	1	1
5	-1	0	0
6	1	1	1
7	-1	1	-1
8	-1	-1	-1
9	0	0	1
10	1	0	0
11	0	1	0
12	0	0	0
13	1	1	-1
14	0	0	-1
15	-1	-1	1
16	1	-1	-1

In order to estimate quadratic effects, a response surface design uses three levels for each factor. Note that the design in Figure 5.26 is a face-centered Central Composite Design with two center points.

12. Open the **Design Evaluation > Design Diagnostics** outline.

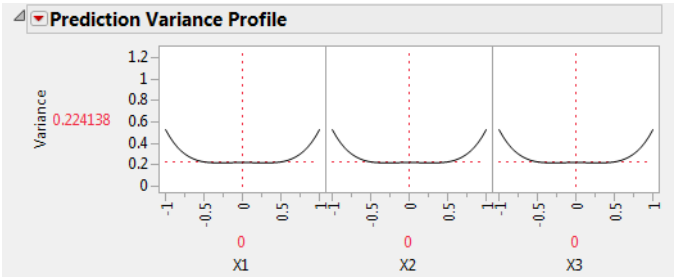
Figure 5.27 Design Diagnostics Outline



The first line in the Design Diagnostics outline identifies the optimality criterion being used. This design is I-optimal.

13. Open the **Design Evaluation > Prediction Variance Profile** outline.

Figure 5.28 Prediction Variance Profile

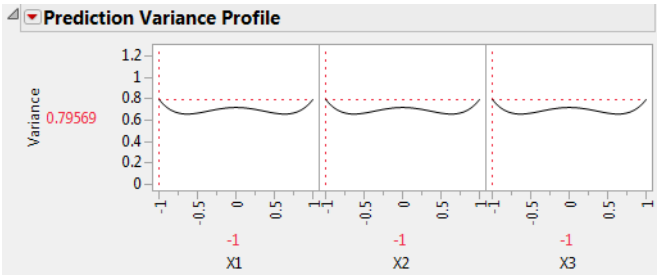


The vertical axis shows the relative prediction variance of the expected value of the response. The *relative prediction variance* is the prediction variance divided by the error variance. When the relative prediction variance is one, its absolute variance equals the error variance of the regression model.

The profiler shows values of the relative prediction variance over the design space. You can move the sliders to explore the prediction variance's behavior. The prediction variance is smallest in the center of the design space. It is fairly constant, with values only slightly larger than 0.2, for factor settings between -0.5 and 0.5. The prediction variance increases as the settings approach the design space boundaries.

14. Click the Prediction Variance Profile red triangle and select **Maximize Variance**.

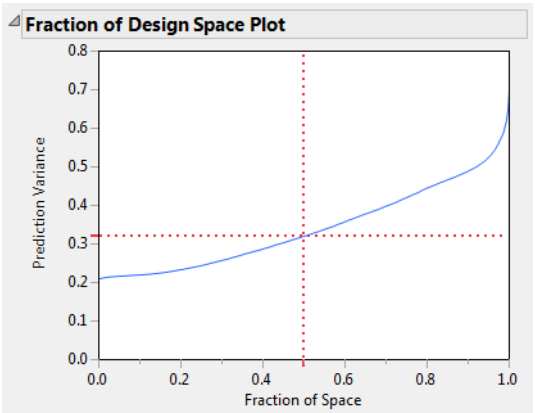
Figure 5.29 Prediction Variance Profile with Relative Variance Maximized



The profiler shows that the maximum value of the relative prediction variance is 0.79569.

15. Open the **Design Evaluation > Fraction of Design Space Plot** outline.

Figure 5.30 Fraction of Design Space Plot



The blue curve in the plot shows the relative prediction variance as a function of the fraction of design space. The red dashed cross hairs indicate that, for 50% of the design space, the prediction variance is about 0.32 or less. Use the cross hair tool to draw other inferences. For example, when the Fraction of Space is 0.95, the Prediction Variance is about 0.52. This means that for 95% of the design space, the relative prediction variance is below 0.52.

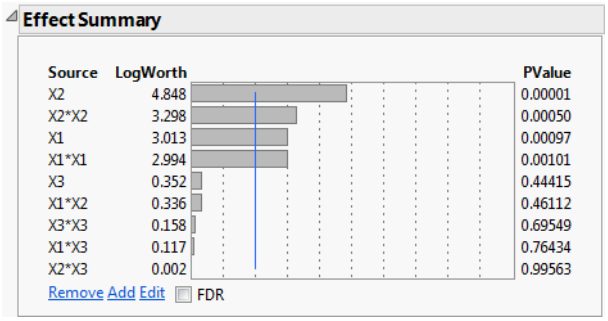
Analyze the Experimental Results

The Custom RSM.jmp sample data table contains the results of the experiment. The Model script opens a Fit Model window showing all of the effects specified in the DOE window's Model outline. This script was saved to the data table by the Custom Design platform.

1. Select **Help > Sample Data Library** and open Design Experiment/Custom RSM.jmp.
2. In the Table panel, click the green triangle next to the **Model** script.
3. Click **Run**.

The Effect Summary report shows the LogWorth and PValue for each effect in the model. The vertical blue line in the plot is set at the value 2. A LogWorth that exceeds 2 is significant at the 0.01 level.

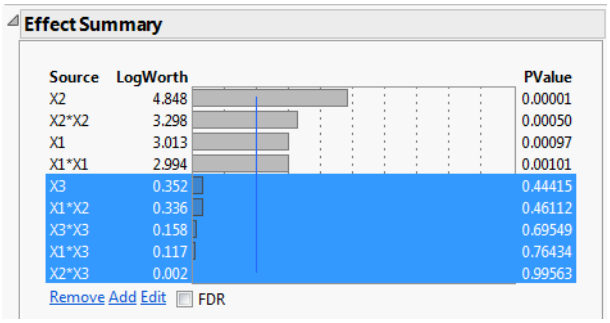
Figure 5.31 Effect Summary Report



The report shows that X1, X2, X1*X1, and X2*X2 are significant at the 0.01 level. None of the other effects are significant at even the 0.10 level. Reduce the model by removing these insignificant effects.

4. In the Effect Summary report, select X3, X1*X2, X3*X3, X1*X3, and X2*X3.

Figure 5.32 Effect Summary Report with Insignificant Effects Selected



5. Click **Remove**.

   The Fit Least Squares report is updated to show a model containing only the significant effects: X1, X2, X1*X1, and X2*X2.

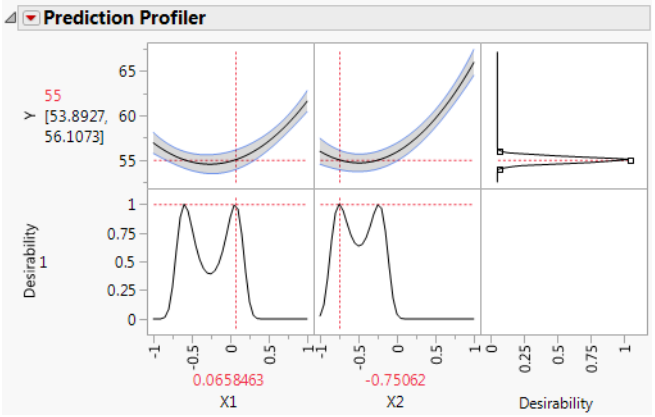
   Use the Prediction Profiler (at the bottom of the Fit Least Squares window) to explore how the predicted response (Y) changes as you vary the factors X1 and X2. Note the quadratic behavior of Y across the values of X1 and X2.

   Remember that you entered response limits for Y in the Responses outline of the Custom Design window. As a result, the Response Limit column property is attached to the Y column in the design table. The Desirability function for Y (in the top plot at right) is based on the information contained in the Response Limit column property. JMP uses this

function to calculate Desirability as a function of the settings of X1 and X2. The traces of the Desirability function appear in the bottom row of plots.

6. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

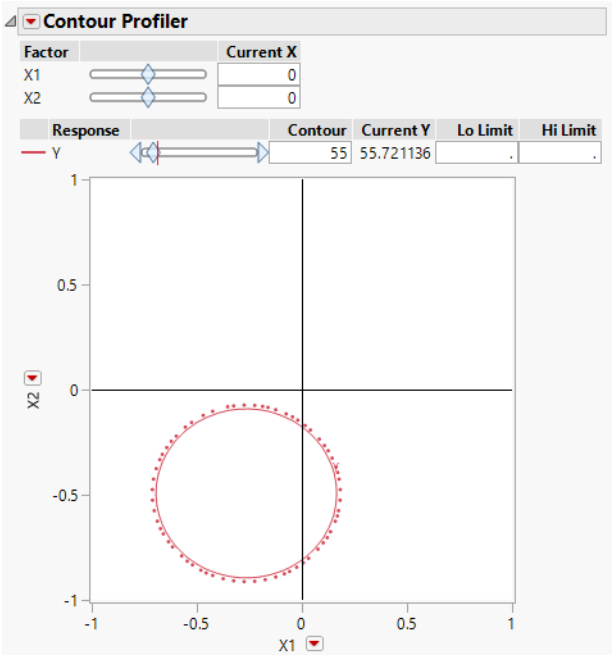
Figure 5.33 Prediction Profiler with Desirability Maximized



The predicted response achieves the target value of 55 at the process settings shown in red above X1 and X2. Figure 5.33 shows that a value of X1 near –0.65 also achieves a predicted value of 55 when X2 = -0.75062. In fact, your Prediction Profiler might show different settings as those that maximize desirability. This is because the predicted response is 55 for many settings of X1 and X2.

7. Click the Response Y red triangle and select **Factor Profiling > Contour Profiler**.
8. In the Contour Profiler report, type 55 as the value for **Contour**.

Figure 5.34 Contour Profiler



The settings of X1 and X2 that correspond to the red contour have predicted response values of 55. You might want to select from among these process settings based on cost efficiency.

Response Surface Design with Flexible Blocking

When optimizing a process, you might need to include qualitative factors in your experiment as well as continuous factors. You might need to block by qualitative factors such as batch or day, or include qualitative factors such as machine or delivery mechanism. But the Response Surface Design platform supports only continuous factors. To obtain a response surface design with a qualitative factor, you can replicate the design over each level of the factor. However, this is inefficient. The Custom Design platform constructs an optimal design with fewer runs.

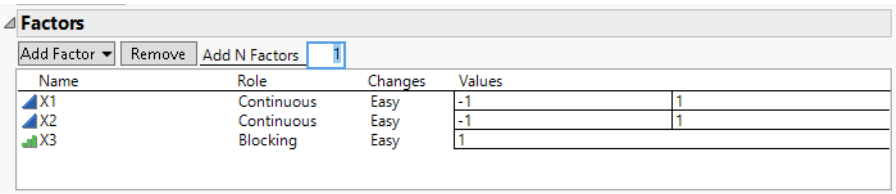
In this example, you construct a response surface design that accommodates two continuous factors and a blocking factor with four runs per block. You can include categorical or discrete numeric factors in a similar fashion.

1. Select **DOE > Custom Design**.
2. Type 2 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.

4. Click **Add Factor** > **Blocking > 4 runs per block**.

   Notice that only one level appears under Values. This is because the number of blocks cannot be determined until the number of runs is determined.

Figure 5.35 Factors Outline with Two Continuous Factors and a Blocking Factor



5. Click **Continue**.

   The Default number of runs is 12. The Factors outline updates to show three levels for the Blocking factor, X3. Because you required X3 to have four runs per block, the 12 runs allow three blocks.

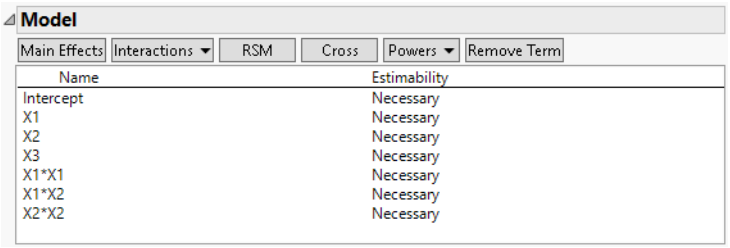
6. Click **RSM**.

   An informational JMP Alert window reminds you that the blocking factor cannot appear in interaction or quadratic terms. JMP adds only the appropriate RSM terms to the list.

7. Click **OK** to dismiss the message.

   Quadratic and interactions terms for X1 and X2 are added to the model. Because you added RSM terms, the Recommended optimality criterion changes from D-Optimal to I-Optimal. You can see this later in the Design Diagnostics outline.

Figure 5.36 Model Outline with Response Surface Effects

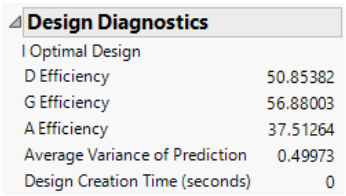


Note: Setting the Random Seed in step 8 and Number of Starts in step 9 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

8. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.

9. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 5, and click **OK**.
10. Click **Make Design**.
11. Open the **Design Evaluation > Design Diagnostics** outline.

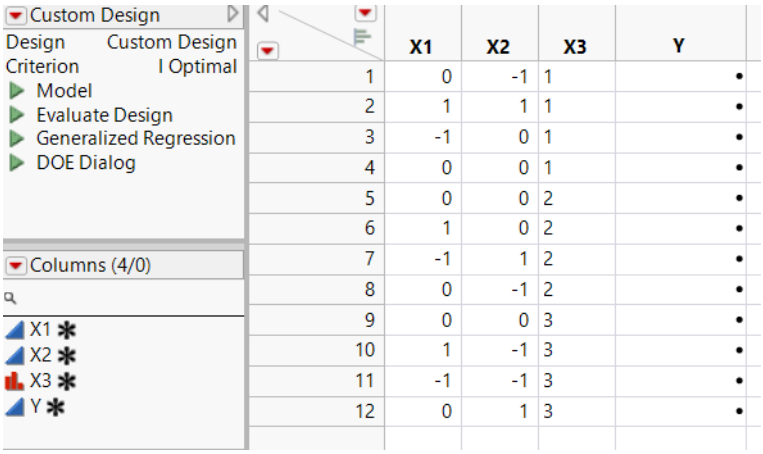
Figure 5.37 Design Diagnostics Outline



The first line in the Design Diagnostics outline identifies the optimality criterion being used. This design is I-optimal.

12. Click **Make Table**.

Figure 5.38 Design Table with Blocking Factor

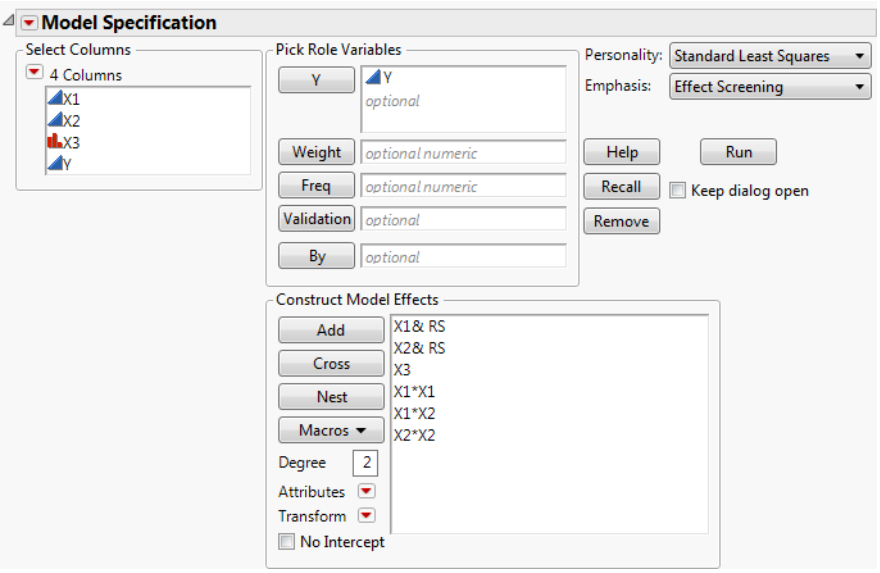


	X1	X2	X3	Y
1	0	-1	1	•
2	1	1	1	•
3	-1	0	1	•
4	0	0	1	•
5	0	0	2	•
6	1	0	2	•
7	-1	1	2	•
8	0	-1	2	•
9	0	0	3	•
10	1	-1	3	•
11	-1	-1	3	•
12	0	1	3	•

Because the default Run Order was Randomize within Blocks, the levels of the blocking factor (X3) are sorted.

13. In the Table panel of the design table, click the green triangle next to the **Model** script.

Figure 5.39 Fit Model Window



Notice the following:

- The blocking factor (X3) is entered as an effect.
- No interactions involving X3 are included.
- The other five effects define a response surface model for X1 and X2.

Comparison of a D-Optimal and an I-Optimal Response Surface Design

In this example, you explore the differences between I-optimality and D-optimality in the context of a two-factor response surface design.

I-Optimal Design

1. Select **DOE > Custom Design**.
2. Type 2 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. Click **Continue**.
5. Click **RSM**.

   Quadratic and interactions terms for X1 and X2 are added to the model. Because you added RSM terms, the Recommended optimality criterion changes from D-Optimal to I-Optimal. You can see this later in the Design Diagnostics outline.

Note: Setting the Random Seed in step 6 and Number of Starts in step 7 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

6. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 383570403, and click **OK**.
7. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 8, and click **OK**.
8. Click **Make Design**.

Figure 5.40 I-Optimal Design

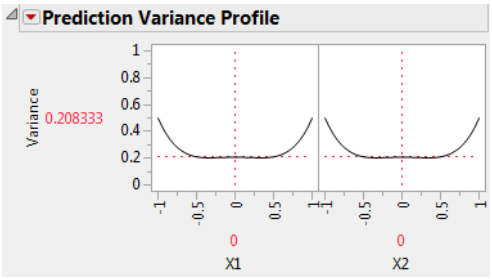
Design

Run	X1	X2
1	0	0
2	1	0
3	-1	-1
4	0	0
5	0	1
6	-1	0
7	0	0
8	-1	1
9	1	-1
10	0	0
11	0	-1
12	1	1

In this I-optimal design, runs 1, 4, 7, and 10 are at the center point (X1 = 0 and X2 = 0). I-optimal designs tend to place more runs in the center (and consequently fewer runs at the extremes) of the design space than do D-optimal designs. You can compare this design to the D-optimal design shown in Figure 5.42.

9. Open the **Design Evaluation** > **Prediction Variance Profile** outline.

Figure 5.41 Prediction Variance Profile for I-Optimal Model



The relative prediction variance of the expected response is smallest in the center of the design space.

10. Open the **Fraction of Design Space Plot** outline.

The Fraction of Design Space Plot appears on the left in Figure 5.44. When the Fraction of Space is 0.95, the vertical coordinate of the blue curve is about 0.5. This means that for about 95% of the design space, the relative prediction variance is below 50% of the error variance.

This Custom Design window contains your I-optimal design. Keep this window open. In the next section, you generate a D-optimal design, and compare the two.

D-Optimal Design

To compare Prediction Variance Profile and Fraction of Design Space plots for the I- and D-optimal designs:

1. In the Custom Design window containing your I-optimal design, click the Custom Design red triangle and select **Save Script to Script Window**.

   A window appears, showing a script that reproduces your work.

2. In this new script window, select **Edit > Run Script**.

   A duplicate Custom Design window appears, but with the Design Evaluation outlines closed.

3. In this new Custom Design window, click **Back**.
4. Click the Custom Design red triangle and select **Optimality Criterion > Make D-Optimal Design**.
5. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 383570403, and click **OK**.
6. Click **Make Design**.

   You current Custom Design window contains your D-optimal design.

Figure 5.42 D-Optimal Design

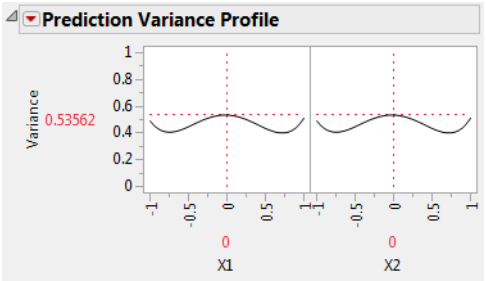
Design

Run	X1	X2
1	0	0
2	1	0
3	-1	-1
4	1	-1
5	0	1
6	-1	0
7	-1	-1
8	-1	1
9	1	-1
10	-1	1
11	0	-1
12	1	1

In this D-optimal design, run 1 is the only run at the center point. D-optimal designs tend to place more runs at the extremes of the design space than do I-optimal designs. Recall that the I-optimal design had four center runs (Figure 5.40).

7. Open the **Design Evaluation** > **Prediction Variance Profile** outline.

Figure 5.43 Prediction Variance Profile for D-Optimal Model



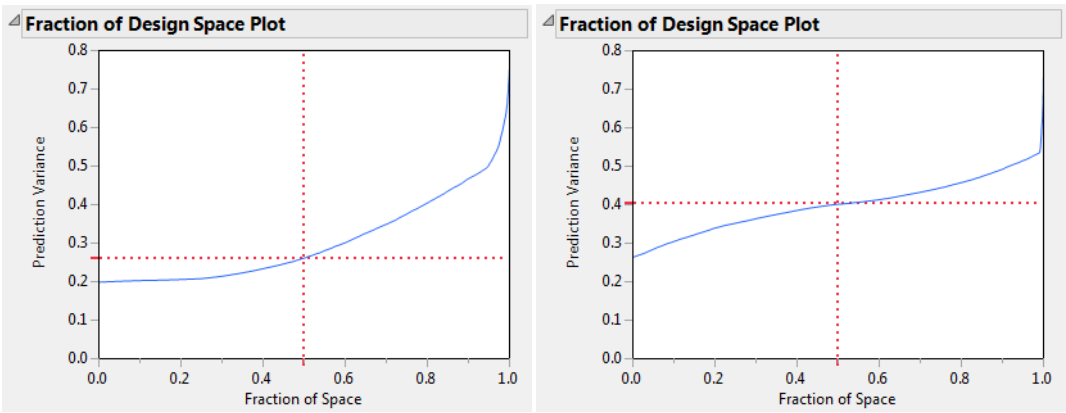
At the center of the design region, the relative prediction variance is 0.53562, as compared to 0.208333 for the I-optimal design (Figure 5.41). This means that the relative standard error is 0.732 for the D-optimal design and 0.456 for the I-optimal design. All else being equal, at the center of the design region, confidence intervals for the expected response based on the D-optimal design are about 60% wider than those based on the I-optimal design.

The Design outline shows that the D-optimal design has nine design points, one for every combination of X1 and X2 set to -1, 0, 1. The D-optimality criterion attempts to keep the relative prediction variance low at each of these design points. Explore the variance at the extremes of the design region by moving the sliders for X1 and X2 to -1 and 1. Note that the variance at these extreme points is usually smaller than the variance for the I-optimal design at these points.

8. Open the **Design Evaluation** > **Fraction of Design Space Plot** outline.

   The Fraction of Design Space Plot appears on the right in Figure 5.44.

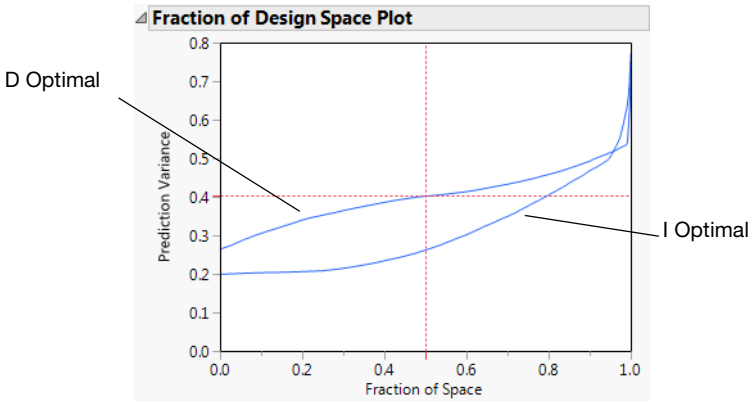
Figure 5.44 Fraction of Design Space Plots (I-Optimal on left, D-Optimal on right)



The red cross-hairs in each plot indicate the maximum prediction variance for 50% of the design space. For 50% of the design space, the prediction variance for the I-optimal design falls below about 0.27. For the D-optimal design, the prediction variance is about 0.4.

9. Right-click in the Fraction of Design Space Plot for your I-optimal design. Select **Edit > Copy Frame Contents**.
10. Right-click in the Fraction of Design Space Plot for your D-optimal design. Select **Edit > Paste Frame Contents**.

Figure 5.45 Fraction of Design Space Plots Superimposed



The variance curve for the I-optimal design is below the curve for the D-optimal design over at least 90% of the design space. This reflects the fact that I-optimality attempts to minimize prediction variance over all of the design space. In contrast, D-optimality focuses on reducing prediction variance at the design points.

Response Surface Design With Constraints and Categorical Factor

In this example, you create a design to optimize the yield of a chemical reaction. Your experimental factors include a categorical factor at three levels, where constraints involve two of the levels. In this example, you will use the Disallowed Combinations Filter to enter the constraints.

Your response is Yield. You have three factors:

- Time: The range of interest is 500 to 560 seconds.
- Temperature: The range of interest is 350 to 750 degrees Kelvin.
- Catalyst: Three catalysts A, B, and C, must be tested.

Your design must allow for constraints on two of the levels of Catalyst:

- When catalyst B is used, the temperature must be above 400.

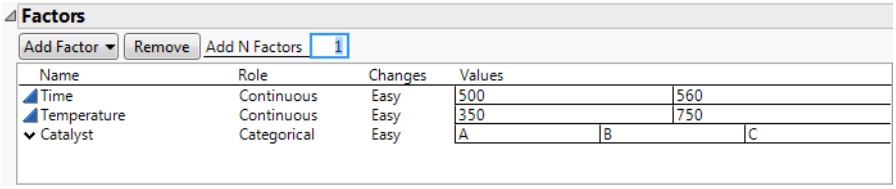
- When catalyst C is used, the temperature must be below 650.

Define the Response and Factors

1. Select **DOE > Custom Design**.
2. In the Response outline, double-click Y and change it to Yield.

   Because your goal is to maximize Yield, leave the **Goal** set to Maximize.
3. In the Factors outline, type 2 next to **Add N Factors**.
4. Click **Add Factor > Continuous**.
5. Click **Add Factor > Categorical > 3 Level**.
6. Rename the factors Time, Temperature, and Catalyst.
7. Change the **Values** for Time to 500 and 560.
8. Change the **Values** for Temperature to 350 and 750.
9. Change the **Values** for Catalyst to A, B, and C.

Figure 5.46 Factor Settings



10. Click **Continue**.

Define the Constraints

1. Select **Use Disallowed Combinations Filter** in the Define Factor Constraints outline.
2. Select Temperature and Catalyst from the **Add Filter Factors** list.
3. Click **Add**.
4. Click 750 in the equation that appears above the Temperature slider and change it to 400. Press Enter (or click elsewhere).

   This disallows factor settings with Temperature values below 400.
5. Click the B block under Catalyst.

   Together with the constraint on Temperature, this disallows factor settings for which Catalyst is B and Temperature is below 400.
6. Click **OR**.

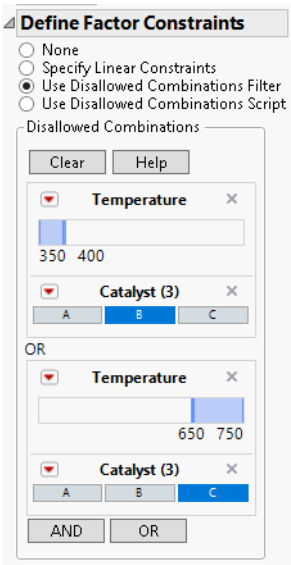
   This allows you to define your second constraint.

7. Select Temperature and Catalyst from the **Add Filter Factors** list. (Your earlier selection may have been retained.)
8. Click **Add**.
9. In the panel that appears beneath the word OR, click 350 in the equation that appears above the Temperature slider and change it to 650. Press Enter (or click elsewhere).

   This disallows factor settings with Temperature values above 650.
10. Click the C block under Catalyst.

    Together with the constraint on Temperature, this disallows factor settings for which Catalyst is C and Temperature is above 650.

Figure 5.47 Constraints Defined



Note: The (3) afterCatalyst indicates the number of catalyst levels.

Add Response Surface Terms and Make Design

1. In the Model outline, select **RSM**.

   A JMP Alert informs you that only quadratic terms for continuous factors are being added to the model.
2. Click **OK** to dismiss the alert.

   JMP adds only the appropriate response surface terms to the model.

Note: Setting the Random Seed in step 3 and Number of Starts in step 4 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

3. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 654321, and click **OK**.
4. (Optional) Click the Custom Design red triangle, select **Number of Starts** and set it to 1000. Click **OK**.
5. Click **Make Design**.

Figure 5.48 Design Satisfying Constraints

Design

Run	Time	Temperature	Catalyst
1	500	750	B
2	500	410	B
3	560	750	A
4	521	750	A
5	542	350	A
6	500	648.7299	C
7	500	350	A
8	533	630	C
9	506	350	C
10	560	530	A
11	560	410	B
12	530	570	B
13	530	550	A
14	530	570	B
15	557	645.1294	C
16	500	570	A
17	560	750	B
18	560	350	C

Because you added RSM terms to the Model outline, this is an I-optimal design. It satisfies your constraints:

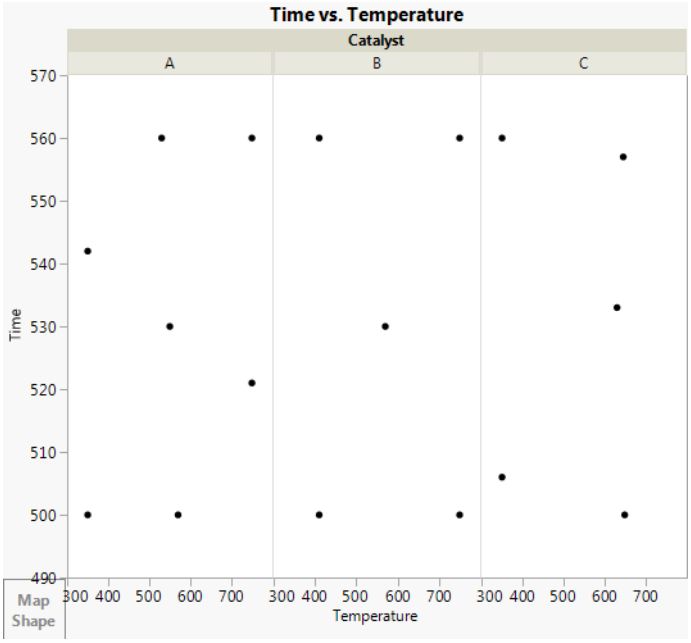
- When catalyst B is used, the temperature must be above 400.
- When catalyst C is used, the temperature must be below 650.

View the Design

Recall that the design region consists of settings of Time between 500 and 560 and Temperature between 350 and 750. Generate a graph to obtain a geometric view of the design points within the design region:

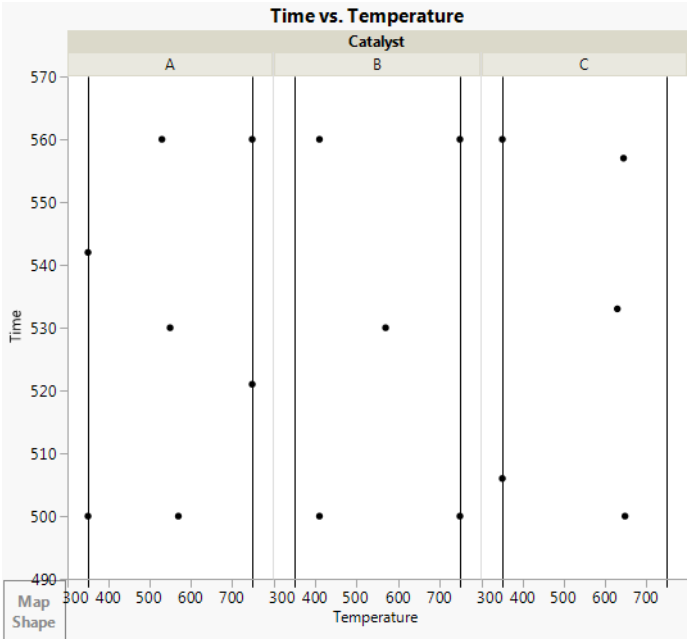
1. Click **Make Table**.
2. Select **Graph > Graph Builder**.
3. Select Time and Temperature and drag them to the center of the template.
4. Deselect the **Smoother** icon. This is the second icon from the left above the template.
5. Select Catalyst and drag it to the **Group X** zone at the top of the template.

Figure 5.49 Design Points for Three Levels of Catalyst



6. Double-click the Temperature axis.
7. Under Reference Lines, add two reference lines:
   - Next to **Value**, enter 350.
   - Click **Add**.
   - Next to **Value**, enter 750.
   - Click **Add**.
8. Click **OK**.

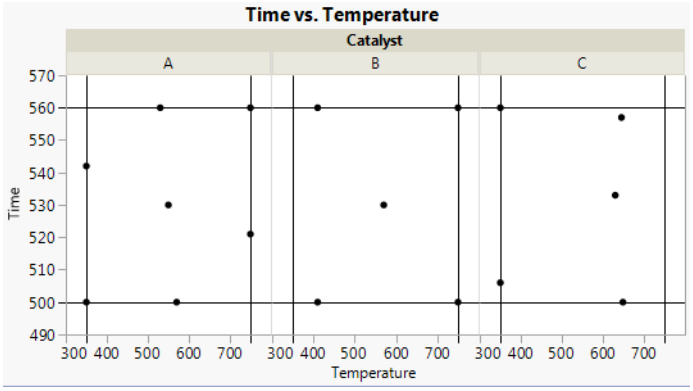
Figure 5.50 Design Bounds on Temperature for Three Levels of Catalyst



9. Double-click the Time axis.
10. Under Reference Lines, add two reference lines:
    - Next to **Value**, enter 500.
    - Click **Add**.
    - Next to **Value**, enter 560.
    - Click **Add**.
11. Click **OK**.
12. Click **Done**.

Note: The plot in Figure 5.51 is vertically resized.

Figure 5.51 Design Regions for Three Levels of Catalyst



Six of the settings for Catalyst A fall on the edges of the design region. All Temperature setting for Catalyst B are above the 400 degree constraint and all Temperature settings for Catalyst C are below the 650 degree constraint.

Mixture Experiments

Both the Custom Design and Mixture Design platforms construct designs for situations where all of your factors are ingredients in a mixture. However, mixture experiments can involve non-mixture process variables, or *process factors*. In addition to mixture only designs, the Custom Design platform can construct designs to accommodate both mixture and process factors. The Custom Design platform also allows the mixture components to sum to any positive number. See the Mixture description in "Factor Types".

- "Mixture Design with Nonmixture Factors"
- "Mixture of Mixtures Design"

Mixture Design with Nonmixture Factors

In this example from Atkinson and Donev (1992), you create a design for an experiment involving both mixture factors and process factors. The design is an 18-run design that is balanced with respect to the levels of a categorical factor. The design enables you to fit a full response surface. You use Design Evaluation plots and results to examine the relative prediction variance of the design.

The design consists of a single response, three mixture factors, and a non-mixture factor:

- The response is Damping, which measures the electromagnetic damping of an acrylonitrile powder.

- The three mixture ingredients are:
  - CuSO4 (copper sulphate), ranging from 0.2 to 0.8
  - Na2S2O3 (sodium thiosulphate), ranging from 0.2 to 0.8
  - Glyoxal (glyoxal), ranging from 0 to 0.6
- The nonmixture environmental factor of interest is Wavelength (the wavelength of an electromagnetic wave) at three levels denoted L1, L2, and L3.

  Wavelength is a continuous variable. However, the researchers were interested only in predictions at three specific wavelengths. For this reason, you treat Wavelength as a categorical factor with three levels.

There are two parts to this example:

- "Create the Design"
- "Evaluate the Design"

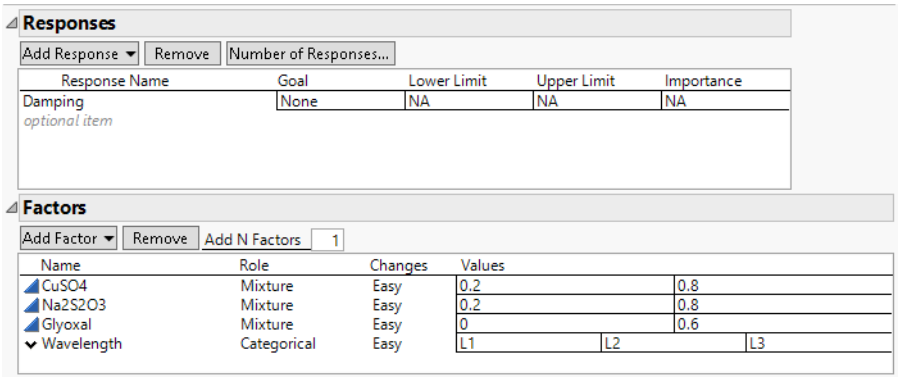
Create the Design

1. Select **DOE > Custom Design**.
2. Double-click Y under Response Name and type Damping.
3. Click **Maximize** under Goal and change it to **None**.

   The goal is set to None because the authors of the study do not mention how much damping is desirable.Select **Help > Sample Data Library** and open Design Experiment/Donev Mixture Factors.jmp.
4. Click the Custom Design red triangle and select **Load Factors**.

   This loads the three mixture ingredients and the categorical environmental factor. Note that the bounds on the values of the three mixture factors are also loaded.

Figure 5.52 Responses Outline and Factors Outline



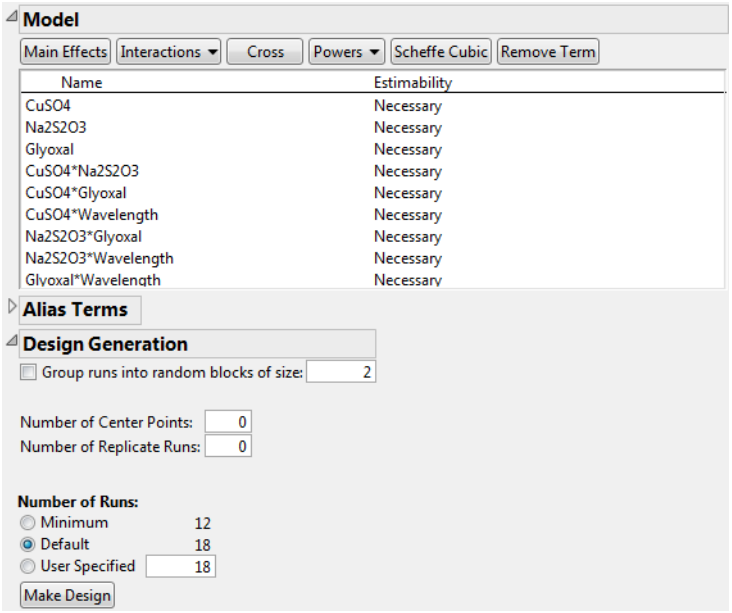
5. In the Model outline, click **Interactions > 2nd**.

An informational JMP Alert window reminds you that JMP removes the main effect terms for non-mixture factors that interact with all the mixture factors. This means that the main effect of Wavelength is removed, but all two-way interactions of mixture factors with Wavelength are added.

6. Click **OK** to dismiss the message.

   The effects in the Model outline define a response surface model in the mixture ingredients along with the additive effect of the wavelength. See Scheffé (1958).

Figure 5.53 Model and Design Generation Outlines



7. Leave the default number of runs at 18.

   The choice of 18 runs allows six runs for each of the three levels of the wavelength factor.

   Note: Setting the Random Seed in step 8 and Number of Starts in step 9 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

8. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
9. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 5, and click **OK**.
10. Click **Make Design**.

Figure 5.54 Design Outline Showing 18-Run Design

Design

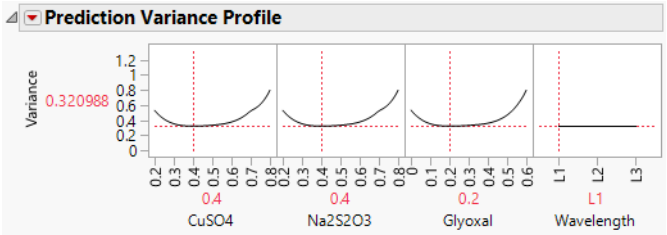
Run	CuSO4	Na2S2O3	Glyoxal	Wavelength
1	0.8	0.2	0	L2
2	0.5	0.2	0.3	L1
3	0.8	0.2	0	L3
4	0.2	0.8	0	L3
5	0.2	0.8	0	L1
6	0.2	0.2	0.6	L1
7	0.2	0.2	0.6	L3
8	0.2	0.5	0.3	L1
9	0.2	0.2	0.6	L2
10	0.5	0.2	0.3	L2
11	0.5	0.5	0	L3
12	0.8	0.2	0	L1
13	0.2	0.8	0	L2
14	0.5	0.2	0.3	L3
15	0.2	0.5	0.3	L2
16	0.5	0.5	0	L1
17	0.2	0.5	0.3	L3
18	0.5	0.5	0	L2

You can check that there are six runs for each level of Wavelength.

Evaluate the Design

1. Open the **Design Evaluation** > **Prediction Variance Profile** outline.

Figure 5.55 Prediction Variance Profile for 18-Run Design



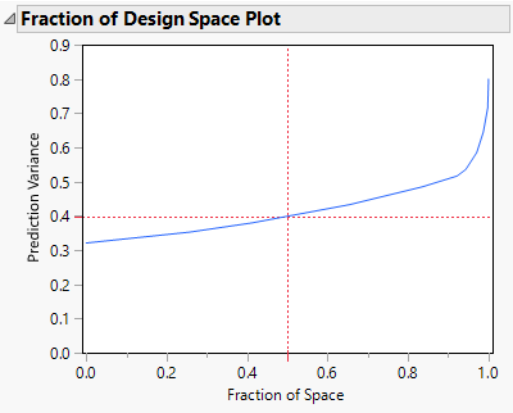
Move the slider for Wavelength to verify that the relative prediction variance profiles for the mixture factors do not change across the levels of Wavelength. Move the slider for any one of the mixture factors. The sliders for the other two mixture factors adjust to make the mixture ingredients sum to one. Notice that the smallest relative prediction variances occur near the center settings for the mixture factors.

2. Click the Prediction Variance Profile red triangle and select **Maximize Variance**.

   Notice that the maximum relative prediction variance over the design space is 0.8 times the error variance.

3. Open the **Fraction of Design Space Plot** outline.

Figure 5.56 Fraction of Design Space Plot for 18-Run Design



Over the entire design space, the relative prediction variance is below 0.8. The minimum relative prediction variance is about 0.32. As seen in Figure 5.55, the minimum occurs near the center settings for the mixture factors.

4. Open the **Design Diagnostics** outline.

Figure 5.57 Design Diagnostics Outline for 18-Run Design

Design Diagnostics

D Optimal Design	
D Efficiency	3.627407
G Efficiency	83.33333
A Efficiency	0.328715
Average Variance of Prediction	0.412864
Design Creation Time (seconds)	0.1

The design is optimal relative to the D-optimality criterion, even though its D-efficiency is very low (3.6%). Because mixture designs are far from orthogonal due to the mixture constraint, they typically have very low D-efficiencies. The Average (relative) Variance of Prediction is 0.412864. This is consistent with the Fraction of Design Space plot in Figure 5.56.

Mixture of Mixtures Design

In this example, construct a design for a mixture of mixtures situation.

Consider the ingredients that go into a cake. Dry ingredients include flour, sugar, and cocoa. Wet ingredients include milk, melted butter, and eggs. The wet and dry components of the cake are two mixtures that are first mixed separately and then blended together. Table 5.1 lists the factors and the ranges over which you vary them as part of your experiment.

Table 5.1 Dry and Wet Components and Experimental Ranges

Mixture	Ingredient	Lower and Upper Levels
Dry	Cocoa	0.1 - 0.2
	Sugar	0 - 0.15
	Flour	0.2 - 0.3
Wet	Butter	0.1 - 0.2
	Milk	0.25 - 0.35
	Eggs	0.05 - 0.20

The dry components (the mixture of cocoa, sugar, and flour) comprise 45% of the combined mixture. The wet components (butter, milk, and eggs) comprise 55%.

Note: With the constraints on the dry and wet ingredients, if you include all 6 factors in the mixture model, a model singularity exists.

The goal of your experiment is to optimize a Taste rating. Taste is rated on a scale of 1 to 10, with 10 representing the best taste.

You construct a 10-run design to fit a main effects model. Because of the constraint on the proportions of dry and wet ingredients, you need to include only five factors in the Model outline to avoid singularity. The choice of which factor not to include is arbitrary.

This example contains two parts:

- "Create the Design"
- "Analyze the Experimental Results"

Create the Design

1. Select **DOE > Custom Design**.
2. Double-click Y under Response Name and type Taste.

   Note that the default goal is Maximize. Because you want to maximize the Taste rating, do not change the goal.
3. Click under Lower Limit and type 0.

   The least desirable rating is 0.
4. Click under Upper Limit and type 10.

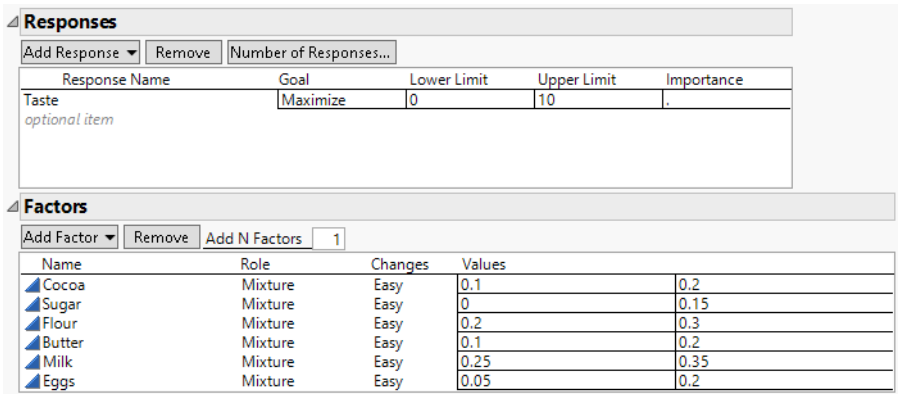
The most desirable rating is 10.

5. Leave the area under Importance blank.

   Because there is only one response, that response is given Importance 1 by default.

6. Select **Help > Sample Data Library** and open Design Experiment/Cake Factors.jmp.
7. Click the Custom Design red triangle and select **Load Factors**.

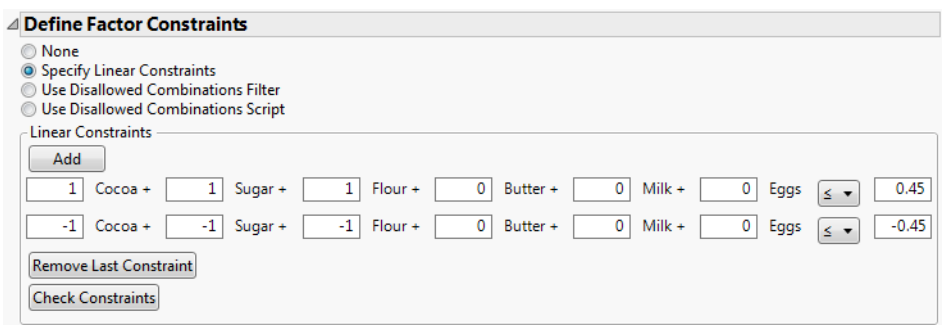
Figure 5.58 Completed Responses and Factors Outlines



Note that the factors are all mixture factors. The Values that define the range of settings for the experiment vary from factor to factor.

8. In the Define Factor Constraints outline, select **Specify Linear Constraints**.
9. In the Linear Constraints panel, click **Add** twice.
10. Enter the constraints shown in Figure 5.59.

Figure 5.59 Define Factor Constraints



Note that the second constraint is equivalent to specifying that the sum of Cocoa, Sugar, and Flour is greater than or equal to 0.45. The two constraints together imply that Cocoa, Sugar, and Flour comprise exactly 45% of the mixture, ensuring that the wet factors constitute the remaining 55%.

11. In the Model outline, select Eggs and click **Remove Term**.

    Because of the equality constraint, a model containing all six effects would be singular.

12. Type 10 next to **User Specified**.

    Your experiment requires baking 10 cakes.

    Note: Setting the Random Seed in step 13 and Number of Starts in step 14 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

13. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
14. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 40, and click **OK**.
15. Click **Make Design**.

    A JMP Alert informs you that your factor constraints include an equality constraint. This was what you intended, because the sum of the dry ingredient proportions is constrained to 45%.

16. Click **OK** to dismiss the JMP Alert.
17. Click **Make Table**.

Figure 5.60 Mixture of Mixtures Design

	Cocoa	Sugar	Flour	Butter	Milk	Eggs	Taste
1	0.1	0.15	0.2	0.2	0.3	0.05	•
2	0.1	0.15	0.2	0.1	0.35	0.1	•
3	0.1	0.15	0.2	0.1	0.25	0.2	•
4	0.1	0.05	0.3	0.2	0.25	0.1	•
5	0.15	0	0.3	0.1	0.35	0.1	•
6	0.1	0.05	0.3	0.15	0.35	0.05	•
7	0.2	0.05	0.2	0.1	0.25	0.2	•
8	0.2	0	0.25	0.2	0.25	0.1	•
9	0.1	0.05	0.3	0.1	0.25	0.2	•
10	0.2	0.05	0.2	0.15	0.35	0.05	•

The settings for the dry ingredients, Cocoa, Sugar, and Flour, sum to 45% of the mixture and the settings for the wet ingredients, Butter, Milk, and Eggs, sum to 55% of the mixture. The settings also conform to the upper and lower limits given in the Factors outline.

Analyze the Experimental Results

The Cake Data.jmp sample data table shows the results of the experiment. The design table contains a Model script that opens a Fit Model window showing the five main effects specified in the DOE window's Model outline. Notice that the main effect of Eggs is not included in the Model outline for this design. The level of Eggs can be determined by the settings of the other 5 factors and the constraint on the dry ingredients. This script was saved to the data table when it was created by Custom Design.

1. Open the Cake Data.jmp sample data table, located in the Design Experiment folder.
2. In the Table panel of the design table, click the green triangle next to the **Model** script.

   The main effect due to Eggs is not included. To avoid model singularities, the Eggs effect was excluded from the Model outline in the Custom Design window. The five effects are designated as Response Surface and Mixture effects.
3. Click **Run**.

   A JMP Alert appears, notifying you that the Profiler cannot be shown because of the mixture of mixtures constraint on the subsets of factors.
4. Click **OK** to dismiss the JMP Alert.

The Parameter Estimates report indicates that Sugar and Flour are significant at the 0.05 level.

Figure 5.61 Parameter Estimates Report

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
(Cocoa-0.1)/0.35	5.1056115	2.360215	2.16	0.0829
Sugar/0.35	16.873989	1.675197	10.07	0.0002*
(Flour-0.2)/0.35	15.750757	2.207353	7.14	0.0008*
(Butter-0.1)/0.35	5.3740398	2.654689	2.02	0.0988
(Milk-0.25)/0.35	1.5973046	2.435648	0.66	0.5409

Experiments with Covariates

Sometimes measurements on the experimental units that are intended for an experiment are available. These measurements might affect the experimental response. It is useful to include these variables, called *covariates*, as design factors. Although you cannot directly control these values, you can ensure that the levels of the other design factors are chosen to yield the most precise estimates of all the effects.

The Custom Design platform constructs a design that selects covariate values in an optimal fashion. Covariate values are selected from an existing data table that provides covariate information about the potential experimental units. You can specify a number of runs that is smaller than the number of experimental units listed in your data table. You can also specify covariates that are hard-to-change. When you make your design and the design has fewer runs than the number of rows in the covariate table, the design table includes a Covariate Row Index column.This column indicates the row from the covariate table that corresponds to each experimental run.

Note: The number of rows in the covariate data table where covariate factors have nonmissing values must be greater than or equal to the specified Number of Runs.

- "Design with Fixed Covariates"
- "Design with Hard-to-Change Covariates"
- "Design with a Linear Time Trend"

Design with Fixed Covariates

In this example, you are interested in modeling the Shrinkage of parts produced by an injection molding process. The Thermoplastic.jmp sample data table in the Design Experiment folder lists 25 batches of raw (thermoplastic) material for potential use in your study. For each batch, material was removed to obtain measurements of Specific Gravity and Tensile Strength. A third covariate, Supplier, is also available.

You want to study the effects of three controllable factors, Temperature (mold temperature), Speed (screw speed), and Time (hold time), on Shrinkage. But you also want to study the effects of the covariates: Specific Gravity, Tensile Strength, and Supplier. Your resources allow for 12 runs.

Create the Design

1. Select **Help > Sample Data Library** and open Design Experiment/Thermoplastic.jmp.
2. Select **DOE > Custom Design**.
3. Double-click Y under Response Name and type Shrinkage.

4. Click **Maximize** under Goal and change it to **Minimize**.
5. Click **Add Factor** and select **Covariate**.
6. Select Specific Gravity, Tensile Strength, and Supplier from the list and click **OK**.

   These are covariates and cannot be controlled.
7. Type 3 next to **Add N Factors**.
8. Click **Add Factor > Continuous**.
9. Rename the three continuous factors Temperature, Speed, and Time.

   These are factors that can be controlled.

Figure 5.62 Responses and Factors Outlines

Responses

Add Response | Remove | Number of Responses...

Response Name	Goal	Lower Limit	Upper Limit	Importance
Shrinkage	Minimize	.	.	.

Factors

Add Factor | Remove | Add N Factors 1

Name	Role	Changes	Values		
Specific Gravity	Covariate	Easy	1.004	1.144	
Tensile Strength	Covariate	Easy	6.42	8.07	
Supplier	Covariate	Easy	A	B	C
Temperature	Continuous	Easy	-1	1	
Speed	Continuous	Easy	-1	1	
Time	Continuous	Easy	-1	1	

10. Click **Continue**.

    The Number of Runs shows the number of rows with covariate values available. You have 25 batches with measured covariates.
11. Type 12 next to **Number of Runs**.

    Note: Setting the Random Seed in step 12 and Number of Starts in step 13 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.
12. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 84951, and click **OK**.
13. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 40, and click **OK**.
14. Click **Make Design**.

Figure 5.63 Twelve-Run Optimal Design

Design

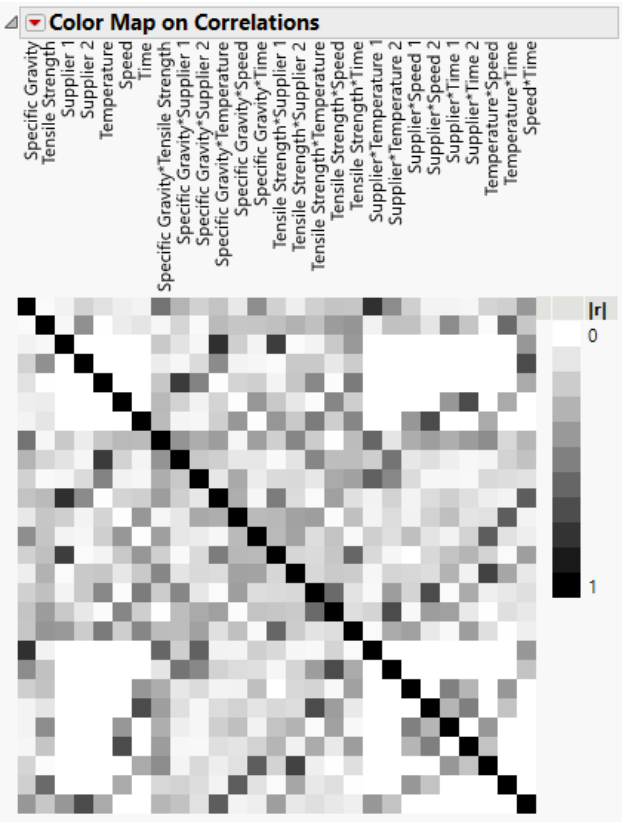
Run	Specific Gravity	Tensile Strength	Supplier	Temperature	Speed	Time
1	1.107	8.07	B	1	1	1
2	1.144	6.82	B	1	-1	-1
3	1.004	6.52	C	1	1	-1
4	1.015	7.46	A	-1	1	-1
5	1.129	6.97	A	1	-1	-1
6	1.139	6.94	C	-1	1	-1
7	1.004	7.12	C	1	-1	1
8	1.067	6.7	B	-1	1	1
9	1.047	6.42	A	-1	-1	1
10	1.03	7.54	B	-1	-1	-1
11	1.094	6.48	A	1	1	1
12	1.113	6.89	C	-1	-1	1

This design is D-optimal, given the potential covariate values. It selects the best sets of covariate values and the best settings for the three controllable factors.

Evaluate the Design

1. Open the **Design Evaluation > Color Map On Correlations** outline.

Figure 5.64 Color Map on Correlations



The seven terms corresponding to main effects appear in the upper left corner of the color map. Notice that these seven terms are close to orthogonal. The largest absolute correlation is between Tensile Strength and Supplier 2. This absolute correlation of about 0.43 is a consequence of the available covariate values. (Figure 5.64 uses JMP default colors.)

Design with Hard-to-Change Covariates

In this example, you construct a design for developing a running shoe for serious runners that has good wear (Wear) properties. Your experiment includes four factors:

- sole thickness (Thickness)
- amount of gel cushioning (Gel)
- outsole material (Outsole)
- midsole material (Midsole)

To obtain generalizable conclusions, you need to test your shoes on a broad base of serious runners. To accommodate your experimental budget, each runner must test several experimental combinations.

Your company has collected data on 100 suitable runners willing to participate in your study. The concomitant variables (*covariates*) measured on these runners are average daily miles run (Miles), weight (Weight), and the foot's strike point (Strike Point).

Create your design:

1. Select **Help > Sample Data Library** and open Design Experiment/Runners Covariates.jmp.
2. Select **DOE > Custom Design**.
3. Double-click Y under Response Name and type Wear.
4. Click **Maximize** under Goal and change it to **Minimize**.
5. Click **Add Factor** and select **Covariate**.
6. Select Miles, Weight, and Strike Point from the list and click **OK**.

   These are the hard-to-change covariates associated with the runners.
7. For one of the factors Miles, Weight, and Strike Point, under Changes, click **Easy** and change it to **Hard**.

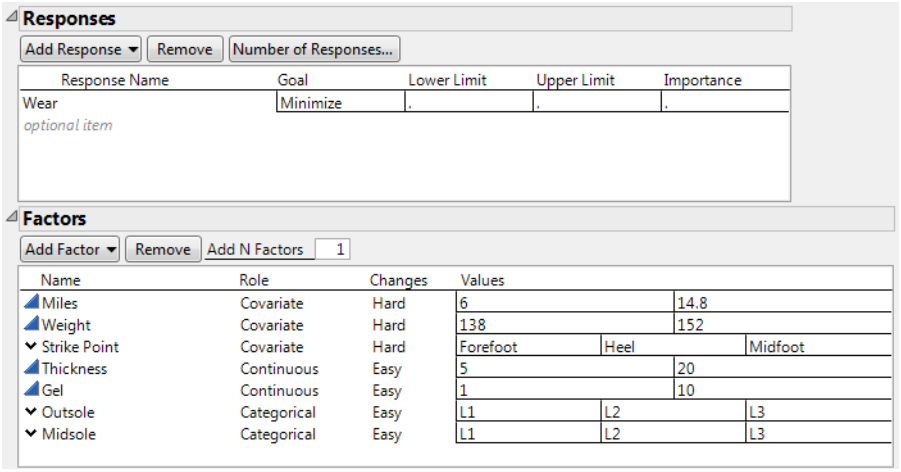
   Note that Changes for all three covariates change to Hard.

To add the remaining factors manually, follow step 8 through step 16. Or, to load factors from a saved table, select **Load Factors** from the Custom Design red triangle. Open the Runners Factors.jmp sample data table, located in the Design Experiment folder. If you select **Load Factors**, skip step 8 through step 16.

8. Type 2 next to **Add N Factors**.
9. Click **Add Factor > Continuous**.
10. Rename the two factors Thickness and Gel.
11. Change the **Values** for Thickness to 5 and 20.
12. Change the **Values** for Gel to 1 and 10.
13. Type 2 next to **Add N Factors**.
14. Click **Add Factor > Categorical > 3 Level.**
15. Rename the two factors Outsole and Midsole.

    Keep the default **Values** for these factors.

Figure 5.65 Responses and Factors Outlines



Responses

Add Response | Remove | Number of Responses...

Response Name	Goal	Lower Limit	Upper Limit	Importance
Wear	Minimize	.	.	.
optional item				

Factors

Add Factor | Remove | Add N Factors 1

Name	Role	Changes	Values		
Miles	Covariate	Hard	6		14.8
Weight	Covariate	Hard	138		152
Strike Point	Covariate	Hard	Forefoot	Heel	Midfoot
Thickness	Continuous	Easy	5		20
Gel	Continuous	Easy	1		10
Outsole	Categorical	Easy	L1	L2	L3
Midsole	Categorical	Easy	L1	L2	L3

16. Click **Continue**.
17. Select **Interactions > 2nd**.

    The specified model fits all two-factor interactions, including covariate by experimental factor interactions.
18. Set the Number of Whole Plots, or runners, to 32 (if it is not already set to that number).
19. Type 64 next to User Specified under Number of Runs (if it is not already set to that number).

    Note: Setting the Random Seed in step 20 and Number of Starts in step 21 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.
20. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
21. (Optional) Click the Custom Design red triangle, select **Number of Starts** and set it to 1 (if it is not already set to that number). Click **OK**.
22. Click **Make Design**.

Figure 5.66 First 20 Runs of Design for Hard-to-Change Covariates

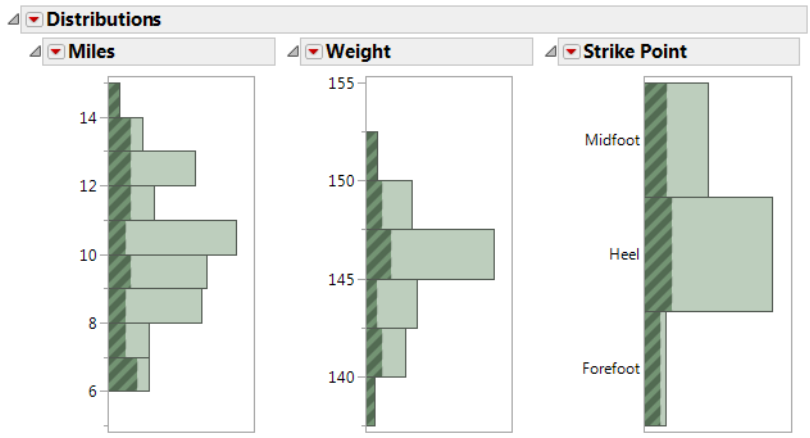
Design

Run	Whole Plots	Miles	Weight	Strike Point	Thickness	Gel	Outsole	Midsole
1	1	7.6	144	Midfoot	5	10	L3	L2
2	1	7.6	144	Midfoot	20	1	L2	L1
3	2	7.6	145	Midfoot	20	10	L3	L3
4	2	7.6	145	Midfoot	5	1	L1	L1
5	3	14.8	141	Midfoot	20	1	L1	L3
6	3	14.8	141	Midfoot	5	10	L2	L1
7	4	6.4	146	Heel	20	10	L1	L2
8	4	6.4	146	Heel	5	1	L2	L3
9	5	11.2	138	Heel	5	1	L2	L3
10	5	11.2	138	Heel	20	10	L3	L2
11	6	11.2	144	Forefoot	20	1	L3	L3
12	6	11.2	144	Forefoot	5	10	L2	L1
13	7	12.8	139	Heel	5	10	L3	L1
14	7	12.8	139	Heel	20	1	L1	L2
15	8	14	139	Midfoot	5	1	L1	L2
16	8	14	139	Midfoot	20	10	L2	L3
17	9	6	150	Heel	20	10	L2	L2
18	9	6	150	Heel	5	1	L3	L3
19	10	10.8	143	Forefoot	20	1	L3	L1
20	10	10.8	143	Forefoot	5	10	L2	L2

Of the 100 runners, 32 are selected based on their covariate values. The rows corresponding to the selected runners are selected in the RunnersCovariates.jmp sample data table. Settings of the experimental factors Thickness, Gel, Insole, and Outsole are determined so that the design is optimal for the model described in the Model Outline.

23. With the RunnersCovariates.jmp sample data table as the active table, select **Analyze > Distribution**.
24. Select all three columns as **Y, Columns**.
25. Check **Histograms Only**.
26. Click **OK**.

Figure 5.67 Histograms for 100 Runners with Selected Runner Data Shaded



The histograms illustrate how the design selected runners to balance runs across the covariates. The distribution of selected runners is similar to the overall distribution of all runners for each covariate.

Design with a Linear Time Trend

Often, experiments conducted in a time sequence experience a linear drift in the response. If you randomize the order of the runs, then the drift's effect does not generally bias the estimated factor effects. However, by accounting for the drift, you can reduce the variance of those effects.

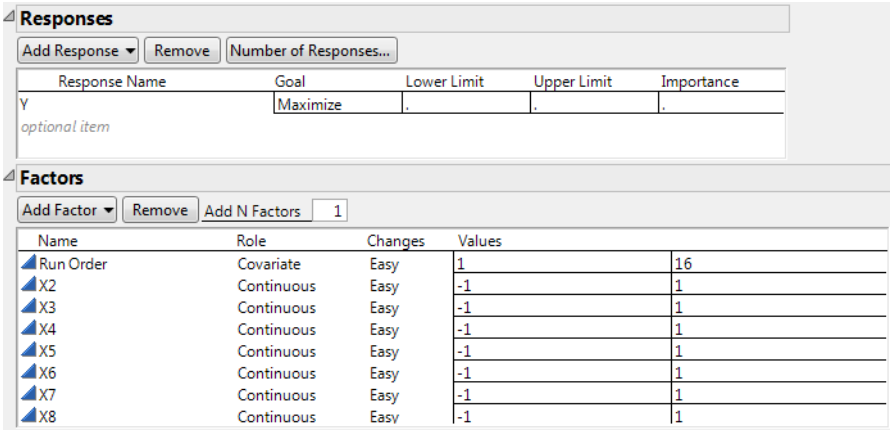
Suppose that there is reason to suspect a strong linear trend in the response over time independent of the factor changes. Then you can construct a design that includes a linear covariate to account for the trend. The resulting design is optimal, given this trend covariate.

In this example, you design an experiment for 7 factors. You construct a 16-run design that is robust to linear trend.

1. Select **File > New > Data Table**.
2. Right-click Column1 and select **Column Info**.
3. Change the column name to Run Order.
4. From the list of **Initialize Data** options, select **Sequence Data**.
5. Type 16 next to **To**.
6. Click **OK**.

   Consecutive integers from 1 to 16 have been entered in the data table.
7. Select **DOE > Custom Design**.
8. Click **Add Factor > Covariate**.
9. Select Run Order and click **OK**.
10. Type 7 next to **Add N Factors**.
11. Click **Add Factor > Continuous**.

Figure 5.68 Responses and Factors Outlines



Responses

Add Response | Remove | Number of Responses...

Response Name	Goal	Lower Limit	Upper Limit	Importance
Y	Maximize	.	.	.
optional item				

Factors

Add Factor | Remove | Add N Factors 1

Name	Role	Changes	Values	
Run Order	Covariate	Easy	1	16
X2	Continuous	Easy	-1	1
X3	Continuous	Easy	-1	1
X4	Continuous	Easy	-1	1
X5	Continuous	Easy	-1	1
X6	Continuous	Easy	-1	1
X7	Continuous	Easy	-1	1
X8	Continuous	Easy	-1	1

12. Click **Continue**.
13. Open the **Alias Terms** outline.
14. Select all of the effects in the list and click **Remove Term**.

    This omits the interaction effects from the correlation color map, leaving only the main effects.

    Note: Setting the Random Seed in step 15 and Number of Starts in step 16 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

15. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 1234, and click **OK**.
16. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 100, and click **OK**.
17. Click **Make Design**.

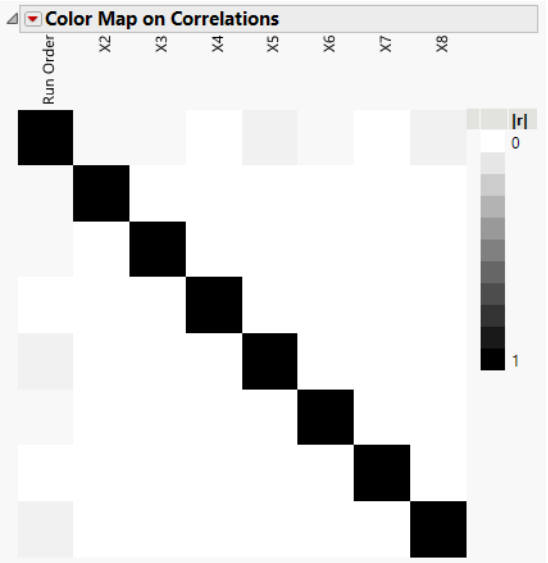
Figure 5.69 Design Outline

Design

Run	Run Order	X2	X3	X4	X5	X6	X7	X8
1	7	-1	-1	1	1	-1	1	1
2	1	-1	1	-1	-1	1	1	-1
3	4	1	1	1	-1	-1	-1	1
4	12	1	1	-1	1	-1	1	-1
5	15	-1	1	-1	-1	-1	-1	1
6	8	1	-1	1	-1	1	1	1
7	11	-1	1	1	-1	-1	1	-1
8	3	1	-1	1	-1	-1	-1	-1
9	5	-1	-1	-1	1	-1	1	1
10	13	-1	-1	1	1	1	-1	-1
11	16	1	1	1	1	1	1	1
12	14	-1	-1	-1	-1	1	-1	1
13	10	1	-1	-1	1	-1	-1	-1
14	9	1	-1	-1	-1	1	1	-1
15	2	1	1	-1	1	1	-1	1
16	6	-1	1	1	1	1	-1	-1

18. Open the **Design Evaluation > Color Map On Correlations** outline.

Figure 5.70 Color Map Showing Absolute Correlations with Run Order



The Color Map shows the following:

- The seven continuous factors, X2 through X8, are orthogonal to each other.
- Run Order, the linear time trend variable, has extremely low absolute correlation with X2 through X8.

19. Open the **Design Evaluation > Estimation Efficiency** outline.

Figure 5.71 Estimation Efficiency Outline

Estimation Efficiency

Term	Fractional Increase in CI Length	Relative Std Error of Estimate
Intercept	0	0.25
Run Order	0.65	0.412
X2	0.006	0.252
X3	4e-4	0.25
X4	0	0.25
X5	0.003	0.251
X6	4e-4	0.25
X7	4e-4	0.25
X8	0.003	0.251

The small absolute correlations of Run Order with X2 through X8 result in very small increases in confidence interval lengths, relative to an ideal orthogonal design. The increases in the lengths of confidence intervals for X2 through X8 are all less than about 0.1%.

In this example, the run order factor is nearly orthogonal to the factor effects. In some cases, your design might have more substantial correlations between the run order factor and other factors. Even in such a situation, including the run order as a factor accounts for any linear trend effect. Including the run order also allows for more precise estimation of the other factor effects.

Experiments with Randomization Restrictions

The Custom Design platform constructs split-plot, split-split-plot, and two-way split-plot (strip-plot) designs that are D-optimal or I-optimal. For more information about constructing these designs, see Goos (2002).

- "Split-Plot Experiment"
- "Two-Way Split-Plot Experiment"

Split-Plot Experiment

Split-plot designs originated in agriculture, but are commonplace in manufacturing and engineering studies. In a split-plot experiment, hard-to-change factors are reset only between one whole plot and the next whole plot. The whole plot is divided into subplots, and the levels of the easy-to-change factors are randomly assigned to each subplot.

The example in this section is adapted from Kowalski, Cornell, and Vining (2002). You are interested in the effects of five factors on the thickness of vinyl that is used to make automobile seat covers. The response and factors in the experiment are described below:

- The response is the thickness of the vinyl that is produced. You want to maximize thickness. A lower limit for thickness values is 10.
- The whole plot factors are the rate of extrusion (extrusion rate) and the temperature (temperature) of drying. These are process variables and are hard to change.
- The subplot factors are three plasticizers whose proportions (m1, m2, and m3) sum to one. These factors are mixture components.

Your experimental budget allows for running 7 settings of these whole plot factors. For each whole plot, you can conduct 4 runs of the subplot factors. This gives you a total of 28 runs.

Create the Design

1. Select **DOE > Custom Design**.
2. Double-click Y under Response Name and type thickness.

   Keep the default goal set to Maximize.
3. Enter a Lower Limit of 10.

To add factors manually, follow step 4 through step 11. Or, to load factors from a saved table, select **Load Factors** from the Custom Design red triangle. Open the Vinyl Factors.jmp sample data table, located in the Design Experiment folder. If you select **Load Factors**, skip step 4 through step 11.

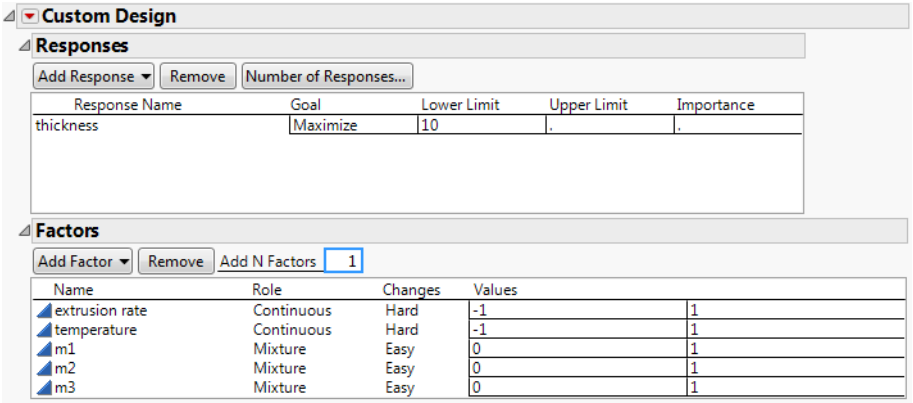
4. Type 2 next to **Add N Factors**.
5. Click **Add Factor > Continuous**.
6. Rename these factors extrusion rate and temperature.

   Keep the default Values of –1 and 1 for these two factors.
7. Click **Easy** and select **Hard** for both extrusion rate and temperature.

   This defines extrusion rate and temperature to be whole plot factors.
8. Type 3 next to **Add N Factors**.
9. Click **Add Factor > Mixture**.
10. Rename the three mixture factors m1, m2, and m3.

    Keep the default Values of 0 and 1 for those three factors.

Figure 5.72 Responses and Factors Outlines



11. Click **Continue**.
12. Click **Interactions > 2nd**.
13. Click **OK** to dismiss the informative message.

    Note that 3 is the default value for the Number of Whole Plots.
14. Type 7 next to **Number of Whole Plots**.
15. Type 28 next to **User Specified**.

    Note: Setting the Random Seed in step 16 and Number of Starts in step 17 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.
16. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
17. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 5, and click **OK**.
18. Click **Make Design**.

Figure 5.73 Design Outline

Design

Run	Whole Plots	extrusion rate	temperature	m1	m2	m3
1	1	-1	-1	0	0	1
2	1	-1	-1	0.6	0	0.4
3	1	-1	-1	0	1	3.55e-15
4	1	-1	-1	0.5	0.5	0
5	2	-1	1	0.4	0	0.6
6	2	-1	1	0	1	0
7	2	-1	1	1	0	0
8	2	-1	1	0	0.5	0.5
9	3	1	1	0	1	0
10	3	1	1	0	0	1
11	3	1	1	0.5	0	0.5
12	3	1	1	1	0	0
13	4	-1	1	0	0	1
14	4	-1	1	0.5	0.5	0
15	4	-1	1	1	0	0
16	4	-1	1	0	1	0
17	5	1	-1	0	0	1
18	5	1	-1	0	1	0
19	5	1	-1	0.5	0	0.5
20	5	1	-1	1	0	0
21	6	-1	-1	0	0.5	0.5
22	6	-1	-1	1	0	0
23	6	-1	-1	0	0	1
24	6	-1	-1	0	1	0
25	7	1	1	0	0	1
26	7	1	1	1	0	0
27	7	1	1	0	0.560358	0.439642
28	7	1	1	0.5	0.5	0

Note that the whole plot factors, extrusion rate and temperature, are reset seven times in accordance with the levels of the factor Whole Plots. Within each level of Whole Plots, the settings for the mixture ingredients, m1, m2, and m3, are assigned at random.

Analyze the Results

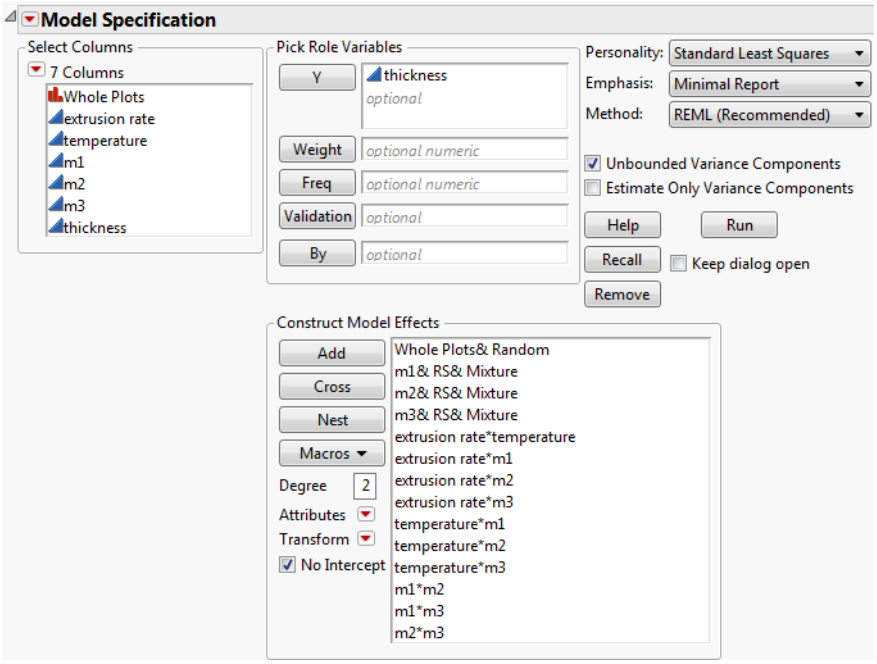
The Vinyl Data.jmp sample data table contains experimental results using a design created in a previous version of JMP.

1. Select **Help > Sample Data Library** and open Design Experiment/Vinyl Data.jmp.

   This sample data table contains 28 runs and response values. The design settings in the table that you created using the Custom Design platform might differ from those used in the Vinyl Data.jmp design.

2. In the Table panel, click the green triangle next to the **Model** script.

Figure 5.74 Fit Model Window



Notice the following in the Fit Model window:

- The factor Whole Plots has the Attribute called Random Effects (**&Random**). This specifies that the levels of Whole Plots are random realizations. They have an associated error term.
- The analysis method is **REML (Recommended)**. This method is specified precisely because the model contains a random effect. For more information about REML models, see *Fitting Linear Models*.

Tip: In the Fit Model window, JMP Pro users can change the Personality to Mixed Model.

3. Click **Run**.

Figure 5.75 Split-Plot Analysis Results

Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|
m1	9.4464782	0.989604	7.99	9.55	<.0001*
m2	5.378571	1.120267	10.35	4.80	0.0007*
m3	5.5546291	1.030955	8.786	5.39	0.0005*
extrusion rate*temperature	-1.866921	0.740748	2.858	-2.52	0.0903
extrusion rate*m1	-0.647611	0.958111	7.261	-0.68	0.5200
extrusion rate*m2	0.2944425	0.997579	8.162	0.30	0.7752
extrusion rate*m3	2.6611621	0.979087	7.733	2.72	0.0272*
temperature*m1	-1.8151	0.961061	7.328	-1.89	0.0990
temperature*m2	1.2044413	0.993319	8.065	1.21	0.2596
temperature*m3	-1.498647	0.97904	7.731	-1.53	0.1657
m1*m2	-6.458845	5.83806	13.36	-1.11	0.2881
m1*m3	8.5532871	5.607183	13.15	1.53	0.1508
m2*m3	10.575333	5.226599	13.04	2.02	0.0640

Random Effect Predictions

REML Variance Component Estimates

Random Effect	Var Ratio	Var Component	Std Error	95% Lower	95% Upper	Pct of Total
Whole Plots	0.6350069	2.476748	2.9711536	-3.346606	8.3001021	38.838
Residual		3.9003483	1.5898101	2.0073487	10.607283	61.162
Total		6.3770963	3.1515558	2.9308168	22.945461	100.000

-2 LogLikelihood = 82.032495402
Note: Total is the sum of the positive variance components.
Total including negative estimates = 6.3770963

The Parameter Estimates report shows that the three mixture ingredients, as well as the extrusion rate*m3 interaction, are significant at the 0.05 level.

The REML Variance Component Estimates report indicates that the variance component associated with Whole Plots is 2.476748. This is 38.838% of the total variation. It follows that the error term associated with whole plot replication is smaller than the residual (or within-plot) error term.

Two-Way Split-Plot Experiment

A two-way split-plot (also known as strip-plot or split-block) design consists of two split-plot components. In industry, these designs arise when batches of material or experimental units from one processing stage pass to a second processing stage. To use a two-way split-plot design, you must be able to reorder the units between stages.

After the first processing stage, you must be able to divide the batches into sub-batches. The second-stage processing factors are applied randomly to these sub-batches. For a specific second-stage experimental setting, all of the sub-batches assigned to that setting can be processed simultaneously. Additional factors can be applied to experimental units after the second processing stage.

In contrast to a split-split-plot design, the second-stage factors are *not nested* within the first-stage factors. After the first stage, the batches are subdivided and formed into new batches. Therefore, both the first- and second-stage factors are applied to whole batches.

Although factors at both stages might be equally hard-to-change, in order to distinguish these factors, JMP denotes the first stage factors as *very-hard*-to-change and the second-stage factors as *hard*-to-change. Additional factors applied to experimental units after the second processing stage are considered *easy*-to-change.

Scenario for a Two-Way Split-Plot Design

This example is based on an experiment to improve the open circuit voltage (OCV) in battery cells (Vivacqua and Bisgaard, 2004). You need to minimize the OCV in order to keep the cells from discharging on their own.

Battery cells move through two stages of processing:

- First stage: A continuous assembly process where batteries are processed in batches of 2000.
- Second stage: A curing process with a 5-day cycle time in a chamber that can accommodate 4000 batteries.

You want to study six two-level continuous factors:

- Four factors (A1, A2, A3, and A4) are applied to the assembly process. You can run 16 trials for the first-stage factors.
- Two factors (C5 and C6) are applied to the curing process. Because curing requires a 5-day cycle time, you can run only 6 cycles (30 days) for the second-stage factors. Using six curing cycles gives you partial replication of the curing settings, enabling you to test for curing effects.

Both the first- and second-stage factors are hard-to-change, suggesting two split-plots. However, the batches of 2,000 batteries from the first-stage experiment can be divided into sub-batches of 500 batteries each. Eight of these sub-batches can be randomly selected and processed simultaneously in the curing chamber.

The experiment has 48 experimental units. Note that the first- and second-stage factors are *crossed*.

Create the Design

To design a two-way split-plot experiment:

1. Select **DOE > Custom Design**.
2. Double-click Y under Response Name and type OCV.
3. Under Goal, click **Maximize** and select **Minimize**.
4. To add factors manually, follow step 5 through step 10. Or, to load factors from a saved table:
    a. Click the Custom Design red triangle and select **Load Factors**.

b. Open the Battery Factors.jmp sample data table, located in the Design Experiment folder.

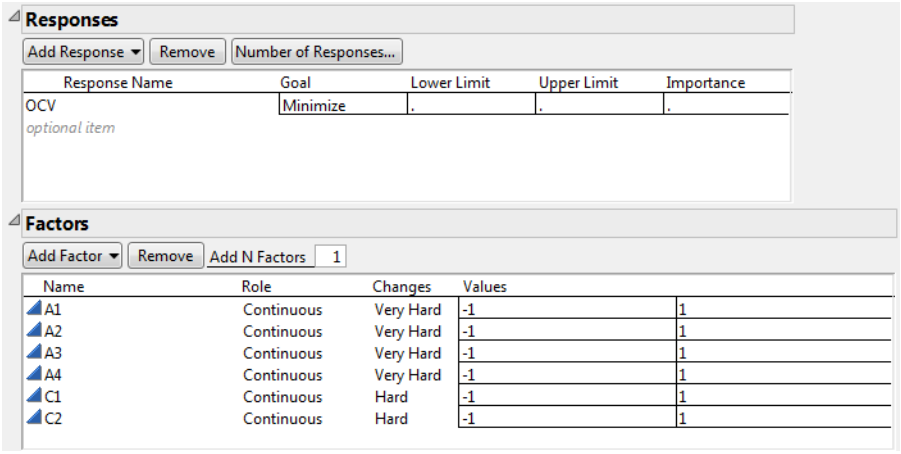
c. Proceed to step 11.

5. Type 6 next to **Add N Factors**.
6. Click **Add Factor > Continuous**.
7. Rename the factors A1, A2, A3, A4, C1, and C2.

   Keep the default Values of -1 and 1 for these factors.
8. For each of the factors A1, A2, A3, and A4, under Changes, click **Easy** and change it to **Very Hard**.

   To distinguish between the first- and second-stage factors, you designate the Changes for the first-stage factors as Very Hard, and the Changes for the second-stage factors as Hard.
9. For each of the factors C1 and C2, under Changes, click **Easy** and change it to **Hard**.

Figure 5.76 Responses and Factors Outlines



10. Click **Continue**.
11. Select **Interactions> 2nd** in the Model outline.
12. In the Design Generation outline, select the option **Hard to change factors can vary independently of Very Hard to change factors**.

    See Figure 5.77. Checking this option creates a two-way split-plot design. If this option is not checked, the design is treated as a split-split-plot design, with nesting of factors at the two levels.
13. Type 16 as the **Number of Whole Plots**.

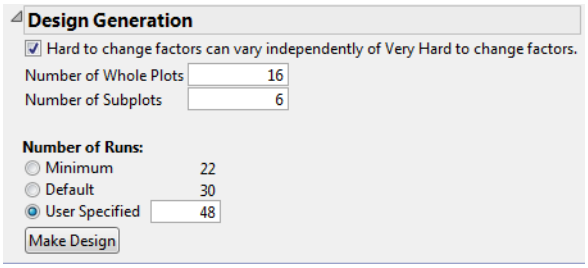
    This is the number of trials that you can run for the first-stage factors.
14. Type 6 as the **Number of Subplots**.

This is the number of trials that you can run for the second-stage factors.

15. Under Number of Runs, type 48 next to **User Specified**.

    This is the total number of experimental units.

Figure 5.77 Design Generation Outline



Note: Setting the Random Seed in step 16 and Number of Starts in step 17 reproduces the exact results shown in this example. In constructing a design on your own, these steps are not necessary.

16. (Optional) Click the Custom Design red triangle, select **Set Random Seed**, type 1866762673, and click **OK**.
17. (Optional) Click the Custom Design red triangle, select **Number of Starts**, type 21, and click **OK**.
18. Click **Make Design**.
19. Click **Make Table**.

Figure 5.78 Partial View of Design Table

	Whole Plots	Subplots	A1	A2	A3	A4	C1	C2	OCV
1	1	1	-1	1	-1	1	-1	1	•
2	1	2	-1	1	-1	1	1	1	•
3	1	3	-1	1	-1	1	1	-1	•
4	2	4	1	1	-1	1	1	1	•
5	2	5	1	1	-1	1	-1	1	•
6	2	6	1	1	-1	1	-1	-1	•
7	3	1	-1	-1	-1	1	-1	1	•
8	3	2	-1	-1	-1	1	1	1	•
9	3	3	-1	-1	-1	1	1	-1	•
10	4	4	-1	-1	-1	-1	1	1	•
11	4	5	-1	-1	-1	-1	-1	1	•
12	4	6	-1	-1	-1	-1	-1	-1	•
13	5	1	1	1	-1	-1	-1	1	•
14	5	2	1	1	-1	-1	1	1	•
15	5	3	1	1	-1	-1	1	-1	•
16	6	4	1	-1	-1	-1	1	1	•

The design table shows 16 levels for Whole Plots. For each level of Whole Plots, the settings of the four assembly factors are constant. From each level of Whole Plots, three batches of 500 batteries (Subplots) are randomly assigned to settings of the curing factors. Two sets of curing conditions are replicated (C1 = -1, C2 = 1 and C1 = 1, C2 = 1). To see this, select columns C1 and C2, right-click in the header area, and select **Sort > Ascending**.

Analyze the Results

The Battery Data.jmp sample data table contains experimental results for the design that you generated.

1. Select **Help > Sample Data Library** and open Design Experiment/Battery Data.jmp.
2. In the Table panel, click the green triangle next to the **Model** script.

   Notice the following in the Fit Model window:

   – The factor Whole Plots has the Attribute called Random Effects (**&Random**). This specifies that the levels of Whole Plots are random realizations. They have an associated error term.

   – The factor Subplots also has the Random Effects Attribute (**&Random**).

   – The analysis Method is **REML (Recommended)**. This method is specified precisely because the model contains random effects. For more information about REML models, see *Fitting Linear Models*.

Tip: In the Fit Model window, JMP Pro users can change the Personality to Mixed Model.

3. Check the option to **Keep dialog open**.
4. Click **Run**.

Figure 5.79 Report for Full Model

Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|
Intercept	38.183196	1.673404	2.075	22.82	0.0016*
A1	-4.657571	1.093074	2.916	-4.26	0.0251*
A2	0.147742	1.093074	2.916	0.14	0.9013
A3	0.9486795	1.093074	2.916	0.87	0.4510
A4	4.0716925	1.190175	3.214	3.42	0.0376*
C1	-1.827487	1.606207	0.868	-1.14	0.4804
C2	-15.73288	1.35446	0.719	-11.62	0.1065
A1*A2	0.1289012	1.080526	2.784	0.12	0.9131
A1*A3	-1.114015	1.080526	2.784	-1.03	0.3837
A1*A4	-1.92398	1.176342	3.04	-1.64	0.1993
A1*C1	-3.271232	0.480531	20.65	-6.81	<.0001*
A1*C2	3.737258	0.478635	20.09	7.81	<.0001*
A2*A3	0.3768179	1.080526	2.784	0.35	0.7520
A2*A4	1.4693535	1.176342	3.04	1.25	0.2992
A2*C1	1.9072313	0.480531	20.65	3.97	0.0007*
A2*C2	-0.161179	0.478635	20.09	-0.34	0.7398
A3*A4	0.0764369	1.176342	3.04	0.06	0.9522
A3*C1	0.1761619	0.480531	20.65	0.37	0.7176
A3*C2	-0.445242	0.478635	20.09	-0.93	0.3633
A4*C1	0.5306856	0.511896	20.84	1.04	0.3118
A4*C2	1.3506119	0.502894	20.12	2.69	0.0142*
C1*C2	-0.425326	1.606207	0.868	-0.26	0.8401

Random Effect Predictions

REML Variance Component Estimates

Random Effect	Var Ratio	Var Component	Std Error	95% Lower	95% Upper	Pct of Total
Whole Plots	1.6095523	15.451509	15.876576	-15.66601	46.569027	45.968
Subplots	0.8919389	8.562507	16.314188	-23.41271	40.537728	25.473
Residual		9.5998804	3.0299487	5.6239301	19.985723	28.559
Total		33.613897	16.824615	15.325965	123.59174	100.000

-2 LogLikelihood = 227.810084

Note: Total is the sum of the positive variance components.

Total including negative estimates = 33.613897

The Parameter Estimates report indicates that four two-way interactions, A1*C1, A1*C2, A2*C1, and A4*C2, and two main effects, A1 and A4, are significant at the 0.05 level.

5. In the Table panel of Battery Data.jmp, click the green triangle next to the **Reduced Model 1** script.

   The script opens a Fit Model window where insignificant interactions have been removed. The remaining effects are all main effects and the four two-way interactions A1*C1, A1*C2, A2*C1, and A4*C2. You are reducing the model in a conservative fashion.

6. Click **Run**.

Figure 5.80 Report for Preliminary Reduced Model

Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|
Intercept	38.249531	1.373127	7.723	27.86	<.0001*
A1	-4.723906	1.175023	8.5	-4.02	0.0034*
A2	0.0497917	1.165418	8.228	0.04	0.9669
A3	0.7560417	1.165418	8.228	0.65	0.5342
A4	4.2328205	1.207654	9.16	3.50	0.0065*
C1	-2.195626	0.866042	1.083	-2.54	0.2234
C2	-15.79922	0.840903	1.008	-18.79	0.0331*
A1*C1	-3.283225	0.452505	25.17	-7.26	<.0001*
A1*C2	3.8035938	0.4498	24.42	8.46	<.0001*
A2*C1	1.8797279	0.452505	25.17	4.15	0.0003*
A4*C2	1.3877006	0.482556	24.95	2.88	0.0081*

Notice that the main effect C2 is now significant at the 0.05 level (Prob>|t| = 0.0331)

7. In the Fit Model window, remove A3.

   The main effect A3 is the only main effect that is not significant and not involved in a two-way interaction.

8. Click **Run**.

Figure 5.81 Report for Reduced Model

Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|
Intercept	38.249531	1.362184	6.639	28.08	<.0001*
A1	-4.723906	1.137313	9.131	-4.15	0.0024*
A2	0.0497917	1.127389	8.818	0.04	0.9658
A4	4.2131272	1.173862	9.921	3.59	0.0050*
C1	-2.166553	0.899138	1.037	-2.41	0.2432
C2	-15.79922	0.874257	0.963	-18.07	0.0389*
A1*C1	-3.282201	0.452234	25.24	-7.26	<.0001*
A1*C2	3.8035938	0.449735	24.43	8.46	<.0001*
A2*C1	1.8843473	0.452234	25.24	4.17	0.0003*
A4*C2	1.3851686	0.4823	24.91	2.87	0.0082*

Random Effect Predictions

REML Variance Component Estimates

Random Effect	Var Ratio	Var Component	Std Error	95% Lower	95% Upper	Pct of Total
Whole Plots	2.0231563	17.459497	9.7154555	-1.582446	36.50144	60.025
Subplots	0.3473621	2.9976761	5.8732929	-8.513766	14.509119	10.306
Residual		8.6298312	2.4692519	5.2816799	16.588508	29.669
Total		29.087005	9.1941655	17.028461	60.633509	100.000

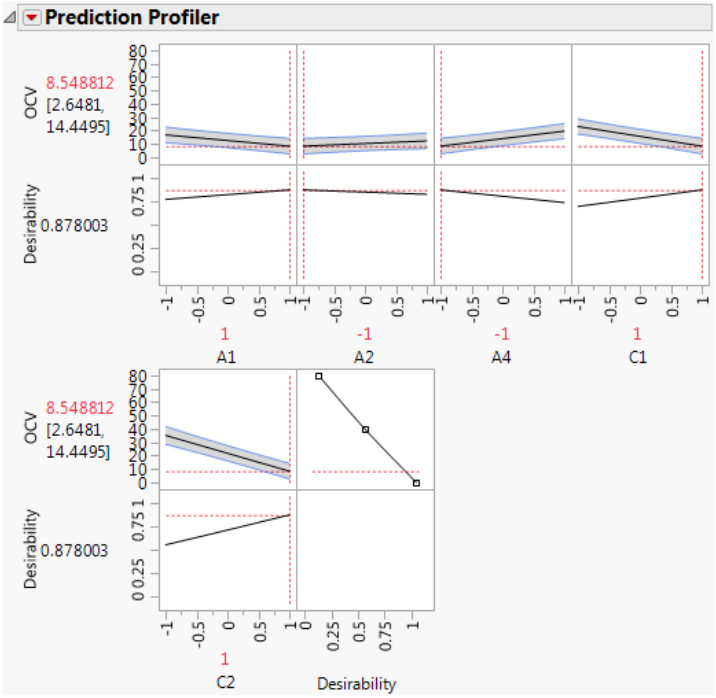
-2 LogLikelihood = 254.17861349
Note: Total is the sum of the positive variance components.
Total including negative estimates = 29.087005

The REML Variance Component Estimates report shows that the variance component associated with Whole Plots is about six times as large as the variance component for Subplots. This suggests that the assembly process is more variable than the curing process. Also, the within (Residual) error is larger than that for Subplots. Efforts to reduce variation should focus on the assembly process and on battery-to-battery differences.

9. Click the Response OCV red triangle and select **Factor Profiling > Profiler**.

10. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 5.82 Prediction Profiler with Settings That Minimize OCV



The profiler shows the five factors identified as active and settings that minimize OCV.

Experiments for Robust Process and Product Design

Often processes or products are impacted by noise factors. These are factors that are not easy or cost effective to control. Finding process or product settings of controllable factors that are least impacted by noise factors is desirable. Historically, Taguchi designs provided a method for experimenting in the presence of noise factors. Note that noise factors must be controlled during an experiment.

Alternatives to Taguchi designs include combined arrays and mixed resolution designs (Borror and Montgomery, 2000). The mixed resolution designs are used to obtain designs that contain both control and noise factors. The goal of such a design is to find robust settings for control variables. The key is the ability to estimate specific effects. In particular, one is interested in estimating main effects of the control variables, main effects of the noise variables, control by control interactions, noise by control interactions, and quadratic terms for the control variables.

Use the custom designer to generate a design for studying robust process settings.

In this example, you are interested in finding optimal settings for four control settings in the presence of three noise factors.

1. Select **DOE > Custom Design**.
2. In the Factors outline, type 4 next to **Add N Factors**.
3. Click **Add Factor > Continuous**.
4. In the Factors outline, type 3 next to **Add N Factors**.
5. Click **Add Factor > Continuous**.
6. Double-click X5 and change it to Z1 to designate a noise factor. Repeat to change X6 and X7 to Z2 and Z3.

Figure 5.83 Factor Outline for Design with 4 Control and 3 Noise Factors

Factors

Add Factor | Remove | Add N Factors 1

Name	Role	Changes	Values	
X1	Continuous	Easy	-1	1
X2	Continuous	Easy	-1	1
X3	Continuous	Easy	-1	1
X4	Continuous	Easy	-1	1
Z1	Continuous	Easy	-1	1
Z2	Continuous	Easy	-1	1
Z3	Continuous	Easy	-1	1

7. Click Continue.

Next you set up the model that for exploring optimal settings for the control factors that are least impacted by the noise factors.

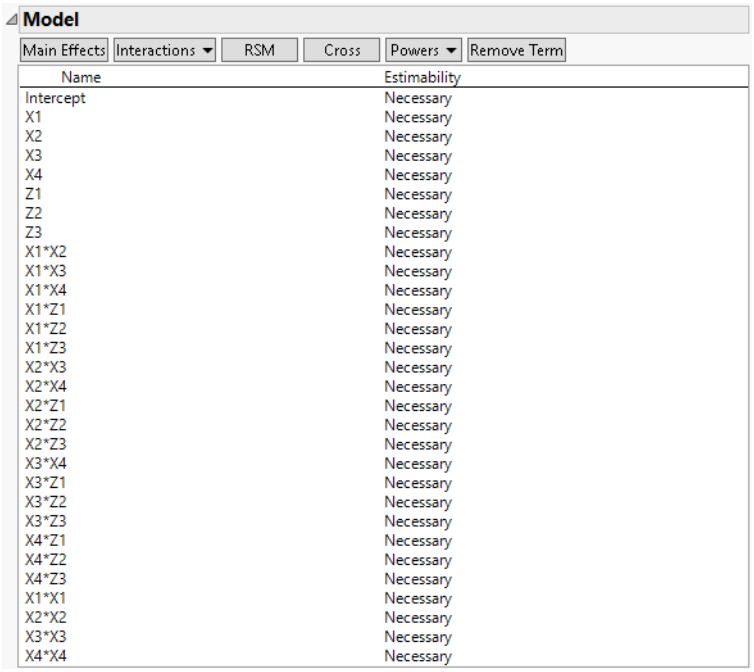
8. The model contains all main effects by default.

9. Click **Interactions > 2nd** to add all two-way interactions.

   This enters the control by control interactions, the control by noise interactions, and the noise by noise interactions. You do not want to estimate the interaction between noise factors.

10. Scroll to the bottom of the list, highlight Z1*Z2, Z1*Z3, and Z2*Z3 and click Remove Term.This removes the noise by noise interactions.
11. In the Factors outline, select X1, X2, X3, and X4.
12. In the Model outline, click Powers > 2nd to add the control variable quadratic terms.

Figure 5.84 Model Outline for Design with Noise Factors



13. Add two center points by entering 2 In the **Number of Center Points:** text box
14. Click **Make Design**.

The result is a 36 run design that enables you to evaluate and optimize your control factors in the presence of your three noise factors. The goal of your analysis is to determine which effects are important and then to select settings for the control factors that are least impacted by the noise factors.

Chapter 6

Augment Designs

Add Additional Runs to an Existing Design

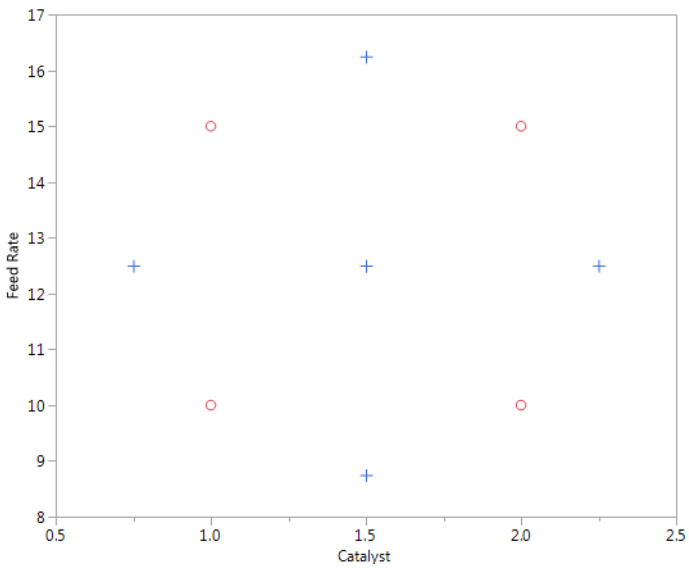
Use the Augment Design platform to add runs to an existing experimental design. Experimentation is often a sequential process and augmenting a designed experiment can help resolve ambiguities that result from a single design.

For an existing design table, the Augment Design platform constructs additional runs that optimize the overall design. You can add runs to accomplish the following objectives:

- Replicate the design a specified number of times.
- Add center points.
- Create a foldover design.
- Add axial points together with center points to transform a screening design to a response surface design.
- Add space filling points to a design.
- Add runs to the design in order to fit a specific model.

For more information about augmenting designs see Goos and Jones (2011).

Figure 6.1 Original Design Runs (+) with Augmented Runs (o)



Contents

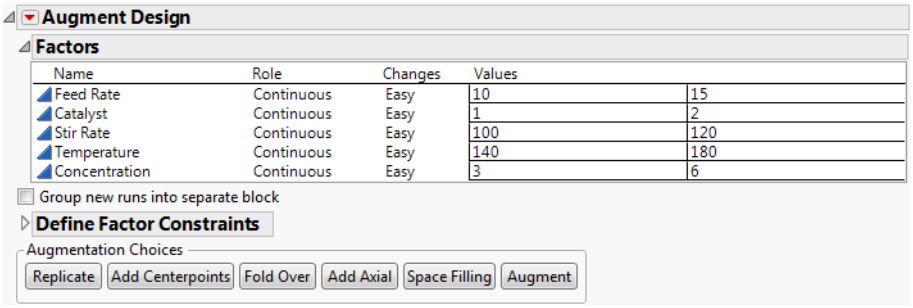
Example Using the Augment Design Platform 215
Launch the Augment Design Platform 221
Augment Design Window 222
Factors 222
Define Factor Constraints 223
Augmentation Choices 226
Augment Design Platform Options 227
Additional Examples of Augmentation Choices 231
Replicate a Design 231
Add Center Points to a Design 233
Create a Foldover Design 233
Add Axial Points to a Design 234
Space Filling Augmentation 235
Statistical Details 237

Example Using the Augment Design Platform

Use the Augment Design Platform to add additional runs to a design. This example demonstrates how to use the Augment Design platform to resolve ambiguities in the results of a screening design. In this study, a chemical engineer investigates the effects of five factors on the percent reaction of a chemical process. The original design was an 8-run design suitable for fitting main effects. This example augments the 8-run design such that all two-factor interactions can be estimated using a total of 16 runs.

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 8 Runs.jmp.
2. Select **DOE > Augment Design**.
3. Select Percent Reacted and click **Y, Response**.
4. Select Feed Rate, Catalyst, Stir Rate, Temperature, and Concentration, and click **X, Factor**.
5. Click **OK**.

Figure 6.2 Factors for the Reactor Example

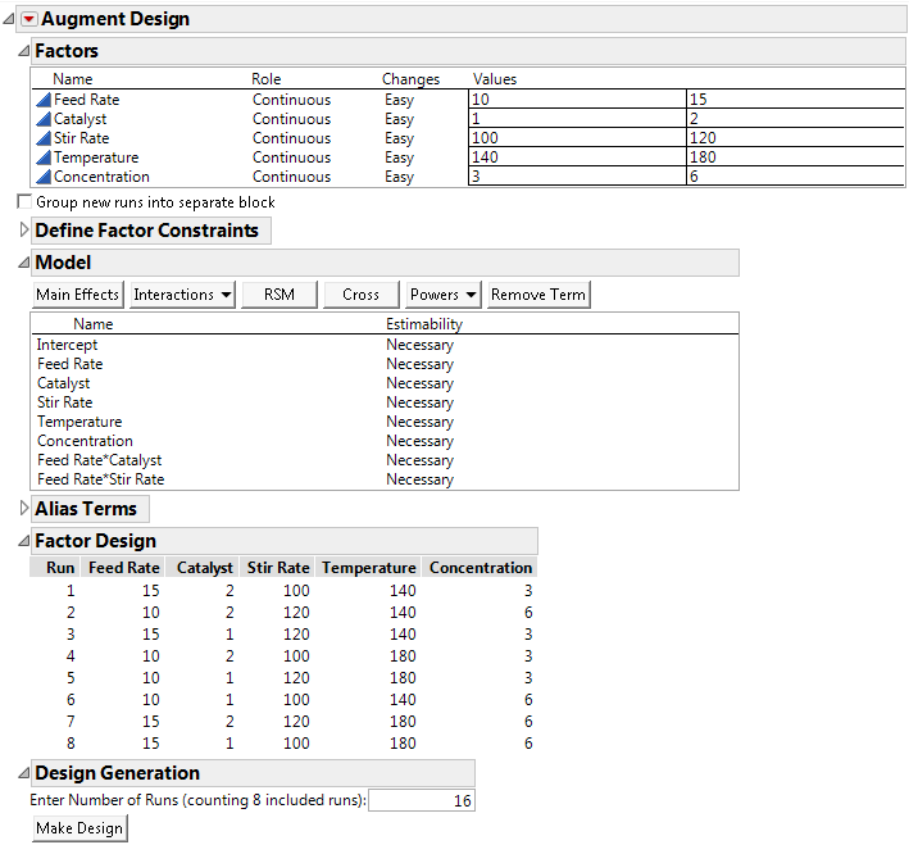


Note: You can select **Group new runs into separate block** to add a blocking factor to any design. However, recall that this example estimates all two-factor interactions in 16 runs, and that cannot be done when there is an additional blocking factor in the model.

6. Click **Augment**.

   The model shown in Figure 6.3 is defined using the Model script in the data table. With 8 runs, you can estimate up to 7 effects. The model in the script contains all 5 main effects and two interaction terms. All terms are defined as necessary for the model. For more information about effect estimability, see "Model".

Figure 6.3 Initial Augment Design Platform



7. In the Model outline, select **Interactions > 2nd**.

   This adds all the two-factor interactions to the model. The minimum number of runs given for the specified model is 16, as shown in the Design Generation text edit box.

   Note: Setting the Random Seed in step 8 and Number of Starts in step 9 reproduces the exact results shown in Figure 6.4. When you are constructing a design on your own, these steps are optional.

8. (Optional) Click the Augment Design red triangle, select **Set Random Seed**, type 282322901, and click **OK**.
9. (Optional) Click the Augment Design red triangle, select **Number of Starts**, type 800, and click **OK**.
10. Click **Make Design**.
11. Click the **Design** disclosure icon to view the design.

Figure 6.4 16 Run Augmented Design

Design

Run	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration
1	15	2	100	140	3
2	10	2	120	140	6
3	15	1	120	140	3
4	10	2	100	180	3
5	10	1	120	180	3
6	10	1	100	140	6
7	15	2	120	180	6
8	15	1	100	180	6
9	15	1	120	180	6
10	10	2	100	140	6
11	10	1	100	180	6
12	15	2	120	140	6
13	15	2	120	180	3
14	10	2	120	180	3
15	10	1	100	140	3
16	10	1	120	140	3

12. Click **Make Table** to generate a design table containing the original design with results and the augmented runs.

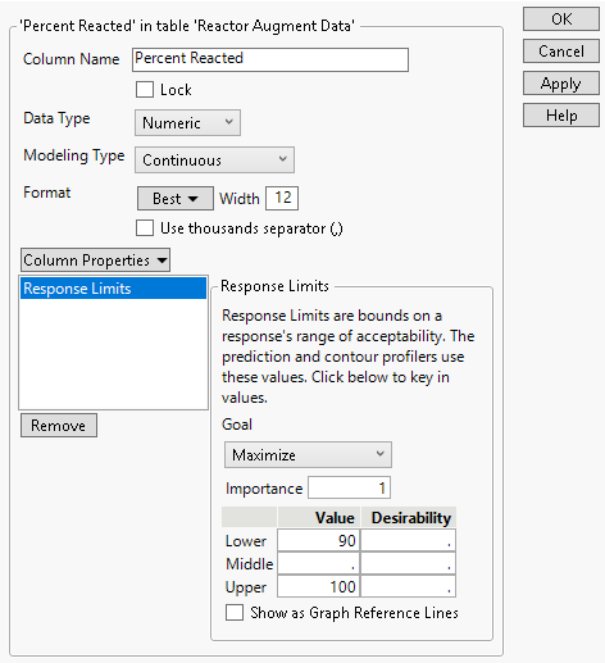
Analyze the Augmented Design

Continuing the augmented design example, suppose you have conducted the additional experimental runs and have recorded results to analyze.

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor Augment Data.jmp.

   You want to maximize Percent Reacted. Note that the column's Response Limits column property in this sample data table is set to **Maximize**. The lower and upper limits are set to 90 and 100, respectively.

Figure 6.5 Response Limits Column Property for the Percent Reacted Column



2. In the Table panel of the data table, click the green triangle next to the **Model** script.

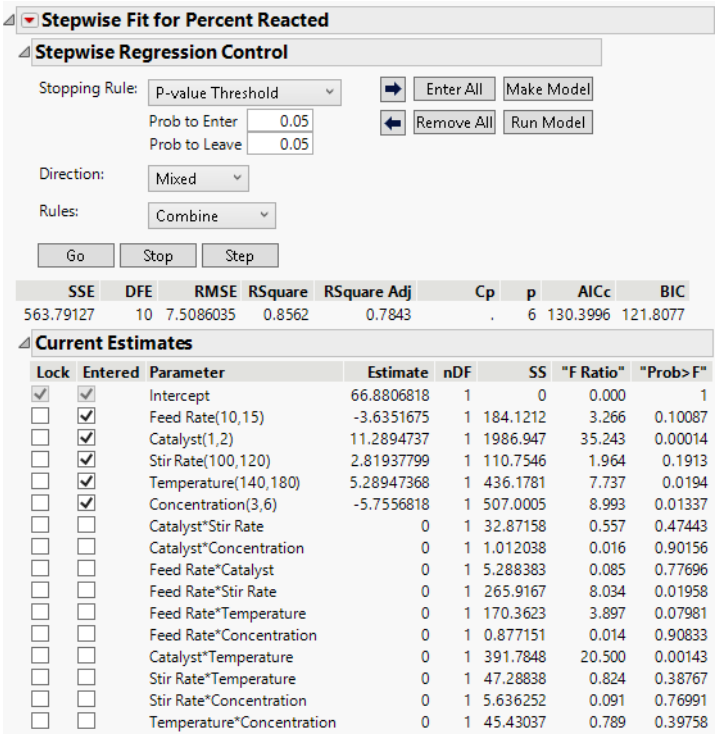
   The **Model** script opens the Fit Model window with all main effects and two-factor interactions as effects.

3. Change the fitting personality from **Standard Least Squares** to **Stepwise** and then click **Run.**

4. In the Stepwise Regression Control panel, select **P-value Threshold** from the **Stopping Rule** menu and **Mixed** from the **Direction** menu. Enter 0.05 for the **Prob to Enter** and 0.05 for the **Prob to Leave**.

5. In the Current Estimates panel, select the **Entered** check boxes for all the main effect terms (Feed Rate - Concentration).

Figure 6.6 Completed Stepwise Model



Stepwise Fit for Percent Reacted

Stepwise Regression Control

Stopping Rule: P-value Threshold | Enter All | Make Model
Prob to Enter 0.05 | Remove All | Run Model
Prob to Leave 0.05
Direction: Mixed
Rules: Combine
Go | Stop | Step

SSE	DFE	RMSE	RSquare	RSquare Adj	Cp	p	AICc	BIC
563.79127	10	7.5086035	0.8562	0.7843	.	6	130.3996	121.8077

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	66.8806818	1	0	0.000	1
☐	☑	Feed Rate(10,15)	-3.6351675	1	184.1212	3.266	0.10087
☐	☑	Catalyst(1,2)	11.2894737	1	1986.947	35.243	0.00014
☐	☑	Stir Rate(100,120)	2.81937799	1	110.7546	1.964	0.1913
☐	☑	Temperature(140,180)	5.28947368	1	436.1781	7.737	0.0194
☐	☑	Concentration(3,6)	-5.7556818	1	507.0005	8.993	0.01337
☐	☐	Catalyst*Stir Rate	0	1	32.87158	0.557	0.47443
☐	☐	Catalyst*Concentration	0	1	1.012038	0.016	0.90156
☐	☐	Feed Rate*Catalyst	0	1	5.288383	0.085	0.77696
☐	☐	Feed Rate*Stir Rate	0	1	265.9167	8.034	0.01958
☐	☐	Feed Rate*Temperature	0	1	170.3623	3.897	0.07981
☐	☐	Feed Rate*Concentration	0	1	0.877151	0.014	0.90833
☐	☐	Catalyst*Temperature	0	1	391.7848	20.500	0.00143
☐	☐	Stir Rate*Temperature	0	1	47.28838	0.824	0.38767
☐	☐	Stir Rate*Concentration	0	1	5.636252	0.091	0.76991
☐	☐	Temperature*Concentration	0	1	45.43037	0.789	0.39758

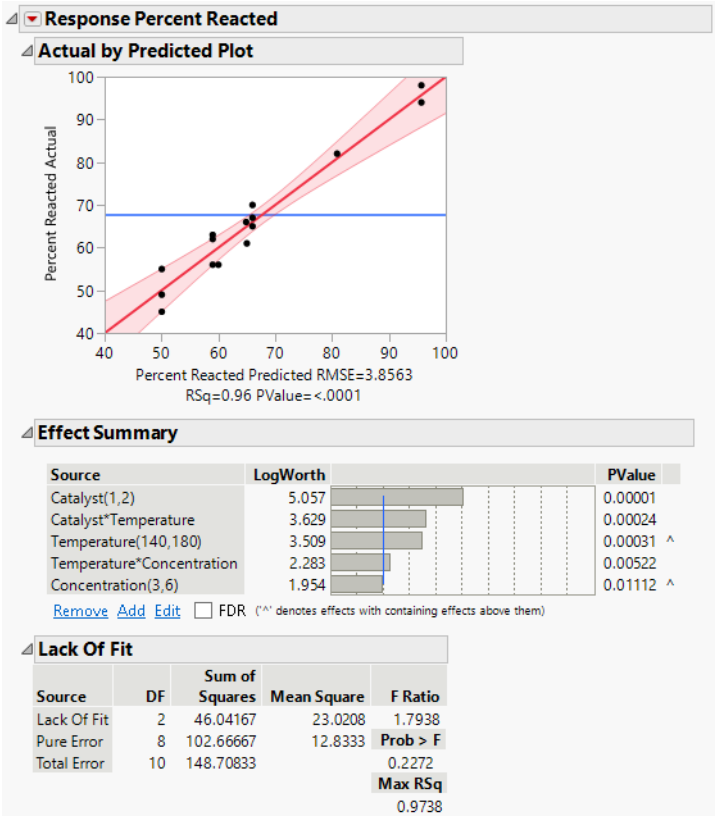
6. Click **Go.**

   Stepwise regression, beginning with the main effects and using a *p*-value based search, results in a model with three main effects and two interaction terms. Click **Make Model** on the Stepwise Regression Control panel.

   This launches the Fit Model platform for the model built using the stepwise procedure.

7. Click **Run.**

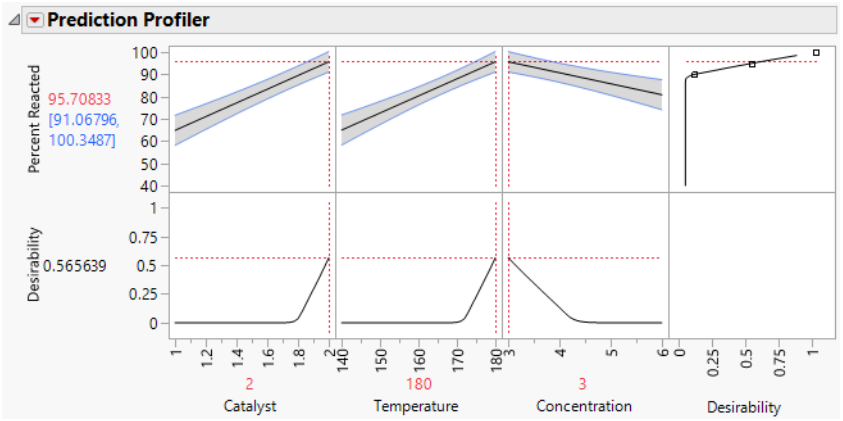
Figure 6.7 Prediction Model Analysis of Variance and Lack of Fit Tests



The Actual by Predicted Plot indicates that the overall model is significant (PValue < .0001). Both the Actual by Predicted Plot and the Lack of Fit tests show no evidence of model misspecification. The Effect Summary report shows that Catalyst is the most significant effect.

8. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 6.8 Maximum Percent Reacted

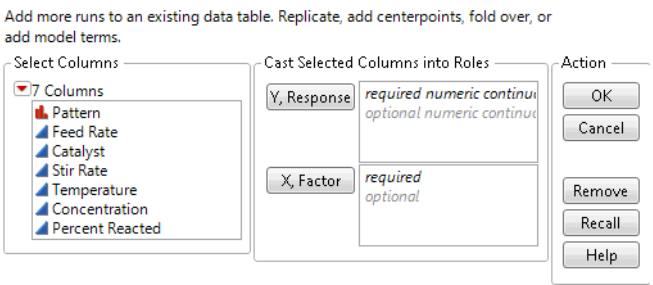


The prediction profile plot shows that the maximum percent reaction occurs at high Catalyst and Temperature and low Concentration. At these extreme settings, the estimate of Percent Reacted is 95.7. The desirability profiles flatten in regions where the predicted response falls outside of the response limits.

Launch the Augment Design Platform

Launch the Augment Design platform by selecting **DOE** > **Augment Design** from the data table that contains the design that you want to augment.

Figure 6.9 Augment Design Launch Window



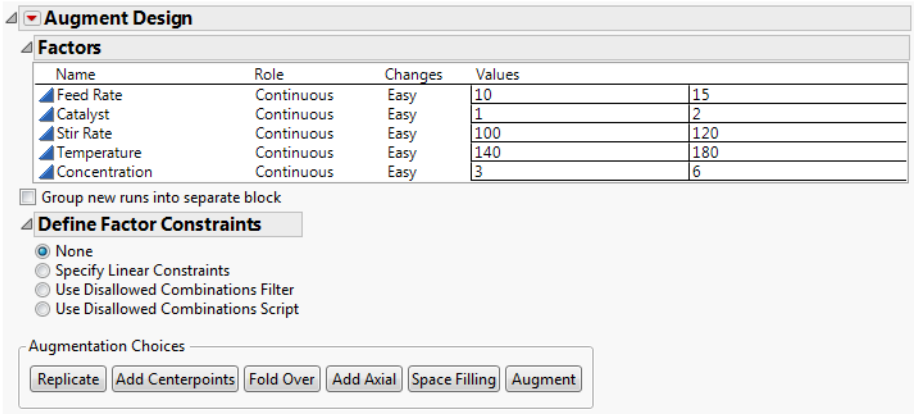
Y, Response Enter the numeric response column or columns. Entering a response is required.

X, Factor Enter the factor columns. Factors can be of any data type or modeling type.

Augment Design Window

The initial Augment Design window consists of the Factors and Define Factor Constraints outlines, and the Augmentation Choices panel.

Figure 6.10 Initial Augment Design Window Using Reactor 8 Runs.jmp



Factors

The factors are those you specified in the Augment Design launch window.

Name Lists all factors listed as **X, Factor** in the Augment Design launch window except for factors with the Random Block design role column property.

Role Specifies the Design Role of the factor specified in the column property. If the factor does not have a Design Role column property and is constant, then **Constant** appears in the Role column. Otherwise, the factor's modeling type appears in the Role column.

Changes Indicates whether the factor levels are Easy, Hard, or Very Hard to change as specified by the Factor Changes column property in the data table. If the factor does not have a Factor Changes column property, then Changes is specified as Easy.

Note: If a factor has a Factor Changes column property that is set to Hard or Very Hard, then the corresponding whole plot factor must be included in the **X, Factor** list in the Augment Design launch window.

Values For continuous factors, shows the minimum and maximum values. For categorical factors, shows the levels.

Tip: Factors that have a role of Categorical or Constant appear in the Name column with a down arrow icon. Click the down arrow to add levels. If the factor is Constant and has a categorical modeling type, multiple levels can be added. If the factor is Constant and has a continuous modeling type, only one level can be added.

Group new runs into separate block Adds a blocking factor to the design with a block for the original design and a block for the augmented runs.

Define Factor Constraints

If you augment a design using the Space Filling or Augment options, you can define restrictions on the design space for the added runs.

Use Define Factor Constraints to restrict the design space. Unless you have loaded a constraint or included one as part of a script, the **None** option is selected. To specify constraints, select one of the other options:

Specify Linear Constraints Specifies inequality constraints on linear combinations of factors. Available only for factors with a Role of Continuous or Mixture. See "Specify Linear Constraints".

Note: When you save a script for a design that involves a linear constraint, the script expresses the linear constraint as a *less than or equal to* inequality (≤).

Use Disallowed Combinations Filter Defines sets of constraints based on restricting values of individual factors. You can define both AND and OR constraints. See "Use Disallowed Combinations Filter".

Use Disallowed Combinations Script Defines disallowed combinations and other constraints as Boolean JSL expressions in a script editor box. See "Use Disallowed Combinations Script".

Note: When you analyze a design that has factor constraints, the model profiler honors the constraints.

Specify Linear Constraints

In cases where it is impossible to vary continuous factors independently over the design space, you can specify linear inequality constraints. Linear inequalities describe factor level settings that are allowed. For an example, see "Mixture of Mixtures Design".

Click **Add** to enter one or more linear inequality constraints.

Add Adds a template for a linear expression involving all the continuous factors in your design. Enter coefficient values for the factors and select the direction of the inequality to reflect your linear constraint. Specify the constraining value in the box to the right of the inequality. To add more constraints, click **Add** again.

Note: The Add option is disabled if you have already constrained the design region by specifying a Sphere Radius.

Remove Last Constraint Removes the last constraint.

Check Constraints Checks the constraints for consistency. This option removes redundant constraints and conducts feasibility checks. A JMP alert appears if there is a problem. If constraints are equivalent to bounds on the factors, a JMP alert indicates that the bounds in the Factors outline have been updated.

Use Disallowed Combinations Filter

This option uses an adaptation of the Data Filter to facilitate specifying disallowed combinations. For detailed information, see *Using JMP*.

Select factors from the Add Filter Factors list and click **Add**. Then specify the disallowed combinations by using the slider (for continuous factors) or by selecting levels (for categorical factors). For an example, see "Response Surface Design With Constraints and Categorical Factor".

The red triangle options for the Add Filter Factors menu are those found in the Select Columns panel of many platform launch windows. See *Using JMP*.

When you click Add, the Disallowed Combinations control panel shows the selected factors and provides options for further control. Factors are represented as follows, based on their modeling types:

Continuous Factors For a continuous factor, a double-ended slider that spans the range of factor settings appears. You can specify disallowed settings by dragging the slider ends or by setting the endpoints by clicking on the text value under the slider. In the slider, a solid blue highlight represents the disallowed values.

Categorical Factor For a categorical factor, the possible levels are displayed either as labeled blocks or, when the number of levels is large, as list entries. Select a level to disallow it. To select multiple levels, hold the Control key. The block or list entries are highlighted to indicate the levels that have been disallowed. When you add a categorical factor to the Disallowed Combinations panel, the number of levels of the categorical factor is given in parentheses following the factor name.

Disallowed Combinations Options

The control panel has the following controls:

Clear Clears all disallowed factor level settings that you have specified. This does not clear the selected factors.

Start Over Removes all selected factors and returns you to the initial list of factors.

AND Opens the Add Filter Factors list. Selected factors become an AND group. Any combination of factor levels specified within an AND group is disallowed.

To add a factor to an AND group later on, click the group's outline to see a highlighted rectangle. Select AND and add the factor.

To remove a single factor, select **Delete** from its red triangle menu.

OR Opens the Add Filter Factors list. Selected factors become a separate AND group. For AND groups separated by OR, a combination is disallowed if it is specified in at least one AND group.

Red Triangle Options for Factors

A factor can appear in several OR groups. An occurrence of the factor in a specific OR group is referred to as an *instance* of the factor.

Delete Removes the selected instance of the factor from the Disallowed Combinations panel.

Clear Selection Clears any selection for that instance of the factor.

Invert Selection Deselects the selected values and selects the values not previously selected for that instance of the factor.

Display Options Available only for categorical factors. Changes the appearance of the display. Options include showing each level as a block, list, or single category, or adding a check box next to each value.

Find Available only for categorical factors. Provides a text box beneath the factor name where you can enter a search string for levels of the factor. Press the Enter key or click outside the text box to perform the search. Once **Find** is selected, Find options appear in the red triangle menu, such as Does not contain, Match Case, Starts with or Ends with, or Clear Find.

Use Disallowed Combinations Script

Use this option to disallow particular combinations of factor levels using a JSL script. This option can be used with continuous factors or mixed continuous and categorical factors.

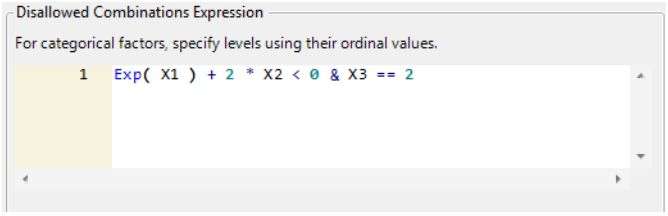
This option opens a script window where you insert a script that identifies the combinations that you want to disallow. The script must evaluate as a Boolean expression. When the expression evaluates as true, the specified combination is disallowed.

When forming the expression for a categorical factor, use the ordinal value of the level or the name of the level. If a factor's levels are high, medium, and low, specified in that order in the Factors outline, their associated ordinal values are 1, 2, and 3. For example, suppose that you have two continuous factors, X1 and X2, and a categorical factor X3 with three levels: L1, L2, and L3, in order. You want to disallow levels where the following is true:

$$e^{X_1} + 2X_2 < 0 \text{ and } X_3 = L2$$

Enter the expression `(Exp(X1) + 2*X2 < 0) & (X3 == 2)` into the script window.

Figure 6.11 Expression in Script Editor



(In the figure, unnecessary parentheses were removed by parsing.) Notice that functions can be entered as part of the Boolean expression. The expression
`(Exp(X1) + 2*X2 < 0) & (X3 == "L2")` is also valid.

Augmentation Choices

The Augment Design platform has the following choices for augmentation:

Replicate Replicates the design a specified number of times. See "Replicate a Design".

Add Centerpoints Adds center points. Specify how many runs you want to add as center points to the design. A center point is a run whose setting for each continuous factor is midway between the high and low settings. See "Center Points, Replicate Runs, and Testing".

If a design contains both continuous and other types of factors, center points might not be balanced relative to the levels of the other factors. Augment Design chooses the center points to maximize the D-, I-, or alias efficiency of the design.

See "Add Center Points to a Design".

Fold Over Creates a foldover design. See "Create a Foldover Design".

Add Axial Adds axial points and center points to a design. Add Axial can be used to generate a central composite design from a screening design. See "Add Axial Points to a Design".

Space Filling Adds additional runs to any design consisting of continuous factors. Additional runs are constructed using the fast flexible filling methodology. See "Space Filling Augmentation". Space Filling does not support grouping new runs in a separate block.

Augment Adds runs to the design using a model. This model typically has more terms than the original model. See "Example Using the Augment Design Platform".

Augment Design Platform Options

The Accelerated Life Test Plan red triangle menu contains the following options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Save X Matrix Saves scripts called Moments Matrix and Model Matrix to the design data table. These scripts contain the moments and design matrices. See "Save X Matrix".

Caution: For a design with nominal factors, the matrix in the Model Matrix script saved by the Save X Matrix option is *not* the coding matrix used in fitting the linear model. You can obtain the coding matrix used for fitting the model by selecting the option Save Columns > Save Coding Table in the Fit Model report that you obtain when you run the Model script.

Optimality Criterion Changes the design optimality criterion. The default criterion, **Recommended**, specifies D-optimality for all design types, unless you added quadratic effects using the RSM button in the Model outline. For more information about the D-, I-, and alias-optimal designs, see "Optimality Criteria".

Note: You can set a preference to always use a given optimality criterion. Select File > Preferences > Platforms > DOE. Select Optimality Criterion and choose your preferred criterion.

Number of Starts Enables you to specify the number of random starts used in constructing the design. See "Number of Starts".

Design Search Time Maximum number of seconds spent searching for a design. The default search time is based on the complexity of the design. See "Design Search Time" and "Number of Starts".

If the iterations of the algorithm require more than a few seconds, a Computing Design progress window appears. If you click **Cancel** in the progress window, the calculation stops and gives the best design found at that point. The progress window also displays D-efficiency for D-optimal designs that do not include factors with Changes set to Hard or Very Hard or with Estimability set to If Possible.

Note: You can set a preference for Design Search Time. Select File > Preferences > Platforms > DOE. Select Design Search Time and enter the maximum number of seconds. In certain situations where more time is required, JMP extends the search time.

Sphere Radius Constrains the continuous factors in a design to a hypersphere. Specify the radius and click **OK**. Design points are chosen so that their distance from 0 equals the Sphere Radius. Select this option before you click Make Design.

Note: Sphere Radius constraints cannot be combined with constraints added using the Specify Linear Constraints option. Also, the option is not available when hard-to-change factors are included (split-plot designs).

Advanced Options

Mixture Sum Sets the sum of the mixture factors to any positive value. Use this option to keep a component of a mixture constant throughout an experiment.

Split Plot Variance Ratio Specifies the ratio of the variance of the random whole plot and the subplot variance (if present) to the error variance. Before setting this value, you must define a hard-to-change factor for your split-plot design, or hard and very-hard-to-change factors for your split-split-plot design. Then you can enter one or two positive numbers for the variance ratios, depending on whether you have specified a split-plot or a split-split-plot design.

Prior Parameter Variance (Available only when the Model outline is available.) Specifies the weights that are used for factors whose Estimability is set to If Possible. The option is updated to show the default weights when you click Make Design. Enter a positive number for each of the terms for which you want to specify a weight. The value that you enter is the square root of the reciprocal of the prior variance. A larger value represents a smaller variance and therefore more prior information that the effect is not active.

Bayesian D- or I-optimality is used in constructing designs with If Possible factors. The default values used in the algorithm are 0 for Necessary terms, 4 for interactions involving If Possible terms, and 1 for If Possible terms. See "The Alias Matrix" and "Optimality Criteria".

A- Optimality Parameter Weights (Use for A-Optimal designs.) Specifies weights for the model parameters. This enables you to place more weight on the variance of the main effects over say 2nd order effects. For more information about parameter weights see Morgan and Stallings (2017).

D Efficiency Weight Specifies the relative importance of D-efficiency to alias optimality in constructing the design. Select this option to balance reducing the variance of the coefficients with obtaining a desirable alias structure. Values should be between 0 and 1. Larger values give more weight to D-Efficiency. The default value is 0.5. This option has an effect only when you select Make Alias Optimal Design as your Optimality Criterion.

For the definition of D-efficiency, see "Optimality Criteria". For more information about alias optimality, see "Alias Optimality".

Save Script to Script Window Creates the script for the design that you specified in the Custom Design window and places it in an open script window.

Additional Examples of Augmentation Choices

- "Replicate a Design"
- "Add Center Points to a Design"
- "Create a Foldover Design"
- "Add Axial Points to a Design"
- "Space Filling Augmentation"

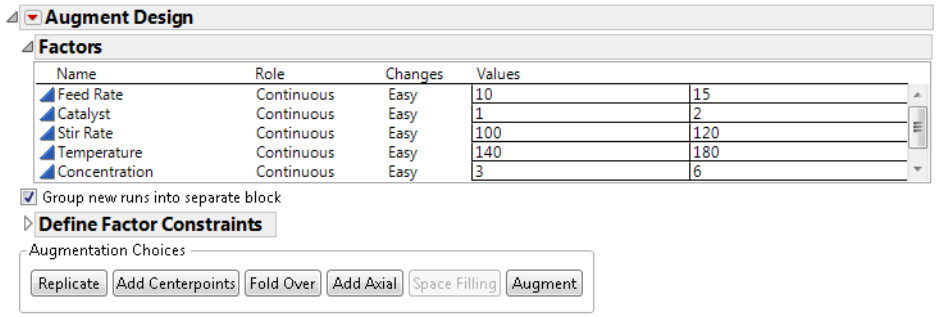
Replicate a Design

Augment a design with replication for a direct check on the assumption that the error variance is constant. Replication also reduces the variability of the regression coefficients in the presence of large process or measurement variability.

This example replicates an 8 run design.

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 8 Runs.jmp.
2. Select **DOE > Augment Design**.
3. Select Percent Reacted and click **Y, Response**.
4. Select Feed Rate, Catalyst, Stir Rate, Temperature, and Concentration and click **X, Factor**.
5. Click **OK**.

Figure 6.12 Choose an Augmentation Type



6. Select **Group new runs into separate block** to add a blocking variable to the design.
7. Click **Replicate**, enter 2, and then click **OK**.

Note: The replicate dialog box asks for the number of times each run is to be performed. Entering 2 specifies that you want each run to appear twice in the resulting design. This results in a single replicate of your original design.

Figure 6.13 Reactor Data Design Augmented with a Replicate

Factor Design

Run	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Block
1	15	2	100	140	3	1
2	10	2	120	140	6	1
3	15	1	120	140	3	1
4	10	2	100	180	3	1
5	10	1	120	180	3	1
6	10	1	100	140	6	1
7	15	2	120	180	6	1
8	15	1	100	180	6	1
9	15	2	100	140	3	2
10	10	2	120	140	6	2
11	15	1	120	140	3	2
12	10	2	100	180	3	2
13	10	1	120	180	3	2
14	10	1	100	140	6	2
15	15	2	120	180	6	2
16	15	1	100	180	6	2

The original 8 runs are in Block 1 and the replicated runs are in Block 2.

Note: The full design can be evaluated from the Augment Design Platform. For more information about design evaluation, see "Evaluate Designs".

8. Click **Make Table**.

Figure 6.14 The Replicated Design

	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Block	Percent Reacted
1	15	2	100	140	3	1	61
2	10	2	120	140	6	1	67
3	15	1	120	140	3	1	56
4	10	2	100	180	3	1	94
5	10	1	120	180	3	1	66
6	10	1	100	140	6	1	56
7	15	2	120	180	6	1	82
8	15	1	100	180	6	1	45
9	15	2	100	140	3	2	•
10	10	2	120	140	6	2	•
11	15	1	120	140	3	2	•
12	10	2	100	180	3	2	•
13	10	1	120	180	3	2	•
14	10	1	100	140	6	2	•
15	15	2	120	180	6	2	•
16	15	1	100	180	6	2	•

The table includes the original design and results in Block 1. The augmented runs are in Block 2. The percent reacted values can be added to the table after the augmented runs have been measured.

Add Center Points to a Design

Augment a design with center points to check for curvature and reduce the prediction error in the center of the factor region. Center points are usually replicated points that allow for an independent estimate of pure error, which can be used in a lack-of-fit test.

This example adds two center points to an 8-run design.

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 8 Runs.jmp.
2. Select **DOE > Augment Design**.
3. Select Percent Reacted and click **Y, Response**.
4. Select Feed Rate, Catalyst, Stir Rate, Temperature, and Concentration and click **X, Factor**.
5. Click **OK**.
6. Select **Group new runs into separate block** to add a blocking variable to the design.
7. Click **Add Centerpoints**, enter 2, and then click **OK**.
8. Click **Make Table**.

Figure 6.15 Design with Two Center Points Added

	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Block	Percent Reacted
1	15	2	100	140	3	1	61
2	10	2	120	140	6	1	67
3	15	1	120	140	3	1	56
4	10	2	100	180	3	1	94
5	10	1	120	180	3	1	66
6	10	1	100	140	6	1	56
7	15	2	120	180	6	1	82
8	15	1	100	180	6	1	45
9	12.5	1.5	110	160	4.5	2	•
10	12.5	1.5	110	160	4.5	2	•

After data has been collected for the two center runs you can then test for curvature in your model.

Create a Foldover Design

Augment a design with a foldover to remove the confounding of two-factor interactions and main effects. A foldover design is especially useful as a follow-up to saturated or near-saturated fractional factorial or Plackett-Burman designs.

This example creates a foldover design of an 8-run design.

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 8 Runs.jmp.
2. Select **DOE > Augment Design**.
3. Select Percent Reacted and click **Y, Response**.

4. Select Feed Rate, Catalyst, Stir Rate, Temperature, and Concentration and click **X, Factor**.
5. Click **OK**.
6. Select **Group new runs into separate block** to add a blocking variable to the design.
7. Click **Fold Over** and then click **OK**.

   In the fold over dialog box, if you select no factors, then all design factors are included in the fold over. If you choose a subset of factors to fold over, the remaining factors are replicates of the original runs.

Figure 6.16 Foldover Design on All Factors

Factor Design

Run	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Block
1	15	2	100	140	3	1
2	10	2	120	140	6	1
3	15	1	120	140	3	1
4	10	2	100	180	3	1
5	10	1	120	180	3	1
6	10	1	100	140	6	1
7	15	2	120	180	6	1
8	15	1	100	180	6	1
9	10	1	120	180	6	2
10	15	1	100	180	3	2
11	10	2	100	180	6	2
12	15	1	120	140	6	2
13	15	2	100	140	6	2
14	15	2	120	180	3	2
15	10	1	100	140	3	2
16	10	2	120	140	3	2

8. Click **Make Table**.

Each run in Block 2 can be obtained by changing the factor levels (from low to high or from high to low) of a run from Block 1.

Add Axial Points to a Design

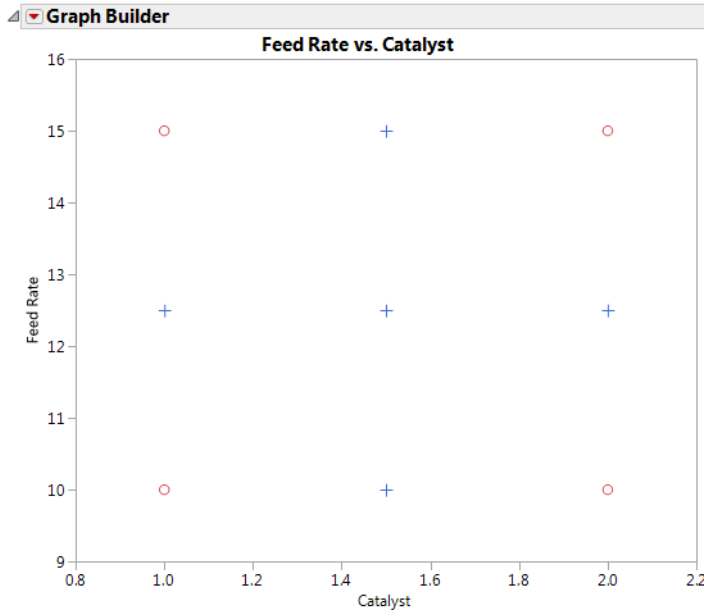
Augment a design with axial and center points to augment a screening design in order to obtain a response surface design. This enables you to fit a response surface model to your data to optimize the settings of your factors with respect to your response.

This example augments an 8 run screening design with axial and center points.

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 8 Runs.jmp.
2. Select **DOE > Augment Design**.
3. Select Percent Reacted and click **Y, Response**.
4. Select Feed Rate, Catalyst, Stir Rate, Temperature, and Concentration and click **X, Factor**.
5. Click **OK**.
6. Select **Group new runs into separate block** to add a blocking variable to the design.

7. Click **Add Axial,** enter 1 for axial value, 2 for number of center points, and then click **OK**.
8. Click **Make Table**.
9. From the Augmented Design table, select **Rows > Color or Mark by Column**.
10. Select Block, from Markers select **Standard**, and then click **OK**.
11. Select **Graph** > **Graph Builder**.
12. Drag Feed Rate to Y and drag Catalyst to X. Then click the smoother icon to remove.

Figure 6.17 Graphical View of the Augmented Design for Two Factors



The "+" points in the graph of Feed Rate versus Catalyst are the added axial and center points. The axial points are on the face of the design space. Increasing or decreasing the axial value in the augmentation of the design would move the axial points beyond or within the design space.

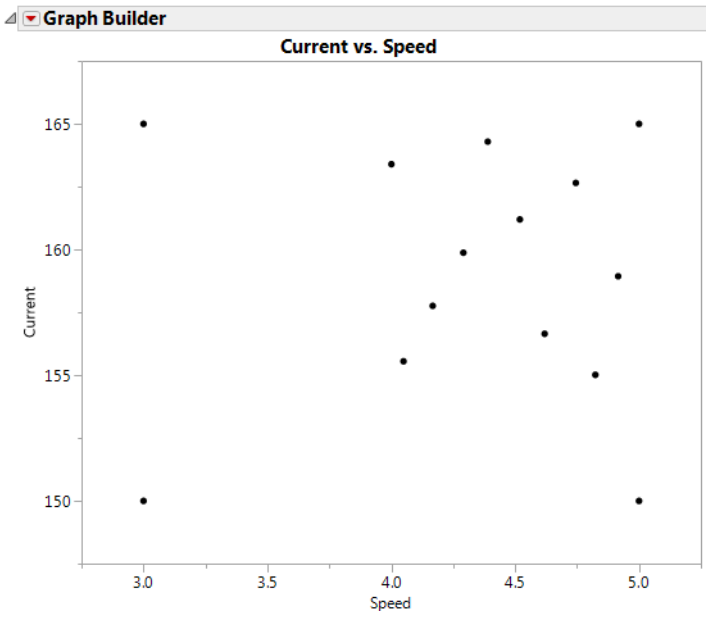
Space Filling Augmentation

Augment a design with space filling to add points to a design consisting of continuous factors. The Space Filling choice accommodates constraints on the design space. You can specify linear constraints or disallowed combinations. For more information about the algorithm used, see "Statistical Details".

This example augments a design to explore a constrained area of the design space where the response, Depth, was maximized, with added constraints on both factors.

1. Select **Help > Sample Data Library** and open Design Experiment/DOE Example 1.jmp.
2. Select **DOE > Augment Design**.
3. Select Depth and click **Y, Response**.
4. Select Speed and Current, click **X, Factor**, and then click **OK**.
5. Open the Define Factor Constraints outline.
6. Select **Specify Linear Constraints** and click **Add**.
7. Enter 1 for Speed, select greater than or equal to, and then enter 4 for the lower bound on Speed.
8. Click **Add**.
9. Enter 1 for Current, select greater than or equal to, and then enter 155 for the lower bound on Current.
10. Click **Space Filling**.
11. Enter 10 and click **OK**.
12. Click **Make Table**.
13. From the Augmented Design table, select Graph Builder.
14. Drag Speed to the X drop zone and drag Current to the Y drop zone.

Figure 6.18 Space Filling Augmented Runs



The augmented runs fill the space in the upper right corner of the design space. This is the constrained design space.

Statistical Details

This section contains statistical details about the algorithm used to select points for the space-filling augmentation choice.

The algorithm that is used to augment designs begins by generating a large number of random points within the specified design region. These points are then clustered using a Fast Ward algorithm into a number of clusters that equals the Number of Additional Runs that you specify.

The final design points are obtained by optimizing the MaxPro (*maximum projection*) criterion over the existing and additional runs. For p factors and n equal to the number of existing and additional runs, the MaxPro criterion strives to find points in the clusters that minimize the following criterion:

$$C_{MaxPro} = \sum_{i}^{n-1} \sum_{j=i+1}^{n} \left[1 / \prod_{k=1}^{p} (x_{ik} - x_{jk})^2 \right]$$

The MaxPro criterion maximizes the product of the distances between design points in a way that involves all factors. This supports the goal of providing good space-filling properties on projections of factors. See Joseph et al. (2015).

Chapter 7

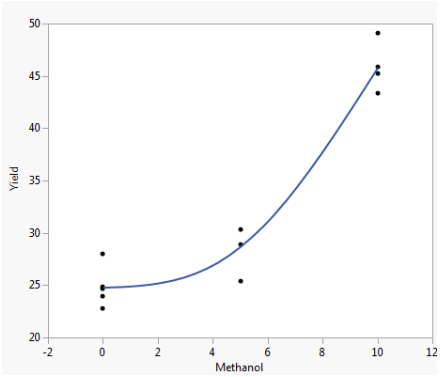
Definitive Screening Designs

Definitive screening designs (DSD) are screening designs. They are appropriate for early stage experimentation work, typically with four or more factors. DSD can be used for combinations of continuous or two-level categorical factors. They work best when most of the factors are continuous. Each continuous factor has three levels allowing one to investigate quadratic model terms for continuous factors.

Here are areas where definitive screening designs are superior to standard screening designs:

- They help identify the causes of nonlinear effects by fielding each continuous factor at three levels. In standard screening designs, continuous factors have only two levels. You can add center points to screening designs, but these points establish only if curvature exists. They do not allow you to identify the factors responsible for quadratic effects.
- They avoid confounding between any effects up through the second order. For continuous factors, definitive screening designs have main effects that are orthogonal to each other and orthogonal to two-factor interactions and quadratic effects. Two-factor interactions are not completely confounded with each other. Confounding occurs in many standard screening designs with a similar number of runs.
- They avoid the need for costly additional experimentation to resolve ambiguity from the initial results of standard screening designs.

Figure 7.1 Plot of Response against Factor Values Showing Curvature



Contents

Overview of Definitive Screening Design 241
Examples of Definitive Screening Designs 243
Definitive Screening Design 244
Comparison with a Fractional Factorial Design 246
Definitive Screening Design with Blocking 248
Comparison of a Definitive Screening Design with a Plackett-Burman Design 252
Definitive Screening Design Window 255
Responses 256
Factors 258
Design Options 260
Design 262
Design Evaluation 262
Output Options 263
Definitive Screening Design Options 265
Simulate Responses 267
Technical Details 270
Structure of Definitive Screening Designs 270
Analysis of Experimental Data 272

Overview of Definitive Screening Design

Investigators use screening designs when they want to identify the factors that have the most substantial effects on a response. A screening design enables you to study a large number of factors in a fairly small experiment.

Many standard screening designs focus on estimating main effects. Definitive screening designs offer advantages over standard screening designs. They avoid confounding of effects and can identify factors having a nonlinear effect on the response. For more information about the advantages and construction of definitive screening designs, see Jones and Nachtsheim (2011a).

For designs containing only continuous factors, compare these properties of definitive screening designs versus standard screening designs:

Note: When quadratic effects are mentioned, the standard screening designs are assumed to have center points.

- Main effects are orthogonal to two-factor interactions.
  - Definitive Screening Designs: Always
  - Standard Screening Designs: Only for Resolution IV or higher
- No two-factor interaction is completely confounded with any other two-factor interaction.
  - Definitive Screening Designs: Always
  - Standard Screening Designs: Only for Resolution V or higher
- All quadratic effects are estimable in models containing only main and quadratic effects.
  - Definitive Screening Designs: Always
  - Standard Screening Designs: Never

These properties are described more fully in the remainder of this section.

Standard Screening Designs

Standard screening designs, such as fractional factorial or Plackett-Burman designs, attempt to study many factors with a relatively small allocation of resources. However, standard screening designs have several undesirable features:

- They can alias some main effects with two-factor interactions. In Plackett-Burman designs, for example, main effects are correlated with several two-factor interactions. If one or more two-factor interaction effects are substantial, then the experimenter must perform additional runs to resolve the ambiguities.

- They can also confound some two-factor interactions with each other. Consequently, if a two-factor interaction effect is substantial, then the experimenter must perform additional runs to resolve the remaining ambiguities.
- Continuous factors are usually set at only two levels (low and high). However, engineers and scientists often prefer designs where continuous factors are set at three levels (low, middle, and high). This is because two levels are not sufficient to detect nonlinearity, which is common in physical systems. You can use a traditional screening design with added center points to detect nonlinearity, but such a design does not identify the responsible factors.

Definitive Screening Designs

Using definitive screening designs, you can do the following:

- Avoid model ambiguity, enabling you to identify important factors more quickly and efficiently.
- Identify the cause of nonlinear effects while avoiding confounding any terms up to second order. So not only can you *detect* nonlinearity, as you might with center points in a traditional screening design, but you can *identify* the responsible factors.

Definitive screening designs offer the following advantages:

- Definitive screening designs require only a small number of runs. For six or more factors, the minimum number of required runs is usually only a few more than twice the number of factors. For more detail on the number of runs, see "Conference Matrices and the Number of Runs".
- Main effects are orthogonal to two-factor interactions. This means that estimates of main effects are not biased by the presence of active two-factor interactions, whether these interactions are included in the model or not. Note that resolution III screening designs confound some main and interaction effects. Also, Plackett-Burman designs produce biased main effect estimates if there are active two-factor interactions.
- No two-factor interaction is completely confounded with any other two-factor interaction. However, a two-factor interaction might be correlated with other two-factor interactions. Note that resolution IV screening designs completely confound some two-factor interaction effects.
- All quadratic effects are estimable in models comprised only of main effects and quadratic terms. This enables you to identify the factors that account for nonlinearity. Note that traditional screening designs with added center points do *not* allow estimation of all quadratic effects in models consisting of main and quadratic effects.
- Quadratic effects are orthogonal to main effects and not completely confounded with two-factor interactions. A quadratic effect might be correlated with interaction effects.
- For 6 through at least 30 factors, it is possible to estimate the parameters of any full quadratic model involving three or fewer factors with high precision.

- For 18 factors or more, they can fit full quadratic models in any 4 factors. For 24 factors or more, they can fit full quadratic models in any 5 factors.

Definitive Screening Design Platform

The Definitive Screening Design platform enables you to construct definitive screening designs for continuous factors and for two-level categorical factors. It also enables you to construct blocked designs. You can add extra non-center runs that enhance the ability of the design to reliably detect effects when many effects are active.

To view the absolute values of the correlations among effects, use the Color Map on Correlations provided as part of the Design Evaluation outline in the Definitive Screening Design window. You can compare the aliasing structure of definitive screening designs to that of other designs by comparing their color maps on correlations. See "Color Map on Correlations".

For more information about the structure of definitive screening designs, see "Structure of Definitive Screening Designs". For information about definitive screening designs with blocks, see "Blocking in Definitive Screening Designs". For suggestions on how to analyze data obtained using definitive screening designs, see "Analysis of Experimental Data".

Fit Definitive Screening Platform

After you run a Definitive Screening Design (DSD), analyze your results using the Fit Definitive Screening platform. Standard model selection methods applied to DSDs can fail to identify active effects. To identify active main effects and second-order effects, the Fit Definitive Screening platform uses an algorithm called *Effective Model Selection for DSDs*. This algorithm leverages the special structure of DSDs. See "The Fit Definitive Screening Platform".

If you create your DSD in JMP, the design table contains a script called Fit Definitive Screening that automatically runs an analysis using the Effective Model Selection for DSDs methodology.

Examples of Definitive Screening Designs

- "Definitive Screening Design"
- "Comparison with a Fractional Factorial Design"
- "Definitive Screening Design with Blocking"
- "Comparison of a Definitive Screening Design with a Plackett-Burman Design"

Definitive Screening Design

Suppose that you need to determine which of six factors have an effect on the yield of an extraction process.

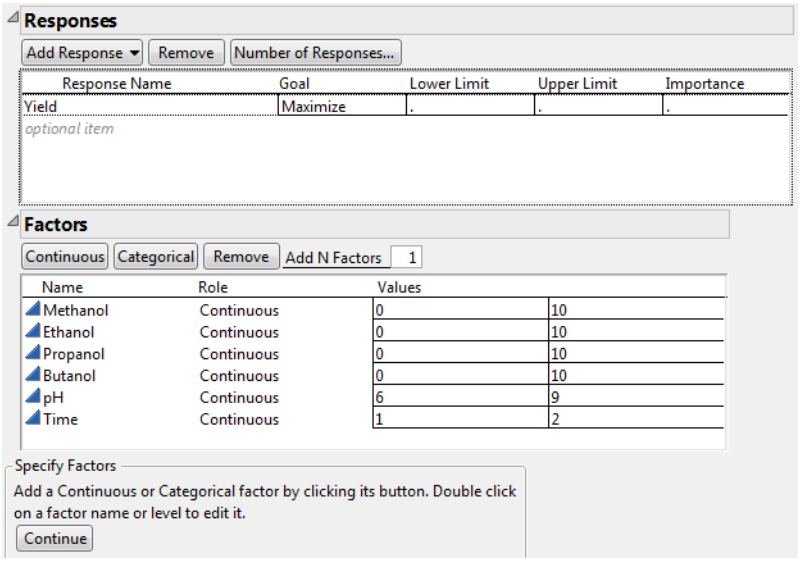
Create the Design

The factors and their settings are given in the data table Extraction Factors.jmp. You create a definitive screening design to investigate.

1. Select **DOE > Definitive Screening > Definitive Screening Design**.
2. Double-click Y under Response Name and type Yield.
3. Select **Help > Sample Data Library** and open Design Experiment/Extraction Factors.jmp.
4. Click the red triangle next to Definitive Screening Design and select **Load Factors**

   The factor names and ranges are added to the Factors outline.

Figure 7.2 Responses and Factors for Extraction Design



5. Click **Continue**.

   The Design Options outline opens. Here you can specify a blocking structure. There is no need to block in this example, so you accept the default selection of **No Blocks Required**.

   You can also choose to add Extra Runs, which greatly enhance your ability to detect second-order effects. A minimum of four Extra Runs is highly recommended and is the default.

6. Click **Make Design**.

The Definitive Screening Design window updates to show a Design outline and a Design Evaluation outline.

Figure 7.3 Design Outline

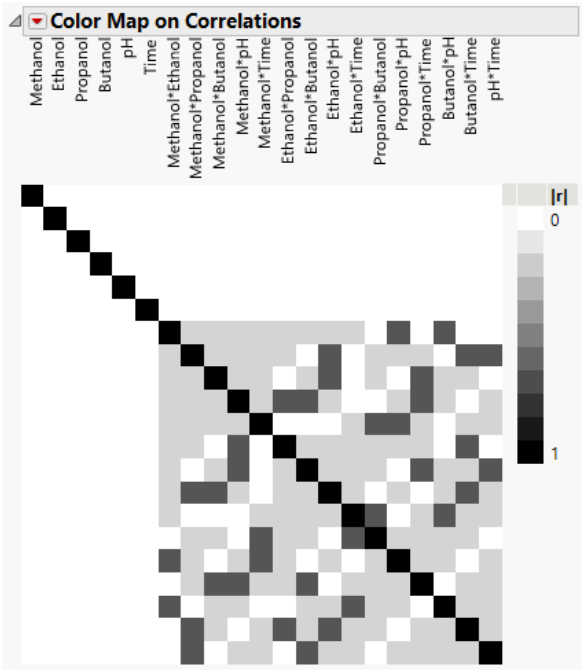
Design

Run	Methanol	Ethanol	Propanol	Butanol	pH	Time
1	5	10	10	10	9	2
2	5	0	0	0	6	1
3	10	5	10	10	6	2
4	0	5	0	0	9	1
5	10	0	5	10	9	1
6	0	10	5	0	6	2
7	10	0	0	5	9	2
8	0	10	10	5	6	1
9	10	10	0	0	7.5	2
10	0	0	10	10	7.5	1
11	10	0	10	0	6	1.5
12	0	10	0	10	9	1.5
13	10	10	0	10	6	1
14	0	0	10	0	9	2
15	10	10	10	0	9	1
16	0	0	0	10	6	2
17	5	5	5	5	7.5	1.5

7. Open the **Design Evaluation > Color Map on Correlations** outline.

   The Color Map on Correlations assigns a color intensity scale to the absolute values of correlations among all main effects and two-factor interactions.

Figure 7.4 Color Map on Correlations for Extraction Design



Note the following:

- The solid white area shows that there is no correlation between main effects or between main effects and two-factor interactions.
- The shaded gray areas indicate that the absolute correlations between two-factor interactions are small.
- The black squares indicate absolute correlations of 1. These all appear on the diagonal, reflecting the expected correlation of an effect with itself.

In the Output Options panel, note that the Run Order is set to **Randomize**.

8. Click **Make Table** to obtain the data table shown in Figure 7.5.

Note: The runs in your design might appear in a different order than the order shown in Figure 7.5.

Figure 7.5 Definitive Screening Design for Extraction Process

	Methanol	Ethanol	Propanol	Butanol	pH	Time	Yield
1	5	0	0	0	6	1	•
2	0	0	10	0	9	2	•
3	5	10	10	10	9	2	•
4	0	5	0	0	9	1	•
5	10	0	10	0	6	1.5	•
6	10	10	10	0	9	1	•
7	10	5	10	10	6	2	•
8	0	10	0	10	9	1.5	•
9	0	10	5	0	6	2	•
10	5	5	5	5	7.5	1.5	•
11	10	10	0	10	6	1	•
12	10	10	0	0	7.5	2	•
13	10	0	5	10	9	1	•
14	0	10	10	5	6	1	•
15	10	0	0	5	9	2	•
16	0	0	0	10	6	2	•
17	0	0	10	10	7.5	1	•

Comparison with a Fractional Factorial Design

Suppose that you had chosen a traditional screening design instead of the definitive screening design in "Definitive Screening Design". This example compares the two designs in terms of confounding.

1. Select **DOE > Classical > Two Level Screening > Screening Design**.
2. Double-click Y under Response Name and type Yield.
3. Select **Help > Sample Data Library** and open Design Experiment/Extraction Factors.jmp.
4. Click the red triangle next to and select **Load Factors**.

   The factor names and ranges are added to the Factors outline.

5. Click **Continue**.
6. Select **Choose from a list of fractional factorial designs**.
7. Click **Continue**.

   Potential designs appear in the Design List.

Figure 7.6 Screening Design List for Six Continuous Factors

Design List

Choose a design by clicking on its row in the list.

Number Of Runs	Block Size	Design Type	Resolution - what is estimable
8		Fractional Factorial	3 - Main Effects Only
8	4	Fractional Factorial	3 - Main Effects Only
12		Plackett-Burman	3 - Main Effects Only
16		Fractional Factorial	4 - Some 2-factor interactions
16	8	Fractional Factorial	4 - Some 2-factor interactions
16	4	Fractional Factorial	4 - Some 2-factor interactions
16	2	Fractional Factorial	4 - Some 2-factor interactions
32		Fractional Factorial	5+ - All 2-factor interactions
32	16	Fractional Factorial	5+ - All 2-factor interactions
32	8	Fractional Factorial	4 - Some 2-factor interactions
32	4	Fractional Factorial	4 - Some 2-factor interactions
32	2	Fractional Factorial	4 - Some 2-factor interactions
64		Full Factorial	>6 - Full Resolution
64	32	Full Factorial	5+ - All 2-factor interactions

8. Select the sixteen-run fractional factorial design with no blocks, shown highlighted in Figure 7.6.
9. Click **Continue.**
10. Open the **Display and Modify Design > Aliasing of Effects** outline.

Figure 7.7 Aliasing of Effects for Fractional Factorial Design

Aliasing of Effects

Effects	Aliases
Methanol*Ethanol	= pH*Time
Methanol*Propanol	= Butanol*Time
Methanol*Butanol	= Propanol*Time
Methanol*pH	= Ethanol*Time
Methanol*Time	= Ethanol*pH = Propanol*Butanol
Ethanol*Propanol	= Butanol*pH
Ethanol*Butanol	= Propanol*pH

The Aliasing of Effects outline for the 16-run fractional factorial design shows that every two-factor interaction is confounded with at least one other two-factor interaction. In this fractional factorial design, the Ethanol*Time interaction is confounded with Methanol*pH. To determine which interaction is active, you need to run additional trials. If the factors had been entered in a different order, the Ethanol*Time interaction might have been aliased with two other two-factor interactions.

In the section "Definitive Screening Design", you constructed a 17-run definitive screening design. The Color Map on Correlations for this DSD (Figure 7.4) shows that no two-factor interactions are confounded with any other two-factor interactions. For the fractional factorial design, there are seven instances of confounded two-factor interactions. If you suspect that there are active two-factor effects, the DSD is the better choice.

You can conduct a more thorough comparison of the two designs using the Compare Designs platform (DOE > Design Diagnostics > Compare Designs). See "Compare Designs".

Definitive Screening Design with Blocking

Suppose that, due to raw material constraints, the extraction experiment requires that you run it using material from two separate lots. You can generate a definitive screening design with a blocking variable to account for the potential lot variation.

Create the Design

The extraction factors and their settings are given in the data table Extraction Factors.jmp. Generate a definitive screening design with a block:

1. Select **DOE > Definitive Screening > Definitive Screening Design**.
2. Double-click Y under Response Name and type Yield.
3. Select **Help > Sample Data Library** and open Design Experiment/Extraction Factors.jmp.
4. Click the red triangle next to Definitive Screening Design and select **Load Factors**.

   The factor names and ranges are added to the Factors outline.
5. Click **Continue**.

   The Design Options outline opens. Here you can specify a blocking structure.
6. Select **Add Blocks with Center Runs to Estimate Quadratic Effects**.

   Leave **Number of Blocks** set at 2.

   You are recreating the design for the Extraction2 Data.jmp sample data table, which was created without Extra Runs. Although four Extra Runs are strongly recommended, you will not add Extra Runs in this example.
7. Next to **Number of Extra Runs**, select 0.
8. Click **Make Design**.

   The Definitive Screening Design window updates to show a Design outline and a Design Evaluation outline.

   Check that Block has been added to the Factors outline and to the Design.
9. In the Factors outline, Double-click Block and type Lot.

   In the Output Options panel, note that the Run Order is set to Randomize within Blocks.

10. Click **Make Table.**

Note: The runs in your design might appear in a different order than the order shown in Figure 7.8.

Figure 7.8 Definitive Screening Design with Block for Extraction Process

	Lot	Methanol	Ethanol	Propanol	Butanol	pH	Time	Yield
1	1	5	5	5	5	7.5	1.5	•
2	1	5	10	10	10	9	2	•
3	1	5	0	0	0	6	1	•
4	1	10	0	5	0	9	2	•
5	1	0	0	0	10	7.5	2	•
6	1	0	10	5	10	6	1	•
7	1	10	10	10	0	7.5	1	•
8	2	10	5	0	10	9	1	•
9	2	0	10	0	0	9	1.5	•
10	2	0	0	10	5	9	1	•
11	2	10	0	10	10	6	1.5	•
12	2	0	5	10	0	6	2	•
13	2	10	10	0	5	6	2	•
14	2	5	5	5	5	7.5	1.5	•

Notice that run 1 is a center point run in Lot 1 and run 14 is a center point run in Lot 2.

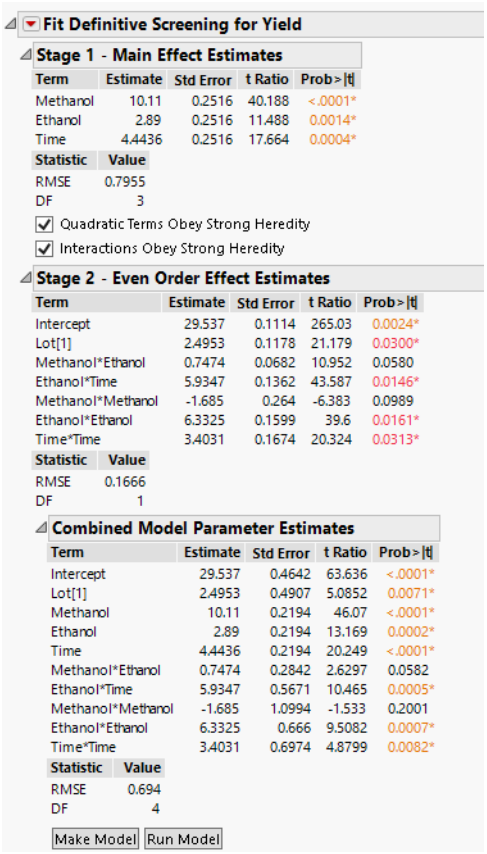
Analyze the Experimental Data

At this point, you conduct your experiment and record your data in the Yield column of the design table (Figure 7.8). The Extraction2 Data.jmp sample data table contains your experimental results. The runs in the Extraction2 Data.jmp sample data table are in a different order than those in Figure 7.8.

To explore all second-order effects, one option is to use All Possible Models regression. Another option is to use forward stepwise regression. However, these standard methods often fail to identify active effects. For this reason, you use the Fit Definitive Screening platform.

1. Select **Help > Sample Data Library** and open Design Experiment/Extraction2 Data.jmp.
2. In the Table panel of the design table, click the green triangle next to the **Fit** Definitive Screening script.

Figure 7.9 Fit Definitive Screening Report



Fit Definitive Screening for Yield

Stage 1 - Main Effect Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Methanol	10.11	0.2516	40.188	<.0001*
Ethanol	2.89	0.2516	11.488	0.0014*
Time	4.4436	0.2516	17.664	0.0004*

Statistic	Value
RMSE	0.7955
DF	3

☑ Quadratic Terms Obey Strong Heredity
☑ Interactions Obey Strong Heredity

Stage 2 - Even Order Effect Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	29.537	0.1114	265.03	0.0024*
Lot[1]	2.4953	0.1178	21.179	0.0300*
Methanol*Ethanol	0.7474	0.0682	10.952	0.0580
Ethanol*Time	5.9347	0.1362	43.587	0.0146*
Methanol*Methanol	-1.685	0.264	-6.383	0.0989
Ethanol*Ethanol	6.3325	0.1599	39.6	0.0161*
Time*Time	3.4031	0.1674	20.324	0.0313*

Statistic	Value
RMSE	0.1666
DF	1

Combined Model Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	29.537	0.4642	63.636	<.0001*
Lot[1]	2.4953	0.4907	5.0852	0.0071*
Methanol	10.11	0.2194	46.07	<.0001*
Ethanol	2.89	0.2194	13.169	0.0002*
Time	4.4436	0.2194	20.249	<.0001*
Methanol*Ethanol	0.7474	0.2842	2.6297	0.0582
Ethanol*Time	5.9347	0.5671	10.465	0.0005*
Methanol*Methanol	-1.685	1.0994	-1.533	0.2001
Ethanol*Ethanol	6.3325	0.666	9.5082	0.0007*
Time*Time	3.4031	0.6974	4.8799	0.0082*

Statistic	Value
RMSE	0.694
DF	4

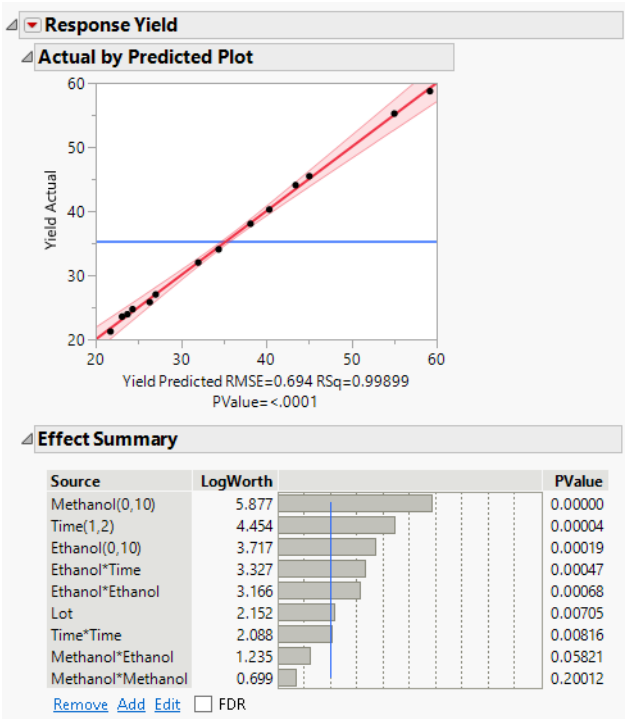
Make Model Run Model

The effects identified by Fit Definitive Screening as potentially active are listed in the Combined Model Parameter Estimates report.

3. Click the **Run Model** button at the bottom of the Combined Model Parameter Estimates report.

   This fits a standard least squares model to the effects identified as potentially active.

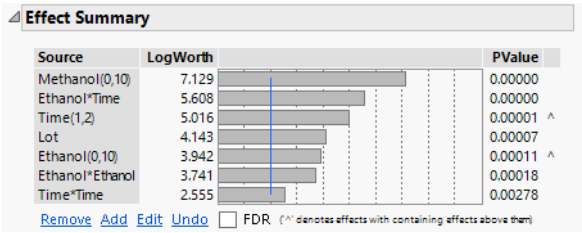
Figure 7.10 Standard Least Squares Report



The Actual by Predicted Plot shows no evidence of lack of fit. The Effect Summary report shows that Methanol*Ethonal and Methanol*Methanol are not significant. You decide to remove these effects from the model.

4. Select Methanol*Ethonal and Methanol*Methanol in the Effect Summary report and click **Remove**.

Figure 7.11 Final Set of Active Effects



The remaining effects are all significant. You conclude that these are the active effects.

Comparison of a Definitive Screening Design with a Plackett-Burman Design

Plackett-Burman designs are an alternative to fractional factorials for screening. However, Plackett-Burman designs have complex aliasing of the main effects by two-factor interactions.

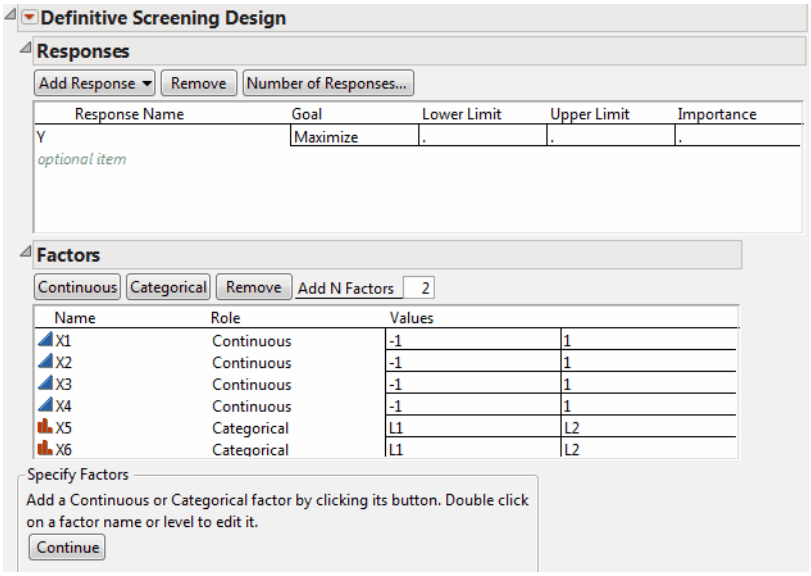
This example shows how to compare a definitive screening with a Plackett-Burman design using the Evaluate Design platform. For an extensive example using the Compare Designs platform, see "Designs of Same Run Size".

The Definitive Screening Design

1. Select **DOE > Definitive Screening > Definitive Screening Design**.
2. Type **4** in the **Add N Factors** box and click **Continuous**.
3. Type **2** in the **Add N Factors** box and click **Categorical.**

   Your window should appear as shown in Figure 7.12.

Figure 7.12 Definitive Screening Dialog with 4 Continuous and 2 Categorical Factors



4. Click **Continue**.

   This example does not require a block. Under the Design Options Outline, check that the No Blocks Required option is selected.

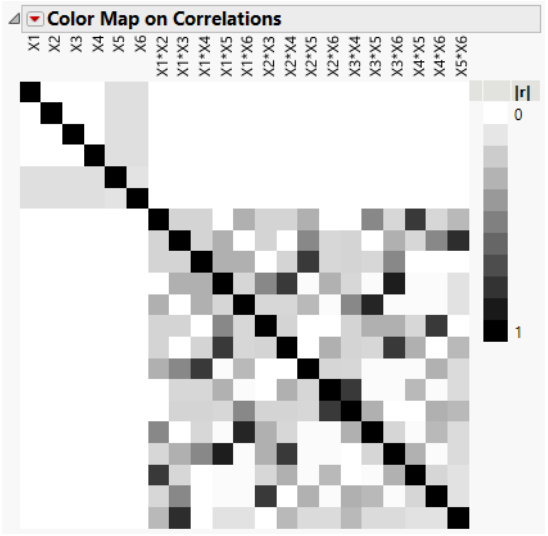
   In order to compare designs of approximately equal sizes. do not add Extra Runs.
5. Next to **Number of Extra Runs**, select 0.

6. Click **Make Design**.

   The design that is generated has 14 runs.

7. Open the **Design Evaluation > Color Map On Correlations** outline.

Figure 7.13 Color Map for Definitive Screening Design



   Notice that the categorical main effects have small correlations with each other and with the continuous factors' main effects. These correlations lead to a small reduction in the precision of the estimates.

8. Do not close your Definitive Design Screening window until you compare the color map with that of the Plackett-Burman design, below.

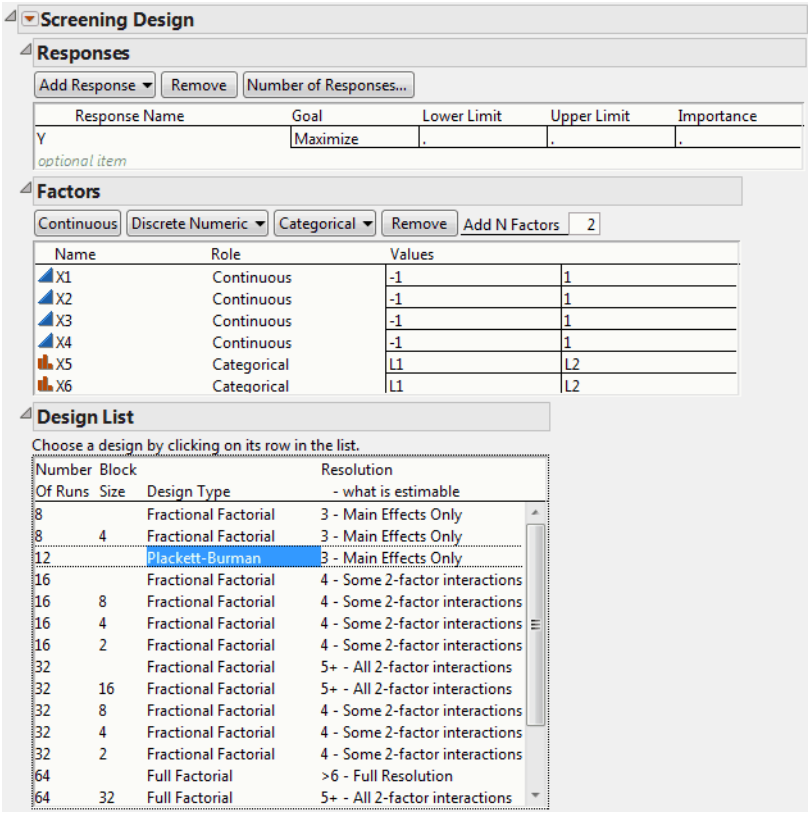
The Plackett-Burman Design

Now create a Plackett-Burman design using the same factor structure.

1. Select **DOE > Classical > Two Level Screening > Screening Design**.
2. Type **4** in the **Add N Factors** box and click **Continuous**.
3. Type **2** in the **Add N Factors** box and click **Categorical > 2 Level.**
4. Click **Continue**.
5. Select **Choose from a list of fractional factorial designs** and click **Continue**.

   Potential designs appear in the Design List.

Figure 7.14 Plackett-Burman Design



6. Select the 12 run Plackett-Burman design (Figure 7.14).
7. Click **Continue.**
8. Open the **Design Evaluation > Color Map On Correlations** outline.

   Compare the color map for the 12-run Plackett-Burman design to the color map for the 14-run definitive screening design.

Figure 7.15 Plackett-Burman Correlations (left) and Definitive Screening Correlations (right)

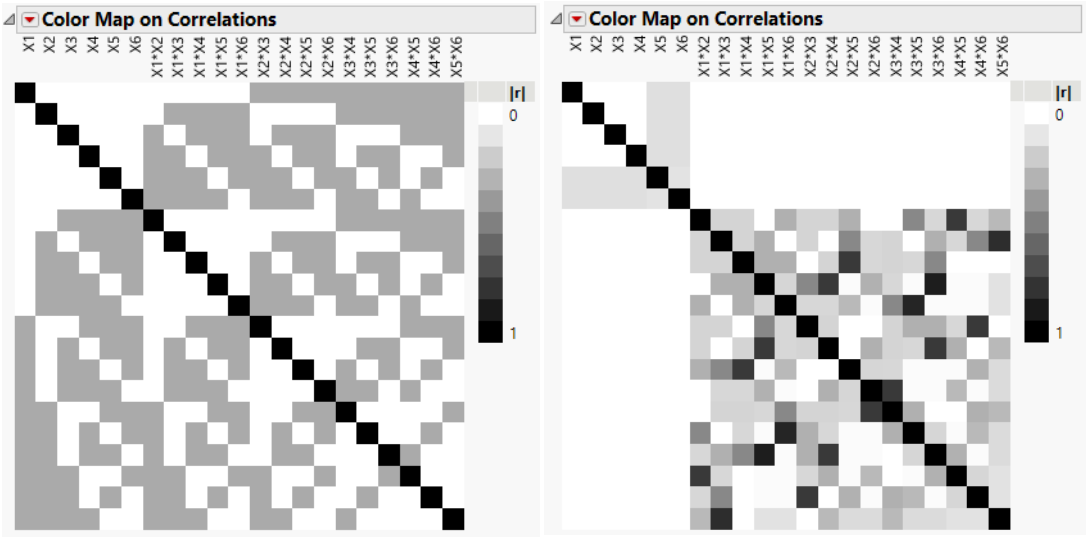


Figure 7.15 shows both color maps, but shows only the portion of the Plackett-Burman color map that involves main effects and two-way interactions. (To construct the color map for the Plackett-Burman design without the three-way interactions, construct the design. Then obtain the color map using Evaluate Design.)

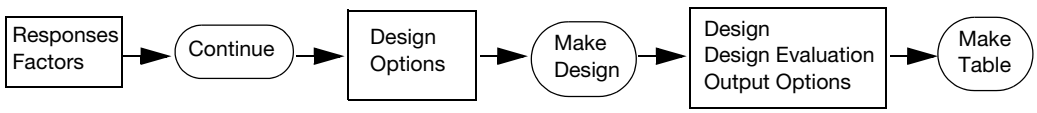
In the color map for the Plackett-Burman design on the left, you see that most two-factor interactions are correlated with main effects. This means that any non-negligible two-factor interaction will bias several main effects. This can lead to a failure to identify an active main effect or the false conclusion that an inactive main effect is active.

Contrast this with the color map for the definitive screening design on the right. With only two additional runs, the definitive screening design trades off a small increase in the variance of the main effects for complete independence of main effects and two-factor interactions.

Definitive Screening Design Window

The definitive screening design window updates as you work through the design steps. See "The DOE Workflow: Describe, Specify, Design". The outlines, separated by buttons that update the outlines, follow the flow in Figure 7.16.

Figure 7.16 Definitive Screening Design Flow

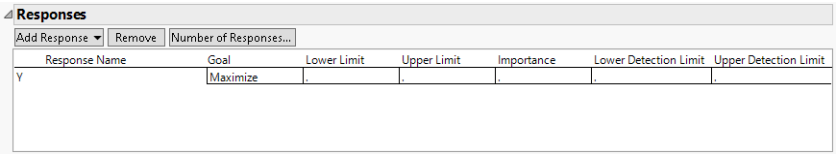


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 7.17 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.
- A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

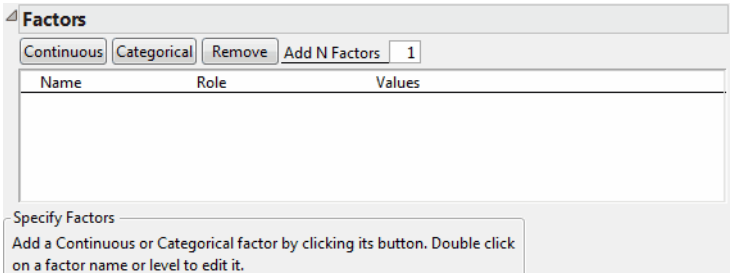
The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

Factors

Add factors in the Factors outline.

Figure 7.18 Factors Outline



The factors outline contains the following buttons.

Continuous Enters the number of continuous factors specified in **Add N Factors**.

Categorical Enters the number of nominal factors specified in **Add N Factors.**

Remove Removes the selected factors.

Add N Factors Adds multiple factors of a given type. Enter the number of factors to add and click Continuous or Categorical. Repeat **Add N Factors** to add multiple factors of different types.

Tip: When you have completed your Factors panel, select Save Factors from the red triangle menu. This saves the factor names and values in a data table that you can later reload. See "Definitive Screening Design Options".

The Factors outline contains the following columns:

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role Specifies the Design Role of the factor. The Design Role column property for the factor is saved to the data table. This property ensures that the factor type is modeled appropriately.

Values The experimental settings for the factors. To insert Values, click the default values and type the desired values.

Editing the Factors Outline

In the Factors outline, note the following:

- To edit a factor name, double-click the factor name.
- To edit a value, click the value in the Values column.

Factor Types

Continuous Numeric data types only. A continuous factor is a factor that you can conceptually set to any value between the lower and upper limits you supply, given the limitations of your measurement system.

Categorical Either numeric or character data types with two levels. For a categorical factor, the value ordering is the order of the values, as entered from left to right. This ordering is saved in the Value Order column property after the design data table is created.

Factor Column Properties

For each factor, various column properties are saved to the data table.

Design Role Each factor is assigned the Design Role column property. The Role that you specify in defining the factor determines the value of its Design Role column property. When you add a block under Design Options, that factor is assigned the Blocking value. The Design Role property reflects how the factor is intended to be used in modeling the experimental data. Design Role values are used in the Augment Design platform.

Factor Changes Each factor is assigned the Factor Changes column property with a setting of Easy. In definitive screening designs, it is assumed that factor levels can be changed for each experimental run. Factor Changes values are used in the Evaluate Design and Augment Design platforms.

Coding If the Design Role is Continuous, the Coding column property for the factor is saved. This property transforms the factor values so that the low and high values correspond to –

1 and +1, respectively. The estimates and tests in the Fit Least Squares report are based on the transformed values.

Value Order If the Design Role is Categorical or Blocking, the Value Order column property for the factor is saved. This property determines the order in which levels of the factor appear.

RunsPerBlock Indicates the number of runs in each block. When a Block is selected in the Design Options outline and you then click Make Design, a factor with the default name Block is added to the Factors list. The RunsPerBlock column property is saved for that factor.

Design Options

The Design Options outline enables you to specify the blocking structure, the number of blocks, and the number of extra runs. Block effects are orthogonal to the main effects. Block sizes need not be equal.

The outline contains the following options:

No Blocks Required Indicates that the design will not contain a blocking factor. This is the default selection.

Add Blocks with Center Runs to Estimate Quadratic Effects Adds the number of blocks specified in the Number of Blocks text box. Constructs a design where block effects are orthogonal to main effects and where the model consisting of all main and quadratic effects is estimable. See "Add Blocks with Center Runs to Estimate Quadratic Effects"

Add Blocks without Extra Center Runs Adds the number of blocks specified in the Number of Blocks text box. Adds only as many center runs as required by the design structure. Constructs a design where block effects are orthogonal to main effects, but the model consisting of all main effects and quadratic effects might *not* be estimable. See "Add Blocks without Extra Center Runs".

Note: Use the **Add Blocks without Extra Center Runs** option only if you can assume that not all quadratic effects are important.

Number of Blocks Indicates the number of blocks to add. The number of blocks that you can add ranges from two to the number of factors.

Number of Extra Runs Adds non-center runs that enable you to conduct *effective model selection*. See "Extra Runs" and "Effective Model Selection for DSDs".

Tip: Adding runs to your design with the Extra Runs option enhances your ability to detect effects in the presence of many active effects. The recommended number of Extra Runs is four, which dramatically improves the power of the design to identify active second-order effects.

Make Design Generates the design, presents it in the Design outline, and provides evaluation information in the Design Evaluation outline. The Output Options panel also appears, enabling you to create the design table.

Blocking in Definitive Screening Designs

This section describes the two blocking options:

- "Add Blocks with Center Runs to Estimate Quadratic Effects"
- "Add Blocks without Extra Center Runs"

Add Blocks with Center Runs to Estimate Quadratic Effects

Note: For more information about the construction and properties of blocked designs that estimate quadratic effects, see Jones and Nachtsheim (2016). The paper also contains information about treating the blocks as random effects.

The **Add Blocks with Center Runs to Estimate Quadratic Effects** option constructs a design with these properties:

- Block effects are orthogonal to main effects.
- The model consisting of all main and quadratic effects is estimable.

If a design contains only continuous factors, a blocked design for k factors having these properties can be constructed:

- Remove the center run from the DSD design for k factors.
- Assign conference matrix foldover pairs to the same block.
- Add one center run to each block.

When some factors are categorical, the **Add Blocks with Center Runs to Estimate Quadratic Effects** option adds pairs of center runs within certain blocks. This structure ensures orthogonality and the ability to estimate all main and quadratic effects.

Because the only requirement on block size is that a block contains a foldover pair, the number of blocks can range from 2 to k, if k is even and from 2 to $k+1$, if k is odd. See "Conference Matrices and the Number of Runs". JMP attempts to construct blocks of equal size.

Add Blocks without Extra Center Runs

The **Add Blocks without Extra Center Runs** option constructs a design that has a single center run when all factors are continuous and two center runs when some factors are categorical. The resulting design has these properties:

- Block effects are orthogonal to main effects.
- Block effects might be confounded with a linear combination of quadratic effects. This implies that the model consisting of all main and quadratic effects might *not* be estimable.

For this reason, use this option only if you can assume that some quadratic effects are negligible.

Construct a blocked design for *k* factors without extra center runs:

- Assign conference matrix foldover pairs to the same block.
- If all factors are continuous, assign the single center run to a single block.
- If there are categorical factors, the unblocked definitive screening design requires the addition of two center runs to the foldover pairs defined by the conference matrix. See "Conference Matrices and the Number of Runs". To construct the blocked design without extra center runs, these two center runs are added to a single block.

Because the only requirement on block size is that a block contains a foldover pair, the number of blocks can range from 2 to *k*, if *k* is even and from 2 to *k+1*, if *k* is odd. See "Conference Matrices and the Number of Runs". JMP attempts to construct blocks of equal size.

Design

The Design outline shows the runs for the definitive screening design. The runs are given in a standard order. To change the run order for your design table, you can select Run Order options in the Output Options panel before generating the table.

Note: Definitive screening designs for four or fewer factors are based on a five-factor design. See "Definitive Screening Designs for Four or Fewer Factors".

Design Evaluation

Note: The Design Evaluation outline is not shown for Cotter designs.

The Design Evaluation outline provides a number of ways to evaluate the properties of the generated design. Open the Design Evaluation outline to see the following options:

Power Analysis Enables you to explore your ability to detect effects of given sizes.

Prediction Variance Profile Shows the prediction variance over the range of factor settings.

Fraction of Design Space Plot Shows how much of the model prediction variance lies below (or above) a given value.

Prediction Variance Surface Shows a surface plot of the prediction variance for any two continuous factors.

Estimation Efficiency For each parameter, gives the fractional increase in the length of a confidence interval compared to that of an ideal (orthogonal) design, which might not exist. Also gives the relative standard error of the parameters.

Alias Matrix Gives coefficients that indicate the degree by which the model parameters are biased by effects that are potentially active, but not in the model.

Color Map on Correlations Shows the absolute correlation between effects on a plot using an intensity scale.

Design Diagnostics Indicates the optimality criterion used to construct the design. Also gives efficiency measures for your design.

Note: The model used for the design diagnostics contains all main effects and two-factor interactions when all two-factor interactions are estimable. Otherwise, the model contains all main effects.

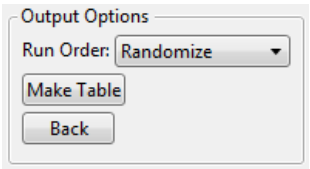
For more details about the Design Evaluation panel, see "Design Evaluation".

Output Options

Use the Output Options panel to perform the following tasks:

- specify the order for the runs in the design data table
- construct the design table
- return to a previous point in the Definitive Screening Design window

Figure 7.19 Output Options Panel



The Output Options panel contains these options:

Run Order

The **Run Order** options determine the order of the runs in the design table. Choices include the following:

Keep the Same Rows in the design table are in the same order as in the Design outline.

Sort Left to Right Columns in the design table are sorted from left to right.

Randomize Rows in the design table are in random order.

Sort Right to Left Columns in the design table are sorted from right to left.

Randomize within Blocks Rows in the design table are in random order within the blocks.

Make Table

Click **Make Table** to construct the Definitive Screening Design data table.

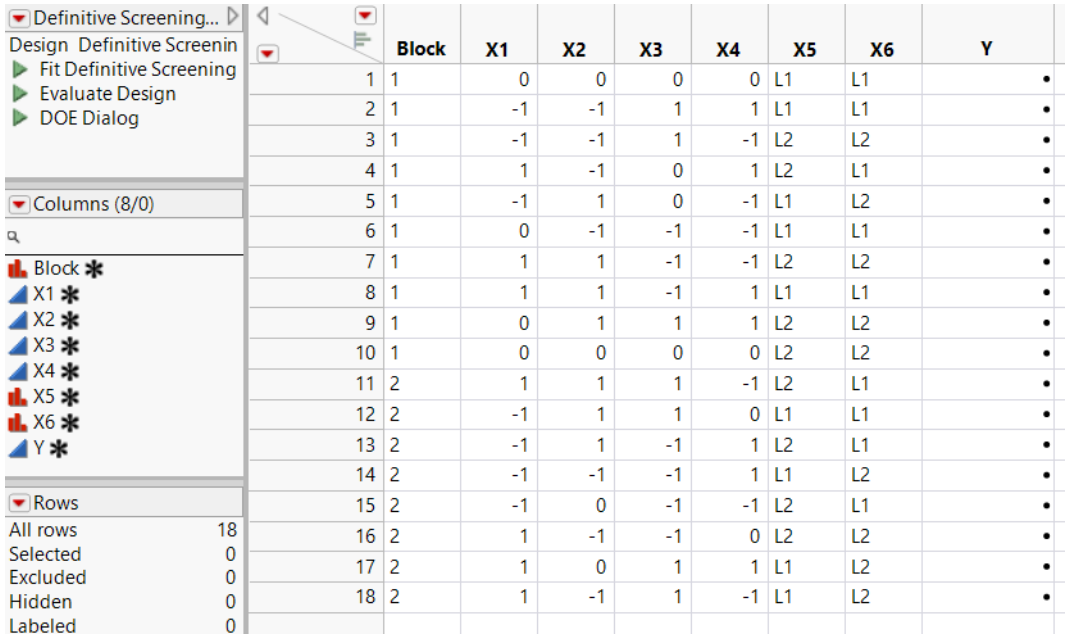
In the Definitive Screening Design table, the Table panel (in the upper left) contains the following scripts. To run a script, click the green triangle next to the script name.

Fit Definitive Screening Runs the **DOE** > **Definitive Screening > Fit Definitive Screening** platform. See "The Fit Definitive Screening Platform".

Evaluate Design Runs the **DOE** > **Design Diagnostics** > **Evaluate Design** platform. See "Evaluate Designs".

DOE Dialog Re-creates the Definitive Screening Design window that you used to generate the design table.

Figure 7.20 Definitive Screening Design Table Showing Scripts



	Block	X1	X2	X3	X4	X5	X6	Y
1	1	0	0	0	0	L1	L1	•
2	1	-1	-1	1	1	L1	L1	•
3	1	-1	-1	1	-1	L2	L2	•
4	1	1	-1	0	1	L2	L1	•
5	1	-1	1	0	-1	L1	L2	•
6	1	0	-1	-1	-1	L1	L1	•
7	1	1	1	-1	-1	L2	L2	•
8	1	1	1	-1	1	L1	L1	•
9	1	0	1	1	1	L2	L2	•
10	1	0	0	0	0	L2	L2	•
11	2	1	1	1	-1	L2	L1	•
12	2	-1	1	1	0	L1	L1	•
13	2	-1	1	-1	1	L2	L1	•
14	2	-1	-1	-1	1	L1	L2	•
15	2	-1	0	-1	-1	L2	L1	•
16	2	1	-1	-1	0	L2	L2	•
17	2	1	0	1	1	L1	L2	•
18	2	1	-1	1	-1	L1	L2	•

Back

The Back button takes you back to where you were before clicking Make Design. You can make changes to the previous outlines and regenerate the design.

Definitive Screening Design Options

The red triangle menu in the Definitive Screening Design platform contains these options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

– A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Save X Matrix Saves the Moments Matrix and Model Matrix to table scripts in the design data table. These scripts contain the moments and design matrices. See "Save X Matrix".

Caution: For a design with nominal factors, the Model Matrix saved by the Save X Matrix option is *not* the coding matrix used in fitting the linear model. You can obtain the coding matrix used for fitting the model by selecting the option Save Columns > Save Coding Table in the Fit Model report that you obtain when you select **run model** from the definitive model fit.

Save Script to Script Window Creates the script for the design that you specified in the Definitive Screening Design window and saves it in an open script window.

Simulate Responses

When you click Make Table to create your design table, the Simulate Responses option does the following for each response:

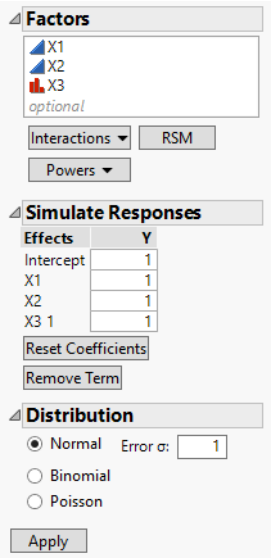
- It adds random response values to the response column in your design table.
- It adds a new a column containing a simulation model formula to the design table. The formula and values are based on a main effects model.

A Model window opens where you can add and remove effects to define a model, specify parameter values, and select a response distribution for simulation. When you click Apply in the Model window, each column containing a simulation model formula is updated.

Control Window

Figure 7.21 shows the Model window for a design with two continuous factors (X1 and X2) and one two-level categorical factor (X3).

Figure 7.21 Simulate Responses Control Window



The window has three outlines:

- Factors
- Simulate Responses
- Distribution

The initial Simulate Responses outline shows terms for a main effects model with values of 1 for all coefficients. The Distribution outline shows a Normal distribution with error standard deviation equal to 1. If you have set Anticipated Coefficients as part of Power Analysis under Design Evaluation in the DOE window, then the initial values in the Simulate Responses outline are the values that you specified as Anticipated Coefficients and Anticipated RMSE (Error Std) in the Power Analysis outline.

Factors

Add terms to the simulation model using the Factors outline.

Interactions Select factors in the list. Then select the interaction order from the Interactions menu. Those interactions are added to the list of Effects in the Simulate Responses outline.

RSM Adds all possible response surface terms to the list of Effects in the Simulate Responses outline.

Powers Select factors in the list. Then select the order from the Powers menu. Those powers are added to the list of Effects in the Simulate Responses outline.

Simulate Responses

To specify a model for simulated values, do the following:

1. For each term in the list of Effects, enter coefficients for the linear model used to simulate the response values. These define a linear function, $L(\mathbf{x}, \beta) = \mathbf{x}'\beta$. See the Simulate Responses outline in Figure 7.21:
   - The vector **x** consists of the terms that define the effects listed under Effects.
   - The vector β is the vector of model coefficients that you specify under Y.
2. Under Distribution, select a response distribution.
3. Click **Apply**. A <Y> Simulated column containing simulated values and their formula is added to the design table, where Y is the name of the response column.

Reset coefficients Sets all coefficients to 0.

Remove Term Remove terms from the list of Effects. Select the effects to remove and click Remove Term. Note that you cannot remove main effects.

Distribution

Choose from one of the available distributions in the Simulate Responses window:

Normal Simulates values from a normal distribution. Enter a value for Error σ, the standard deviation of the normal error distribution. If you have designated factors to have Changes of Hard in the Factors outline, you can enter a value for Whole Plots σ, the whole plot error. If you have designated factors to have Changes of Hard and Very Hard, you can enter values for both the subplot and whole plot errors. When you click Apply, random values and a formula containing a random response vector based on the model are entered in the column <Y> Simulated.

Binomial Simulates values from a binomial distribution. Enter a value for N, the number of trials. Random integer values are generated according to a binomial distribution based on N trials with probability of success $1/(1 + \exp(-L(\mathbf{x}, \beta)))$. When you click Apply, random values and their formula are entered in the column <Y> Simulated. A column called N Trials that contains the value N is also added to the data table.

Poisson Simulates random integer values according to a Poisson distribution with parameter $\exp(L(\mathbf{x}, \beta))$. When you click Apply, random values and their formula are entered in the column <Y> Simulated.

Note: You can set a preference to simulate responses every time you click Make Table. To do so, select **File > Preferences > Platforms > DOE**. Select **Simulate Responses**.

Technical Details

- "Structure of Definitive Screening Designs"
- "Analysis of Experimental Data"

Structure of Definitive Screening Designs

Figure 7.22 shows an example of a definitive design with eight continuous factors. and four Extra Runs that correspond to fake factors. Notice the following:

- Each pair of rows is a foldover pair; each even-numbered row is -1 times the previous row. The foldover aspect of the design removes the confounding of two-factor interactions and main effects.
- Each factor is set at its center value for three runs; this, together with the design's construction, makes all quadratic effects estimable.
- Rows 17 through 20 are the Extra Runs that correspond to the fake factors.
- Adding the center run in the last row enables you to fit a model that includes an intercept and all main and quadratic effects.

This structure is typical of definitive screening designs for continuous factors.

Figure 7.22 Definitive Screening Design for Eight Continuous Factors

Design

Run	X1	X2	X3	X4	X5	X6	X7	X8
1	0	1	1	1	1	1	1	1
2	0	-1	-1	-1	-1	-1	-1	-1
3	1	0	-1	-1	-1	-1	1	1
4	-1	0	1	1	1	1	-1	-1
5	1	-1	0	-1	1	1	-1	-1
6	-1	1	0	1	-1	-1	1	1
7	1	-1	-1	0	1	1	1	1
8	-1	1	1	0	-1	-1	-1	-1
9	1	-1	1	1	0	-1	-1	1
10	-1	1	-1	-1	0	1	1	-1
11	1	-1	1	1	-1	0	1	-1
12	-1	1	-1	-1	1	0	-1	1
13	1	1	-1	1	-1	1	0	-1
14	-1	-1	1	-1	1	-1	0	1
15	1	1	-1	1	1	-1	-1	0
16	-1	-1	1	-1	-1	1	1	0
17	1	1	1	-1	-1	1	-1	1
18	-1	-1	-1	1	1	-1	1	-1
19	1	1	1	-1	1	-1	1	-1
20	-1	-1	-1	1	-1	1	-1	1
21	0	0	0	0	0	0	0	0

Conference Matrices and the Number of Runs

Definitive screening designs in JMP are constructed using *conference matrices* (Xiao et al., 2012). A conference matrix is an m x m matrix $\mathbf{C}$ where m is even. The matrix $\mathbf{C}$ has *0*s on the diagonal, off-diagonal entries equal to *1* or *–1*, and satisfies

$$\mathbf{C}'\mathbf{C} = (m-1)\mathbf{I}_{mxm} \quad .$$

Note: For certain even values of m, it is not known if a conference matrix exists.

Suppose that the number of factors, k, is five or larger. For the case of $k \leq 4$ factors, see "Definitive Screening Designs for Four or Fewer Factors".

Consider the case of k continuous factors and suppose that a conference matrix is available.

- When k is even, the k x k conference matrix is used to define k runs of the design. Its negative, $-\mathbf{C}$, defines the foldover runs. A center point is added to the design to ensure that a model containing an intercept, main effects, and quadratic effects is estimable. So, for k even, the minimum number of runs in the definitive screening design is $2k + 1$.
- When k is odd, a $(k+1)$ x $(k+1)$ conference matrix is used, with its last column deleted. A center point is added. Thus, for k odd, the minimum number of runs in the screening design is $2k + 3$.

A similar procedure is used when some factors are categorical and a conference matrix is available. See Jones and Nachtsheim (2013).

- Instead of a single center point, two additional runs are required. These two runs are center runs where all continuous factors are set at their middle values.
- When there are k factors and k is even, the number of runs in the design is $2k + 2$.
- When k is odd, the number of runs is $2k + 4$.

For those values of m for which a conference matrix is not available, a definitive screening design can be constructed using the next largest conference matrix. As a result, the required number of runs might exceed $2k + 3$, in the continuous case, and $2k + 4$, in the categorical case.

Extra Runs

Extra runs are constructed using fictitious, or *fake*, factors. Adding f fake factors to a design results in 2^f additional runs.

Denote the number of factors in your experimental study by k. Four or eight extra runs can be added to a design. Extra runs are constructed by creating a design for $k + f$ factors, as described in "Conference Matrices and the Number of Runs", and then dropping the last f columns. As few as four extra runs can be highly beneficial in model selection.

For information about how extra runs are used, see "Effective Model Selection for DSDs".

Definitive Screening Designs for Four or Fewer Factors

Definitive screening designs for four or fewer factors are constructed using the five-factor definitive screening design as a base. This is because designs for $k \leq 4$ factors constructed strictly according to the conference matrix approach have undesirable properties. In particular, it is difficult to separate second-order effects.

If you specify $k \leq 4$ factors, a definitive screening design for five factors is constructed and unnecessary columns are dropped. For this reason, the number of runs for an unblocked design with $k \leq 4$ factors is 13 if all factors are continuous or 14 if some factors are categorical.

Analysis of Experimental Data

In general, you want to fit a model that permits the possibility that two-way interactions are active. You also might want to include pure quadratic terms in your model. You might want to postulate a full second-order model, or you might want to specify an a priori model containing only certain second-order terms.

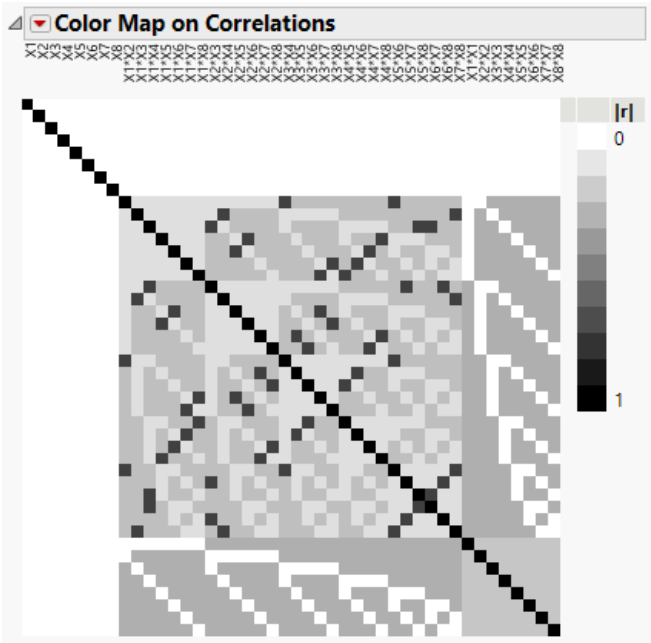
Two-Way Interactions

In fitting such a model, you need to be mindful of two facts:

- two-way interaction effects and quadratic effects are often correlated
- two-way interaction and quadratic effects cannot *all* be estimated simultaneously

Figure 7.23 shows a Color Map on Correlations for the design with eight continuous factors shown in Figure 7.22. The color map is for a full quadratic design. The eight pure quadratic effects are listed to the far right. You can construct this plot by using DOE > Design Diagnostics > Evaluate Design and entering the appropriate terms into the Alias Terms list. See "Alias Terms".

Figure 7.23 Color Map on Correlations for Full Quadratic Model



Hover over the cells of the color map in order to see the absolute correlations between effects. You see that main effects are uncorrelated with all two-way interaction and pure quadratic effects. You also see that none of the effects are completely confounded with other effects because the only black cells are on the main diagonal. But note that some of the absolute correlations between two-factor interactions are substantial, with some at 0.75. Note also that absolute correlations between two-factor interactions and pure quadratic effects are either 0 or 0.3118.

If only main and pure quadratic effects are active, you can fit a saturated model that contains main effects and quadratic effects. This model will result in effect estimates that are unbiased, assuming no active three-way or higher order effects.

Because of the correlations involving second-order effects, you must be careful in fitting a model with two-way interactions. Analysis methodologies include the following, where the first is preferred:

- The method of Efficient Model Selection performs well, especially if many effects are likely to be active. See "Effective Model Selection for DSDs".
- Forward stepwise or all possible subsets regression performs adequately if the following conditions hold:
  - The number of active effects is no more than half the number of runs.
  - There are at most two active two-way interactions or at most one active quadratic effect.

See "Forward Stepwise Regression or All Possible Subsets Regression".

Forward Stepwise Regression or All Possible Subsets Regression

This method consists of first specifying a full response surface model. Then do one of the following:

- Use forward stepwise regression with the Stopping Rule set to Minimum AICc and the Rules set to Combine to ensure model heredity.
- Use All Possible Models regression, where you select the option that imposes the heredity restriction and use the AICc criterion for model selection.

You cannot fit the full response surface model because the number of runs is less than the number of parameters. So your analysis depends on the assumption of effect sparsity, where you assume that the number of active effects is less than the number of runs. This approach has some limitations:

- If the number of active effects exceeds half the number of runs, both stepwise and all possible models regression have difficulty finding the correct model.
- The power of tests to detect moderate quadratic effects is low. A quadratic effect must exceed three times the error standard deviation for the power to exceed 0.9.
- Because of effect confounding, several models might be equivalent. Additional runs will be necessary to resolve the confounding.

Chapter 8

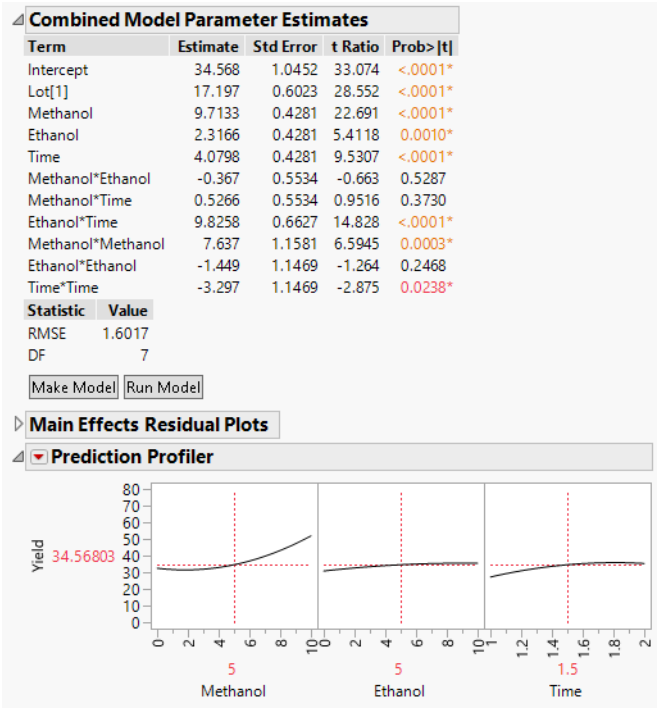
The Fit Definitive Screening Platform

Analyze Data from Definitive Screening Experiments

Use the Fit Definitive Screening platform to analyze definitive screening designs (DSDs) using a methodology called *Effective Model Selection for DSDs*. This methodology takes advantage of the unique structure of definitive screening designs.

Standard model selection methods applied to DSDs can fail to identify active effects. To identify active main effects and second-order effects, the Effective Model Selection for DSDs algorithm leverages the structure of DSDs.

Figure 8.1 Fit Definitive Screening Results



Contents

Overview of the Fit Definitive Screening Platform 277
Identification of Active Effects in DSDs 277
Effective Model Selection for DSDs 277
Example of the Fit Definitive Screening Platform 278
Fit the Model 278
Examine Results 279
Reduce the Model 280
Launch the Fit Definitive Screening Platform 282
Fit Definitive Screening Report 282
Stage 1 - Main Effect Estimates 283
Stage 2 - Even Order Effect Estimates 284
Combined Model Parameter Estimates 285
Main Effects Residual Plots 285
Prediction Profiler 286
Fit Definitive Screening Platform Options 286
Technical Details for the Fit Definitive Screening Platform 286
The Effective Model Selection for DSDs Algorithm 286

Overview of the Fit Definitive Screening Platform

The Fit Definitive Screening platform analyzes definitive screening designs (DSDs) using a methodology that takes advantage of their special structure. The methodology is called *Effective Model Selection for DSDs*. If you created your design in JMP, the design table contains a script called Fit Definitive Screening that automatically runs an analysis using the Effective Model Selection for DSDs methodology.

Identification of Active Effects in DSDs

DSDs are three-level designs that are valuable for identifying main effects and second-order effects in a single experiment. A minimum run-size DSD is capable of correctly identifying active terms with high probability if the number of active effects is less than about half the number of runs and if the effects sizes exceed twice the standard deviation.

However, by augmenting a minimum run-size DSD with four or more properly selected runs, you can identify substantially more effects with high probability. These runs are called *Extra Runs*, and correspond to fictitious inactive factors, called *fake factors*. For information about Extra Runs, see "Structure of Definitive Screening Designs".

Extra Runs substantially increase the design's ability to detect second-order effects. For this reason, Jones and Nachtsheim (2016) strongly encourage the inclusion of at least four Extra Runs.

Effective Model Selection for DSDs

When standard model selection methods are applied to DSDs, they can fail to identify active effects. See Errore et al. (2017). Also, standard selection methods do not leverage the structure of DSDs. The Fit Definitive Screening platform uses the Effective Model Selection for DSDs approach, which takes full advantage of the structure of the DSD.

Jones and Nachtsheim (2016) report on simulation studies using Effect Model Selection for DSDs as well as standard approaches. Denote by c the sum of the number of factors and the number of fake factors in a DSD. In many situations, if the number of active main effects exceeds three, then up to $c/2$ active second-order effects can be reliably identified. Assuming strong effect heredity, if there are three or fewer active main effects, then all active second-order effects can be reliably identified. *Reliable identification* means that the ratio of the absolute value of the coefficient to the error standard deviation exceeds three and that the power to detect the effect exceeds 0.80.

The Fit Definitive Screening platform default settings assume *strong effect heredity*. Strong effect heredity means that the A*B interaction can only be considered for inclusion in the model if both A and B have been included. Strong effect heredity requires that all lower-order components of a model effect be included in the model. In identifying active second-order effects, the algorithm uses strong effect heredity and the results cited earlier about how many active second-order effects can be reliably identified.

In a DSD, main effects and second-order effects are orthogonal to each other. The Effective Model Selection for DSDs approach takes advantage of this fact. The linear space of the response is separated into the subspace spanned by the main effects and the orthogonal complement of this subspace. Miller and Sitter (2005) refer to the linear subspace spanned by the main effects as the *odd space*, because it contains all the information about odd effects: main effects, 3-factor effects, 5-factors effects, and so on. They refer to its orthogonal complement as the *even space*, because it contains all the information about even effects: the intercept, 2-factor effects, 4-factor effects, and so on.

Fit Definitive Screening follows this thinking. The subspace spanned by the main effects is the odd space. Its orthogonal complement, the even space, contains the second-order effects and the block variable, if one exists. For more information about the algorithm, see "The Effective Model Selection for DSDs Algorithm" and Jones and Nachtsheim (2016).

Example of the Fit Definitive Screening Platform

The design in the sample data table Extraction 3 Data.jmp is a definitive screening design for six factors in two blocks. The **Add Blocks with Center Runs to Estimate Quadratic Effects** option was selected and 4 Extra Runs were added in the design generation. The resulting design has 18 runs. This design is used to explore the analysis of a definitive screening design.

- "Fit the Model"
- "Examine Results"
- "Reduce the Model"

Note: For an example of a design with continuous and categorical factors, open the sample data file Peanut Data.jmp and run the Fit Definitive Screening table script.

Fit the Model

1. Select **Help > Sample Data Library** and open Design Experiment/Extraction 3 Data.jmp.
2. Select **DOE > Definitive Screening > Fit Definitive Screening**.
3. Select Yield and click **Y**.

4. Select Lot through Time and click **X**.
5. Click OK.

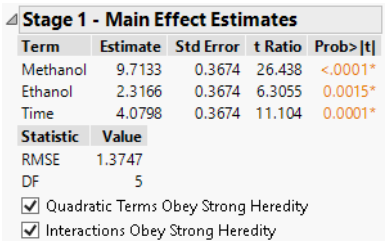
The fit performs a two-stage analysis. For more information about the algorithm, see "Technical Details for the Fit Definitive Screening Platform".

Examine Results

Stage 1: Main Effect Estimates

Stage 1 determines which main effects are likely to be active.

Figure 8.2 Stage 1 Report for Main Effects



Stage 1 - Main Effect Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Methanol	9.7133	0.3674	26.438	<.0001*
Ethanol	2.3166	0.3674	6.3055	0.0015*
Time	4.0798	0.3674	11.104	0.0001*

Statistic	Value
RMSE	1.3747
DF	5

☑ Quadratic Terms Obey Strong Heredity
☑ Interactions Obey Strong Heredity

Note: The fake factors do not appear in the design or as factors in the analysis.

A two-degree-of-freedom error sum of squares is computed from the four runs corresponding to the two fake factors. Because the fake factors are, by construction, inactive, this estimate of error variance is unbiased. For each main effect, the main effects response Y_{ME} is tested against this estimate. In this example, three factors, Methanol, Ethanol, and Time, have *p*-values smaller than the threshold value and are retained as active. For more information about the threshold values, see "Stage 1 Methodology".

The variability from the three inactive factors, Propanol, Butanol, and pH, is pooled with the fake factor sum of squares to produce the five-degree-of-freedom RMSE statistic shown in Figure 8.2.

Stage 2: Even Order Effect Estimates

Stage 2 uses guided subset selection to arrive at a list of second-order effects that are likely to be active. Interactions and quadratic terms are second-order or even order effects.

Figure 8.3 Stage 2 Report for Even-Order Effects

Stage 2 - Even Order Effect Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	34.568	1.3459	25.683	0.0015*
Lot[1]	17.197	0.7757	22.171	0.0020*
Methanol*Ethanol	-0.367	0.7127	-0.515	0.6581
Methanol*Time	0.5266	0.7127	0.7389	0.5369
Ethanol*Time	9.8258	0.8534	11.514	0.0075*
Methanol*Methanol	7.637	1.4914	5.1208	0.0361*
Ethanol*Ethanol	-1.449	1.477	-0.981	0.4299
Time*Time	-3.297	1.477	-2.232	0.1552

Statistic	Value
RMSE	2.0626
DF	2

Because three main effects are identified as active in Stage 1, the guided subset selection procedure for active second-order effects can continue until all second-order effects are included. Because all six second-order effects are reported in Stage 2, it follows that the Stage 2 RMSE remained larger than the Stage 1 RMSE. See "Stage 2 Methodology".

The two-degree-of-freedom RMSE given in the Stage 2 report is the error estimate obtained from the final subset of all six second-order effects.

Combined Results

The effects selected for the model are listed in the Combined Model Parameter Estimates report.

Figure 8.4 Combined Model Parameter Estimates Report

Combined Model Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	34.568	1.0452	33.074	<.0001*
Lot[1]	17.197	0.6023	28.552	<.0001*
Methanol	9.7133	0.4281	22.691	<.0001*
Ethanol	2.3166	0.4281	5.4118	0.0010*
Time	4.0798	0.4281	9.5307	<.0001*
Methanol*Ethanol	-0.367	0.5534	-0.663	0.5287
Methanol*Time	0.5266	0.5534	0.9516	0.3730
Ethanol*Time	9.8258	0.6627	14.828	<.0001*
Methanol*Methanol	7.637	1.1581	6.5945	0.0003*
Ethanol*Ethanol	-1.449	1.1469	-1.264	0.2468
Time*Time	-3.297	1.1469	-2.875	0.0238*

Statistic	Value
RMSE	1.6017
DF	7

The RMSE and degrees of freedom given at the bottom of the report are the usual standard least squares quantities. Use these effects as potential factors for your final model.

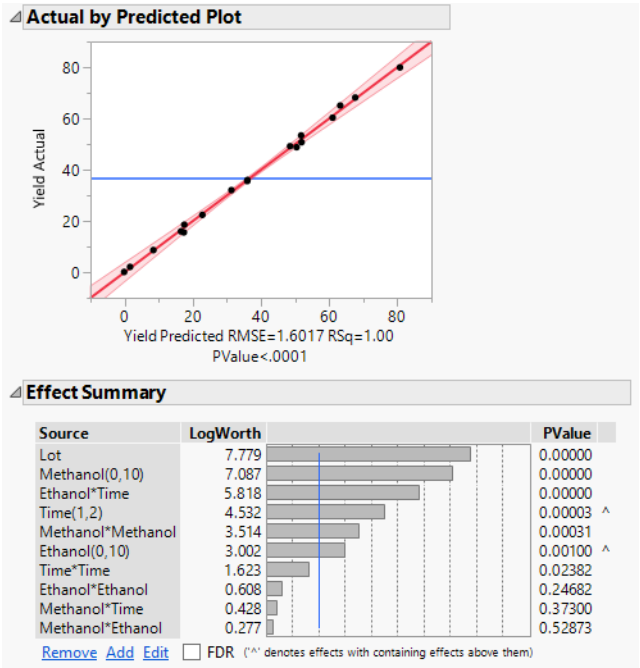
Reduce the Model

The Make Model button enters the model for the listed terms in a Fit Model specification window. To run the model directly using standard least squares, click the Run Model button.

1. Click **Run Model**.

The Actual by Predicted Plot shows no lack of fit. The Effect Summary report suggests that you can reduce the model further.

Figure 8.5 Actual by Predicted Plot and Effect Summary Report

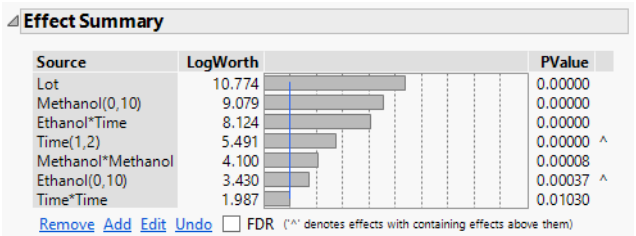


2. Select Methanol*Ethanol in the Effect Summary report and click **Remove**.

   Methanol*Time has *p*-value 0.33750. Remove it next.
3. Select Methanol*Time in the Effect Summary report and click **Remove**.

   Ethanol*Ethanol has *p*-value 0.15885. Remove it next.
4. Select Ethanol*Ethanol in the Effect Summary report and click **Remove**.

Figure 8.6 Effect Summary Report Showing Effects in Final Model



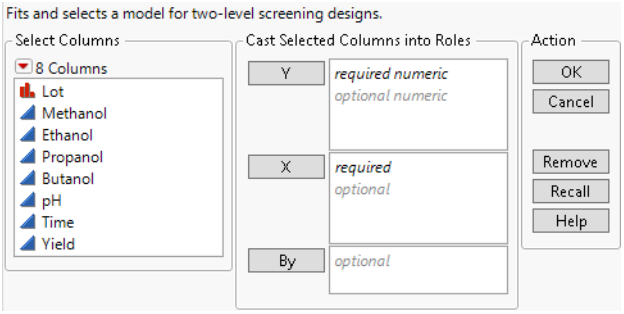
The remaining effects are significant. You conclude that these are the active effects.

Launch the Fit Definitive Screening Platform

To launch the Fit Definitive Screening platform, select **DOE** > **Definitive Screening** > **Fit Definitive Screening**. The launch window in Figure 8.7 uses Extraction3 Data.jmp.

Note: If you created your design in JMP, the design table contains a script called Fit Definitive Screening. Run this script to run the analysis directly.

Figure 8.7 Fit Definitive Screening Launch Window



Y One or more numeric response variables.

X Continuous or two-level categorical factors. Because the platform uses the unique features of a DSD in performing the analysis, these factors must define a DSD or a fold-over design.

By A column whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed. The results appear in separate reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Fit Definitive Screening Report

- "Stage 1 - Main Effect Estimates"
- "Stage 2 - Even Order Effect Estimates"
- "Combined Model Parameter Estimates"
- "Main Effects Residual Plots"
- "Prediction Profiler"

Stage 1 - Main Effect Estimates

The Main Effect Estimates report lists main effects that are identified as active. Main effects with *p*-values less than the threshold *p*-value are considered active.

- If fake factors or center point replicates are available, an estimator of error variance that is independent of the model is constructed. The main effects are tested against this estimate.
- If no fake factors or center point replicates are available, subsets of main effects are tested sequentially against an estimate of error variance constructed from the inactive main effects. For this procedure to be viable, at least one of the main effects must be inactive.

In either case, variability from the inactive main effects is pooled into the error variance used to test the main effects.

Figure 8.8 Stage 1 Report

Stage 1 - Main Effect Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Methanol	9.7133	0.3674	26.438	<.0001*
Ethanol	2.3166	0.3674	6.3055	0.0015*
Time	4.0798	0.3674	11.104	0.0001*

Statistic	Value
RMSE	1.3747
DF	5

☑ Quadratic Terms Obey Strong Heredity
☑ Interactions Obey Strong Heredity

Term Main effects identified as active. These effects have *p*-values less than the threshold when tested as described in "Stage 1 Methodology".

Estimate Parameter estimate for a regression fit of Y on the main effects.

Std Error The standard error of the estimate, computed using the Stage 1 RMSE.

t Ratio The Estimate divided by its Std Error.

Prob>|t| The *p*-value computed using the t Ratio and the degrees of freedom for error (DF).

RMSE The square root of the mean square error that results from the Stage 1 analysis.

- If fake factors or centerpoint replicates are available, the mean square error is the estimate of variance from fake factors and centerpoints pooled with the variance estimate constructed from the main effects that are not identified as active.
- If no fake factors or centerpoint replicates are available, the mean square error is the estimate of variance constructed from the main effects that are not identified as active.

DF The degrees of freedom associated with the error estimate used to construct RMSE.

- If fake factors or centerpoint replicates are available, DF is the sum of the number of fake factors, centerpoint replicates, and main effects not identified as active.

– If no fake factors or centerpoint replicates are available, DF is the number of main effects that are not identified as active.

Quadratic Terms Obey Strong Heredity Check or uncheck to control heredity rules applied to quadratic terms in Stage 2.

Interactions Obey Strong Heredity Check or uncheck to control heredity rules applied to interactions in Stage 2.

Note: Strong effect heredity means that the A*B interaction can only be considered for inclusion in the model if both A and B have been included. Strong effect heredity requires that all lower-order components of a model effect be included in the model.

Stage 2 - Even Order Effect Estimates

The Even Order Effect Estimates report lists second-order effects that are identified as active. Active second-order effects are identified using the guided variable selection procedure or forward selection as described in "Stage 2 Methodology". The block effect (if one is included) is also listed, whether it is significant or not.

Figure 8.9 Stage 2 Report

Stage 2 - Even Order Effect Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	34.568	1.3459	25.683	0.0015*
Lot[1]	17.197	0.7757	22.171	0.0020*
Methanol*Ethanol	-0.367	0.7127	-0.515	0.6581
Methanol*Time	0.5266	0.7127	0.7389	0.5369
Ethanol*Time	9.8258	0.8534	11.514	0.0075*
Methanol*Methanol	7.637	1.4914	5.1208	0.0361*
Ethanol*Ethanol	-1.449	1.477	-0.981	0.4299
Time*Time	-3.297	1.477	-2.232	0.1552

Statistic	Value
RMSE	2.0626
DF	2

Term The block factor and second-order effects identified as active.

Estimate Parameter estimates for a regression fit of Y on the Stage 2 second-order effects defined by Y_{2nd}. See "Decomposition of Response".

Std Error The standard error of the estimate, computed using the Stage 2 RMSE.

t Ratio The Estimate divided by its Std Error.

Prob>|t| The *p*-value computed using the t Ratio and the degrees of freedom for error (DF).

RMSE The square root of the mean square error that results from the Stage 2 analysis. RMSE is estimated as the residual variance from Y_{2nd} after fitting the second order effects identified as active. See "Decomposition of Response".

DF The degrees of freedom associated with the error estimate used to construct RMSE.

Combined Model Parameter Estimates

The Combined Model Parameter Estimates report lists the terms in the final model and their usual standard least squares estimates, standard errors, t ratios, *p*-values, RMSE, and model degrees of freedom.

Below the report are buttons that construct or run the combined model.

Figure 8.10 Combined Model Parameter Estimates Report

Combined Model Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	34.568	1.0452	33.074	<.0001*
Lot[1]	17.197	0.6023	28.552	<.0001*
Methanol	9.7133	0.4281	22.691	<.0001*
Ethanol	2.3166	0.4281	5.4118	0.0010*
Time	4.0798	0.4281	9.5307	<.0001*
Methanol*Ethanol	-0.367	0.5534	-0.663	0.5287
Methanol*Time	0.5266	0.5534	0.9516	0.3730
Ethanol*Time	9.8258	0.6627	14.828	<.0001*
Methanol*Methanol	7.637	1.1581	6.5945	0.0003*
Ethanol*Ethanol	-1.449	1.1469	-1.264	0.2468
Time*Time	-3.297	1.1469	-2.875	0.0238*

Statistic	Value
RMSE	1.6017
DF	7

Make Model Run Model

Make Model Creates a model for the Fit Model window containing the model terms in the Combined Model Parameter Estimates report and the response specified for the Fit Definitive Screening analysis. The Standard Least Squares personality is specified.

Run Model Runs a standard least squares fit for the model terms in the Combined Model Parameter Estimates report and the response specified for the Fit Definitive Screening analysis.

Main Effects Residual Plots

Shows a plot of the residuals from a main effects model fit to all other factors versus the levels of the plotted factor. A factor with residuals that differ across levels indicates an important main effect.

Note: There is not a plot for a block factor.

Prediction Profiler

Shows a prediction profiler for the combined model. For more information about the prediction profiler, see *Profilers*.

Fit Definitive Screening Platform Options

Set Stage 1 p value Set the *p*-value threshold used for Stage 1. See "Stage 1 Methodology".

Set Stage 2 ratio Specify the ratio of the Stage 2 MSE to the Stage 1 MSE. See "Stage 2 Methodology".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Technical Details for the Fit Definitive Screening Platform

The Effective Model Selection for DSDs Algorithm

This section provides a summary of the algorithm used in the Fit Definitive Screening platform. See Jones and Nachtsheim (2016).

Decomposition of Response

The Effective Model Selection algorithm expresses the response, Y, in terms of two responses Y_{ME} and Y_{2nd}, so that $Y = Y_{ME} + Y_{2nd}$.

– Y_{ME} is the predicted value obtained from a regression of Y on the main effects and fake factors.

There is no need to include the block factor in Y_{ME} because of the fold-over structure of the design. The block factor is included in Y_{2nd}.

– Y_{2nd} is given by $Y_{2nd} = Y - Y_{ME}$.

Note: In a DSD, the columns Y_{ME} and Y_{2nd} are orthogonal.

The analysis proceeds in two stages:

- Stage 1: The response Y_{ME} is used to identify main effects. Stage 1 identifies the main effects that are considered active.
- Stage 2: The response Y_{2nd} is used to identify second-order effects. Stage 2 considers all second-order terms in the active main effects from Stage 1 and determines a subset of these containing effects considered to be active.

Note: If there is a blocking factor, it is included in the Stage 2 list of effects even if it is not significant.

Stage 1 Methodology

The Stage 1 methodology depends on whether the design contains fake factors or centerpoint replicates.

Case 1: Fake Factors or Centerpoint Replicates Available

1. Using the fake factors or center point replicates, an estimator of error variance that is independent of the model is constructed. Assuming that there are no active third or higher odd order effects, this estimate is unbiased.
2. Using Y_{ME}, main effects are tested against this estimate. Main effects with *p*-values less than a threshold *p*-value are considered active. The threshold values are the following:
   – For one error degree of freedom, the threshold value is 0.20.
   – For two error degrees of freedom, the threshold value is 0.10.
   – For more than two error degrees of freedom, the threshold value is 0.05.
   – User specified *p*-value is the threshold.

   Note: To specify a different *p*-value threshold select **Set Stage 1 p value** from the Fit Definitive Screening red triangle menu.
3. If no main effect has a *p*-value less than the threshold value, conclude that there are no active main effects and no active two-factor effects. The procedure terminates.
4. If active main effects are found, then variability from the inactive main effects is pooled into the error variance constructed in (1).

Note: If Categorical factors are in the design, the estimated coefficients are recalculated each time a main effect is chosen as active.

Case 2: No Fake Factors or Centerpoint Replicates Available

In this case, there is no model-independent estimator of error variance available. Subsets of main effects are tested sequentially against an estimate of error variance constructed from the inactive main effects. Suppose that there are m main effects.

1. The absolute values of the estimated effects, using Y_{ME} as the response, are ordered from largest to smallest.
2. For each $1 \leq i < m$, the effect with the i^{th} largest absolute value is tested against the adjusted residual sum of squares for the model containing that effect and all effects with larger absolute values.
3. The effects in the model with the smallest p-value are considered to be the active effects.
4. If active main effects are found, then variability from the inactive main effects is used to construct an estimate of error variance, using Y_{ME} as the response.

Note: For the Fit Definitive Screening procedure to work properly in Case 2, at least one of the main effects must be active and at least one must be inactive. If no main effects are active, or if all main effects are active, the procedure will identify a set of main effects, but the procedure for arriving at that subset is compromised.

Stage 2 Methodology

In Stage 2, the factors considered depend on the Strong Heredity options. When strong heredity is selected, only second-order effects involving the factors whose main effects are identified as active in Stage 1 are considered. The Stage 2 methodology depends on the number of active main effects identified in Stage 1.

Case 1: Seven or Fewer Active Main Effects

Stage 2 uses a guided subset selection procedure. The goal is to continue to add second-order effects to the model as long as the ratio of the RMSE from Stage 2 to the RMSE from Stage 1 is greater than the specified threshold. When the ratio is less than or equal to the threshold, this indicates that there are no additional second-order effects to add to the model. The default threshold is 1. Smaller thresholds increases the number of terms likely to identified as active as compared to larger thresholds.

Note: To specify a RMSE ratio threshold other than one, select **Set Stage 2 ratio** from the Fit Definitive Screening red triangle menu.

For Stage 2:

- For one error degree of freedom, the threshold value is 0.20.
- For two error degrees of freedom, the threshold value is 0.10.
- For more than two error degrees of freedom, the threshold value is 0.05.
- User specified *p*-value is the threshold.

1. The variability for Y_{2nd} is tested against the error estimate from Stage 1 to determine if there is additional variability due to second-order effects.
   - If the *p*-value for this test exceeds the threshold value the procedure terminates and no active second-order effects are identified.
2. If the *p*-value for this test is less than or equal to the threshold value, then subsets of size *k*, *k = 1,2,3,...* are successively tested, starting with *k = 1*.
3. For each *k*, the residual sum of squares for each subset of that size is tested against the error estimate from Stage 1. The subset with the smallest RMSE is identified.
4. The procedure continues until a *k* is found with a ratio of RMSE to the Stage 1 RMSE smaller than the Stage 2 ratio.
5. The effects in the subset preceding the one that corresponds to the terminal value of *k* are considered to be the active two-factor effects.

Case 2: Eight or More Active Main Effects

Stage 2 uses forward selection for second order terms when eight or more active main effects are identified in Stage 1.

Chapter 9

Screening Designs

Screening designs are among the most popular designs for industrial experimentation. Typically used in the initial stages of experimentation, they examine many factors in order to identify those factors that have the greatest effect on the response. The factors that are identified are then studied using more sensitive designs. Because screening designs generally require fewer experimental runs than other designs, they are a relatively inexpensive and efficient way to begin improving a process.

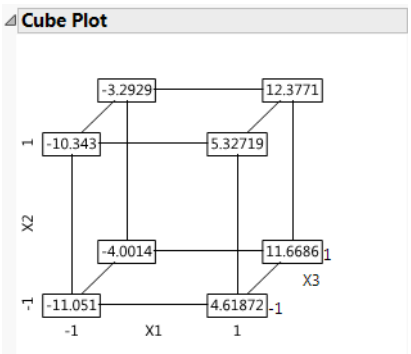
If a standard screening design exists for your experimental situation, you can choose from several standard screening designs. The list includes blocked designs when applicable. Your factors can be two-level continuous factors, three-level categorical factors, or continuous factors that can assume only discrete values (*discrete numeric* factors).

However, there are situations where standard screening designs are not available. In these cases, the Screening Design platform constructs a *main effects screening design*. A main effects screening design is either orthogonal or near orthogonal. It focuses on estimating main effects in the presence of negligible interactions.

Note that JMP also provides two compelling alternatives to screening designs:

- Definitive screening designs are particularly useful if you suspect active two-factor interactions or if you suspect that a plot of a continuous factor's effect on the response might exhibit strong curvature. See "Definitive Screening Designs".
- Custom designs are highly flexible and often more cost-effective than a design obtained using alternative methods. See "Custom Designs".

Figure 9.1 Results from a Fractional Factorial Design



Contents

Overview of Screening Designs 293
Underlying Principles 293
Analysis of Screening Design Results 294
Examples of Screening Designs 294
Compare a Fractional Factorial Design and a Main Effects Screening Design 294
Main Effects Screening Design where No Standard Design Exists 302
Screening Design Window 305
Responses 306
Factors 308
Choose Screening Type 310
Choose from a List of Fractional Factorial Designs 311
Design Type 311
Resolution as a Measure of Confounding 314
Display and Modify Design 315
Main Effects Screening Designs 318
Design Generation 319
Design 320
Design Evaluation 320
Output Options 321
Make Table 322
Screening Design Options 323
Additional Examples of Screening Designs 325
Modify Generating Rules in a Fractional Factorial Design 326
Plackett-Burman Design 331

Overview of Screening Designs

Screening experiments tend to be small and are aimed at identifying the factors that affect a response. Because identification is the goal (rather than sophisticated modeling), continuous factors in a screening design are typically set at only two levels. However, a screening situation might also involve discrete numeric or categorical factors, in which case classical screening designs might not fit your situation. The Screening Design platform can handle all three types of factors: two-level continuous factors, categorical factors, and discrete numeric factors.

There are two types of designs:

- Classical designs: For situations where standard screening designs exist, you can choose from a list that includes fractional factorial designs, Plackett-Burman, Cotter, and mixed-level designs.
- Main effects screening designs: Whether a standard design is available, you can ask JMP to construct a main effects screening design. These designs are orthogonal or near orthogonal and focus on estimating main effects in the presence of negligible interactions. See "Main Effects Screening Designs".

Underlying Principles

The emphasis on studying main effects early on in the experimentation process is supported by the empirical principle of effect hierarchy. This principle maintains that lower order effects are more likely to be important than higher order effects. For this reason, screening designs focus on identifying active main effects. In cases where higher order interactions *are* of interest, screening designs assume that two-factor interactions are more important than three-factor interactions, and so on. See "Effect Hierarchy" and Wu and Hamada (2009).

The efficiency of screening designs also depends on the principle of effect sparsity. Effect sparsity asserts that most of the variation in the response is explained by a relatively small number of effects. See "Effect Sparsity".

To appreciate the importance of effect sparsity, consider an example where you have seven two-level factors. Contrast a full factorial design to a screening design:

- A full factorial design consists of all combinations of the levels of the factors. The number of runs is the product of the numbers of levels for each factor. In this example, a full factorial design has $2^7 = 128$ runs.
- In contrast, a screening design requires only a fraction of the runs in the full factorial design. The main effects of the seven factors can be studied in an eight-run screening design.

Analysis of Screening Design Results

Screening designs are often used to test a large number of factors or interactions. When there are degrees of freedom for error, allowing construction of an error estimate, the experimental results can be analyzed using the usual regression techniques (Analyze > Fit Model).

However, sometimes there are no degrees of freedom for error. In this case, assuming effect sparsity, the Screening platform (DOE > Classical > Two Level Screening > Fit Two Level Screening) provides a way to analyze the results of a two-level design. The Screening platform accepts multiple responses and multiple factors. It automatically shows significant effects with plots and statistics. See "Screening Designs". For an examples in the current chapter, see "Modify Generating Rules in a Fractional Factorial Design" and "Plackett-Burman Design".

Examples of Screening Designs

- "Compare a Fractional Factorial Design and a Main Effects Screening Design"
- "Main Effects Screening Design where No Standard Design Exists"

Compare a Fractional Factorial Design and a Main Effects Screening Design

In this example, suppose an engineer wants to investigate a process that uses an electron beam welding machine to join two parts. The engineer fits the two parts into a welding fixture that holds them snugly together. A voltage applied to a beam generator creates a stream of electrons that heats the two parts, causing them to fuse. The ideal depth of the fused region is 0.17 inches. The engineer wants to study the welding process to determine the best settings for the beam generator to produce the desired depth in the fused region.

For this study, the engineer wants to explore the following seven factors:

- Operator is the technician operating the welding machine. Two technicians typically operate the machine.
- Speed (in rpm) is the speed at which the part rotates under the beam.
- Current (in amps) is a current that affects the intensity of the beam.
- Mode is the welding method used.
- Wall Size (in mm) is the thickness of the part wall.
- Geometry indicates whether the joint is a single-bevel joint or a double-bevel joint.
- Material is the type of material being welded.

Notice that three of these factors are continuous: Speed, Current, and Wall Size. Four are categorical: Operator, Mode, Geometry, and Material. Each of these categorical factors has two levels.

After each processing run, the engineer cuts the part in half. This reveals an area where the two parts have fused. The length of this fused area, measured in inches, is the depth of penetration of the weld. The depth of penetration is the response for the study.

The goals of the study are the following:

- Find which factors affect the depth of the weld.
- Quantify those effects.
- Find specific factor settings that predict a weld depth of 0.17 inches with a tolerance of ±0.05 inches.

Your experimental budget allows you at most 12 runs. Construct and compare two designs for your experimental situation. The first is a classical fractional factorial design using eight runs. The second is a main effects screening design using 12 runs.

Constructing a Standard Screening Design

In this section, construct a standard screening design for this experimental situation.

Specify the Response

1. Select **DOE > Classical > Two Level Screening > Screening Design**.
2. In the Responses panel, double-click Y under Response Name and type Depth.

   Note that the default Goal is Maximize. Your goal is to find factor settings that enable you to obtain a target depth of 0.17 inches with limits of 0.12 and 0.22.
3. Click the default Goal of Maximize and change it to **Match Target**.
4. Click under **Lower Limit** and type 0.12.
5. Click under **Upper Limit** and type 0.22.
6. Leave the area under **Importance** blank.

   Because there is only one response, that response is given Importance 1 by default.

The completed Responses outline appears in Figure 9.2. Now, specify the factors.

Specify Factors

You can enter the factors manually or automatically:

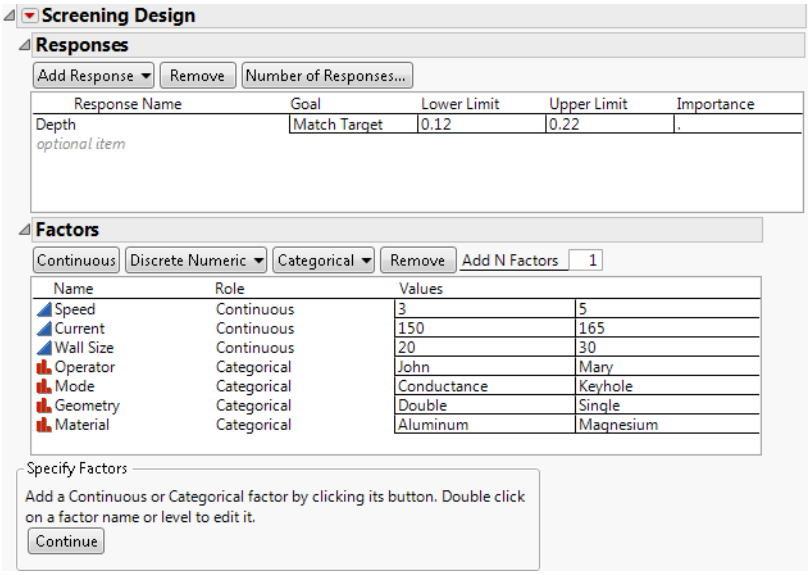
- To enter the factors manually, see "Specify Factors Manually".
- To enter the factors automatically, use the Weld Factors.jmp data table:

1. Select **Help > Sample Data Library** and open Design Experiment/Weld Factors.jmp.
2. Click the Screening Design red triangle and select **Load Factors**. Proceed to "Choose a Design".

Specify Factors Manually

1. Type **3** in the **Add N Factors** box and click **Continuous**.
2. Double-click X1 and type Speed.
3. Use the Tab key to move through the rest of the values and factors. Make the following changes:
   a. Change the Speed values to 3 and 5.
   b. Change X2 to Current, with values of 150 and 165.
   c. Change X3 to Wall Size, with values of 20 and 30.
4. Type **4** in the **Add N Factors** box and select **Categorical > 2 Level**.
5. Double-click X4 and type Operator.
6. Use the Tab key to move through the rest of the values and factors. Make the following changes:
   a. Change the Operator values to John and Mary.
   b. Change X5 to Mode, with values of Conductance and Keyhole.
   c. Change X6 to Geometry, with values of Double and Single.
   d. Change X7 to Material, with values of Aluminum and Magnesium.

Figure 9.2 Responses and Factors Outlines for Weld Experiment



Choose a Design

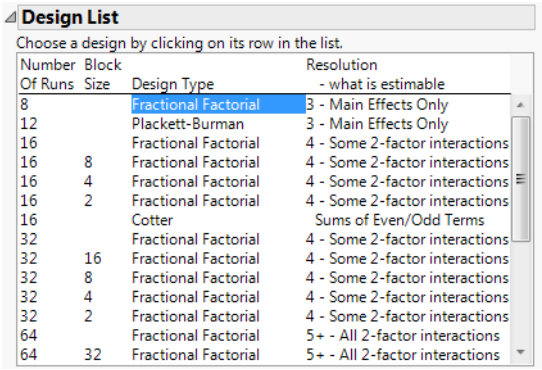
1. Click **Continue**.

   Because the combination of factors and levels that you have specified can be accommodated by a standard fractional factorial design, the Choose Screening Type panel appears. You can either select a standard design from a list or construct a main effects design.

   Note: Setting the Random Seed in the next step reproduces the results shown in this example. When you are constructing a design on your own, this step is not necessary.

2. (Optional) Click the Screening Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
3. Accept the default selection to **Choose from a list of fractional factorial designs** and click **Continue**.
4. Select the first Fractional Factorial design.

Figure 9.3 Design List for Three Continuous Factors and Four Categorical Factors



Design List

Choose a design by clicking on its row in the list.

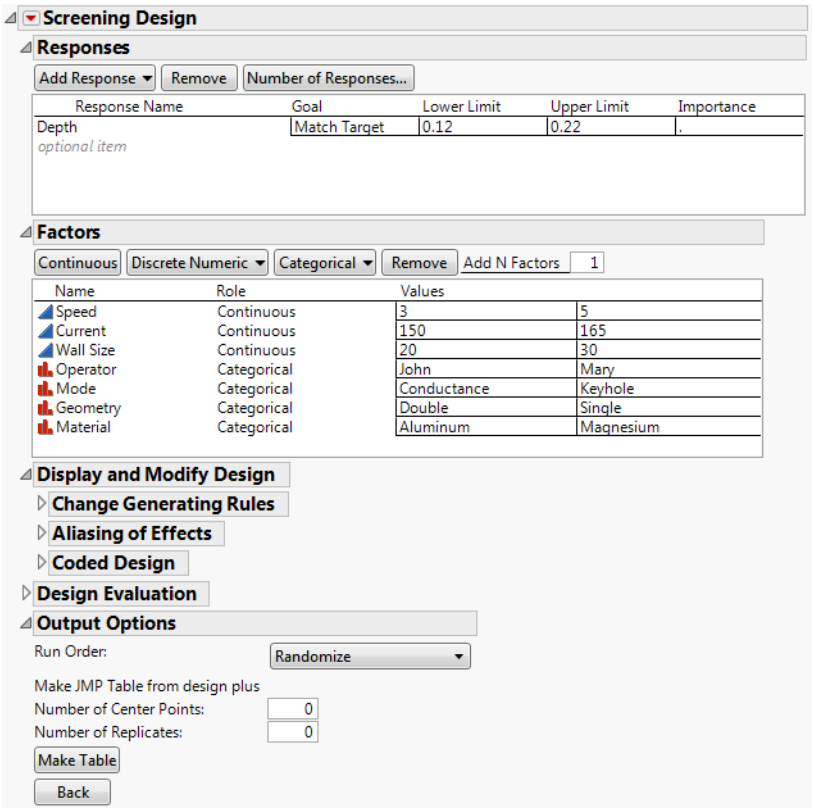
Number Of Runs	Block Size	Design Type	Resolution - what is estimable
8		Fractional Factorial	3 - Main Effects Only
12		Plackett-Burman	3 - Main Effects Only
16		Fractional Factorial	4 - Some 2-factor interactions
16	8	Fractional Factorial	4 - Some 2-factor interactions
16	4	Fractional Factorial	4 - Some 2-factor interactions
16	2	Fractional Factorial	4 - Some 2-factor interactions
16		Cotter	Sums of Even/Odd Terms
32		Fractional Factorial	4 - Some 2-factor interactions
32	16	Fractional Factorial	4 - Some 2-factor interactions
32	8	Fractional Factorial	4 - Some 2-factor interactions
32	4	Fractional Factorial	4 - Some 2-factor interactions
32	2	Fractional Factorial	4 - Some 2-factor interactions
64		Fractional Factorial	5+ - All 2-factor interactions
64	32	Fractional Factorial	5+ - All 2-factor interactions

This specifies an eight-run Resolution 3 fractional factorial design. For information about resolution, see "Resolution as a Measure of Confounding".

5. Click **Continue**.

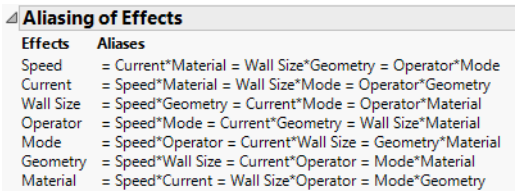
   In the Output Options outline, note that **Run Order** is set to **Randomize**. This means that the design runs will appear in random order. This is the order you should use to conduct your experimental runs.

Figure 9.4 Completed Screening Design Window



6. Open the Aliasing of Effects outline under Display and Modify Design.

Figure 9.5 Aliasing for an Eight-Run Fractional Factorial Design



Recall that you selected a Resolution 3 design (Figure 9.3). In a Resolution 3 design, some main effects are confounded with two-way interactions. The Aliasing of Effects outline indicates that, for this Resolution 3 design, every main effect is completely confounded with three two-way interactions. If you suspect that two-way interactions are active, this is a poor design. For a description of confounding, see "Two-Level Regular Fractional Factorial".

7. Click **Make Table**.

Figure 9.6 The Design Data Table

	Pattern	Speed	Current	Wall Size	Operator	Mode	Geometry	Material	Depth
1	+++++++	5	165	30	Mary	Keyhole	Single	Magnesium	•
2	-+-+-+-	3	165	20	Mary	Conductance	Single	Aluminum	•
3	+--+--+	5	150	30	John	Conductance	Single	Aluminum	•
4	---+++	3	150	20	John	Keyhole	Single	Magnesium	•
5	-++-+--	3	165	30	John	Keyhole	Double	Aluminum	•
6	+-----+	5	165	20	John	Conductance	Double	Magnesium	•
7	+--+---	5	150	20	Mary	Keyhole	Double	Aluminum	•
8	--+--+	3	150	30	Mary	Conductance	Double	Magnesium	•

Notice the following:

- The table uses the names for the responses, factors, and levels that you specified.
- The Pattern column shows the assignment of high and low settings for the design runs.
- This fractional factorial design is a Resolution 3 design. It enables you to study the main effects of seven factors in eight runs.

Constructing a Main Effects Screening Design

Main effects screening designs are orthogonal or near orthogonal designs. In this section, construct a main effects screening design for your seven factors.

1. Open your Screening Design window. If you have closed it, then run the **DOE Dialog** script in the Design Data table.
2. Click **Back**.
3. Click **Continue**.

Note: Setting the Random Seed and the Number of Starts in the next two steps reproduces the exact results shown in this example. When constructing a design on your own, these steps are not necessary.

4. (Optional) Click the Screening Design red triangle, select **Set Random Seed**, type 12345, and click **OK**.
5. (Optional) Click the Screening Design red triangle, select **Number of Starts**, type 50, and click **OK**.
6. In the Choose Screening Type panel, select the **Construct a main effects screening design** option.
7. Click **Continue**.

   Under Number of Runs, the selected option is **Default** with the number of runs set to 12. Keep this setting.
8. Click **Make Design**.

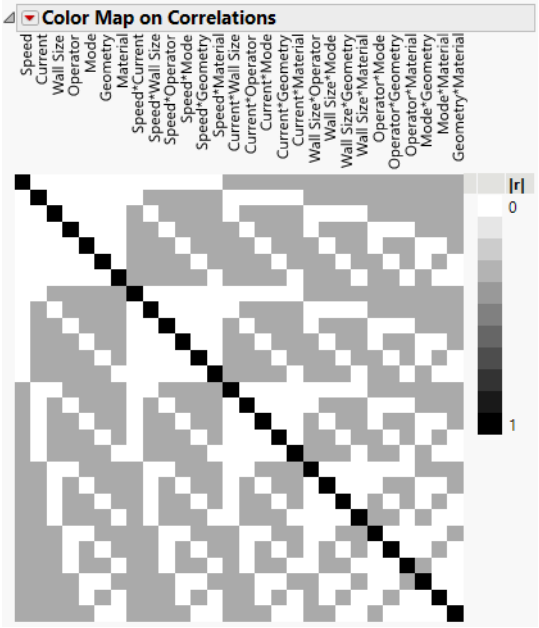
Figure 9.7 Main Effects Screening Design

Design

Run	Speed	Current	Wall Size	Operator	Mode	Geometry	Material
1	3	165	20	John	Conductance	Single	Aluminum
2	5	150	30	Mary	Conductance	Double	Aluminum
3	3	150	30	John	Keyhole	Double	Aluminum
4	5	165	20	Mary	Keyhole	Double	Magnesium
5	3	165	20	Mary	Conductance	Double	Aluminum
6	5	150	20	John	Conductance	Double	Magnesium
7	3	150	30	Mary	Conductance	Single	Magnesium
8	5	150	20	John	Keyhole	Single	Aluminum
9	3	165	30	John	Keyhole	Double	Magnesium
10	5	165	30	John	Conductance	Single	Magnesium
11	3	150	20	Mary	Keyhole	Single	Magnesium
12	5	165	30	Mary	Keyhole	Single	Aluminum

9. Open the Design Evaluation outline and then open the Color Map on Correlations outline.

Figure 9.8 Color Map on Correlations for 12-Run Main Effects Screening Design



The Color Map on Correlations shows that the main effects are uncorrelated with each other. This is indicated by the white off-diagonal cells in the upper left corner of the color map. Each main effect is partially aliased with some two-way interactions, indicated by the gray cells. Hover over one of the gray cells to see that the absolute correlations are 0.333.

In this case, the 12-run main effects screening design is a Plackett-Burman design, which you could have obtained in the Design List. However, in many design situations, the partial aliasing that occurs in a main effects design is preferable to the complete

confounding that occurs in a fractional factorial design that you adapt to your experimental situation.

The next section shows an example of a situation where no standard design exists. In this case, JMP constructs a main effects screening design.

Main Effects Screening Design where No Standard Design Exists

Main effects screening designs are orthogonal or near orthogonal designs for the main effects. You can use them in place of standard designs and in situations where standard designs do not exist. Main effects screening designs are excellent for estimating main effects when interactions are negligible.

In this experimental situation, no standard design exists. You need a design to study 13 factors: 2 are categorical, one with 4 levels and one with 6 levels, and 11 are continuous.

1. Select **DOE > Classical > Two Level Screening > Screening Design.**

   In the Responses panel, there is a single default response called Y. Keep this as the default response.

2. In the Factors panel, click **Categorical** and select **4 Level**.

   This adds the variable X1 with levels L1 through L4.

3. Click **Categorical** and select **6 Level**.

   This adds the variable X2, with levels L1 through L6.

4. Enter 11 next to **Add N Factors**.

5. Click **Continuous**.

   This adds 11 factors, X3 to X13, each at two levels, -1 and 1.

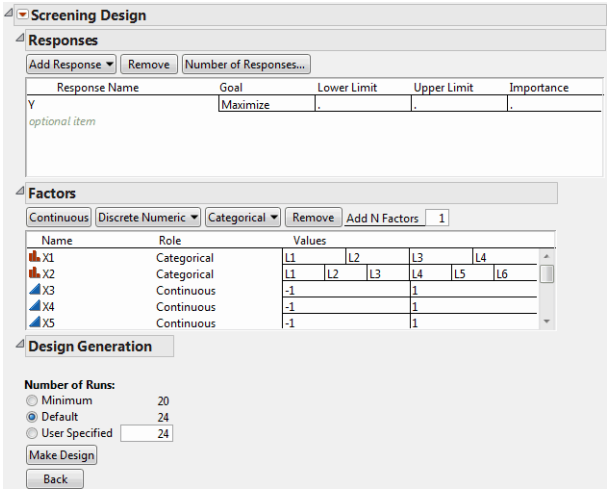
6. Click **Continue**.

   The Design Generation panel appears.

   There is no option to select a design from the Design List since there are no available standard designs in this situation.

   Keep the default number of runs, which is 24.

Figure 9.9 Screening Design Window for 13-Factor Design with Design Generation Panel



7. Click **Make Design**.

   A Design and a Design Evaluation outline appear.

8. Open the Design outline to see the randomized design.

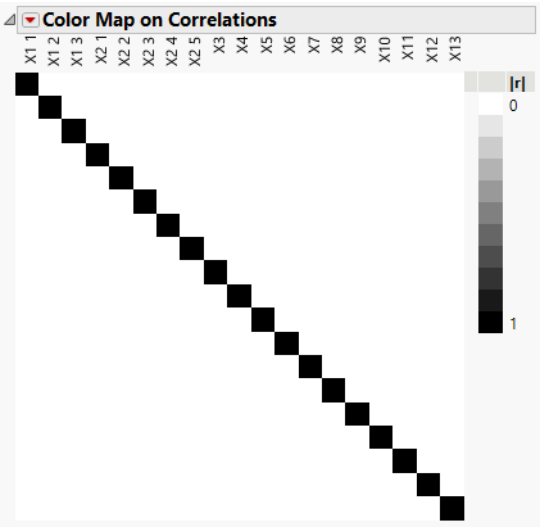
Note: The algorithm that generates the design uses a random starting design. To reproduce this design, save the script with the random seed by selecting **Save Script to Script Window** from the red triangle menu next to the report title.

Next, examine the Color Map on Correlations to see that this specific design is orthogonal.

9. Open the **Design Evaluation > Color Map on Correlations** outline.

   The color map (Figure 9.10) shows black entries (using JMP default colors) on the main diagonal, indicating correlations of one. This is because each diagonal cell corresponds to the correlation of a term with itself, which is one. Off-diagonal correlations are all white, indicating that correlations between distinct terms are zero. Hover over any cell to see the relevant terms and their absolute correlation.

Figure 9.10 Color Map on Correlations



10. Click **Make Table** to construct the design table.

    The table contains the runs for your experiment in random order. Conduct the experiment in this randomized order and insert the results in column Y. Run the **Model** script in the data table to analyze your results.

Screening Design Window

The Screening Design window updates as you work through the design steps. See "The DOE Workflow: Describe, Specify, Design". The outlines that appear are separated by buttons that update the window. These follow the flow in the figures below.

Figure 9.11 Screening Design Flow when a Standard Design Exists

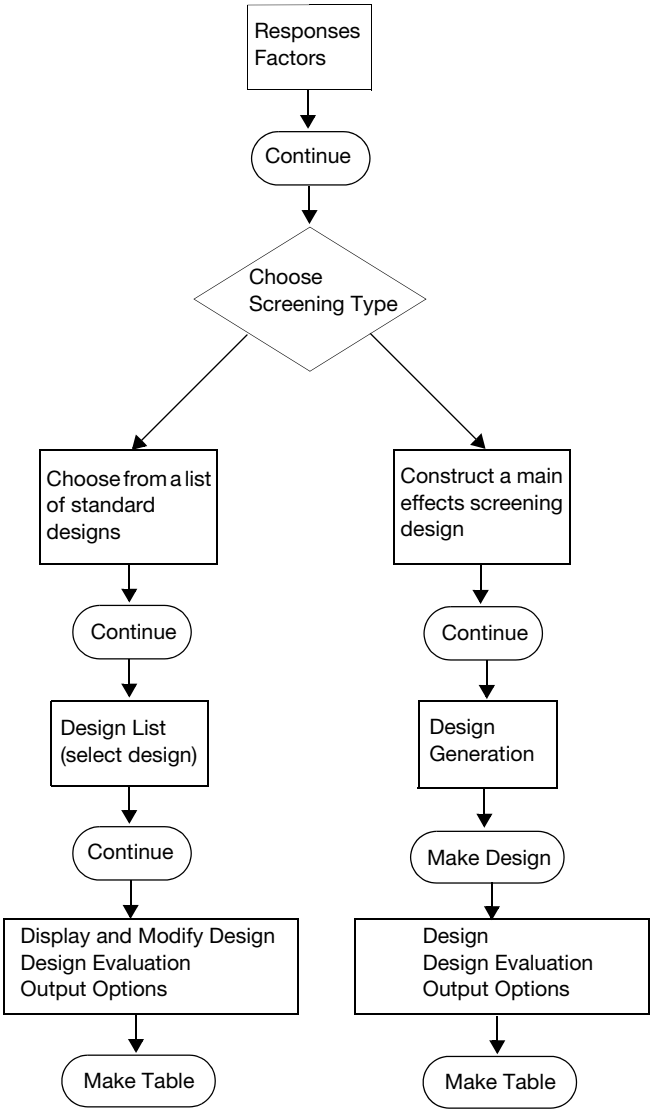
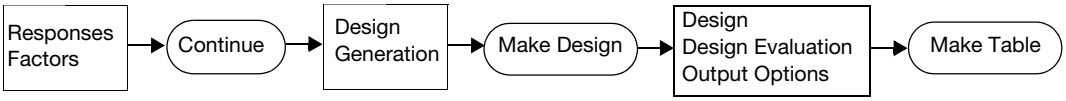


Figure 9.12 Screening Design Flow when No Standard Design Exists

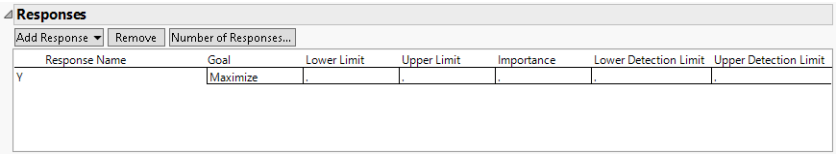


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 9.13 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column

in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.
- A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

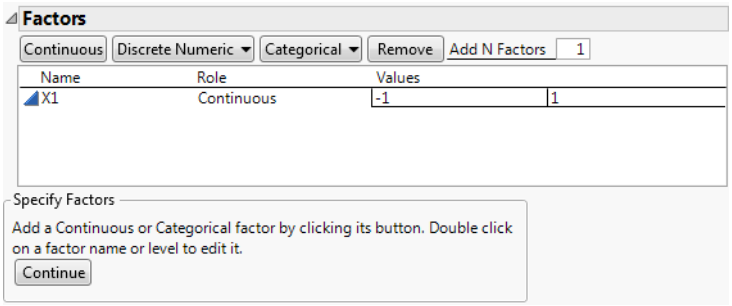
If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

Factors

Add factors in the Factors outline.

Tip: When you have completed the Factors outline, consider selecting **Save Factors** from the red triangle menu. This option saves the factor names, roles, changes, and values in a data table that you can later reload in DOE platforms.

Figure 9.14 Factors Outline



Continuous Adds a Continuous factor. The data type in the resulting data table is Numeric. A continuous factor is a factor that you can conceptually set to any value between the lower and upper limits you supply, given the limitations of your process and measurement system.

Discrete Numeric Adds a Discrete Numeric factor. A discrete numeric factor can assume only a discrete number of numeric values. These values have an implied order. The data type in the resulting data table is Numeric.

A screening design includes all levels of a discrete numeric factor and attempts to balance the levels. Fit Model treats a discrete numeric factor as a continuous predictor.

The default values for a discrete numeric factor with k levels, where $k > 2$, are the integers 1, 2, ..., k. The default values for a discrete numeric factor with $k = 2$ levels are -1 and 1. Replace the default values with the settings that you plan to use in your experiment.

Categorical Adds a Categorical factor. Click to select or specify the number of levels. The data type in the resulting data table is Character. The value ordering of the levels is the order of the values, as entered from left to right. This ordering is saved in the Value Order column property after the design data table is created.

The default values for a categorical factor are L1, L2,..., Lk, where k is the number of levels that you specify. Replace the default values with level names that are relevant for your experiment.

Remove Removes the selected factors.

Add N Factors Adds multiple factors. Enter the number of factors to add, click **Add Factor,** and then select the factor type. Repeat **Add N Factors** to add multiple factors of different types.

Factors Outline

The Factors outline contains the following columns:

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role The Design Role of the factor. The Design Role column property for the factor is saved to the data table. This property ensures that the factor type is modeled appropriately. The Role of the factor determines other factor properties that are saved to the data table. See "Factor Column Properties".

Values The experimental settings for the factors.

Editing the Factors Outline

In the Factors outline, note the following:

- To edit a factor name, double-click the factor name.
- To edit a value, click the value in the Values column.

Factor Column Properties

For each factor, various column properties are saved to the design table after you create the design by selecting Make Table in the Screening Design window. These properties are also saved automatically to the data table that is created when you select the Save Factors option. You can find more information about these column properties and related examples in "Column Properties".

Coding If the Role is Continuous or Discrete Numeric, the Coding column property for the factor is saved. This property transforms the factor values so that the low and high values correspond to –1 and +1, respectively. See "Coding".

Value Order If the Role is Categorical or if a Block variable is constructed, the Value Order column property for the factor is saved. This property determines the order in which levels of the factor appear. See "Value Order".

Design Role Each factor is given the Design Role column property. The Role that you specify in defining the factor determines the value of its Design Role column property. When you select a design with a block, that Block factor is assigned the Blocking value. The Design Role property reflects how the factor is intended to be used in modeling the experimental data. Design Role values are used in the Augment Design platform. See "Design Role".

Factor Changes Each factor is assigned the Factor Changes column property with the value of Easy. The Factor Changes property reflects how the factor is used in modeling the experimental data. Factor Changes values are used in the Augment Design and Evaluate Design platforms. See "Factor Changes".

RunsPerBlock For a blocking factor, indicates the maximum allowable number of runs in each block. When a Blocking factor is specified in the Factors outline, the RunsPerBlock column property is saved for that factor. See "RunsPerBlock".

Choose Screening Type

After you enter your responses and factors and click **Continue**, one of the following results occurs:

- If a standard design can accommodate your factors and levels, two options appear in the Choose Screening Type panel. See "Choose Screening Type Options".
- If no listed standard design exists for your factors and levels, then the Choose Screening Type panel does not appear. The Design Generation outline for constructing a main effects screening design opens. See "Design Generation".

Choose Screening Type Options

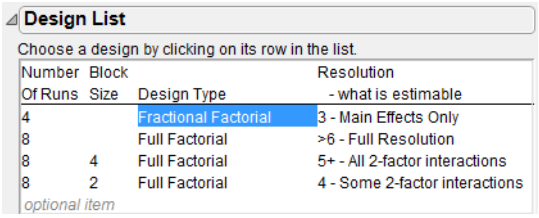
Choose from a list of fractional factorial designs Enables you to select from a list of designs. This option is the default. See "Choose from a List of Fractional Factorial Designs".

Construct a main effects screening design Opens the Design Generation outline where you can specify the number of runs in the main effects screening design. For more information about main effects screening designs, see "Main Effects Screening Designs".

Choose from a List of Fractional Factorial Designs

The list of screening designs that you can choose from includes designs that group the experimental runs into blocks of equal sizes where the size is a power of two. Select the type of screening design that you want to use and click **Continue**.

Figure 9.15 Choosing a Type of Fractional Factorial Design



The Design List contains the following columns:

Number of Runs Total number of runs in the design.

Block Size Number of runs in a block. The number of blocks is the Number of Runs divided by Block Size.

Design Type Description of the type of design. See "Design Type".

Resolution Gives the resolution of the design and a brief description of the type of aliasing. See "Resolution as a Measure of Confounding".

Design Type

The Design List provides the following types of designs:

- "Two-Level Full Factorial"
- "Two-Level Regular Fractional Factorial"
- "Plackett-Burman Designs"

- "Mixed-Level Designs"
- "Cotter Designs"

Two-Level Full Factorial

A full factorial design has runs for all combinations of the levels of the factors. The sample size is the product of the levels of the factors. For two-level designs, this is 2^k where k is the number of factors.

Full factorial designs are orthogonal for all effects. It follows that estimates of the effects are uncorrelated. Also, if you remove an effect from the analysis, the values of the other estimates do not change. Their p-values change slightly, because the estimate of the error variance and the degrees of freedom are different.

Full factorial designs allow the estimation of interactions of all orders up to the number of factors. However, most empirical modeling involves only first- or second-order approximations to the true functional relationship between the factors and the responses. From this perspective, full factorial designs are an inefficient use of experimental runs.

Two-Level Regular Fractional Factorial

A regular fractional factorial design also has a sample size that is a power of two. For two-level designs, if k is the number of factors, the number of runs in a regular fractional factorial design is 2^{k-p} where $p < k$. A 2^{k-p} fractional factorial design is a 2^{-p} fraction of the k-factor full factorial design. Like full factorial designs, regular fractional factorial designs are orthogonal.

A full factorial design for k factors provides estimates of all interaction effects up to degree k. But because experimental runs are typically expensive, smaller designs are preferred. In a smaller design, some of the higher-order effects are *confounded* with other effects, meaning that the effects cannot be distinguished from each other. Although a linear combination of the confounded effects is estimable, it is not possible to attribute the variation to a specific effect or effects.

In fact, fractional factorials are designed by deciding in advance which interaction effects are confounded with other interaction effects. Experimenters are usually not concerned with interactions involving more than two factors. Three-way and higher-order interaction effects are often assumed to be negligible.

Plackett-Burman Designs

Plackett-Burman designs are an alternative to regular fractional factorials for screening. The number of runs in a Plackett-Burman design is a multiple of four rather than a power of two. There are no two-level fractional factorial designs with run sizes between 16 and 32. However, there are 20-run, 24-run, and 28-run Plackett-Burman designs.

In a Plackett-Burman design, main effects are orthogonal and two-factor interactions are only partially confounded with main effects. By contrast, in a regular Resolution 3 fractional factorial design, some two-factor interactions are indistinguishable from main effects. Plackett-Burman designs are useful when you are interested in detecting large main effects among many factors and where interactions are considered negligible.

Mixed-Level Designs

For most designs that involve categorical or discrete numeric factors at three or more levels, standard designs do not exist. In such cases, the screening platform generates *main effects screening designs*. These designs are orthogonal or near orthogonal for main effects.

For cases where standard mixed-level designs exist, the possible designs are given in the Design List. The Design List provides fractional factorial designs for pure three-level factorials with up to 13 factors. For mixed two-level and three-level designs, the Design list includes the complete factorials and the orthogonal-array designs listed in Table 9.1.

If your number of factors does not exceed the number for a design listed in the table, you can adapt that design by using an appropriate subset of its columns.

Table 9.1 Table of Mixed-Level Designs

	Number of Factors	
Design	**Two–Level**	**Three–Level**
L18 John and L18 Taguchi	1	7
L18 Chakravarty	3	6
L18 Hunter	8	4
L36 Taguchi	11	12

Cotter Designs

Note: By default, Cotter designs are not included in the Design List. To include Cotter designs, deselect **Suppress Cotter Designs** in the Screening Design red triangle menu. To always show Cotter designs, select **File > Preferences > Platforms > DOE** and deselect **Suppress Cotter Designs**.

Cotter designs are useful when you must test many factors, some of which might interact, in a very small number of runs. Cotter designs rely on the principle of effect sparsity. They assume that the sum of effects shows an effect if one of the components of the sum has an active effect. The drawback is that several active effects with mixed signs might sum to near zero, thereby failing to signal an effect. Because of this false-negative risk, many statisticians discourage their use.

For k factors, a Cotter design has $2k + 2$ runs. The design structure is similar to the "vary one factor at a time" approach.

The Cotter design is constructed as follows:

- A run is defined with all factors set to their high level.
- For each of the next k runs, one factor in turn is set at its low level and the others high.
- The next run sets all factors at their low level.
- For each of the next k runs, one factor in turn is set at its high level and the others low.
- The runs are randomized.

When you construct a Cotter design, the design data table includes a set of columns to use as regressors. The column names are of the form <factor name> Odd and <factor name> Even. They are constructed by summing the odd-order and even-order interaction terms, respectively, that contain the given factor.

For example, suppose that there are three factors, A, B, and C. Table 9.2 shows how the values in the regressor columns are calculated.

Table 9.2 Cotter Design Table

Effects Summed for Odd and Even Regressor Columns	
AOdd = A + ABC	AEven = AB + AC
BOdd = B + ABC	BEven = AB + BC
COdd = C + ABC	CEven = BC + AC

The Odd and Even columns define an orthogonal transformation. For this reason, tests for the parameters of the odd and even columns are equivalent to testing the combinations on the original effects.

Resolution as a Measure of Confounding

The *resolution* of a design is a measure of the degree of confounding in the design. The trade-off in screening designs is between the number of runs and the resolution of the design.

Experiments are classified by *resolution number* into these groups:

- Resolution 3 means that some main effects are confounded with one or more two-factor interactions. In order for the main effects to be meaningful, these interactions must be assumed to be negligible.
- Resolution 4 means that main effects are not confounded with other main effects or two-factor interactions. However, some two-factor interactions are confounded with other two-factor interactions.
- Resolution 5 means that there is no confounding between main effects, between main effects and two-factor interactions, and between pairs of two-factor interactions. Some two-factor interactions are confounded with three-factor interactions.
- Resolution 5+ means that the design has resolution greater than 5 but is not a full factorial design.
- Resolution 6 means that there is no confounding between effects of any order. The design is a full factorial design.

A *minimum aberration* design is one that minimizes the number of confoundings for a given resolution. A minimum aberration design of a given resolution minimizes the number of *words* in the defining relation that are of minimum length. For a description of *words*, see "Change Generating Rules". For a discussion of minimum aberration designs, see Fries and Hunter (1984).

Display and Modify Design

In the Design List, if you select a fractional factorial design with all continuous or two-level categorical factors, and possibly a blocking factor, the Display and Modify Design outline opens after you click Continue. Modify your design using the reports in this outline. See "Modify Generating Rules in a Fractional Factorial Design" for an example of changing the generating rules to construct a design.

Note: The Change Generating Rules and Aliasing of Effects outlines do not appear for Plackett-Burman designs or Cotter designs, because interactions are not identically equal to main effects.

Change Generating Rules Specify the defining relation for the design. The defining relation determines which fraction of the full factorial design that JMP provides. See "Change Generating Rules".

Aliasing of Effects Shows the confounding pattern for the fractional factorial design. Click the red arrow at the bottom of the panel to see interactions to a specified order. The interactions and their aliases are presented in a data table.

Coded Design Shows the pattern of high and low values for the factors in each run.

Note: For Cotter designs, the Change Generating Rules and Aliasing of Effects outlines do not apply and are not shown.

Change Generating Rules

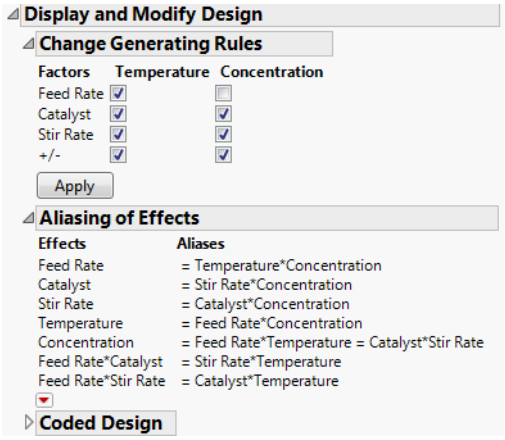
The generating rules define the relation used to construct a specific fractional factorial design. The default generator in the screening platform results in a minimum aberration design. A minimum aberration design minimizes the aliasing of lower-order effects. Your experimental situation might require that you define a fraction of the design that provides a coding or aliasing structure that is different from the standard fraction. You can do this by changing the generating rules in the Display and Modify Design outline. For more information about defining relations and generating rules, see Montgomery (2009).

Tip: If you want to create a design to estimate particular effects, consider using the Custom Designer. For more information about custom designs see "Custom Designs".

The defining relation for a design is determined by the *words* in the generating rules. A *word* is represented by a product of factors, but it is interpreted as the element wise product of the entries in the design matrix for those columns. A defining relation consists of words whose product is a column of ones, called the *identity*.

Figure 9.16 shows the default-generating rules for a 2^{5-2} fractional factorial design (five factors and eight runs).

Figure 9.16 Generating Rules for the Standard 2^{5-2} Design



In each column of the Change Generating Rules panel, the factor listed at the top and the factors in the column whose boxes are selected form a word in the defining relation. For example, the first column indicates that Temperature = Feed Rate*Catalyst*Stir Rate is a word in the defining relation.

- If the +/- box is selected, the sign associated with the generating rule is positive and the corresponding word equals the identity.
- If the +/- box is not selected, the sign associated with the generating rule is negative and the corresponding word equals minus the identity.

See "Obtain the Defining Relations in the 2^{5-2} Design".

The *principal fraction* of a full factorial design is the fractional factorial design obtained by setting all the defining relations equal to the identity. By default, the factorial design that JMP provides is the principal fraction. Notice that the +/- box is selected by default for all generating rules, so that each word in the defining relation equals the identity.

Generating rules determine the coding and aliasing of effects for the design. In some cases, you might want to use a fraction that results in a coding or an aliasing structure that differs from that of the standard fraction.

- To change the generating rules, select the appropriate boxes.
- To see the effect of your selections on the Aliasing of Effects results and on the Coded Design, click **Apply**.

For an example, see "Change the Generating Rules to Obtain a Different Fraction".

Obtain the Defining Relations in the 2^{5-2} Design

Figure 9.16 shows two columns of check boxes:

- The first column represents the word Temperature = Feed Rate*Catalyst*Stir Rate.
- The second column represents the word Concentration = Catalyst*Stir Rate.

Define I to represent a column consisting of the values +1. Because all factor levels are -1 or +1, the word in the first column is equivalent to Temperature*Feed Rate*Catalyst*Stir Rate = I. The word in the second column is equivalent to Concentration*Catalyst*Stir Rate = I. Together, these give the defining relations for the 2^{5-2} design:

I = Temperature*Feed Rate*Catalyst*Stir Rate = Concentration*Catalyst*Stir Rate

Obtain the Aliasing of Effects Relations in the 2^{5-2} Design

The aliasing structure in the Aliasing of Effects outline is determined by the defining relations and the fact that factor levels are +1 and -1. Recall that the first generating rule is Temperature = Feed Rate*Catalyst*Stir Rate and the second is Concentration = Catalyst*Stir Rate.

To obtain the first relation in the Aliasing of Effects outline, notice that applying these two generating rules gives the expression:

Temperature = Feed Rate*Catalyst*Stir Rate = Feed Rate*Concentration

– The second equality follows from replacing Catalyst*Stir Rate by Concentration using the second generating rule.

Now, post-multiply the first and third expressions by Concentration to obtain the following expression:

Temperature*Concentration = Feed Rate*Concentration*Concentration

Because the column for Concentration in the design matrix contains values of -1 and +1, the term Concentration*Concentration represents a column of +1 values. The expression becomes the first alias relation shown in the Aliasing of Effects outline:

Temperature*Concentration = Feed Rate*I = Feed Rate

The other alias relations can be obtained using similar calculations.

Main Effects Screening Designs

If an experiment involves categorical or discrete numeric factors, or if the number of runs is constrained, it might not be possible to construct an orthogonal design for screening main effects. However, a main effects screening design can be constructed. See Lekivetz et al. (2015).

A main effects screening design is a design with good balance properties as described by a Chi-square criterion. See "Chi-Square Efficiency". Such designs have desirable statistical properties for main effect models.

The algorithm used to generate the design attempts to construct an orthogonal array of strength two. Strength-two orthogonal arrays permit orthogonal estimation of main effects when interactions are negligible. These arrays are ideal for screening designs. Regular fractional factorial designs of Resolution 3 and Plackett-Burman designs are examples of strength-two orthogonal arrays.

Consider all possible pairs of levels for factors in the design. The algorithm attempts to balance the number of pairs of levels as far as possible. Given that a fixed number of columns has been generated, a new balanced column is randomly constructed. A measure is defined that reflects the degree of balance achieved for pairs that involve the new column. The algorithm attempts to minimize this measure by interchanging levels within the new column.

Chi-Square Efficiency

Suppose that a design has n runs and p factors corresponding to the columns of the design matrix.

- Denote the levels of factors k and l by $a = 0, 1, \ldots, s_k - 1$and $b = 0, 1, \ldots, s_l - 1$, respectively.
- Denote the number of times that the combination of levels (a,b) appears in columns k and l by $n_{kl}(a,b)$.

A measure of the lack of orthogonality evidenced by columns k and l is given by the following expression:

$$\chi^2_{kl} = \sum_{a=0}^{s_k-1} \sum_{b=0}^{s_l-1} \frac{[n_{kl}(a,b) - n/(s_k s_l)]^2}{n/(s_k s_l)}$$

A measure of the average non-orthogonality of the design is given by this expression:

$$\chi^2 = \sum_{1 \le k < l \le p} \chi^2_{kl} / [p(p-1)/2]$$

The maximum possible value of χ^2, denoted χ^2_{max}, is obtained. The chi-square efficiency of a design is defined as follows:

$$\text{Chi-Square Efficiency} = 100(1 - \chi^2/\chi^2_{max})$$

Chi-square efficiency indicates how close χ^2 is to zero, relative to a design in which pairs of levels show extreme lack of balance.

Design Generation

When you construct a main effects screening design, the Design Generation outline enables you to specify the number of runs. To generate the design, click **Make Design**.

Minimum A lower bound on the number of runs necessary to avoid failures in design generation. When you select Minimum, the resulting design is saturated. There are no degrees of freedom for error.

Note: If you select the Minimum number of runs, there is no error term for testing. You cannot test parameter estimates. This choice is appropriate only when the cost of additional runs is prohibitive.

Default Suggests the number of runs. This value is based on heuristics for creating a balanced design with at least four runs more than the Minimum number of runs.

User Specified Specify the number of runs that you want. Enter that value into the Number of Runs text box. This option enables you to balance the cost of additional runs against the potential gain in information.

Design

The Design outline shows the runs for the main effects screening design. To change the run order for your design table, you can select Run Order options in the Output Options panel before generating the table.

Design Evaluation

Note: The Design Evaluation outline is not shown for Cotter designs.

The Design Evaluation outline provides a number of ways to evaluate the properties of the generated design. Open the Design Evaluation outline to see the following options:

Power Analysis Enables you to explore your ability to detect effects of given sizes.

Prediction Variance Profile Shows the prediction variance over the range of factor settings.

Fraction of Design Space Plot Shows how much of the model prediction variance lies below (or above) a given value.

Prediction Variance Surface Shows a surface plot of the prediction variance for any two continuous factors.

Estimation Efficiency For each parameter, gives the fractional increase in the length of a confidence interval compared to that of an ideal (orthogonal) design, which might not exist. Also gives the relative standard error of the parameters.

Alias Matrix Gives coefficients that indicate the degree by which the model parameters are biased by effects that are potentially active, but not in the model.

Color Map on Correlations Shows the absolute correlation between effects on a plot using an intensity scale.

Design Diagnostics Indicates the optimality criterion used to construct the design. Also gives efficiency measures for your design.

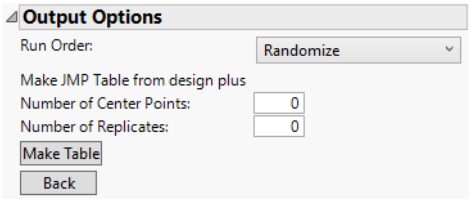
Note: The model used for the design diagnostics contains all main effects and two-factor interactions when all two-factor interactions are estimable. Otherwise, the model contains all main effects.

For more details about the Design Evaluation panel, see "Design Evaluation".

Output Options

Specify details for the output data table in the Output Options panel. When you have finished, click **Make Table** to construct the data table for the design. Figure 9.17 shows the Output Options panel for a standard design selected from the Design List. For a main effects screening design, only Run Order is available.

Figure 9.17 Select the Output Options



Run Order

The **Run Order** options determine the order of the runs in the design table.

Keep the Same Rows in the design table are in the same order as in the Coded Design or Design outlines.

Sort Left to Right Columns in the design table are sorted from left to right.

Randomize Rows in the design table are in random order.

Sort Right to Left Columns in the design table are sorted from right to left.

Randomize within Blocks Rows in the design table are in random order within the blocks. (Not available if you select **Construct a main effects screening design.**)

Center Points and Replicates

Number of Center Points Specifies how many additional runs to add as center points to the design. A center point is a run where every continuous factor is set at the center of the

factor's range. This option is not available if you select **Construct a main effects screening design**.

Suppose that your design includes both continuous and categorical factors. If you request center points in the Output Options panel, the center points are distributed as follows:

1. The settings for the categorical factors are ordered using the value ordering specified in the Factors outline.
2. One center point is assigned to each combination of the settings of the categorical factors in order, and this is repeated, until all center points are assigned.

Number of Replicates For designs in the Design List, specify the number of times to replicate the entire design, including center points. One replicate doubles the number of runs. This option is not available if you select **Construct a main effects screening design**.

Note: If you request center points or replicates and click Make Table repeatedly, these actions are applied to the most recently constructed design table.

Make Table

Click **Make Table** to create a data table that contains the runs for your experiment. In the table, the high and low values that you specified appear for each run.

Figure 9.18 The Design Data Table

Fractional Factorial 3
Design Fractional Factorial
Screening
Model
Evaluate Design
DOE Dialog
Columns (7/0)
Pattern
Feed Rate *
Catalyst *
Stir Rate *
Temperature *
Concentration *
Percent Reacted *

	Pattern	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Percent Reacted
1	−++−+	10	2	120	140	6	•
2	+−+−−	15	1	120	140	3	•
3	−+−+−	10	2	100	180	3	•
4	+++++	15	2	120	180	6	•
5	−−++−	10	1	120	180	3	•
6	+−−++	15	1	100	180	6	•
7	−−−−+	10	1	100	140	6	•
8	++−−−	15	2	100	140	3	•

The name of the table is the design type that generated it.

The design table includes the following scripts:

Screening Runs the DOE > Classical > Two Level Screening > Fit Two Level Screening platform. Only provided when all factors are at two levels.

Model Runs the Analyze > Fit Model platform.

Evaluate Design Runs the DOE > Design Diagnostics > Evaluate Design platform.

DOE Dialog Re-creates the Screening Design window that you used to generate the design table. The script also contains the random seed used to generate your design.

Run the **Screening** or **Model** scripts to analyze the data.

If the design was selected from the Design List, the design table contains a Pattern column. The Pattern column contains entries that summarize the run in the given row. Low settings are denoted by "–", high settings by "+", and center points by "0". Pattern can be useful as a label variable in plots.

Screening Design Options

The Screening Design red triangle menu contains the following options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Save X Matrix Saves the Moments Matrix and Model Matrix to table scripts in the design data table. These scripts contain the moments and design matrices. See "Save X Matrix".

Caution: For a design with nominal factors, the Model Matrix saved by the Save X Matrix option is *not* the coding matrix used in fitting the linear model. You can obtain the coding matrix used for fitting the model by selecting the option Save Columns > Save Coding Table in the Fit Model report that you obtain when you run the Model script

Suppress Cotter Designs Excludes Cotter designs from the Design List. This option is selected by default. Deselect it to show Cotter designs in the Design List.

Note: You can set a preference to always show Cotter designs. Select File > Preferences > Platforms > DOE and deselect Suppress Cotter Designs.

Number of Starts (Available only for Main Effects Screening Designs.) Specify the maximum number of times that the algorithm regenerates entire designs from scratch, attempting to optimize the final design.

Design Search Time (Available only for Main Effects Screening Designs.) Specify the maximum number of seconds spent searching for a design. The default search time is 15 seconds.

If the iterations of the algorithm require more than a few seconds, a Computing Design progress window appears. The progress bar displays *Chi2 Efficiency*. See "Chi-Square Efficiency". If you click **Cancel** in the progress window, the calculation stops and gives the best design found at that point.

Note: You can set a preference for Design Search Time. Select **File > Preferences > Platforms > DOE**. Select **Design Search Time** and enter the maximum number of seconds. If an orthogonal array is found, the search terminates. In certain situations where more time is required, JMP automatically extends the search time.

Number of Column Starts (Available only for Main Effects Screening Designs.) Specify the maximum number of times that the algorithm attempts to optimize a given column before moving on to constructing the next column. The default number of column starts is 50. See "Main Effects Screening Designs".

Save Script to Script Window Creates the script for the design that you specified in the Screening Design window and places it in an open script window.

Additional Examples of Screening Designs

- "Modify Generating Rules in a Fractional Factorial Design"
- "Plackett-Burman Design"

Modify Generating Rules in a Fractional Factorial Design

This example, adapted from Meyer, *et al*. (1996), shows how to use the Screening Design platform when you have many factors. In this example, a chemical engineer investigates the effects of five factors on the percent reaction of a chemical process. The factors are:

- Feed Rate - the amount of raw material added to the reaction chamber in liters per minute
- Catalyst (as a percent)
- Stir Rate - the RPMs of a propeller in the chamber
- Temperature (in degrees Celsius)
- Concentration of reactant

Production constraints limit the size of the experiment to no more than twelve runs. You decide to consider the 8-run fractional factorial design and the 12-run Plackett-Burman design. Also, you suspect the following statements to be true:

- The Temperature*Concentration interaction is active, so you want a design that does not alias this interaction with a main effect.
- The Catalyst*Temperature* interaction is not likely to be active.
- The Stir Rate*Concentration interaction is not likely to be active.

Use this information in constructing your design.

Create the Standard Fractional Factorial Design

To create the standard fractional factorial design, do the following:

- "Specify the Response"
- "Specify the Factors" or "Specify Factors Manually"
- "Choose a Design"

Specify the Response

1. Select **DOE > Classical > Two Level Screening > Screening Design**.
2. Double-click Y under Response Name and type Percent Reacted.

Note that the default Goal is Maximize. The Goal is to maximize the response, but the minimum acceptable reaction percentage is 90 (Lower Limit) and the upper limit is 100 (Upper Limit).

3. Click under Lower Limit and type 90.
4. Click under Upper Limit and type 100.
5. Leave the area under Importance blank.

   Because there is only one response, that response is given Importance 1 by default.

   See Figure 9.19 for the completed Responses outline. Now, specify the factors.

Specify the Factors

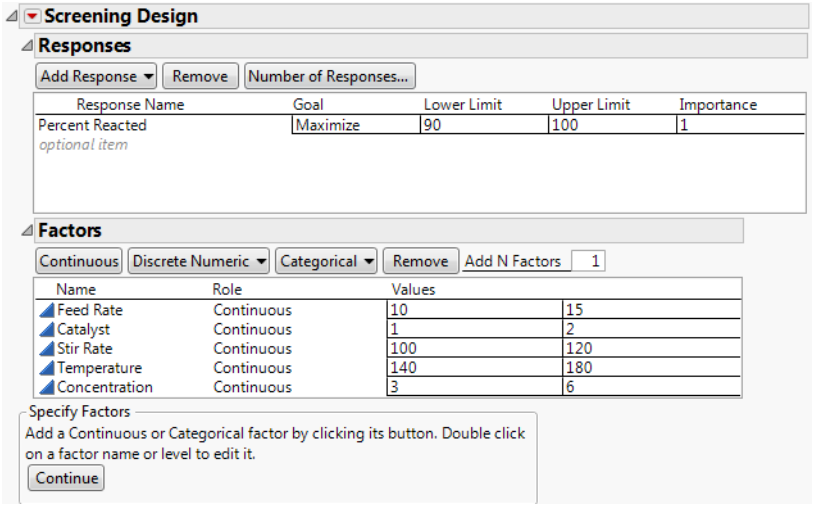
You can enter the factors manually or automatically:

- To enter the factors automatically, use the Reactor Factors.jmp data table:
  1. Select **Help > Sample Data Library** and open Design Experiment/Reactor Factors.jmp.
  2. Click the Screening Design red triangle and select **Load Factors**. Proceed to "Choose a Design".
- To enter the factors manually, follow the steps below.

Specify Factors Manually

1. Add five continuous factors by entering 5 in the **Add N Factors** box and clicking **Continuous**.
2. Change the default factor names (X1-X5) to Feed Rate, Catalyst, Stir Rate, Temperature, and Concentration.
3. Enter the low and high values for each factor:
   - Feed Rate: 10, 15
   - Catalyst: 1, 2
   - Stir Rate: 100, 120
   - Temperature: 140, 180
   - Concentration: 3, 6

Figure 9.19 Responses and Factors Outlines

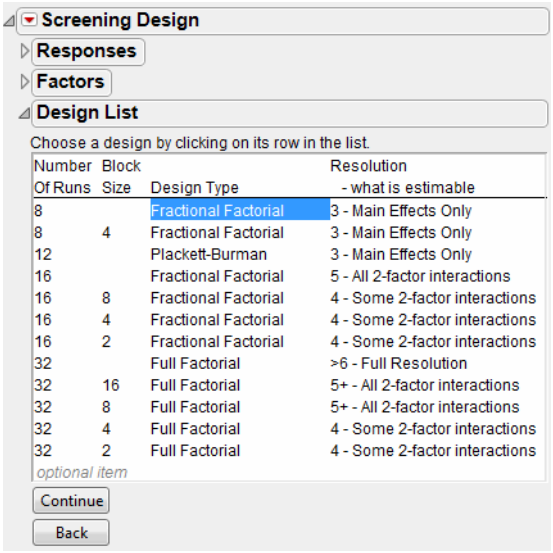


Choose a Design

1. Click **Continue**.
2. From the Choose Screening Type panel, accept the default selection to **Choose from a list of fractional factorial designs** and click **Continue**.

   Designs for the factors and levels that you specified are listed in the Design List (Figure 9.20).

Figure 9.20 Fractional Factorial Designs for Five Continuous Factors



3. The design that you want is the first in the list and happens to be selected by default (Figure 9.20). Accept that selection and click **Continue**.

   Because you are limited to eight runs and have no blocking factor, your best design option is the 8-run fractional factorial design with no blocks. This design is a 2^{5-2} fractional factorial design. It is one quarter of the full factorial design for five factors.

Change the Generating Rules to Obtain a Different Fraction

In this example, you want to know whether the Temperature*Concentration interaction is confounded with a main effect. Use the Display and Modify Design outline to view the aliasing structure for the design that you selected and to change it, if appropriate.

1. Open the Aliasing of Effects outline.

Figure 9.21 Aliasing of Effects Outline

Aliasing of Effects

Effects	Aliases
Feed Rate	= Temperature*Concentration
Catalyst	= Stir Rate*Concentration
Stir Rate	= Catalyst*Concentration
Temperature	= Feed Rate*Concentration
Concentration	= Feed Rate*Temperature = Catalyst*Stir Rate
Feed Rate*Catalyst	= Stir Rate*Temperature
Feed Rate*Stir Rate	= Catalyst*Temperature

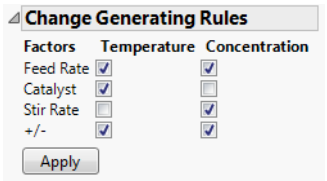
   The Temperature*Concentration interaction, which you suspect is active, is confounded with Feed Rate, a main effect. You want to change the generating rules to construct a design where Feed Rate is aliased with effects that you suspect are inactive, and where the Temperature*Concentration interaction is not aliased with a main effect.

2. Open the Change Generating Rules outline.

   The default-generating rules give you the standard (or *principal*) one-quarter fraction of the full factorial design. Recall that you suspect that the Catalyst*Temperature and Stir Rate*Concentration interactions are not likely to be active. Redefine the generating rules so that these two interactions are confounded with Feed Rate. The redefined generating rules give you a different one-quarter fraction of the full factorial design.

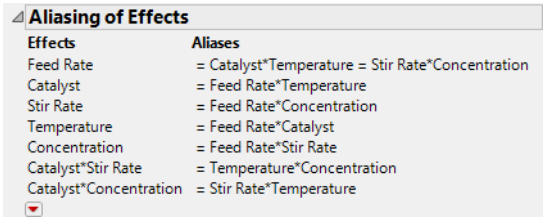
3. Do the following:
   - Deselect Stir Rate in the Temperature column.
   - Deselect Catalyst in the Concentration column.
   - Select Feed Rate in the Concentration column.

Figure 9.22 New Generating Rules



4. Click **Apply**.

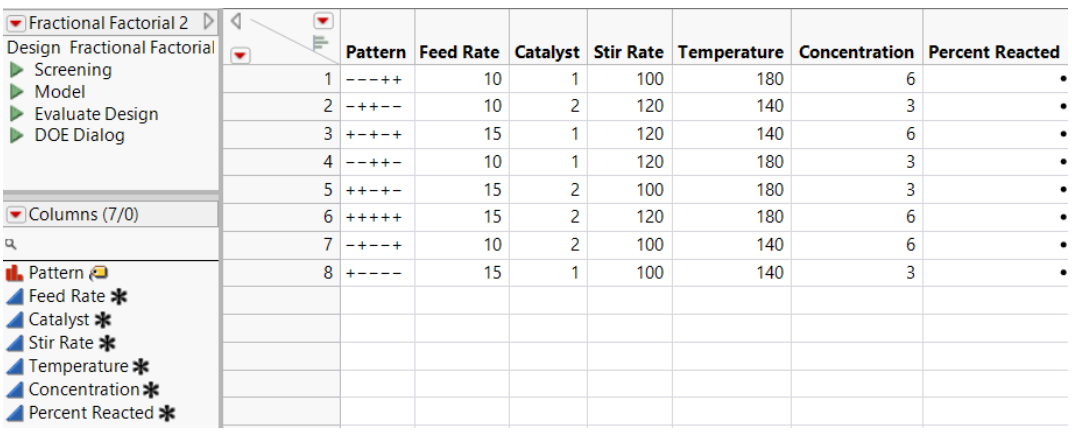
Figure 9.23 Aliasing of Effects Outline for Modified Generating Rules



In the design that you have defined, Feed Rate is confounded with Catalyst*Temperature and Stir Rate*Concentration. Also, the Temperature*Concentration interaction is now confounded with the two-way interaction Catalyst*Stir Rate.

5. In the Output Options outline, accept the default Run Order setting of **Randomize** and click **Make Table**.

Figure 9.24 Eight-Run Fractional Factorial Design Table



	Pattern	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Percent Reacted
1	−−−++	10	1	100	180	6	•
2	−++−−	10	2	120	140	3	•
3	+−+−+	15	1	120	140	6	•
4	−−++−	10	1	120	180	3	•
5	++−+−	15	2	100	180	3	•
6	+++++	15	2	120	180	6	•
7	−+−−+	10	2	100	140	6	•
8	+−−−−	15	1	100	140	3	•

The design table shows the design that you constructed. Notice that the table contains a column for the response that you defined in the Screening window, Percent Reacted, where you can record your experimental results.

The Screening, Model, and DOE Dialog scripts are also included. For more information about these scripts, see "Make Table".

Analyze the Results

Next you conduct the experiment, record your data, and proceed to analyze the results.

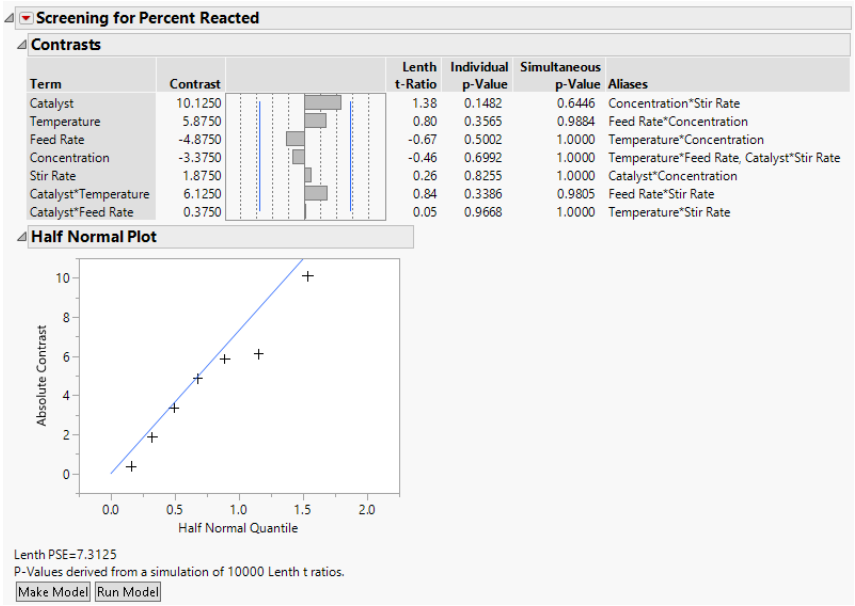
1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 8 Runs.jmp.

   You can estimate seven effects with your eight runs. Of these, you expect only a few to be active. Because you want to estimate seven effects, there are no degrees of freedom for error. For these reasons, you use the Screening platform to analyze the results.

2. Run the **Screening** script in the data table.

   The **Screening** script launches the Screening platform (**DOE > Classical > Two Level Screening > Fit Two Level Screening)** for your response and factors.

Figure 9.25 Report for Screening Example



Note: Since the *p*-values are obtained using a simulation-based technique, your *p*-values might not precisely match those shown here.

The report shows both Individual and Simultaneous *p*-values based on Lenth t-ratios. None of the effects are significant, even with respect to the Individual *p*-values. The Half Normal Plot suggests that the effects reflect only random noise.

For more information about the Screening report, see "The Screening Report".

Plackett-Burman Design

The Fractional Factorial example shows an 8-run fractional factorial design for five continuous factors. But suppose you can afford 4 additional runs. In this example, construct a 12-run Plackett-Burman design. To facilitate completing the Screening window, use the Load Responses and Load Factors commands.

Create the Plackett-Burman Design

1. Select **DOE > Classical > Two Level Screening > Screening Design**.
2. Select **Help > Sample Data Library** and open Design Experiment/Reactor Response.jmp.
3. Click the Screening Design red triangle and select **Load Responses**.
4. Select **Help > Sample Data Library** and open Design Experiment/Reactor Factors.jmp.
5. Click the Screening Design red triangle and select **Load Factors**.

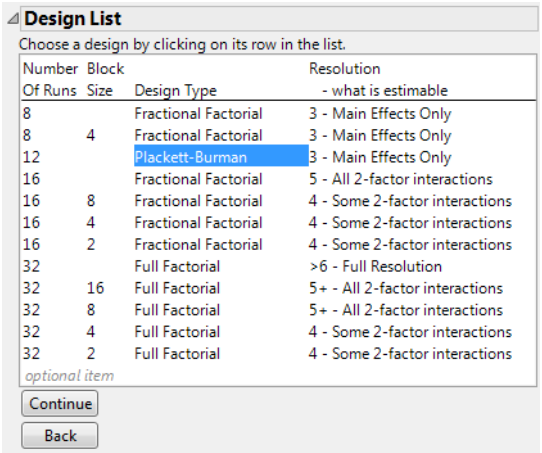
   The **Load Responses** and **Load Factors** commands fill in the Responses and Factors outlines with the response and factor names, goal and limits for the response, and values for the factors. See Figure 9.19 for the completed Responses and Factors outlines.
6. Click **Continue**.

Note: Setting the random seed in the next step reproduces the run order shown in this example. In constructing a design on your own, this step is not necessary.

7. (Optional) Click the Screening Design red triangle, select **Set Random Seed**, type 34567, and click **OK**.
8. From the Choose Screening Type panel, accept the default selection to **Choose from a list of fractional factorial designs** and click **Continue**.
9. Select the Plackett-Burman design.

   Plackett-Burman designs with run sizes that are not a power of two tend to have complex aliasing structures. In particular, main effects can be partially aliased with several two-way interactions. See "Evaluate the Design". Notice that the 12-run Plackett-Burman design is designated as having Resolution 3.

Figure 9.26 Design List Showing Plackett-Burman Screening Design



10. Click **Continue**.
11. Click **Make Table**.

Figure 9.27 Design Table for Placket-Burman Design

	Pattern	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Percent Reacted
1	+++++	15	2	120	180	6	•
2	+---+	15	1	100	140	6	•
3	+-+++	15	1	120	180	6	•
4	-+++-	10	2	120	180	3	•
5	++---	15	2	100	140	3	•
6	---+-	10	1	100	180	3	•
7	-+-++	10	2	100	180	6	•
8	+++--	15	2	120	140	3	•
9	--+-+	10	1	120	140	6	•
10	--+--	10	1	120	140	3	•
11	-+--+	10	2	100	140	6	•
12	+--+-	15	1	100	180	3	•

A column called Percent Reacted is included in the design table. You should conduct your experimental runs in the order shown in the table, recording your results in the Percent Reacted column.

Evaluate the Design

1. Return to your Screening Design window. If you have closed this window, run the DOE Dialog script in your design table.
2. Open the **Design Evaluation > Color Map on Correlations** outline.

Figure 9.28 Color Map for Absolute Correlations

The diagonal cells have correlations of one, as expected. White cells correspond to effects that have correlations equal to 0. The gray and black cells correspond to effects that have correlations greater than zero. Hover over cells to see the effects involved and their absolute correlations. For example, notice that Feed Rate is correlated with several two-way and three-way interactions.

Note: The color map from the screening design contains 3-way interactions for designs with 8 or fewer factors. The color map from the design evaluation script only contains up to 2-way interactions.

3. Open the Alias Matrix outline.

Figure 9.29 Alias Matrix - Partial View Showing Up to Two-Way Interactions

Alias Matrix

Effect	Feed Rate*Catalyst	Feed Rate*Stir Rate	Feed Rate*Temperature	Feed Rate*Concentration	Catalyst*Stir Rate	Catalyst*Temperature	Catalyst*Concentration	Stir Rate*Temperature	Stir Rate*Concentration	Temperature*Concentration
Intercept	0	0	0	0	0	0	0	0	0	0
Feed Rate	0	0	0	0	0.333	-0.33	-0.33	0.333	0.333	0.333
Catalyst	0	0.333	-0.33	-0.33	0	0	0	0.333	-0.33	0.333
Stir Rate	0.333	0	0.333	0.333	0	0.333	-0.33	0	0	0.333
Temperature	-0.33	0.333	0	0.333	0.333	0	0.333	0	0.333	0
Concentration	-0.33	0.333	0.333	0	-0.33	0.333	0	0.333	0	0

Because the design is orthogonal for the main effects, the Alias Matrix gives the numerical values of the correlations between effects. See "Alias Matrix". For example, notice that Feed Rate is partially aliased with six two-way interactions and with four three-way interactions. These are the interactions corresponding to the entries of 0.333 and -0.33 in the row for Feed Rate.

Analyze the Results

The data table Plackett-Burman.jmp contains the results of the designed experiment. Recall that you suspect that the Temperature*Concentration interaction is active. You proceed under the assumption that this is the only potentially active interaction.

1. Select **Help > Sample Data Library** and open Design Experiment/Plackett-Burman.jmp.
2. Run the **Model** script by clicking the icon to its left.
3. Select Temperature in the Select Columns list and Concentration in the Construct Model Effects list.
4. Click **Cross**.
5. Click **Run**.

Figure 9.30 Parameter Estimates for Full Model

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	66.166667	2.639444	25.07	<.0001*
Feed Rate(10,15)	-2.333333	2.850926	-0.82	0.4503
Catalyst(1,2)	10.5	2.850926	3.68	0.0142*
Stir Rate(100,120)	1.3333333	2.850926	0.47	0.6597
Temperature(140,180)	5	2.639444	1.89	0.1167
Concentration(3,6)	-0.5	2.639444	-0.19	0.8572
Concentration*Temperature	-6.5	3.232646	-2.01	0.1006

The Actual by Predicted Plot indicates no lack of model fit. The Parameter Estimates report shows that Catalyst is significant at the 0.05 level and that the Concentration*Temperature interaction is almost significant at the 0.10 level.

Reduce the Model

You want to identify those effects that have the most impact on the response. To see these active effects more clearly, remove insignificant effects using the Effect Summary outline.

Figure 9.31 Effect Summary Outline for Full Model

Effect Summary

Source	LogWorth	PValue
Catalyst(1,2)	1.846	0.01425
Concentration*Temperature	0.998	0.10055
Temperature(140,180)	0.933	0.11671 ^
Feed Rate(10,15)	0.346	0.45033
Stir Rate(100,120)	0.181	0.65969
Concentration(3,6)	0.067	0.85720 ^

Remove Add Edit FDR ('^' denotes effects with containing effects above them)

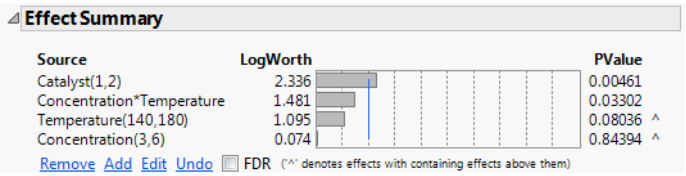
Although Concentration is the least significant effect, it is involved in a higher-order interaction (Concentration*Temperature), as indicated by the caret to the right of its PValue. Based on the principle of effect heredity, Concentration should not be removed from the model while the Concentration*Temperature interaction remains in the model. See "Effect Heredity". The next least significant effect is Stir Rate.

1. In the Effect Summary outline, select Stir Rate and click **Remove**.

   Feed Rate is the next least significant effect that can be removed.

2. In the Effect Summary outline, select Feed Rate and click **Remove**.

Figure 9.32 Effect Summary Outline for Reduced Model



The PValue column indicates that the Catalyst main effect and the Concentration*Temperature interaction are both significant at the 0.05 level. The model should not be reduced any further. If all other interactions are inactive or negligible, then you can conclude that Catalyst and the Concentration*Temperature interaction are active effects.

Chapter 10

The Fit Two Level Screening Platform

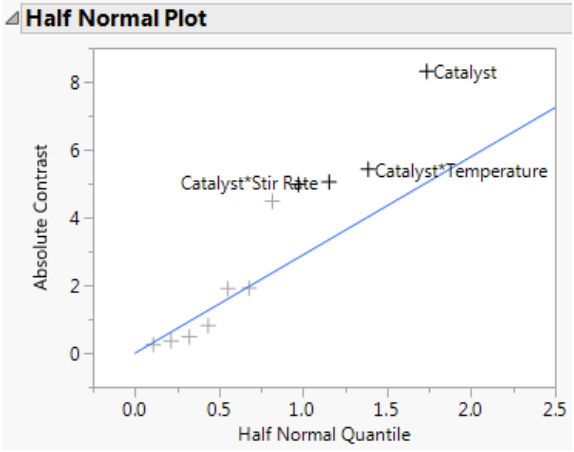
Analyze Data from Screening Experiments

The Fit Two Level Screening platform is a modeling platform that you can use to analyze experimental data results from a screening design. The Fit Two Level Screening platform helps you identify effects that have a large impact on the response.

The Fit Two Level Screening platform is based on the principle of effect sparsity (Box and Meyer 1986). This principle asserts that relatively few of the effects that you study in a screening design are active. Most are inactive, meaning that their true effects are negligible and that their estimates can be treated as random error.

A screening design often provides no degrees of freedom for error when the model of interest includes interaction terms. Consequently, classical tests for effects are not available. In such cases, the Fit Two Level Screening platform is particularly useful.

Figure 10.1 Half Normal Plot from Fit Two Level Screening Report



Contents

Overview of the Fit Two Level Screening Platform 339
Example of Fit Two Level Screening Platform 339
Launch the Fit Two Level Screening Platform 342
The Screening Report 343
Contrasts 344
Half Normal Plot 344
Make or Run Model 345
Additional Fit Two Level Screening Analysis Examples 345
Analyze a Plackett-Burman Design 345
Effect Heredity Example 346
Analyze a Supersaturated Design 348
Technical Details for the Fit Two Level Screening Platform 350
Order of Effect Entry 350
Fit Two Level Screening as an Orthogonal Rotation 350
Lenth's Pseudo-Standard Error 351
Lenth t-Ratios 351

Overview of the Fit Two Level Screening Platform

The analysis of screening designs depends on the principle of *effect sparsity*, where most of the variation in the response is explained by a small number of effects. Under this principle, effects with small estimates are used to estimate the error in the model. This then allows one to test whether the larger effects are active.

You can analyze data from a screening experiment using Fit Model (Analyze > Fit Model) or Fit Two Level Screening (DOE > Classical > Two Level Screening > Fit Two Level Screening). Use the following guidelines to select the appropriate modeling platform:

- If your factors are all two-level and orthogonal, all of the statistics in the Fit Two Level Screening platform are appropriate.
- If you have data from a highly supersaturated main effect design, the Fit Two Level Screening platform is effective in selecting active factors, but it is not effective at estimating the error or the significance. The Monte Carlo simulation to produce *p*-values uses assumptions that are not valid for this case.
- If you have a categorical or a discrete numeric factor with more than two levels, the Fit Two Level Screening platform is not appropriate.JMP treats the associated model terms as continuous. For such factor, the variation is scattered across main effects and polynomial effects. In this situation, it is recommended that you use the Fit Model platform.
- If your data are not orthogonal, the constructed estimates in the Fit Two Level Screening Platform are different from standard regression estimates. JMP can identify large effects, but it does not effectively test each effect. This is because effects are artificially orthogonalized as they are entered into the model, making effects that enter early in the model appear less significant than in standard regression See "Order of Effect Entry".
- For mixture designs, the Fit Two Level Screening platform is not appropriate. See instead "Fitting Mixture Designs".

Example of Fit Two Level Screening Platform

The Reactor Half Fraction.jmp sample data table contains the results of an experiment derived from a design discussed in Box et al. (1978). You are interested in identifying significant effects in a model that contains main effects and two-way interactions. This example uses a model with fifteen parameters for a design with sixteen runs. The example illustrates the analysis both with the Fit Two Level Screening Platform and the Fit Model Platform.

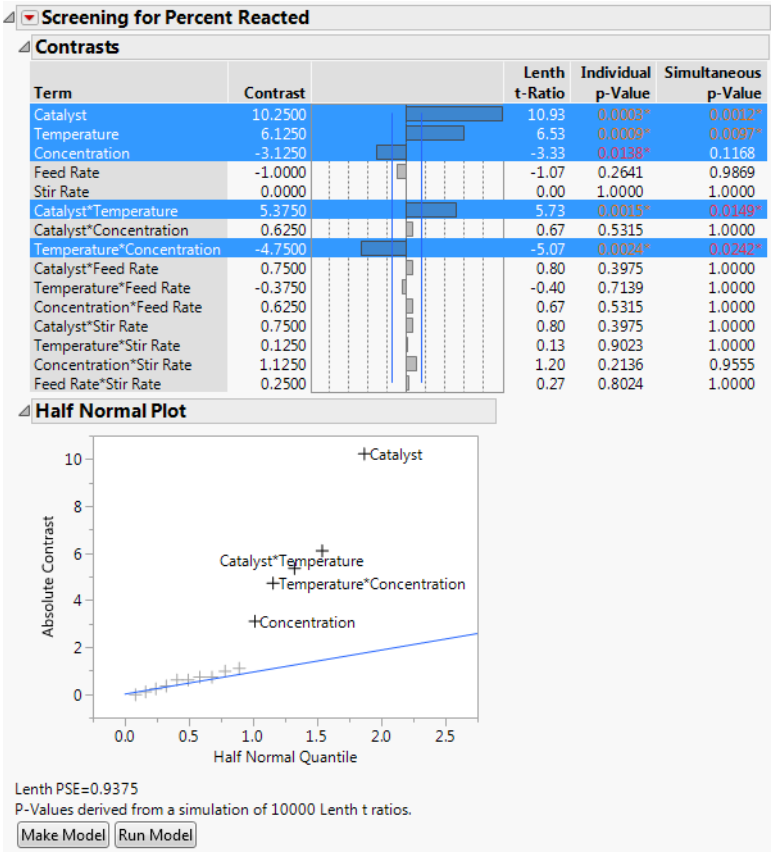
Fit Two Level Screening

1. Select **Help > Sample Data Library** and open Reactor Half Fraction.jmp.

2. Select **DOE > Classical > Two Level Screening > Fit Two Level Screening**.
3. Select Percent Reacted and click **Y**.
4. Select Feed Rate through Concentration, click **X**, and then click **OK**.

Note: The Fit Two Level Screening Platform automatically constructs a model with interaction terms. This is in contrast to the Fit Model Platform, where you manually specify the interactions that you want to include in your model.

Figure 10.2 Reactor Half Fraction.jmp Fit Two Level Screening Design Report



Note the following features of the Screening report:

- The effects that have an individual *p*-value less than 0.10 are selected.
- A *t*-ratio is calculated using Lenth's PSE (pseudo-standard error). The Lenth PSE value is shown below the Half Normal Plot.
- Both individual and simultaneous *p*-values are shown. Those that are less than 0.05 are shown with an asterisk.

– The Half Normal Plot enables you to quickly examine the effects. The effects that are initially selected in the effects list are also labeled in this plot.

In this example, Catalyst, Temperature, and Concentration, along with two of their two-factor interactions, are selected. Alternatively, you can fit the same model in the Fit Model Platform.

Fit Model

1. From the Reactor Half Fraction.jmp table, select **Analyze > Fit Model**.
2. Select Percent Reacted, **click Y**.
3. Select Feed Rate through Concentration, and then select **Macros > Factorial to Degree**.
4. **Click Run**.
5. Open the Parameter Estimates outline.

Figure 10.3 Saturated Reactor Half Fraction.jmp Design Parameter Estimates

Response Percent Reacted

Effect Summary

Parameter Estimates

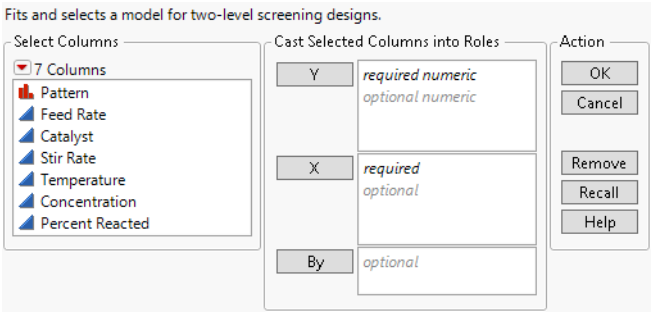
Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	65.25	.	.	.
Feed Rate(10,15)	-1	.	.	.
Catalyst(1,2)	10.25	.	.	.
Stir Rate(100,120)	0	.	.	.
Temperature(140,180)	6.125	.	.	.
Concentration(3,6)	-3.125	.	.	.
Feed Rate*Catalyst	0.75	.	.	.
Feed Rate*Stir Rate	0.25	.	.	.
Feed Rate*Temperature	-0.375	.	.	.
Feed Rate*Concentration	0.625	.	.	.
Catalyst*Stir Rate	0.75	.	.	.
Catalyst*Temperature	5.375	.	.	.
Catalyst*Concentration	0.625	.	.	.
Stir Rate*Temperature	0.125	.	.	.
Stir Rate*Concentration	1.125	.	.	.
Temperature*Concentration	-4.75	.	.	.

Since there are 16 observations and 16 model terms, there are not enough observations to estimate an error term. Without an estimate of error, it is not possible to conduct standard tests. Parameter estimates are provided, but because there are no degrees of freedom for error, standard errors, *t*-ratios, and *p*-values are all missing. This illustrates the strength of the Fit Two Level Screening platform for getting the most information out of a screening design.

Launch the Fit Two Level Screening Platform

Launch the Fit Two Level Screening platform by selecting **DOE > Classical > Two Level Screening> Fit Two Level Screening**.

Figure 10.4 Launch Window for the Fit Two Level Screening Platform



Y The columns to be analyzed. These must have a Numeric data type.

X Continuous or discrete two-level orthogonal factors.

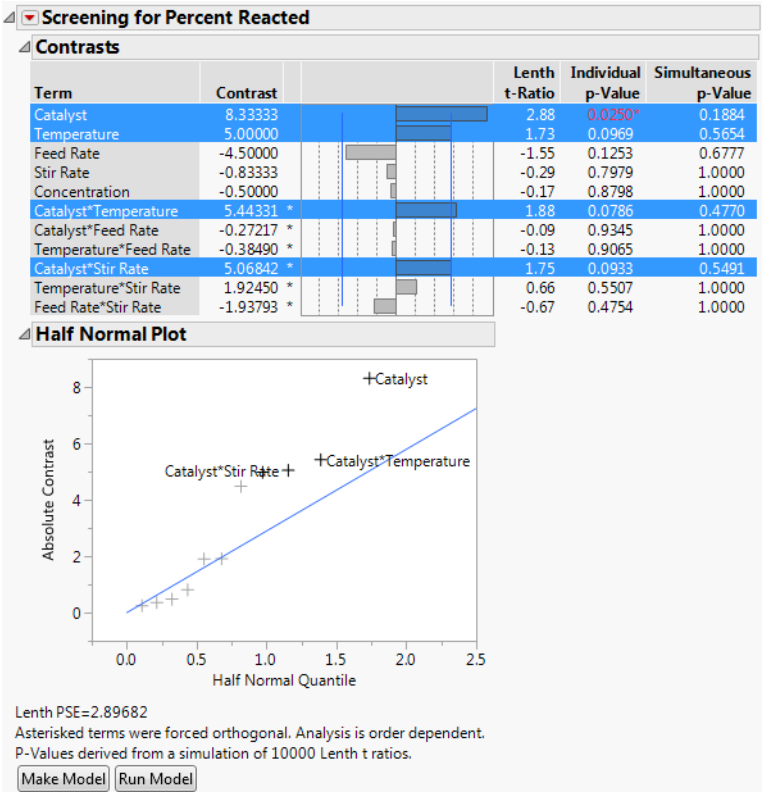
By A column whose level define separate analyses. For each level of the specified column, the corresponding rows are analyzed. The results appear in separate reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Note: If you created your design in the Screening Design Platform from a list of fractional factorial designs, the design table contains a script called **Screening.** Run this script to run the analysis directly.

The Screening Report

The Fit Two Level Screening report shows a summary table of the design contrasts (candidate model terms) and a half normal plot. You can launch or run the selected model from the report.

Figure 10.5 Screening Report



- "Contrasts"
- "Half Normal Plot"
- "Make or Run Model"

Contrasts

The Contrasts report lists model effects, a contrast value, Lenth *t*-ratios, individual and simultaneous *p*-values, and aliases (if there are any). Effects are entered into the analysis following a hierarchical ordering. See "Order of Effect Entry". Effects that have an Individual *p*-Value less than 0.10 are selected.

Term The name of the factor.

Contrast Estimate for the factor. For orthogonal designs, this number is the same as the regression parameter estimate. This is not the case for non-orthogonal designs. An asterisk might appear next to the contrast, indicating a lack of orthogonality.

Bar Chart Shows the Lenth *t*-ratios with blue vertical lines that indicate a value that is significant at the 0.10 level.

Lenth t-Ratio Lenth's *t*-ratio, calculated as *Contrast*/*PSE*, where PSE is Lenth's Pseudo-Standard Error. See "Lenth's Pseudo-Standard Error".

Individual p-Value Analogous to the standard *p*-values for a linear model. Small individual *p*-values indicate a significant effect. See "Lenth's Pseudo-Standard Error".

Note: Do not expect the *p*-values to be exactly the same if the analysis is re-run. The Monte Carlo method should give similar, but not identical, values if the same analysis is repeated.

Simultaneous p-Value A multiple-comparison adjusted *p*-value.

Aliases Appears only if there are exact aliases of later effects to earlier effects.

Half Normal Plot

The Half Normal Plot shows the absolute value of the contrasts plotted against the absolute value of quantiles for the half-normal distribution. The blue line passes through the origin with a slope of the Lenth's estimate of σ. Effects that are small are considered error terms and assumed to have a normal distribution with mean zero and standard deviation σ. These terms fall on the blue line. Significant effects are those that have nonzero means and do not fall on the blue line.

The Half Normal Plot is interactive. You can select model effects by dragging a rectangle around the effects of interest. You can also press Ctrl and click an effect name(s) in the report.

Make or Run Model

The **Make Model** button launches a Fit Model window that is populated with the selected effects.

The **Run Model** button runs an effect screening model for the selected effects.

Note: A JMP alert appears if your selected model does not follow effect heredity. You can **Cancel**, use the **Make Model** button to launch the Fit Model window, and add the missing terms. Or, you can **Continue** and build a model that does not follow effect heredity rules. See "Effect Heredity".

Additional Fit Two Level Screening Analysis Examples

- "Analyze a Plackett-Burman Design"
- "Effect Heredity Example"
- "Analyze a Supersaturated Design"

Analyze a Plackett-Burman Design

Plackett-Burman designs are an alternative to fractional-factorial screening designs. Two-level fractional factorial designs must, by their nature, have a number of runs that are a power of two. However, Plackett-Burman designs exist for 12-, 24-, and 28-run designs.

The Weld-Repaired Castings.jmp sample data table uses a Plackett-Burman design, and is found in Box et al. (1978). Seven factors are thought to be influential on weld quality. The seven factors include Initial Structure, Bead Size, Pressure Treatment, Heat Treatment, Cooling Rate, Polish, and Final Treatment. A Plackett-Burman design with 12 runs is used to investigate the importance of the seven factors. The response is 100 × log(lifetime). (The sample data table also contains four terms that were used to model error, but those terms are not used in this analysis.)

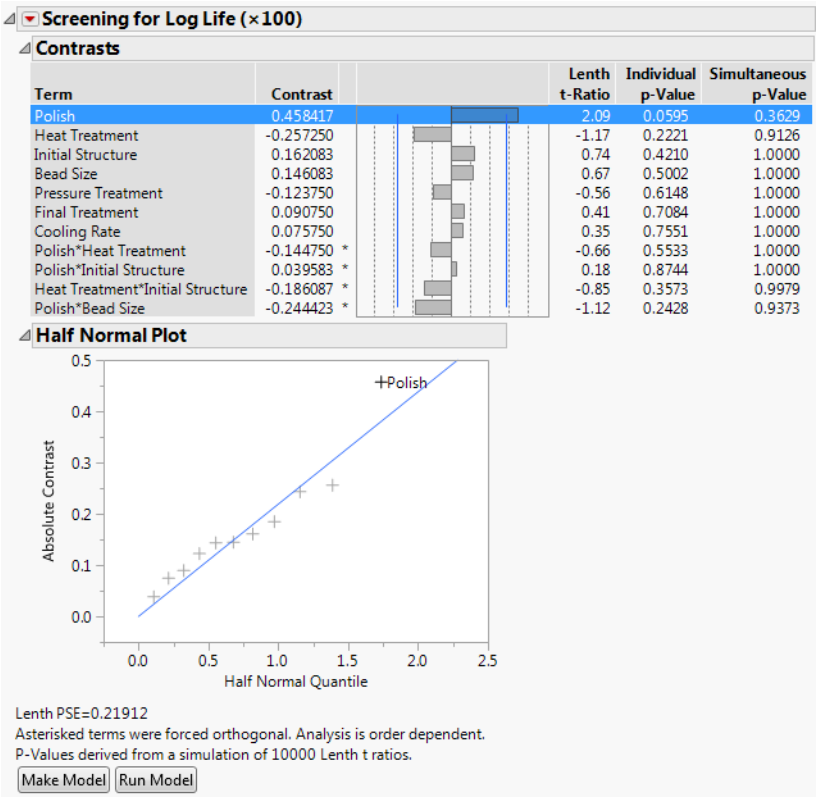
1. Select **Help > Sample Data Library** and open Weld-Repaired Castings.jmp.
2. Select **DOE > Classical > Two Level Screening > Fit Two Level Screening**.

   The launch window is populated based on the column properties. Log Life (x100) is the response **Y**.

   The seven factors Initial Structure, Bead Size, Pressure Treatment, Heat Treatment, Cooling Rate, Polish, and Final Treatment are specified for **X**.

3. Scroll to the bottom of the **X** window. Select ε1, ε2, ε3, and ε4, and click **Remove**.
4. **Click OK**.

Figure 10.6 Screening Report for Weld-Repaired Castings.jmp



The only significant effect identified is Polish. Note that asterisks mark four terms, indicating that they are not orthogonal to the effects preceding them. The contrast values obtained for these effects are after orthogonalization. As a result, these estimates do not match estimates obtained from a corresponding regression analysis. You can use the **Run Model** button to fit the single-factor model.

Effect Heredity Example

This example illustrates building a model when the Fit Two Level Screening platform identifies a model that does not follow effect heredity.

1. Select **Help > Sample Data Library** and open Design Experiment/Plackett-Burman.jmp.
2. Select **DOE > Classical > Two Level Screening> Fit Two Level Screening**.
3. Select Percent Reacted and click **Y**.

4. Select Feed Rate through Concentration and click **X**.
5. Click **OK**.

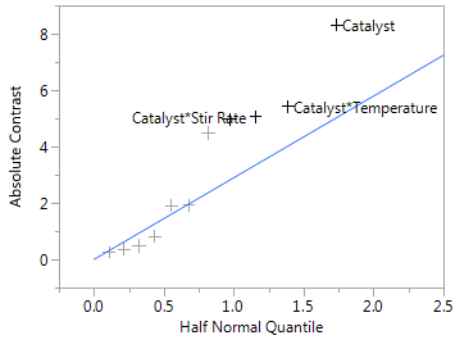
Figure 10.7 Model Contrasts

Screening for Percent Reacted

Contrasts

Term	Contrast		Lenth t-Ratio	Individual p-Value	Simultaneous p-Value
Catalyst	8.33333		2.88	0.0249*	0.1786
Temperature	5.00000		1.73	0.0979	0.5545
Feed Rate	-4.50000		-1.55	0.1248	0.6650
Stir Rate	-0.83333		-0.29	0.7970	1.0000
Concentration	-0.50000		-0.17	0.8796	1.0000
Catalyst*Temperature	5.44331	*	1.88	0.0792	0.4632
Catalyst*Feed Rate	-0.27217	*	-0.09	0.9331	1.0000
Temperature*Feed Rate	-0.38490	*	-0.13	0.9054	1.0000
Catalyst*Stir Rate	5.06842	*	1.75	0.0946	0.5380
Temperature*Stir Rate	1.92450	*	0.66	0.5510	1.0000
Feed Rate*Stir Rate	-1.93793	*	-0.67	0.4720	1.0000

Half Normal Plot



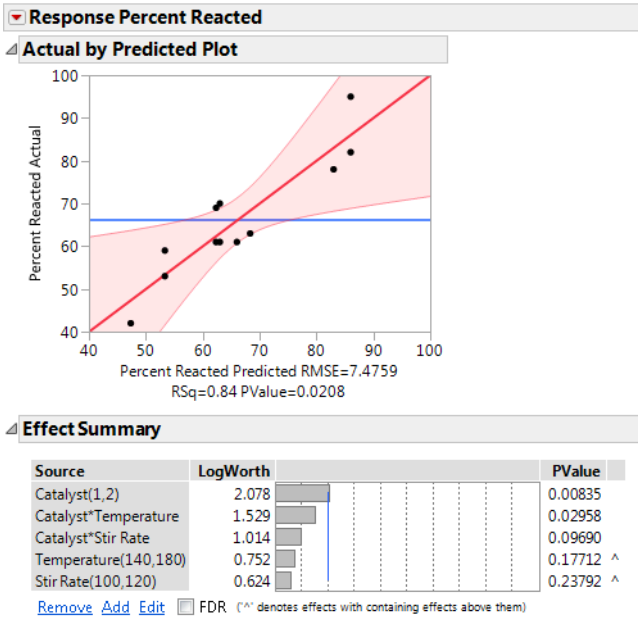
Lenth PSE=2.89682
Asterisked terms were forced orthogonal. Analysis is order dependent.
P-Values derived from a simulation of 10000 Lenth t ratios.

Make Model Run Model

TheCatalyst*Stir Rate interaction is highlighted, but the Stir Rate main effect is not. In accordance with the principle of Effect Heredity, you should add the Stir Rate main effect to the model. See "Effect Heredity".

6. Click **Make Model**.
7. Select Stir Rate and click **Add** in the Construct Model Effects section.
8. Click **Run**.

Figure 10.8 Partial Fit Model Report



The ^ symbol to the right of the PValue for Stir Rate and Temperature show that while not significant, these effects are contained in significant higher-order effect.

Analyze a Supersaturated Design

Supersaturated designs have more factors than runs. They rely heavily on effect sparsity for their analysis, so the Fit Two Level Screening platform is ideal for their analysis. The objective is to determine which effects are active.

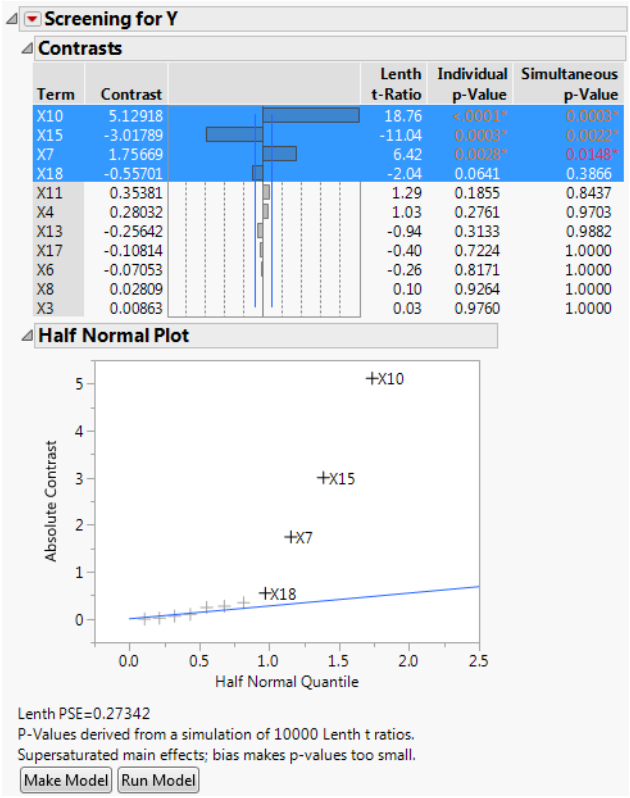
In this example, a simulated design with 18 factors but only 12 runs is considered. Y is generated by:

$$Y = 2(X7) + 5(X10) - 3(X15) + \varepsilon$$

where $\varepsilon \sim N(0,1)$. So, Y has been constructed with three active factors.

1. Select **Help > Sample Data Library** and open Supersaturated.jmp.
2. Select **DOE > Classical > Two Level Screening > Fit Two Level Screening**.
3. Select Y and click **Y**.
4. Select X1 through X18 and click **X**.
5. Click **OK**.

Figure 10.9 Screening Report for Supersaturated.jmp



Note that four factors have been highlighted. Factors X10, X15, and X7 are active based on a 0.05 critical value criteria. In the Half Normal Plot, X18 is close to the blue line, which indicates that it is close to the 0.1 cutoff value for significance. The 0.1 critical value is generous in its selection of factors so that you do not miss those that are possibly active.

The contrast estimates for X10, X15, and X7 of 5.1, –3, and 1.8 are close to their simulated values (5, –3, 2).

The *p*-values are not entirely valid statistically, since they are based on a simulation that assumes orthogonal designs, which is not the case for supersaturated designs. However, as shown, they can be useful in identifying effects for further investigation.

Technical Details for the Fit Two Level Screening Platform

- "Order of Effect Entry"
- "Fit Two Level Screening as an Orthogonal Rotation"
- "Lenth's Pseudo-Standard Error"
- "Lenth t-Ratios"

Order of Effect Entry

The Fit Two Level Screening platform has a carefully defined order of operations:

1. First, the main effect terms enter the model according to the absolute size of their contrast. All effects are orthogonalized to the effects preceding them in the model. This method assures that their order is the same as it would be in a forward stepwise regression. Ordering by main effects also helps in selecting preferred aliased terms later in the process.
2. After main effects, all second-order interactions enter the model, followed by third-order interactions, and so on. The second-order interactions cross with all earlier terms before bringing in a new term because B and C are both larger than D. The complete order of entry is AB, AC, BC, AD, BD, and then CD.
3. An effect that is an exact alias for an effect already in the model appears in the Alias column. Effects that are a linear combination of several previous effects do not appear. If there is partial aliasing (due to a lack of orthogonality), the effects involved are marked with an asterisk.
4. The process continues until n effects are obtained, where n is the number of rows in the data table, thus fully saturating the model. If complete saturation is not possible with the factors, JMP generates random orthogonalized effects to absorb the rest of the variation. They are labeled Null n where n is a number. For example, this situation occurs if there are exact replicate rows in the design.

Fit Two Level Screening as an Orthogonal Rotation

Mathematically, the Fit Two Level Screening platform takes the n values in the response vector and rotates them into n new values. The rotated values are then mapped by the space of the factors and their interactions.

$$\text{Contrasts} = \mathbf{T}' \times \text{Responses}$$

In the above equation, **T** is an n by n orthonormalized set of values starting with the intercept, main effects of factors, two-way interactions, three-way interactions, and so on, until n values have been obtained. Since the first column of **T** is an intercept, and all the other columns are orthogonal to it, these other columns are all contrasts, that is, they sum to zero. Since **T** is orthogonal, it can serve as **X** in a linear model. It does not need inversion, since $\mathbf{T}'$ is equivalent to $\mathbf{T}^{-1}$ and $(\mathbf{T}'\mathbf{T})\mathbf{T}'$. The contrasts are the parameters estimated in a linear model.

If no effect in the model is active other than the intercept, the contrasts are just an orthogonal rotation of random independent variates into different random independent variates. These orthogonally-rotated random variates have the same variance as the original random independent variates. To the extent that some effects are active, the inactive effects still represent the same variation as the error in the model. The hope is that the effects and the design are strong enough to separate the active effects from the random error effects.

Lenth's Pseudo-Standard Error

Lenth's method (Lenth 1989), known as the *Lenth Pseudo Standard Error* (*PSE*), constructs an estimate of the residual standard error using effects that appear to be inactive. Lenth's PSE can be used to estimate the standard error for experiments where contrasts are independent and have a common variance.

If there are n rows, the platform constructs $n - 1$ contrasts. Denote these contrasts by $\hat{C}_i$, where $i = 1, \ldots, n - 1$.

To obtain Lenth's PSE, first calculate the following:

$$v = 1.5[median_{i = 1, \ldots, n-1}|\hat{C}_i|]$$

Lenth's PSE is based on the effects that are likely to be inactive and is defined as follows:

$$PSE = 1.5\left[median_{|\hat{C}_i| < 2.5v}|\hat{C}_i|\right]$$

The value for Lenth's PSE is shown at the bottom of the Screening report.

Lenth t-Ratios

For each contrast, a t-ratio is computed by dividing the contrast by the PSE. The reference distribution of these t-ratios under the null hypothesis is not computationally tractable. Therefore, it is obtained by simulation. The method, described below, is based on a discussion in Ye and Hamada (2000).

Denote the t-ratio for the i^{th} contrast by t_i:

$$t_i = \hat{C}_i/(PSE)$$

Of primary importance in screening experiments is the *individual error rate*, namely the probability of declaring a given effect as active when it is not. For the i^{th} effect, this occurs when $|t_i|$ is large and falls into the upper tail of its reference distribution.

Because the platform constructs a relatively large number of effects, the *experimentwise error rate* is also of importance. The experimentwise error rate is the probability of declaring *any* effect as active when no effects are active. An experimentwise error occurs when no effects are active and the maximum of the absolute t-ratios, $\max|t_i|$, is large and falls into the upper tail of its reference distribution.

The Fit Two Level Screening platform obtains reference distributions for both types of error rates using Monte Carlo simulation. Consider a set of $n - 1$ values that is simulated from a normal distribution with mean 0 and standard deviation equal to PSE. This set of values represents potential contrast values for the experiment under the null hypothesis of no active effects. In all, 10,000 sets of $n - 1$ random contrast values are generated.

Individual p-Values

Consider the i^{th} contrast. Lenth t-ratios are constructed using each of the 10,000*($n - 1$) simulated values. The reference distribution for the individual error rate is approximated by the absolute values of these t-ratios. The Individual p-value is the interpolated fractional position of the observed absolute Lenth t-Ratio among the 10,000*($n - 1$) simulated absolute t-ratios arranged in descending order. This approximates the area to the right of the absolute value of the observed absolute Lenth t-Ratio with respect to the reference distribution.

Simultaneous p-Values

An experimentwise error occurs if any t-ratio leads to rejecting the null hypothesis when all effects are inactive. Equivalently, an experimentwise error occurs if the maximum of the absolute t-ratios, $\max|t_i|$, leads to rejecting the null hypothesis.

To obtain a reference distribution in this case, consider the maximum of the computed absolute t-ratios in each of the 10,000 simulations. These 10,000 maximum values form the reference distribution. The Simultaneous p-value is the interpolated fractional position of the observed absolute Lenth t-Ratio among the 10,000 simulated maximum absolute t-ratios arranged in descending order. This approximates the area to the right of the absolute value of the absolute Lenth t-Ratio with respect to the reference distribution based on the simulated maximum absolute t-ratios.

Monte Carlo Simulation Options

To change the number of default sets of simulations from 10,000, you must assign a value to a global JSL variable named `LenthSimN`. The number of simulations used to derive the *p*-values is reported in the report window.

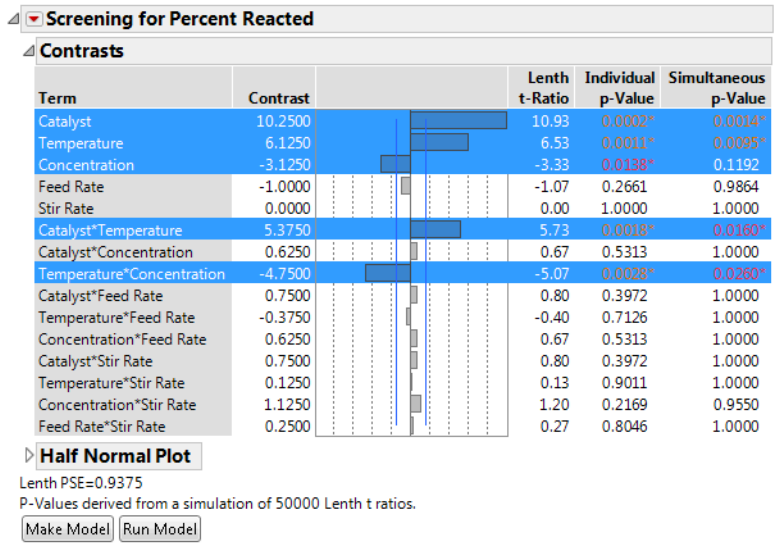
The following example sets the number of simulations to 50,000.

1. Select **Help > Sample Data Library** and open Reactor Half Fraction.jmp.
2. Select **DOE > Classical > Two Level Screening > Fit Two Level Screening**.
3. Select Percent Reacted and click **Y**.
4. Select Feed Rate through Concentration, click **X**, and then click **OK**.
5. Click the red triangle next to Screening for Percent Reacted and select **Save Script > To Script Window**.
6. Insert **LenthSimN=50000;** as the first line of the script.

```
LenthSimN=50000;
Fit Two Level Screening(
  Y( :Percent Reacted),
  X( :Feed Rate, :Catalyst, :Stir Rate, :Temperature, :Concentration )
);
```

7. Right-click in the script window and select **Run Script**.

Figure 10.10 Fit Two Level with N = 50,000 Simulations for *p*-Value Derivations



The number of simulations is stated in the report window.

Note: If `LenthSimN=0`, the standard *t*-distribution is used and simultaneous *p*-values are not provided. This approach is not recommended.

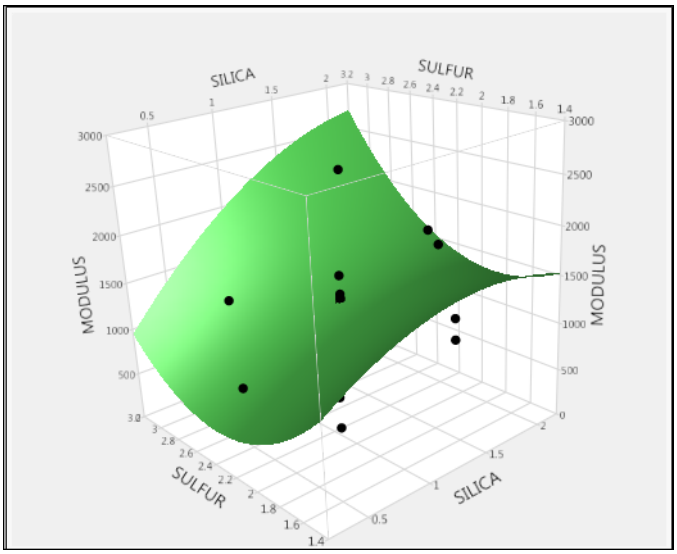
Chapter 11

Response Surface Designs

Response surface designs are useful for modeling quadratic surfaces. A response surface model can identify a point where a minimum or maximum value of the response occurs, if one exists inside the design region. Three distinct values for each factor are necessary to fit a quadratic function, so standard two-level designs are not appropriate for fitting curved surfaces.

Response surface designs are capable of fitting a second-order prediction equation for the response. The quadratic terms in these equations model the curvature in the true response function. If a maximum or minimum exists inside the design region, the point where that value occurs can be estimated.

Figure 11.1 Model for Response Surface Design Results



Contents

Overview of Response Surface Designs 357
Example of a Response Surface Design 359
Construct a Box-Behnken Design 360
Analyze the Experimental Data 361
Explore Optimal Settings 363
Response Surface Design Window 366
Responses 367
Factors 369
Choose a Design 370
Specify Output Options 372
Make Table 373
Response Surface Design Options 374

Overview of Response Surface Designs

The Response Surface Design platform provides the classical central composite and Box-Behnken designs, including blocked versions of these designs. For central composite designs, you can control the placement of axial points and other aspects of the design. Response surface designs are available for continuous factors only and are provided for up to eight factors.

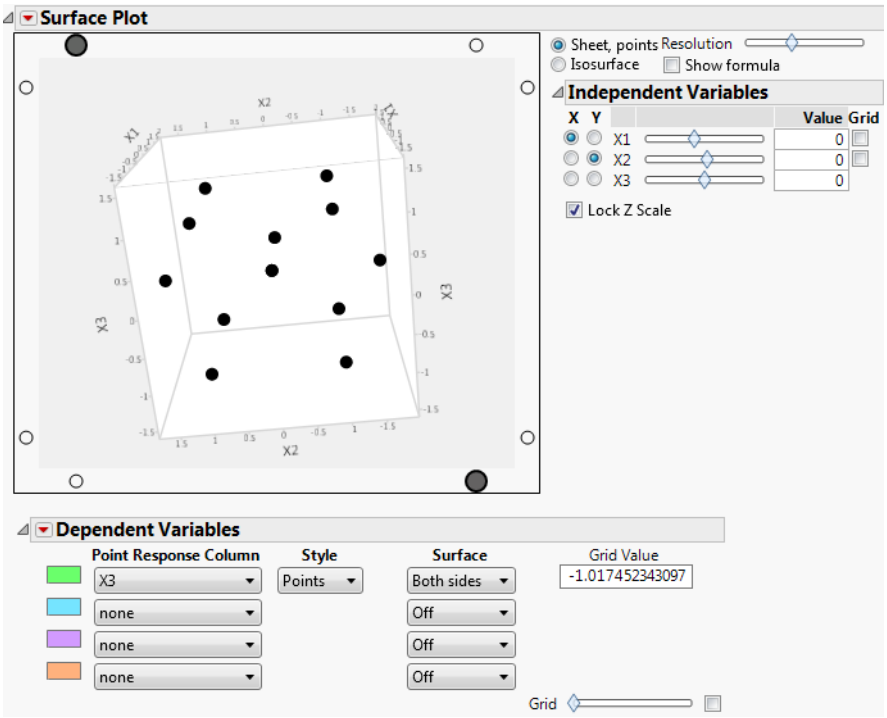
Tip: You can use DOE > Custom Design to construct optimal response surface designs that accommodate your specific experimental situation. Custom Design constructs response surface designs that are much more flexible than classical response surface designs. In particular, you can use the Custom Design platform to create response surface designs that involve categorical factors or more than eight continuous factors. You can also specify the number of runs and restrictions on the design space. For examples, see "Response Surface Experiments".

A *central composite design* (Figure 11.2) combines a two-level fractional factorial design and two other types of points:

- *Center points*, where all the factor values are set to the midrange value.
- *Axial points*, where one factor is set to a high or low value (an *axial* value) and all other factors are set to the midrange value.

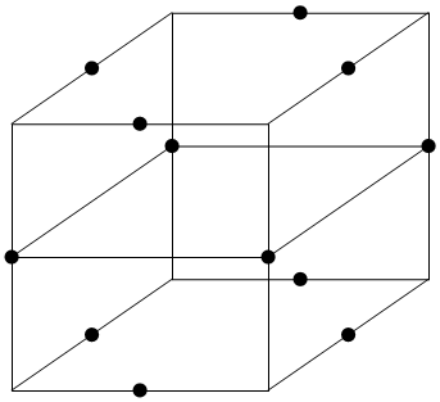
Depending on your selections relative to axial points, a central composite design can have as many as five distinct settings for each factor and the axial points can extend beyond the specified range of the factors.

Figure 11.2 Central Composite Design for Three Factors



A *Box-Behnken design* (Figure 11.3) has only three levels per factor and has no design points at the vertices of the cube defined by the ranges of the factors. This type of design can be useful when you must avoid these points due to engineering considerations. But, the lack of design points at the vertices of the cube means that a Box-Behnken design has higher prediction variance, and so less precision, near the vertices compared to a central composite design.

Figure 11.3 Box-Behnken Design for Three Factors



In JMP, you can construct a response surface design in two ways:

- Using the Response Surface Design platform (for up to eight continuous factors)
- Using the Custom Design platform (and clicking the RSM button in the Model outline)

In both cases, the design table contains a Model script that you can run to fit a model. The Model script applies the Response Surface Effect attribute to each main effect, so that the main effects appear with a **&RS** suffix in the Fit Model window. This attribute ensures that the Fit Least Squares report contains a Response Surface report. For more information about this report, see *Fitting Linear Models*.

Note: The Response Surface outline in the Standard Least Squares report is not shown for response surface designs that contain more than 20 continuous factors.

Example of a Response Surface Design

In this example, you construct a Box-Behnken design for a tire tread experiment. Your objective is to match a target value of 450 for a measure of elongation (Stretch). The stretch varies as a function of the amounts of Silica, Silane, and Sulfur used to manufacture the tire tread compound. You want to experiment over a wide range of factor settings to find the settings that achieve the target.

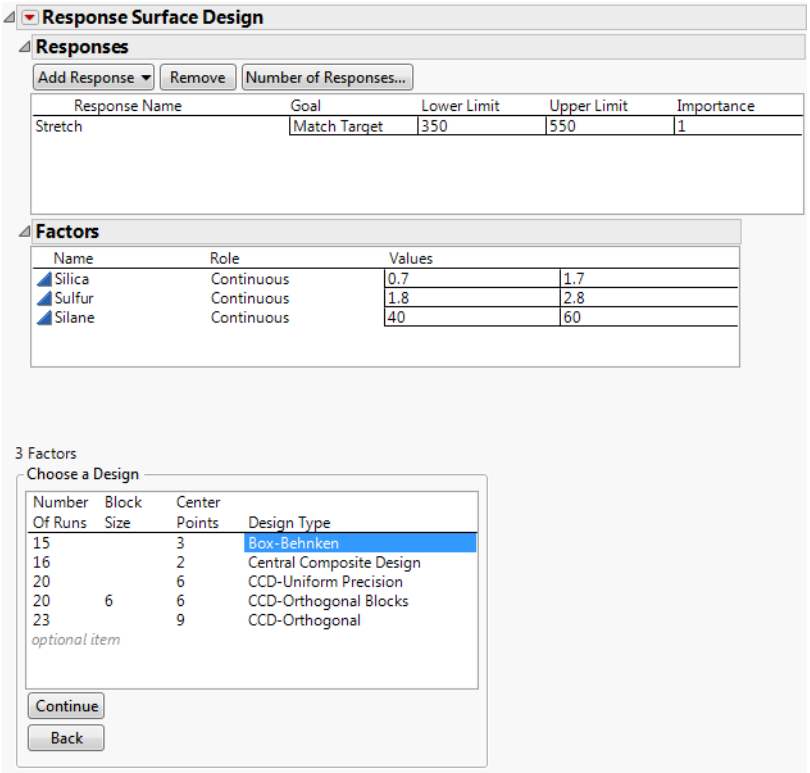
- "Construct a Box-Behnken Design"
- "Analyze the Experimental Data"
- "Explore Optimal Settings"

Construct a Box-Behnken Design

In this example, for convenience, you load the responses and factors from existing tables. When designing a new experiment on your own, enter the responses and factors manually. See "Responses" and "Factors".

1. Select **DOE > Classical > Response Surface Design**.
2. Select **Help > Sample Data Library** and open Design Experiment/Bounce Response.jmp.
3. Click the Response Surface Design red triangle and select **Load Responses**.
4. Select **Help > Sample Data Library** and open Design Experiment/Bounce Factors.jmp.
5. Click the Response Surface Design red triangle and select **Load Factors**.

Figure 11.4 Responses and Factors Outlines for Tire Tread Design



In the Responses outline, notice that the Goal for Stretch is set to Match Target.

In the Choose a Design panel, possible designs appear.

Note: Setting the Random Seed in step 6 reproduces the exact results shown in this example. In constructing a design on your own, this step is not necessary.

6. (Optional) Click the Response Surface Design red triangle and select **Set Random Seed**. Type 12345 and click **OK**.
7. Click **Continue** to retain the Box-Behnken design selection.
8. Click **Make Table**.

Figure 11.5 Box-Behnken Design Table

	Pattern	Silica	Sulfur	Silane	Stretch
1	−0+	0.7	2.3	60	•
2	0+−	1.2	2.8	40	•
3	000	1.2	2.3	50	•
4	000	1.2	2.3	50	•
5	+0−	1.7	2.3	40	•
6	+0+	1.7	2.3	60	•
7	0−+	1.2	1.8	60	•
8	0++	1.2	2.8	60	•
9	−0−	0.7	2.3	40	•
10	−+0	0.7	2.8	50	•
11	000	1.2	2.3	50	•
12	−−0	0.7	1.8	50	•
13	0−−	1.2	1.8	40	•
14	+−0	1.7	1.8	50	•
15	++0	1.7	2.8	50	•

At this point, conduct the experiment and enter the responses into the data table.

Analyze the Experimental Data

1. Select **Help > Sample Data Library** and open Design Experiment/Bounce Data.jmp.

   The file Bounce Data.jmp contains your experiment results.
2. Run the **Model** script.

   Notice that the main effects in the Construct Model Effects list are followed by the **& RS** suffix. This suffix indicates that these are response surface effects, which produce a Response Surface report in the Standard Least Squares report.
3. Click **Run**.

Figure 11.6 Lack of Fit and Effect Tests Reports

Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
Lack Of Fit	3	11.750000	3.91667	0.9792
Pure Error	2	8.000000	4.00000	Prob > F
Total Error	5	19.750000		0.5411
				Max RSq
				0.9999

Box-Cox Transformations

Parameter Estimates

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F
Silica(0.7,1.7)	1	1	12880.125	3260.791	<.0001*
Sulfur(1.8,2.8)	1	1	5778.125	1462.816	<.0001*
Silane(40,60)	1	1	968.000	245.0633	<.0001*
Silica*Sulfur	1	1	52441.000	13276.20	<.0001*
Silica*Silane	1	1	0.250	0.0633	0.8114
Sulfur*Silane	1	1	8556.250	2166.139	<.0001*
Silica*Silica	1	1	2592.923	656.4362	<.0001*
Sulfur*Sulfur	1	1	4653.231	1178.033	<.0001*
Silane*Silane	1	1	0.231	0.0584	0.8186

There is no indication of lack of fit and the Effect Tests report indicates that all but two higher-order terms (Silica*Silane and Silane*Silane) have *p*-values below 0.0001. See *Fitting Linear Models* for more information about interpretation of the tables in Figure 11.6.

4. Click the disclosure icon next to Response Surface to open the report.
5. Click the disclosure icon next to Canonical Curvature.

Figure 11.7 Response Surface Report

Response Surface

Coef

	Silica(0.7,1.7)	Sulfur(1.8,2.8)	Silane(40,60)	Stretch
Silica(0.7,1.7)	26.5	114.5	-0.25	-40.125
Sulfur(1.8,2.8)	.	-35.5	46.25	-26.875
Silane(40,60)	.	.	0.25	11

Solution

Variable	Critical Value
Silica(0.7,1.7)	1.7912411
Sulfur(1.8,2.8)	2.1986422
Silane(40,60)	23.424426

Solution is a SaddlePoint

Critical values outside data range

Predicted Value at Solution 360.38388

Canonical Curvature

Eigenvalues and Eigenvectors

Eigenvalue	62.9095	3.2989	-74.9584
Silica(0.7,1.7)	0.82779	-0.29879	-0.47486
Sulfur(1.8,2.8)	0.52687	0.12315	0.84097
Silane(40,60)	0.19280	0.94634	-0.25937

The Coef table shown as the first part of the report gives a concise summary of the estimated model parameters. The first three columns give the coefficients of the second-order terms. The last column gives the coefficients of the linear terms. To see the prediction expression in its entirely, select **Estimates** > **Show Prediction Expression** from the Response Stretch red triangle.

The Solution report gives the coordinates of the point where the single critical value occurs. In this instance, that point is a saddle point (a point where neither a maximum nor a minimum occurs) and falls outside the range of the design space.

The Canonical Curvature report shows eigenvalues and eigenvectors of the effects. These give information about the nature and direction of the surface's curvature. The large positive eigenvalue of 62.9095 indicates positive curvature and the eigenvector values indicate that the curvature is primarily in the Silica direction. The large negative eigenvalue of -74.9584 indicates negative curvature and the eigenvector values indicate that the curvature is primarily in the Sulfur direction.

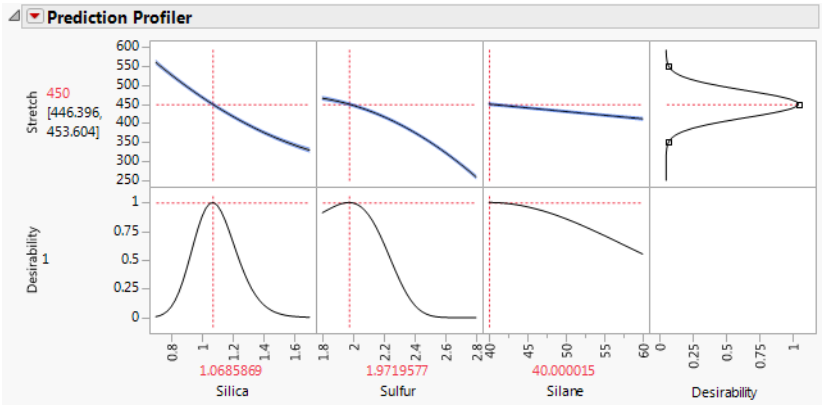
See *Fitting Linear Models* for more information about the response surface analysis tables in Figure 11.7.

Next, use the prediction profiler and the contour profiler to find optimal settings.

Explore Optimal Settings

1. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 11.8 Prediction Profiler for Bounce Data.jmp with Desirability Maximized



Note: Your optimal settings might differ. This is because there are many points for which the predicted Stretch is 450.

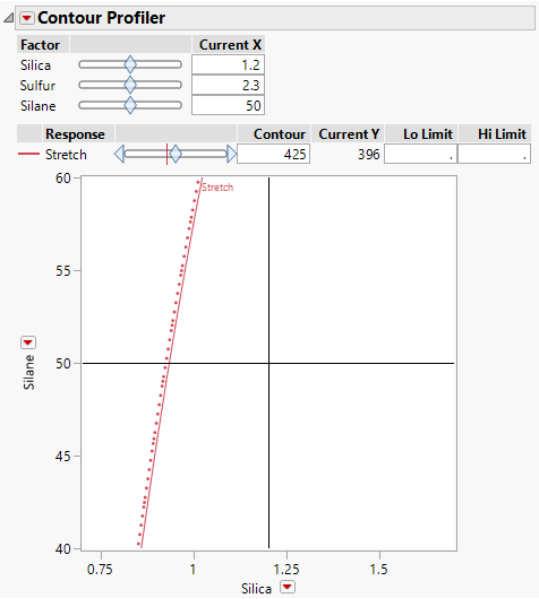
When you specified the response, the goal was set to match a target of 450, with lower and upper limits of 350 and 550. This goal was carried over to the design table and these limits were put in a Response Limits column property for Stretch. A desirability function is constructed from these response limits (top right cell in Figure 11.8). See "Response Limits".

When you maximize the desirability function, JMP identifies one combination of factor level settings (usually out of many possible combinations) that results in a predicted Stretch of 450. Figure 11.8 shows these settings as Silica = 1.069, Sulfur = 1.972, and Silane = 40.000. Next, you use the Contour Profiler to identify other points that maximize the desirability function.

For more information about the Prediction Profiler, see *Profilers*.

2. Click the Response Stretch red triangle and select **Factor Profiling > Contour Profiler.** Suppose that you want to achieve your target while setting Sulfur to the value 2.0. Also, you want to make sure that the settings that you choose for Silane and Silica maintain predicted Stretch within 5 units of 450.
3. On the vertical axis, click the red triangle next to Sulfur and select Silane.

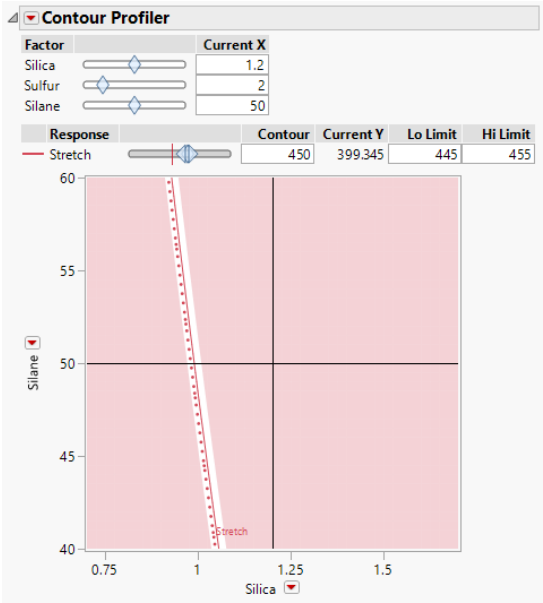
Figure 11.9 Contour Profiler for Bounce Data.jmp



The plot shows the contour of values of Silane and Silica for Stretch at 425 and Sulfur at 2.3.

4. Set the **Current X** for Sulfur to 2.
5. Set the **Contour** for Stretch to 450.
6. Set the **Lo Limit** and **Hi Limit** for Stretch to 445 and 455, respectively. Press Enter.

Figure 11.10 Contour Profiler Showing Optimal Settings for Silica and Silane

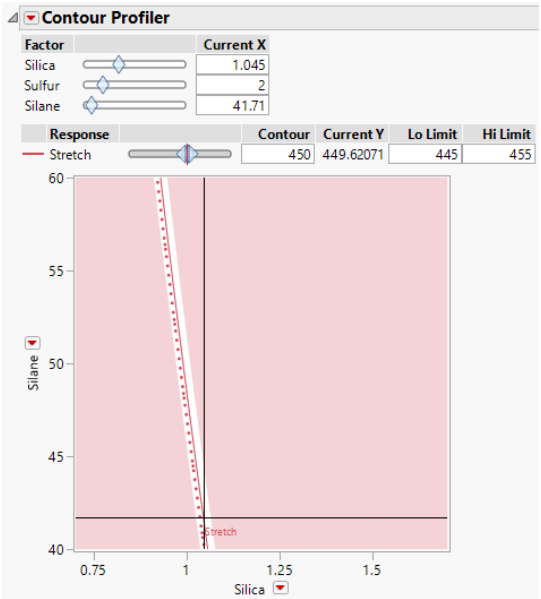


The unshaded band of Silica and Silane values gives predicted Stretch between 445 and 455 when Sulfur is set at 2.0. The values on the solid red curve give predicted Stretch of 450.

7. Drag the crosshairs that appear in the plot to the unshaded band to find settings for Silica and Silane that are best for your process from a practical perspective.

   Suppose that your process is known to be more robust at low levels of Silane than at high levels. Then you might consider the settings in Figure 11.11.

Figure 11.11 Contour Profiler Showing Specific Settings for Silica and Silane



For Sulfur = 2.0, the factor settings identified by the crosshairs are Silica = 1.045 and Silane = 41.71. These settings are shown under Current X. At these settings, the predicted Stretch is 449.62071, shown next to Current Y.

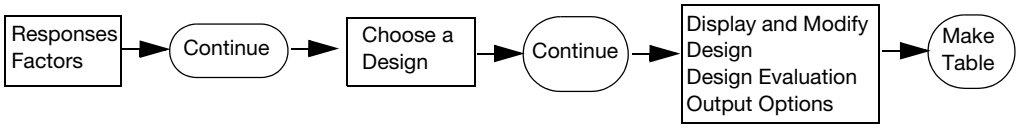
For further information about the Contour Profiler, see *Profilers*.

Response Surface Design Window

The Response Surface Design window walks you through the steps to construct a design for modeling a quadratic surface. You can select a central composite design, a Box-Behnken design, or a blocked version of one of these design types. If you select a central composite design, you can adjust the axial points.

The Response Surface Design window updates as you work through the design steps. The outlines, separated by buttons that update the outlines, follow the flow in Figure 11.12.

Figure 11.12 Response Surface Design Flow

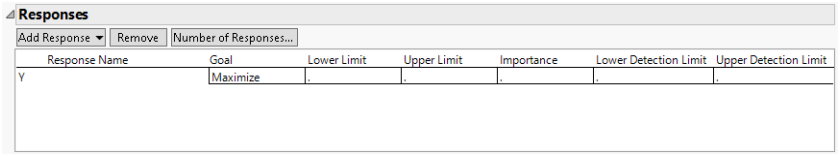


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 11.13 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.
- A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

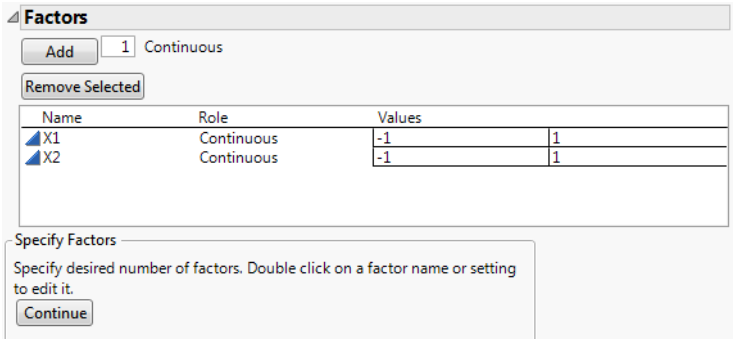
Factors

Factors in a response surface design can only be continuous.

Tip: Use **DOE > Custom Design** to create response surface designs that involve categorical factors.

The initial Factors panel for a response surface design appears with two continuous factors.

Figure 11.14 Factors Outline



The factors outline contains the following buttons.

Add Enters the number of continuous factors specified.

Remove Selected Removes the selected factors.

Tip: When you have completed your Factors panel, select Save Factors from the red triangle menu. This saves the factor names and values in a data table that you can later reload. See "Response Surface Design Options".

The Factors outline contains the following columns:

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role Specifies the Design Role of the factor as Continuous. The Design Role column property for the factor is saved to the data table. This property ensures that the factor type is modeled appropriately.

Values The experimental settings for the factors. To insert Values, click the default values and enter the desired values.

Factor Column Properties

For each factor, various column properties are saved to the data table for the completed design.

Design Role Each factor is assigned the Design Role column property. The Role that you specify in defining the factor determines the value of its Design Role column property. When you select a design with a block, that factor is assigned the Blocking value. The Design Role property reflects how the factor is intended to be used in modeling the experimental data. Design Role values are used in the Augment Design platform.

Factor Changes Each factor is assigned the Factor Changes column property with a setting of Easy. In the Response Surface Design platform, it is assumed that factor levels can be changed for each experimental run. Factor Changes values are used in the Evaluate Design and Augment Design platforms.

Coding If the Design Role is Continuous, the Coding column property for the factor is saved. This property transforms the factor values so that the low and high values correspond to –1 and +1, respectively. The estimates and tests in the Fit Least Squares report are based on the transformed values.

RunsPerBlock Indicates the number of runs in each block. When you select a design with a block and then click Make Table, a factor with the default name Block is added to the Factors list. The RunsPerBlock column property is saved for that factor.

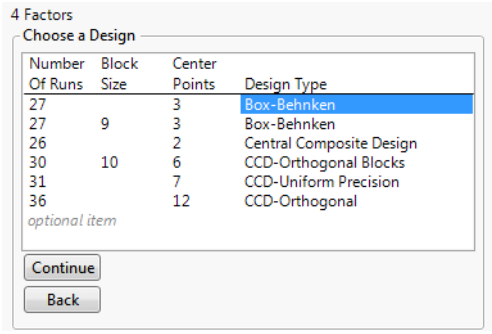
Choose a Design

After you enter your responses and factors and click **Continue**, you select from a list of designs. The designs include two types:

- "Box-Behnken Designs"
- "Central Composite Designs"

Select the design that you want to use and click **Continue**.

Figure 11.15 Choose a Design Panel for Four Factors



Box-Behnken Designs

Box-Behnken designs have only three levels for each factor and have no design points at the vertices of the cube defined by the ranges of the factors. These designs can be useful when it is desirable to avoid extreme settings for engineering considerations. However, these designs result in higher prediction variance near the vertices than do central composite designs.

Central Composite Designs

Central composite designs have center points and axial points. An *axial point* is a point where one factor is set to a high or low value (an *axial* value) and all other factors are set to the midrange, or center, value.

A central composite design can have axial points that fall beyond the faces of the hypercube defined by the specified factor ranges. This means that each factor might require five distinct settings, including two that fall beyond the range of values specified in the Factors outline. However, JMP enables you to place design points on the face.

The following types of central composite designs are available:

Central Composite Design The usual central composite design for the specified number of factors.

CCD-Uniform Precision The number of center points is chosen so that the prediction variance near the center of the design space is very flat.

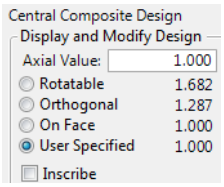
CCD-Orthogonal The number of center points and the axial values are chosen so that the second-order parameter estimates are minimally correlated with the other parameter estimates.

CCD-Orthogonal Blocks The second-order parameter estimates and block effects are minimally correlated with the other parameter estimates.

Axial Value

When you select a central composite design and then click **Continue**, you have the option to provide axial scaling information. In placing axial values, the values shown are used to multiply half of the specified range of a factor. If you specify a value of 1.0 next to Axial Value, then axial points in the resulting design are placed on the faces of the cube defined by the factors. You can set the axial value according to the following options:

Figure 11.16 Axial Value Panel



Rotatable The prediction variance depends only on the scaled distance from the center of the design. The axial points are more extreme than the factor ranges. If this factor range cannot be practically achieved, select **On Face** or specify your own value.

Orthogonal The effects are orthogonal. The axial points are more extreme than the factor ranges. If this factor range cannot be practically achieved, select **On Face** or specify your own value.

On Face Places the axial points at the extremes of the specified factor ranges.

User Specified Places the axial points at a distance specified by the value that you enter in the Axial Value text box.

Inscribe Rescales the design so that the axial points are at the low and high ends of the factor range. The factorial design points are shrunken based on that scaling.

Specify Output Options

You can specify details for the output data table in the Output Options panel. When you finish, click **Make Table** to construct the data table for the design.

Run Order

The **Run Order** options determine the order of the runs in the design table. Choices include the following:

Keep the Same Rows in the design table are in the same order as in the Design and Anticipated Coefficients outline.

Sort Left to Right Columns in the design table are sorted from left to right.

Randomize Rows in the design table are in random order.

Sort Right to Left Columns in the design table are sorted from right to left.

Randomize within Blocks Rows in the design table are in random order within the blocks.

Center Points and Replicates

Number of Center Points The number of center points that appear in the design. A center point is a run where every continuous factor is set at the center of the factor's range. The initial value shown is the number of center points in the design that you selected.

Number of Replicates The number of times to replicate the entire design, including center points. One replicate doubles the number of runs.

Make Table

Click **Make Table** to create a design table that contains the runs for your experiment.

Figure 11.17 Orthogonal Central Composite Design for Bounce Factors and Response

Central Composite D...
Design Central Composite
Model
Evaluate Design
DOE Dialog

Columns (5/0)
Pattern
Silica *
Sulfur *
Silane *
Stretch *

	Pattern	Silica	Sulfur	Silane	Stretch
1	000	1.2	2.3	50	•
2	+++	1.7	2.8	60	•
3	00A	1.2	2.3	66.680317692	•
4	−−+	0.7	1.8	60	•
5	A00	2.0340158846	2.3	50	•
6	−+−	0.7	2.8	40	•
7	000	1.2	2.3	50	•
8	000	1.2	2.3	50	•
9	00a	1.2	2.3	33.319682308	•
10	a00	0.3659841154	2.3	50	•
11	000	1.2	2.3	50	•
12	000	1.2	2.3	50	•

The **Design** note in the Table panel at the upper left gives the design type that generated the table (Central Composite Design). This information can be helpful if you are comparing multiple designs.

Pattern Column

A Pattern column gives a symbolic description of the run in each row in terms of the factor values.

Tip: Pattern can be a useful label variable in plots.

Table 11.1 Pattern Column Description

-	Low value
+	High value
0	Midrange (center) value
a	Low axial value
A	High axial value

Design Table Scripts

The design table includes the following scripts:

Model Runs the **Analyze** > **Fit Model** platform.

Evaluate Design Runs the DOE > Design Diagnostics > Evaluate Design platform.

DOE Dialog Re-creates the Response Surface Design window that you used to generate the design table. The script also contains the random seed used to generate your design.

Response Surface Design Options

The red triangle menu in the Response Surface Design platform contains these options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a

new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Save X Matrix Saves the Moments Matrix and Model Matrix to table scripts in the design data table. These scripts contain the moments and design matrices. See "Save X Matrix".

Caution: For a design with nominal factors, the Model Matrix saved by the Save X Matrix option is *not* the coding matrix used in fitting the linear model. You can obtain the coding matrix used for fitting the model by selecting the option Save Columns > Save Coding Table in the Fit Model report that you obtain when you run the Model script

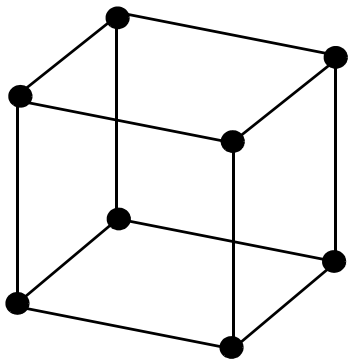
Save Script to Script Window Creates the script for the design that you specified in the Response Surface Design window and saves it in an open script window.

Chapter 12

Full Factorial Designs

A full factorial design defines an experiment where trials are run at all possible combinations of factor settings. A full factorial design allows the estimation of all possible interactions. Full factorial designs are large compared to screening designs, and since high-level interactions are often not active, they can be inefficient. They are typically used when you have a small number of factors and levels and want information about all possible interactions. For example, full factorial designs often form the basis for a measurement systems analysis (MSA).

Figure 12.1 Full Factorial Design for Three Two-Level Factors



Contents

Overview of Full Factorial Design 379
Example of a Full Factorial Design 379
Construct the Design 379
Analyze the Experimental Data 381
Build a Full Factorial Design 385
Responses 385
Factors 387
Select Output Options 389
Make Table 391
Full Factorial Design Options 392

Overview of Full Factorial Design

In a full factorial design, you perform an experimental run at every combination of the factor levels. The sample size is the product of the numbers of levels of the factors. For example, a factorial experiment with a two-level factor, a three-level factor, and a four-level factor has 2 x 3 x 4 = 24 runs.

The Full Factorial Design platform supports both continuous factors and categorical factors with arbitrary numbers of levels. It is assumed that you can run the trials in a completely random fashion.

Full factorial designs are the most conservative of all design types. Unfortunately, because the sample size grows exponentially with the number of factors, full factorial designs are often too expensive to run. Custom designs, definitive screening designs, and screening designs are less conservative but more efficient and cost-effective.

Example of a Full Factorial Design

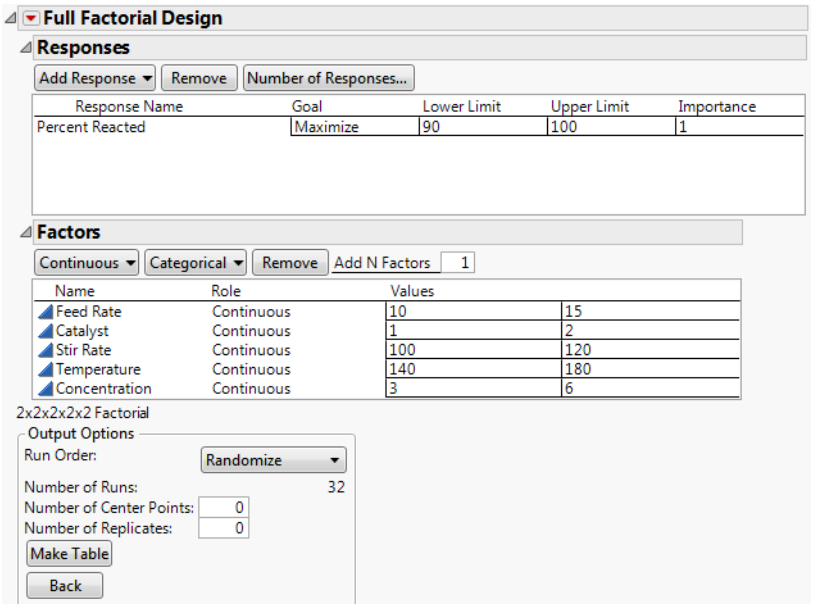
In this example, you construct a full factorial design to study the effects of five two-level factors (Feed Rate, Catalyst, Stir Rate, Temperature, and Concentration) on the yield of a reactor. Because there are five factors, each at two levels, the full factorial design includes at least $2^5 = 32$ runs. For smaller screening designs involving this experimental situation, see "Additional Examples of Screening Designs".

In this example, you load the responses and factors from existing tables. When designing a new experiment on your own, enter the responses and factors manually. See "Responses" and "Factors".

Construct the Design

1. Select **DOE > Classical > Full Factorial Design**.
2. Select **Help > Sample Data Library** and open Design Experiment/Reactor Response.jmp.
3. Click the Full Factorial Design red triangle and select **Load Responses**.
4. Select **Help > Sample Data Library** and open Design Experiment/Reactor Factors.jmp.
5. Click the Full Factorial Design red triangle and select **Load Factors**.
6. Click **Continue**.

Figure 12.2 Full Factorial Example Response and Factors Panels



Note: Setting the Random Seed in step 7 ensures that the runs in your design table appear in the same order as in this example. In constructing a design on your own, this step is not necessary.

7. (Optional) Click the Full Factorial Design red triangle and select **Set Random Seed**. Type 12345 and click **OK**.

   The Run Order in the Output Options panel is set to Randomize. The order of runs in the design table will be random, as determined by the random seed.

   The Number of Runs is set to 32. This is the number of all possible factor level combinations.

8. Click **Make Table**.

   The first column in the design data table shows the factor level combination for each run in terms of + and - signs, indicating high and low factor settings. The table also has an empty Y column named Percent Reacted for entering response values as you conduct the experiment.

Figure 12.3 Full Factorial Design for Reactor Experiment

	Pattern	Feed Rate	Catalyst	Stir Rate	Temperature	Concentration	Percent Reacted
1	++−+−	15	2	100	180	3	•
2	+++++	15	2	120	180	6	•
3	−++−−	10	2	120	140	3	•
4	−+−−+	10	2	100	140	6	•
5	−++++	10	2	120	180	6	•
6	−+−−−	10	2	100	140	3	•
7	++−−+	15	2	100	140	6	•
8	−−+−+	10	1	120	140	6	•
9	−−−−−	10	1	100	140	3	•
10	++++−	15	2	100	180	6	•
11	+−++−	15	1	120	180	3	•
12	−+++−	10	2	120	180	3	•
13	+−−+−	15	1	100	180	3	•
14	−−++−	10	1	120	180	3	•
15	−++−+	10	2	120	140	6	•
16	+++−+	15	2	120	140	6	•
17	+−−++	15	1	100	180	6	•
18	++++−	15	2	120	180	3	•
19	−−−+−	10	1	100	180	3	•
20	+−−−+	15	1	100	140	6	•
21	−++++	10	2	100	180	6	•
22	−−−++	10	1	100	180	6	•
23	++−−−	15	2	100	140	3	•
24	+−+−+	15	1	120	140	6	•
25	−−+++	10	1	120	180	6	•
26	+++−−	15	2	120	140	3	•
27	+−+−−	15	1	120	140	3	•
28	−−+−−	10	1	120	140	3	•
29	−−−−+	10	1	100	140	6	•
30	−+−+−	10	2	100	180	3	•
31	+−−−−	15	1	100	140	3	•
32	+−+++	15	1	120	180	6	•

Analyze the Experimental Data

Next, proceed to analyze the data from the completed experiment. You will use two methods to analyze the results: Screening and Stepwise Regression. Then you will find optimal settings using the Prediction Profiler.

Analysis Using Screening Platform

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 32 Runs.jmp.
2. Run the **Screening** script.

   The Screening report shows a Contrasts report and a Half Normal Plot. The Contrasts report shows estimates for all 31 potential effects, up to the five-way interaction.

Figure 12.4 Contrasts Report for Reactor 32 Runs.jmp

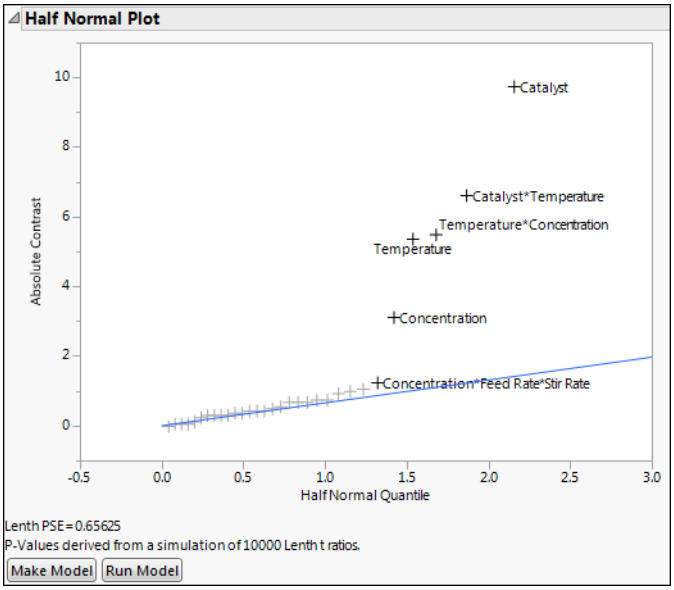
Contrasts

Term	Contrast	Lenth t-Ratio	Individual p-Value	Simultaneous p-Value
Catalyst	9.75000	14.86	<.0001*	<.0001*
Temperature	5.37500	8.19	<.0001*	0.0005*
Concentration	-3.12500	-4.76	0.0011*	0.0152*
Feed Rate	-0.68750	-1.05	0.2883	0.9998
Stir Rate	-0.31250	-0.48	0.6494	1.0000
Catalyst*Temperature	6.62500	10.10	<.0001*	0.0002*
Catalyst*Concentration	1.00000	1.52	0.1335	0.9553
Temperature*Concentration	-5.50000	-8.38	<.0001*	0.0004*
Catalyst*Feed Rate	0.68750	1.05	0.2883	0.9998
Temperature*Feed Rate	-0.43750	-0.67	0.4941	1.0000
Concentration*Feed Rate	0.06250	0.10	0.9266	1.0000
Catalyst*Stir Rate	0.43750	0.67	0.5212	1.0000
Temperature*Stir Rate	1.06250	1.62	0.1139	0.9205
Concentration*Stir Rate	0.43750	0.67	0.4941	1.0000
Feed Rate*Stir Rate	0.37500	0.57	0.5859	1.0000
Catalyst*Temperature*Concentration	-0.12500	-0.19	0.8557	1.0000
Catalyst*Temperature*Feed Rate	0.68750	1.05	0.2883	0.9998
Catalyst*Concentration*Feed Rate	-0.93750	-1.43	0.1587	0.9784
Temperature*Concentration*Feed Rate	0.31250	0.48	0.6494	1.0000
Catalyst*Temperature*Stir Rate	0.56250	0.86	0.3772	1.0000
Catalyst*Concentration*Stir Rate	0.06250	0.10	0.9266	1.0000
Temperature*Concentration*Stir Rate	0.06250	0.10	0.9266	1.0000
Catalyst*Feed Rate*Stir Rate	0.75000	1.14	0.2478	0.9989
Temperature*Feed Rate*Stir Rate	-0.37500	-0.57	0.5859	1.0000
Concentration*Feed Rate*Stir Rate	-1.25000	-1.90	0.0705	0.7592
Catalyst*Temperature*Concentration*Feed Rate	0.31250	0.48	0.6494	1.0000
Catalyst*Temperature*Concentration*Stir Rate	-0.31250	-0.48	0.6494	1.0000
Catalyst*Temperature*Feed Rate*Stir Rate	0.00000	0.00	1.0000	1.0000
Catalyst*Concentration*Feed Rate*Stir Rate	0.75000	1.14	0.2478	0.9989
Temperature*Concentration*Feed Rate*Stir Rate	0.50000	0.76	0.4328	1.0000
Catalyst*Temperature*Concentration*Feed Rate*Stir Rate	-0.25000	-0.38	0.7150	1.0000

Note: Because the *p*-values in the Contrasts report are obtained using a Monte Carlo simulation, you might not obtain the same values as shown in Figure 12.4. See "Lenth's Pseudo-Standard Error".

The six highlighted effects in the Contrasts outline are labeled in the Half Normal Plot.

Figure 12.5 Half Normal Plot for Reactor 32 Runs.jmp



The Half Normal Plot provides strong evidence that at least five of the labeled effects are larger than would be expected if they were the result of random variation. This suggests that these effects are active. The plot does *not* provide a clear indication that the three-way Concentration*Feed Rate*Stir Rate interaction is active.

In the Contrasts outline in Figure 12.4, the Individual p-Value for the three-way Concentration*Feed Rate*Stir Rate interaction is 0.0705 and its Simultaneous p-Value is 0.7592. Because the effect does not stand out on the Half Normal Plot and because its *p*-values are large, you decide not to include this effect in your model.

3. In the Half Normal Plot, drag a rectangle to select all labeled effects except for Concentration*Feed Rate*Stir Rate.
4. Click **Make Model** to open a Fit Model window containing the five effects.
5. Click **Run**.

   The Actual by Predicted plot shows no evidence of lack of fit and the Effect Summary outline shows that all five effects are significant.

Analysis Using Stepwise Regression

1. Return to the Reactor 32 Runs.jmp data table, or reopen it by selecting **Help > Sample Data Library** and opening Design Experiment/Reactor 32 Runs.jmp.
2. Run the **Model** script.

   The Construct Model Effects list contains only up to two-way interactions. You want to consider all interactions.

3. From the Select Columns list, select Feed Rate through Concentration.
4. Click **Macros** > **Full Factorial**.

   All possible effects are added in the Construct Model Effects list.
5. Change the **Personality** to **Stepwise**.
6. Click **Run**.
7. Change the **Stopping Rule** to **Minimum AICc.**

   For designed experiments, AICc is preferred to BIC. This is because BIC is typically a more lenient stopping rule than AICc as it tends to allow inactive effects into the model.
8. Click **Go**.

   The Stepwise procedure selects six effects as potentially active.
9. Click **Run Model**.

   This fits a model using the six effects. The Effect Summary outline indicates that Catalyst*Concentration is not significant at the 0.05 significance level as the *p*-value is 0.0896.
10. Select Catalyst*Concentration in the Effect Summary outline and click **Remove**.

    The five remaining effects are all highly significant. These are the same five effects that you identified using the Screening platform ("Analysis Using Screening Platform").

Optimal Settings Using the Prediction Profiler

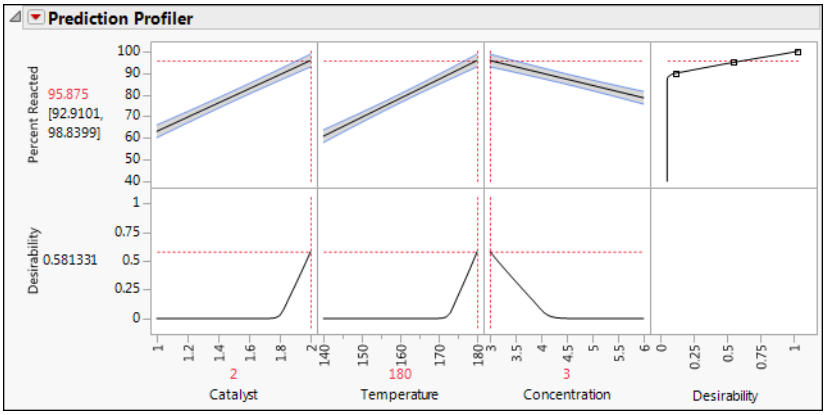
Now, find optimal settings for the three active factors involved in the five significant effects that you retained in your model.

1. In the Reactor 32 Runs.jmp data table, run the **Reduced Model** script.

   The Reduced Model script opens a Fit Model window for the five-effect model that you identified in "Analysis Using Screening Platform" and "Analysis Using Stepwise Regression".
2. Click **Run**.

   The Prediction Profiler report displays Desirability because in the Full Factorial window, you specified a Goal of Maximize when you defined your response. The desirability function shown in the rightmost cell in the top row of the profiler shows that a value of 100 is most desirable and a value of 90 or below is least desirable.
3. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 12.6 Prediction Profiler Showing Settings That Optimize Desirability

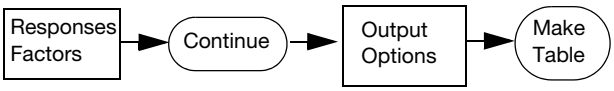


The predicted mean Percent Reacted at the settings that are shown is 95.875, with a confidence interval of 92.91 to 98.84. Note that, for all three factors, the settings that are identified are at the extremes of the ranges used in the experiment. In a future experiment, you should explore the process behavior beyond these settings.

Build a Full Factorial Design

Build a Full Factorial design by selecting **DOE > Classical > Full Factorial Design**. First define your responses and factors. Next, continue to the output options. The design process follows the flow in Figure 12.7. For more information on work flow see "The DOE Workflow: Describe, Specify, Design".

Figure 12.7 Full Factorial Design Flow

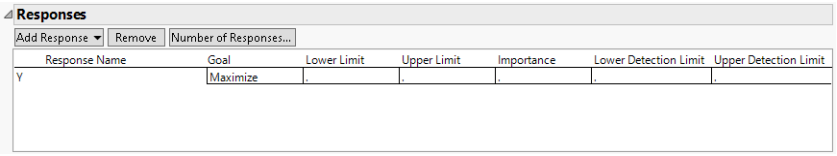


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 12.8 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.

– A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

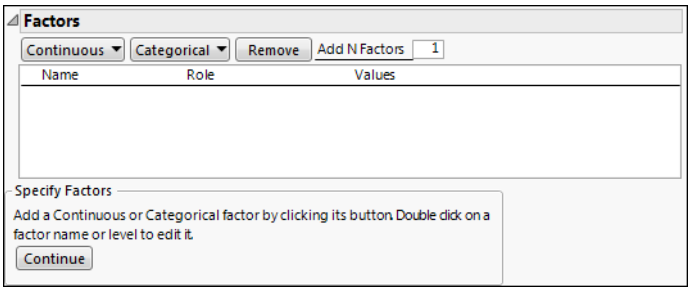
If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

Factors

Factors in a full factorial design can be continuous or categorical.

Tip: When you have completed the Factors outline, consider selecting **Save Factors** from the red triangle menu. This option saves the factor names, roles, changes, and values in a data table that you can later reload in DOE platforms.

Figure 12.9 Factors Outline



Continuous Adds a Continuous factor. The data type in the resulting data table is Numeric. A continuous factor is a factor that you can conceptually set to any value between the lower and upper limits you supply, given the limitations of your process and measurement system.

Categorical Adds a Categorical factor. Click to select or specify the number of levels. The data type in the resulting data table is Character. The value ordering of the levels is the order of the values, as entered from left to right. This ordering is saved in the Value Order column property after the design data table is created.

The default values for a categorical factor are L1, L2,..., Lk, where *k* is the number of levels that you specify. Replace the default values with level names that are relevant for your experiment.

Remove Removes the selected factors.

Add N Factors Adds multiple factors. Enter the number of factors to add, click **Add Factor,** and then select the factor type. Repeat **Add N Factors** to add multiple factors of different types.

Factors Outline

The Factors outline contains the following columns:

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role The Design Role of the factor. The Design Role column property for the factor is saved to the data table. This property ensures that the factor type is modeled appropriately. The

Role of the factor determines other factor properties that are saved to the data table. See "Factor Column Properties".

Values The experimental settings for the factors.

Editing the Factors Outline

In the Factors outline, note the following:

- To edit a factor name, double-click the factor name.
- To edit a value, click the value in the Values column.

Factor Column Properties

For each factor, various column properties are saved to the design table after you create the design by selecting Make Table in the Screening Design window. These properties are also saved automatically to the data table that is created when you select the Save Factors option. You can find more information about these column properties and related examples in "Column Properties".

Coding If the Role is Continuous, the Coding column property for the factor is saved. This property transforms the factor values so that the low and high values correspond to –1 and +1, respectively. See "Coding".

Value Order If the Role is Categorical or if a Block variable is constructed, the Value Order column property for the factor is saved. This property determines the order in which levels of the factor appear. See "Value Order".

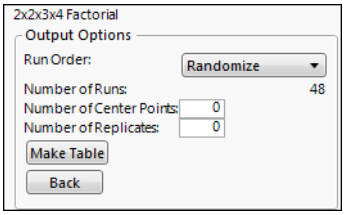
Design Role Each factor is given the Design Role column property. The Role that you specify in defining the factor determines the value of its Design Role column property. The Design Role property reflects how the factor is intended to be used in modeling the experimental data. Design Role values are used in the Augment Design platform. See "Design Role".

Factor Changes Each factor is assigned the Factor Changes column property with the value of Easy. The Factor Changes property reflects how the factor is used in modeling the experimental data. Factor Changes values are used in the Augment Design and Evaluate Design platforms. See "Factor Changes".

Select Output Options

After you enter your responses and factors and click **Continue**, you can make selections for your design table in the Output Options outline. The structure of the full factorial design appears at the top of the outline.

Figure 12.10 Output Options Panel



Run Order

The **Run Order** options determine the order of the runs in the design table.

Keep the Same Rows in the design table are sorted from left to right.

Sort Left to Right Columns in the design table are sorted from left to right.

Randomize Rows in the design table are in random order.

Sort Right to Left Columns in the design table are sorted from right to left.

Center Points and Replicates

Number of Runs Shows the number of runs in the design before you add center points or replicates.

Number of Center Points Specifies how many additional runs to add as center points to the design. A center point is a run where every continuous factor is set at the center of the factor's range.

Suppose that your design includes both continuous and categorical factors. If you request center points in the Output Options panel, the center points are distributed as follows:

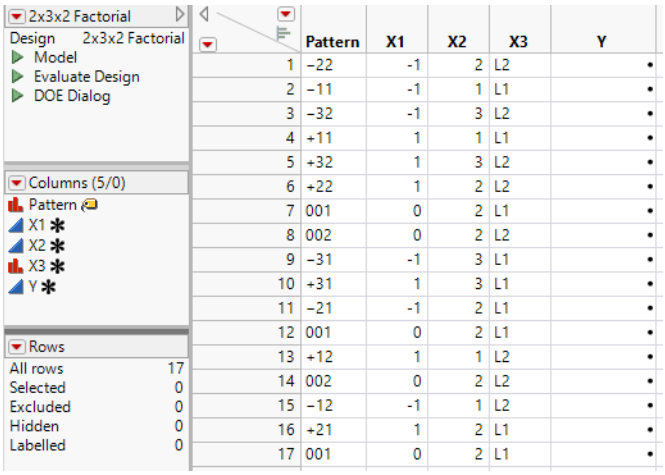
1. The settings for the categorical factors are ordered using the value ordering specified in the Factors outline.
2. One center point is assigned to each combination of the settings of the categorical factors in order, and this is repeated until all center points are assigned.

Number of Replicates The number of times to replicate the entire design, including center points. One replicate doubles the number of runs.

Make Table

Clicking Make Table creates a data table that contains the runs for your experiment. The example in Figure 12.11 shows a full factorial design with five center points for three factors: X1 (a two-level continuous factor), X2 (a three-level continuous factor), and X3 (a two-level categorical factor). The design uses the default values for the factor levels. The center points are in rows 7, 8, 12, 14, and 17. See "Pattern Column".

Figure 12.11 Design Data Table



	Pattern	X1	X2	X3	Y
1	−22	-1	2	L2	•
2	−11	-1	1	L1	•
3	−32	-1	3	L2	•
4	+11	1	1	L1	•
5	+32	1	3	L2	•
6	+22	1	2	L2	•
7	001	0	2	L1	•
8	002	0	2	L2	•
9	−31	-1	3	L1	•
10	+31	1	3	L1	•
11	−21	-1	2	L1	•
12	001	0	2	L1	•
13	+12	1	1	L2	•
14	002	0	2	L2	•
15	−12	-1	1	L2	•
16	+21	1	2	L1	•
17	001	0	2	L1	•

The name of the table, shown in the upper left corner, is the design type that generated it.

Design Table Scripts

The design table includes the following scripts:

Model Runs the Analyze > Fit Model platform.

Evaluate Design Runs the DOE > Design Diagnostics > Evaluate Design platform.

DOE Dialog Re-creates the Full Factorial Design window that you used to generate the design table. The script also contains the random seed used to generate your design.

Run the **Screening** or **Model** scripts to analyze the data.

Pattern Column

The Pattern column contains entries that summarize the run in the given row. You can use Pattern as a label variable in plots.

- For a two-level continuous factor, the low setting is denoted by "–", the high setting by "+", and a center point by "0".

- For a continuous factor with more than two levels:
  - For a non-center point, the factor setting is denoted by an integer that corresponds to the value level for the run.
  - For a center point, the factor setting is denoted by a “0”.
- For a categorical factor, the factor setting is denoted by an integer that corresponds to the value level for the run.

Full Factorial Design Options

The red triangle menu in the Full Factorial Design platform contains these options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor’s column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint’s column contains a column property called ConstraintState that identifies the constraint as a “less than” or a “greater than” constraint. See “ConstraintState”.

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Advanced Options None available.

Save Script to Script Window Creates the script for the design that you specified in the Full Factorial Design window and saves it in an open script window.

Chapter 13

Mixture Designs

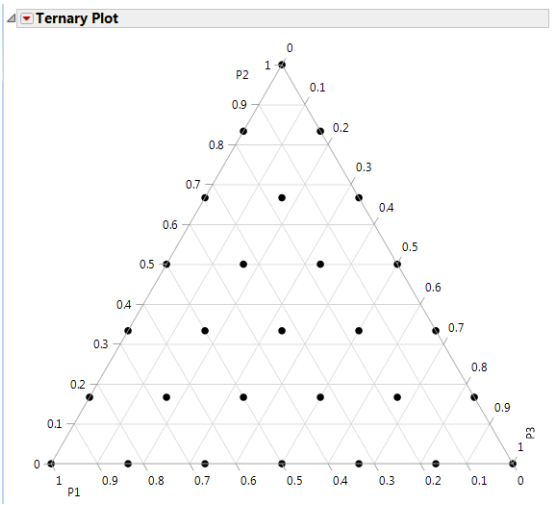
Designs for Formulations

Use the Mixture Design platform to build experiments with factors that are components in a mixture. Choose from several classical mixture design approaches, such as simplex, extreme vertices, and lattice. For the extreme vertices approach, you can supply a set of linear inequality constraints limiting the geometry of the mixture factor space.

In mixture experiments, a factor's value is its proportion of the mixture, which falls between zero and one. Mixture experiments have three or more factors with the sum of the factor proportions equal to one (100%). Mixture experiments differ from other experimental types in that you cannot vary factors independently of one another. When you change the proportion of one factor, the proportion of one or more other factors must also change. This simple fact has a profound effect on every aspect of experimentation with mixtures: the factor space, the design properties, and the interpretation of the results.

Because the mixture components are proportions that sum to one, the feasible region of a mixture design takes the form of a simplex. For example, three factor deigns can be visualized with a triangular 2-D graph and four factor designs can be visualized with a 3-D tetrahedron.

Figure 13.1 Ternary Plot



Contents

Overview of Mixture Designs 397
Mixture Design Window 397
Responses 398
Factors 400
Ternary Plot Overview 401
Examples of Mixture Design Types 402
Optimal Mixture Design 402
Simplex Centroid Design 404
Simplex Lattice Design 406
Extreme Vertices Design 407
ABCD Design 411
Space Filling Design 412
Fitting Mixture Designs 414
Whole Model Tests and Analysis of Variance Reports 415
Understanding Response Surface Reports 415
A Chemical Mixture Example 416
Create the Design 416
Analyze the Mixture Model 418
The Prediction Profiler 419
The Mixture Profiler 420

Overview of Mixture Designs

You can choose from the following types of mixture designs:

Optimal Generates an optimal mixture design using the custom designer. See "Optimal Mixture Design".

Simplex Centroid (Available for unconstrained factors or factors with lower bound constraints.) Generates a design of pure mixtures and blends of factors up to a specified degree. In a pure mixture, one component is at 100% and all other components are at 0%. See "Simplex Centroid Design".

Simplex Lattice (Available for unconstrained factors or factors with lower bound constraints.) Generates a design of pure mixtures and blends in a space filling triangular grid of runs. You can specify the number of grid levels. See "Simplex Lattice Design".

Extreme Vertices (Available when you have linear constraints or restricted upper or lower bounds on one or more of your factors.) Generates a design based on the vertices and combinations of the vertices of a constrained factor space. See "Extreme Vertices Design".

ABCD Design Generates a screening design for mixtures devised by Snee (1975). See "ABCD Design".

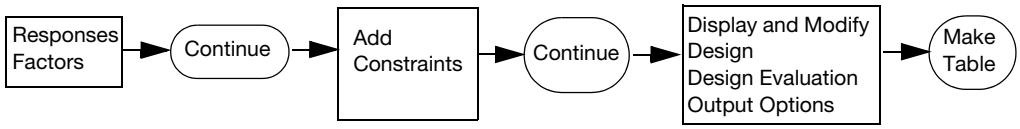
Space Filling Generates a space filling design that accommodates linear constraints. See "Space Filling Design".

Note: The optimal, extreme vertices, and space filling designs can incorporate linear constraints and restrictions on components.

Mixture Design Window

The mixture design window updates as you work through the design steps. See "The DOE Workflow: Describe, Specify, Design". The workflow and outlines vary depending on design type.

Figure 13.2 Mixture Design Workflow

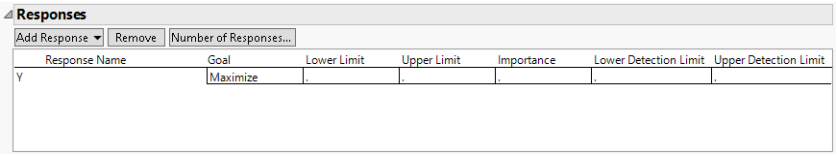


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 13.3 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.
- A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

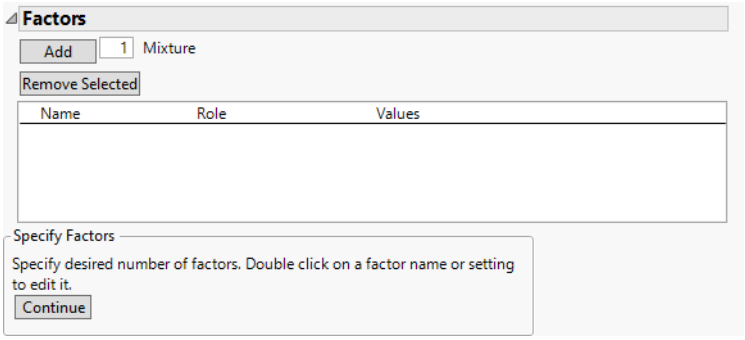
If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

Factors

Add factors in the Factors outline.

Tip: When you have completed the Factors outline, consider selecting **Save Factors** from the red triangle menu. This saves the factor names, roles, changes, and values in a data table that you can later reload.

Figure 13.4 Factors Outline



Add Enter the number of factors to add and click Add.

Remove Selected Removes the selected factors.

Factors List

The Factors list contains the following columns:

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role Specifies the Design Role of the factor. The Design Role column property for the factor is saved to the data table. This property ensures that the factor type is modeled appropriately.

Values The experimental settings for the factors. To insert Values, click the default values and enter the desired values.

Editing the Factors List

In the Factors list, do the following:

- To edit a factor name, double-click the factor name.
- To edit a value, click the value in the Values column.

Linear Constraints

Click the **Linear Constraint** button to enter one or more linear inequality constraints. A template appears for a linear expression involving all the continuous factors in your design. Enter coefficient values for the factors and select the direction of the inequality to reflect your linear constraint. Specify the constraining value in the box to the right of the inequality. To add more constraints, click **Linear Constraint** again.

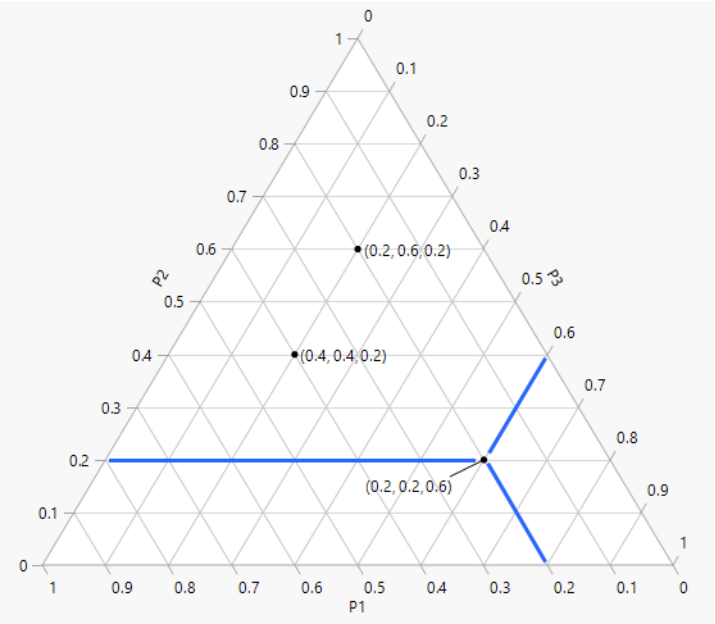
Note: When you save a script for a design that involves a linear constraint, the script expresses the linear constraint as a *less than or equal to* inequality (≤).

Ternary Plot Overview

A ternary plot is a two-dimensional representation of three mixture components that sum to a constant. The plot is an equilateral triangle with an edge for each component. When unconstrained, each vertex of the triangle corresponds to a pure blend where one component is 1 (100%) and all other components are 0. When components are constrained, the feasible mixtures are represented by a portion of the ternary plot. Shading is used to exclude infeasible portions of the plot.

Figure 13.5 displays three components P1, P2, and P3. The three components are unconstrained with the range of each proportion ranging from 0 to 1 (100%). Three points are labeled with their coordinates (P1, P2, P3). One point (0.2, 0.2, 0.6) includes blue lines along the axis grid lines for each mixture component to guide you in reading the axes.

Figure 13.5 Ternary Plot Example



Examples of Mixture Design Types

- "Optimal Mixture Design"
- "Simplex Centroid Design"
- "Simplex Lattice Design"
- "Extreme Vertices Design"
- "ABCD Design"
- "Space Filling Design"

Optimal Mixture Design

Optimal mixture designs can be generated using either the Mixture Design Platform or the Custom Design platform.

Here you use the Mixture Platform to create an example optimal mixture design with three factors:

1. Select **DOE > Classical> Mixture Design**.
2. Use three factors for this example. No changes need to be made to the Factors section.

3. Click **Continue**.
4. Click **Optimal** on the Choose Mixture Design Type panel.
5. (Optional) To match the output of this example, click the Mixture Design red triangle and select Set Random Seed, and then enter 1409.
6. (Optional) To match the output of this example, click the Mixture Design red triangle and select Number of Starts, and then enter 2.
7. In the **Define Factor Constraints** panel, click **Specify Linear Constraints**, and then click **Add**.
8. Enter a 1 in the X1 and X2 boxes and enter 0.8 in the constraint box.

   This constrains X1+X2 to less than 80% of the mixture.
9. In the **Model** panel, click **Interactions** > **2nd**.

   This adds interaction effects to the model.
10. In the **Design Generation** panel, enter 2 for the **Number of Center Points**.
11. Click **Make Design**.

Figure 13.6 12-Run Optimal Design Runs

Design

Run	X1	X2	X3
1	0.8	0	0.2
2	0	0	1
3	0	0.8	0.2
4	0	0	1
5	0	0.4	0.6
6	0.4	0	0.6
7	0	0.8	0.2
8	0.4	0	0.6
9	0.4	0.4	0.2
10	0.8	0	0.2
11	0.264183	0.269687	0.46613
12	0.264183	0.269687	0.46613

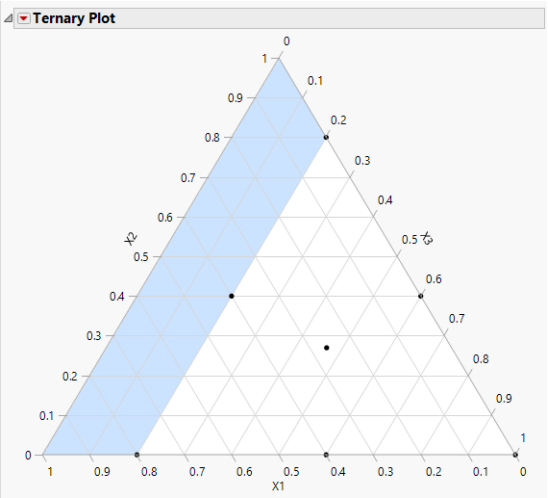
12. Click **Make Table**.

Note: In practice, you would round the mixtures amounts to the appropriate level of precision.

Visualize the Design

1. From the design table, select **Graph >Ternary Plot**.
2. Select X1, X2, and X3 and click **X, Plotting**, and then click **OK**.

Figure 13.7 Ternary Plot for Optimal Design



The ternary plot shows the 7 unique mixtures that make up the design. The shading indicates the constrained region. For more information about ternary plots, see "Ternary Plot Overview".

Simplex Centroid Design

A simplex centroid design of degree *k* with *n* factors consists of mixtures with the following characteristics:

- all one factor mixtures or pure blends
- all blends of two factors at equal levels
- additional blends, up to *k* factors at a time blended at *k* equal levels

A center point run with equal amounts of all the ingredients is included when *k* is equal to the number of factors.

Note: This design can be used with lower bounds on one or more factors.

Creating the Design

To create an example simplex centroid design:

1. Select **DOE > Classical > Mixture Design**.
2. Use three factors for this example. No changes need to be made to the Factors section.
3. Click **Continue**.

4. Click **Simplex Centroid** to generate the design

> **Note:** The default setting of $k = 2$ was used. There is a text box that enables you to adjust the value of k.

Figure 13.8 Three Factor Simplex Centroid Design of Degree 2.

Design

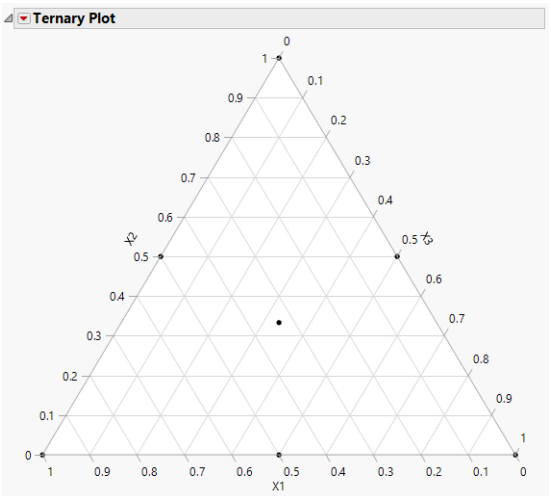
Run	X1	X2	X3
1	1.00000	0.00000	0.00000
2	0.00000	1.00000	0.00000
3	0.00000	0.00000	1.00000
4	0.50000	0.50000	0.00000
5	0.50000	0.00000	0.50000
6	0.00000	0.50000	0.50000
7	0.33333	0.33333	0.33333

5. Click **Make Table**.

Visualize the Design

6. From design table, select **Graph>Ternary Plot**.
7. Select X1, X2, and X3 and click **X, Plotting**, and then click **OK**.

Figure 13.9 Ternary Plot for Simplex Centroid Design



For more information about ternary plots, see "Ternary Plot Overview".

Simplex Lattice Design

The simplex lattice design is a space filling design that creates a triangular grid of runs. The design is the set of blends where the factors' values are i/m, where i varies from 0 to m and the sum of the factors is 1. For a three-level design, a factor can be set at 0, 0.33, 0.66, or 1.

Note: This design can be used with lower bounds on one or more factors.

To create an example simplex lattice design:

1. Select **DOE > Classical > Mixture Design.**
2. Use three factors for this example. No changes need to be made to the Factors section.
3. Click **Continue**.
4. Click **Simplex Lattice** to generate the design.

   Note: The default setting of 5 levels was used. There is a text box that enables you to adjust the number of levels

Figure 13.10 Three Factor Five-Level Simplex Lattice Design

Design

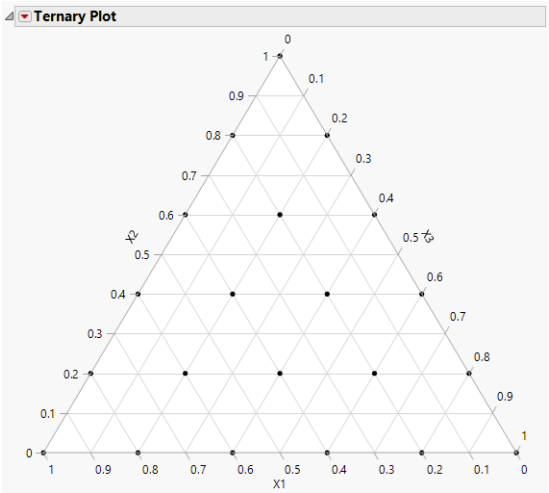
Run	X1	X2	X3
1	0.00000	0.00000	1.00000
2	0.00000	0.20000	0.80000
3	0.00000	0.40000	0.60000
4	0.00000	0.60000	0.40000
5	0.00000	0.80000	0.20000
6	0.00000	1.00000	0.00000
7	0.20000	0.00000	0.80000
8	0.20000	0.20000	0.60000
9	0.20000	0.40000	0.40000
10	0.20000	0.60000	0.20000
11	0.20000	0.80000	0.00000
12	0.40000	0.00000	0.60000
13	0.40000	0.20000	0.40000
14	0.40000	0.40000	0.20000
15	0.40000	0.60000	0.00000
16	0.60000	0.00000	0.40000
17	0.60000	0.20000	0.20000
18	0.60000	0.40000	0.00000
19	0.80000	0.00000	0.20000
20	0.80000	0.20000	0.00000
21	1.00000	0.00000	0.00000

5. Click **Make Table**.

Visualize the Design

6. From the design table, select **Graph >Ternary Plot**.
7. Select X1, X2, and X3 and click **X, Plotting**, and then click **OK**.

Figure 13.11 Ternary Plot for Simplex Lattice Design



For more information about ternary plots, see "Ternary Plot Overview".

As the number of levels increases, the number of design points increases, decreasing the space between design points. In contrast to the simplex centroid design, the simplex lattice design does not necessarily include the centroid.

Extreme Vertices Design

An extreme vertices design requires restricted ranges on one or more of the factors, as defined by limits in the Factors panel or by linear constraints. An extreme vertices design consists of mixtures at the vertices of the factor space and at the averages of vertices up to a specified degree. The centroid point for two neighboring vertices joined by a line is a second-degree centroid. The centroid point for vertices sharing a plane is a third-degree centroid.

An Extreme Vertices Example with Range Constraints

To create an example extreme vertices design:

1. Select **DOE > Classical > Mixture Design**.
2. Under Factors, enter 2 for the number of additional factors to add and click **Add**.
3. Set the ranges for the five factors as shown in Figure 13.12 and click **Continue.**

Figure 13.12 Ranges for Five-factors

Name	Role	Values	
X1	Mixture	0.05	0.25
X2	Mixture	0.1	0.3
X3	Mixture	0.1	0.3
X4	Mixture	0.1	0.4
X5	Mixture	0.05	0.25

4. Enter 4 in the **Degree** text box and click **Extreme Vertices**.

Figure 13.13 Display and Modify Panel for Extreme Vertices Example

The default number of runs for this five-factor, degree 4 extreme vertices design is 116. This is the number of extreme vertices and averages up to degree 4 for these five factors given the constraints on their ranges. JMP uses the default set of mixtures as a candidate set to generate a smaller D Optimal design.

5. (Optional) To match the output of this example, click the Mixture Design red triangle and select Set Random Seed, and then enter 1409.
6. Enter 10 in the Choose desired sample size box and click **Find Subset** to generate the design.

Note: The **Find Subset** option uses the row exchange method (not coordinate exchange) to find the optimal subset of rows.

Figure 13.14 Ten Run D-optimal Extreme Vertices Design

Design

Run	X1	X2	X3	X4	X5
1	0.05000	0.30000	0.20000	0.40000	0.05000
2	0.05000	0.10000	0.30000	0.40000	0.15000
3	0.25000	0.30000	0.10000	0.10000	0.25000
4	0.05000	0.30000	0.30000	0.10000	0.25000
5	0.25000	0.10000	0.30000	0.10000	0.25000
6	0.25000	0.30000	0.30000	0.10000	0.05000
7	0.25000	0.10000	0.30000	0.30000	0.05000
8	0.05000	0.20000	0.10000	0.40000	0.25000
9	0.25000	0.30000	0.10000	0.30000	0.05000
10	0.25000	0.10000	0.10000	0.30000	0.25000

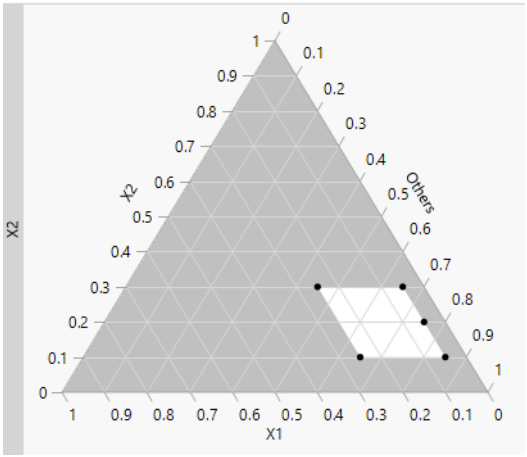
7. Click **Make Table**.

Visualize the Design

8. From the design table, select **Graph >Ternary Plot**.

9. Select X1, X2, X3, X4, and X5 and click **X, Plotting**, and then click **OK**.

Figure 13.15 Partial Output of Ternary Plot for Five-Factor Design



When you have more than three mixture components, the ternary plot shows a series of plots. Each plot has two component axes and the third axis is the sum of all other components. The plots are shaded if you have a constrained region. The unshaded portion represents the feasible region. For more information about ternary plots, see "Ternary Plot Overview".

An Extreme Vertices Example with Linear Constraints

Consider the classic example presented by Snee (1979) and Piepel (1988). This example has three factors, X1, X2, and X3, with factor bounds and three linear constraints.

To create an extreme vertices design for this example:

1. Select **DOE > Classical > Mixture Design**.
2. Enter the values from Figure 13.16 for X1, X2, and X3 and click **Continue**.

Figure 13.16 Values and Linear Constraints for the Snee and Piepel Example

Factors

Name	Role	Values	
X1	Mixture	0.1	0.5
X2	Mixture	0.1	0.7
X3	Mixture	0	0.7

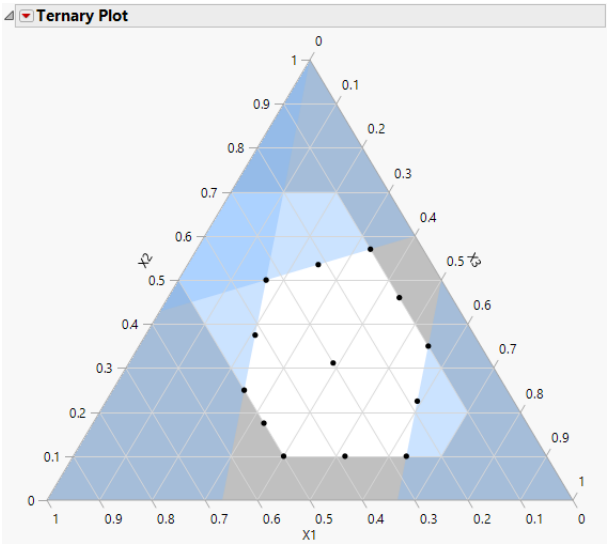
-0.85	X1 +	-0.9	X2 +	-1	X3	≤	-0.9
0.85	X1 +	0.9	X2 +	1	X3	≤	0.95
-0.7	X1 +	0	X2 +	-1	X3	≤	-0.4

3. Click **Linear Constraint** three times. Enter the constraints as shown in Figure 13.16.
4. Click the **Extreme Vertices** button.
5. Click **Make Table**.

Visualize the Design

6. From the design table, select **Graph >Ternary Plot**.
7. Select X1, X2, and X3 and click **X, Plotting**, and then click **OK**.

Figure 13.17 Ternary Plot Showing Piepel Example with Constraints



The ternary plot shows the feasible region as defined by the factor limits and the linear constraints. The design points are at the six vertices of the feasible region, at the six edge mid-points, and the overall centroid. For more information about ternary plots, see "Ternary Plot Overview".

Statistical Details for Extreme Vertices Method

If there are linear constraints, JMP uses the CONSIM algorithm developed by R.E. Wheeler, described in Snee (1979) and presented by Piepel (1988) as CONVRT. The method is also described in Cornell (1990, Appendix 10a). The method combines constraints and checks to see whether vertices violate them. If so, it drops the vertices and calculates new ones. The method named CONAEV for doing centroid points is by Piepel (1988).

If there are no linear constraints (only range constraints), the extreme vertices design is constructed using the XVERT method developed by Snee and Marquardt (1974) and Snee (1975). After the vertices are found, a simplex centroid method generates combinations of vertices up to a specified order.

The XVERT method first creates a full 2^{nf-1} design using the given low and high values of the $nf-1$ factors with smallest range. Then, it computes the value of the one factor left out based on the restriction that the factors' values must sum to one. It keeps points that are not in factor's range. If not, it increments or decrements the value to bring it within range, and decrements or increments each of the other factors in turn by the same amount. This method keeps the points that still satisfy the initial restrictions.

The above algorithm creates the vertices of the feasible region in the simplex defined by the factor constraints. However, Snee (1975) has shown that it can also be useful to have the centroids of the edges and faces of the feasible region. A generalized n-dimensional face of the feasible region is defined by $nf-n$ of the boundaries and the centroid of a face defined to be the average of the vertices lying on it. The algorithm generates all possible combinations of the boundary conditions and then averages over the vertices generated on the first step.

ABCD Design

The ABCD Design is a screening design for mixtures. See Snee (1975).

To create an example ABCD design:

1. Select **DOE > Classical > Mixture Design**.
2. Under Factors, enter 3 for the number of additional factors to add and click **Add**.
3. Click **Continue**.
4. Click **ABCD Design**.

Figure 13.18 14 of the 28 Run 6 Factor ABCD Design

Design

Run	X1	X2	X3	X4	X5	X6
1	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000
3	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000
4	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000
5	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000
6	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000
7	0.50000	0.50000	0.00000	0.00000	0.00000	0.00000
8	0.50000	0.00000	0.50000	0.00000	0.00000	0.00000
9	0.50000	0.00000	0.00000	0.50000	0.00000	0.00000
10	0.50000	0.00000	0.00000	0.00000	0.50000	0.00000
11	0.50000	0.00000	0.00000	0.00000	0.00000	0.50000
12	0.00000	0.50000	0.50000	0.00000	0.00000	0.00000
13	0.00000	0.50000	0.00000	0.50000	0.00000	0.00000
14	0.00000	0.50000	0.00000	0.00000	0.50000	0.00000

5. Click **Make Table**.

Space Filling Design

The Space Filling mixture designs spread design points throughout the design region. It accommodates linear constraints. The design is generated in a fashion similar to the Fast Flexible Filling design method found under DOE > Special Purpose > Space Filling Design ("Fast Flexible Filling Designs").

Two Mixture Design red triangle options relate specifically to Space Filling Designs:

FFF Optimality Criterion For the Fast Flexible Filling mixture design type, enables you to select between the MaxPro criterion (the default) and the Centroid criterion. See "FFF Optimality Criterion".

Advanced Options > Set Average Cluster Size For the Fast Flexible Filling mixture design type, enables you to specify the average number of randomly generated points used to define each cluster or, equivalently, each design point. See "Set Average Cluster Size".

Space Filling Example

To create an example space filling design:

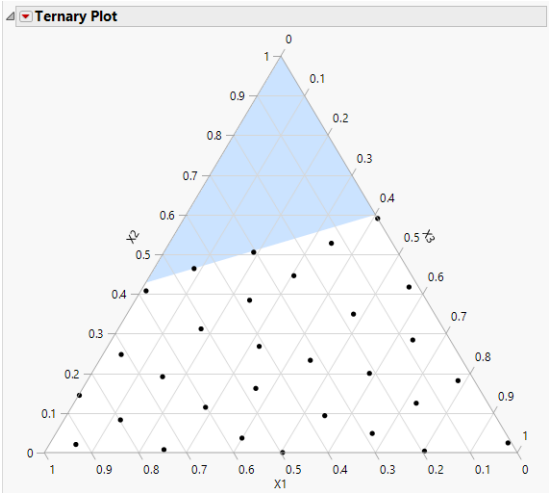
1. Select **DOE > Classical > Mixture Design**.
2. (Optional) To match the output of this example click the Mixture Design red triangle and select Set Random Seed and enter 1409.
3. Use three factors for this example. No changes need to be made to the Factors section.
4. Click **Continue**.
5. Click **Linear Constraints**
6. Enter 0.7 in the X1 box, 1 in the X3 box, change the direction of the inequality to ≥ and enter 0.4 in the constraint box.

   This constrains $0.7X1+X3$ to at least 40% of the mixture.
7. Enter 30 in the **Runs** box.
8. Click **Space Filling**.
9. Click **Make Table**.

Visualize the Design

10. From the design table, select **Graph >Ternary Plot**.
11. Select X1, X2, and X3 and click **X, Plotting**, and then click **OK**.

Figure 13.19 Space Filling Design with One Linear Constraint



This design is constructed using the Centroid FFF Optimality Criterion. Note that the points fall in the constrained design region and are fairly well spread throughout this region. For more information about ternary plots, see "Ternary Plot Overview".

Statistical Details for Mixture Space Filling Designs

FFF Optimality Criterion

The algorithms for Fast Flexible Filling designs begin by generating a large number of random points within the specified design region. These points are then clustered using a Fast Ward algorithm into a number of clusters that equals the Number of Runs that you specified.

The final design points can be obtained by using the default MaxPro (*maximum projection*) optimality criterion or by selecting the Centroid criterion. You can find these options under FFF Optimality Criterion in the report's red triangle menu.

MaxPro For p factors and n equal to the specified Number of Runs, the MaxPro criterion strives to find points in the clusters that minimize the following criterion:

$$C_{MaxPro} = \sum_{i}^{n-1} \sum_{j=i+1}^{n} \left[1 \Big/ \prod_{k=1}^{p} (x_{ik} - x_{jk})^2 \right]$$

The MaxPro criterion maximizes the product of the distances between potential design points in a way that involves all factors. This supports the goal of providing good space-filling properties on projections of factors. See Joseph et al. (2015). The Max Pro option is the default.

Centroid This method places a design point at the centroid of each cluster. It has the property that the average distance from an arbitrary point in the design space to its closest neighboring design point is smaller than for other designs.

Note: You can set a preference to always use a given optimality criterion. Select **File > Preferences > Platforms > DOE**. Select FFF Optimality Criterion and select your preferred criterion.

Set Average Cluster Size

The **Set Average Cluster Size** option is found under Advanced Options in the Mixture Design red triangle menu. This option enables you to specify the average number of uniformly generated points used to define each cluster or, equivalently, each design point.

By default, if the number of Runs for the Space Filling design type is 200 or smaller, a total of 10,000 random uniformly generated points are used as the basis for the clustering algorithm. When the number of Runs exceeds 200, the default value is 50. Increasing this value can be particularly useful in designs with a large number of factors.

Note: Depending on the number of factors and the specified value for Runs, you might want to increase the average number of initial points per design point by selecting **Advanced Options > Set Average Cluster Size.**

Linear Constraints

The design region can be restricted by selecting the **Linear Constraint** option in the Linear Constraints outline.

When you specify linear constraints, the random points that form the basis for the clustering algorithm are randomly distributed within the constrained design region. The clustering algorithm uses these points.

Fitting Mixture Designs

When fitting a model for mixture designs, one must take into account that the factors sum to a constant. A traditional full linear model is not fully estimable.

An appropriate response surface model for mixture responses is the Scheffé polynomial (Scheffé 1958). See the discussion of Cox Mixtures and the Scheffé cubic macro in *Fitting Linear Models*. The Scheffé polynomial model does the following:

- suppresses the intercept

- includes all the linear main-effect terms
- excludes all the square terms (such as X1*X1)
- includes all the cross terms (such as X1*X2)

In this model, the parameters are easy to interpret (Cornell 1990). The coefficients on the linear terms are the fitted response at the extreme points where the mixture consists of a single factor. The coefficients on the cross terms indicate the curvature across each edge of the factor space.

For a mixture model with third-degree polynomial terms, the Scheffé cubic model can be used. The Scheffé cubic model includes terms of the form X1*X2*(X1-X2).

- To fit a Scheffé polynomial model use the Mixture Response Surface macro in the Fit Model platform.
- To fit the Scheffé cubic model use the **Scheffe Cubic** macro in the Fit Model platform.
- If you chose to fit a different mixture model the Fit Model platform includes options for a No Intercept model and Mixture Effects as an effect attribute.

Tip: The custom design model outline has an option for adding the Scheffé cubic model terms. The generated design table will include a script for the Scheffé cubic model.

Whole Model Tests and Analysis of Variance Reports

In a whole-model Analysis of Variance table, JMP traditionally tests that all the parameters are zero except for the intercept. In a mixture model without an intercept, JMP looks for a hidden intercept, in the sense that a linear combination of effects is a constant. If it finds a hidden intercept, it does the whole model test with respect to the intercept model rather than a zero-intercept model. This test is equivalent to testing that all the parameters are zero except the linear parameters, and testing that they are equal.

The hidden-intercept property also causes the R^2 to be reported with respect to the intercept model rather than reported as missing.

Understanding Response Surface Reports

When there are effects marked as response surface effects "&RS," JMP creates additional reports that analyze the fitted response surface. These reports were originally designed for full response surfaces, not mixture models. However, JMP might encounter a no-intercept model and find a hidden intercept with linear response surface terms, but no square terms. Then it *folds* its calculations, collapsing on the last response surface term to calculate critical values for the optimum. This can be done for any combination that yields a constant and involves the last response surface term.

A Chemical Mixture Example

Three plasticizers (p1, p2, and p3) comprise 79.5% of the vinyl used for automobile seat covers (Cornell, 1990). Within this 79.5%, the individual plasticizers are restricted by the following constraints: $0.409 \leq x1 \leq 0.849$, $0 \leq x2 \leq 0.252$, and $0.151 \leq x3 \leq 0.274$. Cornell uses a 14-run design to fit a quadratic mixture model to the collected responses. The design is a 3-degree extreme vertices design with replicate measurements taken at the four vertices and the overall centroid.

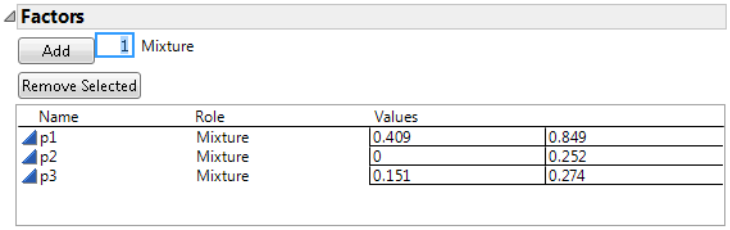
- "Create the Design"
- "Analyze the Mixture Model"
- "The Prediction Profiler"
- "The Mixture Profiler"

Create the Design

To create Cornell's mixture design in JMP

1. Select **Help > Sample Data Library** and open Plastifactors.jmp.
2. Select **DOE > Classical > Mixture Design**.
3. Click the Mixture Design red triangle and select **Load Factors**.

Figure 13.20 Factors and Factor Constraints for the Plasticizer Experiment



4. Click **Continue**.
5. Enter 3 in the **Degree** box.
6. Click **Extreme Vertices**.
7. Click **Make Table**. JMP uses the 9 factor settings to generate a JMP table.

Figure 13.21 Extreme Vertices Mixture Design

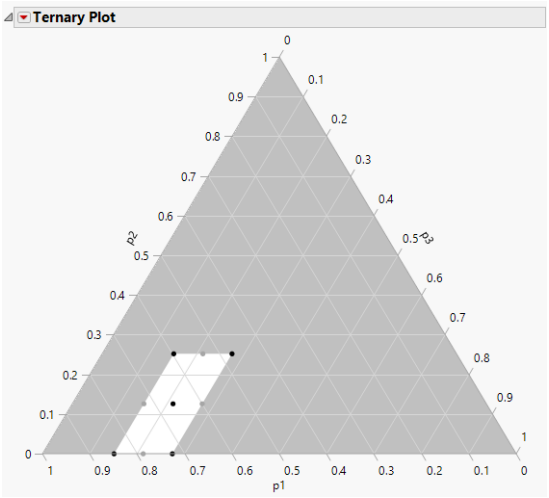
	p1	p2	p3	Y
1	0.597	0.252	0.151	•
2	0.726	0	0.274	•
3	0.6615	0.126	0.2125	•
4	0.6	0.126	0.274	•
5	0.7875	0	0.2125	•
6	0.849	0	0.151	•
7	0.5355	0.252	0.2125	•
8	0.474	0.252	0.274	•
9	0.723	0.126	0.151	•

Note: Your table might differ due to the random seed used to generate the design.

Next, you add an extra five design runs by duplicating the vertex points and overall centroid, to generate the 14 run design.

8. To identify the vertex points and the centroid use a ternary plot. From the design table, select **Graph >Ternary Plot**.
9. Select p1, p2, and p3 and click **X, Plotting**, and then click **OK**.

Figure 13.22 Ternary Plot for Design



10. Highlight the vertices and centroid points.

Figure 13.23 Identify Vertices and Center Point

	p1	p2	p3	Y
1	0.597	0.252	0.151	•
2	0.726	0	0.274	•
3	0.6615	0.126	0.2125	•
4	0.6	0.126	0.274	•
5	0.7875	0	0.2125	•
6	0.849	0	0.151	•
7	0.5355	0.252	0.2125	•
8	0.474	0.252	0.274	•
9	0.723	0.126	0.151	•

11. Select **Edit > Copy**, to copy the selected rows to the clipboard.
12. Click in the first cell of row 10 and select **Edit > Paste** to add the duplicate rows to the table.

Note: A similar design can be obtained using the custom designer with the quadratic model, 2 center points, 4 replicate designs, and 14 runs.

The Plasticizer design with the results (*Y* values) that Cornell obtained are available in the sample data file Plasticizer.jmp.

Analyze the Mixture Model

To analyze the Plasticizer experiment use the sample data table Plasticizer.jmp.

1. Select **Help > Sample Data Library** and open Plasticizer.jmp.
2. Click the green triangle next to the **Model** script in the upper left corner of the data table.

   A completed Fit Model launch window appears.

Note: The model script is saved to the design table when you create a design.

3. Click **Run** to see the response surface analysis.
4. Plasticizer.jmp contains a column called Pred Formula Y. This column was added after the analysis by selecting **Save Columns > Prediction Formula** from the Response Y red triangle.
5. To see the prediction formula, click the plus symbol in the column list:

   0−50.1465**p1* − 282.198**p2* − 911.648**p3* + *p2***p1**317.363 + *p3***p1**1464.330 + *p3***p2**1846.218

Note: These results correct the coefficients reported in Cornell (1990).

The Response Surface Solution report shows that a maximum predicted value of 19.570299 occurs at point (0.63505, 0.15568, 0.20927).

Figure 13.24 Mixture Response Surface Analysis

Response Surface

Coef

	(p1-0.474)/0.375	p2/0.375	(p3-0.151)/0.375	Y
(p1-0.474)/0.375	0	44.62923	205.92138	7.4923077
p2/0.375	.	0	259.62437	-1.491318
(p3-0.151)/0.375	.	.	0	-138.2039

Solution

Variable	Critical Value
(p1-0.474)/0.375	0.635052
p2/0.375	0.1556806
(p3-0.151)/0.375	0.2092675

Solution is a Maximum
Assuming the following mixture sum: p1+p2+p3=1
Predicted Value at Solution 19.570299

The Prediction Profiler

The report contains a prediction profiler.

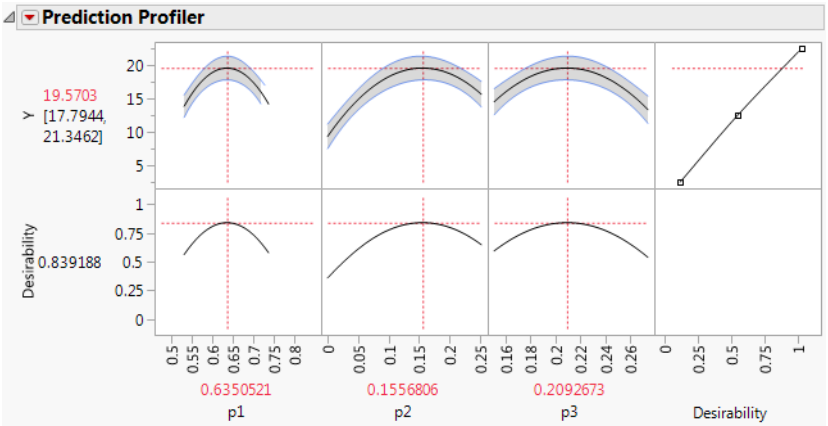
1. From the prediction profiler you can see how the crossed effects show as curvature in the prediction traces. As you change values of one component (drag the red lines) the values of the other two components move in the opposite direction to maintain their ratio and the overall mixture constraint (components must sum to 100%).

   Note: The axes of prediction profiler traces range from the defined upper and lower bounds of the factors, p1, p2, and p3. When you explore the effect of a component and the limit of a second component is reached, it cannot move further and only the third variable changes to maintain the mixture constraint.

2. To the visible profile curves to bounds that take into account the levels of all three components, click the Prediction Profiler red triangle and select **Profile at Boundary > Stop at Boundaries**.
3. To optimize the mixture components click the Prediction Profiler red triangle and select **Optimization and Desirability > Desirability Functions**.
4. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability** to obtain the optimal factor settings for maximum Y.

The profiler displays optimal settings (rounded) of 0.6350 for p1, 0.1557 for p2, and 0.2093 for p3, which give an estimated response of 19.5703.

Figure 13.25 Maximum Desirability in Profiler for Mixture Analysis Example



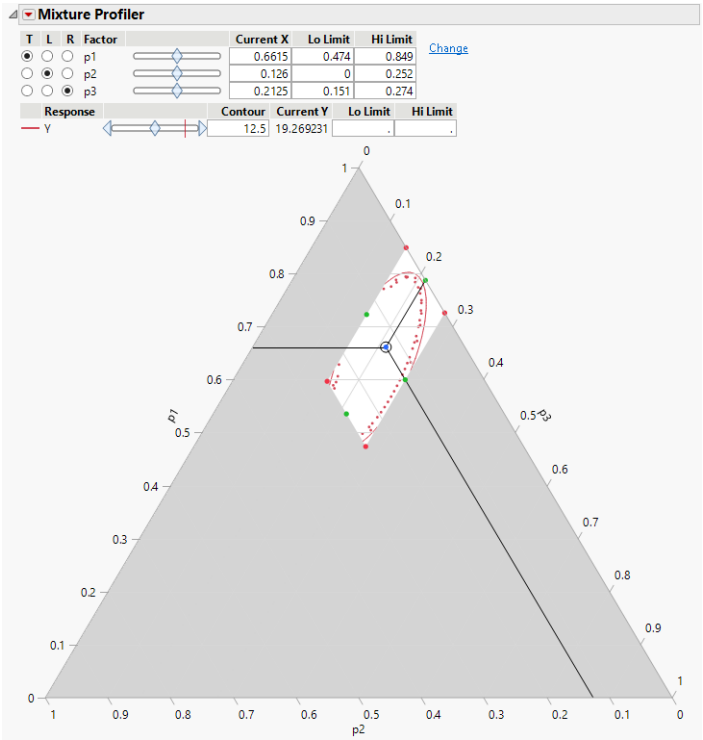
The Mixture Profiler

The Fit Model report also has a **Mixture Profiler** that is useful for visualizing and optimizing response surfaces from mixture experiments. Many of the features are the same as those of the Contour Profiler. However, some are unique to the Mixture Profiler:

- A ternary plot is used instead of a Cartesian plot, which enables you to view three mixture factors at a time.
- If you have more than three factors, radio buttons let you choose which factors to plot.
- If the factors have constraints, you can enter their low and high limits in the Lo Limit and Hi Limit columns. This setting shades non-feasible regions in the profiler.

Click the Response Y red triangle ands elect **Factor Profiling** > **Mixture Profiler** to see the mixture profiler for the plasticizer data.

Figure 13.26 Mixture Profiler for Plasticizer Example



Chapter 14

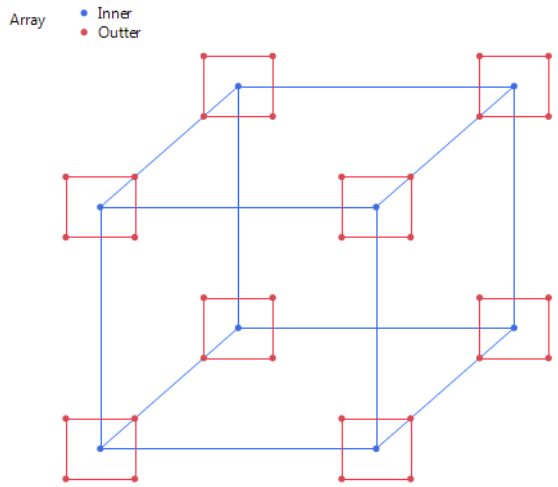
Taguchi Designs

Designs with Noise Factors

A Taguchi design uses crossed array to explore factors in the presence of noise. A Taguchi design has two parts: the inner array is a design for control factors, and the outer array is a design for the noise factors. The full design is the cross product of the two arrays.

Note that noise factors in production that are not easy or cost effective to control must be controlled during an experiment. Alternatives to Taguchi designs include combined arrays and mixed resolution designs (Borror and Montgomery, 2000). For an example of a design with noise factors using the custom designer see "Experiments for Robust Process and Product Design".

Figure 14.1 Taguchi Design with Three Control Factors and Two Noise Factors



Contents

Overview of Taguchi Designs 425
Example of a Taguchi Design 425
Build a Taguchi Design. 429
Responses 430
Factors. 430
Choose Taguchi Inner and Outer Array Designs 431
Display Coded Design 431
Make the Design Table 432
Taguchi Design Options. 432

Overview of Taguchi Designs

A Taguchi Design uses two types of factors: control factors and noise factors.

- An *inner* design is constructed for the control factors.
- An *outer* design is constructed for the noise factors.

The experiment is performed on all combinations of the inner and outer design runs. The noise factors have to be controlled during the experiment. The mean and standard deviation for each setting in the inner array across the noise settings are combined into a signal-to-noise measure for analysis. Table 14.1 lists the signal-to-noise performance statistics.

Table 14.1 Recommended Performance Statistics

Goal	S/N Ratio Formula
nominal is best	$\frac{S}{N} = 10\log\left(\frac{\bar{Y}^2}{s^2}\right)$
larger-is-better (maximize)	$\frac{S}{N} = -10\log\left(\frac{1}{n}\sum_i \frac{1}{Y_i^2}\right)$
smaller-is-better (minimize)	$\frac{S}{N} = -10\log\left(\frac{1}{n}\sum_i Y_i^2\right)$

Example of a Taguchi Design

Use a Taguchi design to study four control factors evaluated across three noise factors.This example is an experiment described by Byrne and Taguchi (1986). The objective of the experiment is to find settings of control factors to maximize the adhesiveness (pull-off force) of nylon tubing.

The design has four signal factors:

Interfer Tubing and connector interference. Signal factor with 3 levels.

Wall Wall thickness of the connector. Signal factor with 3 levels.

Depth Insertion depth of the tubing into the connector. Signal factor with 3 levels.

Adhesive Percent of adhesive. Signal factor with 3 levels.

The design has three noise factors:

Time Conditioning time. Noise factor with 2 levels.

Temperature Temperature. Noise factor with 2 levels.

Humidity Relative humidity. Noise factor with 2 levels.

Create the Design

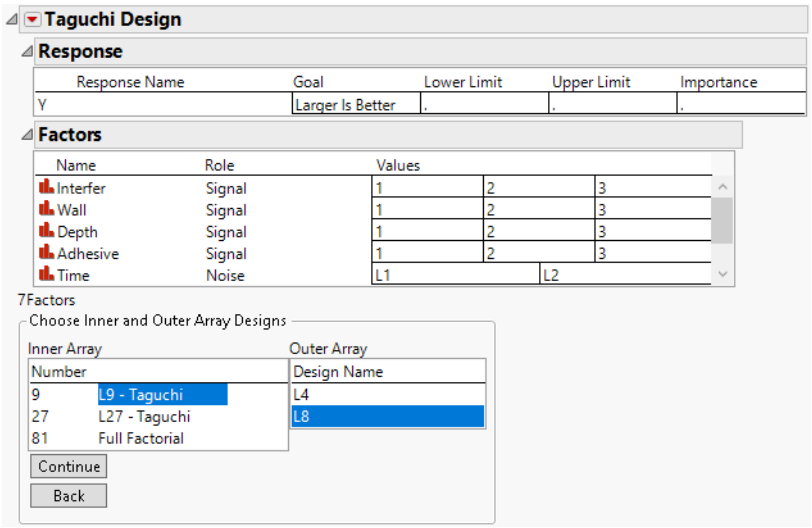
1. Select **DOE > Classical > Taguchi Arrays**.
2. Select **Help > Sample Data Library** and open Design Experiment/Byrne Taguchi Factors.jmp.
3. In the Taguchi Arrays window, click the Taguchi Design red triangle and select **Load Factors**.

   The Factors panel shows the four three-level control (signal) factors and three noise factors.

   Note: The Goal selected for the response determines the SN Ratio formula that is added to your design data table.

4. Ensure that **L9-Taguchi** is selected for the inner array.
5. Click **L8** for the outer array design.

Figure 14.2 Completed Taguchi Design Window



6. Click **Continue**.
7. Click **Make Table** to create the design table shown in Figure 14.3.

   The inner array has nine runs for the four signal factors, each at three levels. The settings for the signal factors are provided in the first four columns of the table.

   The outer design is a full factorial design for the three two-level noise factors. The outer array is provided in eight columns of the data table. The column names are the pattern of the outer array runs. For example, the column named "---" is for the results collected when all of the noise levels are at their low levels. Each of the nine trials are conducted at each of the eight combination of noise factors for a total of 72 experimental trials.

Figure 14.3 Taguchi Design before Data Entry

	Interfer	Wall	Depth	Adhesive	Pattern	− − −	− − +	− + −	− + +	+ − −	+ − +	+ + −	+ + +	Mean	SN Ratio
1	1	1	1	1	----	•	•	•	•	•	•	•	•	•	•
2	1	2	2	2	-000	•	•	•	•	•	•	•	•	•	•
3	1	3	3	3	-+++	•	•	•	•	•	•	•	•	•	•
4	2	1	2	3	0-0+	•	•	•	•	•	•	•	•	•	•
5	2	2	3	1	00+-	•	•	•	•	•	•	•	•	•	•
6	2	3	1	2	0+-0	•	•	•	•	•	•	•	•	•	•
7	3	1	3	2	+-+0	•	•	•	•	•	•	•	•	•	•
8	3	2	1	3	+0-+	•	•	•	•	•	•	•	•	•	•
9	3	3	2	1	++0-	•	•	•	•	•	•	•	•	•	•

The design table allows for entry of the 72 experimental results.

8. Select **Help > Sample Data Library** and open Design Experiment/Byrne Taguchi Data.jmp.

Figure 14.4 Complete Taguchi Design Table (Byrne Taguchi Data.jmp)

	Interfer	Wall	Depth	Adhesive	Pattern	Y---	Y--+	Y-+-	Y-++	Y+--	Y+-+	Y++-	Y+++	Mean Y	SN Ratio Y
1	1	1	1	1	----	15.6	9.5	16.9	19.9	19.6	19.6	20	19.1	17.525	24.02534
2	1	2	2	2	-000	15	16.2	19.4	19.6	19.7	19.8	24.2	21.9	19.475	25.52164
3	1	3	3	3	-+++	16.3	16.7	19.1	15.6	22.6	18.2	23.3	20.4	19.025	25.33476
4	2	1	2	3	0-0+	18.3	17.4	18.9	18.6	21	18.9	23.2	24.7	20.125	25.90425
5	2	2	3	1	00+-	19.7	18.6	19.4	25.1	25.6	21.4	27.5	25.3	22.825	26.90753
6	2	3	1	2	0+-0	16.2	16.3	20	19.8	14.7	19.6	22.5	24.7	19.225	25.32574
7	3	1	3	2	+-+0	16.4	19.1	18.4	23.6	16.8	18.6	24.3	21.6	19.85	25.71081
8	3	2	1	3	+0-+	14.2	15.6	15.1	16.8	17.8	19.6	23.2	24.4	18.3375	24.83231
9	3	3	2	1	++0-	16.1	19.9	19.3	17.3	23.1	22.7	22.6	28.6	21.2	26.15198

The SN Ratio Y column is the performance statistic (larger is better) to maximize the response. It is computed as –10 times the common logarithm of the average of the squared reciprocals of the responses:

$$-10Log10\left[Mean\left[\frac{1}{(Y---)^2}, \frac{1}{(Y--+)^2}, \frac{1}{(Y-+-)^2}, \frac{1}{(Y-++)^2}, \frac{1}{(Y+--)^2}, \frac{1}{(Y+-+)^2}, \frac{1}{(Y++-)^2}, \frac{1}{(Y+++)^2}\right]\right]$$

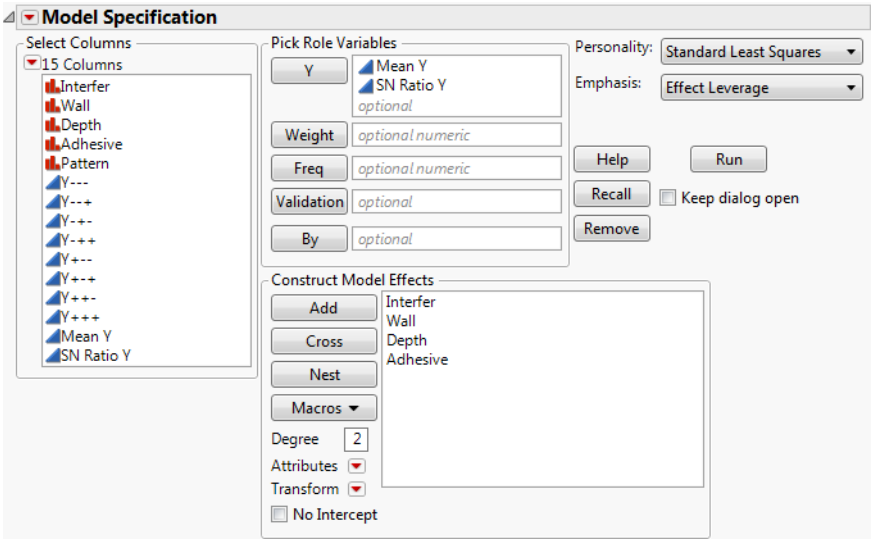
This expression is large when all of the individual y response values are large. That is, you are trying to find the signal settings that result in the largest response across the noise settings.

Analyze the Data

The data are now ready to analyze. The goal of the analysis is to find factor settings that maximize both the mean and the signal-to-noise ratio.

1. In the Byrne Taguchi Data.jmp data table, click the green arrow to run the Model script.

Figure 14.5 Fit Model Launch Window for Taguchi Data

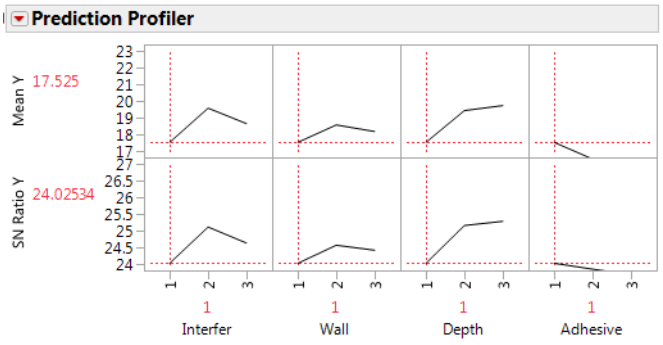


The script launches the Fit Model window. The model includes the main effects of the four signal factors to model the mean (Mean Y) and signal-to-noise ratio (SN Ratio Y) responses.

2. Click **Run**.

The Prediction Profiler at the bottom of the report is a quick way to find settings that give the highest signal-to-noise ratio for this experiment and the mean value at that maximum.

Figure 14.6 The Prediction Profiler

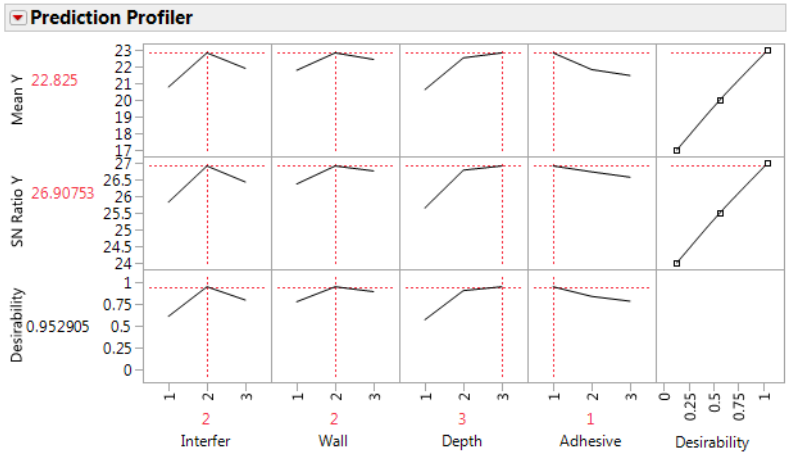


3. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Desirability Functions**.

   This adds a row of traces and a column of function settings to the profiler, as shown in Figure 14.7. The default desirability functions are set to larger-is-better, which is what you want in this experiment. See *Profilers* for more information about desirability functions in the prediction profiler.

4. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 14.7 Best Factor Settings for Byrne Taguchi Data

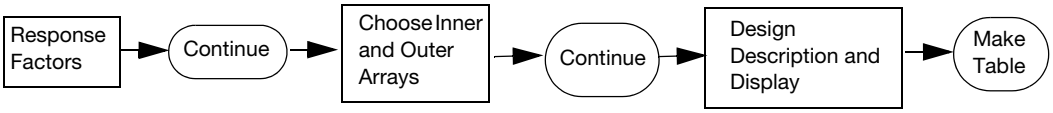


In this example, the optimal settings for Interfer and Wall are 2, Depth is 3, and Adhesive is 1. These settings result in a predicted Mean value of 22.8 and an SN Ratio of 26.9.

Build a Taguchi Design

Build a Taguchi Design by selecting **DOE > Classical > Taguchi Arrays.** First define your responses and factors. Next, continue to the design options, generation, and evaluation. The design process follows the flow in Figure 14.8. See "The DOE Workflow: Describe, Specify, Design".

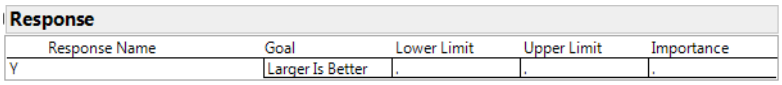
Figure 14.8 Taguchi Design Flow



Responses

Use the Response outline to specify a response for a Taguchi design.

Figure 14.9 Response Outline



The Response outline contains the following columns:

Response Name The name of the response (the default is Y). To change this name, double-click it and enter a different name.

Goal, Lower Limit, Upper Limit The Goal determines the form of the signal-to-noise function (Table 14.1). Select one of the following Goals: Larger Is Better, Nominal is Best, Smaller is Better, or None. When you create your design, JMP saves a formula for the SN Ratio to the data table that reflects your selected goal. To change a goal, click it and enter a different goal. Use the Lower and Upper Limits if you have limits on your response.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for the responses. In a Taguchi Design there is only one response, the Importance is set to 1 by default. No value is needed for the Importance.

Factors

Use the Factors outline to specify factors for a Taguchi Design.

Signal Specify two or more 2- or 3-level signal factors. Signal factors are system control inputs. These are factors that you can control in production.

Noise Specify one or more noise factors. Noise factors are variables that are difficult or expensive to control in production. However, you must be able to control noise factors during the experiment. Click Noise to add a noise factor.

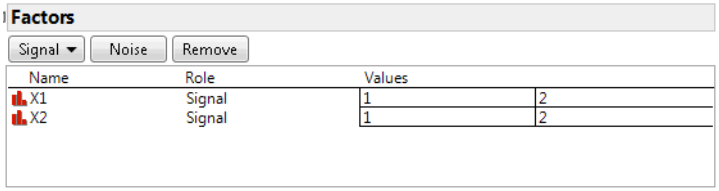
Remove Removes the selected factors.

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter a different name.

Role Specifies the Design Role (Signal or Noise) of the factor. This is set when you add the factor.

Values The experimental settings for the factors. To insert Values, click the default values and enter the desired values.

Figure 14.10 Factors Outline



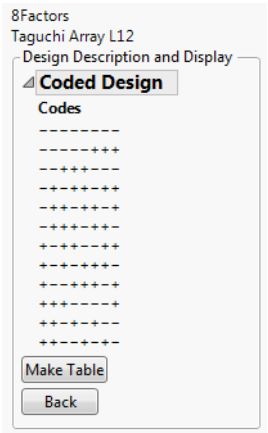
Choose Taguchi Inner and Outer Array Designs

The Choose Inner and Outer Array Design options appear after you click Continue in the Specify Factors panel. Based on your signal factors, you get one or more choices for orthogonal designs for the inner array. The outer array choices for orthogonal "L" designs depend on the number and type of factors in your design.

Display Coded Design

After you select a design type for the inner and outer arrays, the Coded Design is shown when your inner array has two-level factors.

Figure 14.11 Coding for Eight Factor L12 Design



The Coded Design shows the pattern of high and low values for the factors in each run.

Make the Design Table

When you click **Make Table**, a Taguchi Design table similar to the one in Figure 14.12 appears. In the data table, each row represents an inner array run. In the values for the Pattern variable, plus signs designate high levels, and minus signs represent low levels. The table also contains a column for each factor and a column for each pattern that appears in the Pattern column. The columns for the patterns contain the responses to the experiment. If you selected Simulate Responses from the Taguchi Design red triangle menu, the pattern columns are filled with simulated response values.

Figure 14.12 Taguchi Design Table for an Eight Factor L12 Design

	X1	X2	X3	X4	X5	X6	X7	X8	Pattern	run 1	run 2	Mean	SN Ratio
1	1	1	1	1	1	1	1	1	--------	•	•	•	•
2	1	1	1	1	1	2	2	2	-------+++	•	•	•	•
3	1	1	2	2	2	1	1	1	--++++---	•	•	•	•
4	1	2	1	2	2	1	2	2	-++++++	•	•	•	•
5	1	2	2	1	2	2	1	2	-++++++	•	•	•	•
6	1	2	2	2	1	2	2	1	-++++++-	•	•	•	•
7	2	1	2	2	1	1	2	2	+-++++++	•	•	•	•
8	2	1	2	1	2	2	2	1	+-++++++	•	•	•	•
9	2	1	1	2	2	2	1	2	+--++++-	•	•	•	•
10	2	2	2	1	1	1	1	2	+++----+	•	•	•	•
11	2	2	1	2	1	2	1	1	+++++---	•	•	•	•
12	2	2	1	1	2	1	2	1	+++-+-+-	•	•	•	•

Taguchi Design Options

The Taguchi Design red triangle menu contains options for design setup and generation.

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a

column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a

new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Chapter 15

Evaluate Designs

Explore Properties of Your Design

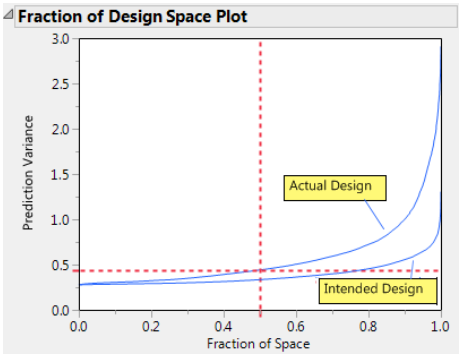
Using the Evaluate Design platform, you can:

- See the strengths and limitations of your existing experimental design.
- Determine your design's ability to detect effects associated with meaningful changes in the response.
- Address prediction variance and the precision of your estimates.
- Gain insight on aliasing.
- Obtain efficiency measures.

The Evaluate Design platform generates the results that appear in the Design Evaluation outlines provided by several DOE platforms. Diagnostics provided by both Evaluate Design and the Design Evaluation outlines include the following:

- power analysis
- a prediction variance profiler and surface plot
- a fraction of design space plot, showing how much of the design space has prediction variance above a given value
- estimation efficiency measures for parameter estimates
- the alias matrix, showing the bias structure for model effects
- a color map showing absolute correlations among effects
- design efficiency values

Figure 15.1 Comparison of Two Fraction of Design Space Plots



Contents

Overview of the Evaluate Design Platform 437
Example of Evaluate Design 438
Assessing the Impact of Lost Runs 438
Evaluating Power Relative to a Specified Model 445
Evaluate Design Launch Window 447
Evaluate Design Window 447
Factors 449
Model 449
Alias Terms 450
Design 451
Design Evaluation 451
Power Analysis 451
Prediction Variance Profile 458
Fraction of Design Space Plot 459
Prediction Variance Surface 460
Estimation Efficiency 462
Alias Matrix 464
Color Map on Correlations 465
Design Diagnostics 467
Evaluate Design Options 469

Overview of the Evaluate Design Platform

The Evaluate Design platform provides powerful tools that enable you to assess the properties of your design, whether it is created by JMP or another tool. You can use the platform before conducting an experiment to choose from several designs. You can also assess the impact of incorrect settings or lost runs in a design that you have conducted. You can modify the terms in your assumed model to see the impact of estimating a modified model. You can also modify the terms that appear in the Alias Terms outline to see the impact on the Alias Matrix.

You start by entering information about the design in the launch window. Then you can modify the assumed model and specify which effects not included in the model are of potential interest. Based on your specifications, the Design Evaluation platform then provides a number of ways to evaluate the properties of the generated design:

Power Analysis Enables you to explore your ability to detect effects of given sizes.

Prediction Variance Profile Shows the prediction variance over the range of factor settings.

Fraction of Design Space Plot Shows how much of the model prediction variance lies below or above a given value.

Prediction Variance Surface Shows a surface plot of the prediction variance for any two continuous factors.

Estimation Efficiency For each parameter, gives the fractional increase in the length of a confidence interval compared to that of an ideal (orthogonal) design, which might not exist. Also gives the relative standard error of the parameters.

Alias Matrix Gives coefficients that indicate the degree to which the model parameters are biased by effects that are potentially active, but not in the model.

Color Map on Correlations Shows the absolute correlations between effects on a plot using an intensity scale.

Note: The default intensity scale is a gray scale. To change the color scale, click the Color Map on Correlations red triangle and select **Blue to Gray to Red**. For a custom color scale, right-click in the plot and select **Color Theme**.

Design Diagnostics Gives efficiency measures for your design.

Note: In several DOE platforms, when you construct a design, a Design Evaluation outline appears. This outline shows results provided by the Evaluate Design platform. The platforms that provide a Design Evaluation outline are: Custom Design, Definitive Screening Design, Screening Design, Response Surface Design, and Mixture Design with Optimal design type.

Example of Evaluate Design

- "Assessing the Impact of Lost Runs"
- "Evaluating Power Relative to a Specified Model"

Assessing the Impact of Lost Runs

An experiment was conducted to explore the effect of three factors (Silica, Sulfur, and Silane) on tennis ball bounciness (Stretch). The goal of the experiment is to develop a predictive model for Stretch. A 15-run Box-Behnken design was selected using the Response Surface Design platform. After the experiment, the researcher learned that the two runs where Silica = 0.7 and Silane = 50 were not processed correctly. These runs could not be included in the analysis of the data.

Use the Evaluate Design platform to assess the impact of not including those two runs. Obtain diagnostics for the intended 15-run design and compare these to the actual 13-run design that is missing the two runs.

Construct the Intended and Actual Designs

Intended Design

1. Select **Help > Sample Data Library** and open Design Experiment/Bounce Data.jmp.
2. Select **DOE > Design Diagnostics > Evaluate Design**.
3. Select Silica, Sulfur, and Silane and click **X, Factor**.

   You can add Stretch as **Y, Response** if you wish. But specifying the response has no effect on the properties of the design.
4. Click **OK**.

Leave your Evaluate Design window for the intended design open.

Tip: Place the Evaluate Design window for the intended design in the left area of your screen. After the next steps, you will place the corresponding window for the actual design to its right.

Actual Design with Missing Runs

In this section, you will exclude the two runs where Silica = 0.7 and Silane = 50. These are rows 3 and 7 in the data table.

1. In Bounce Data.jmp, select rows 3 and 7, right-click in the highlighted area, and select **Hide and Exclude**.
2. Select **DOE > Design Diagnostics > Evaluate Design**.
3. Click **Recall**.
4. Click **OK**.

Leave your Evaluate Design window for the actual design open.

Tip: Place the Evaluate Design window for the actual design to the *right* of the Evaluate Design window for the intended design to facilitate comparing the two designs.

Comparison of Intended and Actual Designs

You can now compare the two designs using these methods:

- "Power Analysis"
- "Prediction Variance Profile"
- "Fraction of Design Space Plot"
- "Estimation Efficiency"
- "Color Map on Correlations"
- "Design Diagnostics"

Power Analysis

In each window, do the following:

1. Open the Power Analysis outline.

   The outline shows default values of 1 for all Anticipated Coefficients. These values correspond to detecting a change in the anticipated response of 2 units across the levels of main effect terms, assuming that the interaction and quadratic terms are not active.

   The power calculations assume an error term (Anticipated RMSE) of 1. From previous studies, you believe that the RMSE is approximately 2.

2. Type 2 next to **Anticipated RMSE**.

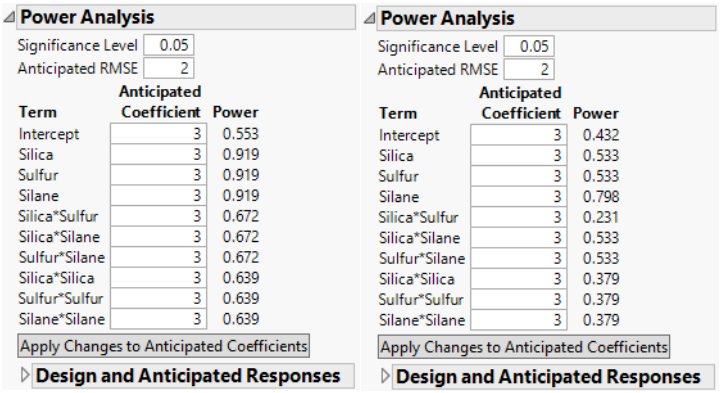
   When you click outside the text box, the power values are updated.

   You are interested in detecting differences in the anticipated response that are on the order of 6 units across the levels of main effects, assuming that interaction and quadratic terms are not active. To set these uniformly, use a red triangle option.

3. Click the Power Analysis red triangle and select **Set Delta for Power**
4. Type 6 as your value for delta.
5. Click **OK**.

Figure 15.2 shows both outlines, with the Design and Anticipated Responses outline closed.

Figure 15.2 Power Analysis Outlines, Intended Design (Left) and Actual Design (Right)



Power Analysis

Significance Level 0.05
Anticipated RMSE 2

Term	Anticipated Coefficient	Power
Intercept	3	0.553
Silica	3	0.919
Sulfur	3	0.919
Silane	3	0.919
Silica*Sulfur	3	0.672
Silica*Silane	3	0.672
Sulfur*Silane	3	0.672
Silica*Silica	3	0.639
Sulfur*Sulfur	3	0.639
Silane*Silane	3	0.639

Apply Changes to Anticipated Coefficients

Design and Anticipated Responses

Power Analysis

Significance Level 0.05
Anticipated RMSE 2

Term	Anticipated Coefficient	Power
Intercept	3	0.432
Silica	3	0.533
Sulfur	3	0.533
Silane	3	0.798
Silica*Sulfur	3	0.231
Silica*Silane	3	0.533
Sulfur*Silane	3	0.533
Silica*Silica	3	0.379
Sulfur*Sulfur	3	0.379
Silane*Silane	3	0.379

Apply Changes to Anticipated Coefficients

Design and Anticipated Responses

The power values for the actual design are uniformly smaller than for the intended design. For Silica and Sulfur, the power of the tests in the intended design is almost twice the power in the actual design. For the Silica*Sulfur interaction, the power of the test in the actual design is 0.231, compared to 0.672 in the intended design. The actual design results in substantial loss of power in comparison with the intended design.

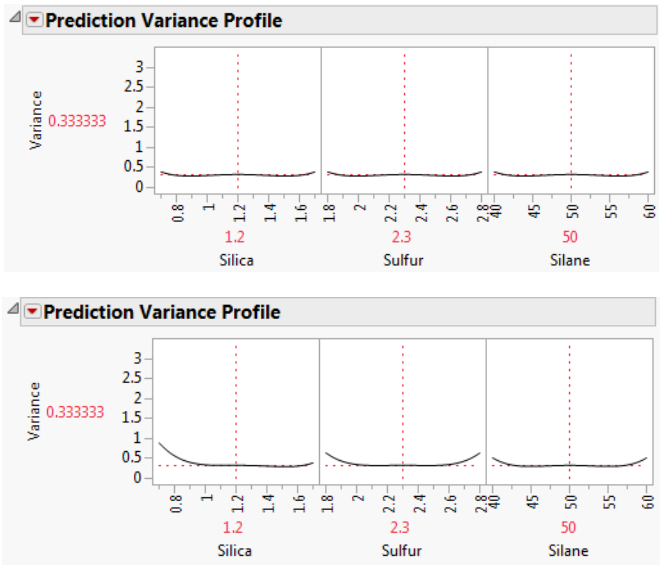
Prediction Variance Profile

1. In each window, open the Prediction Variance Profile outline.
2. In the window for the actual design, right-click on the vertical axis and from the menu select **Edit > Copy Axis Settings**.

   This action creates a script containing the axis settings. Next, apply these axis settings to the Prediction Variance Profile plot for the intended design.
3. In the Evaluate Design window for the intended design, locate the Prediction Variance Profile outline. Right-click on the vertical axis and from the menu select **Edit > Paste Axis Settings**.

   The plots are shown in Figure 15.5, with the plot for the intended design at the top and for the actual design at the bottom.

Figure 15.3 Prediction Variance Profile, Intended Design (Top) and Actual Design (Bottom)



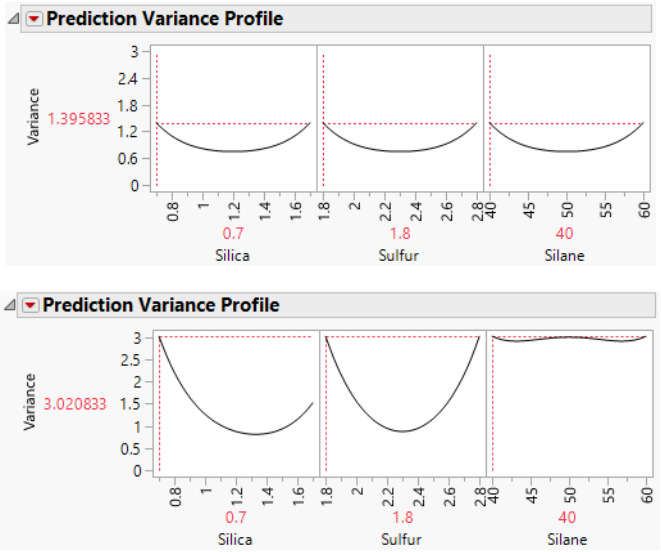
The Prediction Variance Profile plots are profiler views of the relative prediction variance. You can explore the relative prediction variance in various regions of design space.

Both plots show the same relative prediction variance in the center of the design space. However, the variance for points near the edges of the design space appears greater than for the same points in the intended design. Explore this phenomenon by moving all three vertical lines to points near the edges of the factor settings.

4. In both windows, click the Prediction Variance Profile red triangle and select **Maximize Variance**.

   Figure 15.4 shows the maximum relative prediction variance for the intended and actual designs.

Figure 15.4 Prediction Variance Profile Maximized, Intended Design (Top) and Actual Design (Bottom)



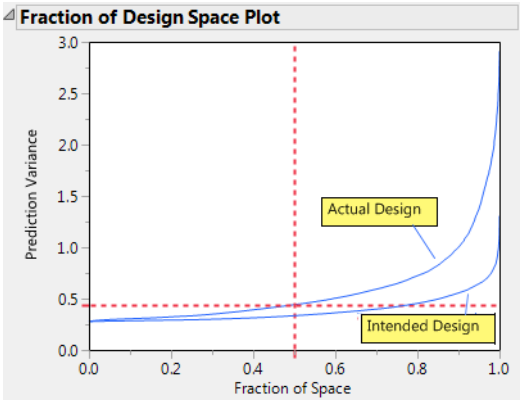
For both designs, the profilers identify the same point as one of the design points where the maximum prediction variance occurs: Silica=0.7, Sulfur=1.8, and Silane=40. The maximum prediction variance is 1.396 for the intended design, and 3.021 for the actual design. Note that there are other points where the prediction variance is maximized. The larger maximum prediction variance for the actual design means that predictions in parts of the design space are less accurate than they would have been had the intended design been conducted.

Fraction of Design Space Plot

1. In each window, open the Fraction of Design Space Plot outline.
2. In the window for the intended design, right-click in the plot and select **Edit > Copy Frame Contents**.
3. In the window for the actual design, locate the Fraction of Design Space Plot outline.
4. Right-click in the plot and select **Edit > Paste Frame Contents**

Figure 15.5 shows the plot with annotations. Each Fraction of Design Space Plot shows the proportion of the design space for which the relative prediction variance falls below a specific value.

Figure 15.5 Fraction of Design Space Plots



The relative prediction variance for the actual design is greater than that of the intended design over the entire design space. The discrepancy increases with larger design space coverage.

Estimation Efficiency

In each window, open the Estimation Efficiency outline.

Figure 15.6 Estimation Efficiency Outlines, Intended Design (Left) and Actual Design (Right)

Estimation Efficiency

Term	Fractional Increase in CI Length	Relative Std Error of Estimate
Intercept	1.236	0.577
Silica	0.369	0.354
Sulfur	0.369	0.354
Silane	0.369	0.354
Silica*Sulfur	0.936	0.5
Silica*Silane	0.936	0.5
Sulfur*Silane	0.936	0.5
Silica*Silica	1.016	0.52
Sulfur*Sulfur	1.016	0.52
Silane*Silane	1.016	0.52

Estimation Efficiency

Term	Fractional Increase in CI Length	Relative Std Error of Estimate
Intercept	1.082	0.577
Silica	0.803	0.5
Sulfur	0.803	0.5
Silane	0.275	0.354
Silica*Sulfur	2.122	0.866
Silica*Silane	0.803	0.5
Sulfur*Silane	0.803	0.5
Silica*Silica	1.268	0.629
Sulfur*Sulfur	1.268	0.629
Silane*Silane	1.268	0.629

In the actual design (right), the relative standard errors for all parameters either exceed or equal the standard errors for the intended design (left). For all except three of the non-intercept parameters, the relative standard errors in the actual design exceed those in the intended design.

The Fractional Increase in CI Length compares the length of a parameter's confidence interval as given by the current design to the length of such an interval given by an ideal design of the same run size. The length of the confidence interval, and consequently the Fractional Increase in CI Length, is affected by the number of runs. See "Fractional Increase in CI Length". Despite the reduction in run size, for the actual design, the terms Silane, Silica*Silane, and Sulfur*Silane have a smaller increase than for the intended design. This is because the two runs that were removed to define the actual design had Silane set to its center point. By removing these runs, the widths of the confidence intervals for these parameters more closely resemble those of an ideal orthogonal design, which has no center points.

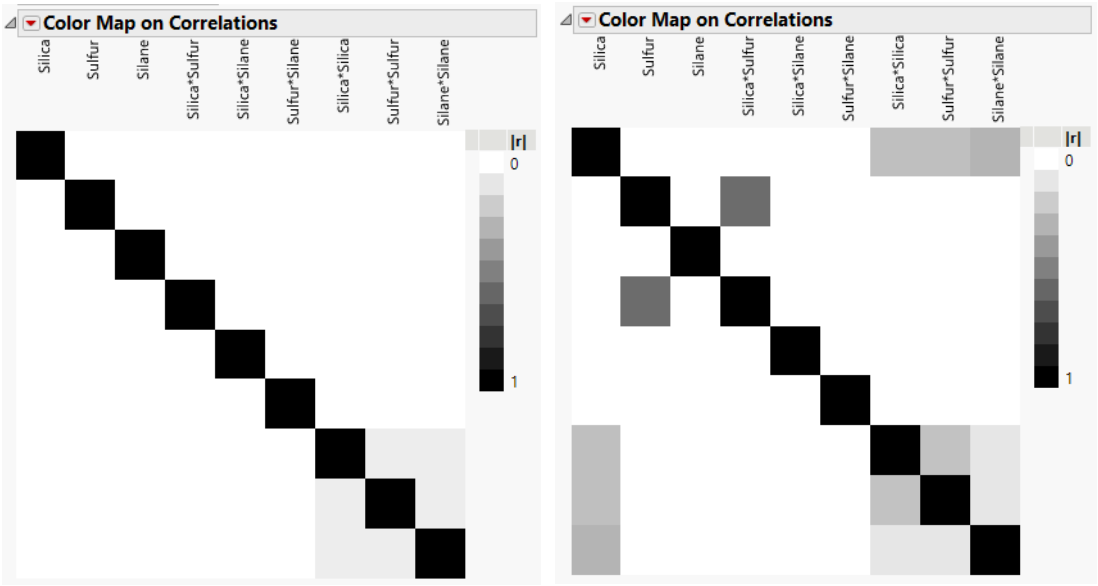
Color Map on Correlations

In each report, do the following:

1. Open the **Color Map On Correlations** outline.

   The two color maps show the effects in the Model outline. Each plot shows the absolute correlations between effects colored using the JMP default white to black intensity scale. Ideally, you would like zero or very small correlations between effects.

Figure 15.7 Color Map on Correlations, Intended Design (Left) and Actual Design (Right)

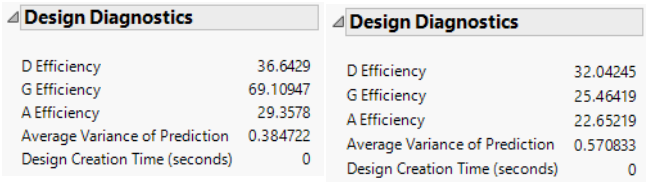


The absolute values of the correlations range from 0 (white) to 1 (black). Hover over a cell to see the value of the absolute correlation. The color map for the actual design shows more absolute correlations that are large than does the color map for the intended design. For example, the correlation between Sulfur and Silica*Sulfur is < .0001 for the intended design, and 0.5774 for the actual design.

Design Diagnostics

In each report, open the Design Diagnostics outline.

Figure 15.8 Design Diagnostics, Intended Design (Left) and Actual Design (Right)



Design Diagnostics

D Efficiency	36.6429
G Efficiency	69.10947
A Efficiency	29.3578
Average Variance of Prediction	0.384722
Design Creation Time (seconds)	0

Design Diagnostics

D Efficiency	32.04245
G Efficiency	25.46419
A Efficiency	22.65219
Average Variance of Prediction	0.570833
Design Creation Time (seconds)	0

The intended design (left) has higher efficiency values and a lower average prediction variance than the actual design (right). The results of the Design Evaluation analysis indicate that the two lost runs have had a negative impact on the design.

Note that both the number of runs and the model matrix factor into the calculation of efficiency measures. In particular, the *D*-, *G*-, and *A*- efficiencies are calculated relative to the ideal design for the run size of the given design. It is not necessarily true that larger designs are more efficient than smaller designs. However, for a given number of factors, larger designs tend to have smaller Average Variance of Prediction values than do smaller designs. For more information on how efficiency measures are defined, see "Design Diagnostics".

Evaluating Power Relative to a Specified Model

For this example, you have constructed a definitive screening design to determine which of six factors have an effect on the yield of an extraction process. The data are given in the Extraction Data.jmp sample data table, located in the Design Experiment folder. Because the design is a *definitive* screening design, each factor has three levels. See "Definitive Screening Designs".

You are interested in the power of tests to detect a strong quadratic effect. You consider a strong effect to be one whose magnitude is at least three times as large as the error variation.

Although the experiment studies six factors, effect sparsity suggests that only a small subset of factors is active. Consequently, you feel comfortable investigating power in a model based on a smaller number of factors. Also, past studies on a related process provide strong evidence to suggest that three of the factors, Propanol, Butanol, and pH, have negligible main effects, do not interact with other factors, and do not have quadratic effects. This leads you to believe that the likely model contains main, interaction, and quadratic effects only for Methanol, Ethanol, and Time. You decide to investigate power in the context of a three-factor response surface model.

Use the Evaluate Design platform to determine the power of your design to detect strong quadratic effects for Methanol, Ethanol, or Time.

1. Select **Help > Sample Data Library** and open Design Experiment/Extraction Data.jmp.

2. Select **DOE > Design Diagnostics > Evaluate Design**.
3. Select Methanol, Ethanol, and Time and click **X, Factor**.

   You can add Yield as **Y, Response** if you wish. But specifying the response has no effect on the properties of the design.
4. Click **OK**.
5. In the Model outline, click **RSM**.

   This adds the interaction and quadratic terms for the three factors.
6. Open the Power Analysis outline.

   Note that the Anticipated RMSE is set to 1 by default. Although you have an estimate of the RMSE from past studies, you need not enter it. This is because the magnitude of the effect of interest is three times the error variation.
7. Under **Anticipated Coefficient**, type 3 next to Methanol*Methanol, Ethanol*Ethanol, and Time*Time.
8. Click **Apply Changes to Anticipated Coefficients**.

Figure 15.9 Power Analysis Outline after Applying Changes to Coefficients

Power Analysis

Significance Level 0.05
Anticipated RMSE 1

Term	Anticipated Coefficient	Power
Intercept	1	0.111
Methanol	1	0.573
Ethanol	1	0.573
Time	1	0.573
Methanol*Methanol	3	0.737
Methanol*Ethanol	-1	0.331
Ethanol*Ethanol	3	0.737
Methanol*Time	-1	0.331
Ethanol*Time	1	0.331
Time*Time	3	0.737

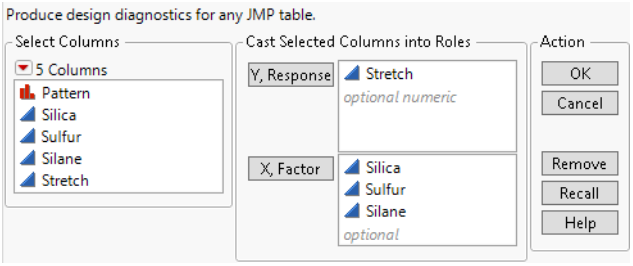
Apply Changes to Anticipated Coefficients

The power of detecting a quadratic effect whose magnitude is three times the error variation is 0.737. This assumes a final model that is a response surface in three factors. It also assumes a 0.05 significance level for the test.

Evaluate Design Launch Window

To launch the Evaluate Design platform, open the data table of interest and select **DOE** > **Design Diagnostics > Evaluate Design**. The example in Figure 15.10 uses the Bounce Data.jmp sample data table, located in the Design Experiment folder.

Figure 15.10 Evaluate Design Launch Window



The launch window contains the following buttons:

Y, Response Enter the response column or columns. Entering a response is optional. Response values are not used in evaluating the design. Responses must be numeric.

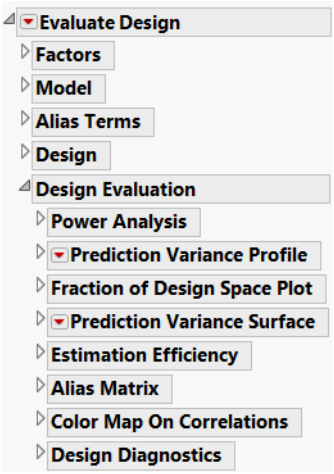
X, Factor Enter the factor columns. Factors can be of any Data Type or Modeling Type.

Evaluate Design Window

The Evaluate Design window consists of two parts. See Figure 15.11, where all outline nodes are closed.

- The Factors, Model, Alias Terms, and Design outlines define the model and design.
- The Design Evaluation outline provides results that describe the properties of your design.

Figure 15.11 Evaluate Design Window Showing All Possible Outlines



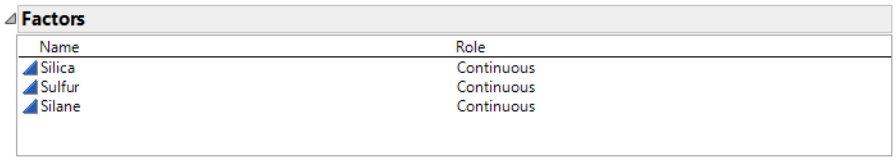
The Factors, Model, Alias Terms, and Design outlines contain information that you enter about the factors, assumed model, potentially aliased effects of interest, and the actual design. JMP populates these outlines using your selections in the launch window and the design table. However, you can modify the effects in the Model and Alias Terms outlines. Once you have made your specifications, the Design Evaluation outlines are updated. You can open these outlines to see reports or control windows that provide information about your design.

- "Factors"
- "Model"
- "Alias Terms"
- "Design"
- "Design Evaluation"
- "Power Analysis"
- "Prediction Variance Profile"
- "Fraction of Design Space Plot"
- "Prediction Variance Surface"
- "Estimation Efficiency"
- "Alias Matrix"
- "Color Map on Correlations"
- "Design Diagnostics"

Factors

The factors outline lists the factors entered in the launch window. You can select factors to construct effects in the Model outline.

Figure 15.12 Factors Outline



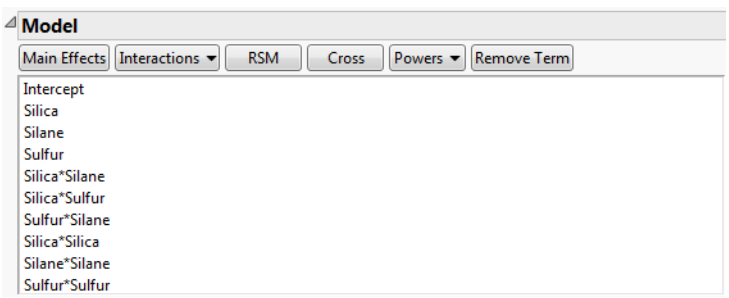
Model

What appears in the Model outline depends upon whether the associated data table contains a Model or Fit Model script:

- If there is a script, the Model outline contains the estimable effects specified in the script. Effects are entered into the model in the order specified in the script. If the model script contains more effects than are estimable, those that are not estimable are not included in the model for evaluation.
- If there is no script, the Model outline contains only main effects.

Figure 15.13 shows the Model outline for the Bounce Data.jmp data table, found in the Design Experiment folder. The Model script in the data table contains response surface effects for the three factors Silica, Silane, and Sulfur. Consequently, the Model outline contains the main effects, two-way interactions, and quadratic effects for these three factors.

Figure 15.13 Model Outline for Bounce Data.jmp



You can add effects to the Model outline using the following buttons:

Main Effects Adds main effects for all factors in the model.

Interactions Adds interaction effects. If no factors are selected in the Factors outline, select 2nd, 3rd, 4th, or 5th to add all appropriate interactions up to that order. Add interactions up to a given order for specific factors by selecting the factor names in the Factors outline, selecting Interactions, and then specifying the appropriate order. Interactions between non-mixture and mixture factors, and interactions with blocking and constant factors, are not added.

RSM Adds interaction and quadratic terms up to the second order (response surface model terms) for continuous factors. Categorical factors are not included in RSM terms. Main effects for non-mixture factors that interact with all the mixture factors are removed.

Cross Adds specific interaction terms. Select factor names in the Factors outline and effect names in the Model outline. Click Cross to add the crossed terms to the Model outline.

Powers Adds polynomial terms. If no factor names are selected in the Factors outline, adds polynomial terms for all continuous factors. If factor names are selected in the Factors outline, adds polynomial terms for only those factors. Select 2nd, 3rd, 4th, or 5th to add polynomial terms of that order.

Scheffé Cubic Adds Scheffé cubic terms for all mixture factors. These terms are used to specify a mixture model with third-degree polynomial terms.

Remove Term Removes selected effects.

Alias Terms

It is possible that effects *not* included in your assumed model are active. In the Alias Terms outline, list potentially active effects that are not in your assumed model but might bias the estimates of model terms. The Alias Matrix entries represent the degree of bias imparted to model parameters by the effects that you specified in the Alias Terms outline. See "The Alias Matrix".

By default, the Alias Terms outline includes all two-way interaction effects that are not in your Model outline (with the exception of terms involving blocking factors). Add or remove terms using the buttons. For a description of how to use these buttons to add effects to the Alias Terms table, see "Model".

In the Evaluate Design platform, the Alias Matrix outline is immediately updated to reflect changes to Alias Matrix effects. In the Custom Design platform, you must click Make Design after modifying the effects in the Alias Terms outline. Within other DOE platforms that construct designs, there is no Alias Terms outline. However, the Alias Matrix outline, containing appropriate effects, appears under Design Evaluation after you construct the design.

Design

The Design outline shows the design runs for the factors that you have specified in the launch window. You can easily view the design as you explore its properties in the Design Evaluation outline.

Design Evaluation

Design Evaluation within the Evaluate Design platform is based on your design and the specifications that you make in the Model and Alias Terms outlines. Several DOE Design platforms provide a Design Evaluation outline: Custom, Definitive Screening, Screening, Response Surface, and Mixture with Optimal design type. Design Evaluation within these platforms is based on the design that you construct.

Power Analysis

The Power Analysis outline calculates the power of tests for the parameters in your model. Power is the probability of detecting an active effect of a given size. The Power Analysis outline helps you evaluate the ability of your design to detect effects of practical importance. The higher your power, the more likely you are to detect significant effects assuming your coefficient and RMSE assumptions are correct. Power depends on the number of runs, the significance level, and the estimated error variation. In particular, you can determine if additional runs are necessary.

This section covers the following topics:

- "Power Analysis Overview"
- "Power Analysis Details"
- "Tests for Individual Parameters"
- "Tests for Categorical Effects with More Than Two Levels"
- "Design and Anticipated Responses Outline"
- "Power Analysis for Coffee Experiment"

Power Analysis Overview

Power is calculated for the effects listed in the Model outline. These include continuous, discrete numeric, categorical, blocking, mixture, and covariate factors. The tests are for individual model parameters and for whole effects. For more information on how power is calculated, see "Power Calculations".

Power is the probability of rejecting the null hypothesis of no effect at specified values of the model parameters. In practice, your interest is not in the values of the model parameters, but in detecting differences in the mean response of practical importance. In the Power Analysis outline, you can compute Anticipated Responses for specified values of the Anticipated Coefficients. This helps you to determine the coefficient values associated with the differences you want to detect in the mean response.

Figure 15.14 shows the Power Analysis outline for the design in the Coffee Data.jmp sample data table, found in the Design Experiment folder. The model specified in the Model script is a main effects only model.

Figure 15.14 Power Analysis for Coffee Data.jmp

Power Analysis

Significance Level 0.05
Anticipated RMSE 1

Term	Anticipated Coefficient	Power
Intercept	1	0.789
Grind	-1	0.789
Temperature	1	0.789
Time	1	0.789
Charge	1	0.789
Station 1	1	0.507
Station 2	1	0.507

Apply Changes to Anticipated Coefficients

Effect	Power
Station	0.888

Design and Anticipated Responses

Anticipated Response	Grind	Temperature	Time	Charge	Station
6	Medium	205	4	2.4	1
-2	Coarse	195	3	1.6	1
2	Medium	205	3	1.6	1
2	Coarse	195	4	2.4	1
2	Coarse	205	4	1.6	2
4	Medium	195	4	2.4	2
0	Medium	195	3	1.6	2
2	Coarse	205	3	2.4	2
-1	Coarse	205	4	1.6	3
-3	Coarse	195	3	2.4	3
-1	Medium	195	4	1.6	3
1	Medium	205	3	2.4	3

Apply Changes to Anticipated Responses

In the Power Analysis outline, you can:

- Specify coefficient values that reflect differences that you want to detect. You enter these as Anticipated Coefficients in the top part of the outline.
- Specify anticipated response values and apply these to determine the corresponding Anticipated Coefficients. You specify Anticipated Responses in the Design and Anticipated Responses panel.

From the Power Analysis red triangle menu, you can:

- Specify a power to determine anticipated coefficients to achieve that power. If coefficients do not exist to satisfy the specified power, then the coefficients are set to zero.
- Specify a value for delta, where anticipated coefficients will be half of the specified delta. The power is then updated based on the anticipated coefficients.

Power Analysis Details

Specify values for the Significance Level and Anticipated RMSE. These are used to calculate the power of the tests for the model parameters.

Significance Level The probability of rejecting the hypothesis of no effect, if it is true. The power calculations update immediately when you enter a value.

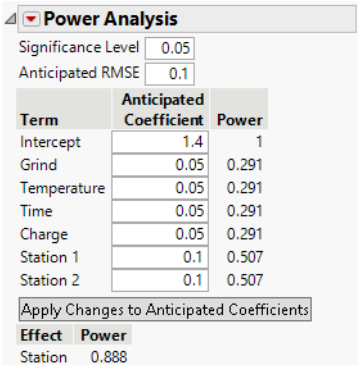
Anticipated RMSE An estimate of the square root of the error variation. The power calculations update immediately when you enter a value.

The top portion of the Power Analysis report opens with default values for the Anticipated Coefficients (Figure 15.14). The default values are based on Delta. See "Advanced Options > Set Delta for Power".

Note: If the design is supersaturated, meaning that the number of parameters to be estimated exceeds the number of runs, the anticipated coefficients are set to 0.

Figure 15.15 shows the top portion of the Power Analysis report where values have been specified for the Anticipated Coefficients. These values reflect the differences you want to detect.

Figure 15.15 Possible Specification of Anticipated Coefficients for Coffee Data.jmp



Tests for Individual Parameters

The Term column contains a list of model terms. For each term, the Anticipated Coefficient column contains a value for that term. The value in the Power column is the power of a test that the coefficient for the term is 0 if the true value of the coefficient is given by the Anticipated Coefficient.

Term The model term associated with the coefficient being tested.

> **Note:** The order in which model terms appear in the Power Analysis report may not be identical to their order in the Parameter Estimates report obtained using Standard Least Squares. This difference can only occur when the model contains an interaction with more than one degree of freedom.

Anticipated Coefficient A value for the coefficient associated with the model term. This value is used in the calculations for Power. These values are also used to calculate the Anticipated Response column in the Design and Anticipated Responses outline. When you set a new value in the Anticipated Coefficient column, click **Apply Changes to Anticipated Coefficients** to update the Power and Anticipated Response columns.

> **Note:** The anticipated coefficients have default values of 1 for continuous effects. They have alternating values of 1 and –1 for categorical effects. You can specify a value for Delta be selecting **Advanced Options > Set Delta for Power** from the red triangle menu. If you change the value of Delta, the values of the anticipated coefficients are updated so that their absolute values are one-half of Delta. See "Advanced Options > Set Delta for Power".

Power Probability of rejecting the null hypothesis of no effect when the true coefficient value is given by the specified Anticipated Coefficient. For a coefficient associated with a numeric factor, the change in the mean response (based on the model) is twice the coefficient value. For a coefficient associated with a categorical factor, the change in the mean response (based on the model) across the levels of the factor equals twice the absolute value of the anticipated coefficient.

Calculations use the specified Significance Level and Anticipated RMSE. For more information about the power calculation, see "Power for a Single Parameter".

Apply Changes to Anticipated Coefficients When you set a new value in the Anticipated Coefficient column, click **Apply Changes to Anticipated Coefficients** to update the Power and Anticipated Response columns.

Tests for Categorical Effects with More Than Two Levels

If your model contains a categorical effect with more than two levels, then the following columns appear below the Apply Changes to Anticipated Coefficients button:

Effect The categorical effect.

Power The power calculation for a test of no effect. The null hypothesis for the test is that all model parameters corresponding to the effect are zero. The difference to be detected is defined by the values in the Anticipated Coefficient column that correspond to the model terms for the effect. The power calculation reflects the differences in response means determined by the anticipated coefficients.

Calculations use the specified Significance Level and Anticipated RMSE. For more information about the power calculation, see "Power for a Categorical Effect".

Design and Anticipated Responses Outline

The Design and Anticipated Responses outline shows the design preceded by an Anticipated Response column. Each entry in the first column is the Anticipated Response corresponding to the design settings. The Anticipated Response is calculated using the Anticipated Coefficients.

Figure 15.16 shows the Design and Anticipated Responses outline corresponding to the specification of Anticipated Coefficients given in Figure 15.15.

Figure 15.16 Anticipated Responses for Coffee Data.jmp

Design and Anticipated Responses

Anticipated Response	Grind	Temperature	Time	Charge	Station
1.6	Medium	205	4	2.4	1
1.4	Coarse	195	3	1.6	1
1.4	Medium	205	3	1.6	1
1.6	Coarse	195	4	2.4	1
1.6	Coarse	205	4	1.6	2
1.5	Medium	195	4	2.4	2
1.3	Medium	195	3	1.6	2
1.6	Coarse	205	3	2.4	2
1.3	Coarse	205	4	1.6	3
1.2	Coarse	195	3	2.4	3
1.1	Medium	195	4	1.6	3
1.2	Medium	205	3	2.4	3

Apply Changes to Anticipated Responses

In the Anticipated Response column, you can specify a value for each setting of the factors. These values reflect the differences you want to detect.

Click **Apply Changes to Anticipate Responses** to update both the Anticipated Coefficient and Power columns.

Anticipated Response The response value obtained using the Anticipated Coefficient values as coefficients in the model. When the outline first appears, the calculation of Anticipated Response values is based on the default values in the Anticipated Coefficient column. When you set new values in the Anticipated Response column, click **Apply Changes to Anticipated Responses** to update the Anticipated Coefficient and Power columns.

Design The columns to the right of the Anticipated Response column show the factor settings for all runs in your design.

Apply Changes to Anticipated Responses When you set new values in the Anticipated Response column, click **Apply Changes to Anticipated Responses** to update the Anticipated Coefficient and Power columns.

Power Analysis for Coffee Experiment

Consider the design in the Coffee Data.jmp data table. Suppose that you are interested in the power of your design to detect effects of various magnitudes on Strength. Recall that Grind is a two-level categorical factor, Temperature, Time, and Charge are continuous factors, and Station is a three-level categorical (blocking) factor.

In this example, ignore the role of Station as a blocking factor. You are interested in the effect of Station on Strength. Since Station is a three-level categorical factor, it is represented by two terms in the Parameters list: Station 1 and Station 2.

Specifically, you are interested the probability of detecting the following changes in the mean Strength:

- A change of 0.10 units as you vary Grind from Coarse to Medium.
- A change of 0.10 units or more as you vary Temperature, Time, and Charge from their low to high levels.
- An increase due to each of Stations 1 and 2 of 0.10 units beyond the overall anticipated mean. This corresponds to a decrease due to Station 3 of 0.20 units from the overall anticipated mean.

You set 0.05 as your Significance Level. Your estimate of the standard deviation of Strength for fixed design settings is 0.1 and you enter this as the Anticipated RMSE.

Figure 15.17 shows the Power Analysis node with these values entered. Specifically, you specify the Significance Level, Anticipated RMSE, and the value of each Anticipated Coefficient.

When you click Apply Changes to Anticipated Coefficients, the Anticipated Response values are updated to reflect the model you have specified.

Figure 15.17 Power Analysis Outline with User Specifications in Anticipated Coefficients Panel

Power Analysis

Significance Level 0.05
Anticipated RMSE 0.1

Term	Anticipated Coefficient	Power
Intercept	1.4	1
Grind	0.05	0.291
Temperature	0.05	0.291
Time	0.05	0.291
Charge	0.05	0.291
Station 1	0.1	0.507
Station 2	0.1	0.507

Apply Changes to Anticipated Coefficients

Effect	Power
Station	0.888

Design and Anticipated Responses

Anticipated Response	Grind	Temperature	Time	Charge	Station
1.6	Medium	205	4	2.4	1
1.4	Coarse	195	3	1.6	1
1.4	Medium	205	3	1.6	1
1.6	Coarse	195	4	2.4	1
1.6	Coarse	205	4	1.6	2
1.5	Medium	195	4	2.4	2
1.3	Medium	195	3	1.6	2
1.6	Coarse	205	3	2.4	2
1.3	Coarse	205	4	1.6	3
1.2	Coarse	195	3	2.4	3
1.1	Medium	195	4	1.6	3
1.2	Medium	205	3	2.4	3

Apply Changes to Anticipated Responses

Recall that Temperature is a continuous factor with coded levels of -1 and 1. Consider the test whose null hypothesis is that Temperature has no effect on Strength. Figure 15.17 shows that the power of this test to detect a difference of 0.10 (=2*0.05) units across the levels of Temperature is only 0.291.

Now consider the test for the whole Station effect, where Station is a three-level categorical factor. Consider the test whose null hypothesis is that Station has no effect on Strength. This is the usual *F* test for a categorical factor provided in the Effect Tests report when you run **Analyze > Fit Model**. See *Fitting Linear Models*.

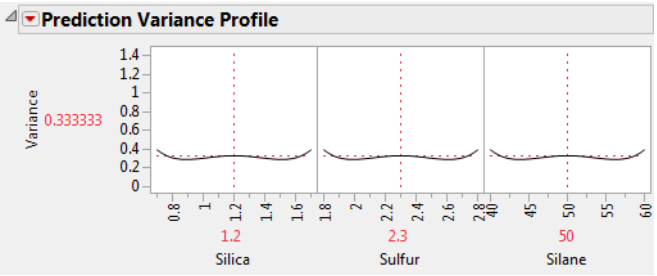
The Power of this test is shown directly beneath the Apply Changes to Anticipated Coefficients button. The entries under Anticipated Coefficients for the model terms Station 1 and Station 2 are both 0.10. These settings imply that the effect of both stations is to increase Strength by 0.10 units above the overall anticipated mean. For these settings of the Station 1 and Station 2 coefficients, the effect of Station 3 on Strength is to decrease it by 0.20 units from the overall anticipated mean. Figure 15.17 shows that the power of the test to detect a difference of at least this magnitude is 0.888.

Prediction Variance Profile

The Prediction Variance Profile helps you to understand where in the design space your predictions have more or less variability. Low prediction variance is desired. Use the **Maximize Variance** option to find the maximum variance. See "Maximize Variance".

The Prediction Variance Profile plots the relative variance of prediction as a function of each factor at fixed values of the other factors. Figure 15.18 shows the Prediction Variance Profile for the Bounce Data.jmp data table, located in the Design Experiment folder.

Figure 15.18 Prediction Variance Profiler



Relative Prediction Variance

For given settings of the factors, the prediction variance is the product of the error variance and a quantity that depends on the design and the factor settings. Before you run your experiment, the error variance is unknown, so the prediction variance is also unknown. However, the ratio of the prediction variance to the error variance is not a function of the error variance. This ratio, called the *relative prediction variance*, depends only on the design and the factor settings. Consequently, the relative variance of prediction can be calculated before acquiring the data. See "Relative Prediction Variance".

After you run your experiment and fit a least squares model, you can estimate the error variance using the mean squared error (MSE) of the model fit. You can estimate the actual variance of prediction at any setting by multiplying the relative variance of prediction at that setting.

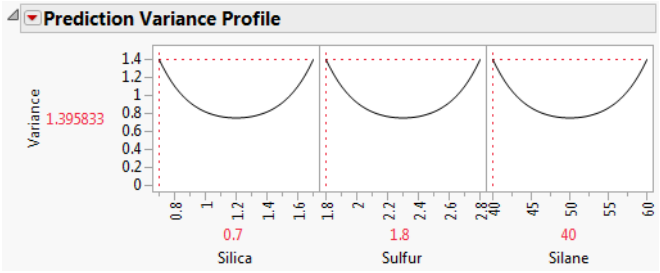
It is ideal for the prediction variance to be small throughout the design space. Generally, the error variance drops as the sample size increases. In comparing designs, you may want to place the prediction variance profilers for two designs side-by-side. A design with lower prediction variance on average is preferred.

Maximize Variance

You can evaluate a design or compare designs in terms of the maximum relative prediction variance. Select the **Maximize Variance** option from the Prediction Variance Profile red triangle menu. JMP uses a desirability function that maximizes the relative prediction variance. The value of the maximum variance in the Prediction Variance Profile is the worst (least desirable from a design point of view) value of the relative prediction variance. The maximum variance can occur at more than one combination of factor settings.

Figure 15.19 shows the Prediction Variance Profile with Maximize Variance selected. The plot is for the Bounce Data.jmp sample data table, located in the Design Experiment folder. The maximum value of the relative prediction variance is 1.3958 when Silica = 0.7, Sulfur = 1.8, and Silane = 40. However, keep in mind that several factor settings can have this same relative variance. The design point with Silica = 1.7, Sulfur = 2.8, and Silane = 60 also has relative prediction variance of 1.3958. To evaluate the prediction variance using a surface, see "Prediction Variance Surface".

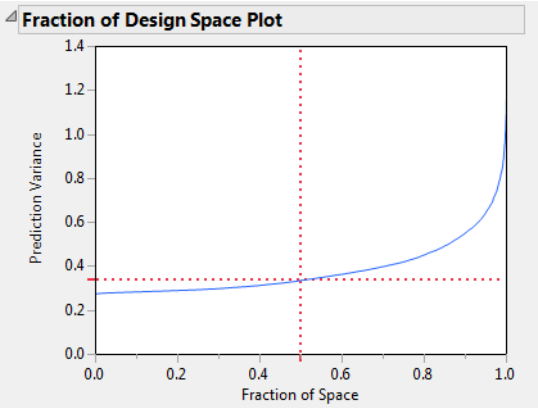
Figure 15.19 Prediction Variance Profile Showing Maximum Variance



Fraction of Design Space Plot

The Fraction of Design Space Plot shows the proportion of the design space over which the relative prediction variance lies below a given value. It is desirable to have a large proportion of the design space with low prediction variance values. Figure 15.20 shows the Fraction of Design Space plot for the Bounce Data.jmp sample data table, located in the Design Experiment folder.

Figure 15.20 Fraction of Design Space Plot



The *X* axis in the plot represents the proportion of the design space, ranging from 0 to 100%. The *Y* axis represents relative prediction variance values. For a point *(x, y)* that falls on the blue curve, the value *x* is the proportion of design space with variance less than or equal to *y*. Red dotted crosshairs mark the value that bounds the relative prediction variance for 50% of design space.

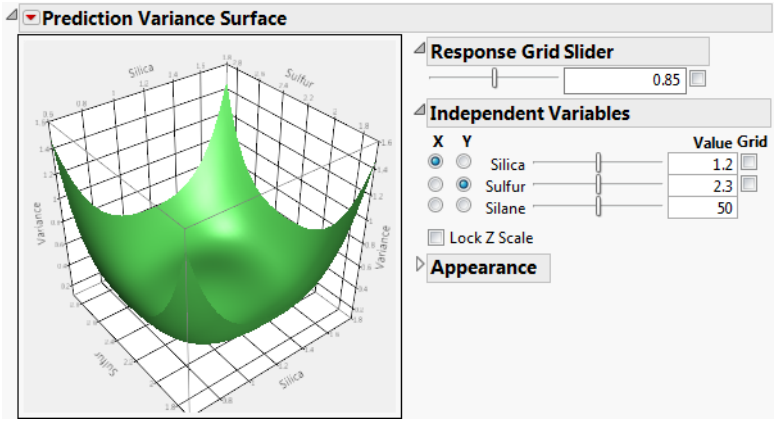
Figure 15.20 shows that the minimum relative prediction variance is slightly less than 0.3, while the maximum is below 1.4. (The actual maximum is 1.395833, as shown in Figure 15.19.) The red dotted crosshairs indicate that the relative prediction variance is less than 0.34 over about 50% of the design space. You can use the crosshairs tool to find the maximum relative prediction variance that corresponds to any Fraction of Space value. Use the crosshairs tool in Figure 15.20 to see that 90% of the prediction variance values are below approximately 0.55.

Note: Monte Carlo sampling of the design space is used in constructing the Fraction of Design Space Plot. Therefore, plots for the same design may vary slightly.

Prediction Variance Surface

The Prediction Variance Surface report plots the relative prediction variance surface as a function of any two design factors. Figure 15.21 shows the Prediction Variance Surface outline for the Bounce Data.jmp sample data table, located in the Design Experiment folder. Show or hide the controls by selecting **Control Panel** on the red triangle menu. See "Control Panel".

Figure 15.21 Prediction Variance Surface



When there are two or more factors, the Prediction Variance Surface outline shows a plot of the relative prediction variance for any two variables. The Prediction Variance Surface outline plots the relative prediction variance formula. Drag on the plot to rotate and change the perspective.

Control Panel

The Control Panel contains options for adjusting the plot:

Response Grid Slider The **Grid** check box superimposes a grid that shows constant values of Variance. The value of the Variance is shown in the text box. The slider enables you to adjust the placement of the grid. Alternatively, you can enter a Variance value in the text box. Click outside the box to update the plot.

Independent Variables This panel enables you to select which two factors are used as axes for the plot and to specify the settings for factors not used as axes. Select a factor for each of the X and Y axes by clicking in the appropriate column. Use the sliders and text boxes to specify values for each factor *not* selected for an axis. The plot shows the three-dimensional slice of the surface at the specified values of the factors that are not used as axes in the plot. Move the sliders to see different slices.

Each grid check box activates a grid for the corresponding factor. Use the sliders to adjust the placement of each grid.

Lock Z Scale locks the *z*-axis to its current values. This is useful when moving the sliders that are not on an axis.

Appearance The **Resolution** slider affects how many points are evaluated for a formula. Too coarse a resolution means that a function with a sharp change might not be represented

very well. But setting the resolution high can make evaluating and displaying the surface slower.

The **Orthographic projection** check box shows a projection of the plot in two dimensions.

The **Contour** menu controls the placement of contour curves. A contour curve is a set of points whose Response values are constant. You can select to turn the contours Off (the default) or place them contours Below, Above, or On Surface.

Estimation Efficiency

This report gives the Fractional Increase in CI (Confidence Interval) Length and Relative Std (Standard) Error of Estimate for each parameter estimate in the model. Smaller is better for both of these values. Figure 15.22 shows the Estimation Efficiency outline for the Bounce Data.jmp sample data table, located in the Design Experiment folder.

Figure 15.22 Estimation Efficiency Outline

Estimation Efficiency

Term	Fractional Increase in CI Length	Relative Std Error of Estimate
Intercept	1.236	0.577
Silica	0.369	0.354
Silane	0.369	0.354
Sulfur	0.369	0.354
Silica*Silane	0.936	0.5
Silica*Sulfur	0.936	0.5
Sulfur*Silane	0.936	0.5
Silica*Silica	1.016	0.52
Silane*Silane	1.016	0.52
Sulfur*Sulfur	1.016	0.52

Fractional Increase in CI Length

The Fractional Increase in CI Length compares the length of a parameter's confidence interval as given by the current design to the length of such an interval given an ideal design:

- The length of the ideal confidence interval for the parameter is subtracted from the length of its actual confidence interval.
- This difference is then divided by the length of the ideal confidence interval.

For an orthogonal D-optimal design, the fractional increase is zero. In selecting a design, you would like the fractional increase in confidence interval length to be as small as possible.

The Ideal Design

The covariance matrix for the ordinary least squares estimator is $\sigma^2(\mathbf{X}'\mathbf{X})^{-1}$. The diagonal elements of $(\mathbf{X}'\mathbf{X})^{-1}$ are the *relative variances* (the variances divided by σ^2) of the parameter estimates. For two-level designs and using the effects coding convention (see "Coding"), the minimum value of the relative variance for any parameter estimate is $1/n$, where n is the number of runs. This occurs when all the effects for the design are orthogonal and the design is D-optimal.

Let $\hat{\beta}$ denote the vector of parameter estimates. The *ideal design*, which may not exist, is a design whose covariance matrix is given as follows:

$$\text{Var}(\hat{\beta}) = (\sigma^2/n)\mathbf{I}_n$$

where $\mathbf{I}_n$ is the n by n identity matrix and σ is the standard deviation of the response.

If an orthogonal D-optimal design exists, it is the ideal design. However, the definition above extends the idea of an ideal design to situations where a design that is both orthogonal and D-optimal does not exist.

The definition is also appropriate for designs with multi-level categorical factors. The orthogonal coding used for categorical factors allows such designs to have the ideal covariance matrix. For a Custom Design, you can view the coding matrix by selecting Save X Matrix from the options in the Custom Design window, making the design table, and looking at the script Model Matrix that is saved to the design table.

Fractional Increase in Length of Confidence Interval

Note that, in the ideal design, the standard error for the parameter estimates would be given as follows:

$$SE_{Ideal}(\hat{\beta}) = (\sigma/\sqrt{n})\mathbf{I}_n$$

The length of a confidence interval is determined by the standard error. The Fractional Increase in Confidence Interval Length is the difference between the standard error of the given design and the standard error of the ideal design, divided by the standard error of the ideal design.

Specifically, for the i^{th} parameter estimate, the Fractional Increase in Confidence Interval Length is defined as follows:

$$FI = \frac{\sigma\sqrt{(\mathbf{X}'\mathbf{X}_{ii})^{-1}} - (\sigma/(\sqrt{n}))}{(\sigma/\sqrt{n})} = \sqrt{n(\mathbf{X}'\mathbf{X}_{ii})^{-1}} - 1$$

where

σ^2 is the unknown response variance,

X is the model matrix for the given design, defined in "The Alias Matrix",

$(\mathbf{X}'\mathbf{X})^{-1}_{ii}$ is the i^{th} diagonal entry of $(\mathbf{X}'\mathbf{X})^{-1}$, and

n is the number of runs.

Relative Std Error of Estimate

The Relative Std Error of Estimate gives the ratio of the standard deviation of a parameter's estimate to the error standard deviation. These values indicate how large the standard errors of the model's parameter estimates are, relative to the error standard deviation. For the i^{th} parameter estimate, the Relative Std Error of Estimate is defined as follows:

$$SE = \sqrt{(\mathbf{X}'\mathbf{X})^{-1}_{ii}}$$

where

X is the model matrix defined in "The Alias Matrix", and

$(\mathbf{X}'\mathbf{X})^{-1}_{ii}$ is the i^{th} diagonal entry of $(\mathbf{X}'\mathbf{X})^{-1}$.

Alias Matrix

The Alias Matrix addresses the issue of how terms that are not included in the model affect the estimation of the model terms, if they are indeed active. In the Alias Terms outline, you list potentially active effects that are not in your assumed model but that might bias the estimates of model terms. The Alias Matrix entries represent the degree of bias imparted to model parameters by the Alias Terms effects. See "Alias Terms".

The rows of the Alias Matrix are the terms corresponding to the model effects listed in the Model outline. The columns are terms corresponding to effects listed in the Alias Terms outline. The entry in a given row and column indicates the degree to which the alias term affects the parameter estimate corresponding to the model term.

In evaluating your design, you ideally want one of two situations to occur relative to any entry in the Alias Matrix. Either the entry is small or, if it is not small, the effect of the alias term is small so that the bias will be small. If you suspect that the alias term may have a substantial effect, then that term should be included in the model or you should consider an alias optimal design.

For more information about the computation of the Alias Matrix, see "The Alias Matrix". See also Lekivetz, R. (2014).

Note the following:

- If the design is orthogonal for the assumed model, then the correlations in the Alias Matrix correspond to the absolute correlations in the Color Map on Correlations.
- Depending on the complexity of the design, it is possible to have alias matrix entries greater than 1 or less than -1.

Alias Matrix Example

Consider the Coffee Data.jmp sample data table, located in the Design Experiment folder. The design assumes a main effects model. You can see this by running the Model script in the data table. Consequently, in the Evaluate Design window's Model outline, only the Intercept and five main effects appear. The Alias Terms outline contains the two-way interactions.

Figure 15.23 Alias Matrix for Coffee Data.jmp

Alias Matrix

Effect	Grind*Temperature	Grind*Time	Grind*Charge	Temperature*Time	Temperature*Charge	Time*Charge
Intercept	0	0	0	0	0	0
Grind	0	0	0	0.333	-0.33	-0.33
Temperature	0	0.333	-0.33	0	0	-0.33
Time	0.333	0	-0.33	0	-0.33	0
Charge	-0.33	-0.33	0	-0.33	0	0
Station 1	-0.41	0	0	0	0	0.816
Station 2	0.707	0	0	0	0	0

The Alias Matrix shows the Model terms in the first column defining the rows. The two-way interactions in the Alias Terms are listed across the top, defining the columns. Consider the model effect Temperature for example. If the Grind*Time interaction is the only active two-way interaction, the estimate for the coefficient of Temperature is biased by 0.333 times the true value of the Grind*Time effect. If other interactions are active, then the value in the Alias Matrix indicates the additional amount of bias incurred by the Temperature coefficient estimate.

Color Map on Correlations

The Color Map on Correlations shows the absolute value of the correlation between any two effects that appear in either the Model or the Alias Terms outline. The cells of the color map are identified above the map. There is a cell for each effect in the Model outline and a cell for each effect in the Alias Terms outline. Smaller values are desired.

By default, the absolute magnitudes of the correlations are represented by a white to gray to black intensity color theme. In general terms, the color map for a good design shows a lot of white off the diagonal, indicating orthogonality or small correlations between distinct terms. Large absolute correlations among effects inflate the standard errors of estimates.

To see the absolute value of the correlation between two effects, hover over the corresponding cell. To change the color scale, click the Color Map on Correlations red triangle and select **Blue to Gray to Red**. For a custom color scale, right-click in the plot and select **Color Theme**. To save a table of the correlations, right-click to the right of the plot below the legend and select **Table of Correlations**.

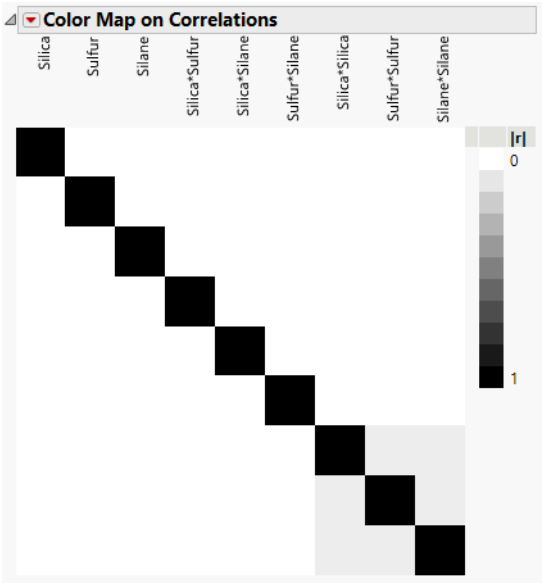
Color Map Example

Figure 15.24 shows the Color Map on Correlations for the Bounce Data.jmp sample data table, found in the Design Experiment folder. The black coloring indicates absolute correlations of one. Note that there are black cells on the diagonal, showing correlations of model terms with themselves.

All other cells are either white or gray. The gray squares correspond to correlations between quadratic terms. To see this, hover over each of the gray squares. The absolute correlations of quadratic terms with each other are small, 0.0714.

From the perspective of correlation, this is a good design. When effects are highly correlated, it is more difficult to determine which is responsible for an effect on the response.

Figure 15.24 Color Map on Correlations



Tip: To save a table of the correlations, right-click to the right of the plot below the legend and select **Table of Correlations**.

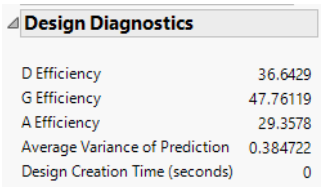
Design Diagnostics

The Design Diagnostics outline shows *D*-, *G*-, and *A*-efficiencies and the average variance of prediction. These diagnostics are not shown for designs that include factors with Changes set to Hard or Very Hard or effects with Estimability designated as If Possible.

When Design Diagnostics is accessed from a DOE platform other than Evaluate Design, the Design Creation Time gives the amount of time required to create the design. When Design Diagnostics is accessed from the Evaluate Design platform, Design Creation Time gives the amount of time required for the Evaluate Design platform to calculate results.

Figure 15.25 shows the Design Diagnostics outline for the Bounce Data.jmp sample data table, found in the Design Experiment folder.

Figure 15.25 Design Diagnostics Outline



Caution: The efficiency measures should not be interpreted on their own. But they can be used to compare designs. Given two designs, the one with the higher efficiency measure is better. While the maximum efficiency is 100 for any criterion, an efficiency of 100% is impossible for many design problems.

Notation

The descriptions of the efficiency measures given below use the following notation:

- **X** is the model matrix
- n is the number of runs in the design
- p is the number of terms, including the intercept, in the model
- $Var(\hat{y}|\underset{\sim}{\mathbf{x}})$ is the relative prediction variance at the point $\underset{\sim}{\mathbf{x}}$. See "Relative Prediction Variance".
- $Var(\hat{y}|\underset{\sim}{\mathbf{x}})_{max}$ is the maximum relative prediction variance over the design region

D Efficiency

The efficiency of the design to that of an ideal orthogonal design in terms of the D-optimality criterion. A design is D-optimal if it minimizes the volume of the joint confidence region for the vector of regression coefficients:

$$D\text{-efficiency} = 100\left(\frac{1}{n}|\mathbf{X}'\mathbf{X}|^{1/p}\right)$$

G Efficiency

The efficiency of the design to that of an ideal orthogonal design in terms of the G-optimality criterion. A design is G-optimal if it minimizes the maximum prediction variance over the design region:

$$G\text{-efficiency} = 100p/(nVar(\hat{y}|\underset{\sim}{\mathbf{x}})_{max})$$

Letting D denote the design region,

$$Var(\hat{y}|\underset{\sim}{\mathbf{x}})_{max} = \underset{\underset{\sim}{\mathbf{x}} \text{ in D}}{maximum}[\underset{\sim}{\mathbf{x}}'(\mathbf{X}'\mathbf{X})^{-1}\underset{\sim}{\mathbf{x}}]$$

Note: G-Efficiency is calculated using Monte Carlo sampling of the design space. Therefore, calculations for the same design may vary slightly.

A Efficiency

The efficiency of the design to that of an ideal orthogonal design in terms of the A-optimality criterion. A design is A-optimal if it minimizes the sum of the variances of the regression coefficients:

$$A\text{-efficiency} = 100p/(n\text{Trace}(\mathbf{X}'\mathbf{X})^{-1})$$

Average Variance of Prediction

At a point $\underset{\sim}{\mathbf{x}}$ in the design space, the relative prediction variance is defined as:

$$Var(\hat{y}|\underset{\sim}{\mathbf{x}}) = \underset{\sim}{\mathbf{x}}'(\mathbf{X}'\mathbf{X})^{-1}\underset{\sim}{\mathbf{x}}$$

where this is the prediction variance divided by the error variance. For more information about the calculation, see Section 4.3.5 in Goos and Jones (2011).

Note: If the design region is constrained by linear constraints or disallowed combinations, the average variance of prediction is calculated from the prediction variance values found in the Fraction of Design Space plot.

Design Creation Time

Design Creation Time gives the amount of time required for the Evaluate Design platform to calculate results.

Evaluate Design Options

The Evaluate Design red triangle menu contains the following options:

Advanced Options > Split Plot Variance Ratio Specify the ratio of the variance of the random whole plot and the subplot variance (if present) to the error variance. Before setting this value, you must define a hard-to-change factor for your split-plot design, or hard and very-hard-to-change factors for your split-split-plot design. Then you can enter one or two positive numbers for the variance ratios, depending on whether you have specified a split-plot or a split-split-plot design.

Advanced Options > Set Delta for Power Specify a value for the difference you want to detect that is applied to Anticipated Coefficients in the Power Analysis report. The Anticipated Coefficient values are set to Delta/2 for continuous effects. For categorical effects, they are alternating values of Delta/2 and –Delta/2. For more information about power analysis, see "Power Analysis".

By default, Delta is set to two. Consequently, the Anticipated Coefficient default values are 1 for continuous effects and alternating values of 1 and –1 for categorical effects. The default values that are entered as Anticipated Coefficients when Delta is 2 ensure these properties:

- The power calculation for a numeric effect assumes a change of Delta in the response mean due to linear main effects as the factor changes from the lowest setting to the highest setting in the design region.
- The power calculation for the parameter associated with a two-level categorical factor assumes a change of Delta in the response mean across the levels of the factor.
- The power calculation for a categorical effect with more than two levels is based on the multiple degree of freedom *F* test for the null hypothesis that all levels have the same response mean. Power is calculated at the values of the response means that are

determined by the Anticipated Coefficients. Various configurations of the Anticipated Coefficients can define a difference in levels as large as Delta. However, the power values for such configurations will differ based on the Anticipated Coefficients for the other levels.

Save Script to Script Window Creates a script that reproduces the Evaluate Design window and places it in an open script window.

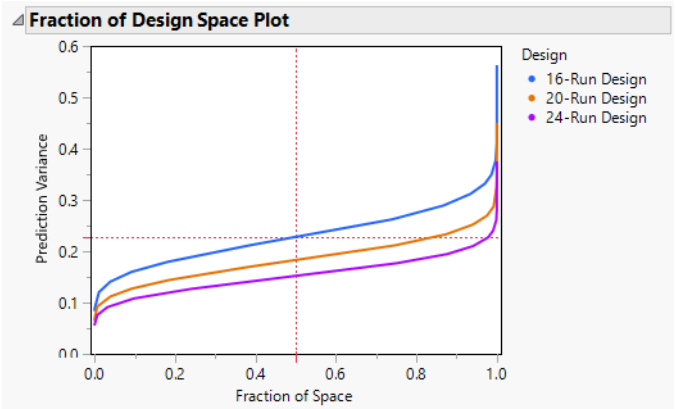
Chapter 16

Compare Designs

Compare and Evaluate Designs Simultaneously

The Compare Designs platform compares up to four designs to a reference design. Use to explore, evaluate, and compare design performance. Diagnostics show how the designs perform relative to each other and how they perform in an absolute sense. To compare designs relative to your specific needs, you can change the terms in the assumed model and in the alias terms list.

Figure 16.1 Comparing Three Designs with Different Run Sizes



Contents

Overview of the Compare Designs Platform 473
Examples of Comparing Designs 473
Designs of Same Run Size 473
Designs of Different Run Sizes 478
Split Plot Designs with Different Numbers of Whole Plots 482
Launch the Compare Designs Platform 486
Compare Designs Window: Specify Model and Alias Terms 488
Reference Design 488
Factors 488
Model 488
Alias Terms 489
Compare Designs Window: Design Evaluation 489
Power Analysis 490
Prediction Variance Profile 493
Fraction of Design Space Plot 495
Relative Estimation Efficiency 496
Alias Matrix Summary 498
Absolute Correlations 500
Design Diagnostics 502
Compare Designs Options 504

Overview of the Compare Designs Platform

The Compare Designs platform, which is an extension of the Evaluate Design platform, enables you to easily compare up to four designs to a reference design. To compare the performance of one or two designs relative to another, you select a reference design that is treated as the base design. You can specify effects in the Model outline, and effects of interest in the Alias Terms outline.

The Design Evaluation report shows diagnostic results and plots covering these areas:

- Power analysis
- Prediction variance
- Fraction of design space
- Relative estimation efficiency
- Alias matrix diagnostics
- Correlations among effects (including confounding)
- Relative efficiency measures for the overall designs

Examples of Comparing Designs

- "Designs of Same Run Size"
- "Designs of Different Run Sizes"
- "Split Plot Designs with Different Numbers of Whole Plots"

Designs of Same Run Size

In this example, you compare two designs for six factors each with 13 runs. One is a 12-run Plackett-Burman (PB) design augmented with a single center point. The other is a Definitive Screening Design (DSD).

Comparison in Terms of Main Effects Only

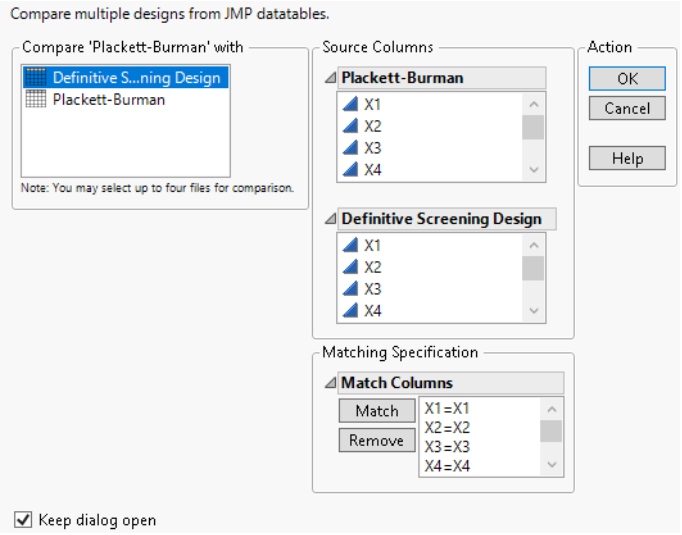
First, compare the two designs assuming that the model to be estimated contains only the main effects.

1. Select **Help > Sample Data**, click **Open the Sample Scripts Directory**, and select Compare Same Run Size.jsl.
2. Right-click in the script window and select **Run Script**.

Two 13-run design tables are constructed: Definitive Screening Design and Plackett-Burman. You want to compare these two designs. Because the Plackett-Burman table is active, it is the reference design to which you compare the DSD.

3. In the Plackett-Burman data table, select **DOE > Design Diagnostics > Compare Designs**.
4. Select Definitive Screening Design from the **Compare 'Plackett-Burman' with** list.
5. Select X1 through X6 in the Plackett-Burman panel and in the Definitive Screening Design panel.
6. Open the Match Columns outline and click **Match**.

Figure 16.2 Launch Window with Matched Columns



This defines the correspondence between the factors in your two designs.

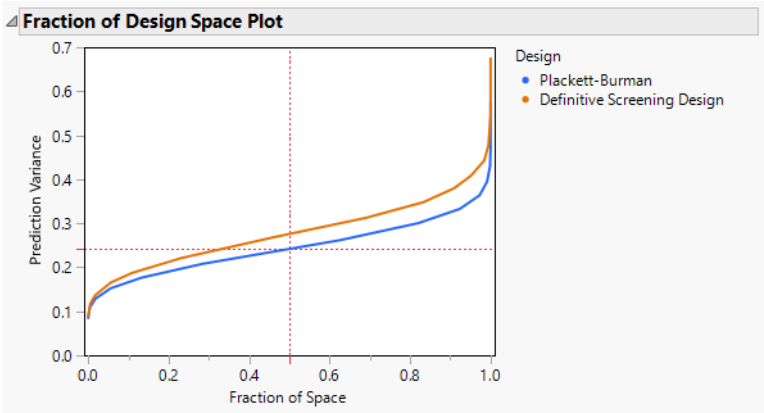
7. Click **OK**.

The reference design is the Plackett-Burman design. In the Design Evaluation outline, comparison metrics compare the PB to the DSD. The designs are compared relative to power, prediction variance, estimation efficiency, aliasing, and design efficiency measures.

Figure 16.3 Power Analysis for PB and DSD Comparison

In terms of power, prediction variance, and estimation efficiency, the PB design outperforms the DSD. Figure 16.3 shows the Power Analysis report with the default settings for the significance level, Anticipated RMSE, and coefficients. For tests for the main effects, the PB design has higher power than does the DSD.

Figure 16.4 Fraction of Design Space Plot for PB and DSD Comparison



The Fraction of Design Space plot indicates that the PB design has smaller prediction variance than the DSD over the entire design space.

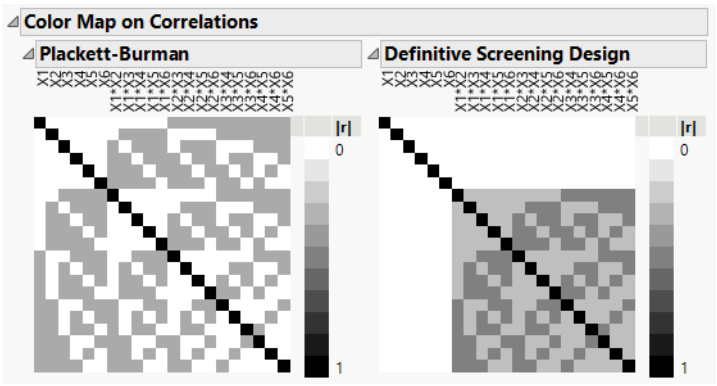
You conclude that, if you suspect that only main effects are active, the PB design is preferable.

Comparison in Terms of Two-Way Interactions

Now suppose you suspect that some two-way interactions might be active. The analysis below shows that if those two-way interactions are actually active, then the PB design might be less desirable than the DSD.

1. In the Absolute Correlations report, open the Color Map on Correlations report and the color map reports under it.

Figure 16.5 Color Maps for PB and DSD Comparison

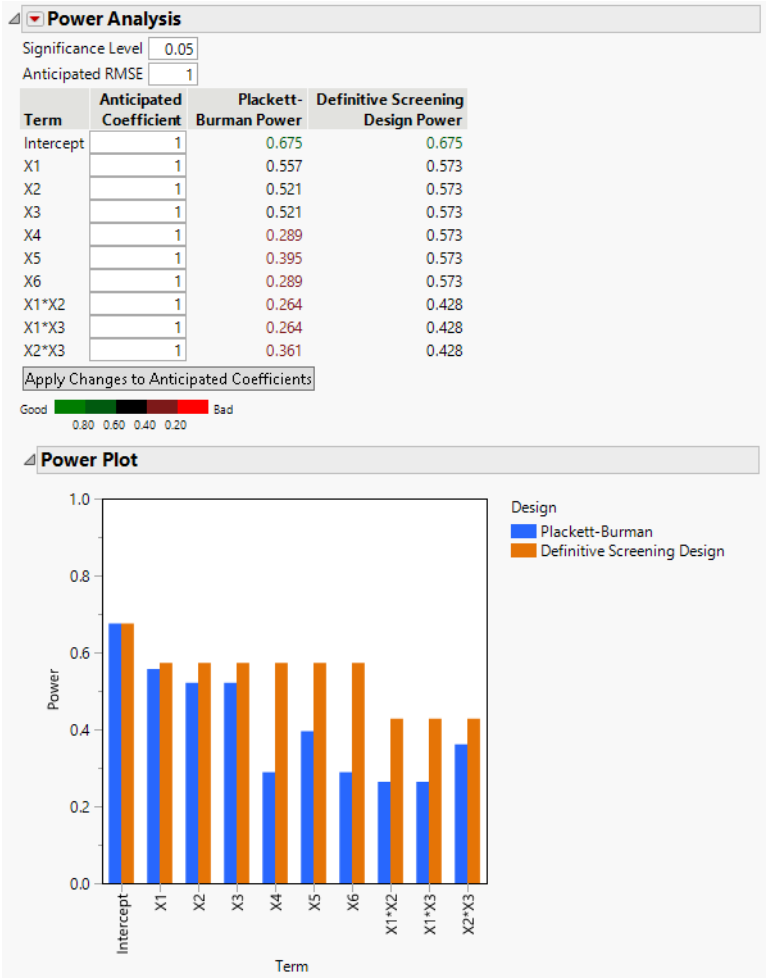


The Color Map on Correlations plots show that the PB design aliases main effects with two-way interactions. In contrast, the DSD does not alias main effects with two-way interactions.

To gain more insight on how the designs compare if some two-way interactions are active, add two-way interactions in the Model outline.

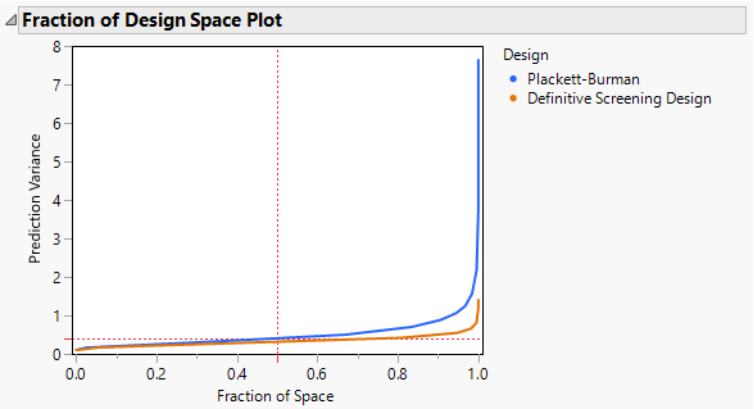
2. In the Factors outline, select X1 through X3.
3. In the Model outline, select **Interactions > 2nd**.

Figure 16.6 Power Analysis for PB and DSD Comparison with Interactions



The Term list shows the three two-way interactions. If these two-way interactions are active, then the DSD has better performance in terms of power across all effects than the PB.

Figure 16.7 Prediction Variance for PB and DSD Comparison with Interactions



The DSD also outperforms the PB design in terms of prediction variance with the three interactions in the model. You can explore the other reports to see that the DSD is preferred when there are potentially active interactions.

Designs of Different Run Sizes

In this example, compare three designs with run sizes 16, 20, and 24. The designs are constructed for main effect models. Use the Compare Designs platform to determine whether the potential benefits of using a larger run size are worth the additional cost in resources.

1. Select **Help > Sample Data**, click **Open the Sample Scripts Directory**, and select Compare Three Run Sizes.jsl.
2. Right-click in the script window and select **Run Script**.

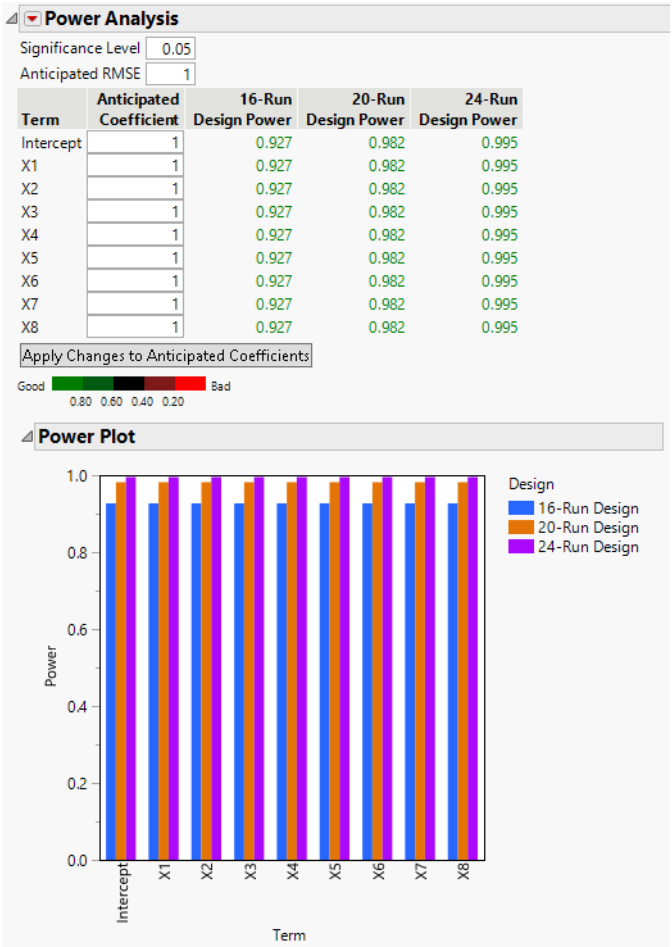
   Three design tables are constructed using Custom Design, with only main effects as entries in the Model outline:

   – 16-Run Design
   – 20-Run Design
   – 24-Run Design

   You want to compare these three designs. Notice that the 16-Run Design table is active.
3. In the 16-Run Design table, select **DOE > Design Diagnostics > Compare Designs**.
4. From the **Compare '16-Run Design' with** list, select 20-Run Design and 24-Run Design.

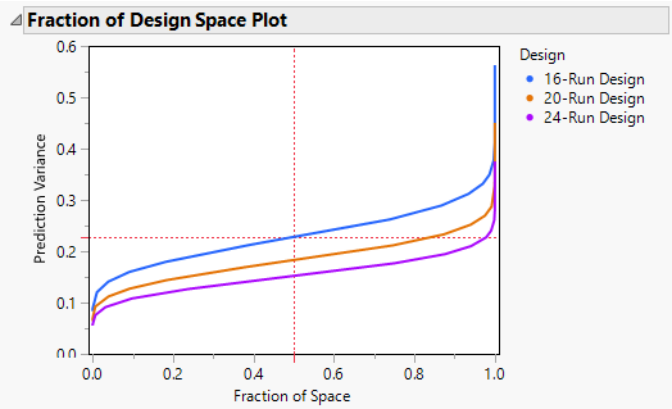
   Panels for each of these designs are added to the launch window. JMP automatically matches the columns in the order in which they appear in the three design tables.
5. Click **OK**.

Figure 16.8 Power Analysis Comparison



All three designs have high power for detecting main effects if the coefficients are on the order of the Anticipated RMSE.

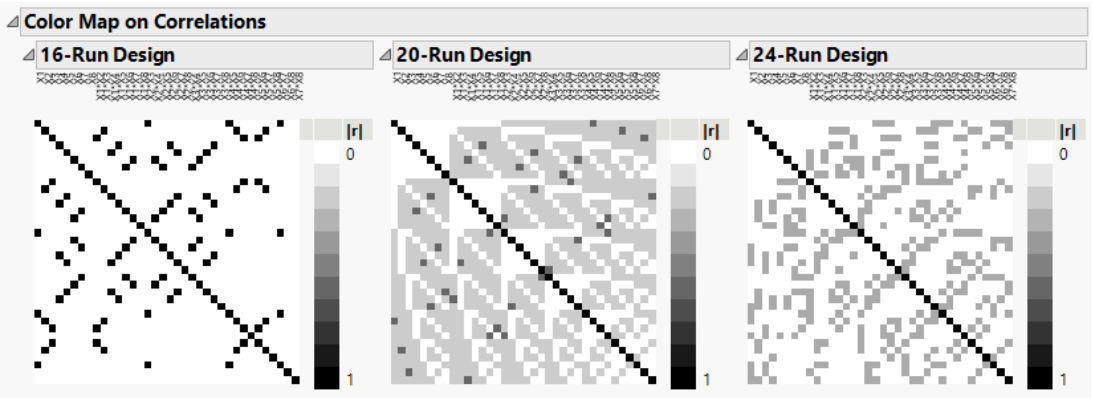
Figure 16.9 Fraction of Design Space Comparison



As expected, the 24-run design is superior to the other two designs in terms of prediction variance over the entire design space. The 20-run design is superior to the 16-run design.

6. In the Absolute Correlations report, open the Color Map on Correlations report and the three color map reports under it.

Figure 16.10 Color Map on Correlations Comparison



For the 16-run design, the Color Map on Correlations indicates that there is confounding of some main effects with some two-factor interactions, and confounding of two-factor interactions.

For the 20-run design, the Color Map on Correlations indicates that there are some large correlations between some main effects and some two-factor interactions, and between some two-factor interactions.

The 24-run design shows only moderate correlations between main effects and two-factor interactions, and between two-factor interactions.

Figure 16.11 Absolute Correlations Comparison

Absolute Correlations

Model x Model	Average Correlation	Number of Confoundings	Number of Pairs
16-Run Design	0.000	0	28
20-Run Design	0.000	0	28
24-Run Design	0.000	0	28
Model x Alias	**Average Correlation**	**Number of Confoundings**	**Number of Pairs**
16-Run Design	0.094	21	224
20-Run Design	0.166	0	224
24-Run Design	0.098	0	224
Alias x Alias	**Average Correlation**	**Number of Confoundings**	**Number of Pairs**
16-Run Design	0.056	21	378
20-Run Design	0.121	0	378
24-Run Design	0.085	0	378

Good Bad
0.10 0.30 0.50 0.70

The Absolute Correlations table summarizes the information shown in the Color Maps on Correlations. Recall that the model for all three designs consists of only main effects and the Alias Matrix contains two-factor interactions.

For the 16-run design, the Model x Alias portion of the table indicates that there are nine confoundings of main effects with two-factor interactions. The Alias x Alias portion indicates that six two-factor interactions are confounded.

Figure 16.12 Design Diagnostics Comparison

Design Diagnostics

	Efficiency of 16-Run Design Relative to 20-Run Design	Efficiency of 16-Run Design Relative to 24-Run Design
D-efficiency	0.800	0.667
G-efficiency	0.800	0.667
A-efficiency	0.800	0.667
I-efficiency	0.800	0.667
Additional Run Size	-4	-8

Good Bad
1.50 1.25 0.80 0.67

The Design Diagnostics report compares the efficiency of the 16-run design to both the 20-run and 24-run designs in terms of several efficiency measures. Relative efficiency values that exceed 1 indicate that the reference design is preferable for the given measure. Values less than 1 indicate that the design being compared to the reference design is preferable. The 16-run design has lower efficiency than the other two designs across all metrics, indicating that the larger designs are preferable.

Split Plot Designs with Different Numbers of Whole Plots

In this example, compare two split-plot designs with different numbers of whole plots. The designs are for three factors:

- A continuous hard-to-change factor
- A continuous easy-to-change factor
- A three-level categorical easy-to-change factor

The designs include all two-factor interactions in the assumed model. You can afford 20 runs and want to compare using 4 or 8 whole plots.

Launch Compare Designs

1. Select **Help > Sample Data**, click **Open the Sample Scripts Directory**, and select Compare Split Plots.jsl.
2. Right-click in the script window and select **Run Script**.

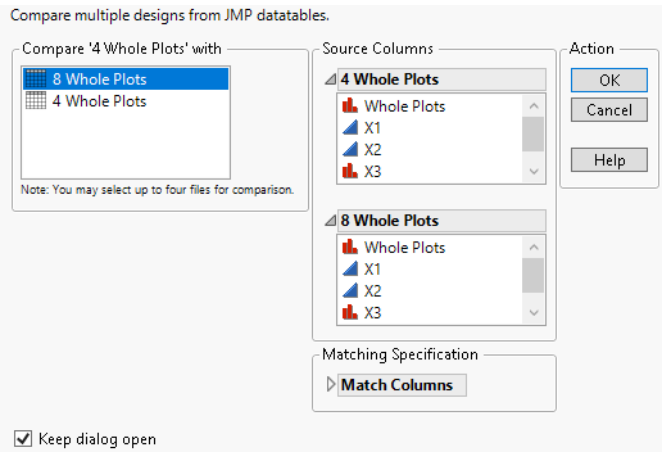
   Two design tables are constructed using Custom Design:

   – 4 Whole Plots
   – 8 Whole Plots

   You want to compare these two designs. Notice that the 4 Whole Plots table is active.
3. In the 4 Whole Plots table, select **DOE > Design Diagnostics > Compare Designs**.
4. From the **Compare '4 Whole Plots' with** list, select 8 Whole Plots.

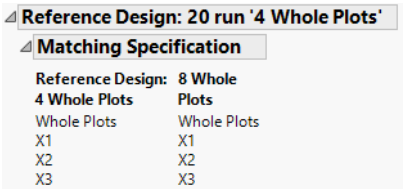
   A panel for this design is added to the launch window. JMP automatically matches the columns in the order in which they appear in the two design tables.

Figure 16.13 Completed Launch Window



5. Click **OK**.
6. Open the **Matching Specification** outline under Reference Design: 20 run '4 Whole Plots'.

Figure 16.14 Matching Specification for Split-Plot Designs

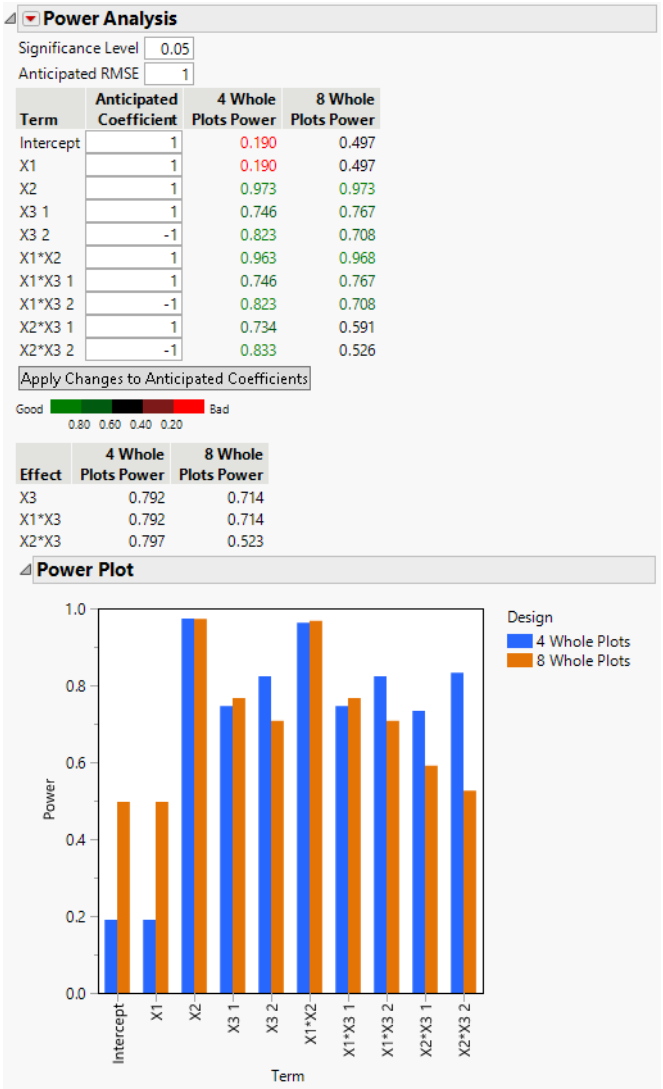


Notice that the Whole Plots column is entered as part of the design. This is necessary because Compare Designs needs to know the whole plot structure.

Examine the Report

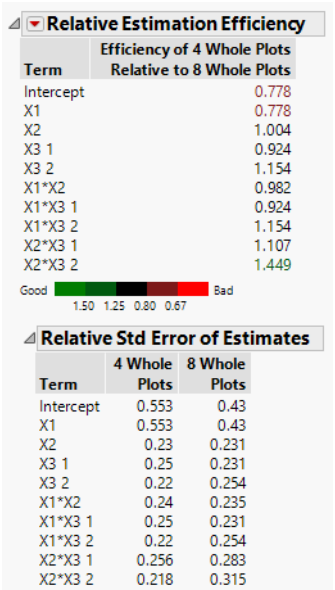
The Design Evaluation report provides various diagnostics that compare the two designs.

Figure 16.15 Power Analysis for Two Split-Plot Designs



The Power Analysis report shows that the power for the whole-plot factor, X1, is much smaller for the four whole-plot design (0.19) than for the eight whole-plot design (0.497). However, the four whole-plot design has higher power to detect split-plot effects, especially the interaction of the two split-plot factors, X2*X3 (0.797 compared to 0.523). Notice that the power for the combined effect X2*X3 is given under the color bar and legend.

Figure 16.16 Relative Estimation Efficiency Comparing Split-Plot Designs



Relative Estimation Efficiency

Term	Efficiency of 4 Whole Plots Relative to 8 Whole Plots
Intercept	0.778
X1	0.778
X2	1.004
X3 1	0.924
X3 2	1.154
X1*X2	0.982
X1*X3 1	0.924
X1*X3 2	1.154
X2*X3 1	1.107
X2*X3 2	1.449

Good Bad
1.50 1.25 0.80 0.67

Relative Std Error of Estimates

Term	4 Whole Plots	8 Whole Plots
Intercept	0.553	0.43
X1	0.553	0.43
X2	0.23	0.231
X3 1	0.25	0.231
X3 2	0.22	0.254
X1*X2	0.24	0.235
X1*X3 1	0.25	0.231
X1*X3 2	0.22	0.254
X2*X3 1	0.256	0.283
X2*X3 2	0.218	0.315

The Relative Estimation Efficiency report shows the relative estimation efficiency for X1 to be 0.778. This indicates that the standard error for X1 is notably larger for the four whole-plot design than for the eight whole-plot design.

Open the Relative Std Error of Estimates report. You can see that the relative standard error for X1 in the four whole-plot design is 0.553, compared to the eight whole-plot error of 0.43.

In the Relative Estimation Efficiency report, the relative estimation efficiency for X2*X3 2 is 1.449, indicating that the standard error for the parameter associated with X2*X3 2 is notably larger for the eight whole-plot design than for the four whole-plot design.

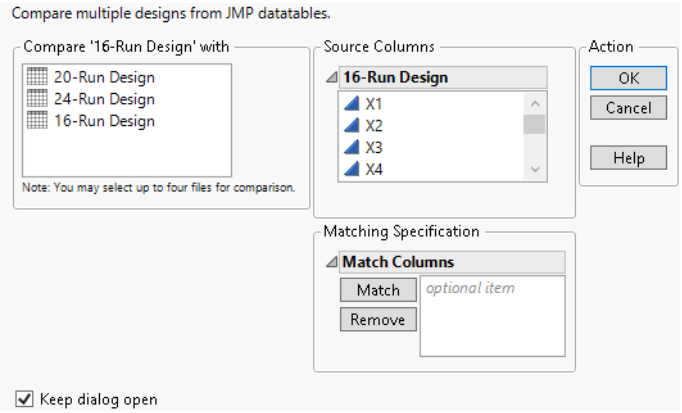
The Power Analysis and the Relative Estimation Efficiency reports indicate that the choice of designs revolves around the importance of detecting the whole plot effect X1. The eight whole-plots design gives you a better chance of detecting a whole plot effect. The four whole-plots design is somewhat better for detecting split-plot effects involving the categorical variable.

Launch the Compare Designs Platform

Launch the Compare Designs platform by selecting **DOE > Design Diagnostics > Compare Designs**. All open data tables appear in the list at the left. The active data table and its columns appear in a Source Columns panel. The design in the initial Source Columns panel is the *reference design*, namely, the design to which other designs are compared. When you add designs to compare to the reference design, their columns appear in panels under the reference design panel.

Figure 16.17 shows the launch window for the three designs in "Designs of Different Run Sizes".

Figure 16.17 Compare Designs Launch Window



Design Table Selection

Select up to four design tables from the list on the left.

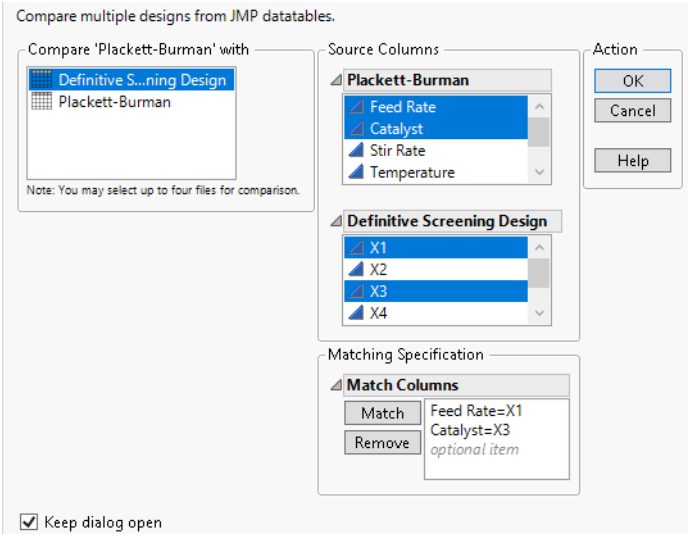
- To compare multiple designs to the reference design, you must select their design tables simultaneously from the list on the left.
- To replace a design (or designs) in the Source Columns list, select the desired table (or tables) from the list at the left. The design table (or tables) under the reference design table are replaced.

Note: The reference design table can be compared to itself, which can be useful when exploring the assignment of design columns to factors.

Match Columns

Specify which columns in each of the design tables correspond to each other in the Match Columns panel. To match columns, select the columns to match in each of the design table Source Columns lists, and then click Match.

Figure 16.18 Selection of Columns for Matching



- To match single columns in each list, select the single column in each list, and then click Match.
- To match several columns that appear in the correct matching order in each list, select them in each list. Click the Match button. They are matched in their list order (Figure 16.18). In this example, Feed Rate is matched with X1, and Catalyst is matched with X3.
- If the lists contain the same numbers of columns and your desired match order is their order of appearance in the lists, you do not have to click Match. When you click OK to run the launch window, JMP matches the columns automatically in their order of appearance. You can review the matching in the report's Matching Specification outline.

Compare Designs Window: Specify Model and Alias Terms

The Compare Designs window consists of two sets of outlines:

- Specify which effects are in the model and which effects are potentially active using the Factors, Model, and Alias Terms outlines.
- Compare the designs using the diagnostics in the Design Evaluation outlines. Changes that you make in the Model and Alias Terms outlines are updated in the Design Evaluation report.

The Compare Designs report uses the column names from the reference design.

This section describes the Reference Design, Factors, Model, and Alias Terms outlines. See "Compare Designs Window: Design Evaluation" for a description of the Design Evaluation outlines.

Reference Design

The name of the window for the reference design appears in the outline title. The Matching Specification outline lists the specifications that you entered in the launch window.

Factors

Use the Factors outline to add effects to the Model and Alias Terms lists.

The Factors outline lists the factors, using the column names from the reference design, and coded values. Because they are not factors, whole plot and subplot columns do not appear in the Factors outline. However, they are required for the analysis.

Model

Add or remove effects to compare your designs for the effects that you believe should be in the model. The Model outline initially lists effects that are in the Model script of the reference design table and that are estimated by all designs being compared. If there is no Model script in the reference design table, the Model outline shows only the main effects that can be estimated by all designs being compared. For more information about how to add and remove effects, see "Model".

Note: If any of the designs are supersaturated, meaning that the number of parameters to be estimated exceeds the number of runs, the Model outline lists only a set of effects that can be estimated.

Alias Terms

Add or remove effects to compare your designs for effects that might be active. The Alias Terms outline initially contains all two-factor interactions that are not in the Model outline. The effects in this outline impact the calculations in the Alias Matrix Summary and Absolute Correlations outline. See "Alias Matrix Summary" and "Absolute Correlations".

For more information about how to add and remove effects, see "Alias Terms".

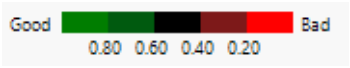
Compare Designs Window: Design Evaluation

- "Power Analysis"
- "Prediction Variance Profile"
- "Fraction of Design Space Plot"
- "Relative Estimation Efficiency"
- "Alias Matrix Summary"
- "Absolute Correlations"
- "Design Diagnostics"

Color Dashboard

Several of the Design Evaluation outlines show values colored according to a color bar. The colors are applied to diagnostic measures and they help you see which values (and designs) reflect good or bad behavior. You can edit the legend values to apply colors that reflect your definitions of good and bad behavior.

Figure 16.19 Color Dashboard



You can modify the color bar by selecting these two options in the red triangle menu for the outline or by right-clicking the color bar:

Show Legend Values Shows or hides the values that appear under the color bar.

Edit Legend Values Specify the values that define the colors.

Power Analysis

Power is the probability of detecting an active effect of a given size. The Power Analysis report helps you evaluate and compare the ability of your designs to detect effects of practical importance. For each of your designs, the Power Analysis report calculates the power of tests for the effects in the Model outline.

The Power Analysis report gives the power of tests for individual model parameters and for whole effects. It also provides a Power Plot and a Power versus Sample Size plot.

Power depends on the number of runs, the significance level, and the estimated error variation. For more information about how power is calculated, see "Power Calculations".

Figure 16.20 Power Analysis Outline for Three Designs

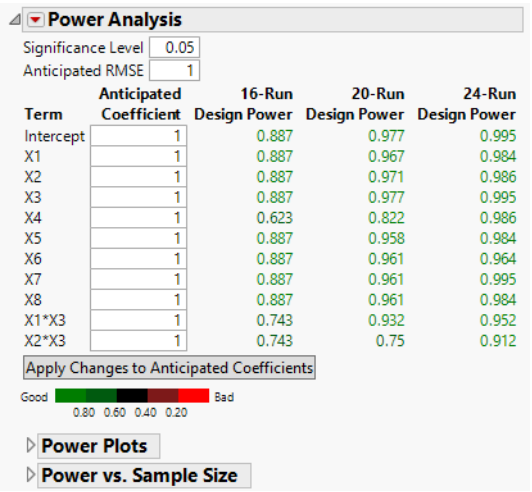


Figure 16.20 shows the Power Analysis outline for the three designs constructed in "Designs of Different Run Sizes". Two two-way interactions have been added to the Model outline.

Power Analysis Report

When you specify values for the Significance Level and Anticipated RMSE, they are used to calculate the power of the tests for the model parameters. Enter coefficient values that reflect differences that you want to detect as Anticipated Coefficients. To update the results for all designs, click Apply Changes to Anticipated Coefficients.

Significance Level The probability of rejecting the hypothesis of no effect, if it is true. The power calculations update immediately when you enter a value.

Anticipated RMSE An estimate of the square root of the error variation. The power calculations update immediately when you enter a value.

The power values are colored according to a color gradient that appears under the Apply Changes to Anticipated Coefficients button. You can control the color legend using the options in the Power Analysis red triangle menu. See "Color Dashboard".

For more information about the Power Plots, see "Power Plot".

Note: If the design is supersaturated, meaning that the number of parameters to be estimated exceeds the number of runs, the Power Analysis outline lists only a set of effects that can be estimated.

Tests for Individual Parameters

The Term column contains a list of model terms. For each term, the Anticipated Coefficient column contains a value for that term. The Power value is the power of a test that the coefficient for the term is zero if the true value of the coefficient is given by the Anticipated Coefficient, given the design, and the terms in the Model outline.

Term The model term associated with the coefficient being tested.

Anticipated Coefficient A value for the coefficient associated with the model term. This value is used in the calculations for Power. When you set a new value in the Anticipated Coefficient column, click **Apply Changes to Anticipated Coefficients** to update the Power calculations.

Note: The anticipated coefficients have default values of 1 for continuous effects. They have alternating values of 1 and –1 for categorical effects.

Power The probability of rejecting the null hypothesis of no effect when the true coefficient value is given by the specified Anticipated Coefficient.

- For a coefficient associated with a numeric factor, the change in the mean response from the high to low setting (based on the model) is twice the coefficient value.
- For a coefficient associated with a categorical factor, the change in the mean response (based on the model) across the levels of the factor equals twice the absolute value of the anticipated coefficient. For more information on power for categorical factors, see "Tests for Categorical Effects with More Than Two Levels"

Calculations use the specified Significance Level and Anticipated RMSE. For more information about the power calculation, see "Power for a Single Parameter".

Apply Changes to Anticipated Coefficients When you set a new value in the Anticipated Coefficient column, click **Apply Changes to Anticipated Coefficients** to update the Power values.

Tests for Categorical Effects with More Than Two Levels

If your model contains a categorical effect with more than two levels, then the following columns appear below the Apply Changes to Anticipated Coefficients button:

Effect The categorical effect.

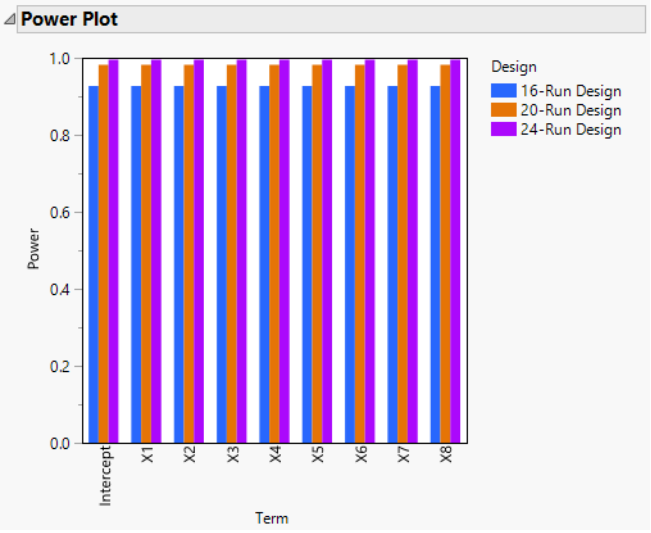
Power The power calculation for a test of no effect. The null hypothesis for the test is that all model parameters corresponding to the effect are zero. The difference to be detected is defined by the values in the Anticipated Coefficient column that correspond to the model terms for the effect. The power calculation reflects the differences in response means determined by the anticipated coefficients.

Calculations use the specified Significance Level and Anticipated RMSE. For more information about the power calculation, see "Power for a Categorical Effect".

Power Plot

The Power Plot shows the power values from the Power Analysis in graphical form. The plot shows the power for each effect and for each design in a side-by-side bar chart.

Figure 16.21 Power Plot for Three Designs

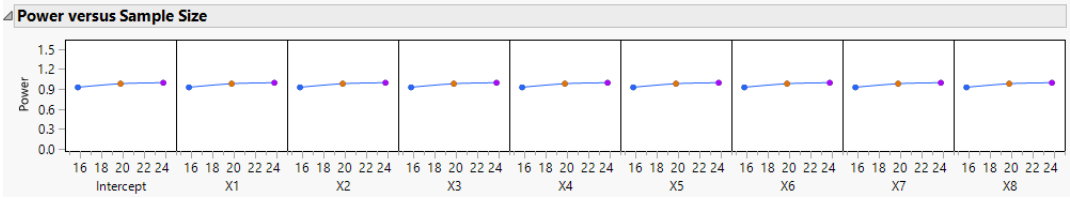


The Power Plot in Figure 16.21 is for the main effects of the three designs constructed in "Designs of Different Run Sizes".

Power versus Sample Size

The Power versus Sample Size plots appears only when the designs that you are comparing differ in run size. The plots enables you to see how sample size affects power for each effect in the model. It conveys the same information as is in the Power Plots graph, but in a different format. The power values at integer sample sizes are connected with line segments.

Figure 16.22 Power versus Sample Size Profiler for Three Designs

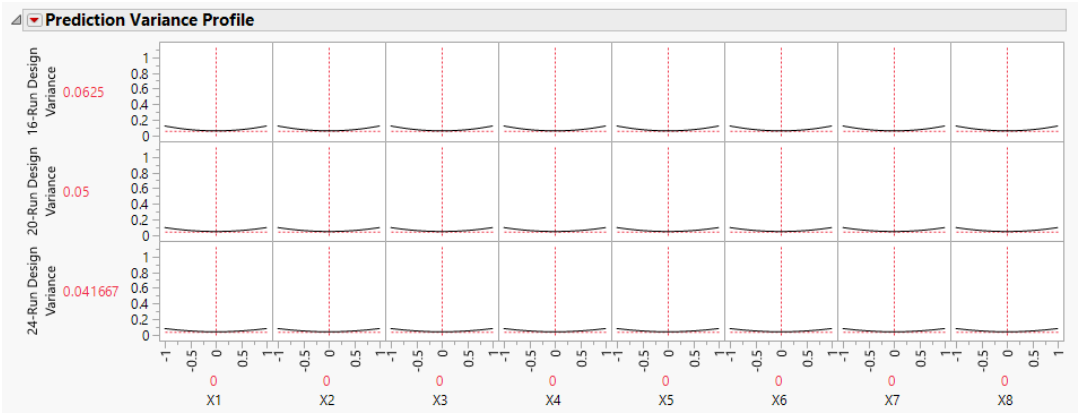


Prediction Variance Profile

The Prediction Variance Profile outline shows profilers of the relative variance of prediction for each design being compared. Each plot shows the relative variance of prediction as a function of each factor at fixed values of the other factors.

To find the maximum value of the relative prediction variance over the design space for the reference design, select the **Maximize Variance** option from the red triangle next to Prediction Variance Profile. See "Maximize Variance".

Figure 16.23 Prediction Variance Profile for Three Designs



The Prediction Variance Profile plot in Figure 16.23 is for the three designs constructed in "Designs of Different Run Sizes". Two two-way interactions, X1*X3 and X2*X3, have been added to the Model outline. The initial value for each continuous factor in the plot is the midpoint of its design settings. The Variance values to the left indicate that, as the number of runs increases, the variance decreases at the center point.

Relative Prediction Variance

For given settings of the factors, the prediction variance is the product of the error variance and a quantity that depends on the design and the factor settings. Before you run your experiment, the error variance is unknown, so the prediction variance is also unknown. However, the ratio of the prediction variance to the error variance is not a function of the error variance. This ratio, called the *relative prediction variance*, depends only on the design and the factor settings. Consequently, the relative variance of prediction can be calculated before acquiring the data. See "Relative Prediction Variance".

After you run your experiment and fit a least squares model, you can estimate the error variance using the mean squared error (MSE) of the model fit. You can estimate the actual variance of prediction at any setting by multiplying the relative variance of prediction at that setting.

Ideally, the prediction variance is small throughout the design space. Generally, the error variance drops as the sample size increases. In comparing designs, a design with lower prediction variance on average is preferable.

Maximize Variance

You can also evaluate a design or compare designs in terms of the maximum relative prediction variance. Select the **Maximize Variance** option from the red triangle next to Prediction Variance Profile. JMP uses a desirability function that maximizes the relative prediction variance. The value of the Variance in the Prediction Variance Profile is the worst (least desirable from a design point of view) value of the relative prediction variance.

Figure 16.24 Prediction Variance Profile Showing Maximum Variance for Three Designs

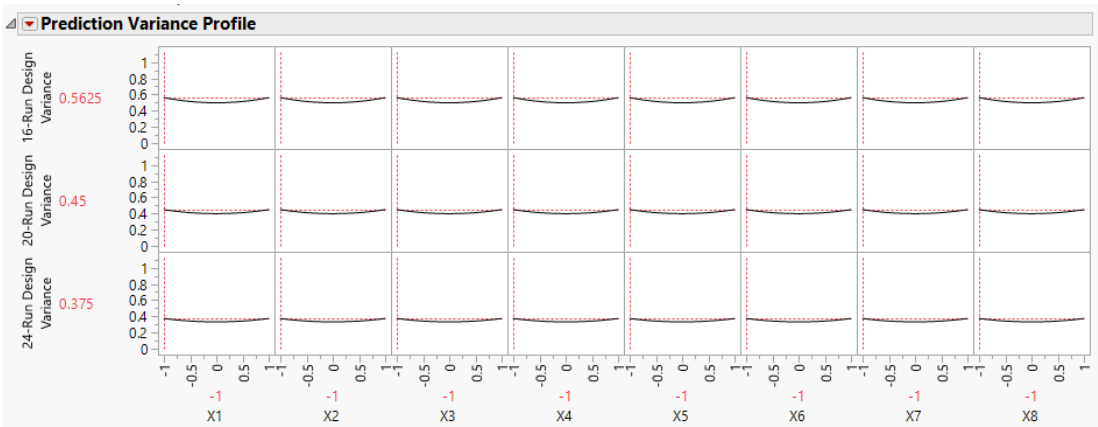


Figure 16.24 shows the Prediction Variance Profile after Maximize Variance was selected for the three designs constructed in "Designs of Different Run Sizes". As expected, the maximum relative prediction variance decreases as the run size increases. The plot also shows values of the factors that give this worst-case relative variance. However, keep in mind that many settings can lead to this same maximum relative variance.

Fraction of Design Space Plot

The Fraction of Design Space Plot shows the proportion of the design space over which the relative prediction variance lies below a given value.

Figure 16.25 Fraction of Design Space Plot for Three Designs

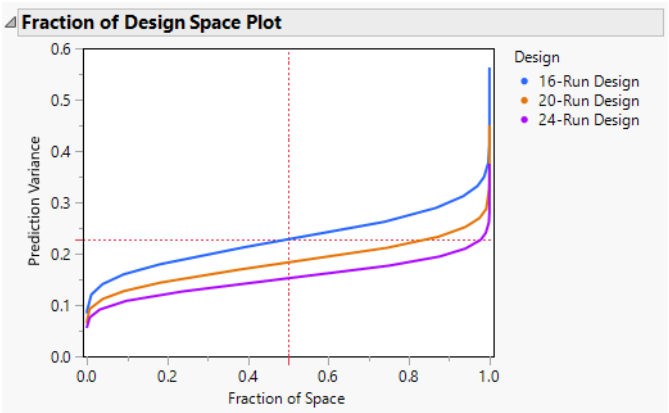


Figure 16.25 shows the Fraction of Design Space plot for the three designs constructed in "Designs of Different Run Sizes". Note the following:

- The *X* axis in the plot represents the proportion of the design space, ranging from 0 to 100%.
- The *Y* axis represents relative prediction variance values.
- For a point (x, y) that falls on a given curve, the value x is the proportion of design space with variance less than or equal to y.
- Red dotted crosshairs mark the value that bounds the relative prediction variance for 50% of design space for the reference design.

Figure 16.25 shows that the relative prediction variance for the 24-run design is uniformly smaller than for the other two designs. The 20-run design has uniformly smaller prediction variance than the 16-run design. The red dotted crosshairs indicate that the relative prediction variance for the 20-run design is less than about 0.23 over about 50% of the design space.

You can use the crosshairs tool to find the maximum relative prediction variance that corresponds to any Fraction of Space value. For example, use the crosshairs tool to see that for the 24-run design, 90% of the prediction variance values are below approximately 0.20.

Note: Plots for the same design might vary slightly, since Monte Carlo sampling of the design space is used in constructing the Fraction of Design Space Plot.

Relative Estimation Efficiency

The Relative Estimation Efficiency report compares designs in terms of the standard errors of parameter estimates for parameters in the assumed model. The standard errors control the length of confidence intervals for the parameter estimates. This report provides an efficiency ratio and the relative standard errors.

The relative estimation efficiency values are colored according to a color gradient shown under the table of relative estimation efficiency values. You can control the color legend using the options in the Relative Estimation Efficiency red triangle menu. See "Color Dashboard".

Figure 16.26 Relative Estimation Efficiency Comparing Two Split-Plot Designs

Relative Estimation Efficiency

Term	Efficiency of 4 Whole Plots Relative to 8 Whole Plots
Intercept	0.778
X1	0.778
X2	1.004
X3 1	0.924
X3 2	1.154
X1*X2	0.982
X1*X3 1	0.924
X1*X3 2	1.154
X2*X3 1	1.107
X2*X3 2	1.449

Good Bad
1.50 1.25 0.80 0.67

Relative Std Error of Estimates

Term	4 Whole Plots	8 Whole Plots
Intercept	0.553	0.43
X1	0.553	0.43
X2	0.23	0.231
X3 1	0.25	0.231
X3 2	0.22	0.254
X1*X2	0.24	0.235
X1*X3 1	0.25	0.231
X1*X3 2	0.22	0.254
X2*X3 1	0.256	0.283
X2*X3 2	0.218	0.315

Figure 16.26 shows the Relative Estimation Efficiency outline for the split-plot designs compared in "Split Plot Designs with Different Numbers of Whole Plots".

Relative Estimation Efficiency

For a given term, the estimation efficiency of the reference design relative to a comparison design is the relative standard error of the term for the comparison design divided by the relative standard error of the term for the reference design. A value less than one indicates that the reference design is not as efficient as the comparison design. A value greater than one indicates that it is more efficient.

Relative Standard Error of Estimates

The Relative Std Error of Estimates report gives the ratio of the standard deviation of a parameter's estimate to the error standard deviation. These values indicate how large the standard errors of the model's parameter estimates are, relative to the error standard deviation. For the i^{th} parameter estimate, the Relative Std Error of Estimate is defined as follows:

$$SE = \sqrt{(\mathbf{X}'\mathbf{X})^{-1}_{ii}}$$

where:

X is the model matrix defined in "The Alias Matrix", and

$(\mathbf{X}'\mathbf{X})^{-1}_{ii}$ is the i^{th} diagonal entry of $(\mathbf{X}'\mathbf{X})^{-1}$.

Alias Matrix Summary

The alias matrix addresses the issue of how terms that are not included in the model affect the estimation of the model terms, if they are indeed active. In the Alias Terms outline, you list potentially active effects that are not in your assumed model but that might bias the estimates of model terms. The alias matrix entries represent the degree of bias imparted to model parameters by the Alias Terms effects. See "Alias Terms" and "Alias Matrix".

The Alias Matrix Summary table lists the terms in the assumed model. These are the terms that correspond to effects listed in the Model outline. Given a design, for each entry in the Term column, the square root of the sum of the squared alias matrix entries for the terms corresponding to effects in the Alias Terms outline is computed. This value is reported in the Root Mean Squared Values column for the given design. For an example, see "Example of Calculation of Alias Matrix Summary Values".

Note: The Alias Matrix Summary report appears only if there are effects in the Alias Terms list.

Figure 16.27 Alias Matrix Summary for Two Designs

Alias Matrix Summary

Term	Root Mean Squared Values Definitive Screening Design	Root Mean Squared Values Plackett-Burman
Intercept	0.0000	0.0000
X1	0.0000	0.2722
X2	0.0000	0.2722
X3	0.0000	0.2722
X4	0.0000	0.2722
X5	0.0000	0.2722
X6	0.0000	0.2722
Total	0.0000	0.2520

Good Bad
0.10 0.30 0.50 0.70

Figure 16.27 shows the Alias Matrix Summary report for the Plackett-Burman and Definitive Screening designs constructed in "Designs of Same Run Size", with only main effects in the Model outline. All two-factor interactions are in the Alias Terms list. The table shows that, for the Definitive Screening Design, main effects are uncorrelated with two-factor interactions.

The Root Mean Squares Values are colored according to a color gradient shown under the Alias Matrix Summary table. You can control the color legend using the options in the Alias Matrix Summary red triangle menu. See "Color Dashboard".

Alias Matrix

The rows of the Alias Matrix are the terms corresponding to the model effects listed in the Model outline. The columns are terms corresponding to effects listed in the Alias Terms outline. The entry in a given row and column indicates the degree to which the alias term affects the parameter estimate corresponding to the model term.

In evaluating your design, you ideally want one of two situations to occur relative to any entry in the Alias Matrix. Either the entry is small or, if it is not small, the effect of the alias term is small so that the bias is small. If you suspect that the alias term might have a substantial effect, then that term should be included in the model or you should consider an alias-optimal design. In fact, alias-optimality is driven by the squared values of the alias matrix.

For additional background on the Alias Matrix, see "The Alias Matrix". See also Lekivetz, R. (2014).

Example of Calculation of Alias Matrix Summary Values

This example illustrates the calculation of the values that appear in the Alias Matrix Summary outline. In this example, you compare the two designs assuming that only main effects are active.

1. Select **Help > Sample Data**, click **Open the Sample Scripts Directory**, and select Compare Same Run Size.jsl.
2. Right-click in the script window and select **Run Script**.

   Two 13-run design tables are constructed:

   – Definitive Screening Design

   – Plackett-Burman

   You are interested only in the Plackett-Burman design. This is the active table.
3. From the Plackett-Burman table, select **DOE > Design Diagnostics > Evaluate Design**.
4. Select X1 through X6 and click **X, Factor**.
5. Click **OK**.
6. Open the Alias Terms outline to confirm that all two-factor interactions are in the Alias Terms list.
7. Open the Alias Matrix outline.

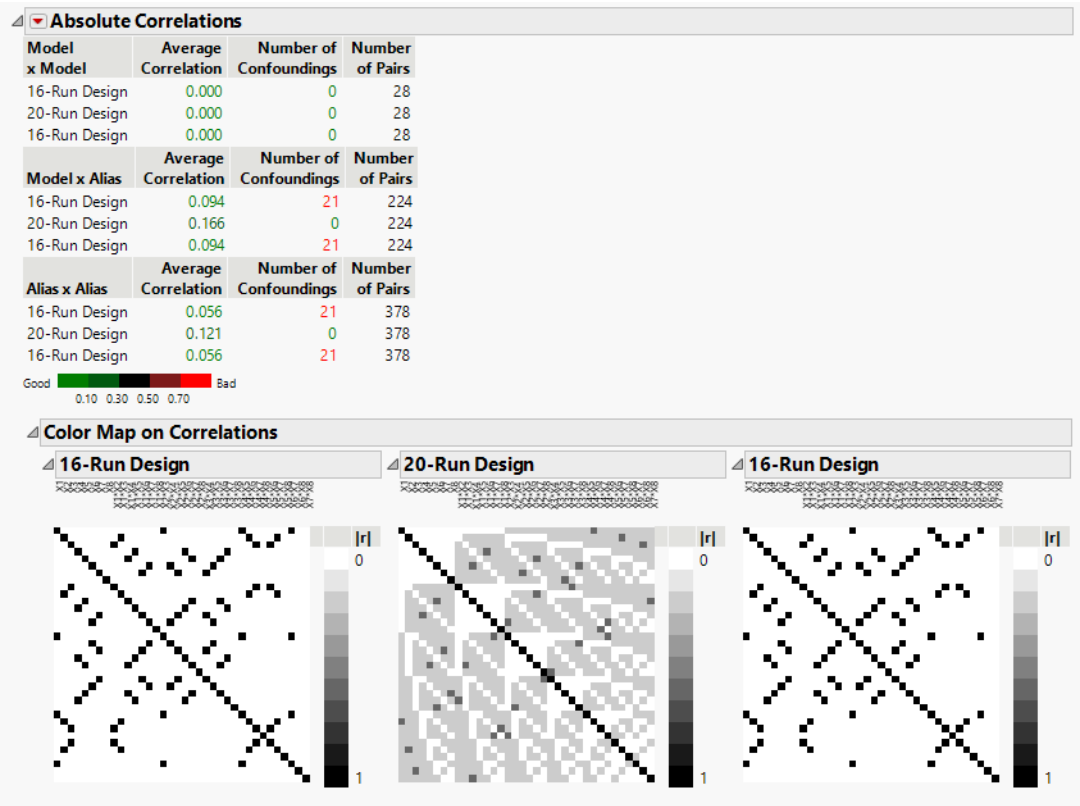
   For each model term listed in the Effect column, the entry in that row for a given column indicates the degree to which the alias term affects the parameter estimate corresponding to the model term.

For example, to obtain the Alias Matrix Summary entry in Figure 16.27 corresponding to X1, square the terms in the row for X1 in the Alias Matrix, average these, and take the square root. You obtain 0.2722.

Absolute Correlations

The Absolute Correlations report summarizes information about correlations between model terms and alias terms.

Figure 16.28 Absolute Correlations Report for Three Designs



Absolute Correlations

Model x Model	Average Correlation	Number of Confoundings	Number of Pairs
16-Run Design	0.000	0	28
20-Run Design	0.000	0	28
16-Run Design	0.000	0	28

Model x Alias	Average Correlation	Number of Confoundings	Number of Pairs
16-Run Design	0.094	21	224
20-Run Design	0.166	0	224
16-Run Design	0.094	21	224

Alias x Alias	Average Correlation	Number of Confoundings	Number of Pairs
16-Run Design	0.056	21	378
20-Run Design	0.121	0	378
16-Run Design	0.056	21	378

Good 0.10 0.30 0.50 0.70 Bad

Figure 16.28 shows the Absolute Correlations report for the three designs constructed in "Designs of Different Run Sizes", with only main effects in the Model outline.

Absolute Correlations Table

The table in the Absolute Correlations report is divided into three sections:

- Model x Model considers correlations between terms corresponding to effects in the Model list.
- Model x Alias considers correlations between terms corresponding to effects in the Model list and terms corresponding to effects in the Alias list.

- Alias x Alias considers correlations between terms corresponding to effects in the Alias list.

Note: If there are no alias terms, only the Model x Model section appears.

For each section of the report, the following are given:

Average Correlation The average of the correlations for all pairs of terms considered in this section of the report.

Number of Confoundings The number of pairs of terms consisting of confounded terms.

Number of Terms The total number of pairs of terms considered in this section of the report.

The values in the Absolute Correlations table are colored according to a color gradient shown under the table. You can control the color legend using the options in the Absolute Correlations red triangle menu. See "Color Dashboard".

Color Map on Correlations

The Color Map on Correlations outline shows plots for each of the designs. The cells of the color map are identified above the map. There are cells for all terms that correspond to effects that appear in either the Model outline or the Alias Terms outline. Each cell is colored according to the absolute value of the correlation between the two terms.

By default, the absolute magnitudes of the correlations are represented by a white to gray to black intensity color theme. In general terms, the color map for a good design shows a lot of white off the diagonal, indicating orthogonality or small correlations between distinct terms. Large absolute correlations among effects inflate the standard errors of estimates.

To see the absolute value of the correlation between two effects, hover over the corresponding cell. To change the color scale, click the Color Map on Correlations red triangle and select **Blue to Gray to Red**. For a custom color scale, right-click in the plot and select **Color Theme**. To save a table of the correlations, right-click to the right of the plot below the legend and select **Table of Correlations**.

Absolute Correlations and Color Map on Correlations Example

Figure 16.28 shows the Absolute Correlations report for the Plackett-Burman and Definitive Screening designs constructed in "Designs of Different Run Sizes". The Model outline contains only main effects, so the Alias Terms outline contains all two-factor interactions. All main effects and two-way interactions are shown in the color maps.

In the Color Map on Correlations for the 16-run design, the black cells off the main diagonal indicate that the corresponding terms have correlation one and therefore are completely confounded. There are nine instances where model terms (main effects) are confounded with alias terms (two factor interactions), and six instances where alias terms are confounded with each other. This is shown in the report under Pairwise Confoundings.

The color maps for the 20- and 24-run designs have no off-diagonal cells that are solid black. It follows that these designs show no instances of confounding between any pair of main or two-way interaction effects. However, it is interesting to note that the 20- and 24-run designs both have a higher Average Correlation for Model x Alias terms than does the 16-run design. Although the 16-run design shows confounding, the average amount of correlation is less than for the 20- and 24-run designs.

Design Diagnostics

The Design Diagnostics outline shows *D*-, *G*-, *A*, and *I*-efficiencies for the reference design relative to the comparison designs. It also shows the Additional Run Size. Given two designs, the one with the higher relative efficiency measure is better.

Figure 16.29 Design Diagnostics for Three Designs

Design Diagnostics

	Efficiency of 16-Run Design Relative to 20-Run Design	Efficiency of 16-Run Design Relative to 24-Run Design
D-efficiency	0.800	0.667
G-efficiency	0.800	0.667
A-efficiency	0.800	0.667
I-efficiency	0.800	0.667
Additional Run Size	-4	-8

Good Bad
1.50 1.25 0.80 0.67

Figure 16.29 shows the Design Diagnostics report for the three designs constructed in "Designs of Different Run Sizes", with only main effects in the Model outline.

The values in the Design Diagnostics table are colored according to a color gradient shown under the table. You can control the color legend using the options in the Design Diagnostics red triangle menu. See "Color Dashboard".

Efficiency and Additional Run Size

Relative efficiencies for each of *D*-, *G*-, *A*-, and *I*-efficiency are shown in the Design Diagnostics report. These are obtained by computing each design's efficiency value and then taking the appropriate ratio. The descriptions of the relative efficiency measures are given in "Relative Efficiency Measures".

Additional Run Size is the number of runs in the reference design minus the number of runs in the comparison design. If your reference design has more runs than your comparison design, then the Additional Run Size tells you how many additional runs you need to achieve the efficiency of the reference design.

Relative Efficiency Measures

Notation

- $\mathbf{X}$ is the model matrix
- p is the number of terms, including the intercept, in the model
- $Var(\hat{y}|\underset{\sim}{\mathbf{x}})$ is the relative prediction variance at the point $\underset{\sim}{\mathbf{x}}$. See "Relative Prediction Variance".

Relative Efficiencies

The relative efficiency of the reference design (*Ref*) to the comparison design (*Comp*) is given by the following expressions:

D Efficiency Eff_{Ref} / Eff_{Comp}, where *Eff* for each design is given as follows:

$$Eff = |\mathbf{X}'\mathbf{X}|^{1/p}$$

G Efficiency Eff_{Comp} / Eff_{Ref}, where *Eff* for each design is given as follows:

$$Eff = Var(\hat{y}|\underset{\sim}{\mathbf{x}})_{max} = \underset{\underset{\sim}{\mathbf{x}} \text{ in D}}{maximum}[\underset{\sim}{\mathbf{x}}'(\mathbf{X}'\mathbf{X})^{-1}\underset{\sim}{\mathbf{x}}]$$

Here, D denotes the design region.

Note: G-Efficiency is calculated using Monte Carlo sampling of the design space. The reported value is based on the larger of $Var(\hat{y}|\underset{\sim}{\mathbf{x}})_{max}$ or the prediction variance from the Monte Carlo sampling. Therefore, calculations for the same design might vary slightly.

A Efficiency Eff_{Comp} / Eff_{Ref}, where *Eff* for each design is given as follows:

$$Eff = \text{Trace}[(\mathbf{X}'\mathbf{X})^{-1}]$$

I Efficiency Eff_{Comp} / Eff_{Ref}, where *Eff* for each design is given as follows:

$$Eff = \frac{\int \underset{\sim}{\mathbf{x}}'(\mathbf{X}'\mathbf{X})^{-1}\underset{\sim}{\mathbf{x}}d\mathbf{x}}{\int d\mathbf{x}}$$

For more information about the calculation, see Section 4.3.5 in Goos and Jones (2011).

Compare Designs Options

Advanced Options > Split Plot Variance Ratio Specify the ratio of the variance of the random whole plot and the subplot variance (if present) to the error variance. Before setting this value, you must define a hard-to-change factor for your split-plot design, or hard and very-hard-to-change factors for your split-split-plot design. Then you can enter one or two positive numbers for the variance ratios, depending on whether you have specified a split-plot or a split-split-plot design.

Chapter 17

Prospective Sample Size and Power

Use the Sample Size and Power platform to investigate the impact of sample size on your ability to answer specific questions when planning experiments or studies.

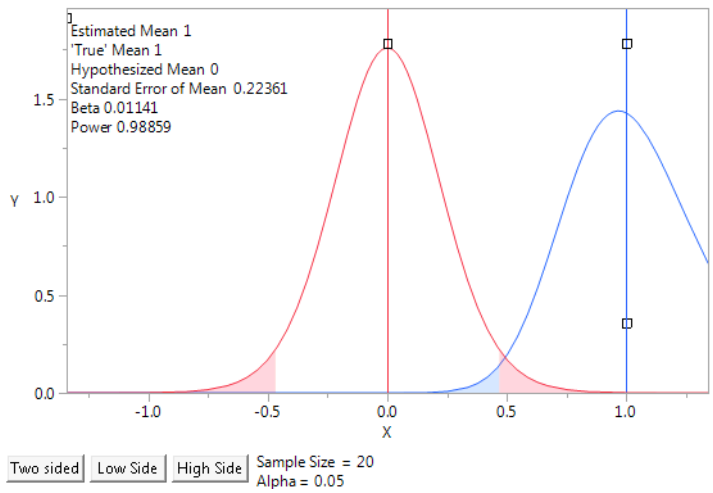
- How many units should I test?
- Will I be able to detect a difference in my treatment means?
- How many units must I test to estimate failure time?

The Sample Size and Power platform is a collection of calculators. The calculators have fields for data entry and calculated values. You can select values to specify and those to calculate. For example, you might specify power and the effect size that you want to detect and then calculate the necessary sample size. Alternatively, you might specify a sample size and power to calculate the effect size that you can detect.

Sample size calculations for hypothesis tests are based on the trade off between Type I and Type II errors. The Type I error, or α, is the probability of rejecting the null hypothesis when it is true. The Type II error, or β, is the probability of not rejecting the null hypothesis when it is false. Generally you would like α and β to be small. Power, or 1 - β, is the probability of rejecting the null hypothesis when it is false.

For more information about power and sample size in JMP, see Barker (2011).

Figure 17.1 Power Animation for One Sample Mean



Contents

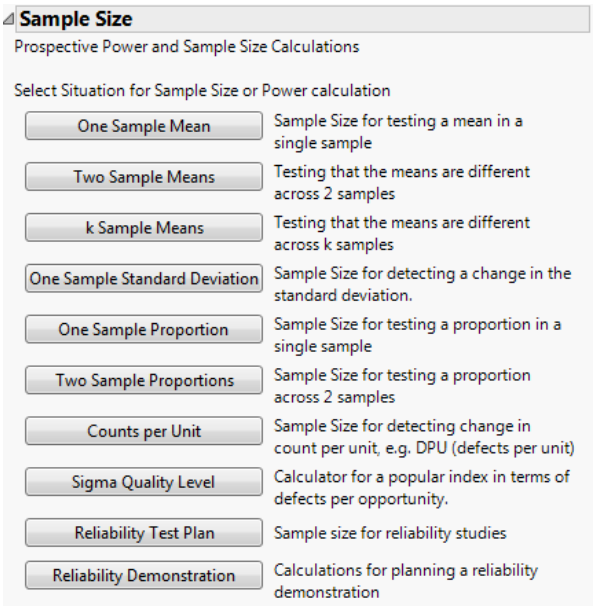
Launch the Sample Size and Power Platform . . . 508
One Sample Mean Calculator . . . 509
Examples of the One Sample Mean Calculator . . . 510
One Sample Mean Calculator Fields . . . 514
Statistical Details for the One Sample Mean Calculator . . . 515
Two Sample Means Calculator . . . 516
Example of the Two Sample Means Calculator . . . 516
Two Sample Means Calculator Fields . . . 517
Statistical Details for the Two Sample Means Calculator . . . 518
k Sample Means Calculator . . . 519
Example of the k Sample Means Calculator . . . 519
k Sample Means Calculator Fields . . . 521
Statistical Details for the k Sample Means Calculator . . . 522
One Sample Standard Deviation Calculator . . . 523
Example of the One Sample Standard Deviation Calculator . . . 523
One Sample Standard Deviation Calculator Fields . . . 524
Statistical Details for the One Sample Standard Deviation Calculator . . . 525
One Sample Proportion Calculator . . . 526
Example of the One Sample Proportion Calculator . . . 527
One Sample Proportion Calculator Fields . . . 528
Statistical Details for the One Proportion Calculator . . . 529
Two Sample Proportions Calculator . . . 530
Example of the Two Sample Proportions Calculator . . . 530
Two Sample Proportions Calculator Fields . . . 531
Statistical Details for the Two Proportions Calculator . . . 532
Counts per Unit Calculator . . . 533
Example of the Counts per Unit Calculator . . . 533
Counts per Unit Calculator Fields . . . 534
Statistical Details for the Counts per Unit Calculator . . . 535
Sigma Quality Level Calculator . . . 536
Example of the Sigma Quality Level Calculator . . . 536
Example Computing the Number of Defects . . . 536
Sigma Quality Level Calculator Fields . . . 537
Statistical Details for the Sigma Quality Calculator . . . 537
Reliability Test Plan Calculator . . . 538
Example of the Reliability Test Plan Calculator . . . 538
Reliability Test Plan Calculator Fields . . . 540
Statistical Details for the Reliability Test Plan Calculator . . . 541
Reliability Demonstration Calculator . . . 542

Example of the Reliability Demonstration Calculator 542
Reliability Demonstration Calculator Fields 544
Statistical Details for the Reliability Demonstration Calculator 545

Launch the Sample Size and Power Platform

The Sample Size and Power platform is a collection of calculators for planning experiments or studies. Launch the Sample Size and Power platform by selecting **DOE > Design Diagnostics > Sample Size and Power**.

Figure 17.2 Sample Size and Power Launch Window



Choose from the following calculators:

- "One Sample Mean Calculator"
- "Two Sample Means Calculator"
- "k Sample Means Calculator"
- "One Sample Standard Deviation Calculator"
- "One Sample Proportion Calculator"
- "Two Sample Proportions Calculator"
- "Counts per Unit Calculator"
- "Sigma Quality Level Calculator"
- "Reliability Test Plan Calculator"
- "Reliability Demonstration Calculator"

One Sample Mean Calculator

Use the One Sample Mean calculator to evaluate sample size for a hypothesis test about one mean. Study the trade offs between sample size, power, significance, and the hypothesized difference to detect. Sample size and power are associated with the following hypothesis test:

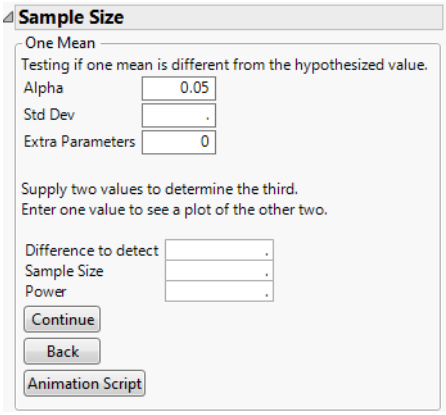
$$H_0: \mu = \mu_0$$

versus the two-sided alternative:

$$H_a: \mu \neq \mu_0$$

where μ is the true mean and μ_0 is the null mean or reference value. The difference to detect is an amount, δ, away from μ_0 that one considers as important to detect based on a set of samples. For the same significance level and power, a larger sample size is needed to detect a small difference than to detect a large difference. It is assumed that the population of interest is normally distributed with mean μ and standard deviation σ.

Figure 17.3 Initial Sample Size and Power Calculator for One Mean



Examples of the One Sample Mean Calculator

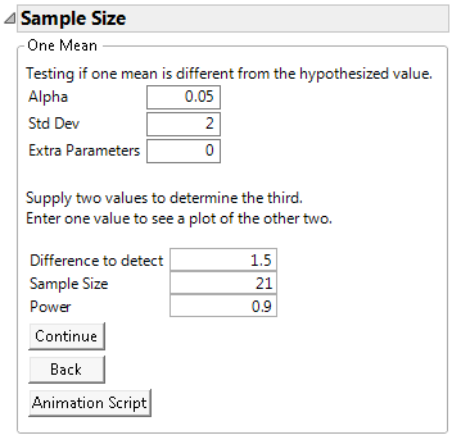
In the following examples, suppose you are interested in demonstrating that the flammability of a new fabric being developed by your company has improved performance over current materials. Previous testing indicates that the standard deviation for time to burn of this fabric is 2 seconds.

Example of Sample Size Calculation

In this initial example, you would like to design an experiment that has 90% power to detect a difference of 1.5 seconds at a significance level of $\alpha = 0.05$. Use the One Sample Mean calculator to calculate the number of fabric samples you need to test.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click the **One Sample Mean** button.
3. Leave **Alpha** set to 0.05.
4. Enter 2 for **Std Dev**.
5. Leave **Extra Parameters** set to 0.
6. Enter 1.5 for **Difference to detect**.
7. Leave **Sample Size** blank.
8. Enter 0.9 for **Power**.
9. Click **Continue**.

Figure 17.4 One-Sample Mean Calculator



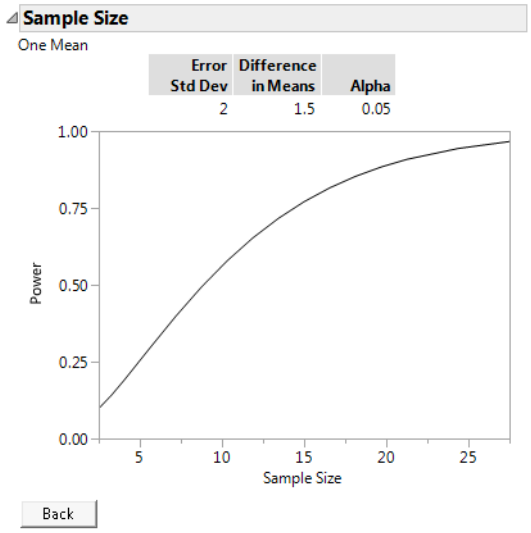
At a significance level of 0.05, 21 fabric samples are needed to have a 90% chance of detecting a significant difference of 1.5 seconds in the burn time.

Example of Power versus Sample Size Plot

To explore trade offs between sample size and power in your fabric experiment, use the plot of power versus sample size.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **One Sample Mean**.
3. Leave **Alpha** set to 0.05.
4. Enter 2 for **Std Dev**.
5. Leave **Extra Parameters** set to 0.
6. Enter 1.5 for the **Difference to detect**.
7. Leave **Sample Size** blank.
8. Leave **Power** blank.
9. Click **Continue** to launch the power by sample size plot.

Figure 17.5 Power by Sample Size



The plot shows a range of sample sizes for which the power varies from about 0.1 to about 0.95. You could reduce the number of tests in your experiment to about 15 and maintain power above 75%. However, if you ran only 10 tests, the power to detect a significant difference of 1.5 would drop to about 50%.

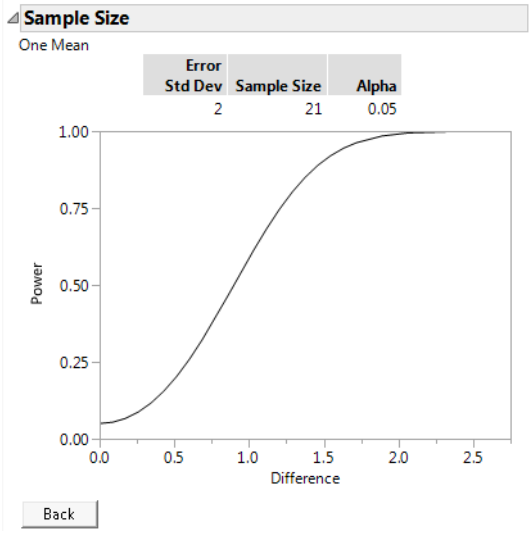
Tip: Use the crosshair tool to obtain sample size and power combinations from the plot.

Example of Power versus Difference Plot

To explore trade offs between power and the magnitude of the difference that you can detect with 21 observations in your fabric experiment, use the plot of power versus difference.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **One Sample Mean**.
3. Leave **Alpha** set to 0.05.
4. Enter 2 for **Std Dev**.
5. Leave **Extra Parameters** set to 0.
6. Leave **Difference to detect** blank.
7. Enter 21 for **Sample Size**.
8. Leave **Power** blank.
9. Click **Continue**.

Figure 17.6 Plot of Power by Difference to Detect for a Sample Size of 21



At a significance level of 0.05 and 21 observations, you can detect a difference of 1.5 seconds with 90% power. If the difference is only one second smaller, then with 21 fabric samples, you have about 50% power of detecting the difference.

Example of an Animation Script

Use an animation script to explore how changing the sample size affects power.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.

2. Click **One Sample Mean**.
3. Leave **Alpha** set to 0.05.

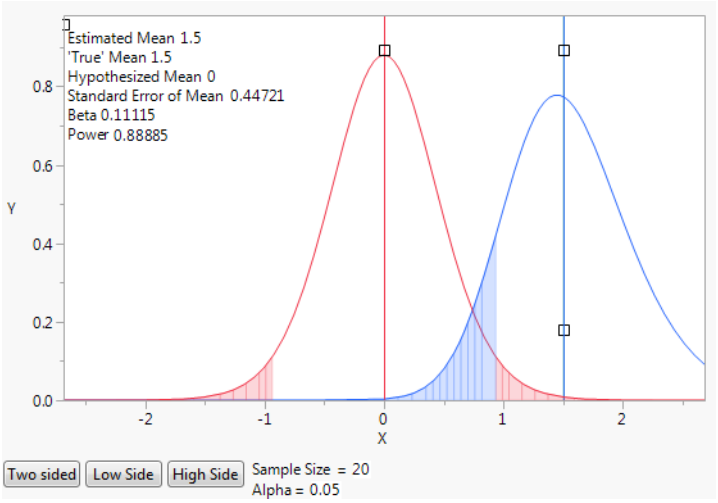
   Tip: You can change the Alpha level after the animation is launched by clicking the Alpha value in the animation window.

4. Enter 2 for **Std Dev**.
5. Leave **Extra Parameters** set to 0.
6. Enter 1.5 as **Difference to detect**.
7. Leave **Sample Size** blank.

   Tip: When you leave the sample size blank, the default sample size is set to 20. You will see the default of 20 after the animation script is launched. You can change the sample size before or after the animation is launched. To change it after the animation is launched, click the Sample Size value in the animation window.

8. Leave **Power** blank.
9. Click **Animation Script**.

Figure 17.7 Initial Animation Script to Illustrate Power



The initial animation plot shows two *t*-density curves:

- The red curve shows the *t*-distribution when the true mean is zero.
- The blue curve shows the *t*-distribution when the true mean is 1.5, which is the difference to be detected.

– The blue shading indicates the probability of committing a type II error. A type II error is the probability of not detecting a difference when there is a difference. The probability of a type II error is often denoted by β.
– The red shading indicates the probability of committing a type I error. A type I error is the probability of concluding that the difference in means is significant when there is no difference. The probability of a type I error is often denoted by α.

Select and drag the square handles to see the changes in statistics based on the positions of the curves. To change the values of Sample Size and Alpha, click their values beneath the plot.

By default, the animation shows a two sided test. Use the **Two Sided**, **Low Side**, and **High Side** buttons to toggle between not equal, less than, or greater than alternative hypotheses.

One Sample Mean Calculator Fields

Specify the following quantities:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Tip: For a one-sided test, use $\alpha = \alpha * 2$. For a one sided test with $\alpha = 0.05$, use the two-sided calculator with $\alpha = 0.10$. The resulting values are those needed for a one-sided test with $\alpha = 0.05$.

Std Dev The assumed standard deviation. An estimate of the error standard deviation could be the root mean square error (RMSE) from a previous model fit.

Tip: Use a standard deviation of 1 to estimate the sample size needed to detect differences measured in standard deviation units.

Extra Parameters The number of parameters other than μ in the hypothesis test. This option can be used for multi-factor designs. Leave the default zero in this field for simple cases.

In a multi-factor design where effects are orthogonal, you can specify the number of additional model parameters here. For example, in a three-factor, two-level design with all three two-factor interactions, the number of additional parameters is five: two parameters for the other main effects, and three parameters for the interactions.

Specify two of the following values to calculate the third value, or specify one value to obtain a plot of the relationship between the other two:

Difference to Detect The smallest difference between the true mean and the hypothesized or reference mean you want to be able to declare statistically significant.

Sample Size The total number of observations (runs, experimental units, or samples) in your experiment.

Power The probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Calculator Buttons

Continue Evaluates the missing value when two parameters are specified, or launches a plot comparing two missing parameters if only one parameter is specified.

Back Returns to the previous Sample Size and Power launch window.

Animation Script Launches an interactive plot to illustrate and explore the relationship between power and the difference to detect. See "Example of an Animation Script".

Statistical Details for the One Sample Mean Calculator

The one sample mean calculations are based on the traditional full versus reduced F test for the following hypothesis test:

$$H_0: \mu = \mu_0 \text{ vs } H_a: \mu \neq \mu_0$$

JMP calculates power as follows:

$$\Pr(\text{reject } H_0 \mid \mu = \mu_0 + \delta) = 1 - F\left(f_{1-\alpha}, 1, n - p - 1, \frac{n\delta^2}{\sigma^2}\right)$$

where:

α is the significance level.

n is the sample size.

p is the number of extra parameters.

δ is the difference to detect.

$f_{1-\alpha}$ is the $(1 - \alpha)^{th}$ quantile of the $F(1, n- p -1)$ distribution.

$F(x, df_1, df_2, nc)$ is the cumulative distribution function of the non-central F distribution with degrees of freedom df_1 and df_2 and non-centrality parameter nc evaluated at x.

Because analytical solutions for δ and n do not exist, numerical solutions are used to solve for them.

For more information about calculations in JMP, see Barker (2011, Section 2.1).

Two Sample Means Calculator

Use the Two Sample Means calculator to evaluate sample size for a hypothesis test about two means. Explore questions such as how large a treatment effect is detectable for a given sample size, significance level, desired power, and assumed variability in the data. Sample size and power are associated with the following hypothesis test:

$$H_0: \mu_1 - \mu_2 = 0$$

versus the two-sided alternative:

$$H_a: \mu_1 - \mu_2 \neq 0$$

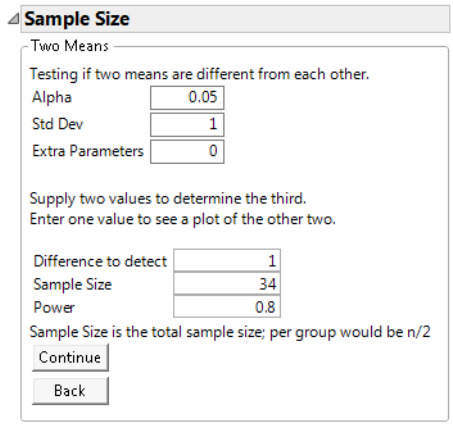
where μ_1 and μ_2 are the true means of the two populations. It is assumed that the populations of interest are normally distributed and that you want to detect a difference of δ between the means.

Example of the Two Sample Means Calculator

Suppose you are comparing two groups and want to detect a 1 standard deviation difference between the group means at a 0.05 significance level with 80% power.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **Two Sample Means**.
3. Leave **Alpha** as 0.05.
4. Enter 1 for **Std Dev**.
5. Leave **Extra Parameters** as 0.
6. Enter 1 as **Difference to detect**.
7. Leave **Sample Size** blank.
8. Enter 0.8 for **Power**.
9. Click **Continue**.

Figure 17.8 Two Sample Means Calculator Result



The sample size is calculated as 34. With 17 samples in each group, you have an 80% chance of detecting a 1 standard deviation difference between the two sample means at a significance level of $\alpha = 0.05$.

Two Sample Means Calculator Fields

Specify the following quantities:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Std Dev The assumed standard deviation where both groups are assumed to have equal standard deviation.

Extra Parameters The number of parameters other than μ_1 and μ_2 in the hypothesis test. This option can be used for multi-factor designs. Leave the default zero in this field for simple cases.

In a multi-factor balanced design, the number of additional model parameters can be specified here. For example, in a three-factor two-level design with all three two-factor interactions, the number of additional parameters is four: one parameter for the other main effects, and three parameters for the interactions.

Specify two of the following parameters to calculate the third value, or specify one value to obtain a plot of the relationship between the other two parameters:

Difference to Detect The smallest difference between the two means that you want to be able to declare statistically significant.

Sample Size The total number of observations (runs, experimental units, or samples) in your experiment. Use half of this sample size in each group.

Power The probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Calculator Buttons

Continue Evaluates the missing value when two parameters are specified, or launches a plot comparing two missing parameters if only one parameter is specified.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the Two Sample Means Calculator

The two sample mean calculations are based on the traditional full versus reduced F test for the following hypothesis test:

$$H_0: \mu_1 - \mu_2 = 0 \text{ vs } H_a: \mu_1 - \mu_2 \neq 0$$

JMP calculates power as follows:

$$\text{power} = \Pr(\text{reject } H_0 \mid (\mu_1 - \mu_2 = \delta)) = 1 - F\left(f_{1-\alpha}, 1, 2n - p - 2, \frac{n\delta^2}{2\sigma^2}\right)$$

where:

α is the significance level.

n is the sample size per group.

p is the number of extra parameters.

δ is the difference to detect.

$f_{1-\alpha}$ is the $(1 - \alpha)^{\text{th}}$ quantile of the $F(1, 2n - p - 2)$ distribution.

$F(x, \text{df}_1, \text{df}_2, nc)$ is the cumulative distribution function of the non-central F distribution with degrees of freedom df_1 and df_2 and non-centrality parameter nc evaluated at x.

Because analytical solutions for δ and n do not exist, numerical solutions are used to solve for them.

For more information about calculations in JMP, see Barker (2011, Section 2.2).

k Sample Means Calculator

Use the k Sample Means calculator to determine an appropriate sample size for a hypothesis test about two to ten means. Sample size and power are associated with the following hypothesis test:

$$H_0: \mu_1 = \mu_2 = \ldots = \mu_k$$

versus the two-sided alternative:

H_a : not all means equal

where:

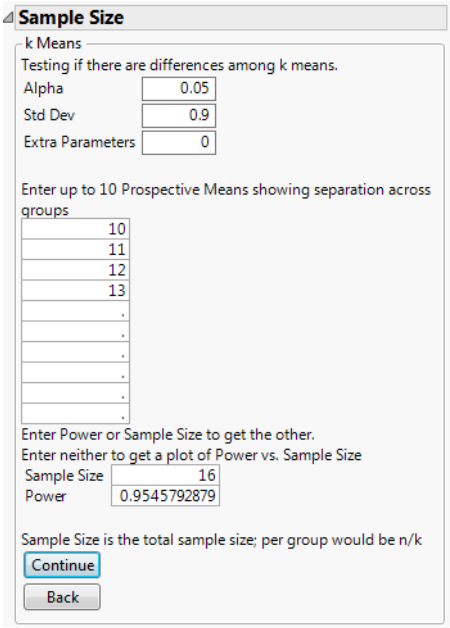
$$X_{ij} \sim N(\mu_j, \sigma^2) \text{ for } i = 1, \ldots, n, \; j = 1, \ldots, k$$

Example of the k Sample Means Calculator

Suppose you have 4 observations from each of 4 groups and you want to test to see whether all of the group means are equal. You expect the group means to be 10, 11, 12, and 13 with a standard deviation of 0.9. Calculate the power of detecting a difference in one or more means at a 0.05 significance level.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **k Sample Means**.
3. Leave **Alpha** as 0.05.
4. Enter 0.9 for **Std Dev**.
5. Leave **Extra Parameters** as 0.
6. Enter 10, 11, 12, and 13 as the four levels of prospective means.
7. Enter 16 for **Sample Size**.
8. Leave **Power** blank.
9. Click **Continue**.

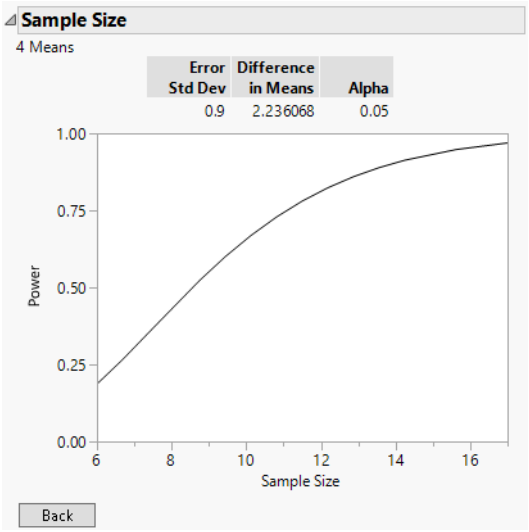
Figure 17.9 Prospective Power for 4 Means



The Power is calculated as approximately 0.95. This means that there is a 95% chance of detecting that at least one of the means is different at a 0.05 significance level, assuming that the population means are 10, 11, 12, and 13, with a standard deviation of 0.9 and a total sample size of 16.

10. Click the 16 for Sample Size and delete it.
11. Click the 0.95 for Power and delete it.
12. Click **Continue** to launch a plot of power versus sample size.

Figure 17.10 Prospective Power for k Means and Plot of Power by Sample Size



From the plot of power versus sample size, you can confirm that a sample size of 16 is acceptable. Hover over the Sample Size axis and drag to extend the axis beyond 16. You can see that the increase in power for sample sizes above 16 is slight.

The plot also reports the difference in means, which is calculated as the square root of the sum of squared deviations from the grand mean. In this case, it is the square root of $(-1.5)^2 + (-0.5)^2 + (0.5)^2 + (1.5)^2$, which is the square root of 5. Therefore, the difference in means in this example is approximately 2.236.

k Sample Means Calculator Fields

Specify the following quantities:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Std Dev The assumed standard deviation where all groups are assumed to have equal standard deviation.

Extra Parameters The number of parameters other than the μ in the hypothesis test. This option can be used for multi-factor designs. Leave the default 0 in this field in simple cases.

Prospective Means The assumed values for the means. At least two means must be specified.

Specify one of the following values to calculate the second, or leave both values blank to obtain a plot of the relationship between the two:

Sample Size The total number of observations (runs, experimental units, or samples) in your experiment. Use a sample size of n/k in each group, where n is the total number of observations.

Power The probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Calculator Buttons

Continue Evaluates the missing value when one parameter is specified, or launches a plot comparing two parameters if no parameters are specified.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the k Sample Means Calculator

The k sample mean calculations are based on the following:

$$\text{power} = 1 - F\left(f_{1-\alpha}, 1k-1, 2kn-p-2k, \frac{n\sum_{j=1}^{k}(\mu_j - \bar{\mu})^2}{\sigma^2} \right)$$

where:

α is the significance level

σ is the common standard deviation among the k groups.

n is the sample size per group.

p is the number of extra parameters.

$\bar{\mu}$ is the mean of the k group prospective means

$f_{1-\alpha}$ is the $(1 - \alpha)^{th}$ quantile of the $F(k - 1, kn - p - k)$ distribution.

$F(x, df_1, df_2, nc)$ is the cumulative distribution function of the non-central F distribution with degrees of freedom df_1 and df_2 and non-centrality parameter nc evaluated at x.

Numerical solutions are used to solve for n.

Difference in Means

The difference in means is calculated as follows:

$$\sqrt{\sum_{j=1}^{k} (\mu_j - \bar{\mu})^2}$$

For more information about calculations in JMP, see Barker (2011, Section 2.3).

One Sample Standard Deviation Calculator

Use the One Sample Standard Deviation calculator to evaluate sample size for a hypothesis test about one standard deviation. Sample size and power are associated with the following hypothesis test:

$$H_0: \sigma = \sigma_0$$

versus the one-sided alternative:

$$H_a: \sigma > \sigma_0 \text{ or } H_a: \sigma < \sigma_0$$

where σ is the population standard deviation and σ_0 is the hypothesized value. The difference to detect is an amount, δ, away from σ_0 that one considers as important to detect based on a set of samples.

Example of the One Sample Standard Deviation Calculator

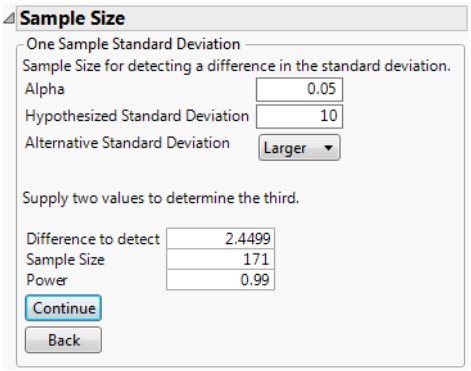
Note: This example is from the online manual of The National Institute of Standards and Technology (NIST). You can access the NIST manual examples at https://www.itl.nist.gov/div898/handbook/prc/section2/prc232.htm.

The variance for resistivity measurements on a lot of silicon wafers is claimed to be 100 ohm-cm squared. The buyer is unwilling to accept a lot if the variance is greater than 155 ohm-cm squared. How many wafers must you test to estimate the lot variance with precision to detect an increase in 55 ohm-cm squared from the target of 100 ohm-cm squared? In terms of standard deviation, the hypothesized standard deviation, σ_0, is 10 (the square root of 100) and σ is 12.4499 (the square root of 100 + 55 = 155). The difference to detect is 12.4499 – 10 = 2.4499. The desired power is 0.99 and the significance level is 0.05.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.

2. Click **One Sample Standard Deviation**.
3. Leave **Alpha** as 0.05.
4. Enter 10 for **Hypothesized Standard Deviation**.
5. Select Larger for **Alternative Standard Deviation**.
6. Enter 2.4499 for **Difference to Detect**.
7. Leave **Sample Size** blank.
8. Enter 0.99 for **Power**.
9. Click **Continue**.

Figure 17.11 One-Sample Standard Deviation Calculator



You must test 171 wafers in order to have a 99% chance of detecting an increase in the standard deviation of 2.4499 from a standard deviation of 10, with an alpha of 0.05.

One Sample Standard Deviation Calculator Fields

Specify the following:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Hypothesized Standard Deviation The hypothesized or baseline standard deviation to which the sample standard deviation is compared.

Alternative Standard Deviation The direction of the change that you want to detect, either Larger or Smaller.

Note: If you select Smaller from the Alternative Standard Deviation menu, enter a negative Difference to Detect.

Specify two of the following values to calculate the third value:

Difference to Detect The smallest detectable difference (how small a difference you want to be able to declare statistically significant). This is the difference between the hypothesized value and the true value.

Sample Size The total number of observations (runs, experimental units, or samples) in your experiment.

Power The probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Calculator Buttons

Continue Evaluates the missing value.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the One Sample Standard Deviation Calculator

The formula that JMP uses to calculate power depends on the setting of the Alternative Standard Deviation option.

For cases where the alternative standard deviation is larger, JMP calculates power as follows:

$$\text{power} = 1 - F\left(\frac{\sigma_0^2 \cdot \chi_{1-\alpha}}{(\sigma_0 + \delta)^2}, n-1\right)$$

$$\delta = \sigma_0 \sqrt{\frac{\chi_{1-\alpha}}{\chi_\beta}} - \sigma_0$$

where:

σ_0 is the hypothesized standard deviation.

σ is the true standard deviation

X_p is the $(1 - p)^{th}$ quantile of the X_{n-1}^2 distribution.

$F(x, n\text{-}1)$ is the cumulative distribution function of the X_{n-1}^2 distribution evaluated at x.

For cases where the alternative standard deviation is smaller, JMP calculates power as follows:

$$\text{power} = 1 - F\left(\frac{\sigma_0^2 \cdot \chi_\alpha}{(\sigma_0 + \delta)^2}, n-1\right)$$

$$\delta = \sigma_0 \sqrt{\frac{\chi_\alpha}{\chi_{1-\beta}}} - \sigma_0$$

where:

σ_0 is the hypothesized standard deviation.

σ is the true standard deviation

X_p is the $(1 - p)^{th}$ quantile of the X_{n-1}^2 distribution.

$F(x, n\text{-}1)$ is the cumulative distribution function of the X_{n-1}^2 distribution evaluated at x.

Numerical solutions are used to solve for n.

For more information about calculations in JMP, see Barker (2011, Section 2.4).

One Sample Proportion Calculator

Use the One Sample Proportion calculator to evaluate sample size for a hypothesis test about one proportion. Sample size and power are associated with the following hypothesis test:

$$H_0: p = p_0$$

versus the two-sided alternative:

$$H_a: p \neq p_0$$

or versus a one-sided alternative:

$$H_a: p < p_0 \text{ or } H_a: p > p_0$$

where p is the population proportion and p_0 is the null proportion.

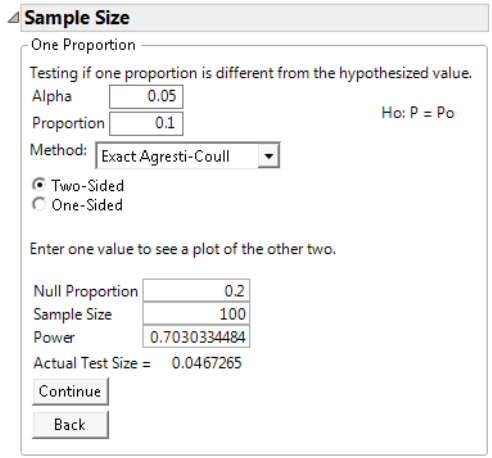
The One Proportion Calculator also provides the actual test size. This is the actual Type I error rate for the specified assumptions. Since the binomial distribution is discrete, the actual test size can differ significantly from the stated Alpha level for small samples or proportions near 0 or 1. To guarantee an alpha level equal to or greater than your stated level, use the Exact Clopper-Pearson method.

Example of the One Sample Proportion Calculator

Suppose that an assembly line has a historical proportion of defects equal to 0.1. Given a sample size of 100 and an alpha level of 0.05, you want to calculate the power to detect a defect rate that differs by 0.1 or more from the historical rate.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **One Sample Proportion**.
3. Leave **Alpha** as 0.05.
4. Leave 0.1 as the value for **Proportion**.
5. Leave the **Method** as Exact Agresti-Coull.
6. Leave the test type as **Two-Sided**.
7. Leave 0.2 as the value for **Null Proportion**.
8. Enter 100 as the **Sample Size**.
9. Click **Continue**.

Figure 17.12 One Sample Proportion Calculator



For a sample size of 100, the power is approximately 70%. The Actual Test Size is approximately 0.0467, which is slightly less than the desired 0.05. With a sample size of 100, if the defect rate is 0.2, the probability of a Type I error of rejecting the null hypothesis is 5%. Alternatively, the probability of a Type II error of not rejecting the null hypothesis is 30% (1 - power).

One Sample Proportion Calculator Fields

Specify the following quantities and test settings:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Proportion The true proportion, which could be known or hypothesized. The default value is 0.1.

Tip: For all other parameters fixed, the largest sample size occurs when the proportion is 0.5.

Method Select a method. Choices are Exact Agresti-Coull or Exact Clopper-Pearson. The Clopper-Pearson method tends to be more conservative (larger sample size) than the Agresti-Coull method.

One-Sided or Two-Sided Select either a one-sided or a two-sided test.

Specify two of the following parameters to calculate the third value, or specify one value to obtain a plot of the relationship between the other two parameters.

Note: The plot uses the normal approximation method rather than an exact method.

Null Proportion The proportion to test against (p_0). The default value is 0.2.

Sample Size The total number of observations (runs, experimental units, or samples) in the experiment.

Power The probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Calculator Buttons

Continue Evaluates the missing value when two parameters are specified, or launches a plot comparing two missing parameters if only one parameter is specified.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the One Proportion Calculator

The one proportion sample size computations use exact methods based on the binomial distribution. Exact calculations guarantee that the stated power level is obtained.

Agresti-Coull Method

The exact Agresti-Coull method uses the adjusted Wald-based test statistic. JMP calculates power under the two-sided null hypothesis as follows:

$$\text{Power} = \sum_{y=0}^{n} \Pr\{Y = y \mid Y \sim \text{binomial}(n, p)\} I\{T(y) \geq \chi_{1-\alpha}\}$$

where:

$$T(y) = \frac{(\hat{p} - p_0)^2}{\frac{\hat{p}(1-\hat{p})}{n+4}}$$

$$\hat{p} = \frac{y+2}{n+4}$$

$I\{T(y_1,y_2) \geq \chi_{1-\alpha}\} = 1$ if $(T(y_1,y_2) \geq \chi_{1-\alpha})$ and 0 otherwise, and

$\chi_{1-\alpha}$ is the $(1 - \alpha)^{\text{th}}$quantile of the χ_1^2 distribution.

Because there is not a closed-form expression for n or p_0, numerical techniques are used to solve for n or p_0.

For more information about the adjusted Wald test statistic, see Agresti and Coull (1998). For more information about calculations in JMP, see Barker (2011, Section 3.3).

Clopper-Pearson Method

The exact Clopper-Pearson method is based on the binomial distribution. This method results in an alpha level equal to or greater than the stated level. The Clopper-Pearson method is more conservative (larger sample size) than the Agresti-Coull method.

The exact Clopper-Pearson method uses the binomial distribution directly. Numerical techniques are used to solve for the unknown parameter.

For more information about the Clopper-Pearson exact method, see Clopper and Pearson (1934) or Agresti and Coull (1998, Section 1).

Two Sample Proportions Calculator

Use the Two Sample Proportions calculator to evaluate sample size for a hypothesis test about two proportions. Sample size and power are associated with the following hypothesis test:

$$H_0: p_1 - p_2 = D_0$$

versus the two-sided alternative:

$$H_a: p_1 - p_2 \neq D_0$$

or versus either of the following one-sided alternatives:

$$H_a: (p_1 - p_2) < D_0 \text{ or } H_a: (p_1 - p_2) > D_0$$

where p_1 and p_2 are the population proportions from two populations, and D_0 is the hypothesized difference in proportions.

The Two Proportions Calculator also provides the actual test size. This is the actual Type I error rate for the specified assumptions. Since the binomial distribution is discrete, the actual test size can differ significantly from the stated Alpha level for small samples or proportions near 0 or 1.

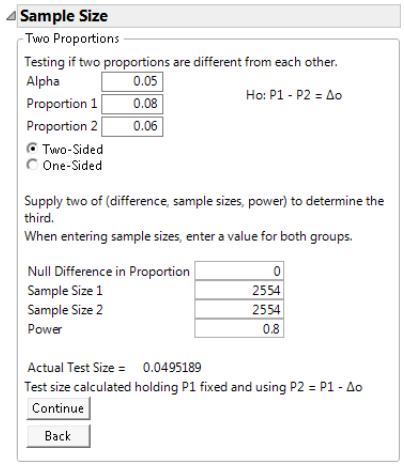
Example of the Two Sample Proportions Calculator

Suppose you are responsible for two silicon wafer assembly lines. Based on the knowledge from many runs, one of the assembly lines has a defect rate of 8% and the other line has a defect rate of 6%. You want to know the sample size necessary to have 80% power to conclude there is a difference in defect rates.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **Two Sample Proportions**.
3. Leave **Alpha** set at 0.05.
4. Enter 0.08 for **Proportion 1**.
5. Enter 0.06 for **Proportion 2**.
6. Select a **Two-Sided** test.
7. Enter 0 for **Null Difference in Proportion**.
8. Leave **Sample Size 1** and **Sample Size 2** blank.
9. Enter 0.8 for **Power**.

10. Click **Continue**.

Figure 17.13 Difference between Two Proportions for a Two-Sided Test



The calculator shows that you should test a sample size of 2,554 wafers from each production line in order to detect a difference in defect rates.

Two Sample Proportions Calculator Fields

Specify the following quantities and test settings:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Proportion 1 The proportion for population 1, which could be known or hypothesized. The default value is 0.5.

Proportion 2 The proportion for population 2, which could be known or hypothesized. The default value is 0.1.

One-Sided or Two-Sided Select either a one-sided or a two-sided test.

Specify two of the following parameters to calculate the third where the sample sizes are considered as one parameter. The sample sizes must both be entered or both be left blank for calculation.

Null Difference in Proportion The difference in proportions under the null hypothesis. The default value is 0.2.

Sample Size 1 The total number of observations (runs, experimental units, or samples) in the experiment to estimate Proportion 1.

Sample Size 2 The total number of observations (runs, experimental units, or samples) in the experiment to estimate Proportion 2.

Note: When sample size is calculated, Sample Size 1 = Sample Size 2. If you enter sample sizes, they do not have to be equal.

Power The probability of rejecting the null hypothesis when it is false. High power is desired. With all other parameters fixed, power increases with sample size.

Calculator Buttons

Continue Evaluates the missing value.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the Two Proportions Calculator

Two proportions calculations are based on exact methods similar to those used for the one proportion calculator. For the two-sided hypothesis

$$H_0: p_1 - p_2 = D_0 \text{ vs } H_a: p_1 - p_2 \neq D_0$$

JMP calculates power under the null hypothesis as follows:

$$\text{Power} = \sum_{y_1=0}^{n_1} \sum_{y_2=0}^{n_2} \Pr(Y_1 = y_1)\Pr(Y_2 = y_2) I\{T(y_1,y_2) \geq \chi_{1-\alpha}\}$$

where the adjusted Wald statistic is defined as follows:

$$T(y_1,y_2) = \frac{(\hat{p}_1 - \hat{p}_2 - \delta_0)^2}{\frac{\hat{p}_1(1-\hat{p}_1)}{n_1+2} + \frac{\hat{p}_2(1-\hat{p}_2)}{n_2+2}} \text{ where } \hat{p}_j = \frac{y_j+1}{n_j+2} \text{ for } j = 1, 2$$

and:

$$I\{T(y_1,y_2) \geq \chi_{1-\alpha}\} = 1 \text{ if } (T(y_1,y_2) \geq \chi_{1-\alpha}) \text{ and 0 otherwise}$$

and $\chi_{1-\alpha}$ is the $(1 - \alpha)^{\text{th}}$ quantile of the χ^2 distribution.

Because there are no closed-form expressions for the n_i or δ_0, numerical techniques are used to solve for the n_i and δ_0.

For more information about the adjusted Wald test statistic, see Agresti and Coull (1998). For more information about calculations in JMP, see Barker (2011, Section 3.3).

Counts per Unit Calculator

Use the Counts per Unit Calculator to evaluate sample size for a one-sided hypothesis test about the defects (counts) per unit. Sample size and power are associated with the following hypothesis test:

$$H_0: \lambda \leq \lambda_0$$

versus a one-sided alternative:

$$H_a: \lambda > \lambda_0$$

where λ is the rate of a Poisson distribution.

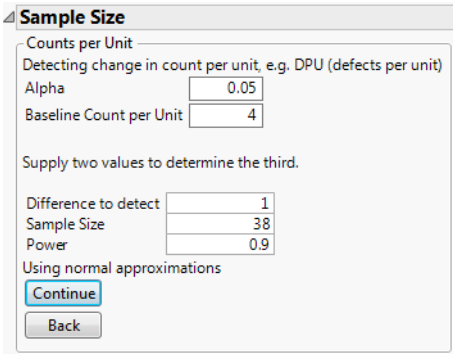
Example of the Counts per Unit Calculator

Consider a wafer manufacturing process with a target of 4 or fewer defects per wafer. You want to verify that a new process meets that target. How many wafers should you test to detect a difference of 1 that the count of defects per wafer is on target with 90% power at the significance level of 0.05?

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **Counts per Unit**.
1. Leave **Alpha** as 0.05 (the chance of failing the test if the new process is as good as the target).
2. Enter 4 as the **Baseline Counts per Unit**, indicating the target of 4 defects per wafer.
3. Enter 1 as the **Difference to detect**.
4. Enter a power of 0.9.

   This is the chance of detecting a change larger than 1, which is equivalent to 5 or more defects per wafer. In this type of situation, the significance level (alpha) is sometimes called the *producer's risk* and the power (beta) is called the *consumer's risk*.
5. Click **Continue**.

Figure 17.14 Counts per Unit Calculator



You must test 38 wafers. The process meets the target if there are fewer than 173 defects.

Note: The 173 defects are the maximum number of defects that can occur in 38 wafers without rejecting the null hypothesis.

Counts per Unit Calculator Fields

Specify the following quantities:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Baseline Count per Unit The number of targeted defects per unit. The default value is 0.1.

Specify two of the following quantities to calculate the third quantity:

Difference to detect The smallest detectable difference to test against specified in defects per unit.

Sample Size The number of units.

Power The probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Calculator Buttons

Continue Evaluates the missing value.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the Counts per Unit Calculator

Calculations for the counts per unit calculator are based on applying a normal approximation to the Poisson distribution. The test uses the following hypothesis:

$$H_0: \lambda \leq \lambda_0 \text{ vs } H_a: (\lambda > \lambda_o)$$

The above hypothesis uses the following test statistic:

$$T = \frac{\bar{d} - \lambda_0}{\sqrt{\lambda_0 / n}}$$

where n is the sample size and $\bar{d}$ is the mean number of defects per unit in the sample. We assume T to be approximately normally distributed. The power calculation is based on the distribution of T under the null and alternative hypotheses.

$$\text{power} = \Pr(\text{reject } H_0 \mid \lambda = \lambda_0 + \delta)$$

$$= \Pr[T > Z_{1-\alpha} \mid (\lambda = \lambda_0 + \delta)] = 1 - \Phi\left(\frac{Z_{1-\alpha} - \delta\sqrt{n/\lambda_0}}{\sqrt{(\lambda_0 - \delta)/\lambda_0}}\right)$$

where $\Phi()$ is the standard normal cumulative distribution function and Z_{1-p} is the $(1 - p)^{\text{th}}$ quantile of the standard normal distribution.

Using $1 - \beta$ to denote the desired power to reject the null hypothesis, the sample size is calculated as follows:

$$n = \frac{\lambda_0}{\delta^2}\left(Z_{1-\alpha} - Z_{\beta}\sqrt{\frac{\lambda_0 + \delta}{\lambda_0}}\right)^2$$

Because an analytical solution for δ does not exist, numerical methods are used to solve for δ given power and n.

For more information about calculations in JMP, see Barker (2011, Section 2.5).

Sigma Quality Level Calculator

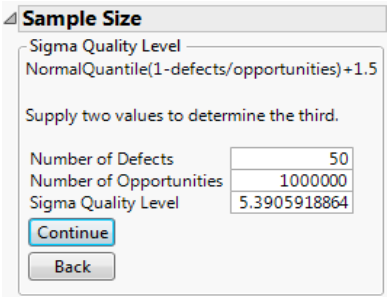
Use the Sigma Quality Level Calculator to explore relationships between the number of defects, number of opportunities for defects, and the Sigma Quality Level. The Sigma Quality Level is a statistic that relates a given defect rate to a six-sigma scale. For example, a defect rate of 3.4 per one million opportunities results in a six-sigma process.

Example of the Sigma Quality Level Calculator

Suppose you want to estimate the Sigma Quality Level for 50 defects in 1,000,000 opportunities at your plant. An opportunity is the unit for the defect count. It could be a single item or you might have multiple defects on a single item where each potential defect is an opportunity.

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click the **Sigma Quality Level**.
3. Enter 50 for the **Number of Defects.**
4. Enter 1000000 as the **Number of Opportunities**.
5. Click **Continue**.

Figure 17.15 Sigma Quality Level Calculator



The Sigma Quality level is approximately 5.4.

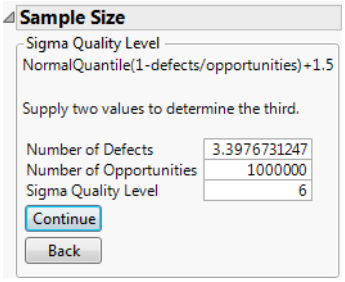
Example Computing the Number of Defects

To calculate the maximum number of defects you can have out of a million opportunities and still have a six-sigma quality level process, follow these steps:

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click the **Sigma Quality Level** button.

3. Leave **Number of Defects** blank.
4. Enter 1000000 as the **Number of Opportunities**.
5. Enter 6 as **Sigma Quality Level**.
6. Click **Continue**.

Figure 17.16 Six-Sigma Quality Level for 1,000,000 Opportunities



The computation shows that the **Number of Defects** cannot be more than 3 in 1,000,000 opportunities to achieve a six-sigma quality level process.

Sigma Quality Level Calculator Fields

Specify two of the following quantities to calculate the third quantity:

Number of Defects The number of defects.

Number of Opportunities The number of opportunities for a defect.

Sigma Quality Level The defect rate in standard deviation units.

Calculator Buttons

Continue Evaluates the missing value.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the Sigma Quality Calculator

Calculations for the sigma quality calculator are based on the definition of sigma quality level σ_q for d defects in n opportunities:

$$\sigma_q = Z_{1-\frac{d}{n}} + 1.5$$

From this definition, d and n can be calculated as follows:

$$d = n[1 - \Phi(\sigma_q - 1.5)]$$

$$n = d[1 - \Phi(\sigma_q - 1.5)]^{-1}$$

where $\Phi()$ is the standard normal cumulative distribution function and $Z_{1-d/n}$ is the $(1 - d/n)^{th}$ quantile from the standard normal distribution.

Reliability Test Plan Calculator

Use the Reliability Test Plan Calculator to determine the sample size, study length, or precision needed for a reliability study. Reliability studies are used to estimate failure times and failure probabilities of a product. The test plan calculator uses expected confidence intervals to define the precision of the estimates. These estimates are based on a fitted failure distribution. To fit a failure distribution, a minimal number of failures must be observed.

Example of the Reliability Test Plan Calculator

A company has developed a new product and wants to calculate how many units to test to estimate the time until 20% of units fail, using a 95% confidence interval with two-sided absolute precision of 200 hours. In other words, when a confidence interval is created for the estimated time, the difference between the upper and lower limits should be approximately 200 hours. The company can run the test for 2,500 hours. In addition, from studies of similar products, they believe the approximate failure distribution to be a Weibull distribution with location parameter $\alpha = 2000$ and scale parameter $\beta = 3$.

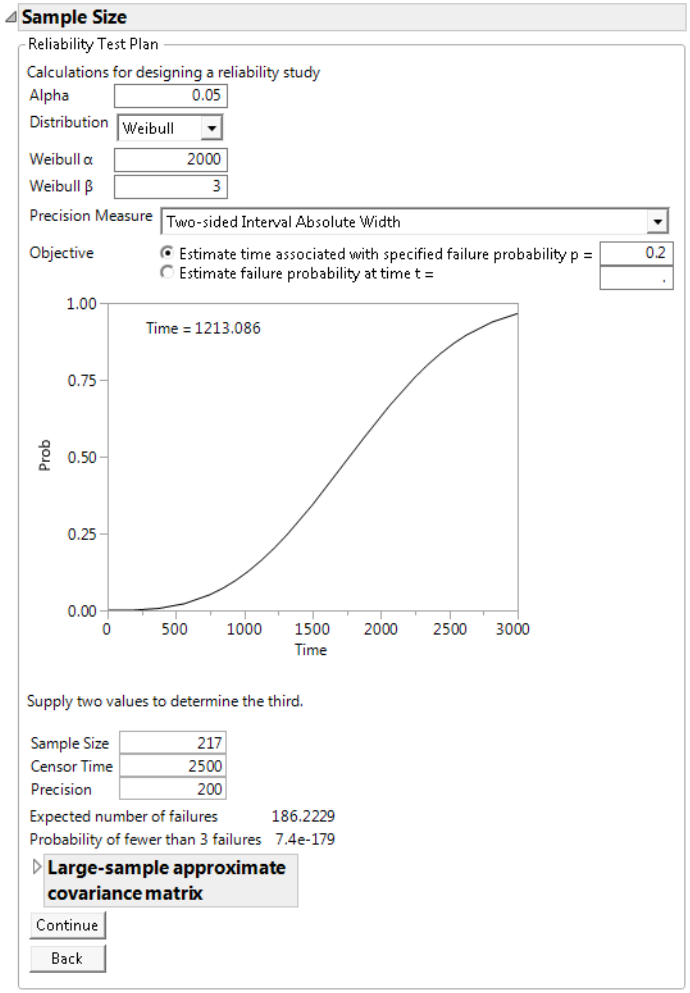
To compute the required sample size:

1. Select **DOE** > **Design Diagnostics > Sample Size and Power**.
2. Click **Reliability Test Plan**.
3. Leave Alpha set at 0.05.
4. Select **Weibull** from the Distribution list.
5. Enter 2000 for the **Weibull** α parameter.
6. Enter 3 for the **Weibull** β parameter.
7. Select **Two-sided Interval Absolute Width** from the Precision Measure list.
8. Select **Estimate time associated with specified failure probability** and enter 0.2 for **p**.

Note: The cumulative distribution function plot is labeled according to the study objective. Here, Time = 1213 is the time estimate for 20% of unit failures.

9. Leave **Sample Size** blank.
10. Enter 2500 for **Censor Time**.
11. Enter 200 for **Precision**.
12. Click **Continue**.

Figure 17.17 Reliability Test Plan Calculator



To estimate the time until 20% of units fail with a precision of 200 hours requires 217 units on test for 2,500 hours. The expected number of failures in this test is approximately 186, much larger than the minimum of 3 needed to estimate the failure distribution.

Reliability Test Plan Calculator Fields

Specify the following quantities and test settings:

Alpha The significance level of the confidence interval used to define the precision measure.

Distribution The assumed failure distribution. Distributions available are: Weibull, Lognormal, Frechet, Loglogistic, SEV, Normal, LEV, and logistic. For more information about these distributions, see *Reliability and Survival Methods*.

Location The location parameter for the failure distribution.

Note: The Location field is denoted Weibull α when the Distribution is set to Weibull.

Scale The scale parameter for the failure distribution.

Note: The Scale field is denoted Weibull β when the Distribution is set to Weibull.

Precision Measure The definition of the precision measure. Definitions are based on the expected confidence interval of the quantity being estimated (failure time or probability). The choices for the precision measure are as follows:

Interval Ratio Defines precision as the square root of the ratio of the upper limit to the lower limit. This ratio is always greater than one since the upper limit is greater than or equal to the lower limit. The interval ratio decreases as the precision in the estimate increases.

Two-sided Interval Absolute Width Defines precision as the width of the confidence interval or the difference between the upper and lower limits.

Lower One-sided Interval Absolute Width Defines precision as the width of the lower side of the interval or the difference between the estimate and the lower limit of the confidence interval for the estimate.

Two-sided Interval Relative Width Defines precision as the width of the confidence interval relative to the estimate. This is the difference between the upper and lower limits divided by the estimate.

Lower One-sided Interval Relative Width Defines precision as the width of the lower side of the interval relative to the estimate. This is the difference between the estimate and the lower limit divided by the estimate.

Objective The objective of the study. Select one of the following objectives and enter the corresponding value:

- Estimate time associated with specified failure probability p.
- Estimate failure probability at time t.

Note: The plot is the cumulative distribution function of the failure distribution. The plot is labeled with the estimate of time or probability based on the study objective.

Specify two of the following quantities to calculate the third quantity:

Sample Size The number of units to include in the reliability test.

Censor Time The amount of time available to run the reliability test.

Precision The level of precision. The definition of the units on this value corresponds to the chosen Precision Measure.

Calculator Buttons

Continue Evaluates the missing value.

Back Returns to the previous Sample Size and Power launch window.

Additional Reliability Test Plan Calculations

In addition to calculating the sample size, censor time, or precision, the following quantities are also calculated:

Expected number of failures The expected number of failures for the specified reliability test.

Probability of fewer than 3 failures The probability that the specified reliability test results in fewer than three failures. This is important because a minimum of three failures is required to obtain stable estimates for the location and scale parameters of the failure distribution. With only one or two failures, the estimates are unstable. If this probability is large, you risk not observing enough failures to reliably estimate the distribution parameters. Increasing the sample size or censor time are both ways to lower the probability of fewer than three failures.

Large-sample approximate covariance matrix Provides the approximate variances and covariance for the location and scale parameters of the failure distribution.

Statistical Details for the Reliability Test Plan Calculator

The reliability test plan is designed to estimate either a quantile:

$$q(p) = \exp[\sigma\Phi^{-1}(p) + \mu]$$

or a failure probability:

$$p(t) = \Phi\left[\frac{\log(t) - \mu}{\sigma}\right]$$

at a specified level of significance with an assumed failure time distribution and precision measure. $\Phi()$ is the standard cumulative distribution function of the assumed failure time distribution with location parameter μ and scale parameter σ. Wald confidence intervals and the precision measure define precision as a function of t (time) and n (sample size). Numerical methods are used to solve for the desired quantity.

For more information about calculations in JMP, see Barker (2011, Section 6).

Reliability Demonstration Calculator

Use the Reliability Demonstration calculator to test a specified number of units for a specified period of time. If fewer than k units fail, the demonstration passes, and you can conclude that the product reliability meets or exceeds a reliability standard. We pose the reliability demonstration as a hypothesis test:

$$H_o: p < p^* \text{ vs } H_a: p \geq p^*$$

where p is the probability of survival time at t^* for the new product and p^* is the standard probability of survival at time t^*.

Example of the Reliability Demonstration Calculator

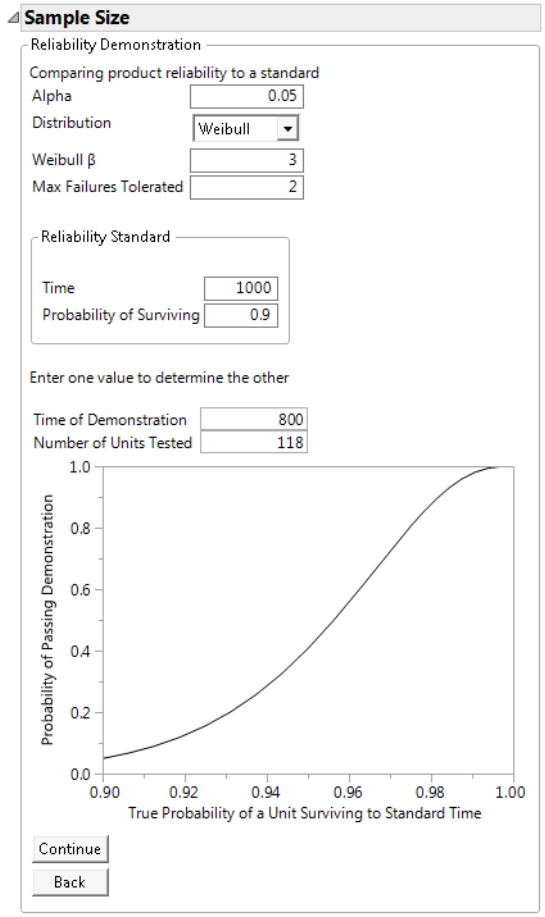
A company wants to estimate the sample size needed for assessing the reliability of a new product against a historical reliability standard of 90% survival after 1,000 hours. From prior studies on similar products, it is believed that the failure time distribution is Weibull, with a β parameter of 3. The company can afford to run the demonstration for 800 hours, and wants the experiment to result in no more than 2 failures.

To compute the required sample size:

1. Select **DOE > Design Diagnostics > Sample Size and Power**.
2. Click **Reliability Demonstration**.
3. Leave Alpha set at 0.05.
4. Select **Weibull** from the Distribution list.
5. Enter 3 for the **Weibull** β.

6. Enter 2 for **Max Failures Tolerated**.
7. Enter 1000 for **Time**.
8. Enter 0.9 for **Probability of Surviving**.
9. Enter 800 for **Time of Demonstration**.
10. Leave **Number of Units Tested** blank.
11. Click **Continue**.

Figure 17.18 Reliability Demonstration Calculator



The company needs to run 118 units in the demonstration. Furthermore, if they observe 2 or fewer failures by 800 hours, they can conclude that the new product reliability is at least as reliable as the standard.

The plot shows that as the true probability of a unit surviving to the time in the reliability standard increases, the probability of passing the demonstration increases. In this example, if a unit has only a 92% chance of surviving to the standard time, then the chance of the new product passing the demonstration is only about 12.5%.

Reliability Demonstration Calculator Fields

Specify the following quantities and test settings:

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Distribution The assumed failure time distribution. Distributions available are: Weibull, Lognormal, Frechet, Loglogistic, SEV, Normal, LEV, and logistic. For more information about these distributions, see *Reliability and Survival Methods*.

Scale The scale parameter for the failure time distribution.

Note: The Scale field is denoted Weibull β when the specified Distribution is Weibull.

Max Failures Tolerated The maximum number of failures allowed for a successful test demonstration.

Reliability Standard The definition of the standard that you are testing. This standard is based on a length of time and the probability of an items survival to that time.

Time The minimum length of time that an item under test should survive.

Probability of Surviving The probability that the item under test survives until the defined Time of the reliability standard

Enter one of the following two values to calculate the other:

Time of Demonstration The length of the demonstration in time.

Number of Units Tested The number of units needed for the demonstration.

Calculator Buttons

Continue Evaluates the missing value.

Back Returns to the previous Sample Size and Power launch window.

Statistical Details for the Reliability Demonstration Calculator

The reliability demonstration depends on the assumed failure time distribution with scale parameter σ. The reliability standard, or probability of survival at time t and location μ is stated as follows:

$$p = 1 - \Phi\left[\frac{\log t - \mu}{\sigma}\right]$$

where μ is solved for using the following:

$$\mu = \log(t) - \sigma\Phi^{-1}(1-p) \quad .$$

To calculate sample size and the size of the test, the probability of survival at time t is posed as a hypothesis test:

$$H_o: p < p^* \text{ vs } H_a: p \geq p^*$$

where p^* is the standard probability of survival at time t^*.

We want to test the hypothesis at the α level or as follows:

$\alpha = \Pr(k \text{ or few failures} \mid H_0 \text{ true})$.

Since the test is of n independent units, the number of failures has a binomial (n, p) distribution where p is the probability of a unit failing before time t. Therefore, we can express α as a function of t and n:

$$\alpha = \Pr\left(x \leq k \mid x \sim \text{Bin}\left\{n, \Phi\left[\frac{\log t - \mu^*}{\sigma^*}\right]\right\}\right)$$

where μ* and σ* are from the assumed reliability standard.

Properties of the binomial and beta distributions result in being able to solve for t using:

$$t = \frac{t^* \exp\{\sigma^* \Phi^{-1}[1 - \text{B}^{-1}(\alpha; n-k, k+1)]\}}{\exp[\sigma^* \Phi^{-1}(1-p^*)]}$$

For n, Brent's method is used to find the root of:

$$B^{-1}(\alpha; n-k, k+1) - 1 + \Phi\left[\frac{\log t - (\log t^*) + \sigma^* \Phi^{-1}(1-p^*)}{\sigma^*}\right] = 0$$

where:

$B^{-1}(\alpha; n-k, k+1)$ is the α quantile of the Beta$(n-k; k+1)$ distribution

and $\Phi()$ is the cumulative distribution function of the assumed failure time distribution.

For more information about calculations in JMP, see Barker (2011, Section 5).

Chapter 18

Choice Designs

Create a Design for Selecting Preferred Product Profiles

A choice (or *discrete choice*) experiment provides data for modeling discrete preferences. Study participants are presented with sets of potential products (or product profiles) with varying attributes. From each set of profiles, a participant selects a preferred profile. For example, in designing a high-end laptop, a computer company might be interested in the relative importance of key features such as: processor speed, hard disk size, screen size, battery life, and price. A choice experiment addresses the relative values of these features to a customer and indicates an optimal set of trade-offs among product features.

The results of a choice experiment are analyzed using conjoint analysis methods. See *Consumer Research*.

Figure 18.1 A Survey with Eight Choice Sets

	Choice Set	Choice ID	Disk Size	Speed	Battery Life	Price
1	1	1	40 GB	2.0 GHz	6 Hrs	$1000
2	1	2	80 GB	1.5 GHz	4 Hrs	$1500
3	2	1	80 GB	1.5 GHz	4 Hrs	$1200
4	2	2	40 GB	1.5 GHz	6 Hrs	$1500
5	3	1	40 GB	1.5 GHz	4 Hrs	$1200
6	3	2	80 GB	2.0 GHz	6 Hrs	$1500
7	4	1	40 GB	2.0 GHz	4 Hrs	$1000
8	4	2	80 GB	1.5 GHz	6 Hrs	$1500
9	5	1	80 GB	2.0 GHz	6 Hrs	$1200
10	5	2	40 GB	1.5 GHz	6 Hrs	$1000
11	6	1	40 GB	1.5 GHz	4 Hrs	$1500
12	6	2	80 GB	1.5 GHz	6 Hrs	$1000
13	7	1	80 GB	1.5 GHz	4 Hrs	$1000
14	7	2	40 GB	2.0 GHz	6 Hrs	$1200
15	8	1	80 GB	1.5 GHz	6 Hrs	$1200
16	8	2	80 GB	2.0 GHz	4 Hrs	$1500

Contents

Overview of Choice Designs 549
Choice Design Terminology 549
Bayesian D-Optimality 550
Example of a Choice Design 550
Example of a Choice Design with Analysis 553
Create a Choice Design for a Pilot Study 553
Analyze the Pilot Study Data 555
Design the Final Choice Experiment Using Prior Information 556
Run the Design and Analyze the Results 558
Choice Design Window 561
Attributes 561
Model 562
Design Generation 563
Design 564
Make Table 566
Choice Design Options 566
Technical Details 567
Bayesian D-Optimality and Design Construction 567
Utility-Neutral and Local D-Optimal Designs 568

Overview of Choice Designs

Discrete choice experiments support the process of designing a product. They help prioritize product features for a company's market so that the company can design a product that people want to buy. The Choice Design platform creates experiments using factors that are attributes of a product. Selecting the attributes to be studied and their values is of critical importance. You must include all attributes that are likely to influence a consumer's decision to buy the product. For more information and guidelines for designing a choice experiment, see Sall (2008).

Choice Design Terminology

The following terminology is associated with choice designs:

- An *attribute* is a feature of a product.
- A *profile* is a specification of product attributes.
- A *choice set* is a collection of profiles.
- A *survey* is a collection of choice sets.
- A *partial profile* is a profile in a choice design where only a specified number of attributes are varied within each choice set. The remaining attributes are not varied.

In a discrete choice experiment, respondents are presented with a survey containing several choice sets. Choice sets usually contain only a small number of profiles to facilitate the decision process. Within each choice set, each respondent specifies which of the profiles he or she prefers. For example, attributes for a laptop experiment might include speed, storage, screen size, battery life, and price. Different combinations of these attributes comprise product profiles. A choice set might consist of two profiles. From each choice set, a respondent chooses the profile that he or she prefers.

In cases where many attributes are involved, you can construct surveys where each choice set contains *partial profiles*. In a choice set with partial profiles, only a specified number of attributes are varied and the remaining attributes are held constant. This reduces the complexity of the choice task.

Bayesian D-Optimality

Because discrete choice models are nonlinear in their parameters, the efficiency of a choice design depends on the unknown parameters. The Choice Design platform uses a Bayesian approach, optimizing the design over a prior distribution of likely parameter values that you specify. The Bayesian D-optimality criterion is the expected logarithm of the determinant of the information matrix, taken with respect to the prior distribution. The Choice Design platform maximizes this expectation with respect to the prior probability distribution. See "Bayesian D-Optimality and Design Construction" and Kessels et al. (2011).

You can also generate the following types of designs:

- Utility-neutral designs - In a utility-neutral design, all choices within a choice set are equally probable. The prior mean is set to 0.
- Local D-optimal designs - A local D-optimal design takes into account the prior on the mean, but does not include any information from a prior covariance matrix.

For more information about utility neutral and local D-optimal designs, see "Utility-Neutral and Local D-Optimal Designs".

Example of a Choice Design

About the Experiment

In this example, a coffee shop is interested in making an ideal cup of coffee to satisfy the majority of its customers. The manager has asked you to determine which factors affect customer preferences. Specifically, you need to determine which settings of the following factors (attributes) result in an ideal cup of coffee:

- Grind size (medium or coarse)
- Temperature (195°, 200°, 205°)
- Brewing time (3 minutes, 3.5 minutes, or 4 minutes)
- Charge (1.6 grams/ounce, 2 grams/ounce, or 2.4 grams/ounce)

Each combination of factor levels is a *profile*. Trying to obtain information about preferences by having every respondent sample every possible profile is not practical. However, you can ask a respondent to select a preferred profile from a *choice set* consisting of a small number of profiles.

In this example, you design an experiment where each respondent indicates his or her preference in several choice sets. Your design will have the following structure:

- ten respondents

- twelve choice sets
- two profiles per choice set
- one survey per respondent containing all 12 choice sets

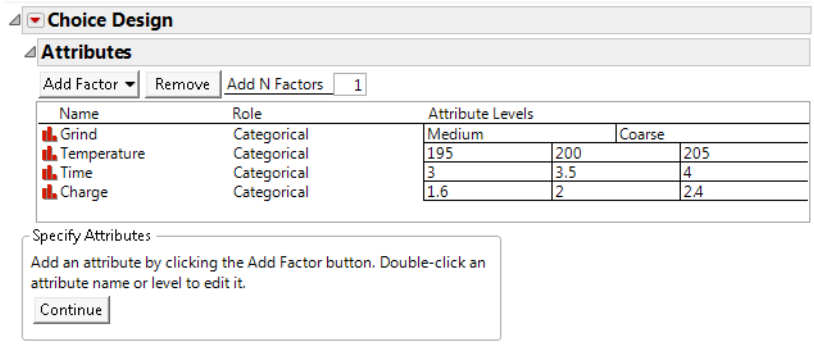
The experiment results in 12 responses per respondent. Analysis of these preferences can be used to draw conclusions about how to make a cup of coffee that pleases most customers.

Create the Design

You can enter factors either manually or automatically using a preexisting table that contains the factors and settings. In this example, for convenience, you use a preexisting table. But, if you are designing a new experiment, you must first enter the factors manually. For more information about entering factors manually, see "Attributes".

1. Select **DOE** > **Consumer Studies > Choice Design**.
2. Select **Help > Sample Data Library** and open Design Experiment/Coffee Choice Factors.jmp.
3. Click the Choice Design red triangle and select **Load Factors**.

Figure 18.2 Choice Design Window with Attributes Defined



4. Click **Continue**.
5. (Optional) Open the DOE Model Controls outline.

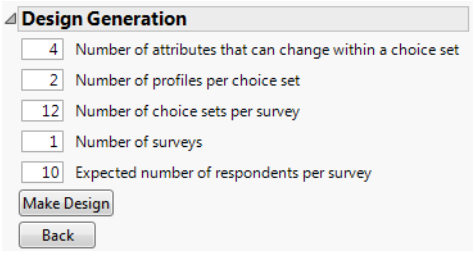
   Notice that there only main effects. In this example, you are only interested in a model that contains the main effects of your four factors. However, if you wanted your design to be capable of estimating additional effects, you add them in this outline.
6. In the Design Generation panel:
   - Keep the **Number of attributes that can change within a choice set** at 4.
   - Keep the **Number of profiles per choice set** at 2.
   - Type 12 for the **Number of choice sets per survey**.

     In this example, the respondents evaluate 12 choice sets.

- Keep the **Number of surveys** at 1.
- Type 10 for the **Expected number of respondents per survey**.

  In this example, there are ten respondents.

Figure 18.3 Completed Design Generation Panel



Note: Setting the Random Seed in step 7 reproduces the exact results shown in this example. In constructing a design on your own, this step is not necessary.

7. (Optional) Click the Choice Design red triangle and select **Set Random Seed**. Type 12345 and click **OK**.
8. Click **Make Design**.

   There are 12 choice sets, each consisting of two coffee profiles.
9. Select **Output separate tables for profiles and responses**.

   This places the descriptions of the choice sets in a data table called Choice Profiles. A second data table (called Choice Runs) is constructed to facilitate entry of the response information.
10. Click **Make Table**.

    The Choice Profiles table shows the 12 choice sets, each consisting of two profiles. The Choice Runs table enables you to record the preferred profile in the column Response using the profile ID. Enter 1 if Choise1 is the preferred profile or 2 if Choice 2 is the preferred profile. Alternatively, if the respondent has no preference, leave the response missing.

    The **Choice** script in the Choice Profiles table facilitates analysis of experimental results. It opens a completed launch window for a Choice Model. For information about Choice Models, see *Consumer Research*.

    The **DOE Dialog** script in the Choice Profiles table relaunches the design dialog.

Example of a Choice Design with Analysis

In this example, a computer manufacturer is interested in manufacturing a new laptop and wants information about customer preferences before beginning an expensive development process. The manufacturer decides to construct a design consisting of two sets of profiles that will be administered to ten respondents. The goal of the choice design is to understand how potential laptop purchasers view the advantages of a collection of four attributes:

- size of hard drive disk (40 GB or 80 GB)
- speed of processor (1.5 GHz or 2.0 GHz)
- battery life (4 Hrs or 6 Hrs)
- cost of computer ($1000, $1200 or $1500)

To construct the design for the ten respondents, you first conduct a small pilot study using only one respondent. Then you analyze the results and use the parameter estimates as prior information in designing the final study for the ten respondents.

Create a Choice Design for a Pilot Study

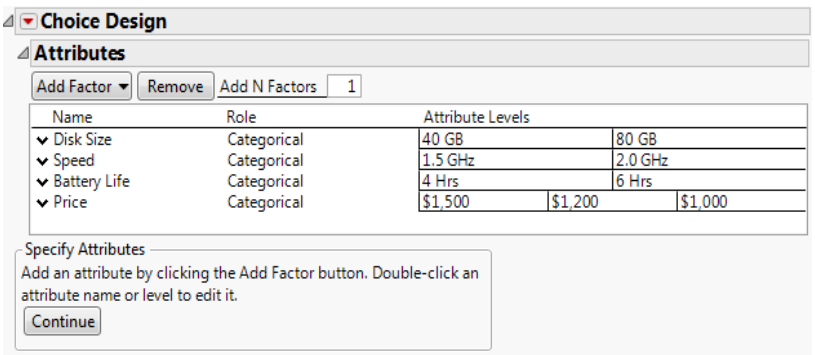
In this section, you construct a choice design for a one-respondent study.

Define Factors and Levels

In this example, you load the factors from an existing table. When designing a new experiment on your own, enter the factors manually.

1. Select **DOE > Consumer Studies > Choice Design**.
2. Select **Help > Sample Data Library** and open Design Experiment/Laptop Factors.jmp.
3. Click the Choice Design red triangle and select **Load Factors**.

Figure 18.4 Choice Design Window with Attributes Defined

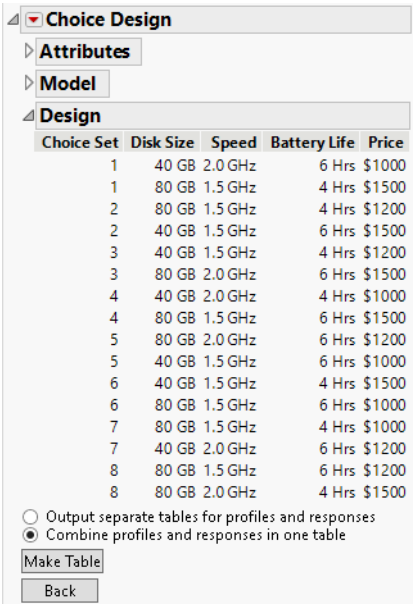


Create the Design

1. Click **Continue**.
2. This pilot survey will be given to a single respondent. The default values in the DOE Model Controls, Prior Specification, and Design Generation panels are appropriate as is.
3. (Optional) Click the Choice Design red triangle and select **Set Random Seed**. Type 12345 and click **OK**.
4. Click **Make Design**.

Note: Setting the Random Seed in step 3 reproduces the exact results shown in this example. In constructing a design on your own, this step is not necessary.

Figure 18.5 Pilot Design



Choice Set	Disk Size	Speed	Battery Life	Price
1	40 GB	2.0 GHz	6 Hrs	$1000
1	80 GB	1.5 GHz	4 Hrs	$1500
2	80 GB	1.5 GHz	4 Hrs	$1200
2	40 GB	1.5 GHz	6 Hrs	$1500
3	40 GB	1.5 GHz	4 Hrs	$1200
3	80 GB	2.0 GHz	6 Hrs	$1500
4	40 GB	2.0 GHz	4 Hrs	$1000
4	80 GB	1.5 GHz	6 Hrs	$1500
5	80 GB	2.0 GHz	6 Hrs	$1200
5	40 GB	1.5 GHz	6 Hrs	$1000
6	40 GB	1.5 GHz	4 Hrs	$1500
6	80 GB	1.5 GHz	6 Hrs	$1000
7	80 GB	1.5 GHz	4 Hrs	$1000
7	40 GB	2.0 GHz	6 Hrs	$1200
8	80 GB	1.5 GHz	6 Hrs	$1200
8	80 GB	2.0 GHz	4 Hrs	$1500

The single survey contains eight choice sets, each consisting of two laptop profiles.

5. Verify that the **Combine profiles and responses in one table** option is selected.

   This places the choice sets and the survey results in the same table.
6. Click **Make Table**.

This survey was designed assuming no prior information. For this reason, some choice sets might not elicit useful information. The plan is to obtain survey results from the single respondent, analyze the results, and then use the results from the pilot survey as prior information in designing the final survey.

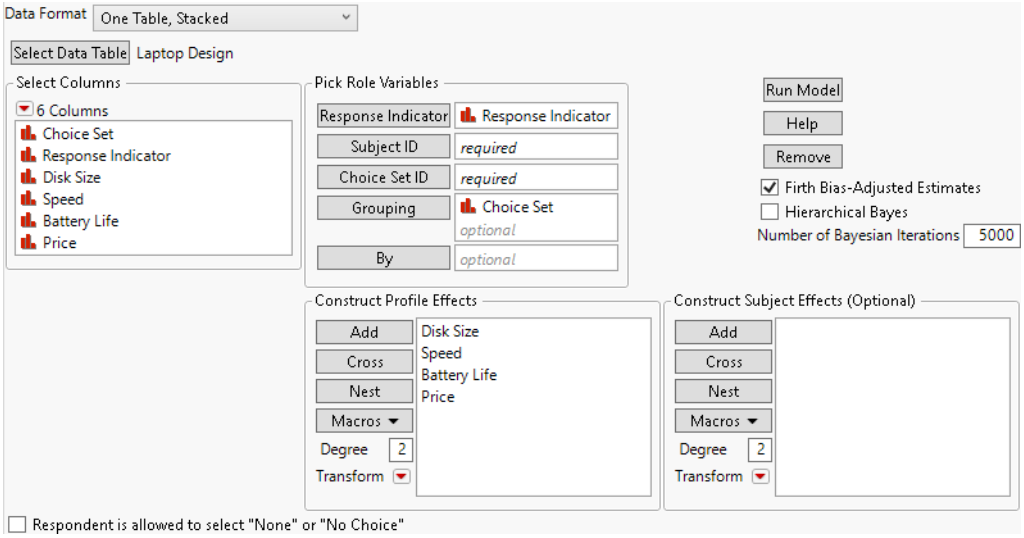
Analyze the Pilot Study Data

Now that the pilot survey design is complete, it is administered to single respondent. The respondent chooses one profile from each set, entering 1 for the chosen profile and 0 for the rejected profile. You will analyze the results using the Choice platform.

Note: For more information about the Choice platform, see *Consumer Research*.

1. Select **Help > Sample Data Library** and open Design Experiment/Laptop Design.jmp.
2. Run the **Choice** script.

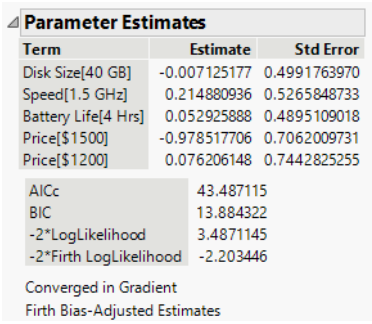
Figure 18.6 Choice Model Launch Window



The only grouping variable is Choice Set because there is a single survey and a single respondent.

3. Click **Run Model**.

Figure 18.7 Parameter Estimates for Pilot Survey



Parameter Estimates

Term	Estimate	Std Error
Disk Size[40 GB]	-0.007125177	0.4991763970
Speed[1.5 GHz]	0.214880936	0.5265848733
Battery Life[4 Hrs]	0.052925888	0.4895109018
Price[$1500]	-0.978517706	0.7062009731
Price[$1200]	0.076206148	0.7442825255

AICc	43.487115
BIC	13.884322
-2*LogLikelihood	3.4871145
-2*Firth LogLikelihood	-2.203446

Converged in Gradient
Firth Bias-Adjusted Estimates

To construct the final choice design that you will give to ten respondents, you need prior means and variances for the parameter estimates. The analysis in Figure 18.7 gives estimates of the parameter means (Estimate) and estimates of their standard errors (Std Error). You will treat the standard errors as prior estimates of the standard deviations. Next, to calculate estimates of the variances of the attributes, construct a JMP table and square the standard errors.

4. Right-click in the Parameter Estimates report and select **Make into Data Table**.
5. In the new data table, right-click in the Std Error column header and select **New Formula Column** > **Transform** > **Square**.

   A column called Std Error^2 is added to the data table. Its values will serve as your estimates of the prior variance for the choice model parameters.

Figure 18.8 Untitled Data Table with Variance Estimates in Last Column

	Term	Estimate	Std Error	Std Error^2
1	Disk Size[40 GB]	-0.007125177	0.4991763970	0.2491770753
2	Speed[1.5 GHz]	0.214880936	0.5265848733	0.2772916288
3	Battery Life[4 Hrs]	0.052925888	0.4895109018	0.2396209229
4	Price[$1500]	-0.978517706	0.7062009731	0.4987198144
5	Price[$1200]	0.076206148	0.7442825255	0.5539564778

Note: Do not close the Untitled data table at this point.

Design the Final Choice Experiment Using Prior Information

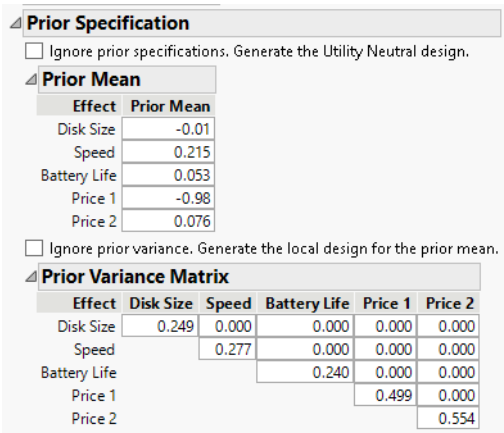
In this section, you use the prior information obtained from the pilot laptop study to construct a final design. The final design will be administered to a set of ten participants.

1. Select **Help > Sample Data Library** and open Design Experiment/Laptop Factors.jmp.
2. Select **DOE** > **Consumer Studies > Choice Design**.
3. Click the Choice Design red triangle and select **Load Factors**.

4. Click **Continue**.
5. From the Untitled table, enter the values in the Estimate column into the Prior Mean outline in the Choice Design window (Figure 18.9).

   You can copy and paste the entire column from the Untitled table, then click in the Disk Size text box under **Prior Mean** in the Prior Mean outline, right-click, and select Paste.
6. From the Untitled table, enter the values in the Std Error^2 column into the diagonal entries in the Prior Variance Matrix outline in the Choice Design window (Figure 18.9). Enter these one-by-one, rounded to three decimal places.

Figure 18.9 Prior Mean and Variance Information from Pilot Study



7. In the Design Generation panel, enter 2 for the **Number of surveys** and five for the **Expected number of respondents per survey**.

   This gives instruments for a total of 10 respondents and allows for two different sets of profiles.
8. Click **Make Design**.
9. Click **Make Table**.

   The design table has 160 rows. There are 16 rows for each of the ten study respondents. Each respondent has 8 choice sets, each with 2 profiles. There are two surveys, each given to 5 respondents. The 160 rows result from the following calculation: 2 profiles * 8 choice sets * 2 surveys * 5 respondents = 160 rows.

   The final design is now ready to be administered to the 10 respondents.

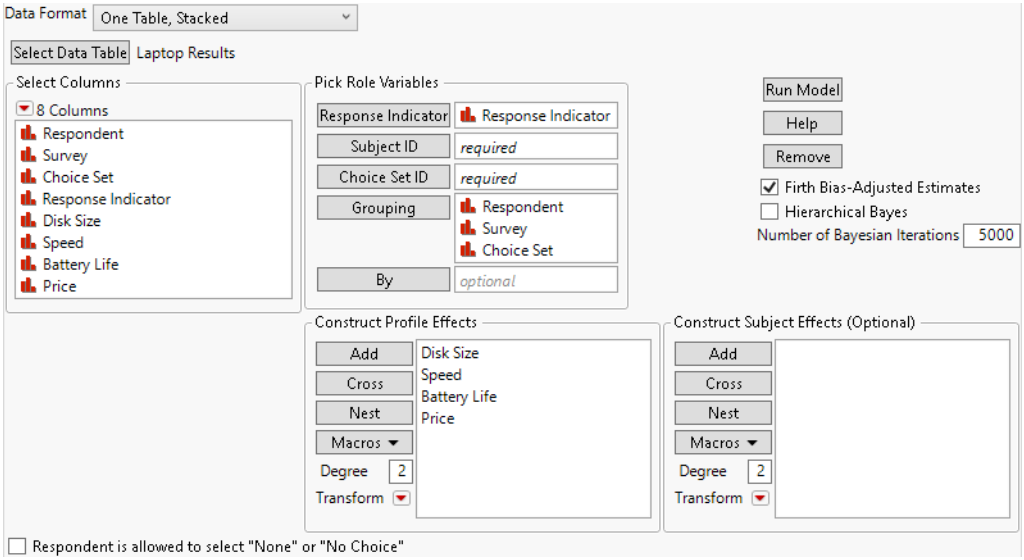
Run the Design and Analyze the Results

In this section, you analyze the results obtained when you obtain results from the final design. In particular, you want to know how changing the price or other characteristics of a laptop affects its desirability as perceived by potential buyers. This desirability is called the *utility value* of the laptop attributes.

Determine Significant Attributes

1. Select **Help > Sample Data Library** and open Design Experiment/Laptop Results.jmp.
2. Run the **Choice** script.

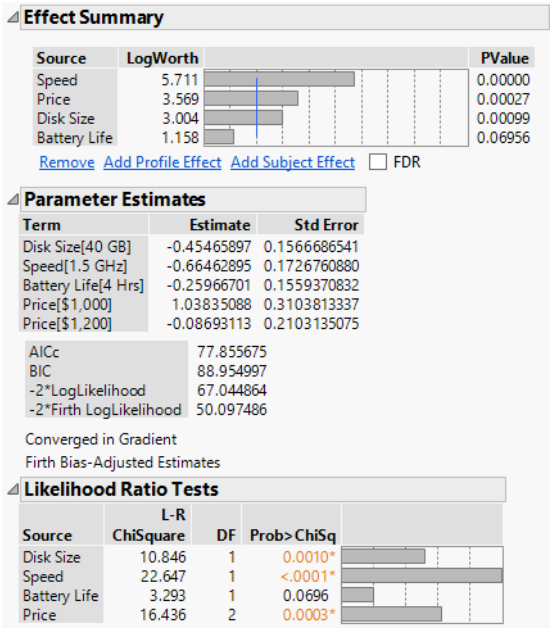
Figure 18.10 Choice Model Launch Window



There are three grouping variables, Respondent, Survey, and Choice Set, because there are multiple surveys and respondents.

3. Click **Run Model**.

Figure 18.11 Initial Analysis of the Final Laptop Design



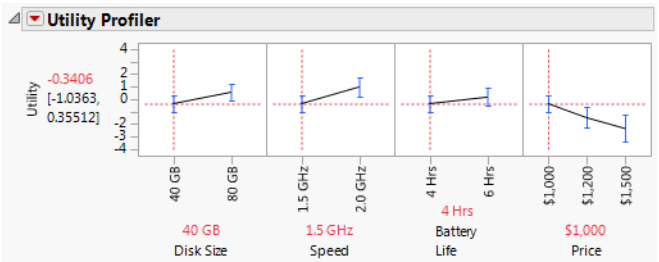
The Effect Summary and Likelihood Ratio Tests outlines indicate that Disk Size, Speed, and Price are significant at the 0.05 level, and that Battery Life is marginally significant.

Find Unit Cost and Trade Off Costs

Next, use the Profiler to see the utility value and how it changes as the laptop attributes change.

1. Click the Choice Model red triangle and select **Utility Profiler**.

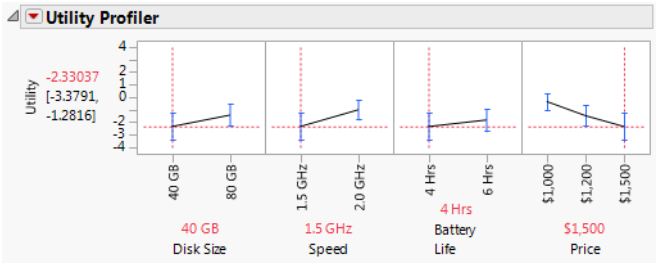
Figure 18.12 Utility Profiler at Price = $1000



When each attribute value is set to its lowest value, the Utility value is –0.3406. The first thing that you want to do is determine the unit utility cost.

2. Move the slider for Price to $1,500.

Figure 18.13 Utility Profiler at Price = $1500

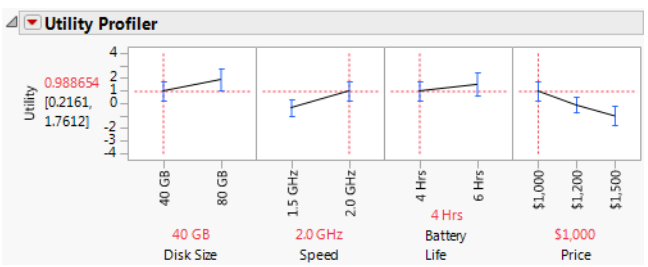


When Price changes from $1,000 to $1,500, the Utility changes from –0.3406 to –2.3303. That is, raising the price of a laptop by $500.00 lowers the utility (or desirability) approximately 2 units. Therefore, you can estimate the unit utility cost to be approximately $250.00.

With this unit utility cost estimate, you can now vary the other attributes, note the change in utility, and find an approximate monetary value associated with attribute changes. For example, the most significant attribute is Speed (Figure 18.11).

3. In the Utility Profiler, set Price back to $1,000, its lowest value, and change Speed to 2.0 GHz, its higher value.

Figure 18.14 Utility Value of Higher Speed



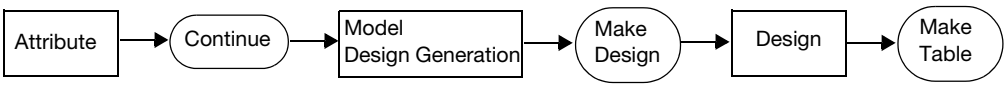
The Utility value changes from the original value shown in Figure 18.12 of –0.3406 to 0.9886, for a total change of 1.3292 units. Using the utility cost estimate or $250.00, the increase in price for a 2.0 GHz laptop over a 1.5 GHz laptop can be computed to be 1.3292*$250.00 = $332.30. This is the dollar value that the Choice study indicates that the manufacturer can use as a basis for pricing this laptop attribute. You can make similar calculations for the other attributes.

Choice Design Window

The Choice Design window walks you through the steps to construct a choice design for modeling attribute preferences. You can specify an assumed model and prior information.

The Choice Design window updates as you work through the design steps. The outlines, separated by buttons that update the outlines, follow the flow in Figure 18.15.

Figure 18.15 Choice Design Flow

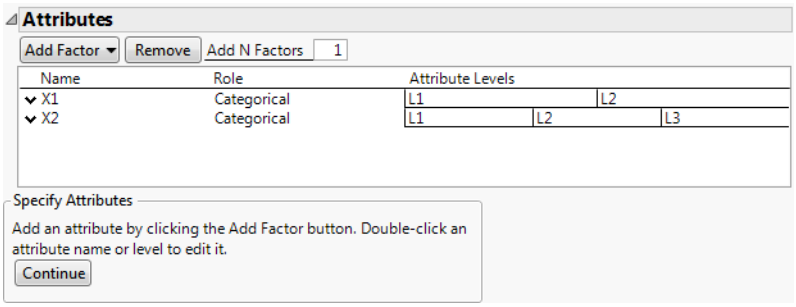


Attributes

Attributes in a choice design can only be categorical.

Tip: When you have completed the Attributes outline, consider selecting **Save Factors** from the red triangle menu. This saves the attribute names, roles, and levels in a data table that you can later reload and reuse.

Figure 18.16 Attributes Outline



The Attributes outline contains the following buttons:

Add Factor Adds an attribute with the selected number of levels.

Remove Removes the selected attributes.

Add N Factors Adds multiple attributes. Enter the number of attributes to add, click **Add Factor,** and then select the number of levels. Repeat **Add N Factors** to add multiple attributes with different numbers of levels.

The Attributes outline contains the following columns:

Name The name of the attribute. Attributes are given default names of X1, X2, and so on. To change a name, double-click it and type the desired name.

Role Specifies the Design Role of the attribute as Categorical.

Attribute Levels The attribute name or description. To insert Attribute Levels, click the default levels and type the desired names.

Editing the Attributes Outline

- To edit the Name of an attribute, double-click the attribute name.
- To edit an Attribute Level, click the level.

Attribute Column Properties

For each attribute, JMP saves the Value Order column property to the data tables constructed by the Choice platform. The Value Order column property specifies that levels appear in reports using the ordering specified in the Attributes outline. See "Value Order".

Model

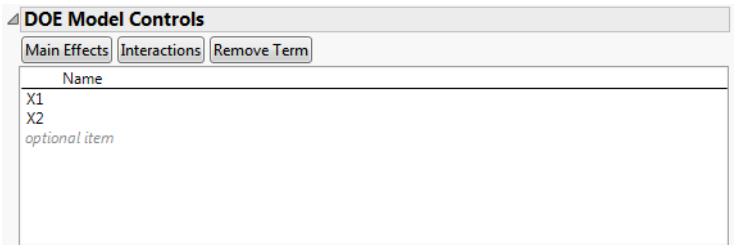
The model outline consists of two parts:

- You can specify your assumed model, which contains all the effects that you want to estimate. See "DOE Model Controls".
- You can specify prior knowledge about the attribute levels, which can result in a better design. See "Prior Specification".

DOE Model Controls

Specify your assumed model in the DOE Model Controls outline. All main effects are included by default. Click the Interactions button to add all two-way interactions.

Figure 18.17 DOE Model Controls Outline



When you construct your design table, JMP saves a Choice script to the data table. The Choice script launches the Choice Platform with the model selected in the DOE Model Controls outline.

The DOE Model Controls outline contains the following buttons:

Main Effects Adds main effects for all attributes in the model.

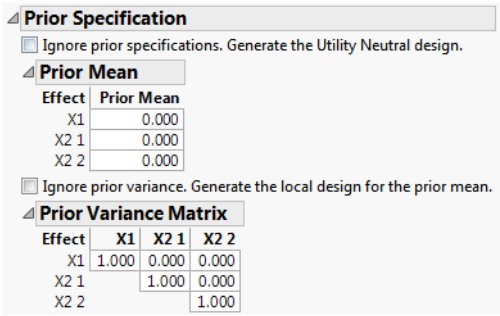
Interactions Adds all second-order interactions. If you do not want to include all of the interactions, select the interactions that you want to remove and click **Remove Term**.

Remove Term Removes selected effects.

Prior Specification

Enter specifications for a multivariate normal prior distribution on the model parameters. Enter the prior distribution's mean in the Prior Mean outline and its covariance matrix in the Prior Variance outline.

Figure 18.18 Prior Specification Outline



You can ignore the prior specifications using these options:

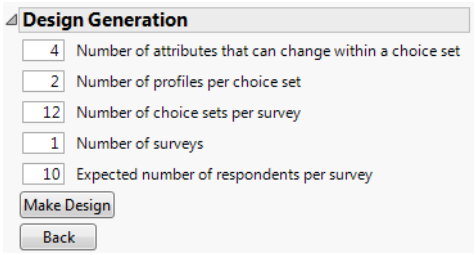
Ignore prior specifications. Generate the Utility Neutral design. Sets the prior means to 0 and generates a locally D-optimal design. This design is called a *utility-neutral* design. See Huber and Zwerina (1996).

Ignore prior variance. Generate the local design for the prior mean. Generates a locally D-optimal design. The local design takes into account the prior on the mean but ignores the covariance matrix. See Huber and Zwerina (1996).

Design Generation

Enter specifications that define the structure for your design in the Design Generation panel.

Figure 18.19 Design Generation Panel for Coffee Example



Note: Figure 18.19 is taken from the coffee example. See "Example of a Choice Design".

Enter a number for each of the following items:

Number of attributes that can change within a choice set Enter a number less than or equal to the total number of attributes. This is often set to the total number of attributes. However, if you are comparing many attributes and want to simplify the selection process for respondents, set this to a number that is smaller than the total number of attributes.

Number of profiles per choice set Enter the number of profiles that a respondent must choose from in stating a preference.

Number of choice sets per survey Enter the number of preferences that you want to obtain from each respondent.

Number of surveys Enter the number of distinct collections of choice sets. This is useful if you want to administer surveys to multiple respondents.

Expected number of respondents per survey Enter the total number of respondents divided by the number of surveys.

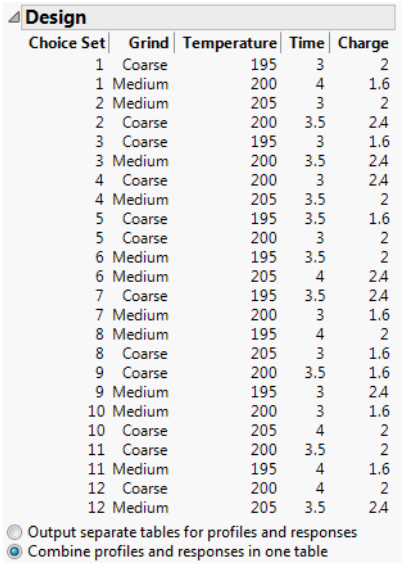
Make Design

Once you have completed the Design Generation outline, click **Make Design** to generate the design. The design appears in the Design outline.

Design

The Design outline shows the runs for a design that is optimal, given your selections. Review the design to ensure that it meets your needs.

Figure 18.20 Design Outline for Coffee Example



Design

Choice Set	Grind	Temperature	Time	Charge
1	Coarse	195	3	2
1	Medium	200	4	1.6
2	Medium	205	3	2
2	Coarse	200	3.5	2.4
3	Coarse	195	3	1.6
3	Medium	200	3.5	2.4
4	Coarse	200	3	2.4
4	Medium	205	3.5	2
5	Coarse	195	3.5	1.6
5	Coarse	200	3	2
6	Medium	195	3.5	2
6	Medium	205	4	2.4
7	Coarse	195	3.5	2.4
7	Medium	200	3	1.6
8	Medium	195	4	2
8	Coarse	205	3	1.6
9	Coarse	200	3.5	1.6
9	Medium	195	3	2.4
10	Medium	200	3	1.6
10	Coarse	205	4	2
11	Coarse	200	3.5	2
11	Medium	195	4	1.6
12	Coarse	200	4	2
12	Medium	205	3.5	2.4

Output separate tables for profiles and responses
Combine profiles and responses in one table

Note: Figure 18.20 is taken from the coffee example. See "Example of a Choice Design".

Note: The algorithm for finding an optimal design is based on a random starting design. Because of this, the design you obtain is not unique. The design algorithm will generate different designs when you click the **Back** and **Make Design** buttons repeatedly.

Output Options

Select one of the following output options:

Output separate tables for profiles and responses Displays two data tables:

- The Choice Profiles table lists the profiles in each row, identified by Survey and Choice Set columns. Within a choice set, the profiles are identified by Choice ID. This table is useful for constructing the survey instruments.
- The Choice Runs table provides an empty Response column where you can enter respondent preferences. Each row corresponds to a single choice set. The rows are sorted by Respondent, Survey, and Choice Set. The choice set IDs are given in the next columns, followed by the Response column. Enter the choice set ID for the respondent's preference in the Response column.

Combine profiles and responses in one table Provides a single Choice Profiles table with an empty Response Indicator column where you can enter respondent preferences. Each row corresponds to a single profile. The table is sorted by Respondent, Survey, and

Choice Set. Enter the value 1 (or another nonzero numerical indicator) for the respondent's preferred profile and a 0 indicator for the other profiles in that choice set.

Note: The values you enter in the Response Indicator column must be numerical.

Make Table

Click **Make Table** to construct the table or tables that you selected in "Output Options". In the table panel of the Choice Profiles table, there is a **Choice** script. Run the script and then click **Run Model** to analyze your experimental results.

For information about the Choice Model report, see *Consumer Research*.

Choice Design Options

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response probabilities and a column containing a choice simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A new column that contains a simulated choice response formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of the marginal utilities for model effects.
- A script called **Choice Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the marginal utilities.

When you click Apply, the formula for the simulated choice response values is updated in the Choice Simulated column. If you click Apply again, the formula and values in this column are updated.

See "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see *Basic Analysis*.

Number of Starts Enables you to specify the number of random starts used in constructing the design. See "Bayesian D-Optimality and Design Construction".

Advanced Options Not available for Choice Designs.

Save Script to Script Window Creates the script for the design that you specified in the Choice Design window and saves it in an open script window.

Technical Details

Bayesian D-Optimality and Design Construction

The Bayesian D-optimality criterion is the expected logarithm of the determinant of the information matrix of the maximum likelihood of the parameter estimators in the multinomial logit model, taken with respect to the prior distribution. The Choice Design platform maximizes this expectation with respect to a sample of parameter vectors that represents the prior probability distribution. See Kessels et al. (2011).

For partial profile designs, JMP uses a two-stage design algorithm:

1. The constant attributes in each choice set are determined using an *attribute balance* approach.
2. The levels of the non-constant attributes are determined using Bayesian D-optimality.

Attribute balance means that the algorithm attempts to balance the number of times each attribute is held constant in the entire design. If two or more attributes are held constant, the algorithm attempts to balance the occurrence of pairs of attributes held constant in the design.

The levels of the non-constant attributes are determined to optimize the Bayesian D-optimal criterion. A random starting design is found. Then levels of the non-constant attributes are generated using a coordinate-exchange algorithm and evaluated until the Bayesian D-optimality criterion is optimized. The calculations, which involve integration with respect to a multivariate normal prior, use the quadrature method described in Gotwalt et al. (2009).

Note: The Bayesian D-optimality criterion can result in choice sets where some non-constant attributes have identical levels. This situation occurs when varying the non-constant levels within a profile would result in uninformative choice sets where all profiles have very high or very low probabilities.

Utility-Neutral and Local D-Optimal Designs

You can use the Choice Design platform to generate a utility-neutral design by setting prior means to 0. In a utility-neutral design, all choices within a choice set are equally probable. See Huber and Zwerina (1996).

You can also generate a local D-optimal design. The local design takes into account the prior of the mean, but does not include any information from a prior covariance matrix. See Huber and Zwerina (1996).

Chapter 19

MaxDiff Design

Create a Design for Selecting Best and Worst Items

MaxDiff (*maximum difference scaling*) studies are an alternative to studies that use standard preference scales to determine the relative importance of items being rated. In a MaxDiff study, a respondent reports only the most and least preferred options from among a small set of choices. This forces respondents to rank options in terms of preference, which often results in rankings that are more definitive than rankings obtained using standard preference scales.

Use the MaxDiff platform when you need to construct a design consisting of choice sets for one factor that can be presented to respondents as part of a MaxDiff study. Conduct your study and then analyze your data use the MaxDiff analysis platform.

Figure 19.1 A MaxDiff Design Table

	Subject	Choice Set	Candy	Choice
1	1	1	Reese's Cups	•
2	1	1	Hershey Bar	•
3	1	1	Snickers	•
4	1	1	Butterfinger	•
5	1	2	Butterfinger	•
6	1	2	Heath Bars	•
7	1	2	Plain M&Ms	•
8	1	2	Snickers	•
9	1	3	Plain M&Ms	•
10	1	3	Snickers	•
11	1	3	Peanut M&Ms	•
12	1	3	Hershey Bar	•
13	1	4	Snickers	•
14	1	4	Peanut M&Ms	•
15	1	4	Reese's Cups	•
16	1	4	Heath Bars	•

Contents

Overview of the MaxDiff Design Platform 571
Example of a MaxDiff Design 571
MaxDiff Design Launch Window 575
MaxDiff Window 575
Design Options Outline 575
Design Outline 576
Pairwise Incidence Matrix 576
Make Table 576
MaxDiff Options 576

Overview of the MaxDiff Design Platform

A *choice set* is a collection of items from which a respondent must select an item that is most preferred (*best* item) and one that is least preferred (*worst* item). Use the MaxDiff platform to specify the number of items that appear in a choice set and the number of choice sets to be presented to a respondent.

The MaxDiff platform provides you with a single survey. If you have several respondents, you can administer the same survey to all respondents or construct a survey for each respondent.

The MaxDiff design is constructed to match the incidence matrix of a balanced incomplete block design as closely as possible. This implies that each pair of items occurs equally often in the design. If the number of choice sets allows a balanced incomplete block to be constructed for the specified number of items in a choice set, the design that is constructed is a balanced incomplete block design.

The items or profiles are considered to be the levels of the treatment factor. The choice sets are considered to be blocks. In a MaxDiff study, the block size is smaller than the number of treatments.

Example of a MaxDiff Design

You are the purchaser for your company's office supplies and you need to buy candy for a holiday party. First, you want to determine which candy types people prefer. To figure this out, you conduct a MaxDiff study.

You ask five randomly chosen associates to rate seven types of candies. Based on your experience with previous studies, you realize that it is difficult for raters to rank seven types of items in order of preference. Instead, you create a design that consists of choice sets of size four. To keep the study manageable, you structure the survey as a MaxDiff study: you ask each associate to specify his or her most preferred and least preferred candy in each of the seven choice sets. (These selections result in a balanced incomplete block design. See Cochran and Cox (1957).) You administer the same survey to each associate.

Create the Design

Construct a table that lists the items or profiles for your choice sets. In this example, your table of items, Candy Profiles.jmp, is already constructed.

1. Select **Help > Sample Data Library** and open Design Experiment/Candy Profiles.jmp.

   The table lists the seven candy types of interest.
2. Select **DOE** > **Consumer Studies** > **MaxDiff Design**.

3. From the Select Columns list, select Candy and click **X, Factor**.
4. Click **OK**.
5. In the Design Options outline, do the following:
   - Set the **Number of Profiles per Choice Set** to **4**.
   - Set the **Number of Choice Sets** to **7**.

Note: Setting the Random Seed in step 6 reproduces the exact results shown in this example. In constructing a design on your own, this step is not necessary.

6. (Optional) Click the MaxDiff Study red triangle and select **Set Random Seed**. Type 12345 and click **OK**.
7. Click **Make Design**.

   The Design outline shows 7 choice sets, each consisting of 4 candy types. The Pairwise Incidence Matrix provides a matrix of the number of choice sets in which each pair of candy types appear together. The diagonal of the matrix is the number of choice sets each candy type appears in. For this example, each candy appears in 4 choice sets and appears with each other candy types in two of the four choice sets.
8. Click **Make Table**.

Figure 19.2 Partial Design Table for Candy Preference Survey

	Subject	Choice Set	Candy	Choice
1	1	1	Reese's Cups	•
2	1	1	Hershey Bar	•
3	1	1	Snickers	•
4	1	1	Butterfinger	•
5	1	2	Butterfinger	•
6	1	2	Heath Bars	•
7	1	2	Plain M&Ms	•
8	1	2	Snickers	•
9	1	3	Plain M&Ms	•
10	1	3	Snickers	•
11	1	3	Peanut M&Ms	•
12	1	3	Hershey Bar	•
13	1	4	Snickers	•
14	1	4	Peanut M&Ms	•
15	1	4	Reese's Cups	•
16	1	4	Heath Bars	•

The design table contains a Choice column for recording preferences. For each choice set, record a 1 for the most preferred candy, a -1 for the least preferred, and a 0 for the other two candies.

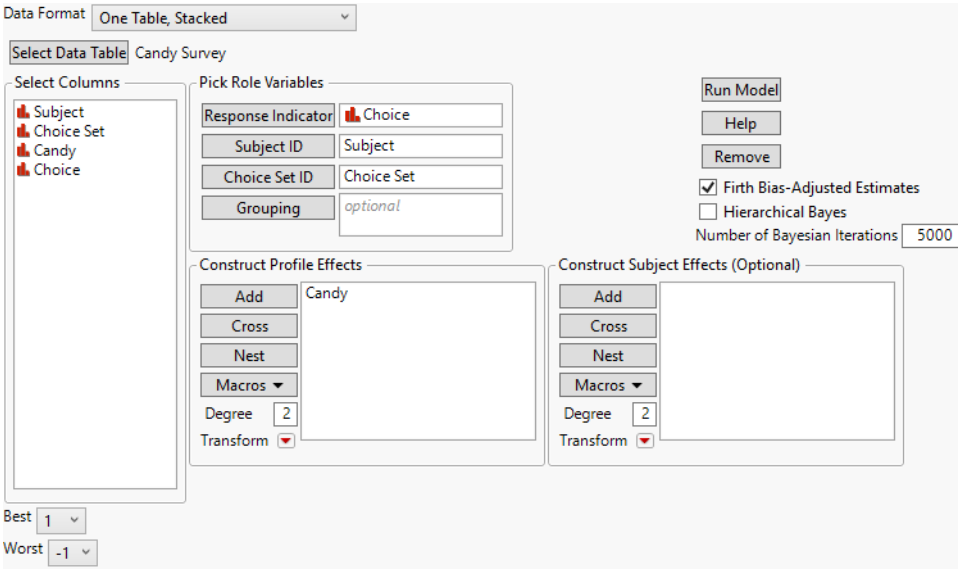
Analyze the Study Results

You conduct the study and record your data in Candy Survey.jmp.

1. Select **Help > Sample Data Library** and open Design Experiment/Candy Survey.jmp.

   The table shows the results of presenting the survey to each of five respondents, listed in the Subject column.
2. Select **Analyze** > **Consumer Research** > **MaxDiff**.
3. Click Select Data Table, select Candy Survey, and click **OK**.
4. Assign column roles:
   - Select Choice and click **Response Indicator**.
   - Select Subject and click **Subject ID**.
   - Select Choice Set and click **Choice Set ID**.
   - Select Candy and click **Add** in the Construct Profile Effects panel.

Figure 19.3 Completed MaxDiff Analysis Launch Window



Because you designated the Best choice as 1 and the Worst choice as -1, you make no change to the Best and Worst choice indicators at the bottom left of the launch window.

5. Click **Run Model**.

Figure 19.4 MaxDiff Report

MaxDiff Results

Marginal Utility	Marginal Probability	Candy
0.9027	0.2664	Plain M&Ms
0.6765	0.2125	Reese's Cups
0.6599	0.2090	Peanut M&Ms
0.2466	0.1382	Heath Bars
0.0302	0.1113	Hershey Bar
-1.061	0.0374	Butterfinger
-1.455	0.0252	Snickers

Parameter Estimates

Likelihood Ratio Tests

Source	L-R ChiSquare	DF	Prob>ChiSq
Candy	39.769	6	<.0001*

The report indicates that Candy is significant. The three candy types with the highest utilities are Plain M&Ms, Reese's Cups, and Peanut M&Ms.

6. Click the MaxDiff Model red triangle and select **All Levels Comparison Report**.

Figure 19.5 All Levels Comparison Report for Candy Types

All Levels Comparison Report

Difference (Row-Column) Standard Error of Difference Wald p-Value	Butterfinger	Heath Bars	Hershey Bar	Peanut M&Ms	Plain M&Ms	Reese's Cups	Snickers
Butterfinger	0	-1.3079	-1.0915	-1.7212	-1.964	-1.7378	0.39343
	0	0.52954	0.52038	0.54692	0.56495	0.54938	0.5059
		0.01964	0.04478	0.0038	0.00162	0.00364	0.44306
Heath Bars	1.30792	0	0.21642	-0.4133	-0.656	-0.4299	1.70135
	0.52954	0	0.47619	0.48741	0.49094	0.46959	0.54758
	0.01964		0.65286	0.40343	0.19185	0.36752	0.0042
Hershey Bar	1.0915	-0.2164	0	-0.6297	-0.8725	-0.6463	1.48493
	0.52038	0.47619	0	0.47406	0.49984	0.49096	0.53808
	0.04478	0.65286		0.19443	0.09149	0.19835	0.00992
Peanut M&Ms	1.72121	0.41329	0.62971	0	-0.2427	-0.0166	2.11463
	0.54692	0.48741	0.47406	0	0.49706	0.49054	0.56901
	0.0038	0.40343	0.19443		0.62896	0.97325	0.00086
Plain M&Ms	1.96396	0.65604	0.87246	0.24275	0	0.22616	2.35738
	0.56495	0.49094	0.49984	0.49706	0	0.49327	0.58562
	0.00162	0.19185	0.09149	0.62896		0.65002	0.00037
Reese's Cups	1.7378	0.42988	0.6463	0.01659	-0.2262	0	2.13123
	0.54938	0.46959	0.49096	0.49054	0.49327	0	0.5714
	0.00364	0.36752	0.19835	0.97325	0.65002		0.00083
Snickers	-0.3934	-1.7013	-1.4849	-2.1146	-2.3574	-2.1312	0
	0.5059	0.54758	0.53808	0.56901	0.58562	0.5714	0
	0.44306	0.0042	0.00992	0.00086	0.00037	0.00083	

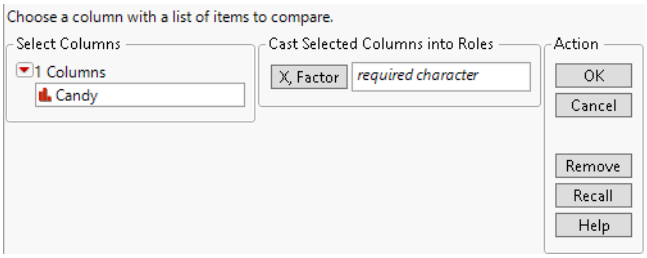
The comparison report indicates which pairs of candy types differ significantly in terms of utility. The third entry in each cell is the *p*-value for the difference defined by the row item's utility minus the column item's utility. The intensity of the color for the *p*-value indicates how significant a difference is. The shading, blue or red, indicates whether the difference (row - column) is negative or positive. The *p*-values are not adjusted to control the multiple comparison error rate and should be used only as a guide. For more information about the All Level Comparisons Report, see *Consumer Research*.

MaxDiff Design Launch Window

To use the MaxDiff Design platform, you need a starting data table. Your starting data table must contain a column of character data that lists the items to be presented to respondents for rating.

With your starting data table active in JMP, select **DOE > Consumer Studies > MaxDiff Design**. If you have no active data tables, you are prompted to navigate to your starting data table.

Figure 19.6 MaxDiff Launch Window using Candy Profiles.jmp

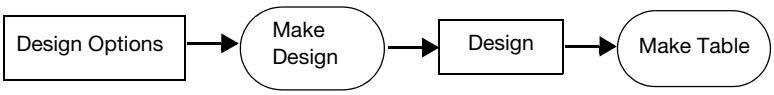


X, Factor The character column that contains the items that respondents will rate.

MaxDiff Window

The MaxDiff Study window updates as you work through the design steps. The outlines, separated by buttons that update the outlines, follow the flow in Figure 19.7

Figure 19.7 MaxDiff Design Flow



The MaxDiff Study window opens showing the Design Options outline. Once you click Make Design, the Design outline appears. To construct the design table, click Make Table.

Design Options Outline

Specify the following:

Number of Profiles per Choice Set The number of items to be included in each choice set.

Number of Choice Sets The total number of sets of items to be presented to and rated by respondents.

Design Outline

The Design outline identifies the choice sets using consecutive positive integers. The items that comprise each choice set are listed.

Pairwise Incidence Matrix

The Pairwise Incidence Matrix outline provides a matrix of the number of choice sets in which each pair of factor levels appear together. The diagonal of the matrix is the number of choice sets each factor level appears in.

Make Table

The Make Table button creates the design table. The design table consists of four columns:

Subject Initially populated by ones. Replace with appropriate identifiers for respondents.

Tip: To easily add respondent identifiers, see *Using JMP*. Use Columns > Recode to change the identifiers to names.

Choice Set A designator for each choice set.

Factor The levels of the factor that you specified. These are the items in the choice set.

Choice A column where you can enter results. Use a numeric value to indicate the best choice, the worst choice, and choices in between. The values 1, -1, and 0 are typically used and are required for analysis by the MaxDiff analysis platform (located at Analyze > Consumer Research > MaxDiff).

MaxDiff Options

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction

– selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table

Simulate Responses Select this option to simulate response values. When you click Make Table, a Simulate Choice window opens along with the design table, and Probability and Choice Simulated columns are added to the design table. The Choice Simulated formula column contains random responses and the Probability formula column contains their probabilities.

Caution: To change the model used in simulating the responses, enter values in the Marginal Utility column in the Simulate Choice window for all factor levels but the last. Because the marginal utilities must sum to zero, you are not permitted to edit the Marginal Utility for the last level. Once you have specified the remaining Marginal Utility values, the last level of the factor is adjusted accordingly.

Number of Starts The number of random starts used in constructing the design. This value is set to 10 by default.

Advanced Options Not applicable for MaxDiff Designs.

Save Script to Script Window Creates the script for the design that you specified and places in an open script window.

Chapter 20

JMP PRO Covering Arrays

Detecting Component Interaction Failures

Covering arrays are used in testing deterministic systems where failures occur as a result of interactions among components or subsystems. The design goal is to reveal if any interaction induces a failure in the system. Application areas include software, circuit, and network design.

Since the tests are deterministic, the emphasis driving the design is the need to cover all required interactions. The Covering Arrays platform constructs highly efficient covering arrays. You can also exclude factor level combinations that are not feasible for your testing protocol.

Figure 20.1 Strength 3 Covering Array

Design

Run	X1	X2	X3	X4	X5
1	L1	L1	L1	L1	L1
2	L1	L1	L1	L2	L2
3	L2	L2	L2	L2	L2
4	L2	L1	L2	L2	L1
5	L2	L2	L2	L1	L2
6	L2	L2	L1	L1	L1
7	L2	L1	L1	L1	L2
8	L1	L2	L2	L1	L1
9	L1	L2	L1	L2	L2
10	L1	L2	L2	L2	L1
11	L1	L1	L2	L1	L2
12	L2	L2	L1	L2	L1

Contents

Overview of Covering Arrays 581
Example of a Covering Array with No Factor Level Restrictions 583
Create the Design 583
Analyze the Experimental Data 585
Example of a Covering Array with Factor Level Restrictions 587
Create the Design 588
Analyze the Experimental Data 592
Covering Array Window 593
Factors 594
Restrict Factor Level Combinations 595
Design 597
Metrics 600
Output Options 600
The Covering Array Data Table 601
Covering Array Options 602
Technical Details for Covering Arrays 603
Algorithm for Optimize 604
Formulas for Metrics 604

JMP PRO Overview of Covering Arrays

You can use covering arrays to test systems where failures occur as a result of interactions among components or subsystems. Covering arrays are often used in areas such as software, circuit, and network design, where the following conditions are likely to be true:

- The cost of testing is usually high.
- Testing focuses on revealing interactions for which failures occur.
- A test run is typically deterministic and results in either success or failure.
- Replicate runs are wasteful because they yield identical results.
- The efficiency of a design is based on how many of the possible conditions are covered without including redundant runs.

Because systems testing is expensive, reducing the amount of testing is critical. Testing all possible interactions is usually prohibitive and often unnecessary. Experience shows that most failures result from the interaction of a small number of components. The size of the largest combination of components likely to drive a failure, called the *strength*, drives the size of the design.

In the Covering Array platform, you specify the required strength of your design. If appropriate, you define factor level combinations that are not permitted. The Covering Array platform constructs a highly efficient design that meets your requirements. It provides metrics that you can use to assess the quality of the design in terms of its coverage. It also provides a script in the data table for the design that enables you to analyze your results.

Covering arrays are often used in situations where certain combinations of factor level settings are not feasible. The Covering Array platform is able to find very efficient covering arrays even when restrictions are placed on factor level combinations.

For background on the structure of covering arrays and algorithms for computing them, see Colbourn (2004), Colbourn et al. (2011), Hartman and Raskin (2004), and Martirosyan (2003). For more information about covering arrays with restrictions on factor levels, see Cohen et al. (2007) and Morgan (2009).

JMP PRO Covering Arrays and Strength

A *covering array of strength t* is a design that tests all combinations of t factor level settings. Consider an interaction defined by specific settings for k factors. If failures occur for all tests involving that interaction, then that interaction *detects* a failure. Using this terminology, a strength t design enables you to detect failures associated with any interaction of up to t factors.

In the literature, covering arrays are also referred to as *factor covering designs*. For background and more information, see Yilmaz et al. (2014), Cohen et al. (2003), and Dalal and Mallows (1998).

To illustrate the nature of covering arrays, consider a situation involving seven categorical factors each with two levels. You want to test all pairwise combinations of factor levels.

A design that might be used in this situation is the 8-run resolution III main effects design:

Figure 20.2 A Resolution III Design with Strength 2

Design

Run	X1	X2	X3	X4	X5	X6	X7
1	L1	L1	L2	L1	L1	L2	L2
2	L2	L2	L2	L2	L1	L1	L2
3	L1	L1	L2	L2	L2	L1	L1
4	L2	L1	L1	L1	L2	L1	L2
5	L1	L2	L1	L1	L1	L1	L1
6	L2	L2	L2	L1	L2	L2	L1
7	L1	L2	L1	L2	L2	L2	L2
8	L2	L1	L1	L2	L1	L2	L1

Note that this factorial design is a strength 2 covering array because all pairwise combinations of levels of any two factors appear. For example, for X1 and X2, the following combinations each appear twice:

- L2 and L1
- L2 and L2
- L1 and L1
- L1 and L2

However, a strength 2 covering array needs only 6 runs :

Figure 20.3 Strength 2 Covering Array

Design

Run	X1	X2	X3	X4	X5	X6	X7
1	2	1	1	2	2	1	2
2	1	2	1	2	1	2	2
3	1	1	2	1	2	2	1
4	2	2	2	2	2	2	1
5	2	2	2	1	1	1	2
6	1	1	1	1	1	1	1

All pairwise combinations of levels of any two factors appear at least once in the six runs. The Covering Array design is more efficient than the Resolution III design because it achieves strength 2 coverage with fewer runs.

The efficiency of a covering array is measured by the number of runs required to achieve the required coverage. The smaller the number of runs, the more efficient the design.

JMP PRO Example of a Covering Array with No Factor Level Restrictions

The data in this example pertain to interoperability in the area of software testing. There are four factors of interest:

- Web Browser (Safari, IE, Firefox, Chrome, Other)
- Operating System (Windows or macOS)
- RAM (4, 8, or 16 MB)
- Connection Speed at three settings (0-1, 1-5, or greater than 5 Mbps)

You are interested in finding out which combinations of these factors are likely to cause failures.

The response is whether the system functions properly for each combination of factor settings.

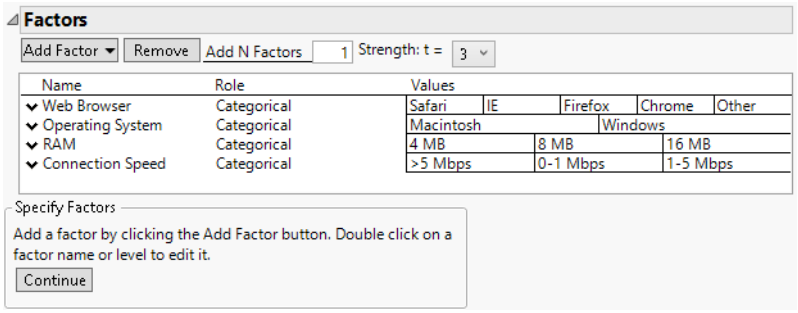
Testing each combination of settings would require 90 (5x2x3x3) trials. To keep the run size manageable, you decide to require Strength 3 coverage, indicating that all combinations of any three factors are tested.

JMP PRO Create the Design

Create the Strength 3 covering array by following these steps.

1. Select **Help > Sample Data Library** and open Design Experiment/Software Factors.jmp.
   The Software Factors.jmp data table contains the factors and their settings.
2. Select **DOE > Special Purpose > Covering Array**.
3. From the menu next to S**trength: t =** , select 3.
4. Click the Covering Array red triangle, select **Load Factors**.
   The Factors outline is populated with the four factors and their levels.

Figure 20.4 Factors Outline for Software Factors



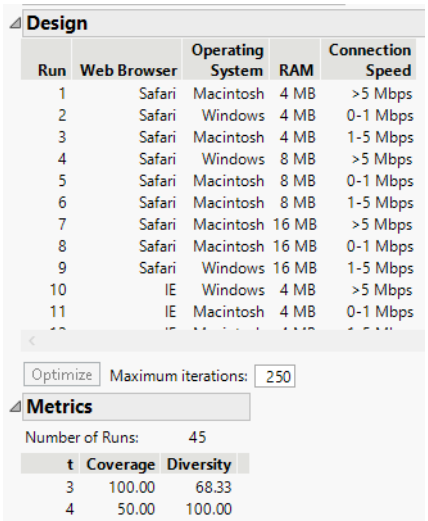
5. Click **Continue**.

   The Restrict Factor Level Combinations outline opens, where you can enter restrictions on the design settings. Because there are no restrictions for this design, do not change the default selection of **None**.

6. Click **Make Design**.

   The Design outline opens to show a 45-run design.

Figure 20.5 Design and Metrics Outlines for Software Factors



In the Metrics outline, consider the row that corresponds to $t = 3$. The Coverage is 100%, indicating that the design covers 100% of the three-factor interactions. This is what you want, because you requested a Strength 3 design. For $t = 3$, the Diversity column indicates that 68.33% of the three-factor interactions that appear are distinct. There is some minor repetition of three-factor combinations.

For $t = 4$, the Coverage is 50%, indicating that the design covers half of the four-factor interactions. There are 90 possible distinct combinations of the four factor settings. The 45 runs in the design comprise one-half of these distinct combinations. The Diversity value of 100% reinforces the fact that none of the four-way interactions are repeated.

7. Click **Make Table**.

Figure 20.6 Partial Design Table for Software Factors

Covering Array
Design Covering Array, Str
DOE Dialog
Analysis

Columns (5/0)
Response
Web Browser *
Operating System *
RAM *
Connection Speed *

Rows
All rows 45
Selected 0
Excluded 0
Hidden 0
Labeled 0

	Response	Web Browser	Operating System	RAM	Connection Speed
1	•	Safari	Macintosh	4 MB	>5 Mbps
2	•	Safari	Windows	4 MB	0-1 Mbps
3	•	Safari	Macintosh	4 MB	1-5 Mbps
4	•	Safari	Windows	8 MB	>5 Mbps
5	•	Safari	Macintosh	8 MB	0-1 Mbps
6	•	Safari	Macintosh	8 MB	1-5 Mbps
7	•	Safari	Macintosh	16 MB	>5 Mbps
8	•	Safari	Macintosh	16 MB	0-1 Mbps
9	•	Safari	Windows	16 MB	1-5 Mbps
10	•	IE	Windows	4 MB	>5 Mbps
11	•	IE	Macintosh	4 MB	0-1 Mbps
12	•	IE	Macintosh	4 MB	1-5 Mbps
13	•	IE	Macintosh	8 MB	>5 Mbps
14	•	IE	Macintosh	8 MB	0-1 Mbps
15	•	IE	Windows	8 MB	1-5 Mbps
16	•	IE	Macintosh	16 MB	>5 Mbps
17	•	IE	Windows	16 MB	0-1 Mbps
18	•	IE	Macintosh	16 MB	1-5 Mbps

The design is presented in a data table. Notice the following in the Table panel at the top left:

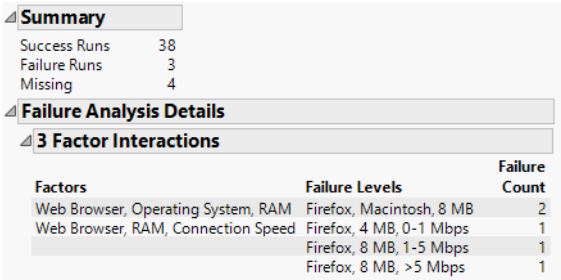
- The Design note indicates that this is a strength 3 covering array.
- The DOE Dialog script reproduces the Covering Array window settings.
- The Analysis script analyzes the experimental data.

JMP PRO Analyze the Experimental Data

Now that you have your design table, you can conduct your experiment and record your data in the Response column of the design table (Figure 20.6). Your experimental results are in the Software Data.jmp sample data table.

1. Select **Help > Sample Data Library** and open Design Experiment/Software Data.jmp.
2. In the Table panel, click the green triangle next to the **Analysis** script.

Figure 20.7 Analysis of Software Experimental Data



Summary

Success Runs	38
Failure Runs	3
Missing	4

Failure Analysis Details

3 Factor Interactions

Factors	Failure Levels	Failure Count
Web Browser, Operating System, RAM	Firefox, Macintosh, 8 MB	2
Web Browser, RAM, Connection Speed	Firefox, 4 MB, 0-1 Mbps	1
	Firefox, 8 MB, 1-5 Mbps	1
	Firefox, 8 MB, >5 Mbps	1

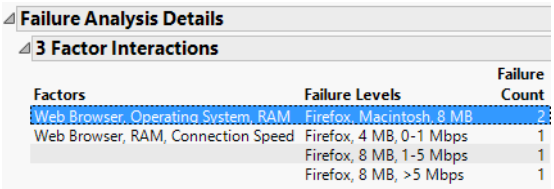
The Summary outline indicates that three tests failed and four tests did not result in a pass or fail outcome.

The Failure Analysis Details outline gives a breakdown of failures in terms of the associated three-way interactions. The outline lists only combinations of factor levels where all tests resulted in failure. If any test that involves a given three-way combination of settings results in success, then that three-way combination of settings cannot be responsible for system failure.

Two failures were associated with Web Browser set to Firefox, Operating System set to macOS, and RAM set to 8 MB. Notice that this combination led to failure regardless of the setting of Connection Speed.

3. Select the first line in the 3 Factor Interactions report.

Figure 20.8 Selection of an Interaction in the Analysis Report



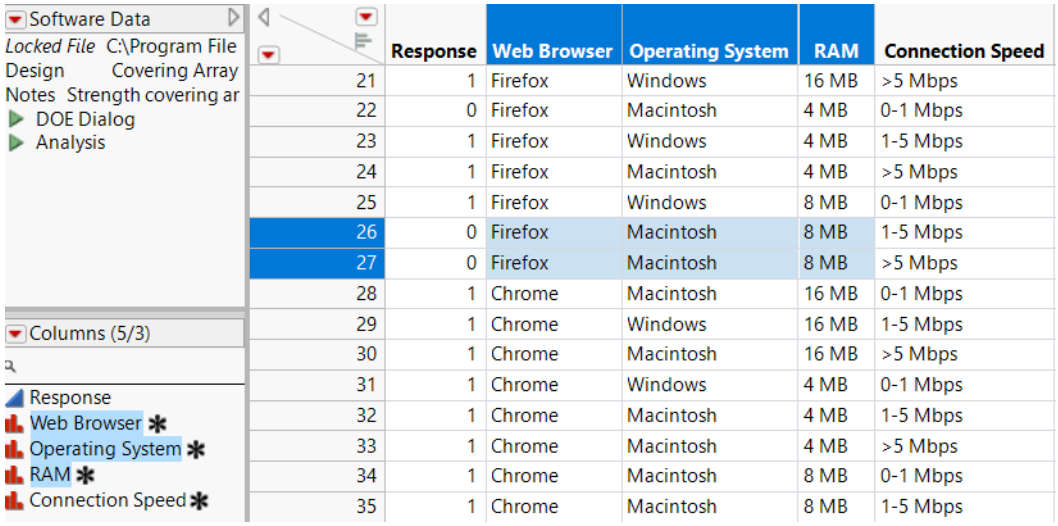
Failure Analysis Details

3 Factor Interactions

Factors	Failure Levels	Failure Count
Web Browser, Operating System, RAM	Firefox, Macintosh, 8 MB	2
Web Browser, RAM, Connection Speed	Firefox, 4 MB, 0-1 Mbps	1
	Firefox, 8 MB, 1-5 Mbps	1
	Firefox, 8 MB, >5 Mbps	1

This action selects the corresponding rows and columns in the data table.

Figure 20.9 Corresponding Selection of Rows and Columns in the Data Table



	Response	Web Browser	Operating System	RAM	Connection Speed
21	1	Firefox	Windows	16 MB	>5 Mbps
22	0	Firefox	Macintosh	4 MB	0-1 Mbps
23	1	Firefox	Windows	4 MB	1-5 Mbps
24	1	Firefox	Macintosh	4 MB	>5 Mbps
25	1	Firefox	Windows	8 MB	0-1 Mbps
26	0	Firefox	Macintosh	8 MB	1-5 Mbps
27	0	Firefox	Macintosh	8 MB	>5 Mbps
28	1	Chrome	Macintosh	16 MB	0-1 Mbps
29	1	Chrome	Windows	16 MB	1-5 Mbps
30	1	Chrome	Macintosh	16 MB	>5 Mbps
31	1	Chrome	Windows	4 MB	0-1 Mbps
32	1	Chrome	Macintosh	4 MB	1-5 Mbps
33	1	Chrome	Macintosh	4 MB	>5 Mbps
34	1	Chrome	Macintosh	8 MB	0-1 Mbps
35	1	Chrome	Macintosh	8 MB	1-5 Mbps

Three failures were associated with combinations of Web Browser, RAM, and Connection Speed. Note that two of these failures, Firefox, 8 MB, 1-5 Mbps and Firefox, 8 MB, >5 Mbps, are among the two failures for the Web Browser, Operating System, and RAM interaction. Selecting any of these rows in the report selects the corresponding rows and columns in the data table.

JMP PRO Example of a Covering Array with Factor Level Restrictions

This example is patterned after an example described in Dalal and Mallows (1998). An originating phone (Near Phone) calls a receiving phone (Far Phone). Each phone call goes through an interface of type A or B. Five factors are of interest:

- Market: USA, UK, Canada, France, Mexico
- Near Phone: ISDN, Bus (Business), Coin, Res (Residential)
- Near Interface: A or B
- Far Phone: ISDN, Bus (Business), Coin, Res (Residential)
- Far Interface: A or B

You are interested in which combinations of pairs of these factors are likely to cause failures. However, certain combinations are not possible:

- An ISDN line on either phone (Near or Far) cannot use interface A.

- Business and Residential lines on the originating phone (Near) cannot use interface B.

JMP PRO Create the Design

The factors and their settings are given in the data table Phone Factors.jmp. Create a Strength 2 covering array by following these steps.

JMP PRO Load Factors

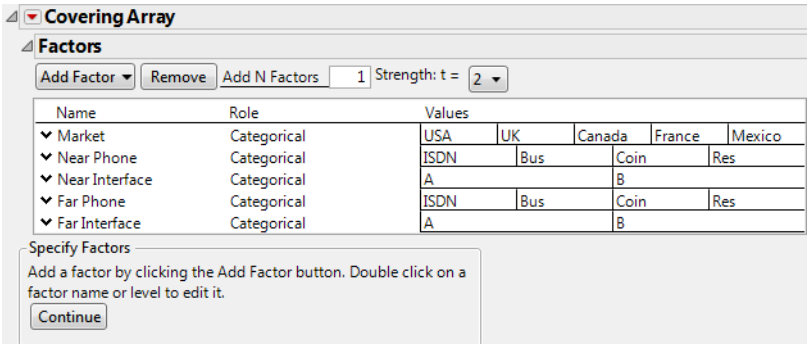
1. Select **Help > Sample Data Library** and open Design Experiment/Phone Factors.jmp.

   The Phone Factors.jmp data table contains the factors and their settings.
2. Select **DOE > Special Purpose > Covering Array**.

   Notice that the menu next to **Strength: t =** , is set to 2 by default.
3. Click the Covering Array red triangle and select **Load Factors**.

   The Factors outline is populated with the five factors and their levels.

Figure 20.10 Factors Outline for Phone Factors



4. Click **Continue**.

   The Restrict Factor Level Combinations outline opens.

JMP PRO Restrict Factor Level Combinations

You can specify disallowed combinations in two ways:

- Use Disallowed Combinations Filter
- Use Disallowed Combinations Script

The filter gives an intuitive way to specify disallowed combinations. The script provides a quick and easy way to specify disallowed combinations, but requires that you have written or saved a script. In this example, if you do not want to specify combinations using the filter, skip to "Specify Disallowed Combinations Using a Script".

Recall that the restrictions are the following:

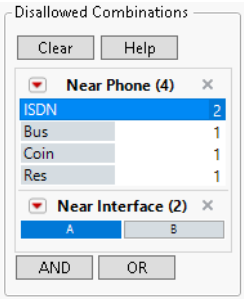
- An ISDN line on either phone (Near or Far) cannot use interface A.
- Business and Residential phones on the originating phone (Near) cannot use interface B.

JMP PRO Specify Disallowed Combinations Using the Filter

Use this approach to enter disallowed combinations using the filter interface. Alternatively, you can paste a script as shown in "Specify Disallowed Combinations Using a Script".

1. Select **Use Disallowed Combinations Filter**.
2. From the Add Filter Factors list, select Near Phone and Near Interface and click the **Add** button.
3. Press Ctrl and click ISDN under Near Phone and A under Near Interface.

Figure 20.11 Disallowed Combinations Panel Showing First Constraints



Both blocks should turn dark. You have added the constraint that an ISDN line on the originating phone (Near) cannot use interface A.

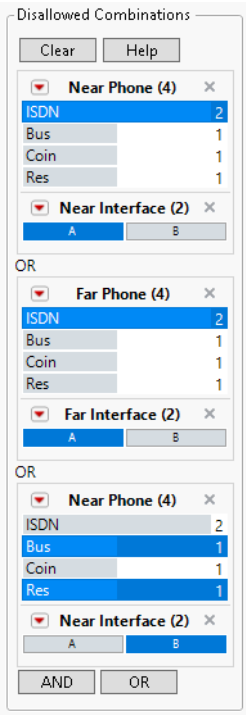
4. Click **OR**.
5. From the Add Filter Factors list, select Far Phone and Far Interface and click the **Add** button.
6. Press Ctrl and click ISDN under Far Phone and A under Far Interface.

   You have added the constraint that an ISDN line on the receiving phone (Far) cannot use interface A.
7. Click **OR**.
8. From the Add Filter Factors list, select Near Phone and Near Interface and click the **Add** button.

9. Press Ctrl and click Bus and Res under Near Phone and B under Near Interface.

   You have added the restriction that Business and Residential lines on the originating phone (Near) cannot use interface B.

Figure 20.12 Completed Disallowed Combinations Filter



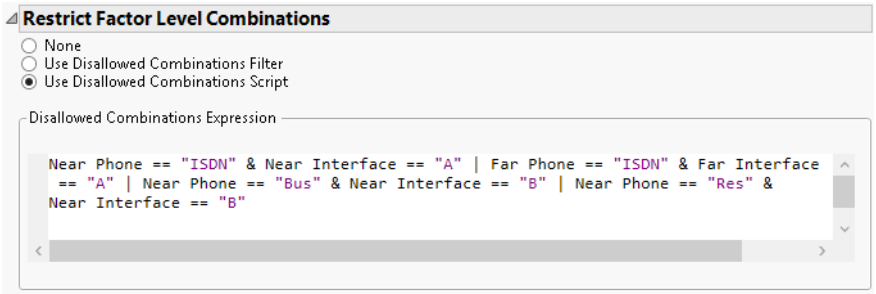
JMP PRO Specify Disallowed Combinations Using a Script

Alternatively, you can specify disallowed combinations by constructing a script. After loading your factors ("Load Factors"), do the following:

1. Click **Continue**.
2. Select **Use Disallowed Combinations Script**.
3. Copy the following script and paste it in the Disallowed Combinations Expression script box:

```
(Near Phone == "ISDN" & Near Interface == "A") |
(Far Phone == "ISDN" & Far Interface == "A")  |
(Near Phone == "Bus" & Near Interface == "B") |
(Near Phone == "Res" & Near Interface == "B")
```

Figure 20.13 Completed Disallowed Combinations Script Window



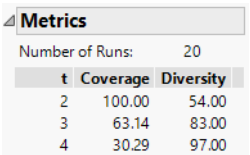
JMP PRO Construct the Design Table

Note: Setting the Random Seed in the next two steps reproduces the exact results shown in this example. When constructing a design on your own, these steps are not necessary.

1. Click the Covering Array red triangle and select **Set Random Seed**.
2. Enter 632 and click **OK**.
3. Click **Make Design**.

   The Design outline opens to show a 20-run design. A Metrics outline is added to the window.

Figure 20.14 Metrics Outline for Phone Design



The Metrics outline indicates that Strength 2 coverage is 100%. This means that all permissible two-factor combinations are represented in the design. The design also covers 65% of all three-factor combinations.

4. Click **Make Table**.

   The design is placed in a design table. A column for the response is provided, as well as various scripts.

Figure 20.15 Covering Array Design Table

	Response	Market	Near Phone	Near Interface	Far Phone	Far Interface
1	•	Mexico	Res	A	Res	A
2	•	Mexico	Bus	A	Bus	A
3	•	France	Coin	B	Bus	B
4	•	UK	Res	A	Coin	A
5	•	UK	ISDN	B	Bus	B
6	•	France	ISDN	B	Res	A
7	•	Canada	Bus	A	Res	A
8	•	Mexico	ISDN	B	ISDN	B
9	•	Canada	ISDN	B	Coin	B
10	•	UK	Bus	A	ISDN	B
11	•	Canada	Res	A	Bus	B
12	•	USA	ISDN	B	ISDN	B
13	•	UK	Coin	B	Res	B
14	•	Mexico	Coin	B	Coin	B
15	•	USA	Bus	A	Bus	A
16	•	USA	Coin	A	Coin	A
17	•	USA	Res	A	Res	A
18	•	France	Res	A	ISDN	B
19	•	France	Bus	A	Coin	B
20	•	Canada	Coin	B	ISDN	B

JMP PRO Analyze the Experimental Data

Now that you have your design table, you can conduct your experiment and record your data in the Response column of the design table. Your experimental results are in the Phone Data.jmp sample data table.

1. Select **Help > Sample Data Library** and open Design Experiment/Phone Data.jmp.
2. In the Table panel, click the green triangle next to the **Analysis** script.

Figure 20.16 Analysis of Phone Experimental Data

Summary

Success Runs	17
Failure Runs	3
Missing	0

Failure Analysis Details

2 Factor Interactions

Factors	Failure Levels	Failure Count
Near Interface, Far Phone	A, Coin	3
Far Phone, Far Interface	Coin, A	2
Market, Near Phone	USA, Coin	1
Market, Near Phone	UK, Res	1
Market, Near Phone	France, Bus	1
Market, Far Phone	USA, Coin	1
Market, Far Phone	UK, Coin	1
Market, Far Phone	France, Coin	1
Market, Far Interface	UK, A	1
Near Phone, Near Interface	Coin, A	1
Near Phone, Far Phone	Bus, Coin	1
Near Phone, Far Phone	Res, Coin	1
Near Phone, Far Interface	Coin, A	1

The Summary outline indicates that three tests failed.

The Failure Analysis Details outline contains a 2 Factor Interactions report, because a two-way interaction is the lowest level interaction that detects a failure.

The 2 Factor Interactions report shows the combinations that might have caused the three failures. It is possible that one combination, Near Interface set to A and Far Phone set to Coin, is responsible for all three failures. Or it is possible that two or three other combinations caused the three failures.

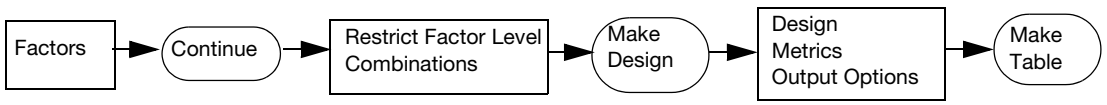
3. Select the first line in the 2 Factor Interactions report.

   In the data table, rows 4, 16, and 19 are selected. Failures occur for these combinations, regardless of the settings for Market, Near Phone, and Far Interface. But note that other combinations of factor settings could account for these failures as well.

JMP PRO Covering Array Window

The covering array window updates as you work through the design steps. For more information about the general flow of DOE windows, see "The DOE Workflow: Describe, Specify, Design". The outlines, separated by buttons that update the outlines, follow the flow in Figure 20.17.

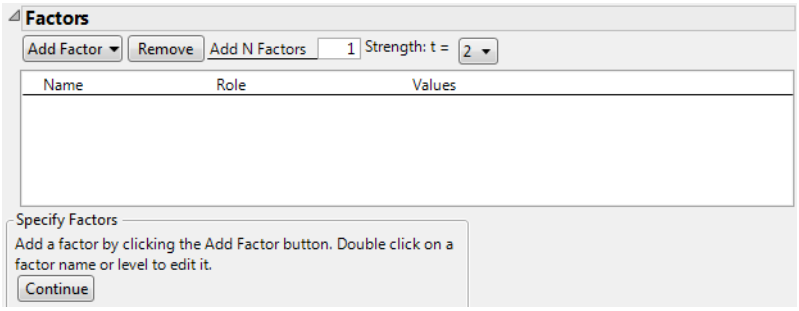
Figure 20.17 Covering Array Flow



JMP PRO Factors

Add factors in the Factors outline.

Figure 20.18 Factors Outline



Add Factor Enters the number of factors specified in **Add N Factors**. All factors are categorical. Select or specify the number of levels.

Remove Removes the selected factors.

Add N Factors Adds multiple factors with a specific number of levels. Enter the number of factors to add, click **Add Factor**, and select or specify the number of levels. Repeat **Add N Factors** to add multiple factors with different numbers of levels.

Strength t = Select a value to specify the strength of the array.

Tip: When you have completed your Factors panel, select **Save Factors** from the red triangle menu. This saves the factor names and values in a data table that you can later reload. See "Covering Array Options".

JMP PRO Factors Table

The Factors table contains the following columns:

Name The name of the factor. When a factor is added, it is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role Specifies the Design Role of the factor. The Design Role for all covering array factors is Categorical. The Design Role column property for the factor is saved to the data table. This property ensures that the factor is modeled appropriately.

Values The settings for the factors. To insert Values, click the default values and enter the desired values. The value ordering in the design table is the order of the values as entered from left to right.

JMP PRO Editing the Factors Table

In the Factors outline, notice the following:

- To edit a factor name, double-click the factor name.
- Categorical factors have a down arrow to the left of the factor name. Click the arrow to add a level.
- To remove a factor level, click the value, click **Delete**, and click outside the text box.
- To edit a value, click the value in the Values column.

JMP PRO Factor Column Properties

For each factor, the Value Labels column property is saved to the design table. The Value Labels column property represents values in a column with specified labels. These labels are shown in the data table and are used in plots and reports. See "Value Labels".

JMP PRO Restrict Factor Level Combinations

When you complete the Factors outline and click Continue, the Restrict Factor Level Combinations outline appears. This outline enables you to specify factor level combinations that are prohibited. Unless you have loaded a constraint or included one as part of a script, the **None** option is selected. To specify constraints, select one of the other options:

Use Disallowed Combinations Filter Defines sets of constraints based on restricting values of individual factors. You can define both AND and OR constraints. See "Use Disallowed Combinations Filter".

Use Disallowed Combinations Script Defines disallowed combinations and other constraints as Boolean JSL expressions in a script editor box. See "Use Disallowed Combinations Script".

JMP PRO Use Disallowed Combinations Filter

This option uses an adaptation of the Data Filter to facilitate specifying disallowed combinations. For detailed information about using the Data Filter, see *Using JMP*.

To add disallowed combinations:

1. Select factors from the Add Filter Factors list and click **Add**.
2. Specify the disallowed combinations by selecting levels.

Note: The red triangle options in the Add Filter Factors menu are the same as those found in the Select Columns panel of many platform launch windows. See *Using JMP*.

When you click Add, the initial panel is updated. The Disallowed Combinations control panel shows the selected factors and provides options for further control.

The Covering Array platform allows only categorical factors. For categorical factors, the possible levels are shown either as labeled blocks or, when the number of levels is large, as list entries. Select a level to disallow it. To select multiple levels, press Ctrl. The block or list entries are highlighted to indicate the levels that have been disallowed. When you add a factor to the Disallowed Combinations panel, the number of levels of the categorical factor is given in parentheses following the factor name.

Disallowed Combinations Options

Clear Clears all disallowed factor level settings that you have specified. This does not clear the selected factors.

AND Opens the Add Filter Factors list. Selected factors become an AND group. Any combination of factor levels specified within an AND group is disallowed.

To add a factor to an AND group later on, click the group's outline to see a highlighted rectangle. Select AND and add the factor.

To remove a single factor, select **Delete** from its red triangle menu.

OR Opens the Add Filter Factors list. Selected factors become a separate AND group. For AND groups separated by OR, a combination is disallowed if it is specified in at least one AND group.

Red Triangle Options for Factors

A factor can appear in several OR groups. An occurrence of the factor in a specific OR group is referred to as an *instance* of the factor.

Delete Removes the selected instance of the factor from the Disallowed Combinations panel.

Clear Selection Clears any selection for that instance of the factor.

Invert Selection Deselects the selected values and selects the values not previously selected for that instance of the factor.

Display Options Changes the appearance of the display.

Blocks Display Shows each level as a block.

List Display Shows each level as a member of a list.

Single Category Display Shows each level.

Check Box Display Adds a check box next to each value.

Radio Box Display Adds a radio selection next to each value for a single selection.

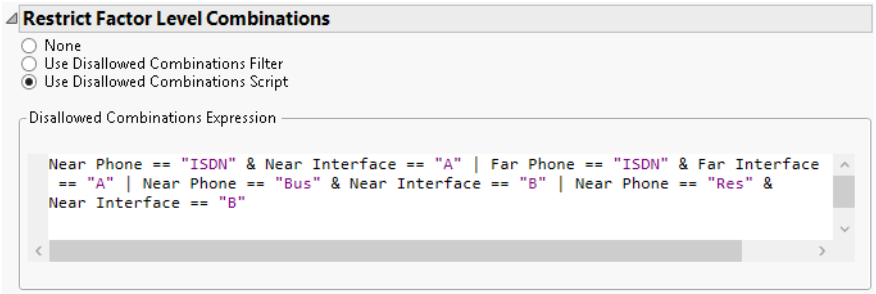
Find (Available for categorical factors.) Provides a text box beneath the factor name where you can enter a search string for levels of the factor. Press Enter or click outside the text box to perform the search. Once **Find** is selected, Find options appear in the search panel black triangle menu. Options include Contain Terms, Contain Phrase, Starts with Phrase or Ends with Phrase, and Invert Result.

JMP PRO Use Disallowed Combinations Script

This option opens a script window where you insert a script that identifies the combinations that you want to disallow. The script must evaluate as a Boolean expression. When the expression evaluates as true, the specified combination is disallowed.

When creating the expression, use the name of the level in quotation marks. Do not use the ordinal value of the level. For example, Figure 20.19 shows the script that you entered in the phone interface example,"Specify Disallowed Combinations Using a Script".

Figure 20.19 Script Window Showing Names of Levels in Quotes



JMP PRO Design

When you click Make Design, the Design and Metrics outlines appear. For designs that require extensive computation, a progress bar appears.

The Design outline shows the design that you have constructed. The first column lists a Run order. You might need to use the scroll bar to view all the runs. The remaining columns show factor settings for each run.

Optimize

Select Optimize to reduce the size of a design that was constructed by the Covering Array platform or that you have loaded using the Load Design red triangle option. Optimize is not available for designs constructed by the Covering Array platform that are known to be optimal. In particular, all unconstrained strength 2 designs for two-level factors constructed by the platform are optimal. Also, any unconstrained strength *t* design for *t+1* factors is optimal for any *t*.

For more information about the algorithm, see "Algorithm for Optimize".

Note: Optimize is time intensive, but can be run repeatedly to yield incrementally better designs.

Use the **Maximum iterations** option to specify a maximum number of iterations to be used in optimizing the design.

JMP PRO Unsatisfiable Constraints

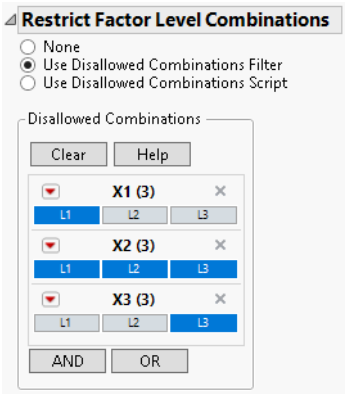
If a set of constraints prohibits the construction of a covering array where all required factor levels are represented, it is said to be *unsatisfiable*.

Example of Unsatisfiable Constraints

Consider a strength 2 design for three factors, each at three levels.

1. Select **DOE > Special Purpose > Covering Array**.
2. Next to **Add N Factors**, type 3.
3. From the **Add Factor** menu, select **3 Level**.
4. Click **Continue**.
5. Select **Use Disallowed Combinations Filter**.
6. From the **Add Filter Factors** list, select all three factors and click the **Add** button.
7. Press Ctrl and select the following levels:
   - For X1, select L1.
   - For X2, select L1, L2, and L3.
   - For X3, select L3.

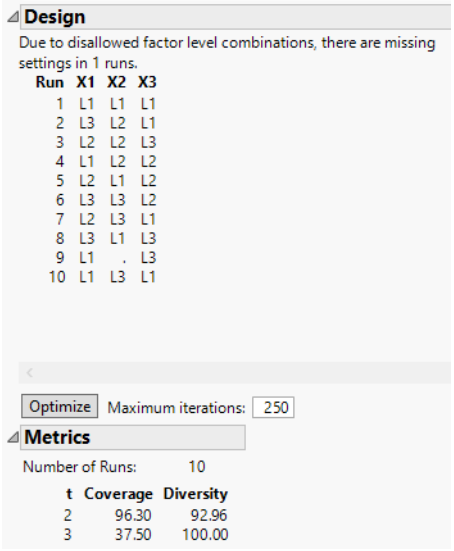
Figure 20.20 Completed Restrict Factor Level Combinations Panel



Note: Setting the Random Seed in the next two steps reproduces the exact results shown in this example. When constructing a design on your own, these steps are not necessary.

8. Click the Covering Array red triangle and select **Set Random Seed**.
9. Enter 12345 and click **OK**.
10. Click **Make Design**.

Figure 20.21 Design and Metrics Outlines



A note beneath the Design outline indicates that one run has a missing setting due to the constraints. That run is run 9. To ensure that the covering array has strength 2, the

combination of X1 set to L1 and X3 set to L3 is required. But for these settings, the constraints prohibit all settings for X2.

JMP PRO Metrics

The Metrics outline gives you information about how well the design meets the strength requirements. See Dalal and Mallows (1998) for background on these metrics for unconstrained designs.

t The number of factors.

Coverage The ratio of the number of distinct *t*-factor settings that appear in the design to the total possible number of *t*-factor settings, expressed as a percentage. A *t*-coverage of 100% indicates that all possible *t*-factor settings are covered by the design. Note that each *t*-factor setting can appear multiple times.

For constrained and unsatisfiable designs, the definition of Coverage is adjusted for the number of *t*-factor settings that are possible once the constraints have been applied to all *t*-factor combinations. See "Formulas for Metrics".

Diversity The ratio of the number of distinct *t*-factor settings in the design to the total number of occurrences of *t*-factor settings in the design, expressed as a percentage. The *t*-diversity measures how well the design avoids replication. A *t*-diversity of 100% indicates that no *t*-factor settings are repeated. A *t*-diversity of 50% indicates that the average number of times that distinct *t*-factor settings appear is two.

For constrained and unsatisfiable designs, the definition of Diversity is adjusted for the number of runs with missing settings. See "Formulas for Metrics".

JMP PRO Output Options

Make Table Constructs the Covering Array data table.

Back Takes you back to the Factors outline. You can make changes to the previous outlines and regenerate the design.

Note: If you have defined Disallowed Combinations in the Restrict Factor Level Combinations outline, these are retained as a script. The script is shown in the Use Disallowed Combinations Script panel when you click Continue.

JMP PRO The Covering Array Data Table

The Covering Array data table contains a first column where you can enter the response. The remaining columns give the factor settings.

The Table panel in the upper left contains a Design note indicating that the design is a Covering Array and giving the Strength of the design. The Table panel also contains the following scripts.

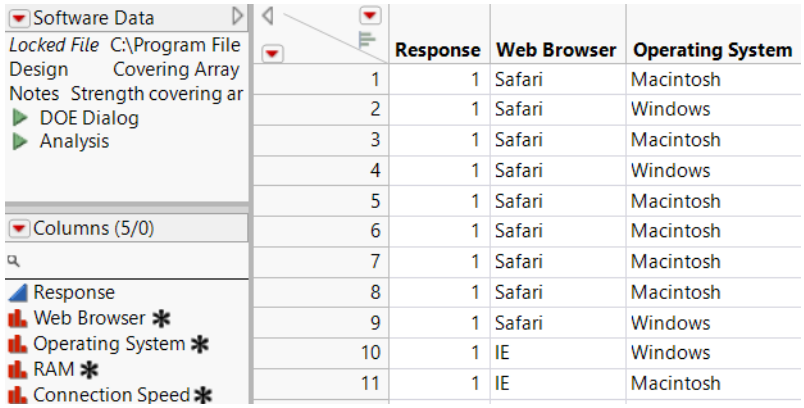
DOE Dialog Re-creates the Covering Array window that you used to generate the design table.

Disallowed Combinations Shows factor level restrictions that you entered in the Restrict Factor Level Combinations outline.

Analysis Provides an analysis of your experimental data. For more information about the script, see "Analysis Script". For background, see Zhang and Zhang (2011).

Tip: To run a script, click the green triangle next to the script name.

Figure 20.22 Partial View of Covering Array Table for Software Data.jmp Showing Scripts



	Response	Web Browser	Operating System
1	1	Safari	Macintosh
2	1	Safari	Windows
3	1	Safari	Macintosh
4	1	Safari	Windows
5	1	Safari	Macintosh
6	1	Safari	Macintosh
7	1	Safari	Macintosh
8	1	Safari	Macintosh
9	1	Safari	Windows
10	1	IE	Windows
11	1	IE	Macintosh

JMP PRO Analysis Script

The Analysis script assumes the following about the Response column:

- The responses are recorded as 0 for failure and 1 for success.
- Missing values are permitted.
- The Response column is continuous.
- You can rename or move the Response column.

The Analysis script produces a report with two outlines:

- The Summary outline gives the number of runs resulting in Success, Failure, and the number of runs for which the response is Missing.
- The Failure Analysis Details report contains a <k> Factor Interactions report. The value of *k* is the smallest number of interactions that detect a failure. (For a definition of *detect*, see "Covering Arrays and Strength".) The three columns contain the following:
  - The **Factors** column lists all *k*-factor combinations that detect failures.
  - The **Failure Levels** column lists the values of the *k* factors in the **Factors** column that detect failures.
  - The **Failure Count** column gives the number of failures corresponding to the *k*-factor combination of **Failure Levels**.

Note: A failure observation can appear in more than one of the *k*-factor combinations listed in the **Failure Levels** column.

The rows in the <k> Factor Interactions report are dynamically linked to the data table. If you select one or more rows in the report, the corresponding rows are selected in the data table.

JMP PRO Covering Array Options

The red triangle menu in the Covering Array platform contains these options:

Save Factors Creates a data table containing a column for each factor that contains its factor levels. Each factor's column contains these column properties: Design Role, Value Order, and Factor Changes. Saving factors enables you to quickly load them into a DOE window.

Note: You can create a factors table for a Covering Array by entering data into an empty table, but remember to assign each column an appropriate Design Role of Categorical. Right-click the column name in the data grid and select **Column Properties > Design Role**. In the Design Role area, select Categorical.

Load Factors Loads factors that you have saved using the Save Factors option into the Factors outline.

Load Design Loads a design from the active data table. If no data table is active, you are prompted to open one. When you select Load Design, a menu appears that enables you to select the columns that you want to specify as factors in the design. All columns are imported as categorical. Columns and their values are listed in the Factors outline. The

Design outline shows a Run for each row in the data table and gives the values of the factors for each run.

The Load Design options enables you to obtain metrics, modify, or construct an Analysis script for an existing design:

- The Metrics outline shows *t*-Coverage and *t*-Diversity for the specified design.
- You can click **Back** to impose factor level restrictions and then construct a new design.
- Clicking **Make Table** constructs a design table where you can enter responses. The table contains an Analysis script for the design.

Set Random Seed Sets the random seed that JMP uses to control certain actions that have a random component. For Covering Arrays, the seed selects a starting design and an iteration count. For most designs, the random seed guarantees reproducibility of the design, but not of the run order.

Note: Upper limits on time as well as iteration count are used to limit design construction time. For some large and high strength designs, depending on the machine, the time limit might override the iteration limit. For such designs, the random seed does not guarantee reproducibility.

To reproduce a design, enter the random seed used to generate it before clicking Make Design.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Advanced Options Not available for Covering Arrays.

Save Script to Script Window Creates the script for the design that you specified in the Covering Array window and places it in an open script window.

JMP PRO Technical Details for Covering Arrays

- "Algorithm for Optimize"
- "Formulas for Metrics"

JMP PRO Algorithm for Optimize

The Optimize button invokes an algorithm that is conceptually similar to a class of covering array optimizers sometimes referred to as *post-construction randomized optimizers* (Nayeri et al., 2013). However, the algorithm in JMP differs from most in that it also addresses designs with constraints. In particular, it optimizes constrained covering arrays as well as unsatisfiable, constrained covering arrays.

The algorithm assumes that the design to be optimized is a covering array of the specified strength. For a K factor design of strength t, the algorithm iteratively examines all ${}_K\mathrm{C}_t$ factor projections to determine whether runs can be eliminated or merged. Consequently, as K or t increases, the run time of the algorithm quickly escalates. To improve performance, the JMP implementation is threaded to use as many CPU cores as are available on your workstation.

JMP PRO Formulas for Metrics

The formulas for Coverage and Diversity depend on whether there are constraints. The following notation is used:

${}_u\mathrm{C}_v$ is the number of combinations of u things taken v at a time

t is the strength of the design

K is the number of factors

$M = {}_K\mathrm{C}_t$

$i = 1, 2, \ldots, M$ is an index that orders all combinations, or *projections*, of t factors

v_{ik} is the number of levels for the k^{th} factor

n_i is the number of distinct t tuples in the design for the i^{th} projection

p_i is the product of the v_{ik} for the factors in the i^{th} projection

r is the number of runs in the design

JMP PRO Unconstrained Design

Coverage and Diversity are given by the following:

$$Coverage = \frac{1}{M}\sum_{i=1}^{M} n_i / p_i$$

$$Diversity = \frac{1}{M}\sum_{i=1}^{M} n_i / r$$

JMP PRO Constrained Design

In a constrained design, certain t tuples are not allowed. This can result in missing values for some t tuples. For some combinations of t factors, there might be no valid t tuples whatsoever. Coverage and diversity must be defined in terms of the possible valid combinations. For this reason, the formulas for constrained designs require additional notation:

a_i is the number of invalid t tuples arising from factors in the i^{th} projection

m is the number of projections where there are no valid t tuples

q_i is the number of runs in the design with missing values for any factor in the i^{th} projection

$r_i = r - q_i$

$M' = M - m$

Coverage and Diversity are given by the following:

$$Coverage = \frac{1}{M'}\sum_{i=1}^{M'} n_i / (p_i - a_i)$$

$$Diversity = \frac{1}{M'}\sum_{i=1}^{M'} n_i / r_i$$

If there are no invalid t tuples (M′ = M) and if there are no missing values ($r_i = r$, for all i), then the definitions for coverage and diversity for constrained designs reduce to the definitions for unconstrained designs. See Morgan (2014).

Chapter 21

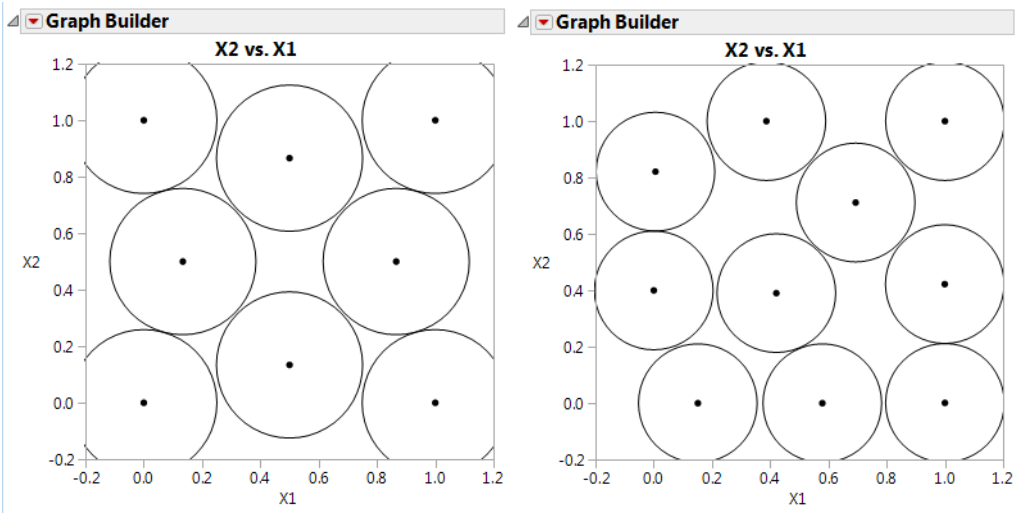
Space-Filling Designs

Space-filling designs are useful in situations where run-to-run variability is of far less concern than the form of the model. Consider a sensitivity study of a computer simulation model. In this situation, and for any mechanistic or deterministic modeling problem, any variability is small enough to be ignored. For systems with no variability, replication, randomization, and blocking are irrelevant.

The Space Filling platform provides designs for situations with both continuous and categorical factors. For continuous factors, space-filling designs have two objectives:

- maximize the distance between any two design points
- space the points uniformly

Figure 21.1 Space-Filling Design



Contents

Overview of Space-Filling Designs . . . 609
Build a Space Filling Design . . . 610
Responses . . . 610
Factors . . . 612
Define Factor Constraints . . . 614
Space Filling Design Methods . . . 618
Design . . . 618
Design Diagnostics . . . 618
Design Table . . . 619
Space Filling Design Options . . . 619
Sphere-Packing Designs . . . 621
Creating a Sphere-Packing Design . . . 621
Visualizing the Sphere-Packing Design . . . 623
Latin Hypercube Designs . . . 624
Creating a Latin Hypercube Design . . . 624
Visualizing the Latin Hypercube Design . . . 626
Uniform Designs . . . 628
Comparing Sphere-Packing, Latin Hypercube, and Uniform Methods . . . 630
Minimum Potential Designs . . . 632
Maximum Entropy Designs . . . 634
Gaussian Process IMSE Optimal Designs . . . 636
Fast Flexible Filling Designs . . . 637
FFF Optimality Criterion . . . 637
Set Average Cluster Size for FFF Designs . . . 639
Constraints . . . 639
Creating and Viewing a Constrained Fast Flexible Filling Design . . . 640
Creating a Space-Filling Design for a Map Shape . . . 641
Example of a Sphere-Packing Design . . . 646
Create the Sphere-Packing Design for the Borehole Data . . . 646
Guidelines for the Analysis of Deterministic Data . . . 647

Overview of Space-Filling Designs

Space-filling designs are useful for modeling systems that are deterministic or near-deterministic. One example of a deterministic system is a computer simulation. Such simulations can be very complex involving many variables with complicated interrelationships. A goal of designed experiments on these systems is to find a simpler empirical model that adequately predicts the behavior of the system over limited ranges of the factors.

In experiments on systems where there is substantial random noise, the goal is to minimize the variance of prediction. In experiments on deterministic systems, there is no variance but there is *bias*. Bias is the difference between the approximation model and the true mathematical function. The goal of space-filling designs is to bound the bias.

One approach to bound the bias is to spread the design points out as far from each other as possible while staying inside the experimental boundaries. The other approach is to space the points out evenly over the region of interest.

The **Space Filling** designer supports the following design methods:

Sphere Packing Maximizes the minimum distance between pairs of design points. See "Sphere-Packing Designs" and "Create the Sphere-Packing Design for the Borehole Data".

Latin Hypercube Maximizes the minimum distance between design points but requires even spacing of the levels of each factor. This method produces designs that mimic the uniform distribution. The Latin Hypercube method is a compromise between the Sphere-Packing method and the Uniform design method. See "Latin Hypercube Designs".

Uniform Minimizes the discrepancy between the design points (which have an empirical uniform distribution) and a theoretical uniform distribution. See "Uniform Designs".

Minimum Potential Spreads points out inside a sphere around the center. See "Minimum Potential Designs".

Maximum Entropy Measures the amount of information contained in the distribution of a set of data. See "Maximum Entropy Designs".

Gaussian Process IMSE Optimal Creates a design that minimizes the integrated mean squared error of the Gaussian process over the experimental region. See "Gaussian Process IMSE Optimal Designs".

Fast Flexible Filling The Fast Flexible Filling method forms clusters from random points in the design space. These clusters are used to choose design points according to an optimization criterion. This is the only method that can accommodate categorical factors and constraints on the design space. You can specify linear constraints and disallowed

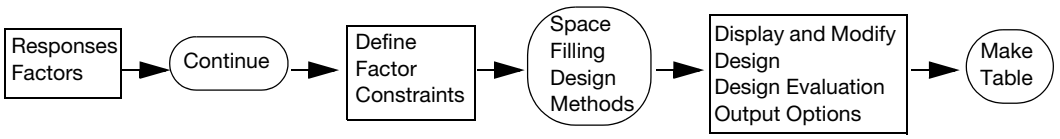
combinations. See "Fast Flexible Filling Designs" and "Creating and Viewing a Constrained Fast Flexible Filling Design".

Note: If the number of runs is 500 or less, a Gaussian Process model is saved to the data table. If the number of runs exceeds 500, a Neural model is saved to the data table.

Build a Space Filling Design

Build a Space Filling Design by selecting **DOE > Special Purpose > Space Filling Design**. First define your responses and factors. Next, continue to the design options, generation, and evaluation. The design process follows the flow in Figure 21.2.

Figure 21.2 Space Filling Design Flow

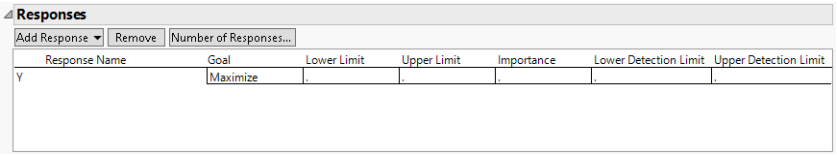


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 21.3 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.
- A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for

the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

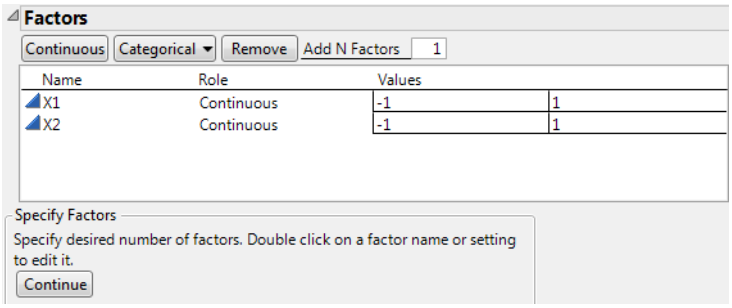
The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

Factors

Add factors for a space filling design in the Factors outline.

Figure 21.4 Factors Outline



The Factors outline contains these options:

Continuous Enters the number of continuous factors specified in **Add N Factors**.

Categorical Enters the number of nominal factors specified in **Add N Factors.**

Remove Removes the selected factors.

Add N Factors Adds multiple factors of a given type. Enter the number of factors to add and click Continuous or Categorical. Repeat **Add N Factors** to add multiple factors of different types.

Tip: When you have completed your Factors panel, select **Save Factors** from the red triangle menu. This saves the factor names and values in a data table that you can later reload. See "Space Filling Design Options".

Factors Outline

The Factors outline contains the following columns:

Name The name of the factor. When added, a factor is given a default name of X1, X2, and so on. To change this name, double-click it and enter the desired name.

Role Specifies the Design Role of the factor. The Design Role column property for the factor is saved to the data table. This property ensures that the factor type is modeled appropriately.

Values The experimental settings for the factors. To insert Values, click the default values and enter the desired values.

Editing the Factors Outline

In the Factors outline, note the following:

- To edit a factor name, double-click the factor name.

- Categorical factors have a down arrow to the left of the factor name. Click the arrow to add a level.
- To remove a factor level, click the value, click **Delete**, and click outside the text box.
- To edit a value, click the value in the Values column.

Factor Types

Continuous Numeric data types only. A continuous factor is a factor that you can conceptually set to any value between the lower and upper limits you supply, given the limitations of your process and measurement system.

Categorical Either numeric or character data types. For a categorical factor, the value ordering is the order of the values as entered from left to right. This ordering is saved in a Value Order column property after the design data table is created.

Factor Column Properties

For each factor, various column properties are saved to the data table.

Design Role Each factor is assigned the Design Role column property. The Role that you specify in defining the factor determines the value of its Design Role column property. The Design Role property reflects how the factor is intended to be used in modeling the experimental data. Design Role values are used in the Augment Design platform.

Factor Changes Each factor is assigned the Factor Changes column property with a setting of Easy. In space-filling designs, it is assumed that factor levels can be changed for each experimental run. Factor Changes values are used in the Evaluate Design and Augment Design platforms.

Coding If the Role is Continuous, the Coding column property for the factor is saved. This property transforms the factor values so that the low and high values correspond to –1 and +1, respectively. The estimates and tests in the Fit Least Squares report are based on the transformed values.

Value Order If the Role is Categorical or Blocking, the Value Order column property for the factor is saved. This property determines the order in which levels of the factor appear.

Define Factor Constraints

Factor constraints can be specified only for space filling designs constructed using the Fast Flexible Filling method.

Use Define Factor Constraints to restrict the design space. Unless you have loaded a constraint or included one as part of a script, the **None** option is selected. To specify constraints, select one of the other options:

Specify Linear Constraints Specifies inequality constraints on linear combinations of factors. Available only for factors with a Role of Continuous or Mixture. See "Specify Linear Constraints".

Note: When you save a script for a design that involves a linear constraint, the script expresses the linear constraint as a *less than or equal to* inequality (≤).

Use Disallowed Combinations Filter Defines sets of constraints based on restricting values of individual factors. You can define both AND and OR constraints. See "Use Disallowed Combinations Filter".

Use Disallowed Combinations Script Defines disallowed combinations and other constraints as Boolean JSL expressions in a script editor box. See "Use Disallowed Combinations Script".

Note: When you analyze a design that has factor constraints, the model profiler honors the constraints.

Specify Linear Constraints

In cases where it is impossible to vary continuous factors independently over the design space, you can specify linear inequality constraints. Linear inequalities describe factor level settings that are allowed. For an example, see "Mixture of Mixtures Design".

Click **Add** to enter one or more linear inequality constraints.

Add Adds a template for a linear expression involving all the continuous factors in your design. Enter coefficient values for the factors and select the direction of the inequality to reflect your linear constraint. Specify the constraining value in the box to the right of the inequality. To add more constraints, click **Add** again.

Note: The Add option is disabled if you have already constrained the design region by specifying a Sphere Radius.

Remove Last Constraint Removes the last constraint.

Check Constraints Checks the constraints for consistency. This option removes redundant constraints and conducts feasibility checks. A JMP alert appears if there is a problem. If constraints are equivalent to bounds on the factors, a JMP alert indicates that the bounds in the Factors outline have been updated.

Use Disallowed Combinations Filter

This option uses an adaptation of the Data Filter to facilitate specifying disallowed combinations. For detailed information, see *Using JMP*.

Select factors from the Add Filter Factors list and click **Add**. Then specify the disallowed combinations by using the slider (for continuous factors) or by selecting levels (for categorical factors). For an example, see "Response Surface Design With Constraints and Categorical Factor".

The red triangle options for the Add Filter Factors menu are those found in the Select Columns panel of many platform launch windows. See *Using JMP*.

When you click Add, the Disallowed Combinations control panel shows the selected factors and provides options for further control. Factors are represented as follows, based on their modeling types:

Continuous Factors For a continuous factor, a double-ended slider that spans the range of factor settings appears. You can specify disallowed settings by dragging the slider ends or by setting the endpoints by clicking on the text value under the slider. In the slider, a solid blue highlight represents the disallowed values.

Categorical Factor For a categorical factor, the possible levels are displayed either as labeled blocks or, when the number of levels is large, as list entries. Select a level to disallow it. To select multiple levels, hold the Control key. The block or list entries are highlighted to indicate the levels that have been disallowed. When you add a categorical factor to the Disallowed Combinations panel, the number of levels of the categorical factor is given in parentheses following the factor name.

Disallowed Combinations Options

The control panel has the following controls:

Clear Clears all disallowed factor level settings that you have specified. This does not clear the selected factors.

Start Over Removes all selected factors and returns you to the initial list of factors.

AND Opens the Add Filter Factors list. Selected factors become an AND group. Any combination of factor levels specified within an AND group is disallowed.

To add a factor to an AND group later on, click the group's outline to see a highlighted rectangle. Select AND and add the factor.

To remove a single factor, select **Delete** from its red triangle menu.

OR Opens the Add Filter Factors list. Selected factors become a separate AND group. For AND groups separated by OR, a combination is disallowed if it is specified in at least one AND group.

Red Triangle Options for Factors

A factor can appear in several OR groups. An occurrence of the factor in a specific OR group is referred to as an *instance* of the factor.

Delete Removes the selected instance of the factor from the Disallowed Combinations panel.

Clear Selection Clears any selection for that instance of the factor.

Invert Selection Deselects the selected values and selects the values not previously selected for that instance of the factor.

Display Options Available only for categorical factors. Changes the appearance of the display. Options include showing each level as a block, list, or single category, or adding a check box next to each value.

Find Available only for categorical factors. Provides a text box beneath the factor name where you can enter a search string for levels of the factor. Press the Enter key or click outside the text box to perform the search. Once **Find** is selected, Find options appear in the red triangle menu, such as Does not contain, Match Case, Starts with or Ends with, or Clear Find.

Use Disallowed Combinations Script

Use this option to disallow particular combinations of factor levels using a JSL script. This option can be used with continuous factors or mixed continuous and categorical factors.

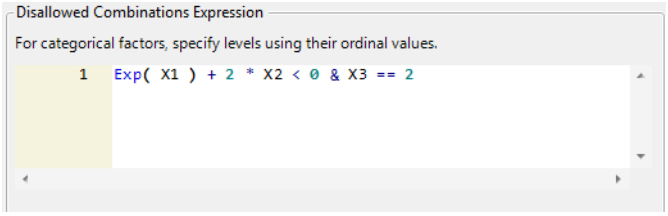
This option opens a script window where you insert a script that identifies the combinations that you want to disallow. The script must evaluate as a Boolean expression. When the expression evaluates as true, the specified combination is disallowed.

When forming the expression for a categorical factor, use the ordinal value of the level or the name of the level. If a factor's levels are high, medium, and low, specified in that order in the Factors outline, their associated ordinal values are 1, 2, and 3. For example, suppose that you have two continuous factors, X1 and X2, and a categorical factor X3 with three levels: L1, L2, and L3, in order. You want to disallow levels where the following is true:

$$e^{X_1} + 2X_2 < 0 \text{ and } X_3 = L2$$

Enter the expression `(Exp(X1) + 2*X2 < 0) & (X3 == 2)` into the script window.

Figure 21.5 Expression in Script Editor



(In the figure, unnecessary parentheses were removed by parsing.) Notice that functions can be entered as part of the Boolean expression. The expression
`(Exp(X1) + 2*X2 < 0) & (X3 == "L2")` is also valid.

Space Filling Design Methods

The following methods for constructing space-filling designs are available:

- "Sphere-Packing Designs"
- "Latin Hypercube Designs"
- "Uniform Designs"
- "Minimum Potential Designs"
- "Maximum Entropy Designs"
- "Gaussian Process IMSE Optimal Designs"
- "Fast Flexible Filling Designs"

Design

The Design outline shows the runs for the space-filling design.

Design Diagnostics

The Design Diagnostics outline shows the values for the space filling design factors scaled from zero to one. The Minimum Distance is based on these scaled values and is the minimum distance from each point to its nearest neighbor. The row number for the nearest neighbor is given in the Nearest Point column. The discrepancy value shown below the table is the integrated difference between the design points based and a uniform distribution.

The MaxPro (maximum projection) criterion is provided for the full design as well as for each level of categorical variables when used. Smaller values are better. For more information about the MaxPro criteria see "MaxPro".

Note: If for any dimension, two points can have the same value (that is, $x_{ik} = x_{jk}$), the MaxPro criterion is undefined. In this case, the Design Diagnostics shows a missing value for MaxPro.

Design Table

Use the Design Table buttons to finish your Space Filling Design construction.

Make Table Constructs the Space Filling Design data table.

Back Takes you back to where you were before clicking **Make Design**. You can make changes to the previous outlines and regenerate the design.

Space Filling Design Options

The red triangle menu in the Space Filling Design platform contains these options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

FFF Optimality Criterion For the Fast Flexible Filling design method, enables you to select between the MaxPro criterion (the default) and the Centroid criterion. See "FFF Optimality Criterion".

Number of Starts Specifies the number of times that the algorithm for the chosen design type initiates to construct a new design. The best design, based on the criterion for the given design type, is returned. Set to 1 by default for all design types. Not used for Fast Flexible Filling Designs.

Advanced Options > Set Average Cluster Size For the Fast Flexible Filling design method, enables you to specify the average number of randomly generated points used to define each cluster or, equivalently, each design point.

Advanced Options > MaxPro Categorical Weight For the Fast Flexible Filling design method, enables you to specify the weight used for categorical factors in the MaxPro calculation. Points that have the same categorical level are multiplied by the weight. This provides better space-filling properties at the sub-design for each level of a categorical factor.

Save Script to Script Window Creates the script for the design that you specified in the Space Filling Design platform and saves it in an open script window.

Sphere-Packing Designs

The Sphere-Packing design method maximizes the minimum distance between pairs of design points. The effect of this maximization is to spread the points out as much as possible inside the design region.

- "Creating a Sphere-Packing Design"
- "Visualizing the Sphere-Packing Design"

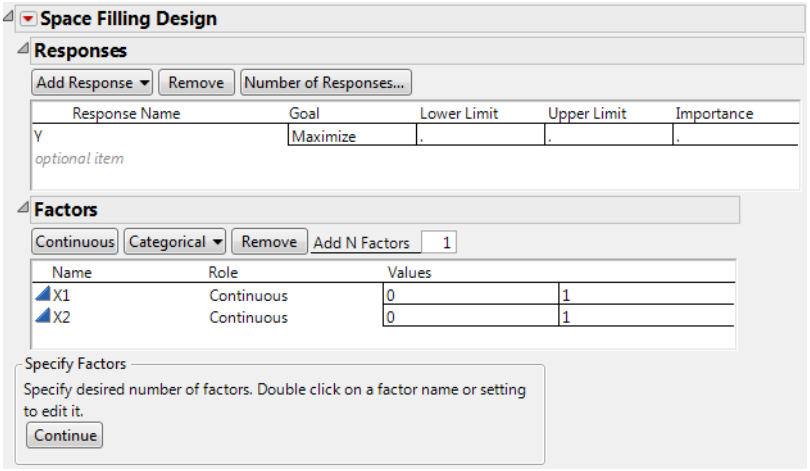
Creating a Sphere-Packing Design

To create a sphere-packing design follow these steps:

1. Select **DOE > Special Purpose > Space Filling Design**.
2. Enter responses and factors.

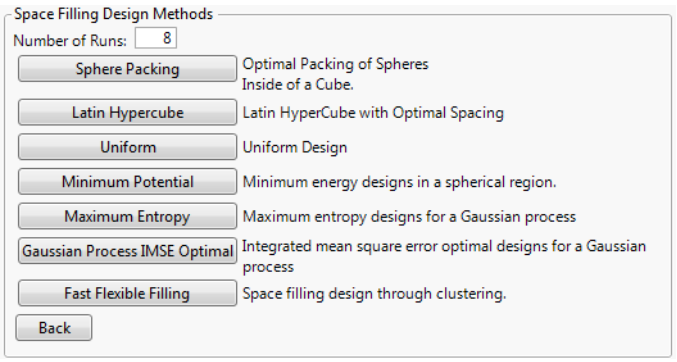
   See "Responses".
3. Alter the factor level values, if necessary. For example, Figure 21.6 shows the two existing factors, X1 and X2, with values that range from 0 to 1 (instead of the default –1 to 1).

Figure 21.6 Space-Filling Dialog for Two Factors



4. Click **Continue**.
5. In the design specification dialog, specify a sample size (**Number of Runs**). Figure 21.7 shows a sample size of eight.

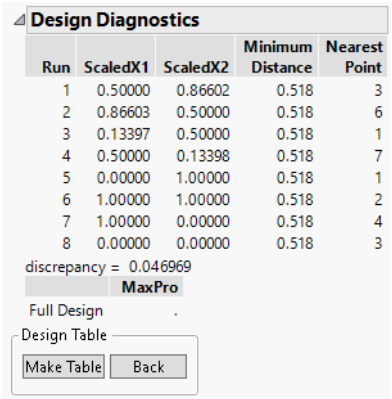
Figure 21.7 Space-Filling Design Dialog



6. Click **Sphere Packing.**

   JMP creates the design and displays the design runs and the design diagnostics. Figure 21.8 shows the Design Diagnostics panel with 0.518 as the **Minimum Distance**. Your results might differ slightly from the ones below, but the minimum distance is the same. The MaxPro statistic is undefined for Sphere Packing as points can have identical values for one dimension.

Figure 21.8 Sphere-Packing Design Diagnostics



Design Diagnostics

Run	ScaledX1	ScaledX2	Minimum Distance	Nearest Point
1	0.50000	0.86602	0.518	3
2	0.86603	0.50000	0.518	6
3	0.13397	0.50000	0.518	1
4	0.50000	0.13398	0.518	7
5	0.00000	1.00000	0.518	1
6	1.00000	1.00000	0.518	2
7	1.00000	0.00000	0.518	4
8	0.00000	0.00000	0.518	3

discrepancy = 0.046969

	MaxPro
Full Design	.

Design Table

Make Table Back

7. Click **Make Table**. Use this table to complete the visualization example, described next.

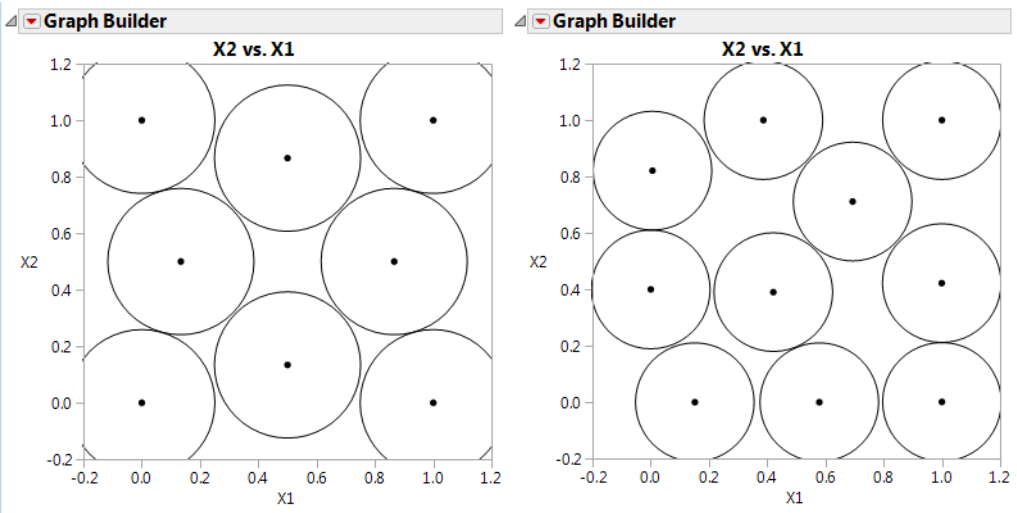
Visualizing the Sphere-Packing Design

To visualize the sphere-packing design use the sphere-packing design table to create a plot with Graph Builder:

1. Select **Graph > Graph Builder**.
2. Specify X1 as **X** and X2 as **Y**.
3. **Click** the smoother icon to remove the smoother (blue line) from your graph.
4. Adjust the frame size so that the frame is square by right-clicking the plot and selecting **Graph > Size/Scale > Size to Isometric**.
5. Right-click the plot and select **Customize**. When the Customize panel appears, click the plus sign to see a text edit area and enter the following script:
   `For Each Row(Circle({:X1, :X2}, 0.518/2))`
   where `0.518` is the minimum distance number that you noted in the Design Diagnostics panel. This script draws a circle centered at each design point with radius 0.259 (half the diameter, 0.518), as shown on the left in Figure 21.9. This plot shows the efficient way JMP packs the design points.
6. Now repeat the procedure exactly as described in the previous section, but with a sample size of 10 instead of eight.

   Remember to change `0.518` in the graphics script to the minimum distance produced by 10 runs. When the plot appears, again set the frame size and create a graphics script using the minimum distance from the diagnostic report as the diameter for the circle. You should see a graph similar to the one on the right in Figure 21.9. Note the irregular nature of the sphere packing. In fact, you can repeat the process a third time to get a slightly different picture because the arrangement is dependent on the random starting point.

Figure 21.9 Sphere-Packing Example with Eight Runs (left) and 10 Runs (right)



Latin Hypercube Designs

In a Latin Hypercube, each factor has as many levels as there are runs in the design. The levels are spaced evenly from the lower bound to the upper bound of the factor. Like the sphere-packing method, the Latin Hypercube method chooses points to maximize the minimum distance between design points, but with a constraint. The constraint maintains the even spacing between factor levels.

- "Creating a Latin Hypercube Design"
- "Visualizing the Latin Hypercube Design"

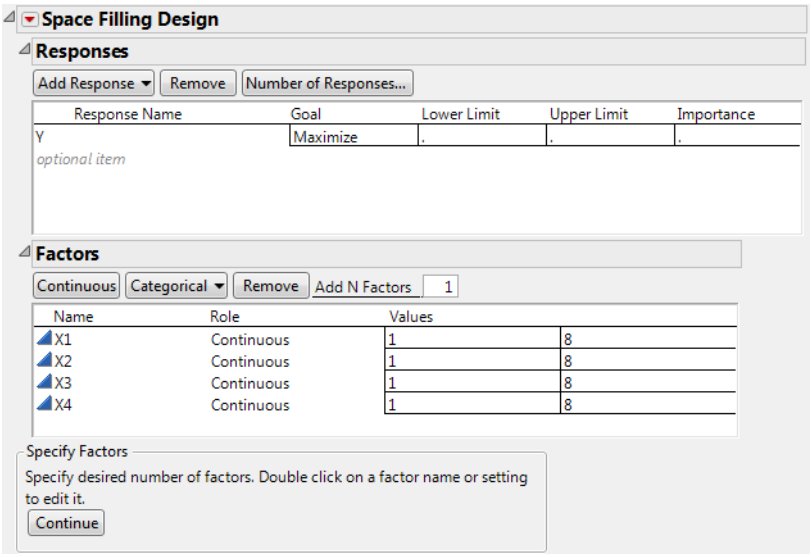
Creating a Latin Hypercube Design

To create a Latin hypercube design follow these steps:

1. Select **DOE > Special Purpose > Space Filling Design**.
2. Enter responses, if necessary, and factors.

   See "Responses".
3. Alter the factor level values, if necessary. Figure 21.10 shows adding four factors and changing their values to 1 and 8 instead of the default –1 and 1.

Figure 21.10 Space-Filling Dialog for Four Factors



4. Click **Continue.**
5. In the design specification dialog, specify a sample size (**Number of Runs**). This example uses a sample size of eight.
6. Click **Latin Hypercube** (Figure 21.7). Factor settings and design diagnostics results appear similar to those in Figure 21.11, which shows the Latin Hypercube design with four factors and eight runs.

Note: The purpose of this example is to show that each column (factor) is assigned each level only once, and each column is a different permutation of the levels.

Figure 21.11 Latin Hypercube Design for Four Factors and Eight Runs with Eight Levels

4 Factors
Space Filling Latin Hypercube

Design

Run	X1	X2	X3	X4
1	4.00000	7.00000	1.00000	4.00000
2	2.00000	6.00000	5.00000	7.00000
3	1.00000	4.00000	6.00000	1.00000
4	7.00000	5.00000	3.00000	8.00000
5	5.00000	2.00000	8.00000	6.00000
6	6.00000	8.00000	7.00000	3.00000
7	8.00000	3.00000	4.00000	2.00000
8	3.00000	1.00000	2.00000	5.00000

Design Diagnostics

Run	ScaledX1	ScaledX2	ScaledX3	ScaledX4	Minimum Distance	Nearest Point
1	0.42857	0.85714	0.00000	0.42857	0.782	2
2	0.14286	0.71429	0.57143	0.85714	0.782	1
3	0.00000	0.42857	0.71429	0.00000	0.926	2
4	0.85714	0.57143	0.28571	1.00000	0.795	2
5	0.57143	0.14286	1.00000	0.71429	0.845	2
6	0.71429	1.00000	0.85714	0.28571	0.892	7
7	1.00000	0.28571	0.42857	0.14286	0.892	6
8	0.28571	0.00000	0.14286	0.57143	0.892	1

discrepancy = 0.039317

	MaxPro
Full Design	11.40992

Visualizing the Latin Hypercube Design

To visualize the nature of the Latin Hypercube technique, create a plot with Graph Builder:

1. Create another Latin Hypercube design using two factors.
2. Select **DOE > Special Purpose > Space Filling Design**.
3. Enter two factors and be sure to change the factor values so that they are 0 and 1 instead of the default –1 and 1.
4. Click **Continue**.
5. Specify a sample size of eight (**Number of Runs**).
6. Click **Latin Hypercube**.

Figure 21.12 Latin Hypercube Design with Two Factors and Eight Runs

2 Factors
Space Filling Latin Hypercube

Design

Run	X1	X2
1	0.28571	0.00000
2	0.00000	0.85714
3	0.57143	0.28571
4	0.42857	0.71429
5	0.14286	0.42857
6	1.00000	0.14286
7	0.85714	0.57143
8	0.71429	1.00000

Design Diagnostics

Run	ScaledX1	ScaledX2	Minimum Distance	Nearest Point
1	0.28571	0.00000	0.404	3
2	0.00000	0.85714	0.452	4
3	0.57143	0.28571	0.404	1
4	0.42857	0.71429	0.404	5
5	0.14286	0.42857	0.404	4
6	1.00000	0.14286	0.452	7
7	0.85714	0.57143	0.404	3
8	0.71429	1.00000	0.404	4

discrepancy = 0.009279

	MaxPro
Full Design	11.32801

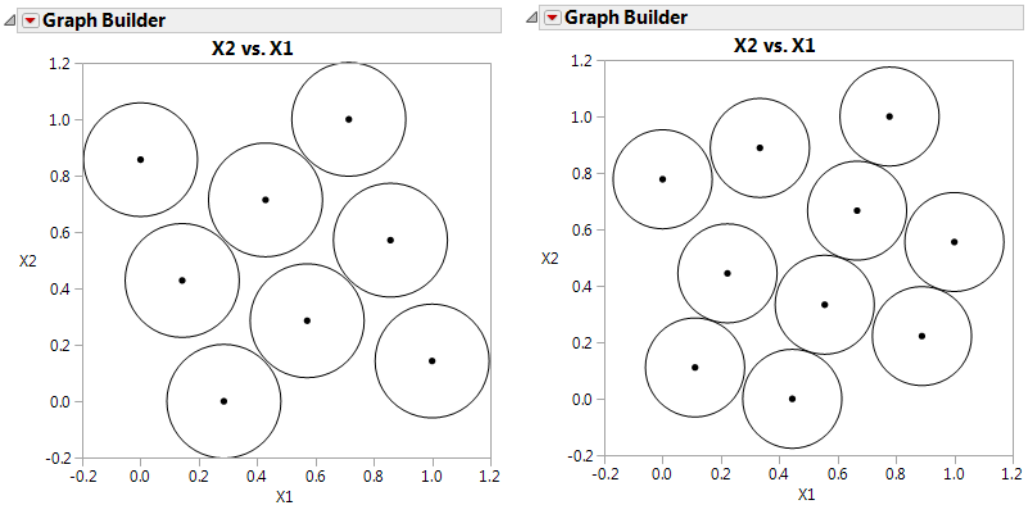
7. Click **Make Table**.
8. Select **Graph > Graph Builder**.
9. Specify X1 as **X** and X2 as **Y**.
10. Click the smoother icon to remove the smoother (blue line) from your graph.
11. Right-click the plot and select **Graph > Size/Scale > Size to Isometric** to adjust the frame size so that the frame is square.
12. Right-click the plot, select **Customize** from the menu. In the Customize panel, click the large plus sign to see a text edit area, and enter the following script:

    ```
    For Each Row(Circle({:X1, :X2}, 0.404/2))
    ```

    where `0.404` is the minimum distance number that you noted in the Design Diagnostics panel (Figure 21.12). This script draws a circle centered at each design point with radius 0.202 (half the diameter, 0.404), as shown on the left in Figure 21.13. This plot shows the efficient way JMP packs the design points.
13. Repeat the above procedure exactly, but with 10 runs instead of eight (step 5). Remember to change `0.404` in the graphics script to the minimum distance produced by 10 runs.

You should see a graph similar to the one on the right in Figure 21.13. Note the irregular nature of the sphere packing. In fact, you can repeat the process to get a slightly different picture because the arrangement is dependent on the random starting point.

Figure 21.13 Comparison of Latin Hypercube Designs with Eight Runs (left) and 10 Runs (right)



Note that the minimum distance between each pair of points in the Latin Hypercube design is smaller than that for the Sphere-Packing design. This is because the Latin Hypercube design constrains the levels of each factor to be evenly spaced. The Sphere-Packing design maximizes the minimum distance without any constraints.

Uniform Designs

The Uniform design minimizes the discrepancy between the design points (empirical uniform distribution) and a theoretical uniform distribution.

Note: These designs are most useful for getting a simple and precise estimate of the integral of an unknown function. The estimate is the average of the observed responses from the experiment.

1. Select **DOE > Special Purpose > Space Filling Design**.
2. Enter responses, if necessary, and factors.

   See "Responses".
3. Alter the factor level values to 0 and 1.
4. Click **Continue.**
5. In the design specification dialog, specify a sample size. This example uses a sample size of eight (**Number of Runs**).

6. Click the **Uniform** button. JMP creates this design and displays the design runs and the design diagnostics as shown in Figure 21.14.

Note: The emphasis of the Uniform design method is not to spread out the points. The minimum distances in Figure 21.14 vary substantially.

Figure 21.14 Factor Settings and Diagnostics for Uniform Space-Filling Designs with Eight Runs

2 Factors
Space Filling Uniform Design

Design

Run	X1	X2
1	0.93092	0.69041
2	0.69276	0.18514
3	0.06860	0.32218
4	0.18696	0.81304
5	0.56496	0.93540
6	0.81410	0.43443
7	0.32218	0.06860
8	0.43643	0.56357

Design Diagnostics

Run	ScaledX1	ScaledX2	Minimum Distance	Nearest Point
1	0.93092	0.69041	0.281	6
2	0.69276	0.18514	0.277	6
3	0.06860	0.32218	0.359	7
4	0.18696	0.81304	0.353	8
5	0.56496	0.93540	0.393	8
6	0.81410	0.43443	0.277	2
7	0.32218	0.06860	0.359	3
8	0.43643	0.56357	0.353	4

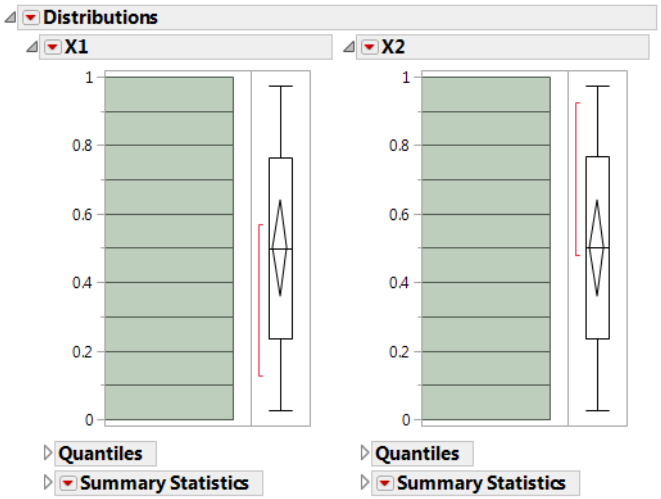
discrepancy = 0.004663

	MaxPro
Full Design	15.40271

7. Click **Make Table**.

A Uniform design does not guarantee even spacing of the factor levels. However, increasing the number of runs and running a distribution on each factor (use **Analyze** > **Distribution**) shows flat histograms.

Figure 21.15 Histograms Are Flat for Each Factor When Number of Runs Is Increased to 20



Comparing Sphere-Packing, Latin Hypercube, and Uniform Methods

To compare space-filling design methods, create the **Sphere Packing**, **Latin Hypercube**, and **Uniform** designs, as shown in the previous examples. The Design Diagnostics tables show the values for the factors scaled from zero to one. The minimum distance is based on these scaled values and is the minimum distance from each point to its closest neighbor. The discrepancy value is the integrated difference between the design points and the uniform distribution.

Figure 21.16 shows a comparison of the design diagnostics for three eight-run space-filling designs. Note that the discrepancy for the Uniform design is the smallest (best). The discrepancy for the Sphere-Packing design is the largest (worst). The discrepancy for the Latin Hypercube takes an intermediate value that is closer to the optimal value. The MaxPro criteria is smaller for the Latin Hypercube design as compared to the Uniform Design.

Also note that the minimum distance between pairs of points is largest (best) for the Sphere-Packing method. The Uniform design has pairs of points that are only about half as far apart. The Latin Hypercube design behaves more like the Sphere-Packing design in spreading the points out.

For both spread and discrepancy, the Latin Hypercube design represents a healthy compromise solution.

Figure 21.16 Comparison of Diagnostics for Three Eight-Run Space-Filling Methods

Space Filling Sphere Packing

Design

Design Diagnostics

Run	ScaledX1	ScaledX2	Minimum Distance	Nearest Point
1	0.50000	0.86602	0.518	3
2	0.86603	0.50000	0.518	6
3	0.13397	0.50000	0.518	1
4	0.50000	0.13398	0.518	7
5	0.00000	1.00000	0.518	1
6	1.00000	1.00000	0.518	2
7	1.00000	0.00000	0.518	4
8	0.00000	0.00000	0.518	3

discrepancy = 0.046969

	MaxPro
Full Design	.

Space Filling Latin Hypercube

Design

Design Diagnostics

Run	ScaledX1	ScaledX2	Minimum Distance	Nearest Point
1	0.28571	0.00000	0.404	3
2	0.00000	0.85714	0.452	4
3	0.57143	0.28571	0.404	1
4	0.42857	0.71429	0.404	5
5	0.14286	0.42857	0.404	4
6	1.00000	0.14286	0.452	7
7	0.85714	0.57143	0.404	3
8	0.71429	1.00000	0.404	4

discrepancy = 0.009279

	MaxPro
Full Design	11.32801

Space Filling Uniform Design

Design

Design Diagnostics

Run	ScaledX1	ScaledX2	Minimum Distance	Nearest Point
1	0.93092	0.69041	0.281	6
2	0.69276	0.18514	0.277	6
3	0.06860	0.32218	0.359	7
4	0.18696	0.81304	0.353	8
5	0.56496	0.93540	0.393	8
6	0.81410	0.43443	0.277	2
7	0.32218	0.06860	0.359	3
8	0.43643	0.56357	0.353	4

discrepancy = 0.004663

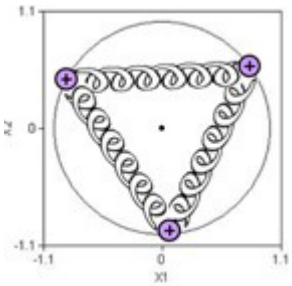
	MaxPro
Full Design	15.40271

Another point of comparison is the time it takes to compute a design. The Uniform design method requires the most time to compute. Also, the time to compute the design increases rapidly with the number of runs. For comparable problems, all the space-filling design methods take longer to compute than the *D*-optimal designs in the Custom Designer.

Minimum Potential Designs

The Minimum Potential design spreads points out inside a sphere. To understand how this design is created, imagine the points as electrons with springs attached to every other point, as illustrated in Figure 21.17. The coulomb force pushes the points apart, but the springs pull them together. The design is the spacing of points that minimizes the potential energy of the system.

Figure 21.17 Minimum Potential Design



Minimum Potential designs:

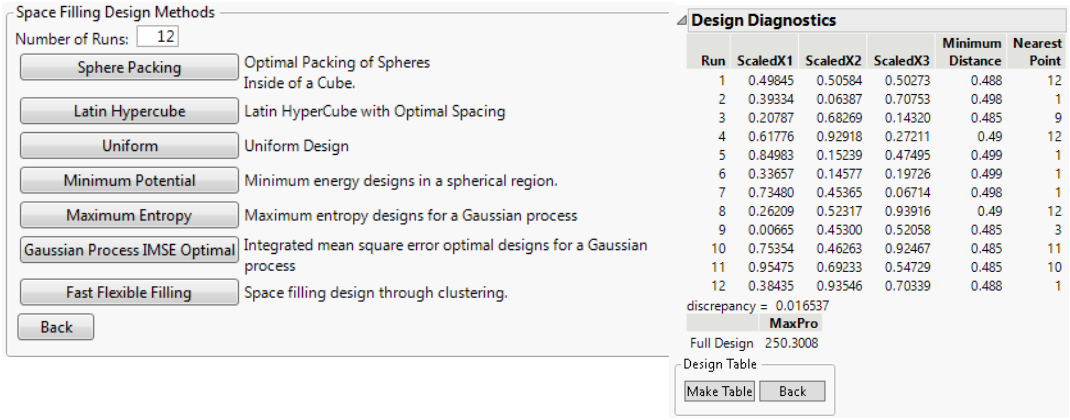
- have spherical symmetry
- are nearly orthogonal
- have uniform spacing

To construct a Minimum Potential example design:

1. Select **DOE > Special Purpose > Space Filling Design**.
2. Add 3 continuous factor.

   See "Factors".
3. Alter the factor level values to 0 and 1.
4. Click **Continue.**
5. In the design specification dialog (shown on the left in Figure 21.18), enter a sample size (**Number of Runs**). This example uses a sample size of 12.
6. Click the **Minimum Potential** button. JMP creates this design and displays the design runs and the design diagnostics (shown on the right in Figure 21.18).

Figure 21.18 Space-Filling Methods and Design Diagnostics for Minimum Potential Design



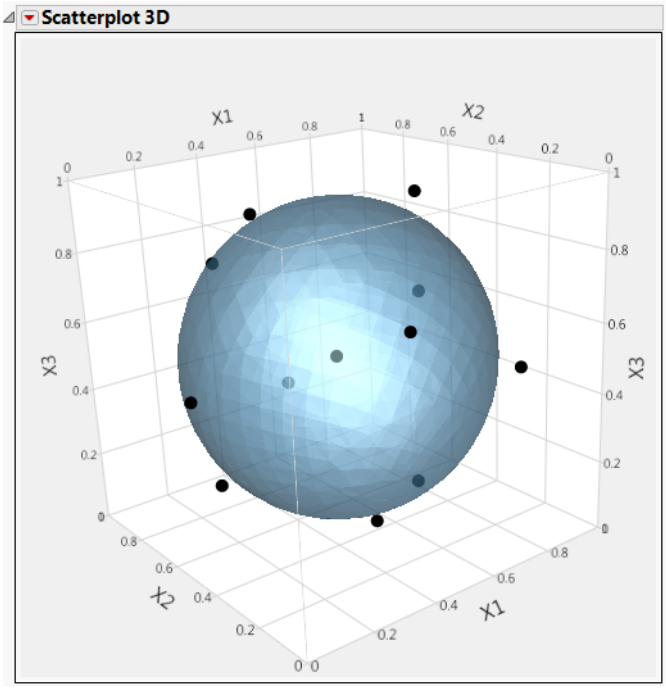
7. Click **Make Table**.

You can see the spherical symmetry of the Minimum Potential design using the Scatterplot 3D graphics platform.

1. After you make the JMP design table, choose the **Graph** > **Scatterplot 3D** command.
2. In the Scatterplot 3D launch dialog, select X1, X2, and X3 as **Y, Columns** and click **OK** to see the initial three-dimensional scatterplot of the design points.
3. To see the results similar to those in Figure 21.19:
   - Click the Scatterplot 3D red triangle and select the **Normal Contour Ellipsoids**.
   - Right-click the plot and select **Settings**, and then increase the marker size using the **Marker Size** slider.

Now the even spread of the points on the surface of the ellipsoid can be visualized.

Figure 21.19 Minimum Potential Design Points on Sphere

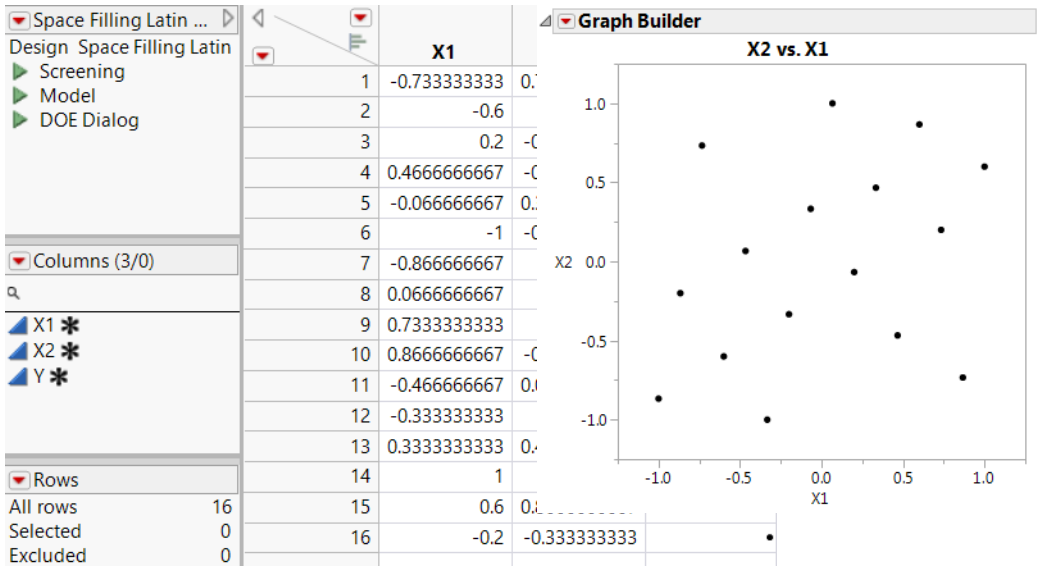


Maximum Entropy Designs

The Maximum Entropy design is an alternative to the Latin Hypercube design for computer experiments. The Latin Hypercube design is a popular design to use along with a Gaussian-Process model. Computer simulation experts like to use the Latin Hypercube design because all projections onto the coordinate axes are uniform.

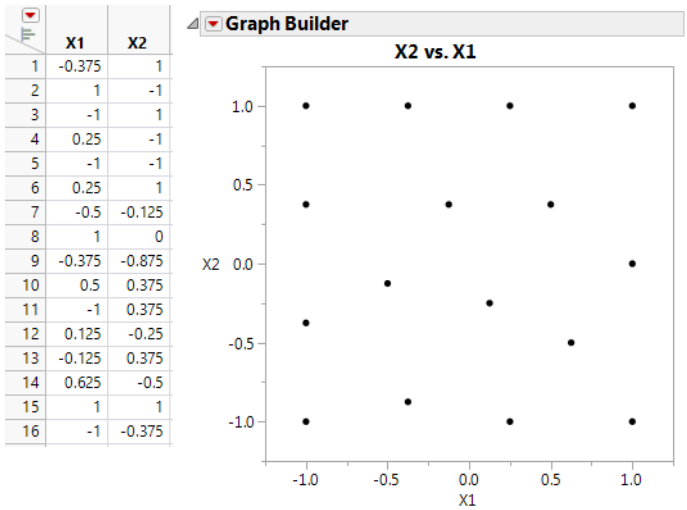
However, as the example in Figure 21.20 shows, the Latin Hypercube design might not provide you with optimal space filling properties. This example is a two-factor 16 run Latin Hypercube with factor level settings set between -1 and 1. The plot of the two factors shows that this design has regions that are missing coverage. In particular, there is poor coverage for X1 near 0 and X2 near -1.

Figure 21.20 Two-factor Latin Hypercube Design



The Maximum Entropy design optimizes a measure of the amount of information contained in an experiment. See the technical note below. With the factor levels set between -1 and 1, the two-factor Maximum Entropy design shown in Figure 21.21 covers the region better than the Latin hypercube design in Figure 20. The space-filling property of the design improves as the number of runs increases.

Figure 21.21 Two-Factor Maximum Entropy Design



Technical Maximum Entropy designs maximize the Shannon information (Shewry and Wynn (1987)) of an experiment, assuming that the data come from a normal (m, s^2 R) distribution, where

$$R_{ij} = \exp\left(-\sum_k \theta_k (x_{ik} - x_{jk})^2\right)$$

is the correlation of response values at two different design points, x_i and x_j. Computationally, these designs maximize $|\mathbf{R}|$, the determinant of the correlation matrix of the sample. If x_i and x_j are far apart, then $\mathbf{R}_{ij}$ approaches zero. If x_i and x_j are close together, then $\mathbf{R}_{ij}$ is near one.

Gaussian Process IMSE Optimal Designs

The Gaussian process IMSE optimal design method constructs designs that are suitable for Gaussian process models. Gaussian process models fit a wide variety of surfaces. Gaussian process IMSE optimal designs minimize the integrated mean squared error of the Gaussian process model over the experimental region. The Gaussian process IMSE optimal design method uses a correlation structure similar to that of the kriging model. See Jones and Johnson (2009).

Covariance Parameter Vector

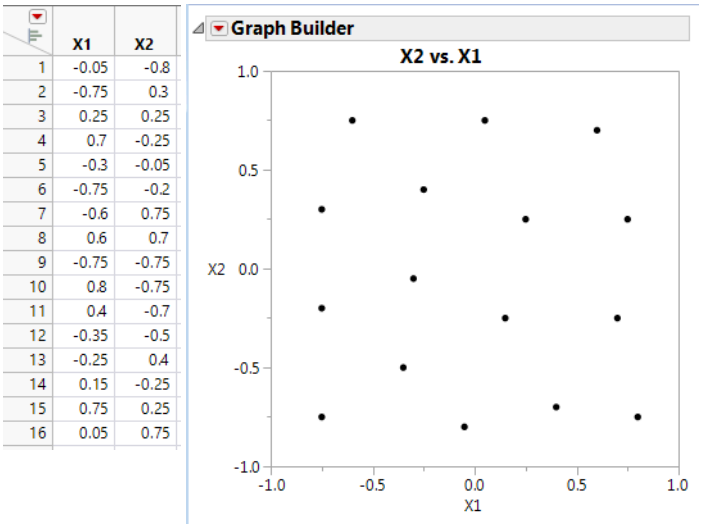
In a Gaussian Process IMSE Optimal Design formulation of the Gaussian process model, the covariance parameter vector determines the correlation structure. There is a Theta for each factor. A theta equal to 0 corresponds to a correlation of 1, causing the fitted surface to be flat in the corresponding factor's direction. As theta increases, the correlation decreases, allowing the surface to be flexible in the factor's direction.

In the Covariance Parameter Vector outline, in the list of values under Thetas, you can enter values that reflect your prior knowledge of the surface.

Comparison of Gaussian Process IMSE Optional Design with Latin Hypercube Design

Gaussian process IMSE optimal designs are alternatives to the Latin Hypercube design. You can compare the IMSE optimal design to the Latin Hypercube (shown previously in Figure 21.20). The table and overlay plot in Figure 21.22 show a Gaussian IMSE optimal design. You can see that the design provides uniform coverage of the factor region.

Figure 21.22 Comparison of Two-factor Latin Hypercube and Gaussian IMSE Optimal Designs



	X1	X2
1	-0.05	-0.8
2	-0.75	0.3
3	0.25	0.25
4	0.7	-0.25
5	-0.3	-0.05
6	-0.75	-0.2
7	-0.6	0.75
8	0.6	0.7
9	-0.75	-0.75
10	0.8	-0.75
11	0.4	-0.7
12	-0.35	-0.5
13	-0.25	0.4
14	0.15	-0.25
15	0.75	0.25
16	0.05	0.75

Note: Both the Maximum Entropy design and the Gaussian Process IMSE Optimal design were created using 100 random starts.

Fast Flexible Filling Designs

The algorithms for Fast Flexible Filling designs begin by generating a large number of random points within the specified design region. These points are then clustered using a Fast Ward algorithm into a number of clusters that equals the Number of Runs that you specified.

The final design points can be obtained by using the default MaxPro (*maximum projection*) optimality criterion or by selecting the Centroid criterion.

Note: If you have Categorical factors or factor constraints, then Fast Flexible Filling is the only space filling design Method available.

FFF Optimality Criterion

The Space Filling Design red triangle menu contains optimality criteria under FFF Optimality Criterion.

MaxPro For p factors and n equal to the specified Number of Runs, the MaxPro criterion strives to find points in the clusters that minimize the following criterion:

$$C_{MaxPro} = \sum_{i}^{n-1} \sum_{j=i+1}^{n} \left[1 / \prod_{k=1}^{p} (x_{ik} - x_{jk})^2 \right]$$

The MaxPro criterion maximizes the product of the distances between potential design points in a way that involves all factors. This supports the goal of providing good space-filling properties on projections of factors. See Joseph et al. (2015). The Max Pro option is the default.

Centroid This method places a design point at the centroid of each cluster. It has the property that the average distance from an arbitrary point in the design space to its closest neighboring design point is smaller than for other designs.

Note: You can set a preference to always use a given optimality criterion. Select **File > Preferences > Platforms > DOE**. Select **FFF Optimality Criterion** and select your preferred criterion.

Categorical Factors

When you have categorical factors, the algorithm proceeds as follows:

- The total number of design points is balanced across the total number of combinations of levels of the categorical factors. Suppose that there are m combinations of levels and that k design points are allocated to each of these.
- A large number of points within the design space defined by the continuous variables is generated. These are grouped into k primary clusters.
- Each of the k primary clusters of points is further clustered into m sub-clusters.
- Within each primary cluster, a design point is calculated for each of the m sub-clusters using the specified FFF optimality criterion.
- For each of the k primary clusters, one of the m combinations of levels is randomly assigned to each of the m sub-cluster design points. This yields a total of km design points.
- For each of the k primary clusters, a design point is chosen for each of the m combinations of levels according to the MaxPro criterion, where points having the same level of a categorical factor are multiplied by the MaxPro Categorical Weight. (For a description of MaxPro Categorical Weight, see "Space Filling Design Options".) This process continues through all k primary clusters 10 times, or until no improvement is found by changing the given design points.

Set Average Cluster Size for FFF Designs

The Set Average Cluster Size option enables you to specify the average number of randomly-generated points used to define each cluster or, equivalently, each design point. The option is found under Advanced Options in the Space Filling Design red triangle menu.

By default, if the Number of Runs is set to 200 or less, a total of 10,000 randomly generated points are used as the basis for the clustering algorithm. When the number of Runs exceeds 200, a default value of 50 is used. Increasing this value can be particularly useful in designs with a large number of factors or where disallowed combinations restrict the distribution of points used in the clustering algorithm.

Note: Depending on the number of factors and the specified value for Number of Runs, you might want to increase the average number of initial points per design point by selecting **Advanced Options > Set Average Cluster Size.**

Constraints

Once you complete the Factors outline, click **Continue**. The Define Factor Constraints outline appears. Use this outline to restrict the design region. For more information about the outline, see "Define Factor Constraints".

You can use the Use Disallowed Combinations Filter and Use Disallowed Combinations Script options to specify disallowed factor level combinations. Or, you can use the Specify Linear Constraints option to specify bounds in terms of linear inequalities. However, the design is generated differently for these two methods.

Use Disallowed Combinations Filter and Use Disallowed Combinations Script

When disallowed combinations are specified, the random points that form the basis for the clustering algorithm are randomly distributed within the unconstrained design region. Then disallowed points are removed and clustering proceeds with the remaining points.

Note: Depending on the nature of the constraints and the specified Number of Runs, the default coverage of the unconstrained design space by the initial randomly generated points might not be sufficient to produce the required Number of Runs. In this case, you might obtain a JMP Alert indicating that the algorithm "Could not find sufficient number of points." To increase the initial number of points that form the basis for the clustering algorithm, specify a larger average number of initial points per design point by selecting **Advanced Options > Set Average Cluster Size.** (See "Set Average Cluster Size for FFF Designs").

Specify Linear Constraints

When you use the **Specify Linear Constraints** option, the random points that form the basis for the clustering algorithm are randomly distributed within the constrained design region. The clustering algorithm uses these points.

Creating and Viewing a Constrained Fast Flexible Filling Design

This example creates and views a constrained fast flexible filling design.

Constructing the Design

1. Select **DOE > Special Purpose > Space Filling Design**.
2. Add 2 continuous factors.
3. Alter the factor level values to 0 and 1 and click **Continue.**
4. In the Define Factor Constraints outline, select **Specify Linear Constraints**.

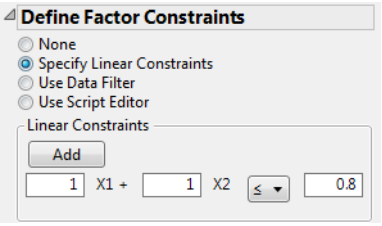
   Notice that Fast Flexible Filling is the only available Space Filling Design Method.
5. Select **Add**.
6. Enter the following coefficients and bound:

   1 for X1

   1 for X2

   0.8 for the bound

Figure 21.23 Linear Constraint



7. Type 200 next to **Number of Runs**.
8. Select **Fast Flexible Filling**.

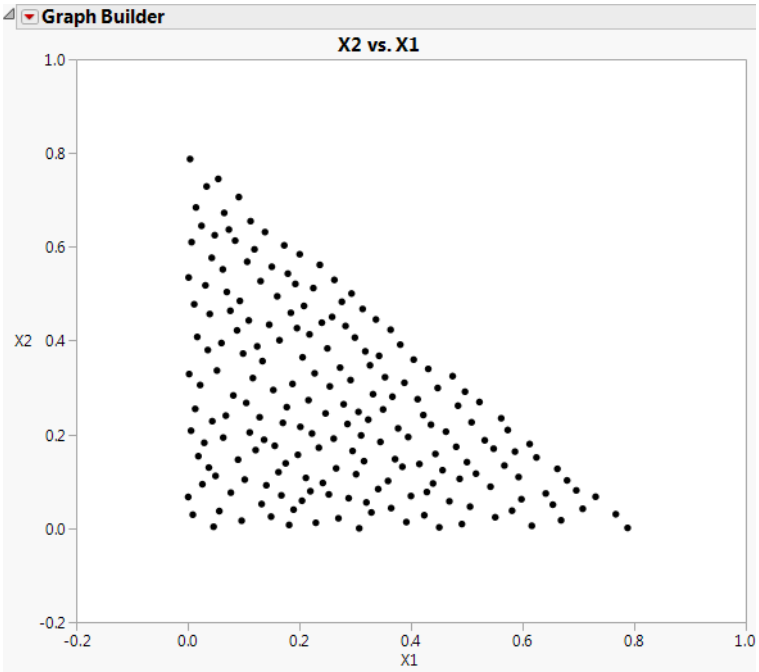
   JMP creates a design that satisfies the constraints. Open the Design outline to view the design.
9. Select **Make Table** to construct the data table.

Constructing the Plot

1. With the data table active, select **Graph > Graph Builder**.
2. Drag X1 to the drop zone labeled **X**.
3. Drag X2 to the drop zone labeled **Y**.
4. Remove the **Smoother** by clicking the smoother icon.
5. Click the Graph Builder red triangle and click **Show Control Panel** to deselect it.

You should see a graph similar to the one in Figure 21.24. Note that the points satisfy the linear constraint $X1 + X2 \leq 0.8$.

Figure 21.24 Fast Flexible Filling Design with One Linear Constraint



Creating a Space-Filling Design for a Map Shape

A fast flexible filling (FFF) design can be used to design an experiment to be executed across a region such as a map shape. For example, if you had two sample collection methods (perhaps for air or soil samples) that you wanted to compare at locations spread through out a US state, consider a FFF design.

Supposed the US state of interest is Georgia. You want 40 samples, 20 for each sampling method, spread throughout the state.To create a FFF design for a map shape, you need a file with the latitude and longitude of the map boundaries (see "Background Maps").

In the following example, you determine the longitude and latitude for the US state of Georgia, define the shape of the state, then create the design. The design will provide the location and the sampling method for each of your 40 samples.

Find Longitude and Latitude for Georgia

1. Navigate to the Maps directory, usually found here:

   Windows: C:/Program Files/SAS/JMP/<Version>/Maps

   macOS: /Library/Application Support/JMP/<Version>/Maps
2. Open US-State-XY.jmp and US-State-Name.jmp.

   Notice that in US-State-Name.jmp, Shape ID 11 corresponds to the US state of Georgia. In this example, that is the state of interest.
3. In US-State-XY.jmp, select **Rows > Row Selection > Select Where...**
4. Select Shape ID and enter 11 in the text box, then click **OK**.

   This selects all rows with a Shape ID of 11, representing Georgia.
5. Select **Tables > Subset** and click **OK** to obtain a subset of the US-State-XY.jmp table of the latitudes and longitudes data for the state of Georgia (the selected rows).
6. Close US-State-XY.jmp and US-State-Name.jmp.

Get the Range of Values

To define the space for the space filling design, you need to know the range of values for the latitude and longitudes in Georgia.Then, you can create design factors with these values.

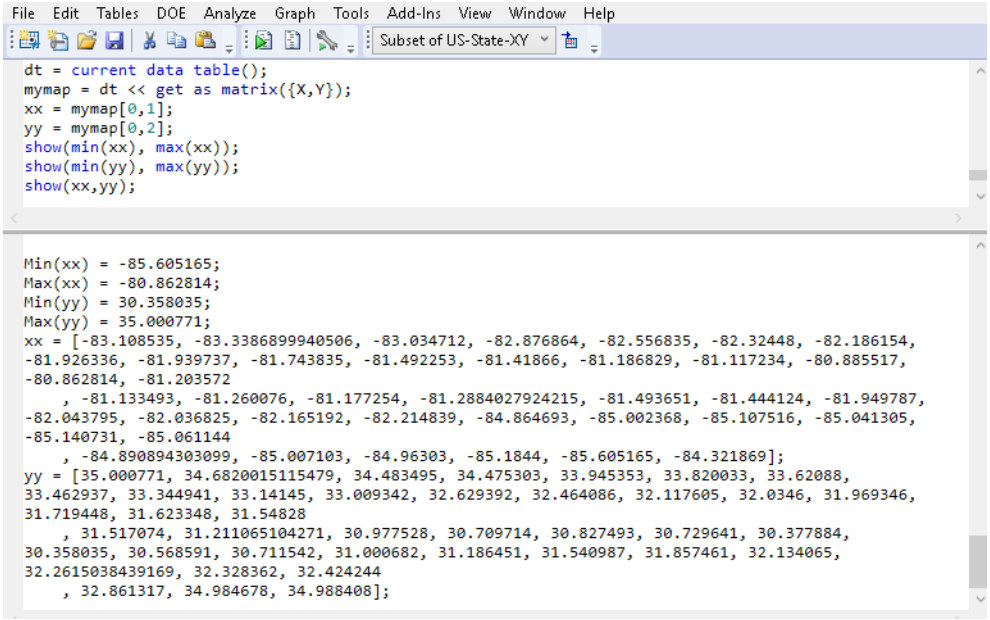
1. Select **File** > **New** > **Script**. In the script window, right-click and select **Show Embedded Log**.
2. Copy the following and paste it into the new script:

```
dt = current data table();
mymap = dt << get as matrix({X,Y});
xx = mymap[0,1];
yy = mymap[0,2];
show(min(xx), max(xx));
show(min(yy), max(yy));
show(xx,yy);
```

3. With the subset of the US-State-XY.jmp table active, run the script.

   The range of values for the latitudes and longitudes in Georgia and the **xx** and **yy** matrices appear in the log.

Figure 21.25 Range Finding Script and Results in Log



Add Factors

1. Select **DOE > Special Purpose > Space Filling Design.**
2. Add two continuous factors to the design. Enter 2 in the **Add N Factors** text box and click **Continuous**.
3. Rename the factors longitude and latitude and enter their min and max values in the Factor outline:
   - For longitude, enter -85.6 and -80.9 as the values. These are the min and max longitude values from the log for *x*.
   - For latitude, enter 30.4 and 35.0 as the values. These are the min and max latitude values from the log for *y*.
4. Add a categorical factor. Click **Categorical** > **2 Level**. Rename the factor Method.

Figure 21.26 Factors

Factors

Name	Role	Values	
longitude	Continuous	-85.6	-80.7
latitude	Continuous	30.3	35
Method	Categorical	L1	L2

5. Click **Continue**.

Define the Shape

To further define the space for the space filling design, you need to define the shape of the state using a polygon.

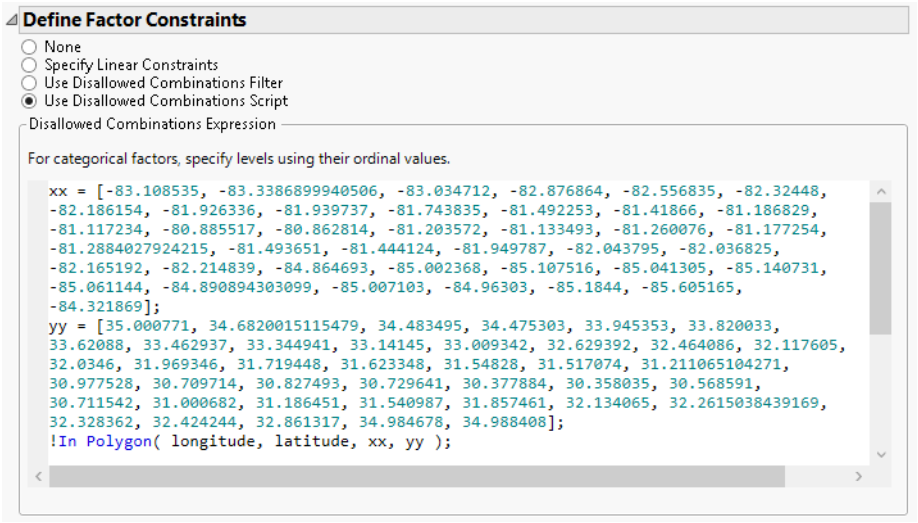
1. Select **Use Disallowed Combinations Script** to open the Disallowed Combinations Expression text box.

   The ranges of longitude and latitude define a square. The state of Georgia is a polygon, not a square. You want to disallow points that fall outside of the state of Georgia polygon.
2. Go to your script window and copy the **xx** and **yy** matrices from your script log. Paste this into the Disallowed Combinations Expression text box.
3. At the end, add the following line:

   ```
   !In Polygon(longitude, latitude , xx, yy);
   ```

   This excludes points that are not in the polygon defined by the **xx** and **yy** coordinates.

Figure 21.27 Disallowed Combinations Expression

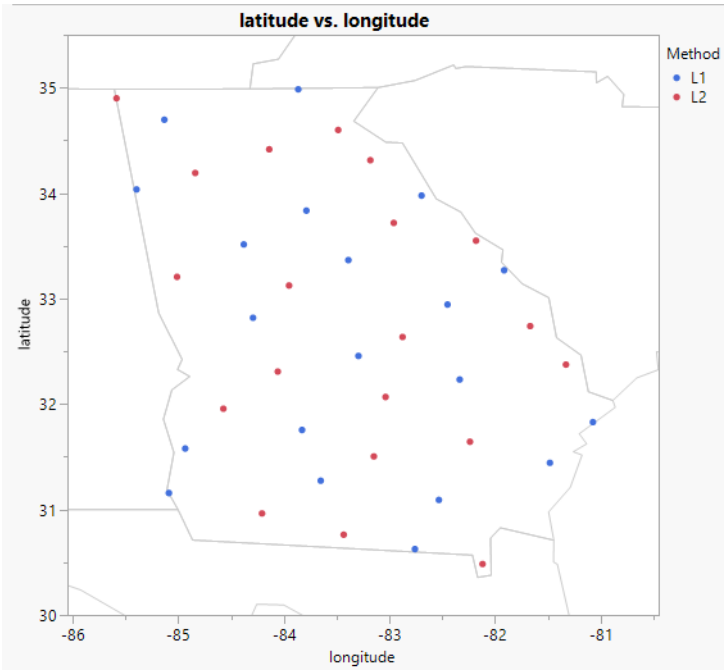


Make the Design

1. Change the **Number of Runs** to 40 and click **Fast Flexible Filling**.
2. Click **Make Table.**
3. From your design table, select **Graph > Graph Builder**.
4. Drag these columns into these zones:
   - longitude to the **X** zone
   - latitude to the **Y** zone
   - Method to the **Colo**r zone

5. Above the graph, click the Smoother element icon to turn it off.
6. To add the state shape, right-click the graph and select **Graph > Background Map**.
7. Under **Boundaries**, select **US States** and click **OK**.

Figure 21.28 Example of FFF Design for the US State of Georgia



Note: The exact location of the points on your map may differ from the one shown due to the random starting point for the FFF algorithm.

The map identifies the 40 locations where you should collect your samples. The sampling method (L1 or L2) is indicated by the color of the points.

The Graph > Background Map window has these additional options:

- To determine the counties for your testing, under **Boundaries**, select **US Counties**.
- To map your testing locations at the street level, under **Images**, select **Street Map Service**.

Example of a Sphere-Packing Design

Worley (1987) presented a model of the flow of water through a borehole that is drilled from the ground surface through two aquifers. The response variable y is the flow rate through the borehole in m^3/year and is determined by the following equation:

$$y = \frac{2\pi T_u (H_u - H_l)}{\ln(r/r_w)\left[1 + \frac{2LT_u}{\ln(r/r_w) r_w^2 K_w} + \frac{T_u}{T_l}\right]}$$

There are eight factors in this model:

r_w = radius of borehole, 0.05 to 0.15 m

r = radius of influence, 100 to 50,000 m

T_u = transmissivity of upper aquifer, 63,070 to 115,600 m^2/year

H_u = potentiometric head of upper aquifer, 990 to 1100 m

T_l = transmissivity of lower aquifer, 63.1 to 116 m^2/year

H_l = potentiometric head of lower aquifer, 700 to 820 m

L = length of borehole, 1120 to 1680 m

K_w = hydraulic conductivity of borehole, 9855 to 12,045 m/year

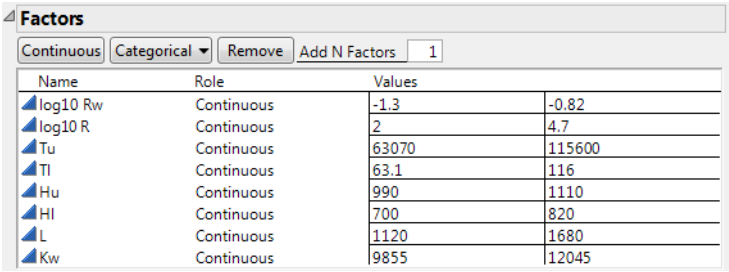
You can use a sphere-packing design to obtain a set of conditions for which to calculate the response y. Then you can build a model to estimate the true model over the range of inputs used in your design. Evaluation of your estimated model can help you understand the impact of each of the eight factors on the response.

Create the Sphere-Packing Design for the Borehole Data

To create a sphere-packing design for the borehole model, you can use a data table of saved factor settings.

1. Select **Help > Sample Data Library** and open Design Experiment/Borehole Factors.jmp
2. Select **DOE > Special Purpose > Space Filling Design**
3. Click the Space Filling Design red triangle and select **Load Factors**.

Figure 21.29 Factors Panel with Factor Values Loaded for Borehole Example



Factors

Continuous | Categorical | Remove | Add N Factors 1

Name	Role	Values	
log10 Rw	Continuous	-1.3	-0.82
log10 R	Continuous	2	4.7
Tu	Continuous	63070	115600
Tl	Continuous	63.1	116
Hu	Continuous	990	1110
Hl	Continuous	700	820
L	Continuous	1120	1680
Kw	Continuous	9855	12045

Note: The logarithm of r and r_w are used as factors.

4. Click **Continue**.
5. Set the Number of Runs to 32 and press Enter.
6. Click **Sphere Packing** to produce the design.
7. Click **Make Table** to make a table showing the design settings for the experiment.

Note: The design table includes a Model table script. This script runs a Gaussian Process model for the response y.

8. (Optional) To see a completed data table for this example, select **Help > Sample Data Library** and open Design Experiment/ Borehole Sphere Packing.jmp.

   Because the designs are generated from a random seed, the settings that you obtain can differ from those shown in the completed table.

Guidelines for the Analysis of Deterministic Data

It is important to remember that deterministic data have no random component. The same input values generate the same output. As a result, p-values from fitted statistical models do not have their usual meanings. A large F statistic (low p-value) is an indication of an effect due to a model term. However, you cannot construct valid confidence intervals for effects or model predictions.

Residuals from a model fit to deterministic data are not a measure of noise. Instead, residuals are a measure of the model bias. Bias is the difference between the true value and the predicted value. Distinct patterns in the residuals indicate that additional terms should be considered for the model in order to reduce bias.

Analysis of the Borehole Sphere-Packing Design

Often, the true model is not available in a simple analytical form. As a result, the prediction bias is known only at observed data points. However, in this example, the functional form of the true model is known. In the Borehole Sphere Packing.jmp data table, the true model column contains the formula of the known function. This formula enables you to profile the prediction bias over the factor input region.

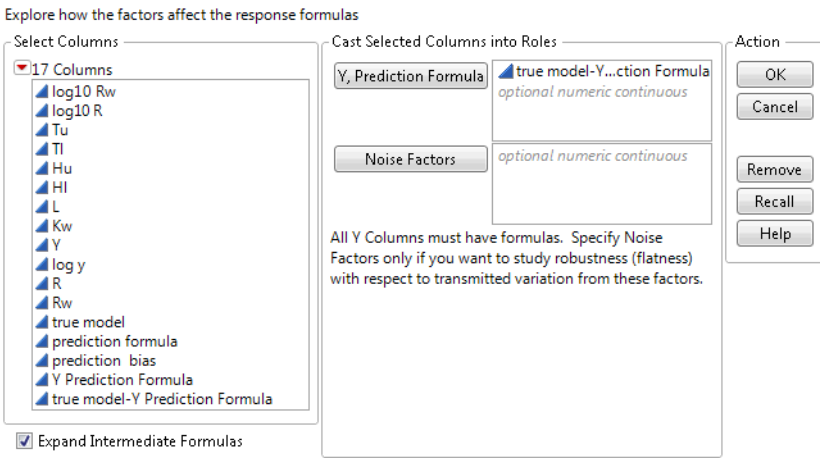
1. Select **Help > Sample Data Library** and open Design Experiment/Borehole Sphere Packing.jmp.
2. Click the green triangle next to the Model (GP from DOE) script.

   Use the Gaussian Process Model report to explore the relationships between the factors and the outcome Y.
3. Click the red triangle next to Gaussian Process Model of Y and select **Save Prediction Formula**.
4. Go back to the Borehole Sphere Packing.jmp data table.
5. In the data grid, select the column headings for true model and Y Prediction Formula.
6. Right-click and select **New Formula Column > Combine > Difference.**

   This creates a new column containing the bias.
7. From the Borehole Sphere Packing.jmp data table, select **Graph > Profiler.**
8. Select true model-Y Prediction Formula and click **Y, Prediction Formula**
9. Select **Expand Intermediate Formulas**.

   This option shows the bias as a function of the eight design factors.

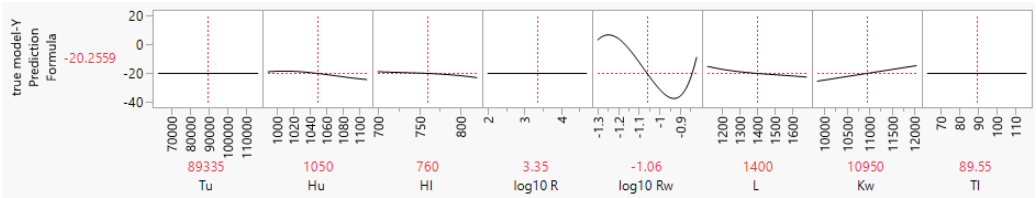
Figure 21.30 Profiler Dialog for Borehole Sphere-Packing Data



10. Click **OK**.

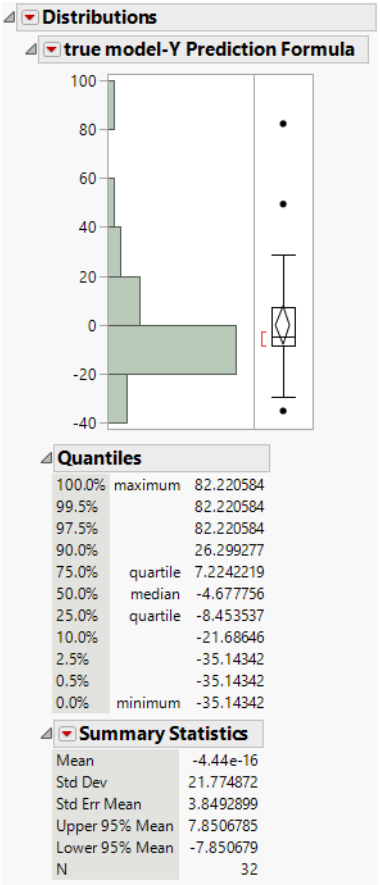
The profiler defaults to the center of the design region. If there were no bias, all profile traces would be constant between the value ranges of each factor. In this example, the variables logRw, Kw, and L show the largest effects on the bias.

Figure 21.31 Profiler for Bias of the Borehole GP Model with Y Axis Set at -40 to 20



You can use the profiler to explore the range of the prediction bias over the entire domain. To find points of minimum and maximum bias, select **Optimization and Desirability** > **Desirability Functions** from the Prediction Profiler red triangle menu. See *Profilers*. To evaluate the prediction bias over the design points, select **Analyze > Distribution** to see a distribution analysis.

Figure 21.32 Distribution of the Prediction Bias



Keep in mind that, in this example, the true model is known. In many applications, the response at any factor setting is unknown. The prediction bias over the experimental data can underestimate the bias throughout the design domain.

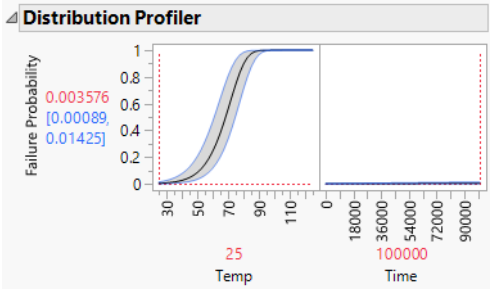
Chapter 22

Accelerated Life Test Designs

Designing Experiments for Accelerated Life Tests

Use the Accelerated Life Test (ALT) Design platform to design plans for accelerated life testing experiments. Product reliability at normal use conditions is often so high that the time required to test the product until it fails is prohibitive. Rather than test the product at normal use conditions, you can test the product under conditions that are more severe than normal use conditions. These severe conditions can cause the product to degrade faster and fail more quickly. You can then use this accelerated failure data to predict product reliability at normal use conditions.You can design initial experiments or augment existing experiments.

Figure 22.1 Profiler Showing Failure Probabilities for ALT Experiment



Contents

Overview of Accelerated Life Test Designs 653
Example of an Accelerated Life Test Design 653
Example of Augmenting an Accelerated Life Test Design 660
Build an Accelerated Life Test Design 665
Specify the Design Structure 666
Specify Acceleration Factors 666
Specify Design Details 667
Review and Update Specifications for ALT Plans 669
Create and Assess the Optimal 671
Create Design Tables 673
Accelerated Life Test Options 673
Statistical Details for the ALT Design Platform 674
Failure Distributions for ALT Designs 674
R Precision Factor (for a 95% CI) 676
Optimality Criterion for Accelerated Life Tests 677

Overview of Accelerated Life Test Designs

Use accelerated life testing when product reliability at normal use conditions is high. You test the product under conditions that are more severe than normal to accelerate the time until failures occur. You use this accelerated failure data to predict product reliability at normal use conditions.

Such a test is called an *accelerated life test* (ALT). The factors that are set to levels that accelerate the time to failure are called *acceleration factors*. The models for accelerated life tests are typically nonlinear models. For more information about nonlinear models, see "Nonlinear Models".

The ALT Design platform creates and evaluates designs for situations involving one or two acceleration factors. You can optimize your design using D-optimality, quantile estimation, or failure probabilities.

Creating an ALT design requires initial estimates of the acceleration model parameters. Use subject matter knowledge to define starting estimates. If you do not have information to inform your starting estimates, you can specify a multivariate normal prior distribution to describe their uncertainty.

You can also use the Accelerated Life Test (ALT) Design platform to augment a current experiment. You might want to augment a current design to decrease the variance of estimates.

Example of an Accelerated Life Test Design

Use the ALT designer to construct a failure probability optimal ALT test design. In this example, suppose that you need to design an accelerated life test for a mechanical component. The single acceleration factor is torque, and the normal use stress is 35 Nm (newton meters). You want to estimate the time at which 10% of the units fail at the normal use stress. You construct a failure probability optimal ALT test.

Your test plan has the following characteristics:

- A total of 100 units are available for testing.
- The life distribution is assumed to be Weibull.
- The life-stress relationship is given by the logarithmic transformation.
- You plan to test torque at three stress levels: 50, 75, and 100 Nm.
- You have prior knowledge to predict initial failure times at the test levels. See "Obtain Prior Estimates".
- The test runs for 5,000 cycles.

- You will monitor the process for failures on a continuous basis.

Obtain Prior Estimates

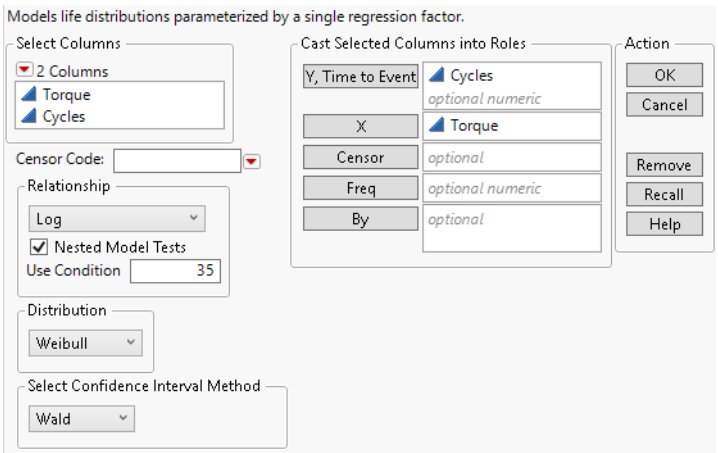
To create an accelerated life test design, you must provide prior estimates of the parameters. Here is an approach to obtaining prior estimates:

1. Use your process knowledge to create a table of hypothetical, but likely, failure times at a small number of stress levels.
2. Use the Fit Life by X platform to fit a model and obtain estimates of the model parameters.
3. Use these estimates as your prior values for creating a design using the ALT Design platform.

Following the approach outlined above, you create a data table containing estimates of the number of failure cycles for a balanced design. Your table consists of five units at each of the three stress levels to be used in your design. Your data table is Torque Prior.jmp, found in the Sample Data Library.

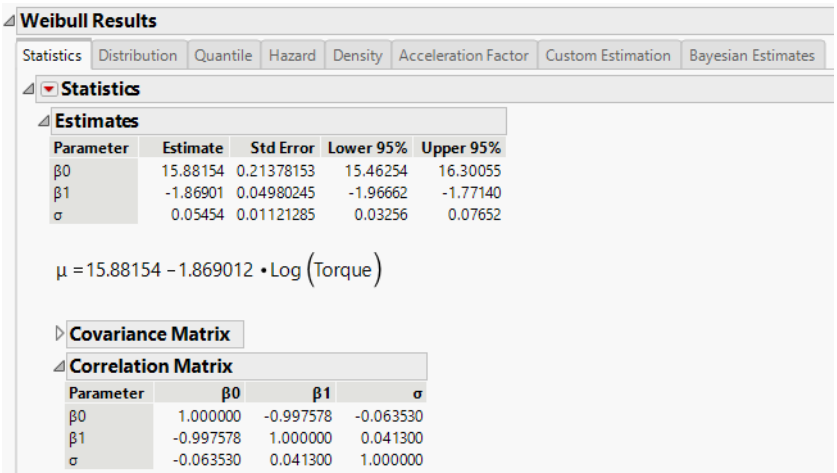
1. Select **Help > Sample Data Library** and open Design Experiment/Torque Prior.jmp.
2. Select **Analyze > Reliability and Survival > Fit Life by X**.
3. Select Cycles and click **Y, Time to Event**.
4. Select Torque and click **X**.
5. From the **Relationship** list, select **Log**.
6. Enter 35 for the **Use Condition**. Recall that 35 Nm is the normal use condition.
7. From the **Distribution** list, select **Weibull**.

Figure 22.2 Fit Life by X Launch Window



8. Click **OK**.
9. Scroll down to the Weibull Results outline and open the Correlation Matrix outline.

Figure 22.3 Fit Life by X Model for Prior Data



The model for the mean is given in the Estimates outline. The Estimate column contains the parameter estimates for the intercept (β0), the linear coefficient (β1), and the scale (σ). The standard errors column contains the standard errors for the estimates. The estimated correlations for the parameter estimates are given in the Correlation Matrix outline. Use these parameter estimates, standard errors, and correlations as your prior values to construct your ALT design.

Enter Basic Specifications

To create an accelerated life test design define accelerating factors, use conditions, and test conditions.

1. Select **DOE** > **Special Purpose > Accelerated Life Test Design**.

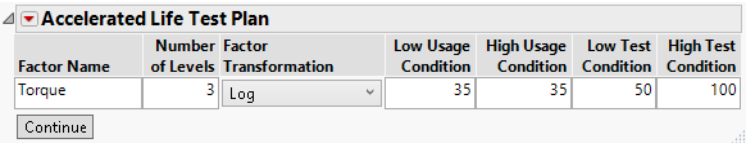
   Notice that **One accelerating factor** and **Continuous Monitoring** are selected by default.
2. Click **Continue**.
3. Under **Factor Name**, click X1 and type Torque.

   Notice that the **Number of Levels** is set to 3 by default.
4. Select **Log** under Factor Transformation.
5. Enter 35 for both the **Low Usage Condition** and the **High Usage Condition**.

   Setting both the low and high usage conditions to 35 indicates that 35 is the normal usage condition.

6. Enter 50 for the **Low Test Condition** and 100 for the **High Test Condition.**

Figure 22.4 Completed ALT Specification Window



7. Click **Continue**.
8. Notice the **Torque Level Values** are set to 50, 75, and 100.

Note: JMP sets the values of the levels evenly spaced between the low and high test conditions. You can change the levels.

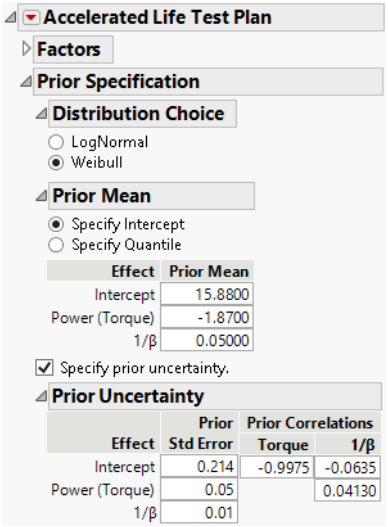
9. Ensure that **Weibull** is selected as the **Distribution Choice**.

Enter Prior Information and Remaining Specifications

Continue the construction of the example ALT design by entering prior information.

1. Under **Prior Mean**, select **Specify Intercept** and enter the prior acceleration model parameter estimates that you obtained using the Fit Life by X platform with your hypothetical data:
   - Enter 15.88 for **Intercept**.
   - Enter -1.87 for **Power (Torque)**.
   - Enter 0.05 for **1/**β (scale parameter for a Weibull model).
2. Select **Specify prior uncertainty**.
3. Enter the estimated standard errors and correlations from the prior acceleration model. Figure 22.3 shows the completed Prior Uncertainty outline:

Figure 22.5 Completed Prior Specification Outline

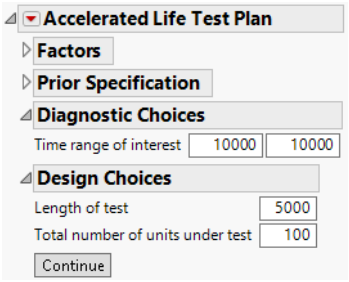


Click the Accelerated Life Test Plan red triangle and select **ALT Optimality Criterion > Make Failure Probability Optimal**.

You plan to conduct the test over 5,000 cycles. You are interested in predictions for as many as 10,000 cycles.

4. In the Diagnostic Choices outline, enter 10,000 for both boxes for **Time range of interest**.
5. In the Design Choices outline. Enter 5,000 for **Length of test** and 100 for **Number of units under test**.

Figure 22.6 Completed Diagnostic and Design Choices Outline

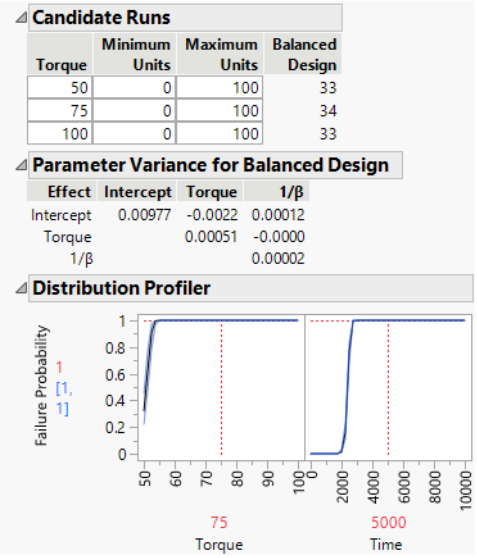


Make the Design

Continue the ALT example by generating the design.

1. Click **Continue**.

Figure 22.7 Balanced Design Diagnostics



Candidate Runs

Torque	Minimum Units	Maximum Units	Balanced Design
50	0	100	33
75	0	100	34
100	0	100	33

Parameter Variance for Balanced Design

Effect	Intercept	Torque	1/β
Intercept	0.00977	-0.0022	0.00012
Torque		0.00051	-0.0000
1/β			0.00002

The number of runs in a balanced design appears in the Candidate Runs outline. The Parameter Variance for Balanced Design outline shows the covariance matrix for the parameters for this design. The Distribution Profiler also appears. When you obtain the optimal design, you can compare the Parameter Variance and Distribution Profiler results to those for the balanced design to see the reduction in uncertainty.

2. Click **Make Design**.

   The optimal experimental design appears, along with other results.

Figure 22.8 Optimal Design

Design

Torque	N Units	Expected Failures	All Censored Probability	Failure Probability
50	81	26.1	0.0	0.32
75	0	0.0	0.0	1
100	19	19.0	0.0	1

The optimal design is computed based on the levels of the test runs, the total number of units to be tested, and the prior information that you specified. The optimal design consists of testing the specified number of units at each torque level:

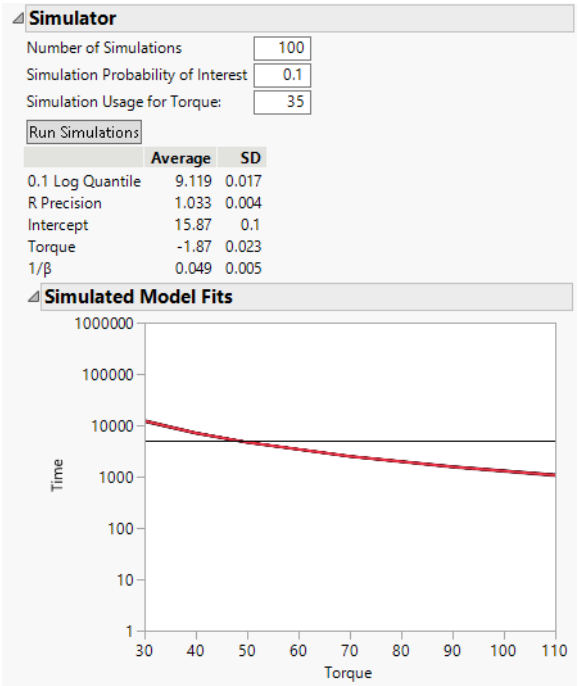
- 81 units at 50 Nm
- 0 units at 75 Nm
- 19 units at 100 Nm

Run Simulations

Evaluate the example ALT design with the simulator. Use the simulator to simulate the results of your test plan based on the assumptions used to build the design. This is a way to evaluate if the test plan is sufficient before running the test plan.

1. Enter 100 for the **Number of Simulations.**
2. Enter 0.1 for the **Simulation Probability of Interest.**
3. Enter 35 for the **Simulation Usage for Torque**.
4. Click **Run Simulations** to run 100 simulations of the experiment. This is to check to see whether the 100 units that you plan to test will be sufficient given your prior assumptions for the design.

Figure 22.9 Simulation Results



The simulated model fits shows that the 100 simulations of the model are all similar as they all fall close to the average shown by the red curve. The Y axis is on a log scale.

5. Double-click the Y axis and select **Linear** from the Type menu and click **OK**.

   The variability in the simulated fits around the 35 normal usage condition is now easier to view. You expect variability in your estimated of the normal usage condition. The simulation helps you evaluate if your ALT plan is acceptable. If there is too much

variability, you might need to increase the number of units on test or adjust the test conditions.

Save Design

Finally, complete the Example ATL design by making a test plan and saving it to a data table.

1. Click **Make Test Plan** to save the design to a data table. This table is a summary of the design.
2. Click **Make Table** to save the design to a data table. This table has 100 rows, one for each test run.

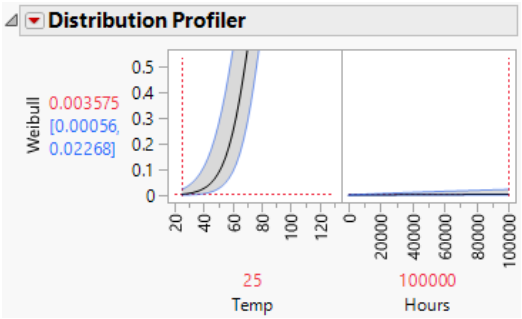
Example of Augmenting an Accelerated Life Test Design

This example shows how to use the Accelerated Life Test Design platform to augment an existing design. In this example, 150 capacitor units were tested across three temperatures (85^o, 105^o, and 125^o Celsius) for 1500 hours. The results are recorded in the Capacitor ALT.jmp sample data table. The resulting model is used to predict the fraction of the population that is failing at 100,000 hours at a normal usage temperature of 25^o Celsius.

Review Current Predictions

1. Select **Help > Sample Data Library** and open Design Experiment/Capacitor ALT.jmp.
2. Click the green triangle to run the **Fit Life by X** table script.
3. In the Distribution Profiler, found in the Comparisons outline on the Distribution tab, change the factor settings:
   - Click 105 above Temp and change it to 25.
   - Click 750.5 above Hours and change it to 100,000.

Figure 22.10 Distribution Profiler for Capacitor Model



Based on your current study, the predicted fraction of the capacitor population that fails at 25° at 100,000 hours is 0.003575, with a 95% confidence interval of 0.00056 to 0.02268. You want to estimate the fraction of failures more precisely. To decrease the width of the confidence interval, a measure of the precision of your estimate, you can augment your study with additional tests.

Augment the Design

You want to augment your design to obtain a more precise estimate of the predicted fraction of failures. Your original design used temperature settings of 85, 105, and 125. In your augmented design, you want to test at temperature values of 90, 110, and 125. Note that two of these settings are new. Augment the design with optimally selected runs:

1. Select **DOE** > **Special Purpose > Accelerated Life Test Design**.
2. Select **One accelerating factor**, select **Continuous Monitoring,** and click **Continue**.
3. Enter Temp for **Factor Name**.
4. Enter 5 for **Number of Levels**.

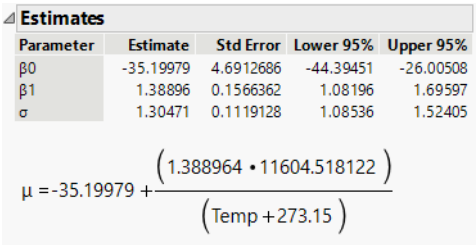
   Even though your augmented runs span only three levels (90, 110, and 125), you must also specify the levels used in the original experiment for a total of five levels. The Factor Transformation is set to Arrhenius Celsius by default.
5. Enter 25 for both **Low Usage Condition** and **High Usage Condition**.
6. Enter 85 for the **Low Test Condition** and 125 for the **High Test Condition**.
7. Click **Continue**.
8. Enter 85, 90, 105, 110, and 125 for the **Temp Level Values**.

   There are three levels from the original experiment (85, 105, and 125). The augmented design has two new levels (90 and 110) and one of the levels from the first experiment (125).

   Note: All levels must be listed.

9. Ensure that **Weibull** is selected as the **Distribution Choice**.
10. Under **Prior Mean**, select **Specify Intercept**. Enter your current acceleration model parameter estimates from the Fit Life by X Estimates outline, found in the Weibull Results outline on the Statistics tab.

Figure 22.11 Parameter Estimates and Fitted Model from Weibull Results Outline



Estimates

Parameter	Estimate	Std Error	Lower 95%	Upper 95%
β0	-35.19979	4.6912686	-44.39451	-26.00508
β1	1.38896	0.1566362	1.08196	1.69597
σ	1.30471	0.1119128	1.08536	1.52405

$$\mu = -35.19979 + \frac{\left(1.388964 \bullet 11604.518122\right)}{\left(\text{Temp} + 273.15\right)}$$

   - Enter -35.200 for **Intercept**.
   - Enter 1.389 for **Activation Energy (Temp)**.

     This is an estimate of the activation energy and is the coefficient of the inverse temperature, measured in degrees Kelvin, multiplied by Boltzmann's constant.
   - Enter 1.305 for **1/**β .

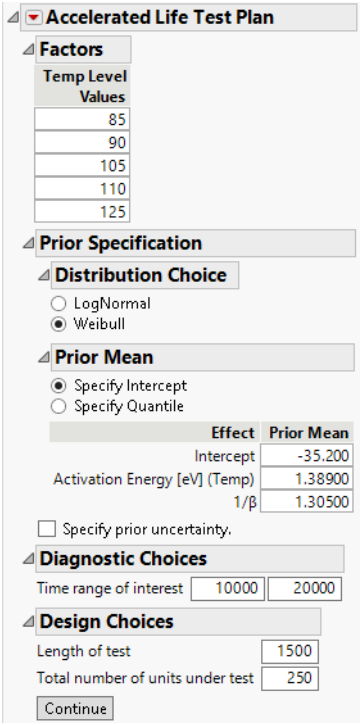
     For the Weibull distribution, JMP uses a parameterization that depends on a location parameter μ and scale parameter σ. In terms of the usual α and β parameterization, the scale parameter is σ = 1/β. See "Weibull".

   In the Accelerated Life Test Plan window, you can specify uncertainty about your prior means by selecting the **Specify prior uncertainty** option. In this example, you do not specify prior uncertainty. Your design is created assuming that the values specified in the Prior Means outline are the true parameter values.
11. Under Design Choices, enter 1500 for **Length of test**.

    The test is conducted over 1500 hours, which was the length of the original design.
12. Enter 250 for **Number of units under test**.

    The previous experiment tested 150 units, and the augmented experiment tests an additional 100 units, for a total of 250 units.

Figure 22.12 Completed Design Details Window



13. Click **Continue**.
14. Under **Candidate Runs** enter 50 as the minimum number of units for the temperatures of 85, 105, and125. These represent the runs already completed.
15. Click the Accelerated Life Test Plan red triangle and select **ALT Optimality Criterion** > **Make Failure Probability Optimal**.

    For more information about this criterion, see "Make Failure Probability Optimal".
16. Click **Make Design**.

    The optimal experimental design appears along with other results.

Figure 22.13 Optimal Design

Design

Temp	N Units	Expected Failures	All Censored Probability	Failure Probability
85	50	6.9	0.0	0.14
90	58	12.3	0.0	0.21
105	50	30.0	0.0	0.6
110	0	0.0	0.0	0.75
125	92	91.2	0.0	0.99

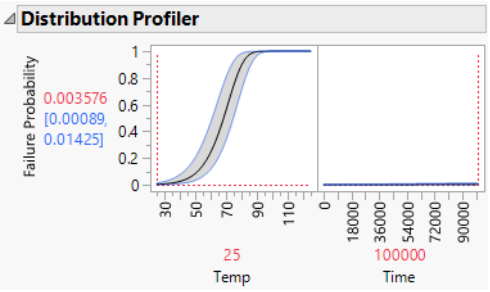
The optimal design is computed based on the levels of the test runs, the minimum number of units under test, the total number of units to be tested (this is the information in the Candidate Runs outline), and other information that you specified. The optimal design consists of testing the specified number of units at each temperature level:

- 50 units at 85°C. Since the previous experiment already tested 50 units at 85°C, no additional units are needed.
- 58 units at 90°C.
- 50 units at 105°C. Since the previous experiment tested 50 units at 105°C, no additional units are needed.
- 0 units at 110°C. This level is not needed.
- 92 units at 125°C. Since the previous experiment tested 50 units at 125°C, 42 additional units are needed.

Compare the Augmented Design to the Original Study

1. In the Distribution Profiler, enter the normal use condition of 25 for Temperature and 100,000 for Time. The estimate of the fraction of the population that is failing is 0.00357, with a 95% confidence interval of 0.00089 to 0.014525. This interval is narrower than the interval from the initial experiment, which was from 0.00056 to 0.02268 (Figure 22.10).

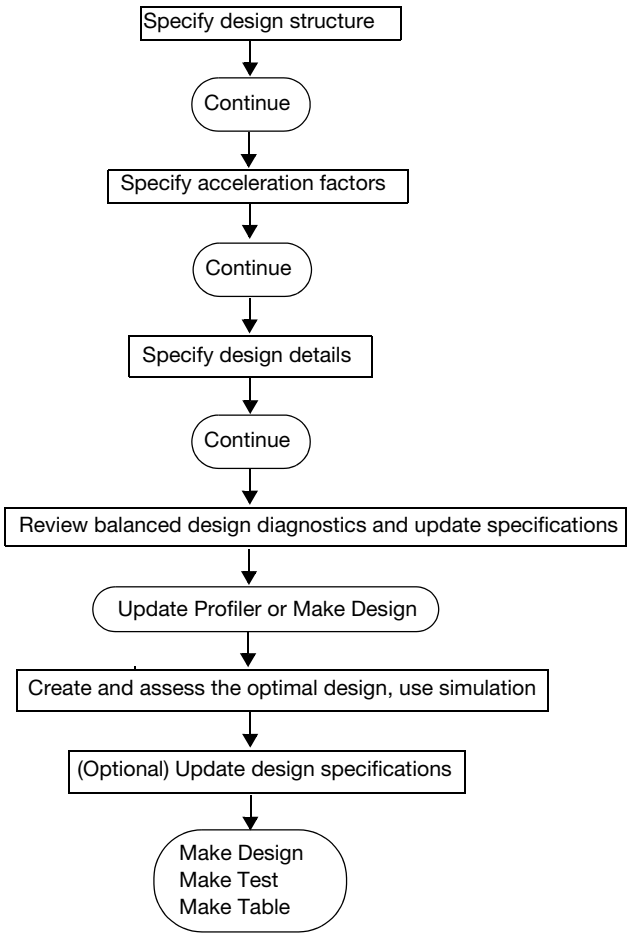
Figure 22.14 Distribution Profiler for Temp = 25 and Time = 100000



Build an Accelerated Life Test Design

Build an accelerated lift test (ALT) design by selecting **DOE > Special Purpose > Accelerated Life Test Design**. The Accelerated Life Test Plan is updated as you work through the design. The steps for building an accelerated life test design follow the flow in Figure 22.15.

Figure 22.15 Accelerated Life Test Plan Flow

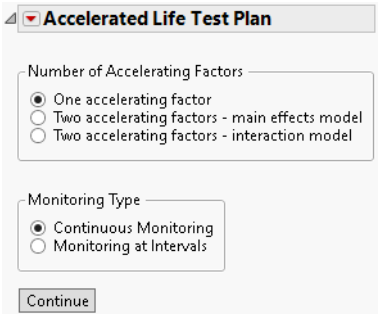


This section describes the outlines in the Accelerated Life Test Plan flow.

Specify the Design Structure

Use the Accelerated Life Test Plan outline to define the model structure and the type of inspection for your design.

Figure 22.16 Initial ALT Design Window



Number of accelerating Factors

One accelerating factor Select for a design with one factor.

Two accelerating factors - main effects model Select for a two factor design for a main effects model

Two accelerating factors - interaction model Select for a two factor design for a model that contains main effects and an interaction term.

Monitoring Type

Continuous Monitoring Select for a design with exact failure times recorded. Failure times beyond the length of the test are right censored.

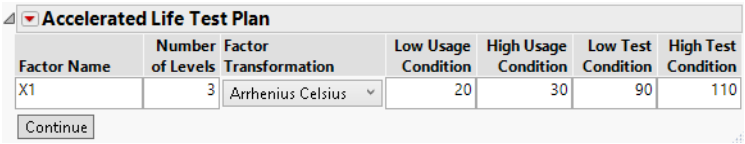
Monitoring at Intervals Select for a design with units inspected for failures at intervals. Failure times are interval censored. Enter the number of inspections, the time of the first inspection, and the time between inspections. For inspection intervals that are irregular, you can change the inspection times later in the Design Choices outline.

Accelerated Life Test Plan Specify details about the acceleration factor or factors.

Specify Acceleration Factors

Specify details about the acceleration factor or factors.

Figure 22.17 ALT Specification Window



Factor Name Enter a name for each acceleration factor.

Number of Levels For each acceleration factor, enter the number of proposed levels that you want to include in the experiment.

Factor Transformation For each acceleration factor, select a transformation function. This transformation describes the life-stress relationship, which is the manner in which the life distribution changes across stress levels. The transformations available are Arrhenius Celsius, Reciprocal, Log, Square Root, and Linear.

Low Usage Condition For each acceleration factor, enter a lower bound for its value in typical usage conditions.

High Usage Condition For each acceleration factor, enter an upper bound for its value in typical usage conditions.

Note: The Low Usage Condition and High Usage Condition values can be equal. Use equal values when the usage condition is a single value.

Low Test Condition For each acceleration factor, enter the low test condition.

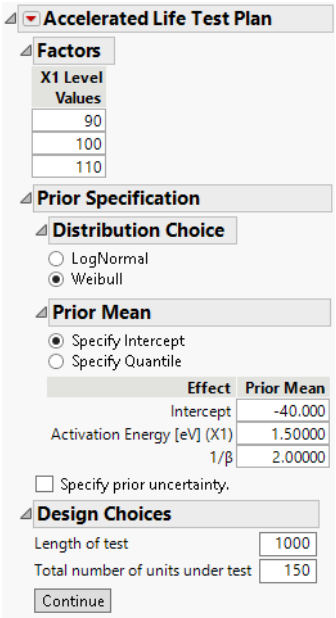
High Test Condition For each acceleration factor, enter the high test condition.

Note: The low and high test conditions define the endpoints of the acceleration factors in the design. The initial candidate uses equally spaced levels between the low and high test conditions. You can adjust the levels before constructing the design.

Specify Design Details

Specify the factor levels, details of the prior distribution, the time range and probability of interest, the length of the test, and the number of units to be tested.

Figure 22.18 Distribution Details



Factors Enter the levels for the acceleration factors. By default, the levels are evenly spaced between the low and high test conditions.

Distribution Choice Select a life distribution (LogNormal or Weibull) for each acceleration factor. See "Statistical Details for the ALT Design Platform".

Prior Mean Enter prior estimates of the acceleration model parameters. The prior estimates are hyperparameters in a Bayesian prior distribution. The Prior Mean values can be a best guess based on subject matter knowledge or they can be based on a previous study.

When there is only one acceleration factor, there is a choice to specify the intercept or the quantile.

Specify Intercept Enter the priors for the model parameters including the intercept.

Specify Quantile Enter the priors for the acceleration factor, the expected failure proportion at a given time, and value for the acceleration factor. This information is used to calculate the intercept.

Specify prior uncertainty Select to enter values for the standard error and correlations of the prior distribution for the acceleration model parameters. The standard errors reflect uncertainty relative to the prior estimates of the acceleration model parameters.

Note: When prior uncertainty measures are not specified, the design is created by treating the values entered under Prior Mean as the true parameter values. This design is close to optimal if the prior mean parameters are close to the true values. However, this design is not robust to misspecification of the parameter estimates. If you are unsure about your prior estimates, use the Prior Uncertainty specifications to reflect your uncertainty.

Diagnostic Choices The choices available depend on the design optimality criteria. The design optimality criteria is found under the Accelerated Lift Test Plan red triangle menu. For a D-optimal design there are no diagnostic choices.

Time range of interest (Available only for a failure probability optimal design.) Specify the time interval over which you want to estimate the fraction of the population that is failing. Enter the lower value in the left box and the upper value in the right box. If you are interested in a specific time point, enter that value in both boxes.

Probability of interest (Available only for a quantile estimate optimal design.) Specify the failure fraction for which you want an estimate of time. For example, if you want to estimate the time at which 10% of the units fail, then enter 0.10.

Design Choices Specify values relating to the length of the test, inspection intervals, and the number of units being tested.

Length of test (Available only for Continuous Monitoring.) Specify the length of time during which units are on test. When you make the design table, record each unit's failure time or whether it was right censored.

Inspection Times (Available only for Monitoring at Intervals.) Specify the times at which inspections are conducted. When you make the design table, these times are used to construct Start Time and End Time columns. The number of units failing in each interval is recorded.

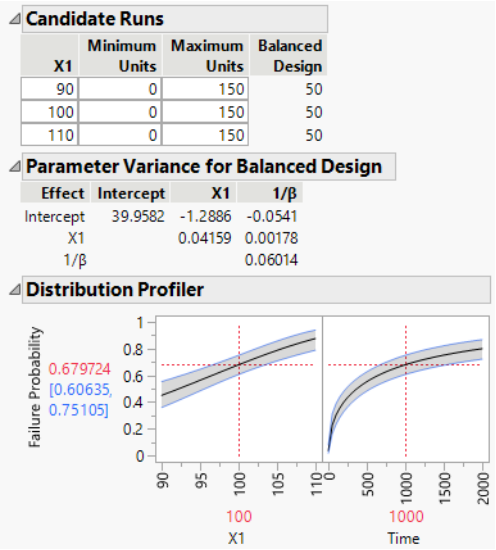
Number of units under test The number of units in the experiment.

- If you are designing an initial experiment, enter the number of units that you plan to test.
- If you are augmenting a previous experiment, enter the number of units tested in the previous experiment plus the number of units for the next experiment.

Review and Update Specifications for ALT Plans

After you specify the accelerated life test plan design details and click **Continue**, three new outlines are added to the window.

Figure 22.19 Additional Outline Nodes



Candidate Runs Enter the minimum and maximum number of units allowed at each level of the acceleration factors. If you are augmenting a previous experiment, for each setting, include the number of units already run at each level in the Minimum Units column.

Parameter Variance for Balanced Design Provides a matrix proportional to the covariance matrix for the estimates of the acceleration model parameters for the balanced design.

Denote the matrix of first partial derivatives of the model with respect to the parameters, θ, by $\mathbf{X}$. Denote the error variance by σ^2. Under general conditions, the least squares estimator of θ is asymptotically unbiased with asymptotic covariance matrix as follows:

$$Cov(\hat{\theta}) = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$$

The Parameter Variance for Balanced Design outline gives $(\mathbf{X}'\mathbf{X})^{-1}$, where derivatives are calculated numerically. The calculation assumes that the values specified in the Prior Mean outline are the true parameter values.

See "Nonlinear Models".

Distribution Profiler Enables you to explore failure probabilities, based on the balanced design, as you vary the acceleration factors and time. The probabilities are based on the assumption that the values specified in the Prior Mean are the true parameter values.

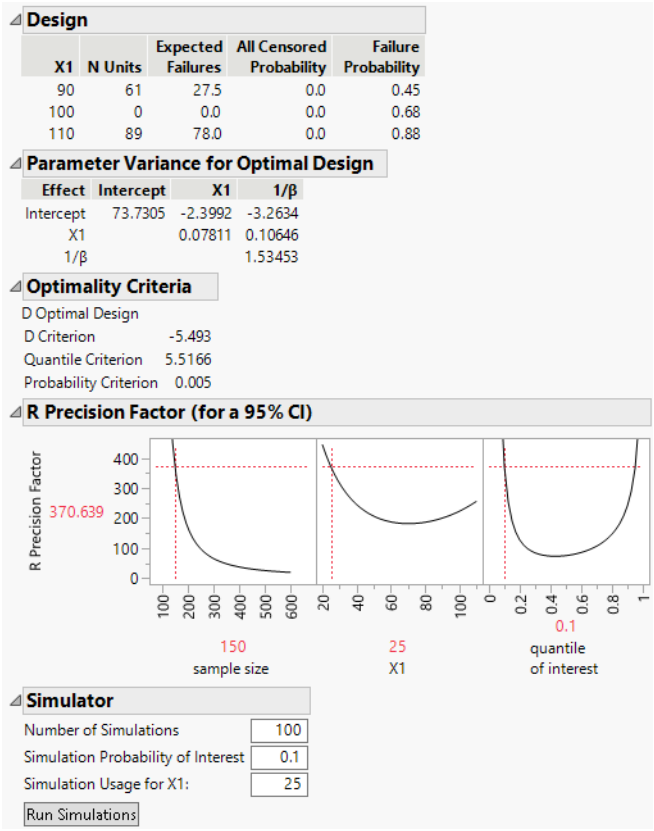
Tip: Click **Update Profiler** to update the profiler if changes are made to Distribution Choice, Prior Mean, or Design Choices.

Create and Assess the Optimal

Click **Make Design** to create the optimal ALT design and see results that address the quality of the design.

Note: By default, the optimal design is D-optimal. To change the optimality criterion click the Accelerated Life Test Plan red triangle and select **ALT Optimality Criterion**.

Figure 22.20 Design Outlines



Design Shows the number of units to be tested at each combination of acceleration levels. For a single factor, the first column gives the levels of the factor. For two factors, the first two columns give the levels of the factors.

N Units The number of units to be tested at each combination of acceleration levels.

Expected Failures Expected number of failures for the design setting. The expected number is computed using the prior model specification.

All Censored Probability Probability that none of the units tested at the combination of acceleration levels fail. The probability is computed using the prior model specification.

Failure Probability The expected probability of failure for the design settings. The failure probability is calculated as the expected failures divided by the number of units on test.

Note: The expected failures and censoring probabilities enable you to judge whether your prior specifications are reasonable.

Parameter Variance for Optimal Design Shows a matrix proportional to the covariance matrix for the estimates of the acceleration model parameters for the optimal design. The calculation assumes that the values specified in the Prior Mean outline are the true parameter values. See "Parameter Variance for Balanced Design".

Note: Compare the values in the Parameter Variance for Optimal Design matrix to those in the Parameter Variance for Balanced Design matrix to determine the extent to which the optimal design reduces the variance of estimates.

Optimality Criteria Values for three optimality criteria are provided:

D Criterion D-optimality of the design. See "Make D-Optimal Design".

Quantile Criterion Time I-optimality of the design. See "Make Quantile Estimate Optimal".

Probability Criterion Probability I-optimality of the design. See "Make Failure Probability Optimal".

R Precision Factor (for a 95% CI) A profiler to evaluate the impact of sample size, level of the factor(s), and the quantile of interest on the R Precision Factor. The R Precision Factor (for a 95% CI) is a measure of the precision of the 95% confidence interval for the quantile of interest.

Simulator Use the simulator to explore your accelerated model before collecting data.

Number of Simulations The number of simulation trials to run.

Simulation Probability of Interest The failure probability of interest.

Simulation Usage Values for the usage conditions of the acceleration factors.

After clicking Run Simulations, a table of summary statistics and a plot of model fits appear.

Create Design Tables

Use the buttons to make the accelerated lift test design, test plan, and table.

Make Design Updates the optimal design if any changes are made to the Distribution Choice, Prior Mean, Prior Variance Matrix, Design Choices, or Candidate Runs.

Make Test Plan Creates a data table where each row corresponds to a distinct design setting. The table shows the acceleration factor design settings and the number of units to test at those design settings.

Make Table Creates a table that you can use for recording your failure time data.

- For Continuous Monitoring, the table contains a row for each unit to be tested and the design settings for that unit. If a unit fails during the experiment, record the unit's failure time in the Failure Time column and a 0 in the Censored column. If a unit does not fails during the experiment, record the final inspection time in the Failure Time column and a 1 in the Censored column.
- For Monitoring at Intervals, the table contains a row for each design setting and time interval combination. The time intervals are defined by the Start Time and End Time columns, which are based on the Inspection Times entered in the Design Choices outline. For each setting and time interval, record the number of units that failed in the Number Failing column.

Accelerated Life Test Options

The Accelerated Life Test Plan red triangle menu contains the following options:

Simulate Responses Adds simulated responses to the table when you click **Make Table**. The simulated responses are created by taking random draws from the chosen distribution at the parameter values specified in the Prior Mean outline. If a simulated response exceeds the specified test length, the observation is censored at the test length value.

ALT Optimality Criterion

Make D-Optimal Design Creates a Bayesian D-optimal design if the number of Monte Carlo spheres is greater than 0. The optimality criterion is the expectation of the logarithm of the determinant of the information matrix with respect to the prior distribution. If the number of Monte Carlo spheres is 0, then the design is a locally D-optimal design. It follows that D-optimality focuses on precise estimates of the coefficients.

Make Quantile Estimate Optimal Default optimality criteria. Creates a design that minimizes the variance of a specified percentile at the specified usage conditions.

Make Failure Probability Optimal Creates a design that minimizes the variance of the estimate of failure probability over the time range of interest at the specified usage conditions.

Advanced Options

N Monte Carlo Spheres Enables you to set the number of nonzero radius values used in the integration. To find a nonlinear design that optimizes a given optimality criterion, JMP minimizes the integral of the log of the determinant of the Fisher information matrix with respect to the prior distribution of the parameters. Such an integral is calculated numerically. For more information about how the integration is performed, see "Nonlinear Design Options".

Tip: By default N Monte Carlo Spheres is set to four. Higher values result in better numerical accuracy but with more computation time.

Save Script to Script Window Creates the script for the design that you specified in the Accelerated Life Test Plan window and places it in an open script window.

Statistical Details for the ALT Design Platform

Failure Distributions for ALT Designs

In the accelerated life test design platform, you can select either a lognormal or Weibull failure distribution.

This section contains the parameterizations for the probability density function (pdf) and cumulative distribution function (cdf) for each distribution. For more information about the Weibull distribution, see *Reliability and Survival Methods*.

Lognormal

Lognormal distributions are used commonly for failure times when the range of the data is several powers of 10. This distribution is often conceptualized as the multiplicative product of many small positive independently and identically distributed random variables. This distribution is appropriate when the logarithms of the data values appear normally distributed. The probability distribution function is usually characterized by strong right-skewness.

The lognormal family is parameterized by a location parameter, μ, and a shape parameter, σ. The lognormal pdf and cdf are given as follows, where the logarithm is to the base *e*:

$$f(x;\mu,\sigma) = \frac{1}{x\sigma}\phi_{nor}\left[\frac{\log(x)-\mu}{\sigma}\right], \quad x > 0$$

$$F(x;\mu,\sigma) = \Phi_{nor}\left[\frac{\log(x)-\mu}{\sigma}\right],$$

The functions

$$\phi_{nor}(z) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z^2}{2}\right)$$

and

$$\Phi_{nor}(z) = \int_{-\infty}^{z}\phi_{nor}(w)dw$$

are the pdf and cdf, respectively, for the standard normal distribution (N(0,1)).

Weibull

The Weibull distribution can be used to model failure time data with either an increasing or a decreasing hazard rate. It is used frequently in reliability analysis because of its tremendous flexibility in modeling many different types of data, based on the values of the shape parameter.

The Weibull pdf and cdf are commonly represented as follows:

$$f(x;\alpha,\beta) = \frac{\beta}{\alpha^{\beta}}x^{(\beta-1)}\exp\left[-\left(\frac{x}{\alpha}\right)^{\beta}\right]; \qquad x > 0, \alpha > 0, \beta > 0$$

$$F(x;\alpha,\beta) = 1-\exp\left[-\left(\frac{x}{\alpha}\right)^{\beta}\right]$$

where α is a scale parameter, and β is a shape parameter. The Weibull distribution reduces to an exponential distribution when β = 1.

Weibull Parameterization

In JMP, the Weibull parameterization defines σ as the scale parameter and μ as the location parameter. These are related to the α and β Weibull parameterization as follows:

$$\alpha = \exp(\mu)$$

and

$$\beta = \frac{1}{\sigma}$$

With these parameters, the pdf and the cdf of the Weibull distribution are expressed as a log-transformed smallest extreme value distribution (SEV) using a location-scale parameterization with $\mu = \log(\alpha)$ and $\sigma = 1/\beta$:

$$f(x;\mu,\sigma) = \frac{1}{x\sigma}\phi_{sev}\left[\frac{\log(x)-\mu}{\sigma}\right], \quad x > 0, \sigma > 0$$

$$F(x;\mu,\sigma) = \Phi_{sev}\left[\frac{\log(x)-\mu}{\sigma}\right]$$

where

$$\phi_{sev}(z) = \exp[z - \exp(z)]$$

and

$$\Phi_{sev}(z) = 1 - \exp[-\exp(z)]$$

are the pdf and cdf, respectively, for the standardized smallest extreme value ($\mu = 0$, $\sigma = 1$) distribution.

R Precision Factor (for a 95% CI)

The R precision factor is a measure of the precision of the estimate of t_p, the p^{th} quantile of the failure time distribution. An approximate 95% confidence interval for $\log(t_p)$ is:

$$\log(\hat{t}_p) \pm z_{0.975}\sqrt{\widehat{Var}[\log(\hat{t}_p)]} = \log(\hat{t}_p) \pm \log(\hat{R})$$

resulting in a confidence interval for t_p;

$$[\hat{t}_p / \hat{R}, \hat{t}_p \hat{R}]$$

where

$$\hat{R} = \exp\left[z_{0.975}\sqrt{\widehat{Var}[\log(\hat{t}_p)]}\right]$$

To obtain R, use the large sample approximation of the variance of the $\log(t_p)$:

$$R = \exp z_{0.975}\sqrt{AVar[\log(\hat{t}_p)]}$$

Optimality Criterion for Accelerated Life Tests

Consider an accelerated life test (ALT) to be conducted across at most two acceleration factors x_1 and x_2 with J_1 and J_2 levels, respectively. The goal of test planning is to determine the number of samples to allocate to each combination of the factor levels.

The computation of the optimality criterion depends on the overall Fisher information matrix from the individual level-specific information matrices

$$I(\theta) = \sum_{i,j} n_{ij} I_{ij}(\theta)$$

as well as the inclusion of prior uncertainty S where appropriate.

D-optimality

The D-optimal design $\mathbf{D}_\mathrm{d}$ is the design that minimizes the following expression across all possible values of n_{jj}

$$\log\left|\int_\theta [\mathbf{S}^{-1} + \mathbf{I}(\theta)]^{-1}\pi(\theta)d\theta\right|$$

where $|\cdot|$ is the determinant operation and $\pi(\theta)$ is the prior distribution over the parameters. This distribution is usually a multivariate normal distribution. The log of the determinant is used for numerical stability.

Quantile Optimality

The quantile optimal design $\mathbf{D}_\mathrm{q}$ is the design that minimizes the expression:

$$\int_{\theta} [\mathbf{c}'\mathbf{I}(\theta)c]\pi(\theta)d\theta$$

where the vector **c** depends on the use condition.

For the single use condition

$$\mathbf{c}' = \{1, 0, 0, 0, z_p\}$$

where z_p is the p^{th} quantile of the failure distribution.

For the range of use condition let

$$\varepsilon_i = \frac{G(x_{1iUh}) - G(x_{1iUl})}{g(x_{iUh}) - g(x_{iUl})}$$

where $G(\cdot)$ is the antiderivative of $g(\cdot)$, x_{iUh} is the highest use level considered for the i^{th} factor, x_{iUl} is the lowest use level considered for the same factor. Then

$$\mathbf{c}' = \{1, \varepsilon_1, \varepsilon_2, \varepsilon_1, \varepsilon_2, z_p\}$$

where z_p is the p^{th} quantile of the failure distribution.

Failure Probability Optimality

The failure probability optimal design $\mathbf{D}_f$ is the design that minimizes the expression:

$$\int_{\theta} \left\{ \frac{\frac{1}{\sigma^2}\left[\frac{\phi(z_u(\theta))}{\Phi(z_u(\theta))}\right]^2 c'_{fp} I(\theta) c_{fp}}{-\log(\Phi(z_u(\theta)))} \right\} (\pi(\theta)d\theta)$$

where $z_U(\boldsymbol{\theta})$ is the standardized log-time at use conditions as defined in "Quantile Optimality". $\mathbf{c}_{fp}$ is similar to **c** as defined in "Quantile Optimality" except that the last entry is $z_U(\boldsymbol{\theta})$. The quantity in the numerator is based on the asymptotic variance of $\log\Phi(z_U(\boldsymbol{\theta}))$, where the log-transformation is used for numerical stability. The asymptotic variance is then standardized in terms of the original $\log\Phi(z_U(\boldsymbol{\theta}))$ for consistency and smaller is better. For a

range of use conditions, the integral is taken inside of $\log\Phi(\cdot)$ rather than over the asymptotic variance for ease of computation. It is still appropriate as an optimality metric by the property that the variance of a sum bounds the sum of variances and both the logarithmic function and cdf are monotone functions. As such, minimizing the above expression minimizes the true asymptotic variance of the failure probability by bounding it from above.

Chapter 23

Nonlinear Designs

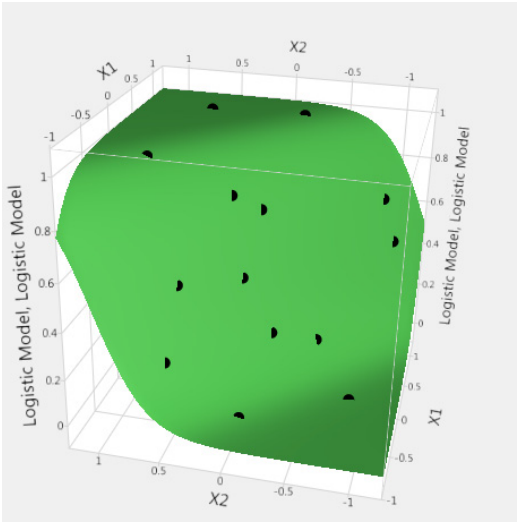
When the goal of your experiment is to fit a model that is nonlinear in the unknown parameters, use a nonlinear design to place design points in areas that are key to fitting the nonlinear model. Although you could use an orthogonal design that is optimal for a linear model, such designs, in general, do not place design points in locations that minimize the uncertainty (or maximize the precision) of the estimates of the fitted parameters.

The efficiency of a nonlinear design depends on the values of the unknown parameters. This creates a circular problem in that to find the best design, you need to know the parameters in advance. JMP uses a Bayesian approach to construct a nonlinear design that maximizes the average efficiency over specified ranges of the values of the parameter. To properly specify these ranges, you must have some insight about the system of interest.

Nonlinear designs offer these advantages compared to designs for linear models:

- Predictions using a well-chosen model are likely to be good over a wider range of factor settings.
- It is possible to model response surfaces with complex curvature and with asymptotic behavior.

Figure 23.1 Design Points for a Nonlinear Model



Contents

Overview of Nonlinear Designs 681
Examples of Nonlinear Designs 682
Create a Nonlinear Design with No Prior Data 682
Augment a Design Using Prior Data 687
Create a Design for a Binomial Response 691
Nonlinear Design Launch Window 694
Nonlinear Design Window 695
Factors 695
Parameters 696
Design Generation 696
Design 696
Make Table or Augment Table 697
Nonlinear Design Options 697
Statistical Details for Nonlinear Designs 698
Nonlinear Models 698
Radial-Spherical Integration of the Optimality Criterion 699
Finding the Optimal Design 699

Overview of Nonlinear Designs

Construct designs to fit models that are nonlinear in their parameters using the Nonlinear Design platform. You can construct optimal designs or optimally augment existing data for nonlinear models. Nonlinear designs based on information that is descriptive of the underlying process can yield more accurate estimates of model parameters and prediction of process behavior than is possible with standard designs for polynomial models. For background on nonlinear models, see "Nonlinear Models".

The efficiency of a design for a nonlinear model depends on the unknown values of the parameters that the design is intended to estimate. For this reason, JMP uses a Bayesian approach to construct designs that are efficient over a wide range of likely parameter values. You can specify a range of values for the unknown parameters and a distribution for the prior. The prior distribution choices include Uniform, Normal, Lognormal, and Exponential.

The Nonlinear Design platform uses a Bayesian approach, optimizing the design over a prior distribution of likely parameter values that you specify. The Bayesian D-optimality criterion is the expectation of the logarithm of the determinant of the information matrix with respect to a sample of parameter vectors that represents this prior probability distribution. The information matrix entries depend on the prediction variances at the design points. Little information is contributed by observations with low variance, where the response is almost certain. It follows that an optimal design places some design settings at high-variance points. See Gotwalt et al. (2009).

A principal of optimal design is that, over the feasible region of experimentation, the optimal design places points in locations with the highest variance of prediction. Though this may seem counter-intuitive, if an alternative design put points at other locations, the prediction variance at the design points of the optimal design would be even higher. For models that are linear in the parameters, the high-variance points tend to be at the vertices of the experimental region. But this is not necessarily true for models that are nonlinear in the parameters.

Note: Nonlinear designs are computed using a random starting design. For this reason, nonlinear designs that you obtain for identical specifications usually differ.

To use the Nonlinear Design platform, you must have an existing data table. That data table must contain the following:

- A column for the response.
- A column for each factor.
- A column that contains a formula showing the relationship between the factors and the response. This formula must include the unknown parameters.

Note: This is the same format as is required for a data table used in the Nonlinear platform for modeling.

Your table can come in one of two forms:

- It might be a template, containing only column information and no rows. See "Create a Nonlinear Design with No Prior Data".
- It might contain rows with predictor information. In this case, the predictor values are included in the nonlinear design. See "Augment a Design Using Prior Data".

Examples of Nonlinear Designs

- "Create a Nonlinear Design with No Prior Data"
- "Augment a Design Using Prior Data"
- "Create a Design for a Binomial Response"

Create a Nonlinear Design with No Prior Data

This example shows how to create a design when you have not yet collected data, but have a guess for the unknown parameters. In this example, you model the fractional yield (Observed Yield) of an intermediate product in a chemical reaction. The fractional yield is a function of reaction time and temperature. See Box and Draper (1987).

Create the Design

1. Select **Help > Sample Data Library** and open Design Experiment/Reaction Kinetics Start.jmp.

   Notice the following:

   – The data table contains no rows because no data have been collected.

   – The columns for the predictors, Reaction Temperature and Reaction Time, have the Coding, Design Role, and Factor Changes properties. To see these properties, click ✱ in the Columns panel. They tell JMP how to treat these predictors when constructing a design. For information about how to save these column properties, see "Column Properties".

   – The Observed Yield column will contain response data obtained by running the experiment.

   – The Yield Model column contains the formula that relates the predictors to the response, Observed Yield. Click ✙ in the Columns panel to see the formula. The formula is nonlinear in the parameters t1 and t3.

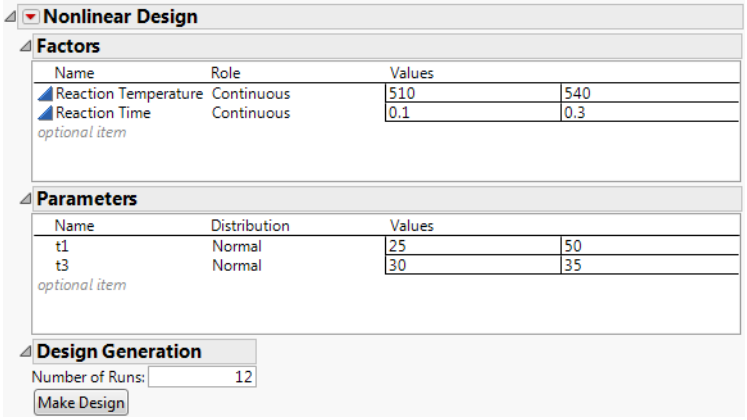
2. Select **DOE > Special Purpose > Nonlinear Design**.
3. Select Observed Yield and click **Y, Response**.
4. Select Yield Model and click **X, Predictor Formula**.
5. Click **OK**.

   In this example, the values 510 and 540 for Reaction Temperature and 0.1 and 0.3 for Reaction Time were specified using the Coding column property. Alternatively, you can specify a reasonable range of values directly in the Factors outline.
6. Change the values of the parameter t1 to 25 and 50, and t3 to 30 and 35.

   These new values represent a reasonable range of parameter values for the experimental situation. The default values were constructed based on the initial parameter values that were specified in the definition of the prediction formula. For information about constructing formulas, see *Using JMP*.

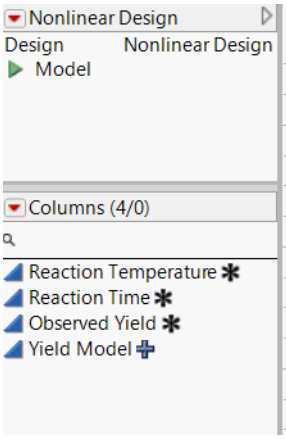
   Notice that the prior distribution, shown under Distribution, for each of t1 and t3 is set to Normal by default.
7. Change the number of runs to 12 in the Design Generation panel.

Figure 23.2 Completed Outlines for Reaction Kinetics Experiment



8. Click **Make Design**.
9. Click **Make Table**.

Figure 23.3 Design Table



	Reaction Temperature	Reaction Time	Observed Yield	Yield Model
1	540	0.3	•	58.093235
2	540	0.11	•	57.3016426
3	540	0.3	•	58.093235
4	530	0.2	•	54.9142081
5	510	0.1	•	11.2590804
6	540	0.3	•	58.093235
7	540	0.3	•	58.093235
8	510	0.1	•	11.2590804
9	517	0.1	•	17.8546023
10	530	0.21	•	55.9838913
11	540	0.11	•	57.3016426
12	540	0.3	•	58.093235

Your design should be similar to the one shown in Figure 23.3. The runs might be in a different order, and the values for Reaction Temperature and Reaction Time, and consequently those computed for Yield Model, can be slightly different. Notice that values appear in the Yield Model column because the column contains the formula for the model. Also notice that the table contains a **Model** script that you can use to fit a nonlinear model to your observations.

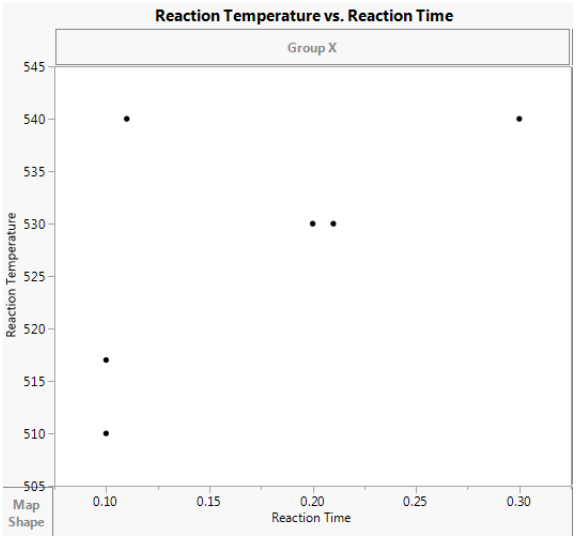
Now that you have created your design table, run your experiment, and record the responses in the Observed Yield column. The data table Reaction Kinetics.jmp, found in the Design Experiment folder contains observed results for the design.

Explore the Design

Before analyzing your results, construct a plot to see the design settings.

1. Select **Help > Sample Data Library** and open Design Experiment/Reaction Kinetics.jmp.
1. Select **Graph > Graph Builder**.
2. Drag and drop Reaction Temperature into the **Y** zone.
3. Drag and drop Reaction Time into the **X** zone.
4. Click the second icon above the graph to deselect the Smoother.

Figure 23.4 Design Settings

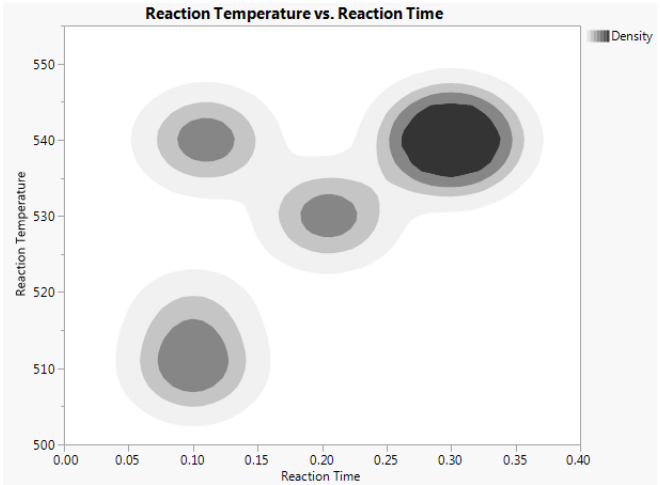


Notice that the points are located in three areas. There are no points at low temperature and high time (the lower right corner of the graph). Unlike orthogonal designs, nonlinear designs do not necessarily place design points at the corners of the design region. In this example, design points at low temperature and high time would be inefficient.

To see the density of design points in the remaining three corners, use the Contour tool.

5. Click to turn on the Contour tool.
6. Click **Done**.

Figure 23.5 Design Settings with Density Contours



Notice that there are comparatively few points at low time and high temperature. From the design table, you can see that there are only two such points. Because of the model and the parameter specifications, the optimal design places more design points at high time and high temperature.

Analyze the Results

Now that you visually explored your design, analyze your results.

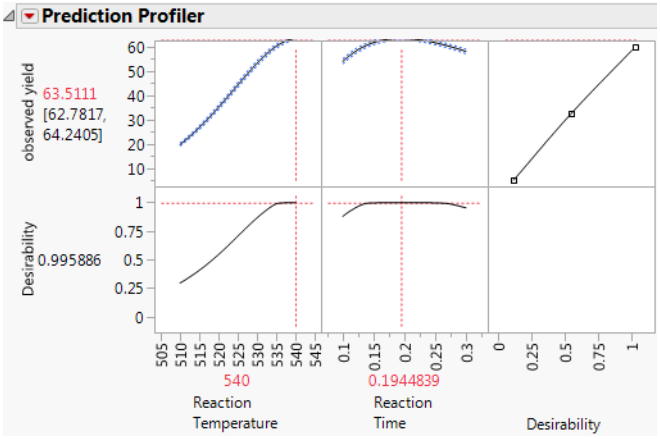
Note: Rather than conduct step 1 through step 4, you can run the **Model** script.

1. Select **Analyze > Specialized Modeling > Nonlinear**.
2. Select Observed Yield and click **Y, Response**.
3. Select Yield Model and click **X, Predictor Formula**.

   Notice that the model appears in the Options for fitting custom formulas panel.
4. Click **OK**.
5. Click **Go** in the Control Panel.

   The iterative search for a solution proceeds until one of the Stop Limit values is reached. Then, the Solution and Correlation of Estimates reports appear.
6. Click the Nonlinear Fit red triangle and select **Profilers > Profiler.**
7. To maximize the yield, click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 23.6 Time and Temperature Settings for Maximum Yield



The estimated maximum yield is approximately 63.5% at a reaction temperature of 540 degrees Kelvin and a reaction time of 0.1945 minutes.

Augment a Design Using Prior Data

This example shows how to create a nonlinear design when you have prior data. In this example, the data pertain to a chemical reaction. You want to model the rate of uptake (velocity) of available organic substrate as a function of the concentration of that substrate. See Meyers (1986). You have already run an experiment, but you want to leverage your results to obtain more precise estimates of the parameters.

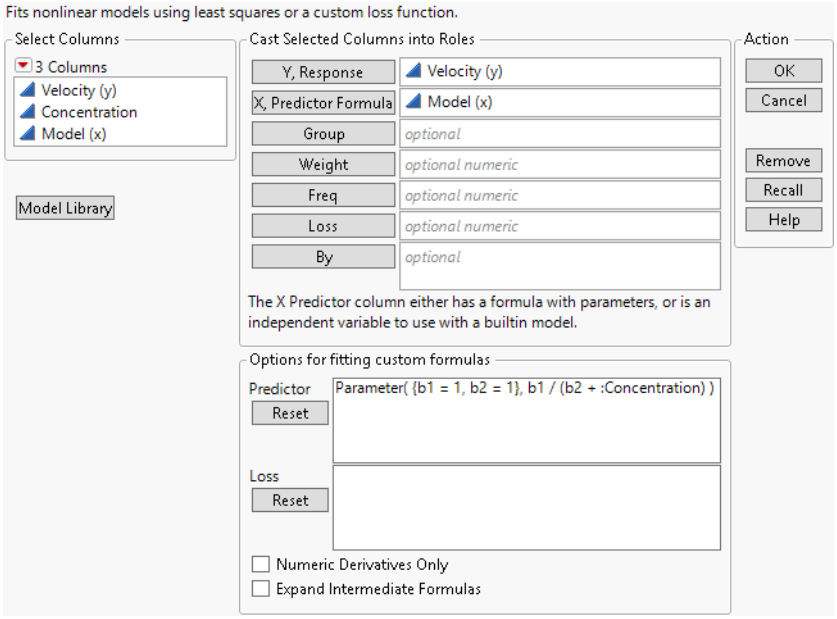
Obtain Prior Parameter Estimates

Use your existing experimental data to obtain better estimates of the parameter values.

1. Select **Help > Sample Data Library** and open Nonlinear Examples/Chemical Kinetics.jmp.
2. Click the plus sign next to Model (x) in the Columns panel. The formula editor opens.
3. The**Parameters** outline in the middle bottom of the formula editor shows the current values of the model parameters.The values (b1 = 1 and b2 = 1) are your initial estimates. They are used to compute the Model (x) values in the data table. For your next experiment, you want to replace these with better estimates.
4. Click **Cancel** to close the formula editor window.
5. Select **Analyze > Specialized Modeling > Nonlinear**.
6. Select Velocity (y) and click **Y, Response**.
7. Select Model (x) and click **X, Predictor Formula**.

   Notice that the formula given by Model (x) appears in the Options for fitting custom formulas panel.

Figure 23.7 Nonlinear Analysis Launch Window

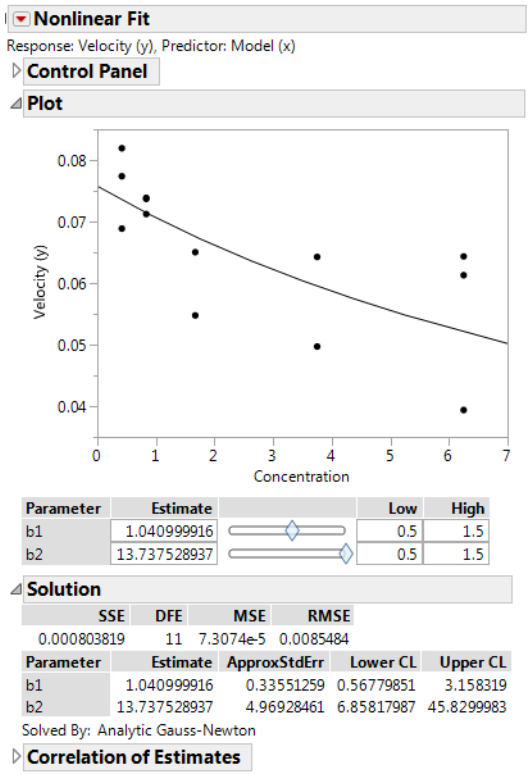


8. Click **OK**.
9. In the Control Panel, click **Go**.

   The iterative search for a solution proceeds until one of the Stop Limit values is reached. Then, the Solution and Correlation of Estimates reports appear. Also, an option appears in the Control Panel enabling you to add confidence limits to the Solution report.
10. In the Control Panel, click **Confidence Limits**.

    Confidence intervals for the parameters b1 and b2 appear in the Solution report.

Figure 23.8 Nonlinear Fit Results



Parameter	Estimate	Low	High
b1	1.040999916	0.5	1.5
b2	13.737528937	0.5	1.5

SSE	DFE	MSE	RMSE
0.000803819	11	7.3074e-5	0.0085484

Parameter	Estimate	ApproxStdErr	Lower CL	Upper CL
b1	1.040999916	0.33551259	0.56779851	3.158319
b2	13.737528937	4.96928461	6.85817987	45.8299983

The Lower CL and Upper CL values for b1 and b2 define ranges of values for these parameters. Next, use these intervals to define a range for the prior values in your augmented nonlinear design.

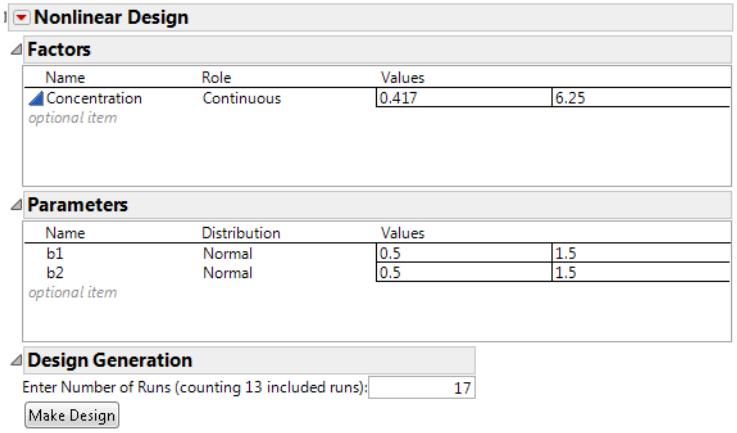
Note: Do not close the Nonlinear Fit report because these results are needed in the next steps.

Augment the Design

Now, create a design to estimate the nonlinear parameters more precisely.

1. With the Chemical Kinetics.jmp data table active, select **DOE > Special Purpose > Nonlinear Design**.
2. Select Velocity (y) and click **Y, Response**.
3. Select Model (x) and click **X, Predictor Formula**.
4. Click **OK**.

Figure 23.9 Nonlinear Design Outlines for Factors and Parameters

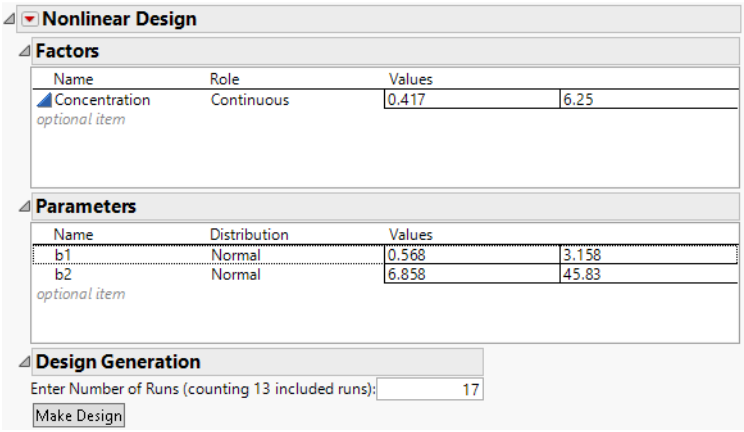


In the Chemical Kinetics.jmp data, the values for Concentration range from 0.417 to 6.25. Therefore, these values initially appear as the low and high values in the Factors outline. You want to change these values to encompass a broader interval.

5. Click 0.417 and type 0.1. Click 6.25 and type 7.
6. Leave the prior Distribution for each parameter set to Normal.

   The range of Values for the parameters reflects the uncertainty of your knowledge about them. You could specify a range that you think covers 95% of possible parameter values. The confidence limits from the Nonlinear Fit report shown in Figure 23.8 provide such a range. Replace the Values for the parameters in the Parameters outline with the confidence limits, rounding to three decimal places.
7. In the DOE Nonlinear Design window, enter these values into the Parameters for b1 and b2:
   - b1: 0.568 and 3.158
   - bb2: 6.858 and 45.830

Figure 23.10 Updated Values for Factor and Parameters



8. Enter 40 for the **Number of Runs** in the Design Generation panel.
9. Click **Make Design**.

   The Design outline opens, showing the Concentration and Velocity (y) values for the original 13 runs and new Concentration settings for the additional 27 runs.
10. Click **Make Table**.

    This creates a new JMP design table that contains the settings and results for the original 13-run design and settings for 27 new runs. Instead of creating a new data table, you can add the new runs to your existing data table by clicking Augment Table instead of Make Table.

The new runs reflect the broader interval of Concentration values and the range of values for b1 and b2 obtained from the original experiment, which are used to define the prior distribution. Both should lead to more precise estimates of b1 and b2.

Create a Design for a Binomial Response

In some applications, the measurement of interest is a pass/fail (binomial) measurement. In this example, you will construct a nonlinear design for a binomial response with one design factor. For more information on designs in nonlinear settings and the binomial case in particular, see Gotwalt et al. (2009).

You plan to model the probability of success for your binomial response as a function of a single factor x. using a generalized linear model with a logistic link function

$$Y(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

and the variance of Y as the weight.

This model is nonlinear in the unknown parameters β_0, and β_1. Use the nonlinear designer to plan an experimental design.Your goal is to model the effect of x on your binomial response Y.

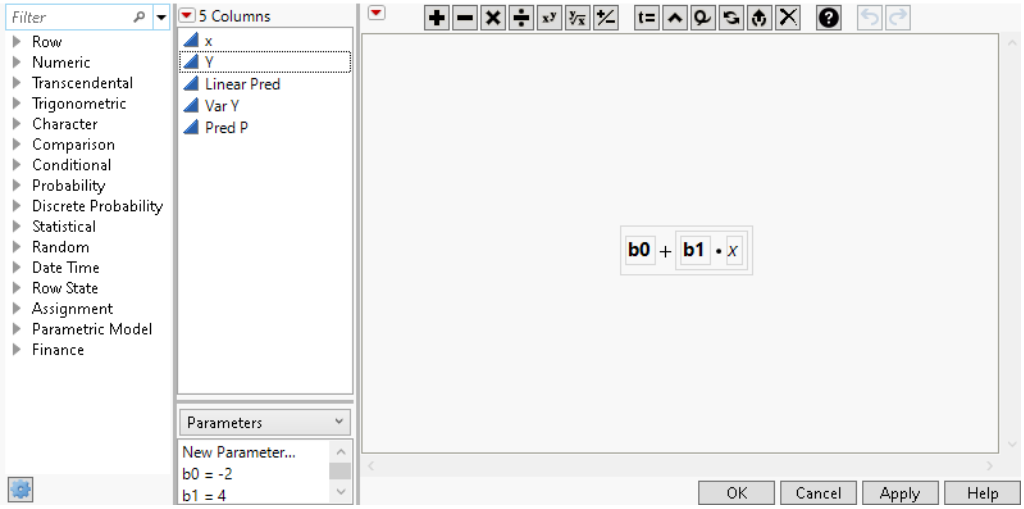
To generated the desired non-linear design you must have a data table containing columns for the predictor, a column containing a formula for the nonlinear response model that you are fitting, and a column with a formula for your weights.

Data Table for Design Construction

The One Factor Logistic Design.jmp data table, found in the Design Experiment folder, contains the following:

- Column x for the predictor. The Coding property defined for this column sets the low value to 0 and the high value to 1.
- Column Y, for the response. This is the column for collected responses (0/1) from your testing.
- Column Linear Pred that contains a formula for the linear portion of the link function. To view the formula, click the plus sign to the right of Linear Pred in the Columns panel. The model formula includes the initial estimates for the parameters **b0** and **b1**. Initial parameter values are set when you define a formula. These values are shown in the formula element panel in the lower center of the formula editor window.

Figure 23.11 Linear Predictor Formula with Initial Parameter Estimates

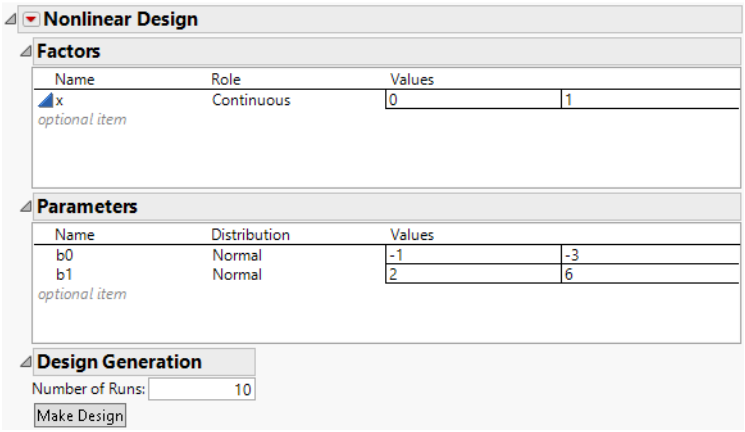


- Column Var Y that contains the formula for the variance of the response based on the assumed logistic model. This is *p(1-p)* where *p* is the logistic function. This column is used as the weight column.
- Column Pred P that contains the logistic link function.

Create the Design

1. Select **Help > Sample Data Library** and open Design Experiment/One Factor Logistic Design.jmp.
2. Select **DOE > Special Purpose > Nonlinear Design**.
3. Select Y and click **Y, Response**.
4. Select Linear Pred and click **X, Predictor Formula**.
5. Select Var Y and click **Weight**.
6. Click **OK**.

Figure 23.12 Nonlinear Design Window

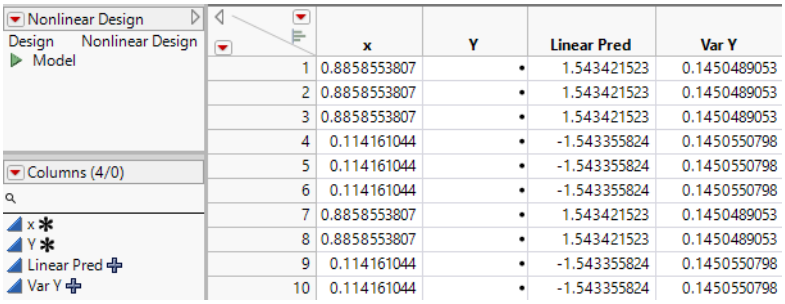


The Factors outline shows the factor x with the specified values of 0 and 1. The Parameters outline shows the two model parameters with each prior distribution set to a normal distribution. JMP computes default Values based on your initial settings for the parameter values. You can change the assumed distribution and ranges for the parameters in this outline. Leave all settings as they appear. The default number of runs is 10.

7. Click the Nonlinear Design red triangle and select **Advanced Options > N Monte Carlo Spheres**. Set the number of spheres to 0 to obtain a locally optimal design. Click **OK**.
8. Click **Make Design**.
9. Click **Augment Table**.

   This adds the 10 run design to One Factor Logistic Design.jmp (Figure 23.13). Your design table will be different because the optimization algorithm has a random component.

Figure 23.13 Augmentation of One Factor Logistic Design.jmp



	x	Y	Linear Pred	Var Y
1	0.8858553807	•	1.543421523	0.1450489053
2	0.8858553807	•	1.543421523	0.1450489053
3	0.8858553807	•	1.543421523	0.1450489053
4	0.114161044	•	-1.543355824	0.1450550798
5	0.114161044	•	-1.543355824	0.1450550798
6	0.114161044	•	-1.543355824	0.1450550798
7	0.8858553807	•	1.543421523	0.1450489053
8	0.8858553807	•	1.543421523	0.1450489053
9	0.114161044	•	-1.543355824	0.1450550798
10	0.114161044	•	-1.543355824	0.1450550798

The model saved to the data table is a nonlinear model to fit the “Linear Pred” used in the creation of the design. After you collect your data, use a generalized linear model (GLM) with a logit link function to model the effect of x on Y.

Nonlinear Design Launch Window

To use the Nonlinear Design platform, you must have an existing data table that contains the following:

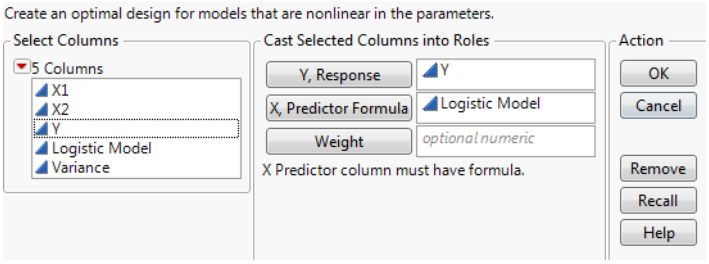
- A column for the response.
- A column for each factor.
- A column that contains a formula showing the relationship between the factors and the response. This formula must include the unknown parameters.

For information about formulas, see *Using JMP*.

The table can contain values for the predictors and response. If it does, the design that you construct augments the design that is implicit in the table. There can be no row containing missing predictor values.

With your starting data table active, select **DOE > Special Purpose > Nonlinear Design**.

Figure 23.14 Nonlinear Launch Window



Y, Response The numeric column for response values.

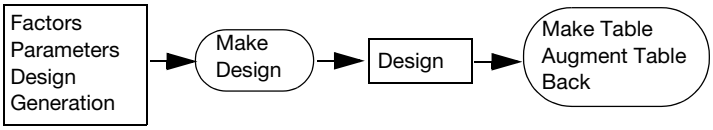
X, Predictor Formula The numeric column that contains the formula for the nonlinear model. This formula must contain parameters.

Weight (Optional) A numeric column that assigns weights to the observations.

Nonlinear Design Window

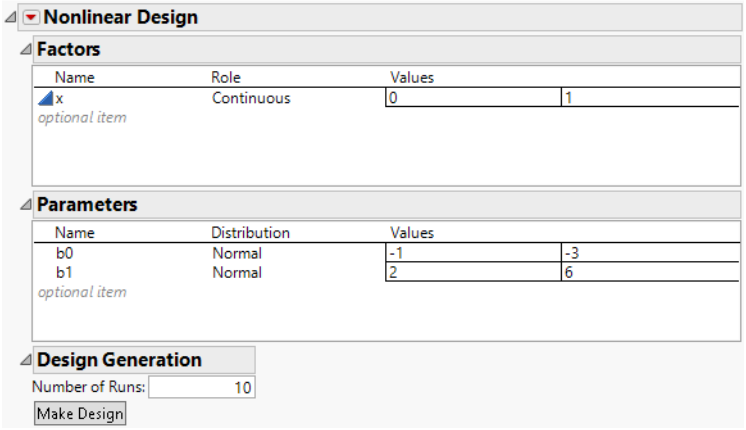
The Nonlinear Design window updates as you work through the design steps. The outlines, separated by buttons that update the outlines, follow the flow in Figure 23.15.

Figure 23.15 Nonlinear Design Flow



The initial design window shows the Factors, Parameters, and Design Generation outlines.

Figure 23.16 Initial Design Window



Factors

The column names used in the model formula are automatically inserted in the Name column of the Factors outline. Each factor's role is set to Continuous.

For each factor, the Values are initially set to -1 and 1. Or, if you have defined a value using the Coding column property, those values are used instead. You can change these values in the factors outline.

Parameters

The parameter names used in the model formula are automatically inserted in the Name column of the Parameters outline.

For each parameter, the Values are initially set to a symmetric interval around the initial value specified in the parameter definition. This interval is obtained by taking the initial value's distance to 0, and constructing an interval of this width around the initial value. These values are used in defining prior distributions for the model parameters.

Note: Adjust the Values for the parameters in conjunction with your choice of distribution to reflect your uncertainty about the model parameters.

Four families of prior distributions are listed under Distribution. The Values that you specify for the parameters determine which member of the family of prior distributions that is used. Denote the low value by *low* and the high value by *high*. Then the distributions are determined as follows:

- Uniform: The distribution is uniform on the interval (*low*, *high*).
- Normal, Lognormal, Exponential: The distribution is the one where *low* is the 0.025 quantile and where *high* is the 0.975 quantile.

Design Generation

JMP provides a suggested number of runs, determined as follows:

- If you are not augmenting a data table, the number of runs is four times the number of parameters plus two.
- If you are augmenting a data table, the number of runs is the number of runs in the data table plus two times the number of parameters.

Note: If you are augmenting a design, the number of runs that JMP suggests or that you specify includes those runs corresponding to observations in your data table. Adjust the number of runs appropriately.

Design

When you click Make Design, JMP constructs the design and adds a Design outline to the Nonlinear Design window. In the Design outline, you can review the factor level settings.

Make Table or Augment Table

The Make Table button creates a new design table. If your original table included existing runs, the new table also includes the existing runs. The **Augment Table** button adds the new runs to your existing table.

Nonlinear Design Options

The red triangle menu in the Nonlinear Design platform contains these options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Not available.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Load Factors Not available.

Save Constraints Not available.

Load Constraints Not available.

Simulate Responses Adds response values to the design table. Select this option before you click Make Table. Then, in the resulting design table, the response columns contain simulated values.

Note: To set a preference to always simulate responses, select **File > Preferences > Platforms > DOE** and select **Simulate Responses**.

Number of Starts Sets the number of times that a nonlinear design is created using the quadrature method. Among the designs created, the platform selects the design that maximizes the optimality criterion.

Advanced Options > Number of Monte Carlo Samples Sets the number of octahedra per sphere used in computing the optimality criterion. The default value is one octahedron. See "Radial-Spherical Integration of the Optimality Criterion".

Advanced Options > N Monte Carlo Spheres Sets the number of nonzero radius values used in computing the optimality criterion. The default is two. See "Radial-Spherical Integration of the Optimality Criterion".

Note: If N Monte Carlo Spheres (the number of radii) is set to zero, then only the center point is used in the calculations. This gives a local design that is optimal for the initial values of the parameters. For some situations, this is adequate.

Statistical Details for Nonlinear Designs

- "Nonlinear Models"
- "Radial-Spherical Integration of the Optimality Criterion"
- "Finding the Optimal Design"

Nonlinear Models

Denote the vector of n responses by $\mathbf{Y} = (Y_1, Y_2, ..., Y_n)'$. A nonlinear model is defined by the following properties:

- The Y_i are independent and identically distributed with an exponential family distribution.
- The expected value of each Y_i given a vector of predictor values $\mathbf{x}_i$ is a nonlinear function of parameters, θ. Denote this function as follows:

$$E(Y_i(\mathbf{x})) = f(\theta, \mathbf{x}_i)$$

- Each Y_i is expressed as follows:

$$Y_i = f(\theta, \mathbf{x}_i) + \varepsilon_i$$

- The vector of errors, $\varepsilon = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_n)'$ has mean $\mathbf{0}$ and covariance matrix $\sigma^2\mathbf{I}$, where $\mathbf{I}$ is the n x n identity matrix.

Denote the matrix of first partial derivatives of the function f with respect to the parameters θ by X. Under general conditions, the least squares estimator of θ is asymptotically unbiased, with asymptotic covariance matrix given as follows:

$$Cov(\hat{\theta}) = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$$

For the proof of this result, see Wu (1981) and Jennrich (1969).

Radial-Spherical Integration of the Optimality Criterion

The optimality criterion is the expectation of the logarithm of the determinant of the information matrix with respect to the prior distribution. Consequently, finding an optimal nonlinear design requires maximizing the integral of the log of the determinant of the Fisher information matrix with respect to the prior distribution of the parameters. This integral must be calculated numerically. The approach used in the Nonlinear Design platform is based on Gotwalt et al. (2009).

For normal distribution priors, the integral is reparameterized into a radial direction and a number of angular directions equal to the number of parameters minus one. The radial part of the integral is computed using Radau-Gauss-Laguerre quadrature with an evaluation at radius = 0. This is done by constructing a certain number of hyperoctahedra and randomly rotating each of them.

If the prior distribution is not normal, then the integral is reparameterized so that the new parameters have a normal distribution. Then the radial-spherical integration method is applied.

Note: If the prior distribution for the parameters does not lend itself to a solution and the process fails, a message is added to the window that the Fisher information matrix is singular in a region of the parameter space. When this occurs, consider changing the prior distribution or the ranges of the parameters.

Finding the Optimal Design

The method used to find an optimal design is similar to the coordinate exchange algorithm described in Meyer and Nachtsheim (1995). For more information about how the nonlinear optimal design is obtained, see Gotwalt et al. (2009). The general approach proceeds as follows:

- Random designs are tested until a nonsingular starting design is found.
- Iterations are conducted, where each iteration consists of a pass through all the runs.
- For each run, factors are optimized one at a time.
- The objective function is the Bayesian D-optimality criterion. This is the expectation of the logarithm of the determinant of the information matrix with respect to the prior distribution.
- Iterations terminate once the change in the objective function is small.

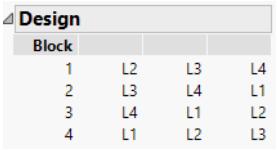
Chapter 24

Balanced Incomplete Block Designs

A Balanced Incomplete Block Design (BIBD) can be used when it is not possible to include all treatments or factor combinations in every block. Use BIBDs for testing a treatments in b blocks when only k treatments can be run in any one block (and $k < a$). For example, you might have a situation where there are limitations on the number of treatments that can be tested at the same time. The BIBD design is constructed such that each treatment and each pair of treatments occurs together in blocks an equal number of times. This makes the design *balanced*. Not including all treatments in every block makes the design *incomplete*.

Use the Balanced Incomplete Block Platform to construct BIBDs. In addition, you can use the platform to construct complete block designs where each treatment occurs in every block.

Figure 24.1 BIBD for 4 Treatments in 4 Blocks of Size 3



Design

Block			
1	L2	L3	L4
2	L3	L4	L1
3	L4	L1	L2
4	L1	L2	L3

Contents

Example of Balanced Incomplete Block Designs 703
Build a Balanced Incomplete Block Design 705
Specify Treatments, Blocks, and Make Design 705
Verify Design and Make Table 706
Balanced Incomplete Block Design Options 707
Statistical Details for Balanced Incomplete Block Designs 708

Example of Balanced Incomplete Block Designs

In this example, you want to test three new product formulations in your plant, and you want to block by day. There is a limit of two formulations that can be run per day. In addition, you need to finish your experiment in one work week. You need a design with at most five blocks.

1. Select **DOE > Special Purpose > Balanced Incomplete Block Design**.
2. Set the Treatment Size to 3.

   Tip: Treatment size can refer to the number of levels of a single factor or combinations of levels of two or more factors.

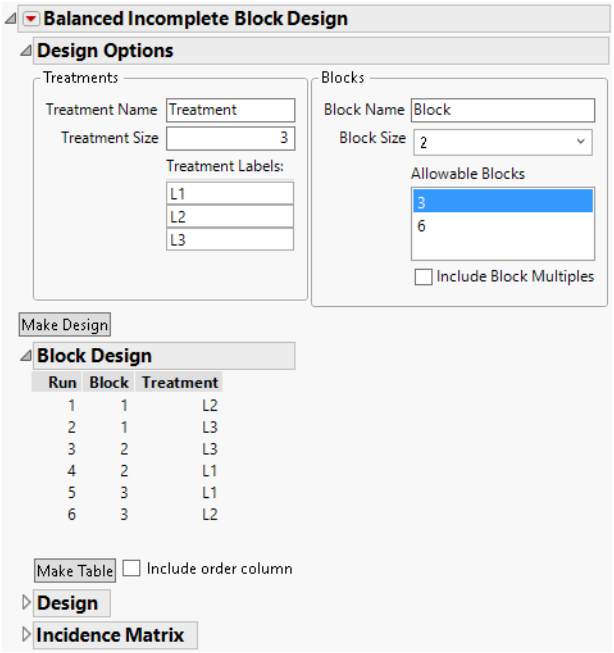
3. Select 2 for the **Block Size**.
4. Select 3 for **Allowable Blocks**.

   You can select 3 or 6 blocks. In this example, each block is a day and you can have no more than 5 days or 5 blocks. The number of runs is 6 because you have 3 blocks of size 2.

Note: You can enter a Treatment Name, Block Name, and Treatment Labels to customize your design table.

5. Click **Make Design**.

Figure 24.2 BIBD for 3 Treatments in 3 Blocks of Size 2

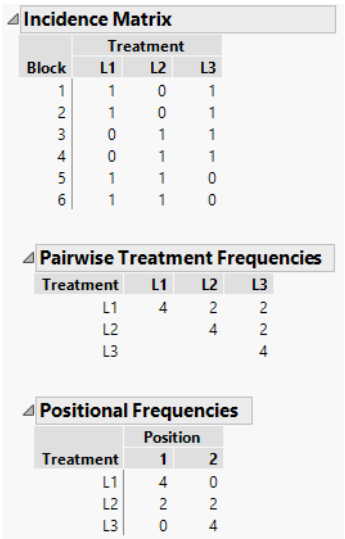


The design has six runs in three blocks with two treatments in each block as seen in the Block Design outline. For additional design details open the Design and Incidence Matrix outlines. See "Verify Design and Make Table".

Initially, you had to select a design with 3 or 6 blocks. You selected 3 blocks because of time constraints in your plant. However, suppose you were allowed to run your experiment for 6 days. You could then update the design to use 6 blocks.

6. Select 6 in the **Allowable Blocks** menu.
7. Click **Make Design** to update the design.
8. Open the **Incidence Matrix** outline.

Figure 24.3 BIBD Design Details for 3 Treatments in 6 Blocks of Size 2



Incidence Matrix

	Treatment		
Block	L1	L2	L3
1	1	0	1
2	1	0	1
3	0	1	1
4	0	1	1
5	1	1	0
6	1	1	0

Pairwise Treatment Frequencies

Treatment	L1	L2	L3
L1	4	2	2
L2		4	2
L3			4

Positional Frequencies

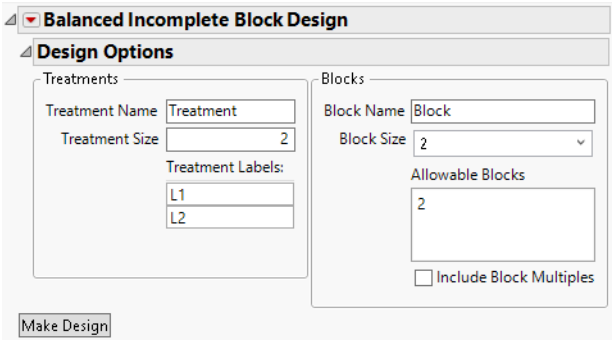
	Position	
Treatment	1	2
L1	4	0
L2	2	2
L3	0	4

Now the design has 6 blocks. The incidence matrix shows the treatments that are in each block. The Pairwise Treatment Frequencies shows that each treatment appears four times (diagonal entries) and that each treatment pair occurs twice in the design (off diagonal entries). The blocks each have two positions. The Positional Frequencies show that treatment L1 always occurs in position 1, L2 is split between the two positions, and L3 occurs in position 2.

Build a Balanced Incomplete Block Design

Build a Balanced Incomplete Block Design by selecting **DOE > Special Purpose > Balanced Incomplete Block Design**.

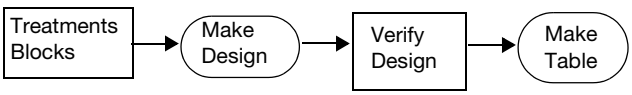
Figure 24.4 Balanced Incomplete Block Design Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Balanced Incomplete Block Design window updates as you work through the design. The steps to build a BIBD follow the flow in Figure 24.5.

Figure 24.5 Balanced Incomplete Block Design Flow



Specify Treatments, Blocks, and Make Design

The following options are in the Design Options outline:

Treatment Name The name of the treatment factor.

Treatment Size The number of treatments.

> **Tip:** Treatments can be factor-level combinations.

Treatment Labels Labels for the treatments. The default labeling is L1 to L*a* where *a* is the total number of treatments.

Block Name The name of the blocking factor.

Block Size The number of treatments that can be run in any one block.

Note: Values for Block Size are determined by the number of treatments. If a value is not in the list, it is because the BIBD platform cannot create a design with that block size for the given number of treatments.

Allowable Blocks Allowable number of blocks in the design based on the number of treatments and the block size.

Include Block Multiples Allows for the selection of a design with the number of blocks equal to a multiple (that you specify) of the initial allowable block sizes. The additional (multiple) blocks are cyclic permutations of the original design.

Specify Multiplier (Available when Include Block Multiples is selected.) Multiplier that you want to apply to the initial allowable block sizes.

Make Design Generates the design.

Verify Design and Make Table

Block Design Shows the BIBD in a traditional design layout.

Make Table Creates the BIBD data table.

Include order column Includes a column in the design table for the order of runs within each block.

Design Shows the BIBD in a block layout where each row represents a block and the treatments in that block.

Incidence Matrix Shows the treatments that occur in each block. Each row of the matrix corresponds to a block in the design with a 1 if the treatment is in the block and 0 otherwise.

Pairwise Treatment Frequencies Shows the number of times each treatment occurs in the same block with every other treatment on the off-diagonal entries. The diagonal entries are the total number of times each treatment occurs.

Positional Frequencies Shows the number of times each treatment occurs in each block position.

Balanced Incomplete Block Design Options

The red triangle menu in the Balanced Incomplete Block Design platform contains these options:

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Save Script to Script Window Creates the script for the design that you specified in the Balanced Incomplete Block window and places it in an open script window.

Statistical Details for Balanced Incomplete Block Designs

The total number of observations in a BIBD is determined by the following equation:

$$N = ar = bk$$

where

a is the number of treatments

r is the number of times each treatment occurs in the design

b is the number of blocks

k is the number of treatments in each block

The number of blocks in which each pair of treatments occurs is given by the following:

$$\lambda = \frac{r(k-1)}{a-1}$$

where λ must be an integer.

These conditions are necessary for a BIBD to exist, but not sufficient. The BIBD platform presents values for which it can generate a BIBD.

When the Include Block Multiples is selected, the original BIBD is copied the number of times specified. The additional (multiple) blocks are cyclic permutations of the original design. The positions of treatments in the copies differs from the original.

For more information about BIBDs, see SAS Institute Inc. (2013).

Chapter 25

MSA Designs

Create a Design for a Measurement Systems Analysis Experiment

Measurement systems analysis experiments, also known as *gage R&R studies*, are based on full factorial designs. The MSA platform generates a full factorial design for a measurement systems analysis (MSA) experiment and provides design diagnostic measures. For analysis of MSA experiments, see *Quality and Process Methods*.

Figure 25.1 MSA Factor Settings

Name	MSA Role	# of Levels	Variance	Randomize
Part	Part	5	1	Yes
Operator	Operator	4	1	Yes
Gauge	Gauge	2	1	Yes

Contents

Overview of MSA Designs 711
Example of MSA Design 711
Build an MSA Design 714
Responses 714
Factors 717
MSA Design 718
MSA Design Diagnostics 718
MSA Design Table 719
MSA Design Options 719
Statistical Details for the MSA Design Platform 720

Overview of MSA Designs

A measurement systems analysis (MSA) design is a full factorial design where the factors are random effects. The focus of the study is to estimate the variation in the response due to each factor. The MSA Design platform generates a full factorial design with diagnostic measures specific to an MSA study.

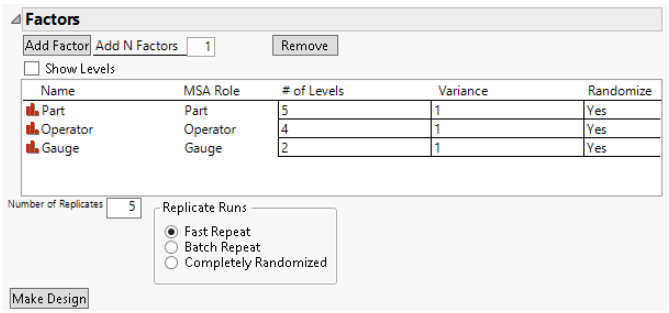
Example of MSA Design

In this example, you want to study the variability in your measurement systems due to instruments and operators. You have two scales in your lab and four operators who have time to participate in the study. You have a set of five standard parts to weigh and each operator makes six measurements per part per instrument. You do not have previous estimates of variability, so you design an experiment based on a variance of one, which results in diagnostics that are scaled to the true variability.

1. Select **DOE > Special Purpose > MSA Design**.
2. Click on the **Match Target** Goal for the response Y, and select **Match Target**.
3. Set the **# of Levels** for Part to 5.
4. Set **Add N Factors** to 2 and click **Add Factor**.
5. Rename X2 to Operator, click on None to set the MSA Role to Operator, and type 4 in the **# of Levels** column.
6. Rename X3 to Gauge, click on None to set the MSA Role to Gauge, and type 2 in the **# of Levels** column.
7. In the **Number of Replicates** box, type 5.

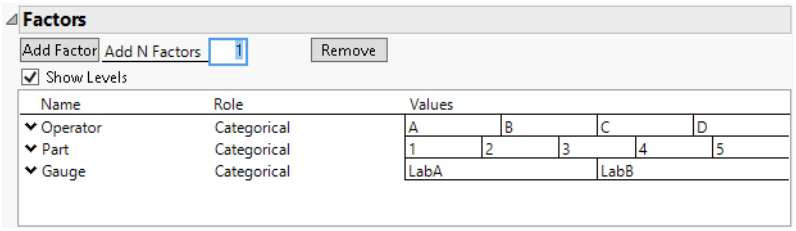
   Specifying 5 replicates indicates that each run of the design contains 6 measurements.
8. Leave the **Replicate Runs** set to Fast Repeat to run the replicates sequentially.

Figure 25.2 Factor Settings



9. Select **Show Levels** (check box in the Factors outline).
10. Edit the factor Values as shown in Figure 25.3.

Figure 25.3 Factor Values



Note: Setting the Random Seed in step 11 reproduces the design and diagnostics shown in this example.

11. (Optional) Click the MSA red triangle, select **Set Random Seed**, and type 123.
12. Click **Make Design**.

    Note: Click **Make Table** to generate a table for data collection. See "MSA Design Table" for analysis scripts that are included in the design table.

13. Scroll down to the Design Diagnostics outline and click the gray disclosure icon to open.

    The Design Diagnostics report contains settings for assumed variances that are then used to estimate confidence intervals, variance proportions, and EMP Monitoring Classifications. These estimates are used to evaluate the strength of your MSA study design. Explore different variance assumptions, or use the **Back** button to make adjustments to your design.
14. Adjust the expected Variance of your factors to explore the impact on your design diagnostics.
    - Set Operator Variance to 0.5.
    - Set Part Variance to 10.
    - Set Gauge Variance to 0.25.

Figure 25.4 MSA Design Diagnostics

Variance Estimates

Note: The diagnostics are based on a completely randomized crossed design and might not accurately reflect the manner in which this design is actually run.

Factor	Variance	95% CI Lower Bound	95% CI Upper Bound
Part	10	1.1816	27.021
Operator	0.5	0.0384	1.5745
Gauge	0.25	0.0002	1.2544
Part*Operator	0	0	0
Part*Gauge	0	0	0
Operator*Gauge	0	0	0
Part*Operator*Gauge	0	0	0
Error	0.5	0.4064	0.6016

Spec Limit Failure Probability

Response Name	Spec Limit Failure Probability	95% CI Lower Bound	95% CI Upper Bound
Y	0.1699	0.0008*	0.3962

Variance Proportions

Factor	Variance Proportion	95% CI Lower Bound	95% CI Upper Bound
Part	88.9%	48.1%	96.9%
Operator	4.4%	0.3%	25.3%
Gauge	2.2%	0.0%	16.9%
Part*Operator	0.0%	0.0%	0.0%
Part*Gauge	0.0%	0.0%	0.0%
Operator*Gauge	0.0%	0.0%	0.0%
Part*Operator*Gauge	0.0%	0.0%	0.0%
Error	4.4%	1.7%	21.3%

EMP Monitoring Classification

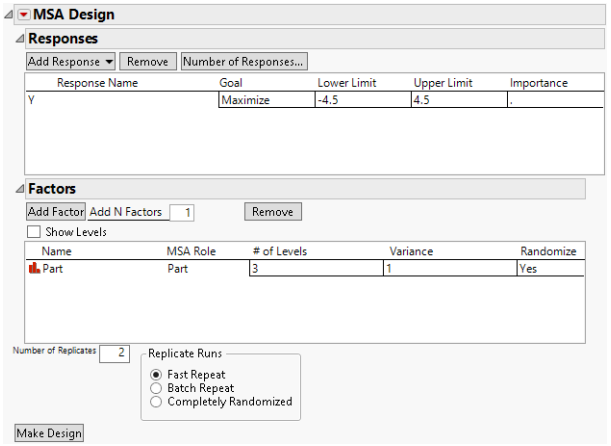
Monitoring Type	Probability of Classification
First Class	74.3%
Second Class	22.8%
Third Class	2.7%
Fourth Class	0.3%

All diagnostics are based on simulations that are computed from the sampling distributions of the appropriate mean squares. Based on the variance estimate assumptions, the study results in a 74% probability of a first class process. Note that the diagnostics are based on a completely randomized design. In this example, you selected the Fast Repeat to run the replicates sequentially. Often logistics of running the MSA requires the replicates to be run sequentially.

Build an MSA Design

Build an MSA design by selecting **DOE > Special Purpose > MSA Design**.

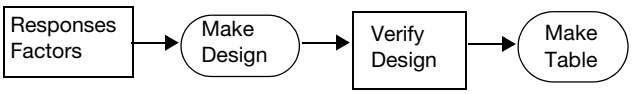
Figure 25.5 MSA Design Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The MSA design window updates as you work through the design steps. The steps to build an MSA design follow the flow in Figure 25.6.

Figure 25.6 MSA Design Flow

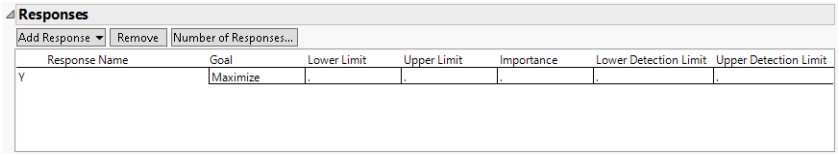


Responses

Use the Responses outline to specify one or more responses.

Tip: When you have completed the Responses outline, consider selecting **Save Responses** from the red triangle menu. This option saves the response names, goals, limits, and importance values in a data table that you can later reload in DOE platforms.

Figure 25.7 Responses Outline



Add Response Enters a single response with a goal type of Maximize, Match Target, Minimize, None, or Functional. If you select Match Target, enter limits for your target value. If you select Maximize or Minimize, entering limits is not required but can be useful if you intend to use desirability functions.

JMP PRO **Functional** (Available only in JMP Pro.) Adds multiple Y columns with FDE column properties to the design data table. These columns can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Remove Removes the selected responses.

Number of Responses Enters additional responses so that the number that you enter is the total number of responses. If you have entered a response other than the default Y, the Goal for each of the additional responses is the Goal associated with the last response entered. Otherwise, the Goal defaults to Match Target. Click the Goal type in the table to change it.

The Responses outline contains the following columns:

Response Name The name of the response. When added, a response is given a default name of Y, Y2, and so on. To change this name, double-click it and enter the desired name.

Goal, Lower Limit, Upper Limit The Goal tells JMP that you want to maximize or minimize your response, match a target, or that you have no goal for your response. JMP assigns a Response Limits column property, based on these specifications, to each response column in the design table. It uses this information to define a desirability function for each response. The Profiler and Contour Profiler use these desirability functions to find optimal factor settings. For further details, see the Profiler chapter in *Profilers* and "Response Limits".

- A Goal of Maximize indicates that the best value is the largest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Minimize indicates that the best value is the smallest possible. If there are natural lower or upper bounds, you can specify these as the Lower Limit or Upper Limit.
- A Goal of Match Target indicates that the best value is a specific target value. The default target value is assumed to be midway between the Lower Limit and Upper Limit.

– A Goal of None indicates that there is no goal in terms of optimization. No desirability function is constructed.

Note: If your target response is not midway between the Lower Limit and the Upper Limit, you can change the target after you generate your design table. In the data table, open the Column Info window for the response column (**Cols** > **Column Info**) and enter the desired target value.

Importance When you have several responses, the Importance values that you specify are used to compute an overall desirability function. These values are treated as weights for the responses. If there is only one response, then specifying the Importance is unnecessary because it is set to 1 by default.

JMP PRO **Detection Limits** The lower and upper detection limits are bounds beyond which the response cannot be measured. The detection limits are saved as a column property. You can use these limits to specify a censored response in the Generalized Regression platform. See the Generalized Regression chapter in *Fitting Linear Models*.

Editing the Responses Outline

In the Responses outline, note the following:

- Double-click a response to edit the response name.
- Click the goal to change it.
- Click on a limit or importance weight to change it.
- For multiple responses, you might want to enter values for the importance weights.

Response Limits Column Property

The Goal, Lower Limit, Upper Limit, and Importance that you specify when you enter a response are used in finding optimal factor settings. For each response, the information is saved in the generated design data table as a Response Limits column property. JMP uses this information to define the desirability function. The desirability function is used in the Prediction Profiler to find optimal factor settings. For further details about the Response Limits column property and examples of its use, see "Response Limits".

If you do not specify a Lower Limit and Upper Limit, JMP uses the range of the observed data for the response to define the limits for the desirability function. Specifying the Lower Limit and Upper Limit gives you control over the specification of the desirability function. For more details about the construction of the desirability function, see the Profiler chapter in *Profilers*.

Note: Detection limits are not available for MSA designs, however, they can be added via JSL.

Factors

The MSA Factors outline defaults to a 3-part study.

Note: The MSA designer requires at least one factor with a part role.

Add Factor Click to add one or more factors.

Remove Removes the selected factors.

Add N Factors Adds multiple factors. Enter the number of factors to add and click **Add Factor**.

Show Levels Select to show factor levels. When selected, the following options are available.

Name Factor name with a menu to add a level.

Role Fixed to a categorical factor. MSA factors are categorical by default.

Values Specifies the factor levels, which you can click to edit.

Name Specifies the name of the factor, which you can click to edit. Initially, three factors are in the factor list; Operator, Part, and Gauge. You can add or remove factors from this list.

MSA Role Specifies the MSA Role of the factor. Click to select from Operator, Part, Gauge, or None.

# of Levels Specifies the number of levels for a factor.

Variance Specifies a prior variance due to the factor. The default is one, which results in diagnostics that are scaled to the true variability.

Randomize Specifies to randomize the factor in the design table.

Number of Replicates Specifies the number of replicates between 0 and 10 to include in your design. The number of replicates is the number of measurements in addition to an initial measurement. The default is two replicates, which results in three measurements for each run. To specify a single measurement per run, set the number of replicates to zero.

Replicate Runs Specifies the run order for replicate runs in the experiment.

Fast Repeat replicates are run sequentially.

Batch Repeat The factor combinations are run in batches. Each replicate set makes up a new batch of runs. The order of runs is the same from batch to batch.

Completely Randomized the factor combinations are run in batches. Each replicate set makes up a new batch of runs. The order of runs is randomized within each batch.

Note: A Replicates column is included in the final design table. For the completely randomized run option, the Replicates column is hidden.

MSA Design

The Design outline shows the runs for the MSA design.

MSA Design Diagnostics

All diagnostics in the Design Diagnostics report are based on simulations that are computed from the sampling distributions of the appropriate mean squares. You can make changes to the variance estimate for each model factor. The design diagnostics update with changes to the variance estimates. This enables you to explore the ability of your design to measure your systems under different assumed models.

MSA Design Variance Estimates

The Variance Estimates report enables you to specify a variance estimate for each MSA model factor. Diagnostics are based on your variance estimates and include simulated confidence interval (CI) bounds. Simulated estimates and CI bounds are also provided for the probability of a response falling out of specifications. The specifications are defined by the lower and upper limits set for the response.

MSA Design Variance Proportions

The Variance Proportions report contains diagnostics for the estimated variance proportions based on the assumed variances.The report also includes simulated approximate CI bounds and measurement systems EMP classification probability. For more information about EMP classification see *Quality and Process Methods*.

MSA Design Simulation Results and Options

The Simulation report contains box plots of the distribution of the simulated variance proportions used to estimate the variance proportions CI bounds. The simulations are based on a large number of trials.

The Simulation Results red triangle menu contains the following options:

Show Statistics Opens summary statistics for each model factor.

Show Points Shows or hides the points in the box plots that fall beyond the whiskers.

Make Data Table Opens a data table of the simulation results.

MSA Design Table

Use the buttons to finish your MSA design construction.

Make Table Constructs the MSA design data table. In addition to the design, the table includes a run order column, a replicates column, and the following scripts for analysis:

EMP Measurement Systems Analysis Launches the MSA analysis platform. See "Measurement Systems Analysis".

Variability Chart Launches the Variability Gauge Chart platform. See "Variability Gauge Charts".

DOE Dialog Launches the MSA Design platform with the specifications used to generate the current design table.

Operator Worksheets Opens a data table for each operator to use as a worksheet for data collection.

Back Takes you back to make adjustments to the design specifications.

MSA Design Options

The MSA Design red triangle menu contains the following options:

Save Responses Saves the information in the Responses panel to a new data table. You can then quickly load the responses and their associated information into most DOE windows. This option is helpful if you anticipate reusing the responses.

Load Responses Loads responses that you saved using the Save Responses option.

Save Factors Saves the information in the Factors panel to a new data table. Each factor's column contains its levels. Other information is stored as column properties. You can then quickly load the factors and their associated information into most DOE windows.

Note: It is possible to create a factors table by entering data into an empty table, but remember to assign each column an appropriate Design Role. Do this by right-clicking on the column name in the data grid and selecting **Column Properties > Design Role**. In the Design Role area, select the appropriate role.

Load Factors Loads factors that you saved using the Save Factors option.

Save Constraints (Unavailable for some platforms) Saves factor constraints that you defined in the Define Factor Constraints or Linear Constraints outline into a data table, with a

column for each constraint. You can then quickly load the constraints into most DOE windows.

In the constraint table, the first rows contain the coefficients for each factor. The last row contains the inequality bound. Each constraint's column contains a column property called ConstraintState that identifies the constraint as a "less than" or a "greater than" constraint. See "ConstraintState".

Load Constraints (Unavailable for some platforms) Loads factor constraints that you saved using the Save Constraints option.

Set Random Seed Sets the random seed that JMP uses to control actions that have a random component. These actions include one or more of the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design, enter the random seed that generated the design before clicking **Make Design**.

Note: The random seed associated with a design is included in the DOE Dialog script that is saved to the design data table.

Statistical Details for the MSA Design Platform

The variance and variance proportion simulations are computed using simulated mean square terms. The mean square terms are simulated from chi-square distributions scaled by the associated expected mean square. The simulated variances are computed using method of moments estimators and are then used in computing the confidence bounds for the variance proportions.

For example, consider the variance for Part. The simulation computations follow these steps:

1. Simulate the mean square for Part, the mean squares for each two-way interaction involving Part, as well as any mean squares for higher order interactions involving Part.
2. Compute the variance using inclusion-exclusion:

   variance of Part = mean square of Part – mean square of two-way interactions + mean square of three-way interactions -…+/-mean square of highest order interaction
3. The simulated variance proportion for Part is then computed as Variance of Part/Total Variance.

Note: Only the distributions of the mean square terms are known directly. Specifically, Mean Square = Chi-Squared Distribution(df)*Expected Mean Square. The distributions of the variance and variance proportions are approximated by the simulated values. Because of this, the confidence bounds can differ each time a variance term is updated or each time the platform is run.

Chapter **26**

Group Orthogonal Supersaturated Designs

Create Group Orthogonal Supersaturated Designs

Group orthogonal supersaturated designs are a special case of screening designs. They are appropriate in early stage work when the number of factors to be investigated is larger than the number of feasible runs. A supersaturated design is a design with fewer observations than model parameters. That is, the design cannot estimate all main effects simultaneously. This makes detection of significant effects difficult. A group orthogonal supersaturated design is a special class of two-level supersaturated designs with properties that are desirable for model selection.

Contents

Overview of Group Orthogonal Supersaturated Designs 725
Example of Group Orthogonal Supersaturated Designs 725
Build a Group Orthogonal Supersaturated Design 727
Specify Factors, Structure, and Make Design 727
Verify Design and Make Table 728
Group Orthogonal Supersaturated Design Options 729
Statistical Details for Group Orthogonal Supersaturated Designs 730

Overview of Group Orthogonal Supersaturated Designs

Group orthogonal supersaturated designs (GOSSD) are constructed to have a specific structure to facilitate model selection. The design factors are partitioned into groups such that factors within a group are correlated with other factors within the same group yet factors are orthogonal to factors in other groups. One group consists of three or more "fake" factors. These fake factors are not assigned to design factors. They are used only in the analysis of the design. The fake factors provide an unbiased estimate of the variance assuming that second-order and higher-order effects are negligible.

Practical considerations for setting up a GOSSD include:

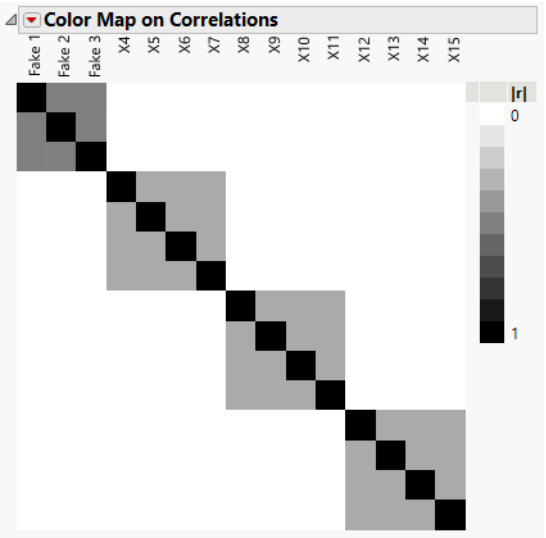
- More groups are better than fewer groups.
- Factors that are thought to be active should be placed in separate groups.
- Choose the levels of factors so that anticipated coefficients are positive.

Example of Group Orthogonal Supersaturated Designs

In this example, you want to design an experiment to study 12 factors in at most 12 runs. You want a design with at least 16 parameters. One parameter is needed for the intercept and at least three fake factors are desired.

1. Select **DOE > Special Purpose > Group Orthogonal Supersaturated > Group Orthogonal Supersaturated Design**.
2. Select **Number of Runs**.
3. Enter 12 for the **Number of Runs** and click away from the text box.
4. Select the second option in the Structure window with 16 parameters in 4 groups of size 4.
5. Click on the X1 factor name and change the name to Fake 1.
6. Repeat for X2 and X3 to name them as Fake 2 and Fake 3, respectively.
7. Click **Make Design.**

Figure 26.1 GOSSD Color Map on Correlations

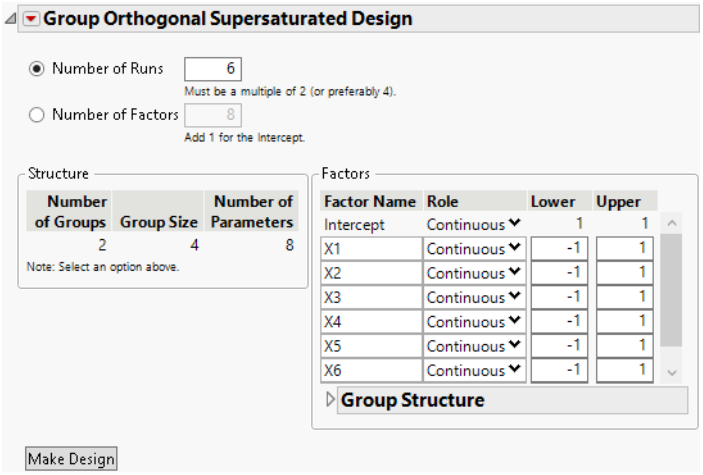


The structure of the design is evident in the color map. Factors within a group are correlated. Factors in different groups are not correlated, which indicates that they are orthogonal.

Build a Group Orthogonal Supersaturated Design

Build a Group Orthogonal Supersaturated Design by selecting **DOE > Special Purpose > Group Orthogonal Supersaturated > Group Orthogonal Supersaturated Design**.

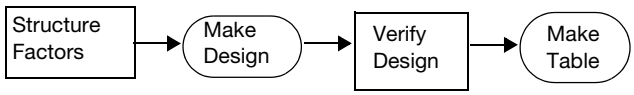
Figure 26.2 Group Orthogonal Supersaturated Design Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The GOSSD window updates as you work through the design . The steps to build a GOSSD follow the flow in Figure 26.3

Figure 26.3 Group Orthogonal Supersaturated Design Flow



Specify Factors, Structure, and Make Design

The following options are available in the Design Options outline.

Tip: Provide either the number of runs or the number of factors to define possible design structures.

Number of Runs Defines the number of design runs. The number needs to be a multiple of 2 and preferably a multiple of 4.

Number of Factors Defines the number of factors. Include a factor for the intercept and three to five fake factors. The number of factors must be a multiple of 8.

Structure Provides a list of possible structures for your design. The columns in this list are dependent on whether you set the number of runs or the number of factors.

Number of Groups The number of groups in the design. The minimum is 2.

Group Size The number of factors in each group.

Number of Runs (Available if you set the number of factors) The number of runs in the design.

Number of Parameters (Available if you set the number of runs.) The number of parameters in the design. The parameters include one intercept, your design factors, and the remaining parameters are assigned to fake factors.

Factors

Factor Name The name for the intercept is fixed. The remaining factors can be renamed.

Role Enables you to define a factor as continuous or categorical.

Lower Enables you to set the lower limit of each factor.

Upper Enables you to set the upper limit of each factor.

Group Structure Shows factors that appear in each group.

Swap Enables you to swap factors between groups.

Make Design Generates the design.

Verify Design and Make Table

Design Displays the design matrix.

Make Table Creates the GOSSD data table. The design table includes the following scripts:

Model (Executes only if the Response column is populated.) Runs the **DOE > Special Purpose > Group Orthogonal Supersaturated > Fit Group Orthogonal Supersaturated** platform**.**

Doe Dialog Re-creates the GOSSD Design window that you used to generate the design table.

GOSSD Structure Shows the structure of the design (number of runs, number of parameters, number of groups, group size).

Multivariate Runs the **Analyze > Multivariate** platform for the design factors.

Model Matrix Contains the design matrix.

DOE Simulate (Available when you simulate responses.) Launches the simulate window.

Design Evaluation

Correlation Matrix Shows the matrix of correlations between effects.

Color Map on Correlations Shows the absolute correlations between effects on a plot using an intensity scale. Includes a red triangle menu that contains an option to save the correlations to a data table.

Group Orthogonal Supersaturated Design Options

The red triangle menu in the Group Orthogonal Supersaturated Design platform contains the following options:

Simulate Responses Adds response values and a column containing a simulation formula to the design table. Select this option before you click Make Table.

When you click Make Table, the following occur:

- A set of simulated response values is added to each response column.
- For each response, a new column that contains a simulation model formula is added to the design table. The formula and values are based on the model that is specified in the design window.
- A Model window appears where you can set the values of coefficients for model effects and specify one of three distributions: Normal, Binomial, or Poisson.

Note: Not all distributions are available for all design types.

- A script called **DOE Simulate** is saved to the design table. This script reopens the Model window, enabling you to re-simulate values or to make changes to the simulated response distribution.

Make selections in the Model window to control the distribution of simulated response values. When you click Apply, a formula for the simulated response values is saved in a new column called <Y> Simulated, where Y is the name of the response. Clicking Apply again updates the formula and values in <Y> Simulated.

For additional details, see "Simulate Responses".

Note: JMP PRO You can use Simulate Responses to conduct simulation analyses using the JMP Pro Simulate feature. For more information and DOE examples, see the Simulate chapter in *Basic Analysis*.

Save Script to Script Window Creates the script for the design that you specified in the Group Orthogonal Supersaturated Design window and places it in an open script window.

Statistical Details for Group Orthogonal Supersaturated Designs

For statistical details of design construction see Jones, et.al. (2019).

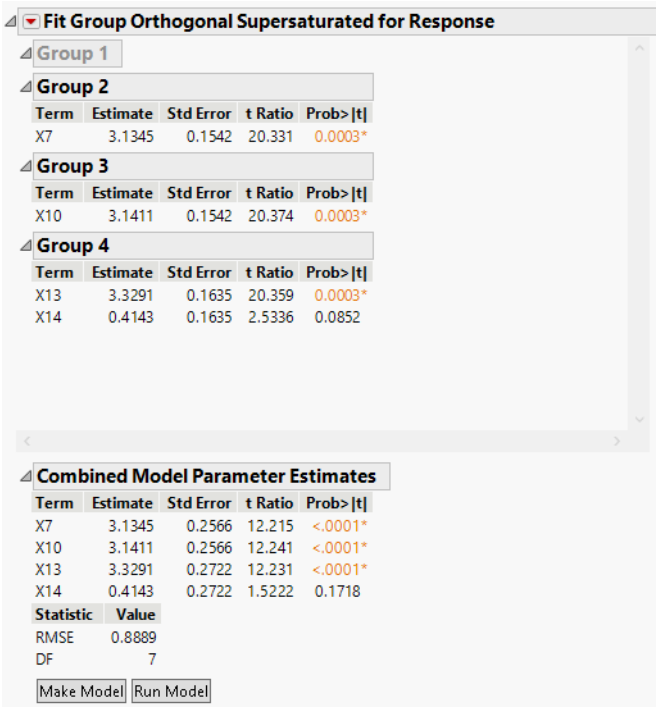
Chapter 27

The Fit Group Orthogonal Supersaturated Platform

Analyze Data from Group Orthogonal Supersaturated Designs

Use the Fit Group Orthogonal Supersaturated platform to analyze group orthogonal supersaturated designs. This analysis technique takes advantage of the group orthogonal structure of group orthogonal supersaturated designs.

Figure 27.1 Fit Group Orthogonal Supersaturated Results



Fit Group Orthogonal Supersaturated for Response

Group 1

Group 2

Term	Estimate	Std Error	t Ratio	Prob>|t|
X7	3.1345	0.1542	20.331	0.0003*

Group 3

Term	Estimate	Std Error	t Ratio	Prob>|t|
X10	3.1411	0.1542	20.374	0.0003*

Group 4

Term	Estimate	Std Error	t Ratio	Prob>|t|
X13	3.3291	0.1635	20.359	0.0003*
X14	0.4143	0.1635	2.5336	0.0852

Combined Model Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
X7	3.1345	0.2566	12.215	<.0001*
X10	3.1411	0.2566	12.241	<.0001*
X13	3.3291	0.2722	12.231	<.0001*
X14	0.4143	0.2722	1.5222	0.1718

Statistic	Value
RMSE	0.8889
DF	7

Make Model Run Model

Contents

Overview of the Fit Group Orthogonal Supersaturated Platform 733
Example of Fit Group Orthogonal Supersaturated 733
Launch the Fit Group Orthogonal Supersaturated Platform 736
The Fit Group Orthogonal Supersaturated Report 737
Fit Group Orthogonal Supersaturated Platform Options 738

Overview of the Fit Group Orthogonal Supersaturated Platform

A group orthogonal supersaturated design (GOSSD) is constructed to facilitate the identification of active effects. These designs group factors such that within a group, factors are correlated but between groups, factors are orthogonal. This structure leads to two analysis features:

- Fake, or dummy, factors assigned to the first group are used to estimate variance, assuming that the second- or higher-order effects are negligible.
- Between-group orthogonality enables a two-step model selection procedure.

The Fit Group Orthogonal Supersaturated Platform performs a two-step model selection procedure, enables you to view a regression plot of all factors, and then provides a model for the active effects with the option to examine this model with the Fit Model platform.

Example of Fit Group Orthogonal Supersaturated

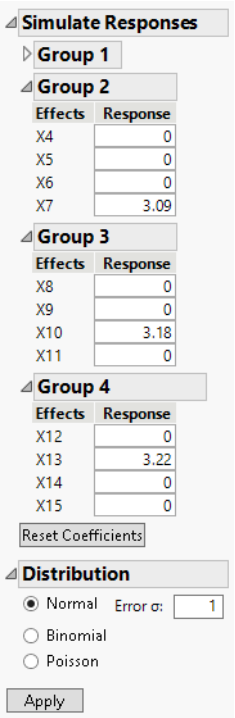
In this example, you want to design an experiment to study 12 factors in at most 12 runs. For a GOSSD, you need a design with at least 16 parameters. One parameter is needed for the intercept, at least three parameters are needed for the fake factors, and 12 parameters are needed for the design factors. Use simulated responses to fit a model.

Tip: You are not limited to three fake factors. More can be used.

Generate the Design

1. Select **DOE > Special Purpose > Group Orthogonal Supersaturated > Group Orthogonal Supersaturated Design**.
2. Select **Number of Runs**.
3. Enter 12 for the **Number of Runs** and click away from the text box.
4. Select the second option in the Structure section with 16 parameters in 4 groups of size 4.
5. Click on the X1 factor name and change the name to Fake 1.
6. Repeat for X2 and X3 to name them as Fake 2 and Fake 3, respectively.
7. Click the Group Orthogonal Supersaturated Design red triangle and select **Simulate Responses**.
8. Click **Make Design.**
9. Click **Make Table.**

Figure 27.2 Simulate Responses Window



The Simulate Responses window shows the settings that were used to generate the design table. Note that the outline box for Group 1 is closed. Group 1 contains the "fake" factors. These are all set to 0 as they are assumed to have no impact on the model. Your values might differ from those shown in Figure 27.2. This simulation has one active effect per group.

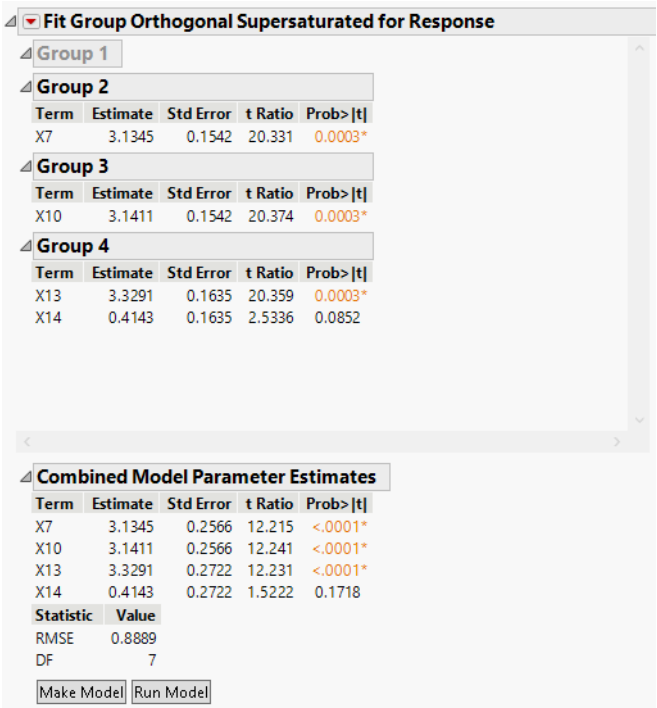
Tip: Use the Simulate Responses Window to enter new values for a different simulation.

Analyze the Simulated Response

1. Select **DOE > Special Purpose > Group Orthogonal Supersaturated > Fit Group Orthogonal Supersaturated**.
2. Select Response and click **Y**.
3. Select Fake 1 through X15 and click **X**.
4. Click **OK**.

Tip: The design table generated using the Group Orthogonal Supersaturated platform contains a script called Model. Run this script to run this analysis directly.

Figure 27.3 Fit Group Orthogonal Supersaturated Results



Fit Group Orthogonal Supersaturated for Response

Group 1

Group 2

Term	Estimate	Std Error	t Ratio	Prob>|t|
X7	3.1345	0.1542	20.331	0.0003*

Group 3

Term	Estimate	Std Error	t Ratio	Prob>|t|
X10	3.1411	0.1542	20.374	0.0003*

Group 4

Term	Estimate	Std Error	t Ratio	Prob>|t|
X13	3.3291	0.1635	20.359	0.0003*
X14	0.4143	0.1635	2.5336	0.0852

Combined Model Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
X7	3.1345	0.2566	12.215	<.0001*
X10	3.1411	0.2566	12.241	<.0001*
X13	3.3291	0.2722	12.231	<.0001*
X14	0.4143	0.2722	1.5222	0.1718

Statistic	Value
RMSE	0.8889
DF	7

Make Model Run Model

Your simulation set X7, X10, and X13 as factors with effects. The Fit GOSSD procedure identified those three factors as significant. In addition, case X14 was also identified. In practice, you would need to determine whether the size and significance of the effect warrants further attention.

Note: Your simulation results might differ from those shown in Figure 27.3 due to the generation of random results. You might see an alert that a group has one or more additional effects that might be active. This is a limitation of a supersaturated design; your findings might not be conclusive.

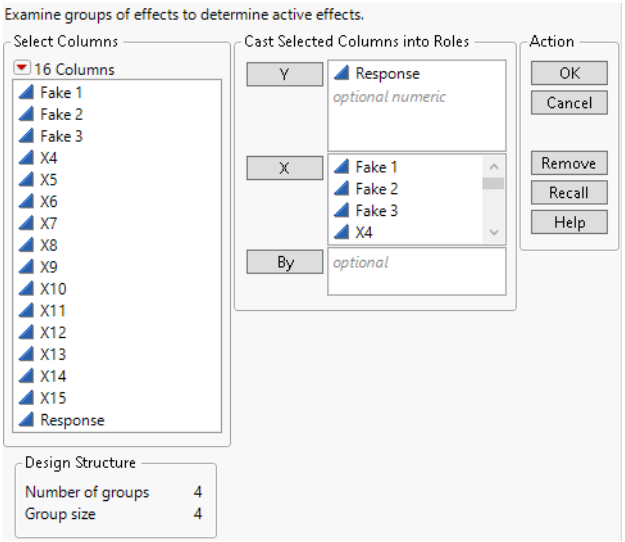
5. Click **Run Model** to run the identified model.

Launch the Fit Group Orthogonal Supersaturated Platform

Launch the Fit Group Orthogonal Supersaturated Design platform by selecting **DOE > Special Purpose > Group Orthogonal Supersaturated > Fit Group Orthogonal Supersaturated**.

Note: If you created your design in JMP using the Group Orthogonal Supersaturated platform, the design table contains a script called Model. Run this script to run the analysis directly.

Figure 27.4 Fit Group Orthogonal Supersaturated Launch Window



Y One or more numeric response variables.

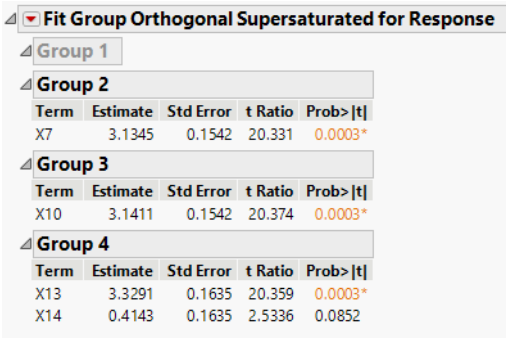
X Two-level continuous or categorical factors. Because the platform uses the unique structure of a GOSSD in performing the analysis, these factors must define a GOSSD.

For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Fit Group Orthogonal Supersaturated Report

The Fit Group Orthogonal Supersaturated Report has two sections. The first section reports the estimates of active effects within each group, fitting one group at a time. Only active groups are shown. This is Stage 1 of the analysis. In Stage 2 a combined model is constructed

Figure 27.5 Stage 1 Findings



Fit Group Orthogonal Supersaturated for Response

Group 1

Group 2

Term	Estimate	Std Error	t Ratio	Prob>|t|
X7	3.1345	0.1542	20.331	0.0003*

Group 3

Term	Estimate	Std Error	t Ratio	Prob>|t|
X10	3.1411	0.1542	20.374	0.0003*

Group 4

Term	Estimate	Std Error	t Ratio	Prob>|t|
X13	3.3291	0.1635	20.359	0.0003*
X14	0.4143	0.1635	2.5336	0.0852

Term Main effects that are identified as active.

Estimate Parameter estimate for a regression fit of Y on the main effect.

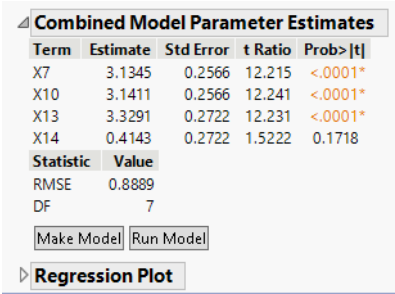
Std Error The standard error of the estimate, computed using the fake factors in Group 1.

t Ratio The Estimate divided by its Std Error.

Prob>|t| The *p*-value computed using the t Ratio and the degrees of freedom for error (DF).

The second section of the report contains the results of Stage 2 of the analysis. Stage 2 constructs the model used to evaluate the design results.

Figure 27.6 Stage 2 Results



Combined Model Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
X7	3.1345	0.2566	12.215	<.0001*
X10	3.1411	0.2566	12.241	<.0001*
X13	3.3291	0.2722	12.231	<.0001*
X14	0.4143	0.2722	1.5222	0.1718

Statistic	Value
RMSE	0.8889
DF	7

Make Model Run Model

Regression Plot

Term Main effects that are identified as active.

Estimate Parameter estimate for a regression fit of Y on the main effect.

Std Error The standard error of the estimates obtained by fitting the specified model and intercept using the full data set.

t Ratio The Estimate divided by its Std Error.

Prob>|t| The *p*-value computed using the t Ratio and the degrees of freedom for error (DF).

RMSE The square root of the mean square error.

DF The degrees of freedom associated with the error estimate used to construct the RMSE.

Fit Group Orthogonal Supersaturated Platform Options

The red triangle menu in the Fit Group Orthogonal Supersaturated Design platform contains the following options:

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Statistical Details for the Fit Group Orthogonal Supersaturated Platform

A two-stage strategy is used to fit group orthogonal supersaturated designs. The first stage identifies which groups contain active effects.

- Obtain an estimate of σ from the fake factors and then discard the fake factors from further analyses.
- Test for significance of each group of factors.

The second stage considers significance of individual factors in active groups. This stage uses a guided subsets procedure. See Jones and Nachtsheim (2016). For complete statistical details, see Jones, et al. (2019).

Chapter 28

Sample Size Explorers

Sample Size and Power Calculators

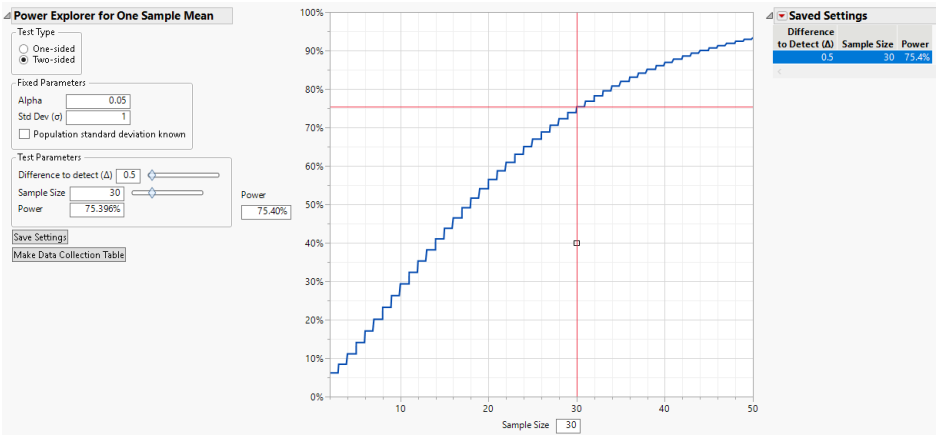
When planning experiments or studies, use the Sample Size Explorers to answer questions like these:

- How many units should I test?
- Will I be able to detect a difference in my treatment means?
- How many samples are needed to construct an interval with a specified width?
- How many units must I test to estimate failure time?

The Sample Size Explorers are a collection of interactive utilities that enable you to adjust study parameters and assumptions to explore different study designs. The tools are grouped by use:

- Power for hypothesis testing.
- Confidence intervals for interval estimation.
- Reliability demonstrations.

Figure 28.1 Sample Size Explorer for One Sample Mean



Contents

Overview of the Sample Size Explorers Platform 741
Example of Sample Size Explorers 741
Launch Sample Size Explorers 742
Power Explorers for Hypothesis Tests 743
Power for One Sample Mean 743
Power for One Sample Proportion 746
Power for One Sample Variance 748
Power for One Sample Equivalence 750
Power for Two Independent Sample Means 752
Power for Two Independent Sample Proportions 756
Power for Two Independent Sample Variances 759
Power for Two Independent Sample Equivalence 761
Power for ANOVA 764
Confidence Interval Calculators 765
Margin of Error for One Sample Mean 766
Margin of Error for One Sample Proportion 769
Margin of Error for One Sample Variance 770
Margin of Error for Two Independent Sample Means 772
Margin of Error for Two Independent Sample Proportions 774
Margin of Error for Two Independent Sample Variances 777
Reliability Demonstration Calculators 778
Parametric Reliability Demonstration 778
Nonparametric Reliability Demonstration 780
Saved Settings in the Sample Size Explorers 781

Overview of the Sample Size Explorers Platform

The Sample Size Explorers are used for sample size and power calculations. The Sample Size Explorers enable you to evaluate and compare the implications of your study assumptions on the number of observations (runs, experimental units, or samples) needed for your study. There are explorers for hypothesis testing, interval estimation, and reliability demonstrations.

For more information about power and sample size, see Barker (2011).

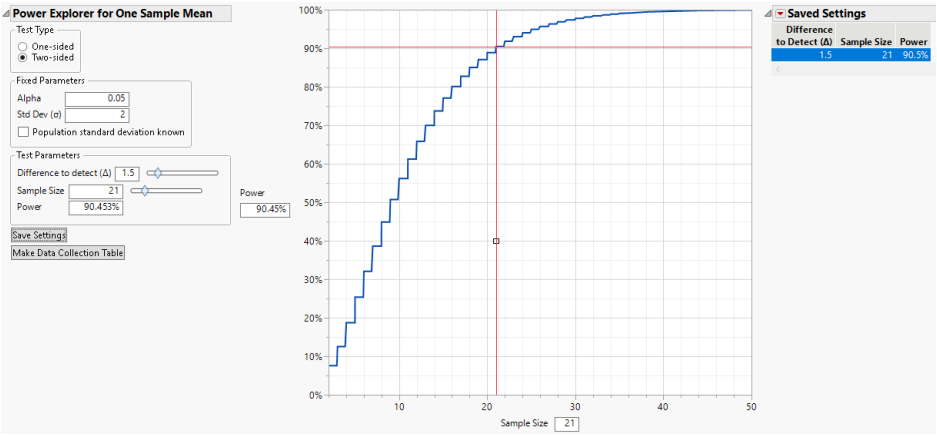
Example of Sample Size Explorers

You can use the Sample Size Explorers to help determine the number of units to test in a study of a new material. Suppose you are interested in demonstrating that the flammability of a new fabric being developed by your company has improved performance over current materials. Previous testing indicates that the standard deviation for time to burn of this fabric is 2 seconds.

Use the Power Explorer for One Sample Mean to calculate the number of fabric samples you need to test. You would like to design an experiment that has 90% power to detect a difference of 1.5 seconds at a significance level of $\alpha = 0.05$.

1. Select **DOE > Sample Size Explorers > Power > Power for One Sample Mean**.
2. Leave **Alpha** set to 0.05.
3. Enter 2 for **Std Dev**.
4. Enter 1.5 for **Difference to detect**.
5. Enter 0.9 for **Power**.
6. Click Save Settings to save these settings.
7. Use the sliders and or text boxes to explore other sample size and power possibilities for your study.

Figure 28.2 One Sample Mean Power Explorer



At a significance level of 0.05, 21 fabric samples are needed to have a 90.4% chance of detecting a significant difference of 1.5 seconds in the burn time.

Launch Sample Size Explorers

Launch Sample Size Explorers by selecting **DOE > Sample Size Explorers**. There are multiple calculators to select from.

For hypothesis testing, select **DOE > Sample Size Explorers > Power**:

- "Power for One Sample Mean"
- "Power for One Sample Proportion"
- "Power for One Sample Variance"
- "Power for One Sample Equivalence"
- "Power for Two Independent Sample Means"
- "Power for Two Independent Sample Means"
- "Power for Two Independent Sample Variances"
- "Power for Two Independent Sample Equivalence"
- "Power for ANOVA"

For interval estimation, select **DOE > Sample Size Explorers > Confidence Intervals**:

- "Margin of Error for One Sample Mean"
- "Margin of Error for One Sample Proportion"
- "Margin of Error for One Sample Variance"

- "Margin of Error for Two Independent Sample Means"
- "Margin of Error for Two Independent Sample Proportions"
- "Margin of Error for Two Independent Sample Variances"

For reliability demonstrations, select **DOE > Sample Size Explorers > Reliability**:

- "Parametric Reliability Demonstration"
- "Nonparametric Reliability Demonstration"

Power Explorers for Hypothesis Tests

Use the power explorers to calculate sample size for hypothesis tests. Explore the impact of sample size on the power of a hypothesis test to detect a specified difference where the difference could be from a constant (one-sample case) or between two groups (two-sample case). For the one-sample case, the hypothesis is about the single parameter (mean, proportion, or variance). For the two-sample case, the hypothesis is about comparing two groups. For more than two groups, you can use the Power Explorer for ANOVA, see "Power for ANOVA".

For hypothesis testing, select **DOE > Sample Size Explorers > Power**:

- "Power for One Sample Mean"
- "Power for One Sample Proportion"
- "Power for One Sample Variance"
- "Power for One Sample Equivalence"
- "Power for Two Independent Sample Means"
- "Power for Two Independent Sample Means"
- "Power for Two Independent Sample Variances"
- "Power for Two Independent Sample Equivalence"
- "Power for ANOVA"

Power for One Sample Mean

Use the Power Explorer for One Sample Mean to determine a sample size for a hypothesis test about one mean. Select **DOE > Sample Size Explorers > Power > Power for One Sample Mean**. Explore the trade-offs between variability assumptions, sample size, power, significance, and the hypothesized difference to detect. Sample size and power are associated with the following hypothesis test:

$$H_0: \mu = \mu_0$$

versus the two-sided alternative:

$$H_a: \mu \neq \mu_0$$

or versus a one-sided alternative:

$$H_a: \mu < \mu_0 \text{ or } H_a: \mu > \mu_0$$

where μ is the true mean and μ_0 is the null mean or reference value. The difference to detect is an amount, Δ, away from μ_0 that one considers important to detect. For the same significance level and power, a larger sample size is needed to detect a small difference than to detect a large difference. It is assumed that the population of interest is normally distributed with mean μ and standard deviation σ.

Power Explorer for One Sample Mean Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Test Type Specifies a one or two-sided hypothesis test.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the *significance level* of the test. The default alpha level is 0.05.

Std Dev (σ) Specifies the assumed population standard deviation.

Tip: Use a standard deviation of 1 to estimate the sample size needed to detect differences measured in standard deviation units.

Population standard deviation known Specifies use of calculations based on a known, rather than assumed, population standard deviation.

Test Parameters Parameters that are inter-related and update as you make changes.

Difference to detect (Δ) Specifies smallest difference between the true mean and the hypothesized or reference mean that you want to be able to declare statistically significant.

Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for One Sample Power

The one sample mean calculations are based on the traditional *t test* when σ is unknown. For the case when σ is known, the calculations use the *z* test. For the case when σ is unknown, the power is calculated according to the alternative hypothesis.

For a one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid \mu = \mu_0 + \delta) = 1 - T\left(t_{1-\alpha,\, n-1}; n-1, \frac{\delta}{\sigma/(\sqrt{n})}\right)$$

For a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid \mu = \mu_0 - \delta) = T\left(-t_{1-\alpha,\, n-1}; n-1, \frac{\delta}{\sigma/(\sqrt{n})}\right)$$

For a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid \mu = \mu_0 \pm \delta) = 1 - T\left(t_{1-\alpha/2,\, n-1}; n-1, \frac{\delta}{\sigma/(\sqrt{n})}\right) + T\left(-t_{1-\alpha/2,\, n-1}; n-1, \frac{\delta}{\sigma/(\sqrt{n})}\right)$$

where:

α is the significance level

n is the sample size

σ is the assumed population standard deviation

δ is the difference to detect

$t_{1-\alpha,n}$ is the $(1 - \alpha)^{th}$ quantile of the central t-distribution with ν degrees of freedom

$T(t; \nu, \lambda)$ is the cumulative distribution function of the non-central t distribution with ν degrees of freedom and non-centrality parameter λ.

When σ is known the *z* distribution is used in the above equations for the power calculations. Because closed-form solutions for δ and n do not exist, numerical routines are used to solve for them.

Power for One Sample Proportion

Use the Power Explorer for One Sample Proportion to determine a sample size for a hypothesis test about one proportion. Select **DOE > Sample Size Explorers > Power > Power for One Sample Proportion**. Explore the trade offs between sample size, power, significance, and the hypothesized difference to detect. Sample size and power are associated with the following hypothesis test:

$$H_0: p = p_0$$

versus the two-sided alternative:

$$H_a: p \neq p_0$$

or versus a one-sided alternative:

$$H_a: p < p_0 \text{ or } H_a: p > p_0$$

where p is the population proportion and p_0 is the null proportion.

Power Explorer for One Sample Proportion Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Test Type Specifies a one or two-sided hypothesis test.

Model Type

Exact Test Specifies calculations based on the Clopper-Pearson methodology.

Normal Approximation Specifies calculations based on the normal approximation methodology.

Tip: Since the binomial distribution is discrete, the actual test size can differ significantly from the stated Alpha level for small samples or proportions near 0 or 1. To guarantee an alpha level equal to or greater than your stated level, use the Exact test.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Test Parameters Parameters that are inter-related and update as you make changes.

Assumed proportion (p0) Specifies the proportion that you anticipate or assume for your study, the null hypothesis value.

Alternative proportion (pA) Specifies the proportion that you test against, the alternative hypothesis value.

Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the One Sample Proportion Explorer

For the exact test, the power is computed based on the form of the alternative hypothesis.

For one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid p = p_a > p_0) = \sum_{x = x_{1-\alpha, n, p_0}}^{n} \binom{n}{x} p_a^x (1-p_a)^{n-x}$$

for a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid p = p_a < p_0) = \sum_{x = 0}^{x = x_{\alpha, n, p_0} - 1} \binom{n}{x} p_a^x (1-p_a)^{n-x}$$

for a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid p = p_a) = \sum_{x = x_{1-\alpha/2, n, p_0}}^{n} \binom{n}{x} p_a^x (1-p_a)^{n-x} + \sum_{x = 0}^{x = x_{\alpha/2, n, p_0} - 1} \binom{n}{x} p_a^x (1-p_a)^{n-x}$$

where $x_{q,n,p}$ is the q^{th} quantile of a binomial distribution with n trials and probability p.

For the normal approximation, the power is computed based on the form of the alternative hypothesis.

For a one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid p = p_a > p_0) = 1 - \Phi\left(z_{1-\alpha}\sqrt{\left(\frac{p_0(1-p_0)}{p_a(1-p_a)}\right)} - \frac{(p_a - p_0)}{\sqrt{\frac{p_a(1-p_a)}{n}}} \right)$$

For a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid p = p_a < p_0) = \Phi\left(-z_{1-\alpha}\sqrt{\left(\frac{p_0(1-p_0)}{p_a(1-p_a)}\right)} - \frac{(p_a - p_0)}{\sqrt{\frac{p_a(1-p_a)}{n}}} \right)$$

For a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid p = p_a) =$$

$$1-\Phi\left(z_{1-\alpha/2}\sqrt{\left(\frac{p_0(1-p_0)}{p_a(1-p_a)}\right)} - \frac{(p_a - p_0)}{\sqrt{\frac{p_a(1-p_a)}{n}}} \right) + \Phi\left(-z_{1-\alpha/2}\sqrt{\left(\frac{p_0(1-p_0)}{p_a(1-p_a)}\right)} - \frac{(p_a - p_0)}{\sqrt{\frac{p_a(1-p_a)}{n}}} \right)$$

Power for One Sample Variance

Use the Power Explorer for One Sample Variance to determine a sample size for a hypothesis test about one variance. Select **DOE > Sample Size Explorers > Power > Power for One Sample Variance**. Explore the trade offs between sample size, power, significance, and the hypothesized difference to detect (defined as a ratio between the null and alternative hypothesis values). Sample size and power are associated with the following hypothesis test:

$$H_0: \sigma = \sigma_0$$

versus the two-sided alternative:

$$H_a: \sigma \neq \sigma_0$$

or versus a one-sided alternative:

$$H_a: \sigma < \sigma_0 \text{ or } H_a: \sigma > \sigma_0$$

where σ is the true variance and σ_0 is the null variance or reference value. The difference to detect is an amount away from σ_0 that one considers as important to detect based on a set of samples. This difference is expressed as the ratio of σ_0/σ_a, or the ratio of your null variance and your assumed variance under the alternative hypothesis. For the same significance level and power, a larger sample size is needed to detect a small difference in variances than to detect a large difference. It is assumed that the population of interest is normally distributed with mean μ and standard deviation σ.

Power Explorer for One Sample Variance Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Test Type Specifies a one or two-sided hypothesis test.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Test Parameters Parameters that are inter-related and update as you make changes.

Ratio of variances (Null/Alternative) Specifies the ratio of the variance under the null hypothesis (reference variance) to the variance under the alternative hypothesis (expected variance).

Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the One Sample Variance Explorer

The power calculations for testing the variance of one sample group is based on the χ^2 test. Calculations are based on the form of the alternative hypothesis.

For a one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid \sigma = \sigma_0/\rho, \rho < 1) = 1 - \chi^2(\rho x_{1-\alpha,\, n-1}; n-1)$$

for a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid \sigma = \sigma_0/\rho, \rho > 1) = \chi^2(\rho x_{\alpha,\, n-1}; n-1)$$

for a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid \sigma = \sigma_0/\rho, \rho \neq 1) = 1 - \chi^2(\rho x_{1-\alpha/2,\, n-1}; n-1) + \chi^2(\rho x_{\alpha/2,\, n-1}; n-1)$$

where:

α is the significance level

n is the sample size

$\rho = \sigma_0/\sigma_a$

$x_{1-\alpha,n}$ is the $(1 - \alpha)^{th}$ quantile of a central χ^2 distribution with ν degrees of freedom

$\chi^2(x, \nu)$ is the cumulative distribution function of a central χ^2 distribution with ν degrees of freedom.

Power for One Sample Equivalence

Use the Power Explorer for One Sample Equivalence to determine a sample size for an equivalence test about one mean. Select **DOE > Sample Size Explorers > Power > Power for One Sample Equivalence**. Explore the trade offs between variability assumptions, sample size, power, significance, and the equivalence range. Sample size and power are associated with the following hypothesis test:

$$H_0: \mu - \mu_0 \geq \delta_M \text{ or } H_0: \mu - \mu_0 \leq \delta_m$$

versus the alternative:

$$H_a: \delta_m < \mu - \mu_0 < \delta_M$$

where μ is the true mean, μ_0 is the reference value, and (δ_m, δ_M) is the equivalence range. For the same significance level and power, a larger sample size is needed to detect a small difference than to detect a large difference. It is assumed that the population of interest is normally distributed with mean μ and standard deviation σ.

Power Explorer for One Sample Equivalence Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Equivalence Range

Maximum difference Specifies the maximum value, above which the mean is considered different from the reference mean

Minimum difference Specifies the minimum value, below which the mean is considered different from the reference mean.

Note: Typically, the equivalence range is symmetric.

Fixed Parameters

Alpha The probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Std Dev (σ) The assumed population standard deviation.

Population standard deviation known Specifies calculations based on a known, rather than assumed, population standard deviation.

Test Parameters Parameters that are inter-related and update as you make changes.

Difference in Means Specifies the difference between the true mean and the hypothesized or reference mean such that the two means are considered equivalent.

Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed your experiment.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the One Sample Equivalence Explorer

The power calculations for testing equivalence in one sample group is based on methods described in Chow et al. (2008).

If σ is unknown, the power (1-β) is computed as follows:

$$1-\beta = T\left(-t_{1-\alpha,\, n-1};n-1,\, \frac{\delta-\delta_M}{\sigma/(\sqrt{n})}\right) - T\left(t_{1-\alpha,\, n-1};n-1,\, \frac{\delta-\delta_m}{\sigma/(\sqrt{n})}\right)$$

where:

α is the significance level

n is the sample size

σ is the assumed population standard deviation

δ is the difference to detect

(δ_m, δ_M) is the equivalence range

$t_{1-\alpha,\nu}$ is the $(1-\alpha)^{th}$ quantile of the central t-distribution with ν degrees of freedom

$T(t; \nu, \lambda)$ is the cumulative distribution function of the non-central t distribution with ν degrees of freedom and non-centrality parameter λ.

If σ is known, then power $(1-\beta)$ is computed as follows:

$$1-\beta = \Phi\left(\frac{|\delta-\delta_M|}{\sigma/\sqrt{n}} - z_{1-\alpha}\right) + \Phi\left(\frac{|\delta-\delta_m|}{\sigma/\sqrt{n}} - z_{1-\alpha}\right) - 1$$

Power for Two Independent Sample Means

Use the Power Explorer for Two Independent Sample Means to determine a sample size for a hypothesis test about the means from two groups. Select **DOE > Sample Size Explorers > Power > Power for Two Independent Sample Means**. Explore the trade offs between sample size, power, significance, and the hypothesized difference to detect. Sample size and power are associated with the following hypothesis test:

$$H_0: \mu_1 - \mu_2 = 0$$

versus the two-sided alternative:

$$H_a: \mu_1 - \mu_2 \neq 0$$

or versus a one-sided alternative:

$$H_a: \mu_1 - \mu_2 < 0 \text{ or } H_a: \mu_1 - \mu_2 > 0$$

where μ_1 and μ_2 are the true means of the two populations. It is assumed that the populations of interest are normally distributed and that you want to detect a difference of δ between the means.

Power Explorer for Two Independent Sample Means Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Test Type Specifies a one or two-sided hypothesis test.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Group 1 StdDev (σ_1) Specifies the assumed standard deviation for one of your groups, Group 1. An estimate of the error standard deviation could be the root mean square error (RMSE) from a previous model fit.

Group 2 StdDev (σ_2) Specifies the assumed standard deviation for the second group, Group 2. An estimate of the error standard deviation could be the root mean square error (RMSE) from a previous model fit.

Population standard deviations known Specifies calculations based on a known, rather than assumed, population standard deviations.

Test Parameters **Difference to detect (Δ)** Specifies the smallest difference between the group means that you want to be able to declare statistically significant.

Group 1 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 1 in your experiment. Select Lock to lock the Group 1 sample size.

Group 2 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 2 in your experiment. Select Lock to lock the Group 2 sample size.

Total Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment. The power curve is based on total sample size. Set Lock to lock the total sample size

Note: Adjusting the sample size for one group adjusts the total sample size unless the total sample size is locked. In that case, adjusting the sample size for one group adjust the sample size for the second group. Adjusting the total sample size adjusts the two groups proportionally, unless one group is locked. Use only one lock at a time.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for Two Independent Means

The power calculations for testing the difference in means of two sample groups are based on the traditional t test, or if σ_1 and σ_2 are known, the z test.

For the case when σ is unknown and it is assumed that $\sigma_1=\sigma_2=\sigma$, the power is calculated based on the form of the alternative hypothesis. For a one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid \mu_1 = \mu_2 + \delta) = 1 - T\left(t_{1-\alpha,\, n_1+n_2-2};\, n_1+n_2-2,\, \frac{\delta}{\sigma\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}\right)$$

For a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid \mu_1 = \mu_2 - \delta) = T\left(-t_{1-\alpha,\, n_1+n_2-2};\, n_1+n_2-2,\, \frac{\delta}{\sigma\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}\right)$$

For a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid \mu_1 = \mu_2 \pm \delta) =$$

$$1-T\left(t_{1-\frac{\alpha}{2},\, n_1+n_2-2};n_1+n_2-2,\ \frac{\delta}{\sigma\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}\right)+T\left(-t_{1-\frac{\alpha}{2},\, n_1+n_2-2};n_1+n_2-2,\ \frac{\delta}{\sigma\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}\right)$$

where:

α is the significance level

n_1 and n_2 are the group sample sizes

σ is the pooled standard deviation

δ is the difference to detect

$t_{1-\alpha,\nu}$ is the $(1 - \alpha)^{th}$ quantile of the central t-distribution with ν degrees of freedom

$T(t;\ \nu,\ \lambda)$ is the cumulative distribution function of the non-central t distribution with ν degrees of freedom and non-centrality parameter λ.

When σ_1 and σ_2 are not assumed to be equal then the degrees of freedom, $n_1+ n_2 -2$ in the above equations, is computed as follows:

$$\mathrm{df}=\frac{\left(\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}\right)^2}{\frac{1}{n_1-1}\left(\frac{\sigma_1^2}{n_1}\right)^2+\frac{1}{n_2-1}\left(\frac{\sigma_2^2}{n_2}\right)^2}$$

When σ_1 and σ_2 are known the z distribution is used for the power calculation. The power is calculated based on the form of the alternative hypothesis. For a one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid \mu_1 = \mu_2 + \delta) = 1 - \Phi\left(z_{1-\alpha} - \frac{\delta}{\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}}\right)$$

For a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid \mu_1 = \mu_2 - \delta) = \Phi\left(-z_{1-\alpha} - \frac{\delta}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}\right)$$

For a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid \mu_1 = \mu_2 \pm \delta) = 1 - \Phi\left(z_{1-\frac{\alpha}{2}} - \frac{\delta}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}\right) + \Phi\left(-z_{1-\alpha} - \frac{\delta}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}\right)$$

where:

α is the significance level

n_1 and n_2 are the group sample sizes

σ_1 and σ_2 are known group standard deviations

δ is the difference to detect

$z_{1-\alpha}$ is the $(1 - \alpha)^{th}$ quantile of the z-distribution

$\Phi(x)$ is the cumulative distribution function of the normal distribution.

Power for Two Independent Sample Proportions

Use the Power Explorer for Two Independent Sample Proportions to determine a sample size for a hypothesis test for proportions from two groups. Select **DOE > Sample Size Explorers > Power > Power for Two Independent Sample Proportions**. Explore the trade offs between sample size, power, significance, and the hypothesized difference to detect. Sample size and power are associated with the following hypothesis test:

$$H_0: p_1 - p_2 = D_0$$

versus the two-sided alternative:

$$H_a: p_1 - p_2 \neq D_0$$

or versus either of the following one-sided alternatives:

$$H_a: (p_1 - p_2) < D_0 \text{ or } H_a: (p_1 - p_2) > D_0$$

where p_1 and p_2 are the population proportions from two populations, and D_0 is the hypothesized difference in proportions.

Power Explorer for Two Independent Sample Proportions Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Test Type Specifies a one or two-sided hypothesis test.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Test Parameters Parameters that are inter-related and update as you make changes.

Group 1proportion (p1) Specifies the proportion that you anticipate or assume for Group 1.

Group 2 proportion (p2) Specifies the proportion that you anticipate or assume for Group 2.

Group 1 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 1 in your experiment. Select Lock to lock the Group 1 sample size.

Group 2 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 2 in your experiment. Select Lock to lock the Group 2 sample size.

Total Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment. The power curve is based on total sample size. Select Lock to lock the total sample size.

Note: Adjusting the sample size for one group adjusts the total sample size unless the total sample size is locked. In that case, adjusting the sample size for one group adjust the sample size for the second group. Adjusting the total sample size adjusts the two groups proportionally, unless one group is locked. Use only one lock at a time.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for Two Independent Sample Proportions

The power calculations for testing the difference in proportions from two sample groups are based on the normal approximation. The calculations depend on the form of the alternative hypothesis. For a one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid p_1 = p_2 + \delta) = 1 - \Phi\left(z_{1-\alpha} - \frac{\delta}{\sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}}\right)$$

For a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid p_1 = p_2 - \delta) = \Phi\left(-z_{1-\alpha} - \frac{\delta}{\sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}}\right)$$

For a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid p_1 = p_2 \pm \delta) =$$

$$1 - \Phi\left(z_{1-\frac{\alpha}{2}} - \frac{\delta}{\sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}}\right) + \Phi\left(-z_{1-\alpha} - \frac{\delta}{\sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}}\right)$$

where:

α is the significance level

n_1 and n_2 are the group sample sizes

p_1 and p_2 are the group proportions

δ is the difference to detect

$z_{1-\alpha}$ is the $(1 - \alpha)^{th}$ quantile of the z-distribution

$\Phi(x)$ is the cumulative distribution function of the normal distribution.

Power for Two Independent Sample Variances

Use the Power Explorer for Two Independent Sample Variances to determine a sample size for a hypothesis test for variances from two groups. Select **DOE > Sample Size Explorers > Power > Power for Two Independent Sample Variances**. Explore the trade offs between sample size, power, significance, and the hypothesized difference to detect. Sample size and power are associated with the following hypothesis test:

$$H_0: \sigma_1 = \sigma_2$$

versus the two-sided alternative:

$$H_a: \sigma_1 \neq \sigma_2$$

or versus a one-sided alternative:

$$H_a: \sigma_1 < \sigma_2 \text{ or } H_a: \sigma_1 > \sigma_2$$

where σ_1 is the variance for Group 1 and σ_2 is the variance for Group 2. The difference to detect is an amount away from σ_2 that one considers as important to detect based on a set of samples. This difference is expressed as the ratio of σ_2/σ_1, or the ratio of your variances. For the same significance level and power, a larger sample size is needed to detect a small difference in variances than to detect a large difference. It is assumed that the populations of interest are normally distributed.

Power Explorer for Two Independent Sample Variances Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Test Type Specifies a one or two-sided hypothesis test.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Test Parameters Parameters that are inter-related and update as you make changes.

Ratio of variances (Group 2/Group 1) Specifies the ratio of the variances.

Group 1 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 1 in your experiment. Select Lock to lock the Group 1 sample size.

Group 2 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 2 in your experiment. Select Lock to lock the Group 2 sample size.

Total Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment. The power curve is based on total sample size. Select Lock to lock the total sample size.

Note: Adjusting the sample size for one group adjusts the total sample size unless the total sample size is locked. In that case, adjusting the sample size for one group adjust the sample size for the second group. Adjusting the total sample size adjusts the two groups proportionally, unless one group is locked. Use only one lock at a time.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Statistical Details for the Two Independent Sample Variances Explorer

The power calculations for testing the ratio of variances from two sample groups is based on the standard F test. The calculations depend on the form of the alternative hypothesis. For a one-sided, higher alternative:

$$\Pr(\text{reject } H_0 \mid \sigma_1 = \sigma_2/\rho, \rho < 1) = 1 - F\left(\frac{1}{\rho} f_{1-\alpha, n_1-1, n_2-1}; n_1 - 1, n_2 - 1\right)$$

For a one-sided, lower alternative:

$$\Pr(\text{reject } H_0 \mid \sigma_1 = \sigma_2/\rho, \rho > 1) = F\left(\frac{1}{\rho} f_{\alpha, n_1-1, n_2-1}; n_1 - 1, n_2 - 1\right)$$

For a two-sided alternative:

$$\Pr(\text{reject } H_0 \mid \sigma_1 = \sigma_2/\rho, \rho \neq 1) =$$

$$1 - F\left(\frac{1}{\rho} f_{1-\frac{\alpha}{2}, n_1-1, n_2-1}; n_1-1, n_2-1\right) + F\left(\frac{1}{\rho} f_{\frac{\alpha}{2}, n_1-1, n_2-1}; n_1-1, n_2-1\right)$$

where:

α is the significance level

n_1 and n_2 are the group sample sizes

$\rho = \sigma_2/\sigma_1$

$f_{1-\alpha,\nu1,\nu2}$ is the $(1 - \alpha)^{th}$ quantile of an F distribution with ν_1 and ν_2 degrees of freedom

$F(x, \nu)$ is the cumulative distribution function of an F distribution with ν degrees of freedom.

Power for Two Independent Sample Equivalence

Use the Power Explorer for Two Sample Equivalence to determine a sample size for an equivalence test of two groups. Select **DOE > Sample Size Explorers > Power > Power for Two independent Sample Equivalence**. Explore the trade offs between variability assumptions, sample size, power, significance, and the equivalence range. Sample size and power are associated with the following hypothesis test:

H_0: $\mu_1 - \mu_2 \geq \delta_M$ or H_0: $\mu_1 - \mu_2 \leq \delta_m$

versus the alternative:

H_a: $\delta_m < \mu_1 - \mu_2 < \delta_M$

where μ_1 and μ_2 the true group means and (δ_m, δ_M) is the equivalence range. For the same significance level and power, a larger sample size is needed to detect a small difference than to detect a large difference. It is assumed that the populations of interest are normally distributed.

Power Explorer for Two Independent Sample Equivalence Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Equivalence Range

Maximum difference Specifies the margin above which the mean is considered different from the reference mean

Minimum difference Specifies the margin below which the mean is considered different from the reference mean.

Note: Typically, the equivalence margin is symmetric. However, it does not have to be symmetric.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Group 1 StdDev (σ_1) Specifies the assumed population standard deviation for one of your groups, Group 1.

Group 2 StdDev (σ_2) Specifies the assumed population standard deviation for the second group, Group 2.

Population standard deviations known Specifies calculations based on a known, rather than assumed, population standard deviations.

Test Parameters Parameters that are inter-related and update as you make changes.

Difference in Means Specifies the difference between group means that defines the equivalence range.

Group 1 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 1 in your experiment. Select Lock to lock the Group 1 sample size.

Group 2 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 2 in your experiment. Select Lock to lock the Group 2 sample size.

Total Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment. The power curve is based on total sample size. Select Lock to lock the total sample size.

Note: Adjusting the sample size for one group adjusts the total sample size unless the total sample size is locked. In that case, adjusting the sample size for one group adjust the sample size for the second group. Adjusting the total sample size adjusts the two groups proportionally, unless one group is locked. Use only one lock at a time.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the Two Independent Sample Equivalence Explorer

The power calculations for testing equivalence of two group means is based on methods described in Chow et al. (2008).

If σ_1 and σ_2 are unknown, the power (1-β) is computed as follows:

$$1-\beta = \left(-t_{1-\alpha,\, df};df, \frac{\delta - \delta_M}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \right) - T\left(t_{1-\alpha,\, df};df, \frac{\delta - \delta_m}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \right)$$

where:

$$df = \begin{cases} n_1 + n_2 - 2 & \text{if } \sigma_1 = \sigma_2 \\ \dfrac{\left(\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}\right)^2}{\frac{1}{n_1 - 1}\left(\frac{\sigma_1^2}{n_1}\right)^2 + \frac{1}{n_2 - 1}\left(\frac{\sigma_2^2}{n_2}\right)^2} & \text{if } \sigma_1 \neq \sigma_2 \end{cases}$$

α is the significance level

n_1 and n_2 are the group sample sizes

σ_1 and σ_2 are the assumed group standard deviations

δ is the difference to detect

(δ_m, δ_M) is the equivalence range

$t_{1-\alpha,\nu,}$is the $(1 - \alpha)^{th}$ quantile of the central t-distribution with ν degrees of freedom

$T(t; \nu, \lambda)$ is the cumulative distribution function of the non-central t distribution with ν degrees of freedom and non-centrality parameter λ.

If σ is known, then power (1-β) is computed as follows:

$$1-\beta = \Phi\left(\left|\frac{\delta-\delta_M}{\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}}\right| - z_{1-\alpha}\right) + \Phi\left(\left|\frac{\delta-\delta_m}{\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}}\right| - z_{1-\alpha}\right) - 1$$

Power for ANOVA

Use the Power Explorer for ANOVA to determine a sample size for a study of k groups or treatments to be analyzed using ANOVA. Select **DOE > Sample Size Explorers > Power > Power for ANOVA**. Explore the trade offs between variability assumptions, sample size, power and significance. Sample size and power are associated with the following hypothesis test:

$$H_0: \mu_1 = \mu_2 = \ldots = \mu_k$$

versus the two-sided alternative:

H_a : not all means equal

where:

$$X_{ij} \sim N(\mu_j, \sigma^2) \text{ for } i = 1, \ldots, n,\ j = 1, \ldots, k$$

Power Explorer for ANOVA Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Number of groups Specifies the number of groups or treatments in your experiment.

Within-group variance (σ^2) Specifies the assumed population variance for each group where the standard deviation is assumed to be equal across groups.

Test Parameters Parameters that are inter-related and update as you make changes.

Between-group variance Specifies the variance of the individual group means around the grand mean.

Sample size (per group) Specifies the number of observations (runs, experimental units, or treatments) needed for each group in your experiment.

Power Specifies the probability of rejecting the null hypothesis when it is false. With all other parameters fixed, power increases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the ANOVA Explorer

The power calculations for testing differences among means from multiple groups assuming equal standard deviations for each group is based on the standard F test. The power (1-β) is computed as follows:

$$1 - \beta = F\left(f_{q,\, K-1,\, N-K},\ \frac{\sigma^2_{BG}}{\sigma^2}\right)$$

where

$F(q, df_1, df_2, \lambda)$ is a non-central F-distribution with noncentrality parameter λ.

K is the number of groups

n is the number of samples within each group (assumed equal for all groups)

$N = nK$

σ^2_{BG} is the variation of the group means around the grand mean

σ^2 is the within group variance (assumed equal for all groups)

Confidence Interval Calculators

Use the confidence interval calculators to select a study sample size to obtain an interval estimate with pre-specified characteristics. Explore the impact of sample size on the margin of error of an interval estimate. Explore sample size for a confidence, prediction, or tolerance interval for one mean. Explore sample size for a confidence interval for two means, or one or two proportions or variances.

For interval estimation, select **DOE > Sample Size Explorers > Confidence Intervals**:

- "Margin of Error for One Sample Mean"
- "Margin of Error for One Sample Proportion"
- "Margin of Error for One Sample Variance"
- "Margin of Error for Two Independent Sample Means"
- "Margin of Error for Two Independent Sample Proportions"
- "Margin of Error for Two Independent Sample Variances"

Margin of Error for One Sample Mean

Use the Interval Explorer for one sample mean to determine a sample size for a confidence, prediction, or tolerance interval. Select **DOE > Sample Size Explorers > Confidence Intervals > Margin of Error for One Sample Mean**. Explore the trade offs between variability assumptions, sample size, significance, and the margin of error.

Interval Explorer for One Sample Mean Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Interval Purpose

Confidence Specifies a confidence interval for a mean.

Prediction Specifies a prediction interval for one future observation.

Tolerance Specifies a tolerance interval to cover a proportion of the population.

Interval Type

Bound Specifies a one-sided interval (upper or lower bound)

Interval Specifies a two-sided interval.

Fixed Parameters

Alpha Specifies the confidence level is 1 - alpha. The default alpha level is 0.05 for a 95% confidence interval.

Std Dev (σ) Specifies the assumed population standard deviation.

Proportion (Available only when Tolerance is selected for Interval Purpose.) Specifies the proportion of the population for the tolerance interval to cover.

Population standard deviation known Specifies calculations based on a known, rather than assumed, population standard deviation.

Interval Parameters Parameters that are inter-related and update as you make changes.

Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed to construct your interval.

Margin of Error (Available only when Interval is selected for Interval Type .) Specifies the half-width of the interval. With all other parameters fixed, margin of error decreases as sample size increases.

Bound Size (Available only when Bound is selected for Interval Type.) Specifies the distance from the bound to the estimate. With all other parameters fixed, the bound decreases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the One Sample Interval Explorer

The calculation for each interval type uses the standard normal-based procedures if σ is known and *t* distribution procedures or approximations otherwise.

Confidence Intervals

The margin of error (MOE) for confidence intervals is calculated as follows:

$$\text{MOE} = \begin{cases} z_{1-\alpha/2}\dfrac{\sigma}{\sqrt{n}} & \text{if } \sigma \text{ known} \\ t_{1-\alpha/2,\, n-1}\dfrac{s}{\sqrt{n}} & \text{if } \sigma \text{ unknown} \end{cases}$$

The bound is calculated as follows:

$$\text{bound} = \begin{cases} \pm z_{1-\alpha}\dfrac{\sigma}{\sqrt{n}} & \text{if } \sigma \text{ known} \\ t_{1-\alpha,\, n-1}\dfrac{s}{\sqrt{n}} & \text{if } \sigma \text{ unknown} \end{cases}$$

Prediction Intervals

The margin of error (MOE) for prediction intervals is calculated as follows:

$$\text{MOE} = \begin{cases} z_{1-\alpha/2}\sigma\sqrt{\left(1+\frac{1}{n}\right)} & \text{if } \sigma \text{ known} \\ t_{1-\alpha/2,\,n-1}s\sqrt{\left(1+\frac{1}{n}\right)} & \text{if } \sigma \text{ unknown} \end{cases}$$

The bound is calculated as follows:

$$\text{bound} = \begin{cases} \pm z_{1-\alpha}\sigma\sqrt{\left(1+\frac{1}{n}\right)} & \text{if } \sigma \text{ known} \\ \pm t_{1-\alpha,\,n-1}s\sqrt{\left(1+\frac{1}{n}\right)} & \text{if } \sigma \text{ unknown} \end{cases}$$

Tolerance Intervals

For the tolerance interval on a proportion q of the population, the margin of error or bound is computed based on approximate procedures described in Krishnamoorthy and Mathew (2009).

The margin of error (MOE) for tolerance intervals is calculated as follows:

$$\text{MOE} = \begin{cases} \sigma\sqrt{\left(\frac{N\chi^2_{q,\,1,\,\frac{1}{n}}}{\chi^2_{\alpha,\,N}}\right)} & \text{if } \sigma \text{ known} \\ s\sqrt{\left(\frac{(n-1)\chi^2_{q,\,1,\,\frac{1}{n}}}{\chi^2_{\alpha,\,n-1}}\right)} & \text{if } \sigma \text{ unknown} \end{cases}$$

The bound is calculated as follows:

$$\text{bound} = \begin{cases} \pm \dfrac{t_{1-\alpha, N, z_q\sqrt{n}}}{\sqrt{n}} & \text{if } \sigma \text{ known} \\ \pm \dfrac{t_{1-\alpha, n-1, z_q\sqrt{n}}}{\sqrt{n}} & \text{if } \sigma \text{ unknown} \end{cases}$$

where N is a suitably large integer value to represent asymptotic behavior. JMP sets the value of N to 2000.

Margin of Error for One Sample Proportion

Use the Interval Explorer for One Sample Proportion to determine a sample size for a confidence interval. Select **DOE > Sample Size Explorers > Confidence Intervals > Margin of Error for One Sample Proportion**. Explore the trade offs between the assumed proportion, sample size, significance, and the margin of error for your interval. Calculations use the Normal approximation.

Interval Explorer for One Sample Proportion Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Interval Type

Bound Specifies a one-side of the interval (upper or lower bound)

Interval Specifies a two-sided interval.

Fixed Parameters

Alpha Specifies the confidence level, 1 - Alpha. The default alpha level is 0.05 for a 95% confidence interval.

Interval Parameters Parameters that are inter-related and update as you make changes.

Proportion Specifies the assumed proportion for the interval.

Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed to construct your interval.

Margin of Error Specifies the half-width of the interval. With all other parameters fixed, margin of error decreases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the One Sample Proportion Interval Explorer

The interval calculations for capturing a population proportion is based on the Agresti-Coull method, see Agresti and Coull(1998).

The margin of error (MOE) for confidence intervals is calculated as follows:

$$\text{MOE} = z_{1-\alpha/2}\sqrt{\frac{\tilde{p}(1-\tilde{p})}{\tilde{n}}}$$

The bound is calculated as follows:

$$\text{bound} = \pm z_{1-\alpha}\sqrt{\frac{\tilde{p}(1-\tilde{p})}{\tilde{n}}}$$

where

$$\tilde{n} = n + z_q^2$$

and

$$\tilde{p} = \frac{np + z_q^2/2}{\tilde{n}}$$

and $q = 1 - \alpha/2$ for an interval or $q = 1 - \alpha$ for a bound.

Margin of Error for One Sample Variance

Use the Interval Explorer for One Sample Variance to determine a sample size for a confidence interval. Select **DOE > Sample Size Explorers > Confidence Intervals > Margin of Error for One Sample Variance**. Explore the trade offs between sample size, significance, and the margin of error for your interval.

Interval Explorer for One Sample Variance Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Interval Type

Lower Bound Specifies a one-sided lower interval.

Upper Bound Specifies a one-sided upper interval.

Interval Specifies a two-sided interval.

Fixed Parameters

Alpha Specifies the confidence level, 1 - Alpha. The default alpha level is 0.05 for a 95% confidence interval.

Interval Parameters Parameters that are inter-related and update as you make changes.

Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed to construct your interval.

Interval Width Specifies the full width of the interval. With all other parameters fixed, interval width decreases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the One Sample Variance Interval Explorer

The interval calculations for capturing a population variance is based on the X^2 distribution. Note that the interval bounds are not symmetric around the sample estimate.

The interval width is computed as follows:

$$\text{width} = (n-1)\left(\frac{1}{\chi^2_{\alpha/2,\, n-1}} - \frac{1}{\chi^2_{1-\alpha/2,\, n-1}}\right)$$

The lower bound is computed as follows:

$$\text{lower bound} = \frac{n-1}{\chi^2_{1-\alpha,\, n-1}}$$

The upper bound is computed as follows:

$$\text{upper bound} = \frac{n-1}{\chi^2_{\alpha,\, n-1}}$$

Margin of Error for Two Independent Sample Means

Use the Interval Explorer for Two Independent Sample Means to determine a sample size for a confidence interval. Select **DOE > Sample Size Explorers > Confidence Intervals > Margin of Error for Two Independent Sample Means**. Explore the trade offs between variability assumptions, sample size, significance, and the margin of error for your interval.

Interval Explorer for Two Independent Sample Mean Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Interval Type

Bound Specifies a one-sided interval (upper or lower bound)

Interval Specifies a two-sided interval

Fixed Parameters

Alpha Specifies the confidence level, 1 - Alpha. The default alpha level is 0.05 for a 95% confidence interval.

Group 1 StdDev (σ_1) Specifies the assumed standard deviation for one of your groups, Group 1.

Group 2 StdDev ($\sigma 2$) Specifies the assumed standard deviation for the second group, Group 2.

Population standard deviations known Specifies calculations based on known population standard deviations.

Interval Parameters Parameters that are inter-related and update as you make changes.

Group 1 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 1 in your experiment. Select Lock to lock the Group 1 sample size.

Group 2 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 2 in your experiment. Select Lock to lock the Group 2 sample size.

Total Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment. The margin of error curve is based on total sample size. Select Lock to lock the total sample size.

Note: Adjusting the sample size for one group adjusts the total sample size unless the total sample size is locked. In that case, adjusting the sample size for one group adjust the sample size for the second group. Adjusting the total sample size adjusts the two groups proportionally, unless one group is locked. Use only one lock at a time.

Margin of Error Specifies the half-width of the interval. With all other parameters fixed, margin of error decreases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the Two Independent Sample Interval Explorer

The interval calculations for capturing the difference in population means is based on the standard normal or *t* distributions based on whether σ_1 and σ_2 are known or unknown.

The margin of error (MOE) is calculated as follows:

$$\text{MOE} = \begin{cases} z_{1-\alpha/2}\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}} & \text{if } \sigma_1 \text{and} \sigma_2 \text{ known} \\ t_{1-\alpha/2,\, n-1}\sqrt{\frac{s_1^2}{n_1}+\frac{s_2^2}{n_2}} & \text{if } \sigma_1 \text{and} \sigma_2 \text{ unknown} \end{cases}$$

When σ_1 and σ_2 are unknown, the bounds are calculated as follows:

$$\text{bound} = \pm t_{1-\alpha,\, df}\sqrt{\frac{s_1^2}{n_1}+\frac{s_2^2}{n_2}}$$

where

$$df = \begin{cases} n_1 + n_2 - 2 & \text{if } \sigma_1 = \sigma_2 \\ \dfrac{\left(\dfrac{s_1^2}{n_1} + \dfrac{s_2^2}{n_2}\right)^2}{\dfrac{1}{n_1 - 1}\left(\dfrac{s_1^2}{n_1}\right)^2 + \dfrac{1}{n_2 - 1}\left(\dfrac{s_2^2}{n_2}\right)^2} & \text{if } \sigma_1 \neq \sigma_2 \end{cases}$$

When σ is known, the bounds are calculated as follows:

$$\text{bound} = \pm z_{1-\alpha}\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$$

Margin of Error for Two Independent Sample Proportions

Use the Interval Explorer for Two Independent Sample Proportions to determine a sample size for a confidence interval for the difference in two proportions, for log relative risk, or for log odds. Select **DOE > Sample Size Explorers > Confidence Intervals > Margin of Error for Two Independent Sample Proportions**. Explore the trade offs between variability assumptions, sample size, significance, and the margin of error for your interval.

Interval Explorer for Two Independent Sample Proportions Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Interval Type

Bound Specifies one side of the interval (upper or lower bound)

Interval Specifies a two-sided interval.

Comparison Method

Difference in Proportions Specifies a confidence interval for the difference in two proportions (p_1 - p_2).

Log Relative Risk Specifies a confidence interval for the relative risk (p_1/p_2) on a log scale.

Log Odds Specifies a confidence interval for the odds ($p_1/(1-p_2)$) on a log scale.

Fixed Parameters

Alpha Specifies the confidence level, 1 - Alpha. The default alpha level is 0.05 for a 95% confidence interval.

Interval Parameters Parameters that are inter-related and update as you make changes.

Group 1 Proportion Specifies the assumed proportion for Group 1.

Group 2 Proportion Specifies the assumed proportion for Group 2.

Group 1 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 1 in your experiment. Select Lock to lock the Group 1 sample size.

Group 2 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 2 in your experiment. Select Lock to lock the Group 2 sample size.

Total Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment. The margin of error curve is based on total sample size. Select Lock to lock the total sample size.

Note: Adjusting the sample size for one group adjusts the total sample size unless the total sample size is locked. In that case, adjusting the sample size for one group adjust the sample size for the second group. Adjusting the total sample size adjusts the two groups proportionally, unless one group is locked. Use only one lock at a time.

Margin of Error Specifies the half-width of the interval. With all other parameters fixed, margin of error decreases as sample size increases.

Bound Size Specifies the bound of the interval. With all other parameters fixed, margin of error decreases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the Two Independent Proportions Interval Explorer

The calculations for each interval type are based on normal approximations.

Difference in Proportions

For intervals about the difference in proportions the margin of error (MOE) is calculated as follows:

$$\text{MOE} = z_{1-\alpha/2}\sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}$$

The bounds are calculated as follows:

$$\text{bound} = \pm z_{1-\alpha}\sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}$$

Log Relative Risk

For the logarithm of the relative risk the margin of error (MOE) is calculated as follows:

$$\text{MOE} = z_{1-\alpha/2}\sqrt{\frac{(1-p_1)}{n_1 p_1} + \frac{(1-p_2)}{n_2 p_2}}$$

The bounds are calculated as follows:

$$\text{bound} = \pm z_{1-\alpha}\sqrt{\frac{(1-p_1)}{n_1 p_1} + \frac{(1-p_2)}{n_2 p_2}}$$

Log Odds

For the logarithm of the odds ratio, the margin of error is calculated as follows:

$$\text{MOE} = z_{1-\alpha/2}\sqrt{\frac{1}{n_1 p_1} + \frac{1}{n_1(1-p_1)} + \frac{1}{n_2 p_2} + \frac{1}{n_2(1-p_2)}}$$

The bounds are calculated as follows:

$$\text{bound} = \pm z_{1-\alpha}\sqrt{\frac{1}{n_1 p_1} + \frac{1}{n_1(1-p_1)} + \frac{1}{n_2 p_2} + \frac{1}{n_2(1-p_2)}}$$

Margin of Error for Two Independent Sample Variances

Use the Interval Explorer for Two Independent Sample Variance to determine a sample size for a confidence interval. Select **DOE > Sample Size Explorers > Confidence Intervals > Margin of Error for Two Independent Sample Variances**. Explore the trade offs between sample size, significance, and the margin of error for your interval.

Interval Explorer for Two Independent Sample Variances Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Interval Type

Lower Bound Specifies a one-sided lower interval.

Upper Bound Specifies a one-sided upper interval.

Interval Specifies a two-sided interval.

Fixed Parameters

Alpha Specifies the confidence level, 1 - Alpha. The default alpha level is 0.05 for a 95% confidence interval.

Interval Parameters

Group 1 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 1 in your experiment. Select Lock to lock the Group 1 sample size.

Group 2 Sample Size Specifies the number of observations (runs, experimental units, or samples) needed for Group 2 in your experiment. Select Lock to lock the Group 2 sample size.

Total Sample Size Specifies the total number of observations (runs, experimental units, or samples) needed for your experiment. The margin of error curve is based on total sample size. Select Lock to lock the total sample size.

Note: Adjusting the sample size for one group adjusts the total sample size unless the total sample size is locked. In that case, adjusting the sample size for one group adjust the sample size for the second group. Adjusting the total sample size adjusts the two groups proportionally, unless one group is locked. Use only one lock at a time.

Interval Width Specifies the full width of the interval. With all other parameters fixed, interval width decreases as sample size increases.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the Ratio Two Independent Variances Interval Explorer

The interval calculations for capturing the ratio of two population variances is based on the *F*-distribution. The margin of error (MOE) is calculated as follows:

$$\text{MOE} = F_{1-\alpha/2,\, n_1-1,\, n_2-1} - F_{\alpha/2,\, n_1-1,\, n_2-1}$$

The lower bound is calculated as follows:

$$\text{lower bound} = F_{\alpha,\, n_1-1,\, n_2-1}$$

The upper bound is calculated as follows:

$$\text{upper bound} = F_{1-\alpha,\, n_1-1,\, n_2-1}$$

Reliability Demonstration Calculators

Use the Reliability Demonstration calculators to test a specified number of units for a specified period of time. If fewer than *k* units fail, the demonstration passes, and you can conclude that the product reliability meets or exceeds a reliability standard.

For reliability demonstrations, select **DOE > Sample Size Explorers > Reliability**:

- "Parametric Reliability Demonstration"
- "Nonparametric Reliability Demonstration"

Parametric Reliability Demonstration

Use the Demonstration Explorer for Parametric Reliability to determine the number of units to put on test for a reliability demonstration. Select **DOE > Sample Size Explorers > Reliability > Parametric Reliability Demonstration**. Explore the trade offs between time, significance, number of failures, and the number of units to test.

Demonstration Explorer for Parametric Reliability Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Distribution Specifies the assumed failure time distribution. Distributions available are: Weibull and Lognormal. For more information about these distributions, see *Reliability and Survival Methods*.

Demonstration Time Specifies the minimum length of time that an item under test should survive with the intended reliability.

Demonstration Reliability Specifies the probability that an item under test survives until the defined Demonstration Time of the reliability standard

Scale parameter Specifies the scale parameter for the failure time distribution.

Note: The Scale field is denoted Weibull β when the specified Distribution is Weibull.

Demonstration Parameters Parameters that are inter-related and update as you make changes.

Maximum failures allowed Specifies the maximum number of failures allowed for a successful test demonstration.

Sample Size Specifies the total number of units on test needed for your demonstration.

Test Time Specifies th length of the demonstration test in time.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the Parametric Reliability Demonstration Explorer

For a parametric reliability demonstration test, the test time needed to demonstrate that the product achieves a demonstration time T_d with reliability R_d is computed using a Binomial distribution with probability of success computed using the assumed lifetime distribution.

The test time for the Weibull distribution is computed as follows:

$$\text{test time} = T_d\left(\frac{-\log(P)}{\log(R_d)}\right)^{1/\beta}$$

The test time for the lognormal distribution is computed as follows:

$$\text{test time} = \exp\left(\frac{z_{1-e^{-\log P}} - z_{1-R_d}}{\sigma}\right)$$

where

$$\log P = -\log\left(\frac{Sf}{Sf+1}\right)$$

$$Sf = \frac{n-c}{c+1} f_{\alpha,\, 2(n-c),\, 2(c+1)}$$

and c is the maximum number of failures allowed during the demonstration.

Nonparametric Reliability Demonstration

Use the Demonstration Explorer for Nonparametric Reliability to determine the number of units to put on test for a reliability demonstration when there is not an assumed failure rate distribution. Select **DOE > Sample Size Explorers > Reliability > Nonparametric Reliability Demonstration**. Explore the trade offs between time, significance, number of failures, and the number of units to test.

Demonstration Explorer for Nonparametric Reliability Options

Set study assumptions and explore sample sizes using the radio buttons, text boxes, and sliders. The curve updates as you make changes to the settings. Alternatively, change settings by dragging the cross hairs on the curve or adjusting the values in the axis text boxes.

Fixed Parameters

Alpha Specifies the probability of a type I error, which is the probability of rejecting the null hypothesis when it is true. It is commonly referred to as the significance level of the test. The default alpha level is 0.05.

Demonstration Parameters Parameters that are inter-related and update as you make changes.

Maximum failures allowed Specifies the maximum number of failures allowed for a successful test demonstration.

Sample Size Specifies the total number of units on test needed for your demonstration.

Demonstration Reliability Specifies the probability that the item under test survives until the defined Demonstration Time of the reliability standard.

Save Settings Saves the current settings to the Saved Settings table. This enables you to save a set of alternative study plans. See "Saved Settings in the Sample Size Explorers".

Make Data Collection Table Creates a new data table that you can use for data collection.

Statistical Details for the Nonparametric Reliability Demonstration Explorer

For a nonparametric reliability demonstration test, the demonstrated reliability is computed based on the Binomial distribution. The α quantile of the Binomial cdf can be computed by a transformation to an α quantile from the *F*-distribution, see Jowett, (1963).

The reliability is calculated as follows:

$$R = \frac{Sf}{Sf+1}$$

where

$$Sf = \frac{n-c}{c+1} f_{\alpha,\, 2(n-c),\, 2(c+1)}$$

and c is the maximum number of failures allowed during the demonstration.

Saved Settings in the Sample Size Explorers

The Saved Settings report provides a table of saved settings enabling you to compare study scenarios. Each time you click the **Save Settings** button, a new row is added to the table. Click on a row in the table to reset the explorer to those settings. Right-click in the table and select **Columns** to display additional parameter values in the table.

The Saved Settings red triangle contains the following options:

Clear Selected Rows Removes the selected rows from the Saved Settings table.

Clear All Clears all rows from the Saved Setting table.

Make Table Creates a new a data table that contains all Saved Settings table columns including those that are hidden.

Appendix A

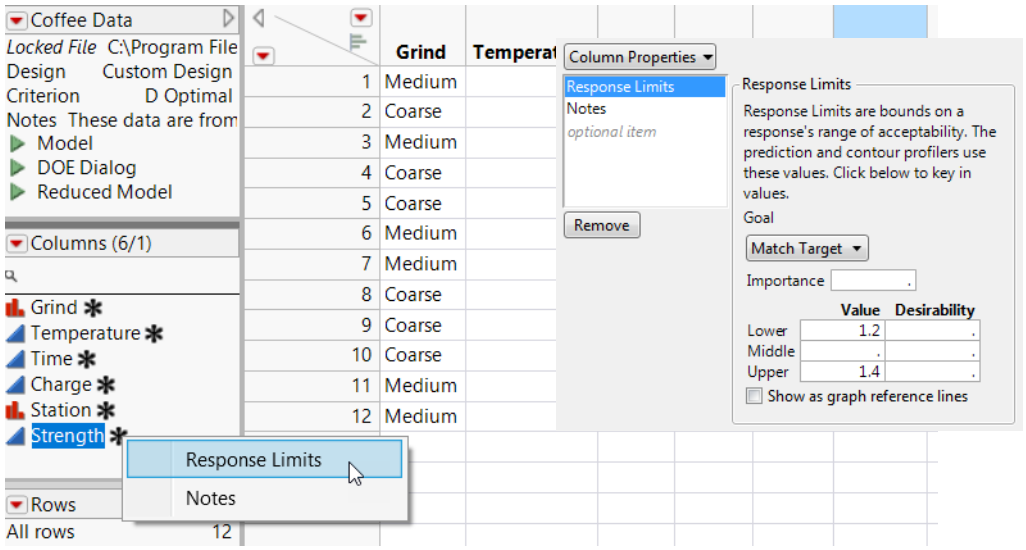
Column Properties

Understanding Column Properties Assigned by DOE

When you construct a design using the DOE platforms, column properties are saved to the data table that contains the resulting design. This appendix provides detail about only those column properties that are saved to designs that the DOE platform constructs. Examples illustrate how each column property is assigned and used by the DOE platforms. Descriptions of column properties not assigned by the DOE platforms are provided in *Using JMP*.

Some of the column properties described in this appendix are useful in general modeling situations. To use the properties more generally, you can specify them yourself. This ability is particularly useful when your design has not been created by a DOE platform. Some of the examples in this appendix illustrate situations where you add a column property on your own.

Figure A.1 Column Property Asterisks and Column Info Window



Contents

Adding and Viewing Column Properties . . . 785
Response Limits . . . 786
Response Limits Example . . . 788
Editing Response Limits . . . 791
Detection Limits . . . 794
Functional Response . . . 794
Design Role . . . 794
Design Role Example . . . 795
Coding . . . 797
Low and High Values . . . 797
Coding Column Property and Center Polynomials . . . 798
Coding Example . . . 798
Assigning Coding . . . 802
Mixture . . . 805
Mixture Example . . . 807
Factor Changes . . . 810
Factor Changes Example . . . 811
Value Order . . . 813
Value Order Example . . . 815
Assigning Value Order . . . 816
Value Labels . . . 817
Value Labels Example . . . 817
RunsPerBlock . . . 819
RunsPerBlock Example . . . 819
ConstraintState . . . 820
ConstraintState Example . . . 821

Adding and Viewing Column Properties

The DOE platforms automatically save certain column properties to the design tables that they construct. However, some of the column properties associated with designed experiments are useful in general modeling situations. To use these column properties with data tables that have not been created using DOE platforms, you can add them yourself.

Adding a Column Property

To assign a column property to one or more columns, do the following:

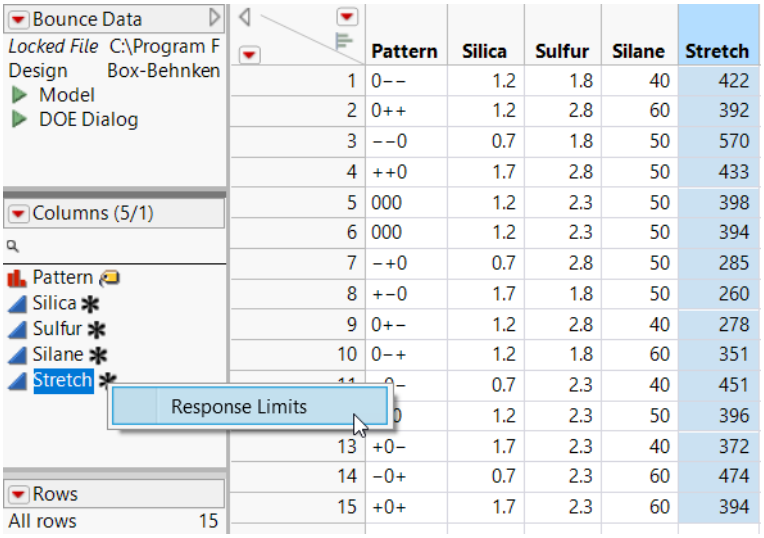
1. Select the column or columns to which you want to assign a property.
2. Do one of the following:
   - Right-click the header area, select **Column Properties**, and select the property.
   - Right-click the header area, select **Column Info**, and select the property from the Column Properties menu.
   - Select **Cols > Column Info** and select the property from the Column Properties menu.
3. In the column property panel that appears, specify values and select options as appropriate.
4. Click **Apply** to add the column property or click **OK** to add the column property and close the column properties window.

Tip: A column might already contain a property that you want to apply to other columns. Use the Standardize Attributes command to apply that property to other columns. See *Using JMP*.

Viewing a Column Property

To view the properties assigned to a specific column, in the columns panel, click the column property asterisk icon ✱. Click a property to see its settings or to edit it. Figure A.2 shows the column properties assigned to Stretch in the Bounce Data.jmp sample data table, located in the Design Experiment folder.

Figure A.2 Asterisk Icon for Stretch Revealing Two Column Properties



	Pattern	Silica	Sulfur	Silane	Stretch
1	0−−	1.2	1.8	40	422
2	0++	1.2	2.8	60	392
3	−−0	0.7	1.8	50	570
4	++0	1.7	2.8	50	433
5	000	1.2	2.3	50	398
6	000	1.2	2.3	50	394
7	−+0	0.7	2.8	50	285
8	+−0	1.7	1.8	50	260
9	0+−	1.2	2.8	40	278
10	0−+	1.2	1.8	60	351
11	[illegible]	0.7	2.3	40	451
12	[illegible]	1.2	2.3	50	396
13	+0−	1.7	2.3	40	372
14	−0+	0.7	2.3	60	474
15	+0+	1.7	2.3	60	394

You can also view column properties by accessing the Column Properties list in the Column Info menu. Select the column or columns whose column properties you want to view and do one of the following:

- Right-click the header area, select **Column Info**, and select the property from the Column Properties list.
- Select **Cols > Column Info** and select the property from the Column Properties list.

Response Limits

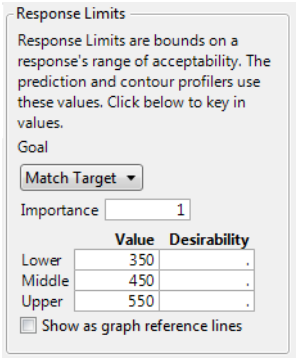
Using the Response Limits column property, you can specify the following:

- bounds on the range of variation for a response
- a desirability goal
- a measure of the importance of the response
- desirability values

The Response Limits column property defines a desirability function for the response. The Profiler and Contour Profiler use desirability functions to find optimal factor settings. See *Profilers*.

Figure A.3 shows the Response Limits panel in the Column Info window for the response Stretch in the Bounce Data.jmp sample data table, found in the Design Experiment folder.

Figure A.3 Example of the Response Limits Panel



The Response Limits panel consists of the following areas:

Goal Select your response goal from the menu. Available goals are Maximize, Match Target, Minimize, and None. JMP defines a desirability function for the response to match the selected goal. If you specify limits, the desirability function is defined using these limits. If you do not specify limits, JMP bases the desirability function on conservative limit values derived from the distribution of the response. If None is selected as the goal, then all response values are considered equally desirable. See also "Responses".

Importance Enter a relative weighting for each of several responses in computing the overall desirability function. The Importance value can be any positive number. When no Importance value is specified, JMP treats all responses in a given analysis as having equal Importance values. If there is only one response, it receives Importance 1.

Value Specify Lower and Upper limits and a Middle value for your response. JMP uses these values to construct a desirability function for the response. If you do not specify limits, JMP bases the desirability function on conservative limit values. If your goal is Match Target and no Middle value is specified, then the target is defined to be the midpoint of the Lower and Upper limits.

Desirability Specify values that reflect the desirability of your Lower, Middle, and Upper values. Desirability values should be between 0 and 1. If you do not specify Desirability values, JMP assigns values in accordance with the selected Goal.

Show as Graph Reference Lines Shows horizontal reference lines for the Lower, Middle, and Upper values in the Actual by Predicted Plot and the Prediction Profiler. This option applies only if limits are specified.

Response Limits Example

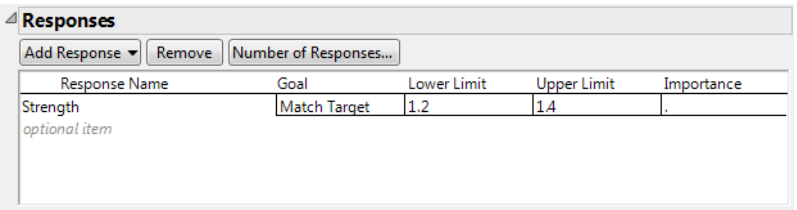
The Coffee Data.jmp sample data table (located in the Design Experiment folder) contains the results of an experiment that was performed to optimize the Strength of coffee. For a complete description of the experimental design and analysis, see "The Coffee Strength Experiment".

Your goal is to find factor settings that enable you to brew coffee with a target strength of 1.3, which is considered to be the most desirable value. Values less than 1.2 and greater than 1.4 are completely undesirable. The desirability of values between 1.2 and 1.4 decreases as their distance from 1.3 increases.

1. Select **Help > Sample Data Library** and open Design Experiment/Coffee Data.jmp.
2. In the Table panel, click the green triangle next to the **DOE Dialog** script.

   The DOE Dialog script re-creates the Custom Design dialog that was used to create the experimental design in Coffee Data.jmp.
3. Open the Responses outline.

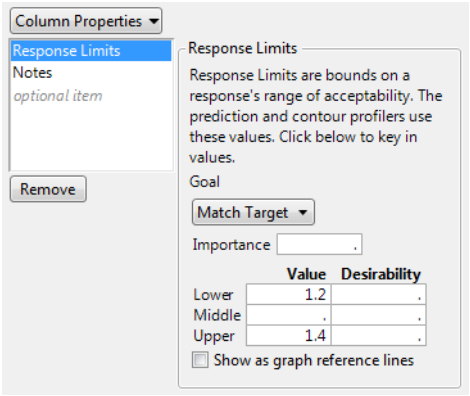
Figure A.4 Responses Outline in Custom Design Window



When you designed this experiment, you specified a response Goal of Match Target with a Lower Limit of 1.2 and an Upper Limit of 1.4. Since there is only one response, you did not specify a value for Importance, because it is 1 by default. When you constructed the design table, JMP assigned the Response Limits column property to Strength.

4. Close the Custom Design window.
5. In the Coffee Data.jmp sample data table, select the Strength column and select **Cols > Column Info**.
6. Select Response Limits in the Column Properties list.

Figure A.5 Response Limits Column Property for Strength



Notice the following:

- The Goal is set to **Match Target**.
- Importance is missing. When Importance is missing, JMP treats all responses in a given analysis as having equal Importance values. So JMP assigns Strength an Importance value of one.
- The Lower limit is 1.2.
- The Upper limit is 1.4.
- No Middle value is specified.

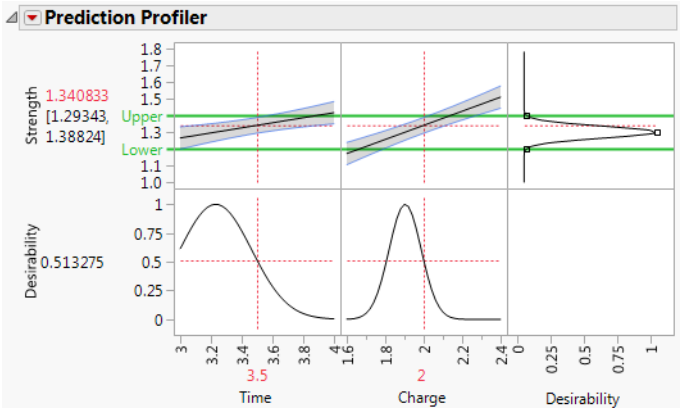
  Because no Middle value is specified, the target is defined to be the midpoint of the Lower and Upper limits, which is 1.3.
- No Desirability values are specified.

7. Select the **Show as Graph Reference Lines** option.

   This option shows horizontal reference lines for the Lower, Middle, and Upper values in the Actual by Predicted Plot and the Prediction Profiler.
8. Click **OK**.
9. In the Coffee Data.jmp data table, click the green triangle next to the **Reduced Model** script.
10. Click **Run**.

    The Prediction Profiler appears at the bottom of the report.

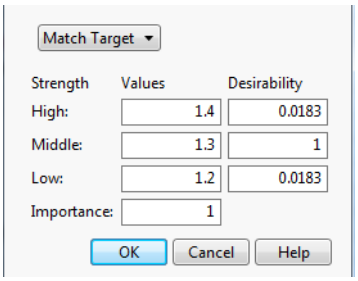
Figure A.6 Profiler Showing Desirability Function for Strength



The desirability function for Strength appears in the plot at the right above Desirability. This plot appears because the data table contains a Response Limits column property for Strength. The Prediction Profiler also shows reference lines for the Lower and Upper limits for Strength.

11. Press Ctrl and click the Strength plot for Desirability.

Figure A.7 Response Goal Window for Strength



Notice the following:

- JMP determines the Middle value to be the midpoint of the High and Low limits that you specified in the Response Limits column property.
- Because the Goal is set to Match Target, JMP sets the Desirability for the Middle value to 1.
- JMP sets the Desirability for the High and Low values to very small numbers, 0.0183.
- The Desirability plot in Figure A.6 shows how JMP uses the Desirability values shown in Figure A.7. The Desirability function for Strength is essentially 0 beyond the Low and High values and it increases to 1 gradually as Strength approaches the target of 1.3. The Importance value is set to 1 since there is only one response in the model.

12. Click **Cancel** to exit the window.

13. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

    The settings for Time and Charge are updated to show settings for the factors that maximize the desirability function for Strength. However, many other settings also maximize the desirability function. See *Profilers* for information about how to identify other settings that maximize the desirability function.

Editing Response Limits

In the Vinyl Data.jmp sample data table, a Response Limits column property is already assigned to the response thickness. The property has a goal of maximizing thickness. Suppose that instead of maximizing thickness, you want the sheets of vinyl to have a thickness between 6 and 10, with a target thickness of 8.5.

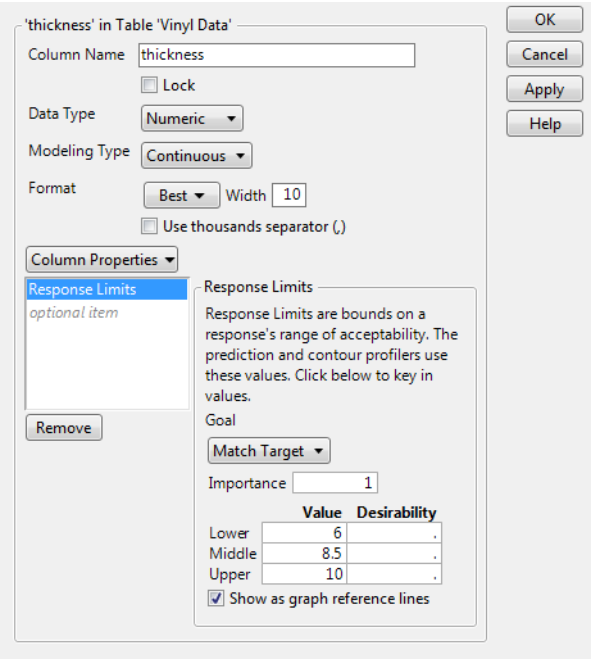
1. Select **Help > Sample Data Library** and open Design Experiment/Vinyl Data.jmp.
2. Select the thickness column and select **Cols > Column Info.**

   The Response Limits property appears in the Column Properties list as the only property assigned to thickness. The Response Limits panel appears to the right of the list.
3. Click **Maximize** and select **Match Target**.
4. Type 1 for the **Importance** value.
5. Under Value, type 6 for the Lower value, 8.5 for the Middle value, and 10 for the Upper value.

   This is an example of asymmetric response limits. Values of thickness as small as 6 or as large as 10 are acceptable. However, the target for thickness is 8.5.
6. Select **Show as Graph Reference Lines**.

   This option shows horizontal reference lines for the Lower, Middle, and Upper values in the Actual by Predicted Plot and the Prediction Profiler.

Figure A.8 Completed Response Limits Panel

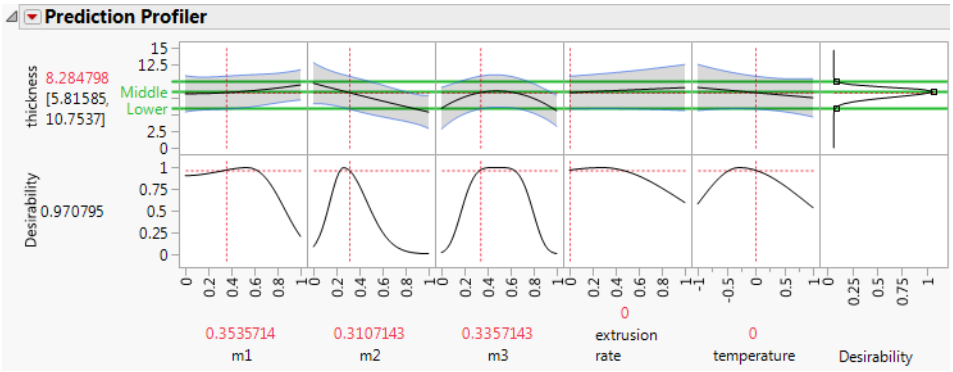


7. Click **OK**.
8. In the Vinyl Data.jmp data table, click the green triangle next to the **Model** script.

   Note that m1, m2, and m3 are mixture factors. Also, the design involves a random Whole Plots factor. Because of this, the default Method is REML (Recommended).
9. Click **Run**.
10. Click the Response thickness red triangle and select **Row Diagnostics > Plot Actual by Predicted**.

    The reference lines for the Lower, Middle, and Upper limits appear on the Actual by Predicted Plot.
11. Click the Response thickness red triangle and select **Factor Profiling > Profiler**.

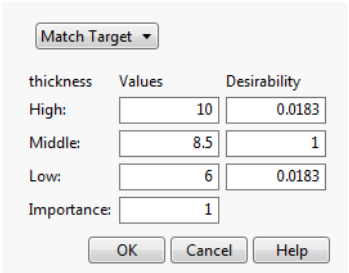
Figure A.9 Prediction Profiler Showing Asymmetric Desirability Function



The plot at the right above Desirability shows the desirability function that JMP has constructed for thickness. The desirability is 1 at the Middle limit of 8.5. The desirability is essentially 0 for thickness values below 6 and above 10.

12. Press Ctrl and click the thickness plot for Desirability.

Figure A.10 Response Goal Window for Thickness



This window shows your settings for the High, Middle, and Low Values. It also shows the Desirability values that JMP assigns, based on your goal of Match Target. The Desirability function shown in Figure A.9 is a continuous curve that matches the Desirability settings in Figure A.10 at the High, Middle, and Low Values. At other values, the Desirability function assigns desirabilities that are consistent with the selected goal.

13. Click **Cancel**.
14. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

    The settings for the factors are updated to show values that maximize the desirability function for thickness. Keep in mind that many other settings also maximize the desirability function. The predicted response at these optimal settings is 8.5. Recall that you set 8.5 as the target setting, with limits of 6 and 10.
15. Close the Vinyl Data.jmp sample data table without saving the changes.

JMP PRO Detection Limits

The Detection Limits column property defines bounds beyond which the response cannot be measured. You can use these limits to specify a censored response in the Generalized Regression platform.

JMP PRO Functional Response

The FDE X column property specifies a column as a functional response. Multiple Y columns with the FDE column property can be used in the Functional Data Explorer platform with the DOE factors as supplementary factors.

Design Role

Factors in designed experiments, as well as in more general models, can behave in various ways. JMP uses the design role column property to describe these behaviors. These are the possible design roles:

- Continuous
- Discrete Numeric
- Categorical
- Blocking
- Covariate
- Mixture
- Constant
- Uncontrolled
- Random Block
- Signal
- Noise

In many of the JMP DOE platforms, you can specify factors with different design roles. In some platforms, your design requirements cause JMP to define factors. For example, Whole Plots and Subplots are factors that JMP creates when you specify very-hard-to-change and hard-to-change factors. In platforms where various design roles can occur, when JMP creates the design table for your design, each factor is assigned the Design Role column property.

- For descriptions of the design roles other than Random Block, Signal, and Noise, see "Factor Types".
- For a description of the Random Block design role, see "Changes and Random Blocks".
- For descriptions of the Signal and Noise design roles, see "Factors".

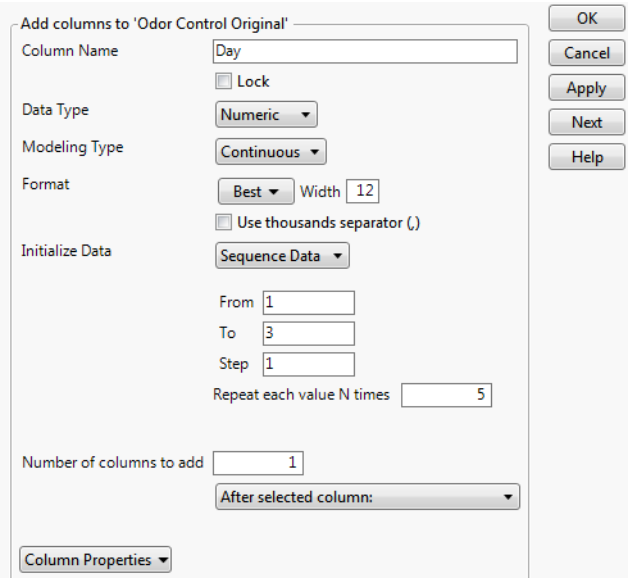
Design Role Example

The experiment in the Odor Control Original.jmp sample data table studies the effect of three factors on odor. You designed the 15-run experiment and it was conducted. However, when the results were reported to you, you learned that the experiment was conducted over three days. The first 5 runs were conducted on Day 1, the second 5 runs on Day 2, and the remaining 5 runs on Day 3.

Since variations in temperature and humidity might have an effect on the response, you want to include Day as a random blocking factor. It is easy to add a column for Day to the design table. But you want to use the Evaluate Design platform to compare the design with the unexpected block to your original design. You also want the ability to use the Augment Design platform in case you need to augment the design. To use the Evaluate Design and Augment Design platforms, you need to add the Design Role column property to your new Day column.

1. Select **Help > Sample Data Library** and open Odor Control Original.jmp.
2. Select the first column, Run.
3. Select **Cols > New Columns.**
4. Next to **Column Name**, type Day.
5. From the **Initialize Data** list, select **Sequence Data**.
6. Enter the following:
   - 1 for **From**
   - 3 for **To**
   - 1 for **Step**
   - 5 for **Repeat each value N times**
7. Next to **Number of columns to add**, type 1.

Figure A.11 Completed New Columns Window



8. Click **OK**.

   The Day column is added as the second column in the data table.
9. Select the Day column.
10. Select **Cols > Column Info**.
11. From the Column Properties list, select **Design Role**.
12. In the Design Role panel, click **Continuous** and select **Random Block**.
13. Click **OK**.

    In the columns panel, an asterisk appears next to Day.
14. Click the asterisk next to Day to verify that the Design Role column property has been assigned.
15. Close the Odor Control Original.jmp sample data table without saving the changes.

Coding

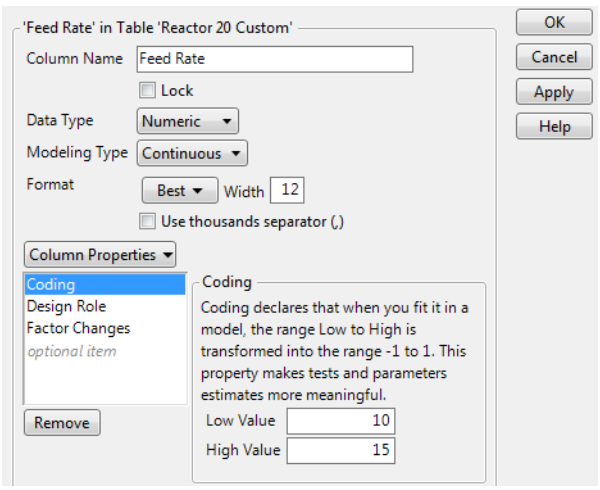
The Coding column property applies only to columns with a numeric data type. It applies a linear transformation to the data in the column. In the Coding column property window, you specify a Low Value and a High Value. The Low Value and High Value in your original data are transformed to –1 and +1. JMP uses the transformed data values whenever the column is entered as a model effect in the Fit Model platform.

The coding property is useful for the following reasons:

- Coded predictors lead to parameter estimates that are more easily interpreted and compared.
- Coded predictors help reduce multicollinearity in models with interaction and higher-order terms.

When any DOE platform other than Accelerated Life Test Design creates a design, JMP defines a Coding column property for each non-mixture factor with a numeric data type. Figure A.12 shows the Coding column property panel for the column Feed Rate in the Reactor 20 Custom.jmp sample data table, found in the Design Experiment folder.

Figure A.12 Coding Property Panel for Feed Rate



Low and High Values

When the Coding property is applied as part of design construction in a DOE platform, the Low Value is initially set to the minimum setting of the factor and the High Value is initially set to the maximum setting.

When you apply the Coding property to a column that does not contain that property, JMP inserts the minimum value as the Low Value and the maximum value as the High Value. You can change these values as needed.

Caution: After the Coding column property is assigned to a column, JMP does not automatically update it when you make changes to the values in the column. If you change the values in a column that has a Coding column property, review the High Value and Low Value to ensure that they are still appropriate.

The Coding column property centers each value in a column by subtracting the midpoint of the High Value and Low Value. It then divides by half the range. Suppose that H is the High Value and L is the Low Value. Then every X in the column is transformed to the following:

$$\frac{X-(H+L)/2}{(H-L)/2}$$

For each factor, the transformed values have a midpoint equal to 0 and range from -1 to +1.

Coding Column Property and Center Polynomials

The Center Polynomials option is located in the Fit Model launch window, within the Model Specification red triangle menu. Center Polynomials centers a continuous column involved in a polynomial term by subtracting the mean of each value in the column. See *Fitting Linear Models*.

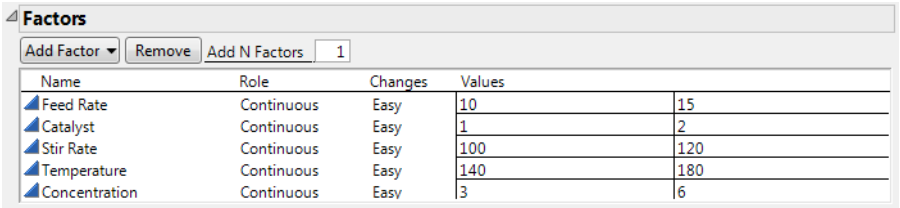
If the Coding column property is assigned to a column, then the Center Polynomials option has no effect on that column. In a polynomial term involving that column, the values are centered and scaled as specified by their Coding property. Suppose that another column in the model does not have the Coding property and that you select Center Polynomials. Then that column is centered by its mean in any polynomial term where it appears.

Coding Example

The Reactor 20 Custom.jmp sample data table contains data from a 20-run design that was constructed using the Custom Design platform. The experiment investigates the effects of five factors on a yield response (Percent Reacted) for a chemical process.

1. Select **Help > Sample Data Library** and open Design Experiment/Reactor 20 Custom.jmp.
2. In the Table panel, click the green triangle next to the **DOE Dialog** script.
3. Open the **Factors** outline.

Figure A.13 Factors Outline for Design Used in Reactor 20 Custom.jmp

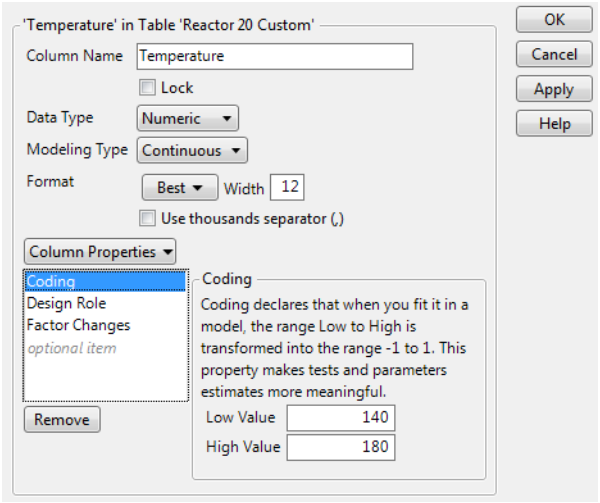


Notice that the settings for Temperature range from 140 to 180. When the design was generated, the Coding column property was assigned to Temperature. The Low Value is set to 140 and the High Value is set to 180.

4. Close the Custom Design window.
5. In the Reactor 20 Custom.jmp sample data table, click the asterisk next to Temperature in the columns panel and select **Coding**.

   The Column Info window appears and shows the Coding column property panel. You can see that JMP added the column property, specifying the Low Value and High Value, when it constructed the design table. In fact, by repeating this step, you can verify that JMP added the Coding property for all five factors.

Figure A.14 Coding Panel for Temperature

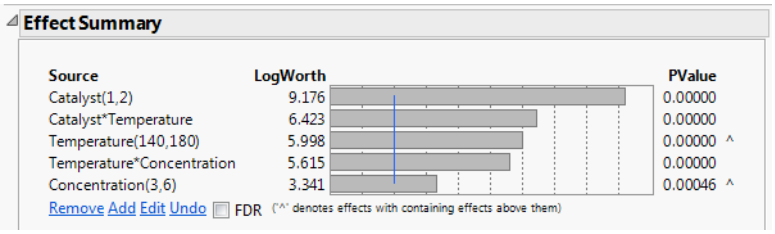


6. Click **Cancel** to close the Column Info window.
7. In the Reactor 20 Custom.jmp sample data table, click the green triangle next to the **Reduced Model** script.

   This script fits a model that contains only the five effects determined to be significant based on an analysis of the full model.

8. Click **Run**.

Figure A.15 Effect Summary Report for Reduced Model



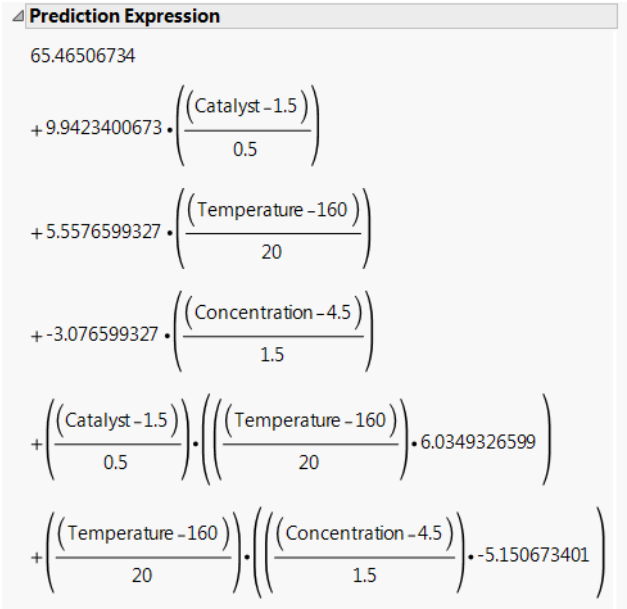
In the Source list, the High and Low values used in the Coding column property appear in parentheses to the right of the main effects, Catalyst, Temperature, and Concentration. The ranges imposed by the Coding property are not shown for the interaction effects.

Tip: Notice the "^" symbols to the right of the PValues for Temperature and Concentration. These symbols indicate that these main effects are components of interaction effects with smaller *p*-values. If an interaction effect is included in the model, then the principle of effect heredity requires that all component effects are also in the model. See "Effect Heredity".

9. Click the Response Percent Reacted red triangle and select **Estimates > Show Prediction Expression**.

   Look at the Prediction Expression outline to see how coding affects the prediction formula.

Figure A.16 Prediction Expression for Reduced Model



Each factor is transformed as specified by the Coding column property. For example, for Temperature, notice the following:

- The Low Value in the Coding property was set to 140. The Temperature value of 140 is transformed to -1.
- The High Value in the Coding property was set to 180. The Temperature value of 180 is transformed to +1.
- The midpoint of the Low and High values is 160. The Temperature value of 160 is transformed to 0.

The transformed values help you compare the effects. The estimated coefficient for Catalyst is 9.942 and the estimated coefficient for Concentration is -3.077. It follows that the predicted effect of Catalyst on Percent Reacted is more than three times as large as the effect of Concentration on Percent Reacted. Also, the coefficients indicate that predicted Percent Reacted increases as Catalyst increases and decreases as Concentration increases.

The transformed values help you interpret the coefficients:

- When all factors are at their midpoints, their transformed values are 0. The predicted Percent Reacted is the intercept, which is 65.465.
- When Catalyst and Concentration are at their midpoints, a 20 unit increase in Temperature increases the Percent Reacted by 5.558 units.
- Suppose that Concentration is at its midpoint, so that its transformed value is 0:

- When Catalyst is at its midpoint, a 20 unit increase in Temperature increases the Percent Reacted by 5.558 units.
- When Catalyst is at its high setting, a 20 unit increase in Temperature increases the Percent Reacted by 5.558 + 6.035 = 11.593 units.

It follows that the coefficient of the interaction term, 6.035, is the increase in the slope of the model for predicted Percent Reacted for a 0.5 unit change in Catalyst.

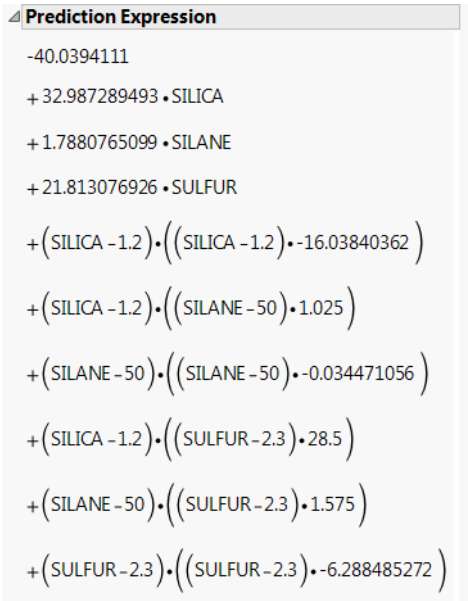
Assigning Coding

The experimental data in the Tiretread.jmp sample data table results from an experiment to study the effects of SILICA, SILANE, and SULFUR on four measures of tire tread performance. In this example, you will consider only one of the responses, ABRASION.

You will first fit a model using the uncoded factors. Then you will assign the coding property to the factors and rerun the model to obtain meaningful parameter estimates.

1. Select **Help > Sample Data Library** and open Tiretread.jmp.
2. Select **Analyze > Fit Model**.
3. Select ABRASION and click **Y**.
4. Select SILICA, SILANE, and SULFUR and click **Macros > Response Surface**.
5. Check **Keep dialog open**.
6. Click **Run**.
7. Click the Response ABRASION red triangle and select **Estimates > Show Prediction Expression**.

Figure A.17 Prediction Expression for Model with Uncoded Factors



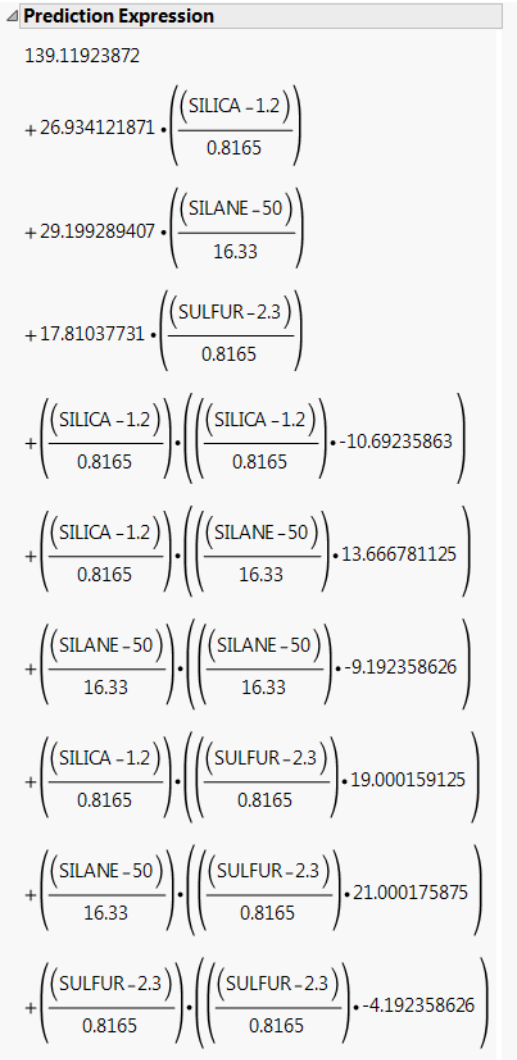
The coefficients do not help you compare effect sizes. The sizes of the coefficients do not reflect the impact of the effects on ABRASION over the range of their settings. Also, the coefficients are not easily interpreted. For example, the coefficients do not facilitate your understanding of the predicted response when SILICA is at the midpoint of its range.

Apply the Coding column property to the three factors to see how coding makes the coefficients more meaningful.

8. In the Tiretread.jmp data table, select SILICA, SILANE, and SULFUR in the Columns panel. Click the Columns red triangle and select **Standardize Attributes.**.
9. Select **Column Properties > Coding** in the Standardize Properties panel.
10. Click **OK**.

    An asterisk appears in the Columns panel next to SILICA, SILANE, and SULFUR indicating that these have been assigned a column property.
11. In the Fit Model window, click **Run**.
12. Click the Response ABRASTION red triangle and select **Estimates > Show Prediction Expression**.

Figure A.18 Prediction Expression for Model with Coded Factors



The coefficients for the coded factors enable you to compare effect sizes. SILANE has the largest effect on ABRASION over the range of design settings. The effects of SILICA and the SILANE*SULFUR interaction are large as well.

The coefficients for the coded factors are also more easily interpreted. For example, when all factors are at the center of their ranges, the predicted value of ABRASION is the intercept, 139.12.

13. Close the Tiretread.jmp sample data table without saving the changes.

Mixture

The Mixture column property is useful when a column in a data table represents a component of a mixture. The components of a mixture are constrained to sum to a constant. Because of this, they differ from non-mixture factors. The Mixture column property serves two purposes:

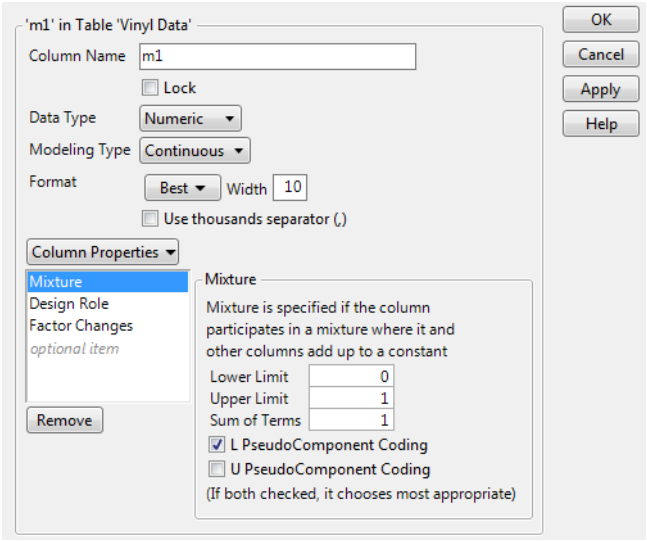
- It identifies a column as a mixture component.

  If you add a column with the Mixture column property as a model effect in the Analyze > Fit Model window, JMP automatically generates a no-intercept model.
- It defines the coding for a mixture component.

  Coding for mixture components differs from that for non-mixture factors. However, as with non-mixture factors, a benefit of coding for mixture factors is that it helps you interpret parameter estimates. See "PseudoComponent Coding".

Figure A.19 shows the Mixture column property panel for the factor m1 in the Vinyl Data.jmp sample data table, found in the Design Experiment folder.

Figure A.19 Mixture Column Property Panel



In the Mixture column property panel, you can specify the following:

Lower Limit Specifies the low value used in PseudoComponent Coding. When the Mixture property is applied as part of design construction in a DOE platform, the Lower Limit is set to the minimum setting of the factor. When you apply the Mixture property to a column that does not contain that property, JMP inserts the minimum value as the Lower Limit. You can change this value as needed.

Upper Limit Specifies the high value used in PseudoComponent Coding. When the Mixture property is applied as part of design construction in a DOE platform, the Upper Limit is set to the maximum setting of the factor. When you apply the Mixture property to a column that does not contain that property, JMP inserts the maximum value as the Upper Limit. You can change this value as needed.

Sum of Terms Specifies the sum of the mixture components. When you apply the Mixture property to a column that does not contain that property, JMP inserts a default value of 1 for the Sum of Terms.

L PseudoComponent Coding Transforms data values so that the Lower Limit corresponds to 0.

U PseudoComponent Coding Transforms data values so that the Upper Limit corresponds to 0.

PseudoComponent Coding

A pseudo-component is a linear transformation. Let S denote the sum of the mixture components. Suppose that i columns $X_1, X_2, \ldots, X_q$ have been assigned the Mixture column property. Suppose that the columns and effects constructed from these columns are entered as effects in the Fit Model window.

Define the following:

$$L = \sum_{i=1}^{q} L_i \text{ , where } L_i \text{ is the Lower Limit for } X_i$$

$$U = \sum_{i=1}^{q} U_i \text{ , where } U_i \text{ is the Upper Limit for } X_i$$

Let x_i denote a value of the column X_i. The L PseudoComponent at x_i is defined as follows:

$$x_i^L = (X_i - L_i)/(S - L)$$

The U PseudoComponent at x_i is defined as follows:

$$x_i^U = (U_i - X_i)/(U - S)$$

If you select both **L PseudoComponent Coding** and **U PseudoComponent Coding**, the Fit Model platform uses the L coding if $(S - L) < (U - S)$. Otherwise, the U coding is used.

In Fit Model, mixture factors are transformed using pseudo-components before computing parameter estimates. This helps make parameter estimates more meaningful. In reports dealing with parameter estimates, the mixture main effects are given by the coding transformation. Other reports, such as the profilers, are based on the uncoded values.

Mixture Example

The data in the Donev Mixture Data.jmp sample data table, found in the Design Experiment folder, are based on an example from Atkinson and Donev (1992). The design includes three mixture factors and one non-mixture factor:

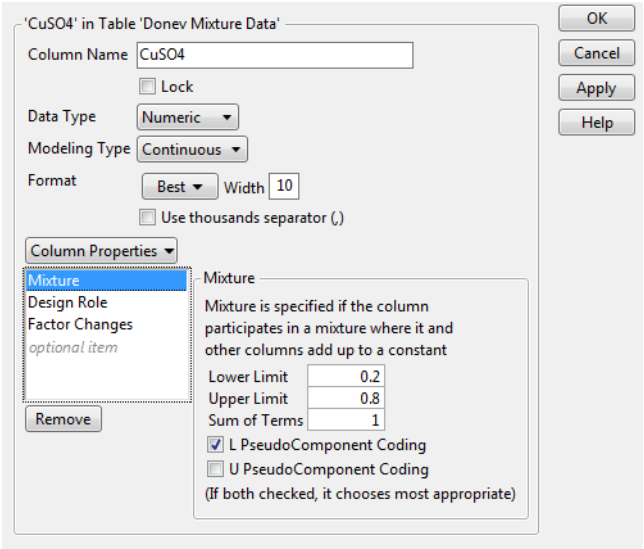
- The response is the electromagnetic Damping of an acrylonitrile powder.
- The three mixture ingredients are copper sulphate (CuSO4), sodium thiosulphate (Na2S2O3), and Glyoxal.
- The non-mixture environmental factor of interest is the Wavelength of light.

Though Wavelength is theoretically continuous, the researchers were interested only in predictions at three discrete wavelengths. As a result, Wavelength is treated as a categorical factor with three levels.

For more information about using Custom Design to construct a design for this situation, see "Mixture Experiments".

1. Select **Help > Sample Data Library** and open Design Experiment/Donev Mixture Data.jmp.
2. Click the asterisk next to CuSO4 in the columns panel and select **Mixture**.

Figure A.20 Mixture Column Property Panel for CuSO4



Notice the following:

- The Lower Limit is 0.2, the minimum design setting for CuSO4.
- The Upper Limit is 0.8, the maximum design setting for CuSO4.
- The Sum of Terms is set to 1. This is the sum of the three mixture factors.
- The **L PseudoComponent Coding** option is selected. See "PseudoComponent Coding".

3. Click **Cancel**.
4. Click the asterisk next to Glyoxal in the columns panel and select **Mixture**.

   For this factor, note the following:

   - The Lower Limit is 0, the minimum design setting for Glyoxal.
   - The Upper Limit is 0.6, the maximum design setting for Glyoxal.
5. Click **Cancel**.
6. In the Donev Mixture Data.jmp data table, click the green triangle next to the **Model** script.

   The model contains the main effects of the mixture factors and two-way interactions for all four factors.
7. Click **Run**.

   In the Parameter Estimates report, the mixture factors appear in their pseudo-component coded form. When the mixture factors appear in interactions, they are not denoted in coded form. Nevertheless, the model fitting is based on the pseudo-components. The first three terms in the Parameter Estimates report (Figure A.21), show the coded form for the mixture factors.

Figure A.21 Parameter Estimates Report

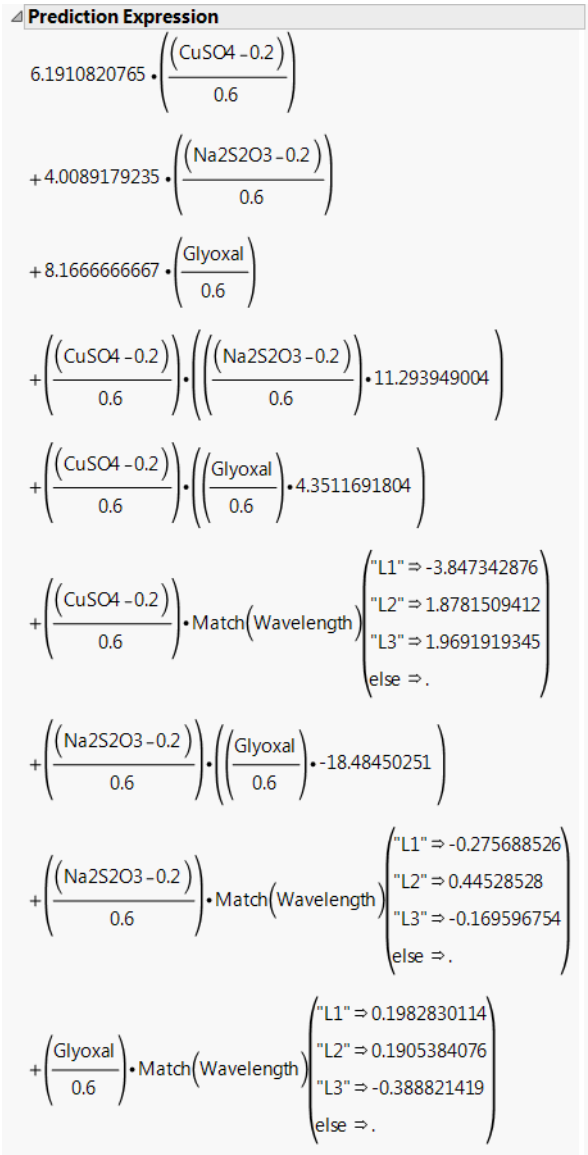
Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
(CuSO4-0.2)/0.6	6.1910821	0.918805	6.74	0.0005*
(Na2S2O3-0.2)/0.6	4.0089179	0.918805	4.36	0.0048*
Glyoxal/0.6	8.1666667	0.921638	8.86	0.0001*
CuSO4*Na2S2O3	11.293949	4.728922	2.39	0.0542
CuSO4*Glyoxal	4.3511692	4.512775	0.96	0.3722
CuSO4*Wavelength[L1]	-3.847343	1.113399	-3.46	0.0135*
CuSO4*Wavelength[L2]	1.8781509	1.079113	1.74	0.1324
Na2S2O3*Glyoxal	-18.4845	4.512775	-4.10	0.0064*
Na2S2O3*Wavelength[L1]	-0.275689	1.064004	-0.26	0.8042
Na2S2O3*Wavelength[L2]	0.4452853	1.103681	0.40	0.7006
Glyoxal*Wavelength[L1]	0.198283	1.090627	0.18	0.8617
Glyoxal*Wavelength[L2]	0.1905384	1.090647	0.17	0.8671

8. Click the Response Damping red triangle and select **Estimates > Show Prediction Expression**.

   The Prediction Expression report shows the model that was fit. Note that the mixture factors are transformed using the L PseudoComponent coding.

Figure A.22 Prediction Expression for Damping Model



Suppose that you are interested in predictions at Wavelength L2. Suppose also that Na2S2O3 and Glyoxal are set to their low values, 0.2 and 0 respectively, and that CuSO4 is set to its high value, 0.8. In this case, the predicted Damping equals the parameter estimate for CuSO4 (6.191) plus the parameter estimate for CuSO4*Wavelength[L2] (1.878). You can verify this in the Prediction Profiler.

9. Click the Response Damping red triangle and select **Save Columns > Save Coding Table**.

Figure A.23 First Three Columns of Coding Table Showing Coded Mixture Factors

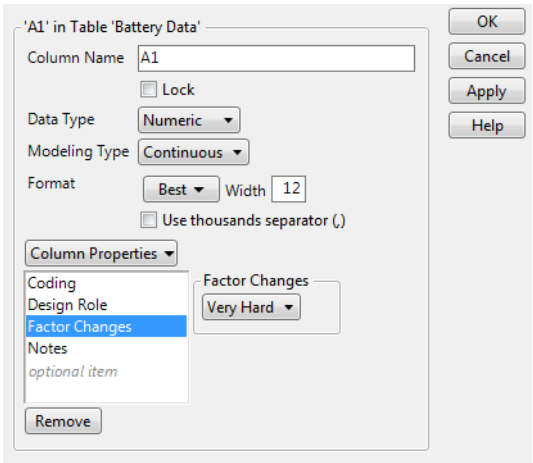
	(CuSO4-0.2)/0.6	(Na2S2O3-0.2)/0.6	Glyoxal/0.6
1	1	0	0
2	0.4	0.6	0
3	0	1	0
4	0.5	0	0.5
5	0	0.5	0.5
6	0	0	1
7	1	0	0
8	0.6	0.4	0
9	0	1	0
10	0.5	0	0.5
11	0	0.5	0.5
12	0	0	1
13	1	0	0
14	0.4	0.6	0
15	0	1	0
16	0.5	0	0.5
17	0	0.5	0.5
18	0	0	1

For this particular design, the L Pseudo Component coding transforms the mixture factors to range between 0 and 1. Note that this does not happen in general

Factor Changes

The Factor Changes column property indicates how difficult it is to change factor settings in a designed experiment. The possible specifications for Factor Changes are Easy, Hard, and Very Hard. For example, Figure A.24 shows the Factor Changes column property panel for the factor A1 in the Battery Data.jmp sample data table, located in the Design Experiment folder.

Figure A.24 Factor Changes Column Property Panel



- When a design contains factors that are hard-to-change and very-hard-to-change, it must also include a subplot and a whole plot factor:
  - The levels of the whole plot factor define the groups of runs for which the levels of the very-hard-to-change factors are held constant.
  - The levels of the subplot factor define the groups of runs for which the levels of the hard-to-change factors are held constant.
- When a design contains only factors that are hard-to-change, but no factors that are very-hard-to-change, it should include a whole plot factor. The levels of the whole plot factor define the groups of runs for which the levels of the hard-to-change factors are held constant. See "Changes and Random Blocks".

Augment and Evaluate Design

For the Evaluate Design and Augment Design platforms, the Factor Changes column property identifies factors with Changes specified as Hard or Very Hard. However, these platforms also require that the whole plot and subplot factors be entered as model effects in the launch windows. This is because the whole plot and subplot factors are part of the design structure.

Custom Design

The Custom Design platform enables you to create designs where all factor changes are Easy. You can also construct split-plot, split-split plot, or two-way split-plot (strip-plot) designs. When constructing these designs, you need to identify the factors whose values are hard-to-change or very-hard-to-change. In the Factors outline, you can identify factors as having Changes that are Easy, Hard, or Very Hard. When the Custom Design platform constructs the design table, the Factor Changes property is assigned to every factor that appears in the Factors outline.

The Custom Design platform is the only platform that constructs designs for factors with Changes that are Hard or Very Hard. Other DOE platforms also assign the Factor Changes column property to factors that they construct, but the value of the column property is set to Easy for their factors.

If you Load Factors in the Custom Design window using a table of factors, you can assign the Factor Changes column property to columns in that table. When you Load Factors using that table, your Factor Changes specifications appear in the Factors outline.

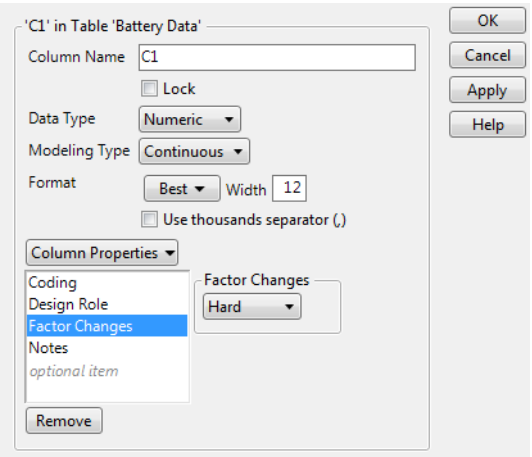
Factor Changes Example

The Battery Data.jmp sample data table, found in the Design Experiment folder, contains data from an experiment that studies the open current voltage of batteries (OCV). The design is a two-way split-plot design. For further background, see "Two-Way Split-Plot Experiment".

1. Select **Help > Sample Data Library** and open Design Experiment/Battery Data.jmp.

2. Click the asterisk to the right of the factor C1 in the columns panel.
3. Select **Factor Changes**.

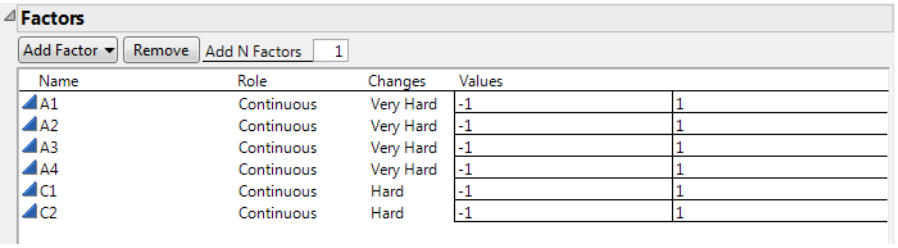
Figure A.25 Factor Changes Panel for C1



The value of Factor Changes for C1 is Hard. Figure A.24 shows that the value of Factor Changes for A1 is Very Hard.

4. Click **OK**.
5. In the data table, click the green triangle next to the **DOE Dialog** script.
6. Open the **Factors** outline.

Figure A.26 Factors Outline for Battery Experiment



Name	Role	Changes	Values	
A1	Continuous	Very Hard	-1	1
A2	Continuous	Very Hard	-1	1
A3	Continuous	Very Hard	-1	1
A4	Continuous	Very Hard	-1	1
C1	Continuous	Hard	-1	1
C2	Continuous	Hard	-1	1

The factors A1, A2, A3, and A4 have Changes set to Very Hard, and the factors C1 and C2 have Changes set to Hard. When the Custom Design platform constructs the design table, it saves these specifications to the appropriate columns as Factor Changes column properties.

In the Design outline, notice the Whole Plots and Subplots factors.

Figure A.27 Design Outline Partial View

Design

Run	Whole Plots	Subplots	A1	A2	A3	A4	C1	C2
1	1	1	-1	-1	1	-1	-1	-1
2	1	2	-1	-1	1	-1	1	-1
3	1	3	-1	-1	1	-1	-1	1
4	2	4	-1	1	1	1	1	1
5	2	5	-1	1	1	1	-1	1
6	2	6	-1	1	1	1	1	-1
7	3	7	1	1	1	-1	-1	1
8	3	8	1	1	1	-1	1	1
9	3	9	1	1	1	-1	-1	-1
10	4	10	1	-1	1	1	1	1
11	4	11	1	-1	1	1	1	-1
12	4	12	1	-1	1	1	-1	-1

To account for the factor changes that are Hard and Very Hard, two factors are created by the Custom Design platform. The Whole Plots factor groups the runs where the Very Hard factor levels are constant and the Subplots factor groups the runs where the Hard factors levels are constant. These factors need to be included as model effects when you enter columns with the Factor Changes column property in the Evaluate Design and Augment Design platforms.

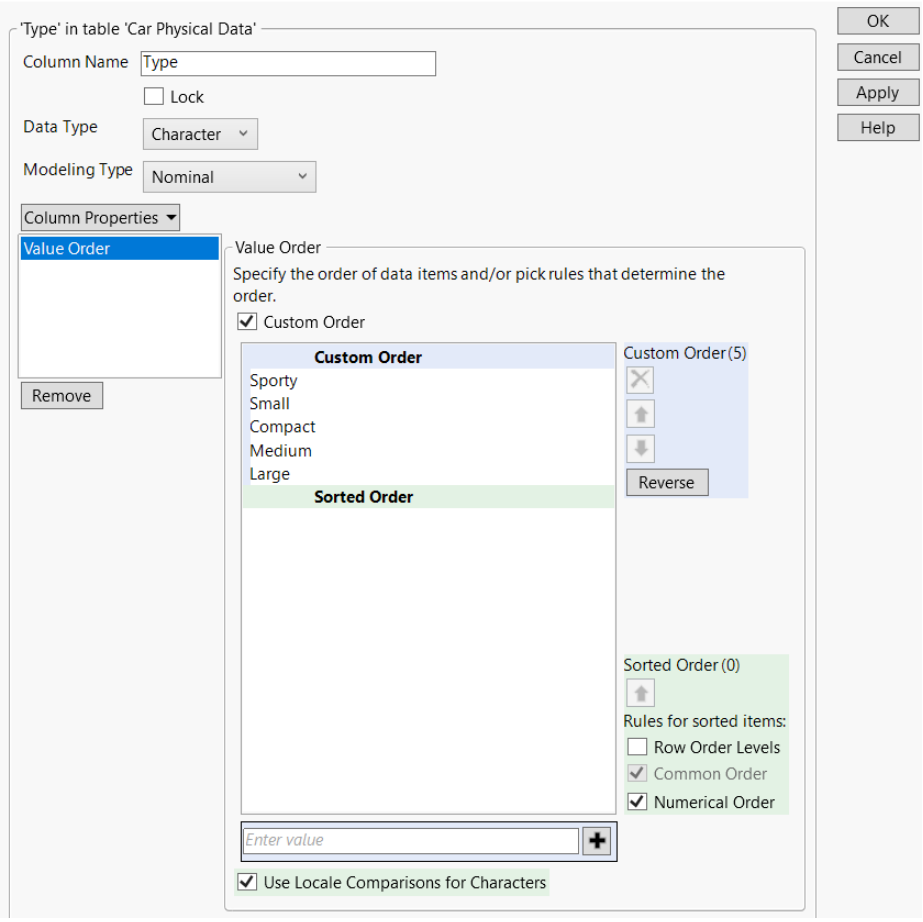
Value Order

The Value Order column property assigns an order to the values in a column. That order is then used in plots and analyses. You can specify the order in which you want values to appear in reports.

Note: For certain values that have a natural ordering, such as days of the week, JMP automatically orders these in the appropriate way in reports. See *Using JMP*.

Figure A.28 shows the Value Order panel for the Type column in the Car Physical Data.jmp sample data table. Reports that involve the values of Type place these levels in the order Sporty, Small, Compact, Medium, and Large. Use the buttons to the right of the Value Order list to specify your desired ordering for the values.

Figure A.28 Value Order Column Property for Type



In designs created using most DOE platforms, categorical factors, including the constructed factors Whole Plots and Subplots, and blocking factors are assigned the Value Order property. This property orders the levels according to the order in which they appear in the Factors outline. The levels of constructed factors are consecutive integers and the Value Order property specifies this natural ordering. You can modify the Value Order specification for any factor to meet your needs.

The Value Order property is not assigned by the Covering Array or Taguchi Arrays platforms. The Covering Array platform assigns the Value Labels column property. See "Value Labels".

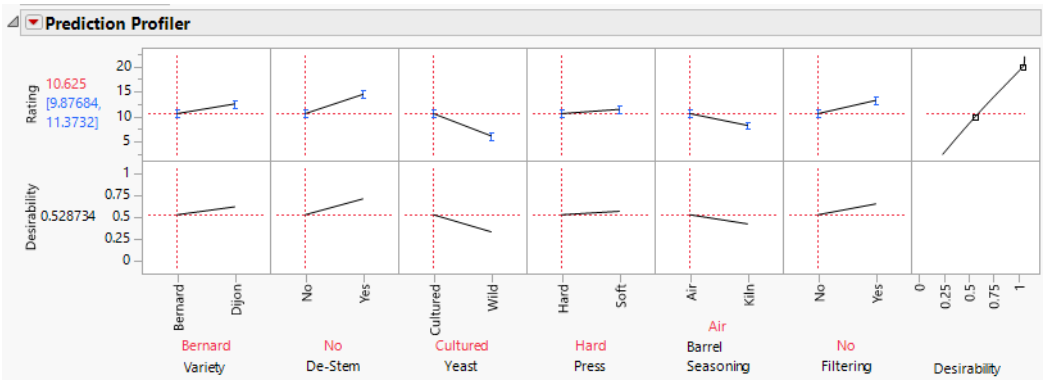
Value Order Example

Suppose that you want the values for a factor to appear in a different order in the Prediction Profiler. Consider an example of a wine tasting experiment, constructed using Custom Design. Wine is rated by five experts, each listed as a Rater in the Wine Data.jmp sample data table. Rater is a fixed blocking factor. Nine factors are studied. Rating is the response.

1. Select **Help > Sample Data Library** and open Design Experiment/Wine Data.jmp.
2. In the Table panel, click the green triangle next to the **Reduced Model** script.
3. Click **Run**.

   The Prediction Profiler appears at the bottom of the report.

Figure A.29 Profiler with Original Value Order



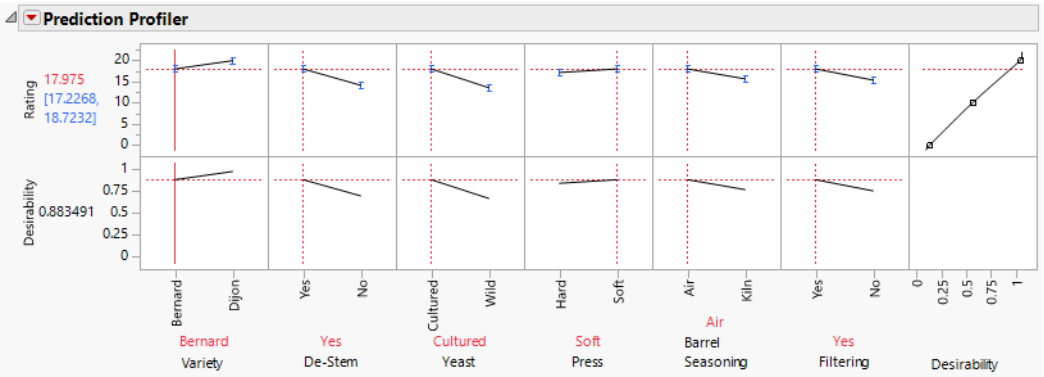
Notice that the values for De-Stem and Filtering appear in the order No followed by Yes. You want to reverse these, so that the Yes level appears first.

Tip: To arrange the profiler click the Prediction Profiler red triangle and select **Appearance > Arrange in rows**. For this example enter 7 and click **OK**.

4. Close the Response Rating report.
5. In the data table, click the asterisk next to De-Stem in the columns panel and select Value Order.
6. Click **Reverse**.
7. Click **OK**.
8. Click the asterisk next to Filtering in the columns panel and select Value Order.
9. Click **Reverse**.
10. Click **OK**.
11. Again, click the green triangle next to the **Reduced Model** script.

12. Click **Run**.

Figure A.30 Profiler with New Value Orders



The levels for De-Stem and Filtering now appear in the order Yes followed by No.

13. Close the Wine Data.jmp sample data table without saving the changes.

Assigning Value Order

Consider the Candy Bars.jmp sample data table. Of the 18 brands lists under Brand, Hershey and M&M/Mars have the largest numbers of types of candy as listed in the Name column. You want these two brands to appear first in reports.

1. Select **Help > Sample Data Library** and open Candy Bars.jmp.
2. Select the Brand column.
3. Select **Cols > Column Info.**
4. Under Column Properties, select Value Order.
5. In the Value Order list, select Hershey.
6. Click the **up arrow** five times.

   Hershey is now first in the Value Order list.
7. In the Value Order list, select M&M/Mars.
8. Click the **up arrow** seven times.

   M&M/Mars is now second in the Value Order list.
9. Click **OK**.

   An asterisk indicating the Value Order column property appears next to Brand in the columns panel. JMP now lists Hershey and M&M/Mars first in reports involving Brand.
10. Select **Analyze > Distribution**.

11. Select Calories and click **Y, Columns**.
12. Select Brand and click **By**.
13. Click **OK**.
14. Press Ctrl, click the Calories red triangle, and select **Display Options > Horizontal Layout**.

    Note that the Distribution reports for Hershey and M&M/Mars appear first among the 18 brands.
15. Close the Candy Bars.jmp sample data table without saving the changes.

Value Labels

The Value Labels column property represents values in a column with specified labels. These labels are displayed in the data table and are used in plots and reports. In the data table, you can view the original values by double-clicking within a cell. For more information about how to assign and work with the Value Labels column property, see *Using JMP*.

The Covering Arrays platform is the only DOE platform that assigns the Value Labels column property. The Covering Arrays platform saves factors to the data table with a Nominal modeling type. The underlying values are consecutive integers ranging from 1 to the number of levels that you specify in the Covering Array Factors outline. The Values that you specify in the Factors outline are the Value Labels that are assigned to the underlying integers.

Value Labels Example

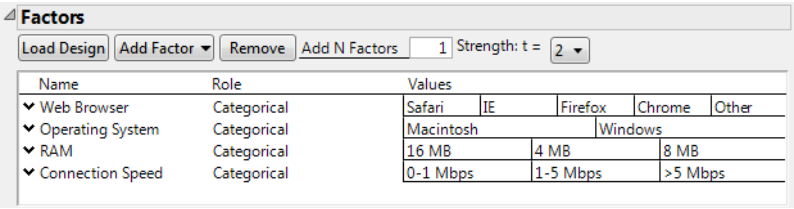
You want to test an internet-based software application to detect issues arising from components in the operating environment. The four components of interest consist of a browser, the operating system, the computer's RAM, and the connection speed. To minimize testing time, you restrict yourself to testing two-way interactions.

Construct a Strength 2 covering array to test the required combinations of factor levels.

1. Select **DOE > Special Purpose > Covering Array**.
2. Select **Help > Sample Data Library** and open Design Experiment/Software Factors.jmp.
3. Click the Covering Array red triangle and select **Load Factors**.

   The factors and their levels appear in the Factors outline.

Figure A.31 Factors Outline for Software Factors



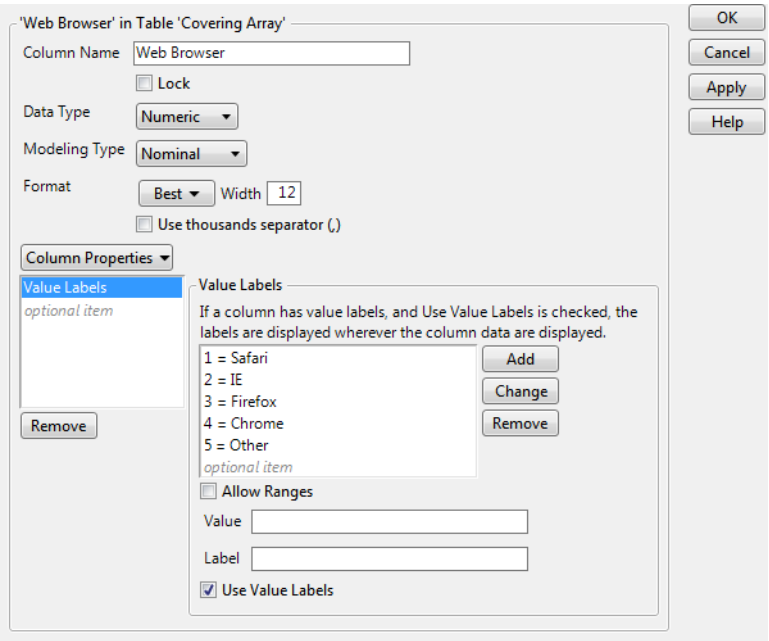
Notice that the Role of the four factors is described as Categorical.

4. Click **Continue**.

   The Restrict Factor Level Combinations outline opens. Since all combinations of settings are possible, leave this set to **None**.

5. Click **Make Design**.
6. Click **Make Table**.
7. In the columns panel, click the asterisk next to Web Browser and select **Value Labels**.

Figure A.32 Column Info Window for Factor A

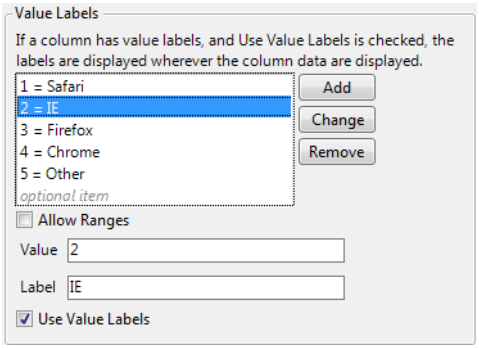


Notice that Web Browser has a Numeric data type and a Nominal modeling type. The underlying values of Web Browser are the integers 1, 2, 3, 4, and 5. These values are assigned value labels corresponding to the values that you entered when you constructed the design.

Change the value label for 2 from IE to Internet Explorer.

8. Select **2 = IE** in the Value Labels list.

Figure A.33 Value Labels Panel with Selection



9. Type **Internet Explorer** next to Label.
10. Click **Change**.

    The change appears in the data table.

Note: To use the numeric values and not the labels, deselect **Use Value Labels**.

RunsPerBlock

When you use the DOE platforms to construct a design containing a blocking factor, the factor is assigned the Design Role column property with the value Blocking. JMP also assigns the RunsPerBlock column property to each Blocking factor. The RunsPerBlock property indicates the maximum allowable number of runs in each block. This property is used by the Evaluate Design and Augment Design platforms to indicate the blocking structure for the factor. See "Blocking".

Note: The RunsPerBlock column property is assigned by JMP as part of design construction. You cannot directly assign this column property.

RunsPerBlock Example

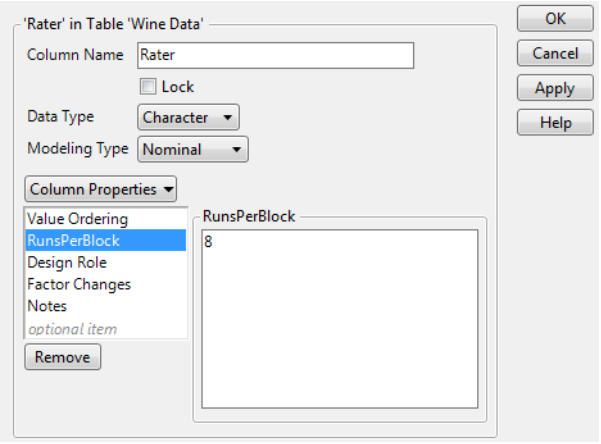
Consider the wine tasting experiment described in "Example of a Custom Design". Wine samples are tasted by five raters (Rater) and each rater tastes eight samples.

1. Select **Help > Sample Data Library** and open Design Experiment/Wine Data.jmp.

2. Click the asterisk next to Rater in the columns panel and select **Design Role**.

   Notice that the Design Role is set to Blocking.
3. Click **Cancel**.
4. Click the asterisk next to Rater in the columns panel and select **RunsPerBlock**.

Figure A.34 RunsPerBlock Column Property Panel for Rater



Notice that the value of RunsPerBlock is 8. The design constructed by the DOE Dialog script has 40 runs. Since there are five raters, JMP constructs a design with 40/5 = 8 runs for each rater.

ConstraintState

In the Custom and Mixture Design platforms, you can Save Constraints that you specify for a design. When you select Save Constraints, the coefficients of each linear constraint appear in a column in a data table. The value that bounds the inequality is given in the last row of the table.

Each constraint column is assigned the ConstraintState column property. This property specifies the direction of the inequality that defines the constraint. When you select Load Constraints from a design platform, the ConstraintState column property tells JMP the direction of the inequality.

Note: The ConstraintState column property is assigned by JMP as part of design construction. You cannot directly assign this column property.

ConstraintState Example

The sample data table Piepel.jmp, located in the Design Experiment folder, contains a mixture design with three continuous factors. The design is based on an experiment presented in Snee (1979) and Piepel (1988), where there are boundary constraints on each factor and three additional linear constraints. In this example, you do the following:

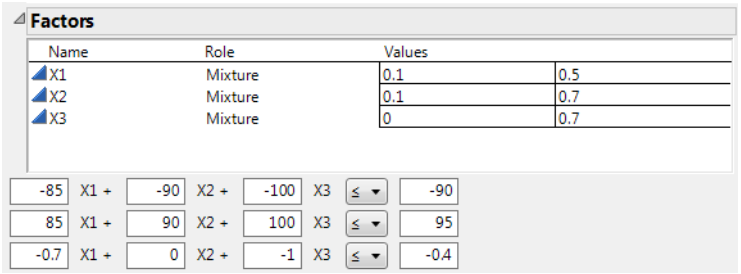
1. Change one of the three additional constraints
2. Save the constraints to a table
3. Observe how the ConstraintState column property describes the direction of the inequality in the constraint

For further development of this example, see "An Extreme Vertices Example with Linear Constraints".

1. Select **Help > Sample Data Library** and open Design Experiment/Piepel.jmp.
2. In the Table panel, click the green triangle next to the **DOE Dialog** script.

   Notice the three linear constraints below the Factors outline. To make the constraints more interpretable, you want to reformulate the first constraint in terms of a "greater than or equal to" inequality.

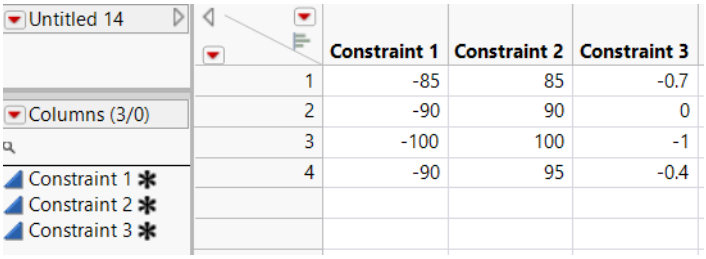
Figure A.35 Linear Constraints beneath Factors Outline



3. In the first constraint, do the following:
   - Type 85 next to X1.
   - Type 90 next to X2.
   - Type 100 next to X3.
   - Select ≥ from the inequality menu.
   - Type 90 to the right of the inequality sign.
4. Click the Mixture Design red triangle and select **Save Constraints**.

   A table containing information about the constraints appears.

Figure A.36 Constraint Table



Untitled 14

Columns (3/0)

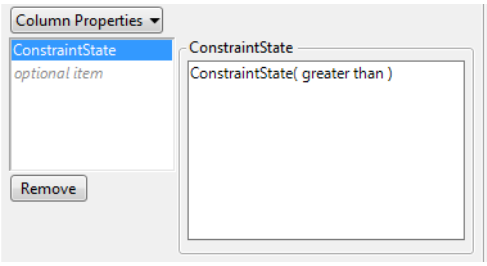
Constraint 1 *
Constraint 2 *
Constraint 3 *

	Constraint 1	Constraint 2	Constraint 3
1	-85	85	-0.7
2	-90	90	0
3	-100	100	-1
4	-90	95	-0.4

Each column contains the coefficients of the factors X1, X2, and X3 in rows 1 through 3. Row 4 contains the value that appeared to the right of the inequality sign.

5. Click the asterisk next to Constraint 1.
6. Click **ConstraintState**.

Figure A.37 ConstraintState Column Property Panel



The ConstraintState panel for X1 indicates that the direction of the inequality is “greater than” indicating greater than or equal to ≥.

7. Click **Cancel**.

Appendix B

Technical Details

This appendix contains information on the Alias Matrix, power calculations and the Relative Prediction Variance.

- "The Alias Matrix"
- "Power Calculations"
- "Relative Prediction Variance"

Contents

The Alias Matrix 825
Designs with Hard or Very Hard Factor Changes 825
Designs with If Possible Effects 826
Power Calculations 826
Power for a Single Parameter 826
Power for a Categorical Effect 828
Relative Prediction Variance 829

The Alias Matrix

The Alias Matrix entries represent the degree of bias imparted to model parameters by the effects that you specified in the Alias Terms outline. The Alias Matrix is also used in defining alias optimality. See "Alias Optimality".

Calculations for the Alias Matrix are based on the model matrix. See "Model Matrix".

Let $\mathbf{X}_1$ be the model matrix corresponding to the terms in the Model outline. Denote the matrix of model terms for the effects specified in the Alias Terms outline by $\mathbf{X}_2$.

The assumed model is given as follows:

$$\mathbf{Y} = \mathbf{X}_1\beta_1 + \varepsilon$$

Suppose that some of the alias terms are active and that the true model is given as follows:

$$\mathbf{Y} = \mathbf{X}_1\beta_1 + \mathbf{X}_2\beta_2 + \varepsilon$$

The least squares estimator of β_1 is given by:

$$\hat{\beta}_1 = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{Y}$$

Under the usual regression assumptions, the expected value of $\hat{\beta}_1$ is given by:

$$E[\hat{\beta}_1] = \beta_1 + \mathbf{A}\beta_2$$

where $\mathbf{A} = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2$.

The matrix **A** is called the *alias matrix*.

Designs with Hard or Very Hard Factor Changes

For designs with hard-to-change or very-hard-to-change factors, the alias matrix is given as follows:

$$\mathbf{A} = (\mathbf{X}_1'\mathbf{V}^{-1}\mathbf{X}_1)^{-1}\mathbf{X}_1'(\mathbf{V}^{-1}\mathbf{X}_2)$$

where **V** is the block diagonal covariance matrix of the responses.

Designs with If Possible Effects

For designs with If Possible effects (Bayesian D- and I-optimal designs), the alias matrix is given as follows:

$$\mathbf{A} = (\mathbf{X}_1'\mathbf{X}_1 + \mathbf{K}^2)^{-1}\mathbf{X}_1'\mathbf{X}_2$$

where **K** is a diagonal matrix with these values:

- $k = 0$ for Necessary terms
- $k = 1$ for If Possible main effects, powers, and interactions involving a categorical factor with more than two levels
- $k = 4$ for all other If Possible terms

In the Bayesian case, the alias matrix gives the aliasing of effects corresponding to a ridge regression with a prior variance of $\mathbf{K}^{-1}$. For additional detail on Bayesian designs, see "Bayesian D-Optimality" and "Bayesian I-Optimality".

In the Custom Design platform, you can control the weights used for If Possible terms by selecting **Advanced Options > Prior Parameter Variance** from the red triangle menu. There you can set prior variances for all model terms by specifying the diagonal elements of **K**. The option updates to show the default weights when you click Make Design.

Power Calculations

The Power Analysis report gives power calculations for single parameter values and, when the design includes a categorical factor with three or more levels, for whole effects. This section describes the calculations in the two cases:

- "Power for a Single Parameter"
- "Power for a Categorical Effect"

Power for a Single Parameter

This section describes how power for the test of a single parameter is computed. Use the following notation:

X The model matrix. See *Fitting Linear Models* for information on the coding for nominal effects. Also, See "Model Matrix".

Note: You can view the model matrix by running Fit Model. Then select **Save Columns > Save Coding Table** from the red triangle menu for the main report.

β_ι The parameter corresponding to the term of interest.

$\hat{\beta}_i$ The least squares estimator of β_i

β_i^A The Anticipated Coefficient value. The difference you want to detect is $2\beta_i^A$.

The variance of $\hat{\beta}_i$ is given by the i^{th} diagonal entry of $\sigma^2(\mathbf{X}'\mathbf{X})^{-1}$, where σ^2 is the error variance. Denote the i^{th} diagonal entry of $(\mathbf{X}'\mathbf{X})^{-1}$ by $(\mathbf{X}'\mathbf{X})_{ii}^{-1}$.

The error variance, σ^2, is estimated by the MSE, and has $n - p - 1$ degrees of freedom, where n is the number of observations and p is the number of terms other than the intercept in the model. If $n - p - 1 = 0$, then JMP sets the degrees of freedom for the error to 1. This allows the power to be estimated for parameters in a saturated design.

The test of $H_0:\beta_i = 0$ is given by:

$$\frac{\hat{\beta}_i}{\sqrt{MSE(\mathbf{X}'\mathbf{X})_{ii}^{-1}}}$$

or equivalently by:

$$F_0 = \frac{\hat{\beta}_i^2}{MSE(\mathbf{X}'\mathbf{X})_{ii}^{-1}}$$

Under the null hypothesis, the test statistic F_0 has an F distribution on 1 and $n - p - 1$ degrees of freedom.

If the true value of β_i is β_i^A , then F_0 has a noncentral F distribution with noncentrality parameter given by:

$$\lambda = \frac{(\beta_i^A)^2}{\sigma^2(\mathbf{X}'\mathbf{X})_{ii}^{-1}}$$

To compute the power of the test, first solve for the α-level critical value F_c:

$$\alpha = 1 - FDist(F_c, 1, n - p - 1)$$

Then calculate the power as follows:

$$Power = 1 - FDist(F_c, 1, n - p - 1, \lambda)$$

Power for a Categorical Effect

This section describes how power for the test for a whole categorical effect is computed. Use the following notation:

X Model matrix. See "The Alias Matrix".

β Vector of parameters.

$\hat{\beta}$ Least squares estimate of β.

β^A Vector of Anticipated Coefficient values.

L Matrix that defines the test for the categorical effect. The matrix **L** identifies the values of the parameters in β corresponding to the categorical effect and sets them equal to 0. The null hypothesis for the test of the categorical effect is given by:

$$H_0: L\beta = \mathbf{0}$$

r Rank of **L**. Alternatively, *r* is the number of levels of the categorical effect minus one.

Note: You can view the design matrix by running Fit Model. Then select **Save Columns > Save Coding Table** from the red triangle menu for the main report.

The covariance matrix of $\hat{\beta}$ is given by $\sigma^2(\mathbf{X}'\mathbf{X})^{-1}$, where σ^2 is the error variance.

The error variance, σ^2, is estimated by the MSE, and has $n - p - 1$ degrees of freedom, where n is the number of observations and p is the number of terms other than the intercept in the model. If $n - p - 1 = 0$, then JMP sets the degrees of freedom for the error to 1. This allows the power to be estimated for parameters in a saturated design.

The test of $H_0: L\beta = \mathbf{0}$ is given by:

$$F_0 = \left((\mathbf{L}\hat{\beta})'[\mathbf{L}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{L}']^{-1}(\mathbf{L}\hat{\beta})\right)/(rMSE)$$

Under the null hypothesis, the test statistic F_0 has an F distribution on r and $n - p - 1$ degrees of freedom.

If the true value of β is β^A, then F_0 has a noncentral F distribution with noncentrality parameter given by:

$$\lambda = \left((\mathbf{L}\beta^A)'[\mathbf{L}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{L}']^{-1}(\mathbf{L}\beta^A)\right)/\sigma^2$$

To compute the power of the test, first solve for the α-level critical value F_c:

$$\alpha = 1 - FDist(F_c, r, n-p-1)$$

Then calculate the power as follows:

$$Power = 1 - FDist(F_c, r, n-p-1, \lambda)$$

Relative Prediction Variance

Consider the following notation:

X Model matrix. See "Model Matrix". Custom designs provide a script that shows the model matrix. See "Save X Matrix".

σ^2 Error variance.

$\hat{\beta}$ Vector of least squares estimates of the parameters.

$\mathbf{x}_i'$ The i^{th} row of **X**.

Using this notation, the predicted response for the i^{th} row of **X** is given by:

$$\hat{\mathbf{Y}} = \mathbf{x}_i'\hat{\beta}$$

The relative prediction variance at the settings defined by $\mathbf{x}_i$ is given by:

$$\frac{\mathbf{x}_i'var(\hat{\mathbf{Y}})\mathbf{x}_i}{\sigma^2} = \frac{\mathbf{x}_i'var(\mathbf{x}_i'\hat{\beta})\mathbf{x}_i}{\sigma^2} = \mathbf{x}_i'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_i$$

Appendix C

References

Agresti, A., and Coull, B. A. (1998). "Approximate Is Better than 'Exact' for Interval Estimation of Binomial Proportions." *American Statistician* 52:119–126.

Agresti, A., and Caffo, B. N. (2000). "Simple and Effective Confidence Intervals for Proportions and Differences of Proportions Result from Adding Two Successes and Two Failures." *American Statistician* 54:280–288.

Atkinson, A. C., and Donev, A. N.(1992). *Optimum Experimental Designs*. New York: Oxford University Press.

Barker, C. (2011). "Power and Sample Size Calculations in JMP." Technical Report, Cary NC: SAS Institute Inc. Accessed May 6, 2018. https://www.jmp.com/content/dam/jmp/documents/en/technical-reports/powerAndSampleSize.pdf.

Borror, C. M., and Montgomery, D. C. (2000). "Mixed Resolution Designs as Alternatives to Taguchi Inner-Outer Array Designs for Robust Design Problems." Quality and Reliability Engineering International 16:117–128.

Box, G. E. P. (1988). "Signal–to–Noise Ratio, Performance Criteria, and Transformations." *Technometrics* 30:1–40.

Box, G. E. P., and Behnken, D. W. (1960). "Some New Three-Level Designs for the Study of Quantitative Variables." *Technometrics* 2:455–475.

Box, G. E. P., and Draper, N. R. (1987). *Empirical Model–Building and Response Surfaces*. New York: John Wiley & Sons.

Box, G. E. P., Hunter, W. G., and Hunter, J. S. (1978). *Statistics for Experimenters*. New York: John Wiley & Sons.

Box, G. E. P., Hunter, W. G., and Hunter, J. S. (2005). *Statistics for Experimenters: Design, Innovation, and Discovery*. 2nd ed. New York: John Wiley & Sons.

Box, G. E. P., and Meyer, R. D. (1986). "An Analysis of Unreplicated Fractional Factorials." *Technometrics* 28:11–18.

Box, G. E. P., and Wilson, K. B. (1951). "On the Experimental Attainment of Optimum Conditions." *Journal of the Royal Statistical Society, Series B* 13:1–45.

Byrne, D. M., and Taguchi, G. (1986). "The Taguchi Approach to Parameter Designs." *Quality Congress Transactions* 177:168–177.

Chow, S. -C., Shao, J., and Wang, H. (2008). *Sample Size Calculations in Clinical Research*. 2nd ed. Boac Raton, FL: Chapman & Hall/CRC.

Clopper, C. J., and Pearson, E. S. (1934). "The Use of Confidence or Fiducial Limits Illustrated in the Case of the Binomial." *Biometrika* 26:404–413.

Cochran, W. G., and Cox, G. M. (1957). *Experimental Designs*. 2nd ed. New York: John Wiley & Sons.

Cohen, M. B., Gibbons, P. B., Mugridge, W. B., and Colbourn, C. J. (2003). "Constructing Test Suites for Interaction Testing." In *Proceedings 25th International Conference on Software Engineering*, 38–48.

Cohen, M. B., Dwyer, M. B., and Shi, J. (2007). "Interaction Testing of Highly-Configurable Systems in the Presence of Constraints." In *Proceedings of the 2007 International Symposium on Software Testing and Analysis*, 129–139.

Colbourn, C. (2004). "Combinatorial Aspects of Covering Arrays." *Le Matematiche* 58:121–167.

Colbourn, C., and Dinitz, J., eds. (2010). *Handbook of Combinatorial Designs*. 2nd ed. Boca Raton, FL: CRC Press.

Cornell, J. A. (1990). *Experiments with Mixtures*. 2nd ed. New York: John Wiley & Sons.

Dalal, S. R., and Mallows, C. L. (1998). "Factor-Covering Designs for Testing Software." *Technometrics* 40:234–243.

DuMouchel, W., and Jones, B. (1994). "A Simple Bayesian Modification of D-Optimal Designs to Reduce Dependence on an Assumed Model." *Technometrics* 36:37–47.

Errore, A., Jones, B., Li, W., and Nachtsheim, C. (2017). "Using Definitive Screening Designs to Identify Active First- and Second-Order Factor Effects." *Journal of Quality Technology* 49:244–264.

Fries, A., and Hunter, W. G. (1980). "Minimum Aberration 2^{k-p} Designs." *Technometrics* 22:601–608.

Gotwalt, C. M., Jones, B. A., and Steinberg, D. M., (2009). "Fast Computation of Designs Robust to Parameter Uncertainty for Nonlinear Settings." *Technometrics* 51:88–95.

Goos, P. (2002). *The Optimal Design of Blocked and Split Plot Experiments*. New York: Springer.

Goos, P., and Jones, B. (2011). *Optimal Design of Experiments: A Case Study Approach*. New York: John Wiley & Sons.

Hamada, M., and Wu, C. F. J., (1992). "Analysis of Designed Experiments with Complex Aliasing." *Technometrics* 24:130–137.

Hartman, A., and Raskin, L. (2004). "Problems and Algorithms for Covering Arrays." *Discrete Mathematics* 284:149–156.

Horn, R., and Johnson, C. (2012). *Matrix Analysis*. 2nd ed. Cambridge: Cambridge University Press.

Huber, J., and Zwerina, K. (1996). "The Importance of Utility Balance in Efficient Choice Designs." *Journal of Marketing Research* 33:307–317.

Jennrich, R. I. (1969). "Asymptotic Properties of Non-Linear Least Squares Estimators." *Annals of Mathematical Statistics* 40:633–643.

Johnson, M. E., and Nachtsheim, C. J. (1983). "Some Guidelines for Constructing Exact D–Optimal Designs on Convex Design Spaces." *Technometrics* 25:271–277.

Jones, B. (1991). "An Interactive Graph For Exploring Multidimensional Response Surfaces." In *Proceedings of the Annual Joint Statistical Meetings of the American Statistical Association*. Alexandria. VA: American Statistical Association.

Jones, B., and Goos, P. (2007). "A Candidate-set-free Algorithm for Generating D-optimal Designs." *Applied Statistics* 56:1–18. Accessed January 23, 2017. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3001117/.

Jones, B., and Goos, P. (2015). "Optimal Design of Blocked Experiments in the Presence of Supplementary Information about the Blocks." *Technometrics* 47:01–317.

Jones, B., and Johnson, R. T. (2009). "Design and Analysis for the Gaussian Process Model." *Quality and Reliability Engineering International* 25:515–524. Accessed January 23, 2017. https://www.stat.osu.edu/~comp_exp/jour.club/Jones_Johnson_QREI09.pdf.

Jones, B., Lekivetz, R., Majumdar, D., Nachtsheim, C. J., and Stallings, J. W. (2019). "Construction, Properties, and Analysis of Group-Orthogonal Supersaturated Designs."*Technometrics* 62: 403 - 414. Accessed December 15, 2020. https://doi.org/10.1080/00401706.2019.1654926

Jones, B., Lin, D. K. J., and Nachtsheim, C. J. (2008). "Bayesian D-Optimal Supersaturated Designs." *Journal of Statistical Planning and Inference* 138:86–92.

Jones, B., and Nachtsheim, C. J. (2011a). "A Class of Three-Level Designs for Definitive Screening in the Presence of Second-Order Effects." *Journal of Quality Technology* 43:1–15. Accessed January 23, 2017. https://www.jmp.com/en_us/whitepapers/jmp/class-three-level-definitive-screening.html.

Jones, B., and Nachtsheim, C. J. (2011b). "Efficient Designs with Minimal Aliasing." *Technometrics* 53:62–71.

Jones, B., and Nachtsheim, C. J. (2013). "Definitive Screening Designs with Added Two-Level Categorical Factors." *Journal of Quality Technology* 45:121–129.

Jones, B., and Nachtsheim, C. J. (2016a). "Blocking Schemes for Definitive Screening Designs." *Technometrics* 58:74–83.

Jones, B., and Nachtsheim, C. J. (2016b). "Effective Model Selection for Definitive Screening Designs." *Technometrics*. Accessed September 28, 2017. https://www.tandfonline.com/doi/full/10.1080/00401706.2016.1234979.

Joseph, V. R., Gul, E., and Ba, S. (2015). "Maximum Projection Designs for Computer Experiments." *Biometrika* 102:371–380.

Jowett, G. (1963). "The Relationship Between the Binomial and F Distributions." Journal of the Royal Statistical Society 13:55–57.

Kessels, R., Jones, B., Goos, P. (2011). "Bayesian Optimal Designs for Discrete Choice Experiments with Partial Profiles." *Journal of Choice Modelling* 4:52–74.

Kowalski, S. M., Cornell, J. A., and Vining, G. G. (2002). "Split Plot Designs and Estimation Methods for Mixture Experiments with Process Variables." *Technometrics* 44:72–79.

Krishnamoorthy, K., and Mathew, T. (2009). *Statistical Tolerance Regions: Theory, Applications, and Computations*. New Your: John Wiley & Sons.

Lenth, R. V. (1989). "Quick and Easy Analysis of Unreplicated Fractional Factorials." *Technometrics* 31:469–473.

Lekivetz, R. (2014). "What is an Alias Matrix?" Accessed January 23, 2017. https://blogs.sas.com/content/jmp/2014/06/05/what-is-an-alias-matrix/.

Lekivetz, R., Sitter, R., Bingham, D., Hamada, M. S., Moore, L. M., and Wendelberger, J. R. (2015). "On Algorithms for Obtaining Orthogonal and Near-Orthogonal Arrays for Main-Effects Screening." *Journal of Quality Technology* 47:2–13.

Lin, D. K. J. (1993). "A New Class of Supersaturated Design." *Technometrics* 35:28–31.

Martirosyan, S. (2003). "Perfect Hash Families, Identifiable Parent Property Codes and Covering Arrays." Ph.D. diss., Universitat Duisburg-Essen.

Meyer, R. K., and Nachtsheim, C. J. (1995). The Coordinate Exchange Algorithm for Constructing Exact Optimal Designs." *Technometrics* 37:60–69.

Meyer, R. D., Steinberg, D. M., and Box, G.(1996). Follow-up Designs to Resolve Confounding in Multifactor Experiments. *Technometrics* 38:307.

Miller, A., and Sitter, R. R. (2005). "Using Folded-Over Nonorthogonal Designs." *Technometrics* 47:502–513.

Montgomery, D. C. (2013). *Design and Analysis of Experiments*. 8th ed. New York: John Wiley & Sons.

Morgan, J. (2009). "Constrained Covering Arrays: Resolving Invalid Level Combination Constraints." Technical Report, Cary NC: SAS Institute Inc., also in *Supplemental Proceedings of the 20th International Symposium on Software Reliability Engineering*. Accessed May 6, 2018. https://www.jmp.com/content/dam/jmp/documents/en/technical-reports/ConstrainedCovArray09.pdf.

Morgan, J. (2014). "Computing Coverage and Diversity Metrics for Constrained Covering Arrays," Technical Report, Cary NC: SAS Institute Inc. Accessed May 6, 2018. https://www.jmp.com/content/dam/jmp/documents/en/technical-reports/CoverageDiversity2014.pdf.

Morgan, J. P., and Stallings, J. (2017). "Optimal experimental design that targets meaningful information." *WIREs Comput Stat* 2017, 9:e1393. doi: 10.1002/wics.1393.

Myers, R. H., Montgomery, D. C., and Anderson-Cook, C. M. (2009). *Response Surface Methodology, Process and Product Optimization Using Designed Experiments*. 3rd ed. New York: John Wiley & Sons.

Mylona, K., Goos, P., and Jones, B. (2014). "Optimal Design of Blocked and Split-plot Experiments for Fixed Effects and Variance Component Estimation." *Technometrics* 56:132–144.

Nayeri, P., Colbourn, C., and Konjevod, G. (2013). "Randomized Postoptimization of Covering Arrays." *European Journal of Combinatorics* 34:91–103.

Piepel, G. F. (1988). "Programs for Generating Extreme Vertices and Centroids of Linearly Constrained Experimental Regions." *Journal of Quality Technology* 20:125–139.

Sall, J. (2008). "Choice Experimental Designs Are Different." Accessed January 22, 2016. https://blogs.sas.com/content/jmp/2008/12/16/choice-experimental-designs-are-different/.

SAS Institute Inc. (2013). "The %MktBIBD Macro." Accessed September 27, 2017. https://support.sas.com/rnd/app/macros/mktbibd/mktbibd.htm.

Schoen, E. D. (1999). "Designing Fractional Two-level Experiments with Nested Error Structures." *Journal of Applied Statistics* 26: 495–508.

Shewry, M. C., and Wynn, H. P. (1987). "Maximum Entropy Sampling." *Journal of Applied Statistics* 14:165–170.

Snee, R. D., and Marquardt, D. W. (1974). "Extreme Vertices Designs for Linear Mixture Models." *Technometrics* 16:391–408.

Snee, R. D. (1975). "Experimental Designs for Quadratic Models in Constrained Mixture Spaces." *Technometrics* 17:149–159.

Snee, R. D. (1979). "Experimental Designs for Mixture Systems with Multicomponent Constraints." *Communications in Statistics -Theory and Methods* 8:303–326.

Snee, R. D. (1985). "Computer Aided Design of Experiments - Some Practical Experiences." *Journal of Quality Technology* 17:222–236.

Vivacqua, C. A., and Bisgaard, S. (2004). "Strip-Block Experiments for Process Improvement and Robustness." *Quality Engineering* 16:495–500.

Worley, B. A. (1987). *Deterministic Uncertainty Analysis*. Technical Report ORN-0628, Oak Ridge National Laboratory.

Wu, C. F. J. (1981). "Asymptotic Theory of Nonlinear Least Squares Estimation." *Annals of Mathematical Statistics* 9:501–513.

Wu, C. F. J., and Hamada, M. S. (2009). *Experiments: Planning, Analysis, and Optimization*. 2nd ed. New York: John Wiley & Sons.

Xiao, L., Lin, D. K. J., and Bai, F. (2012). "Constructing Definitive Screening Designs Using Conference Matrices." *Journal of Quality Technology* 44:2–8.

Ye, K. Q., and Hamada, M. (2000). "Critical Values of the Lenth Method for Unreplicated Factorial Designs." *Journal of Quality Technology* 32:57–66.

Yilmaz, C., Fouche, S., Cohen, M. B., Porter, A., Demiroz, G., and Koc, U. (2014). "Moving Forward with Combinatorial Interaction Testing." *Computer* 47:37–45.

Zahran, A., Anderson-Cook, C. M., and Myers, R. H. (2003). "Fraction of Design Space to Assess the Prediction Capability of Response Surface Designs." *Journal of Quality Technology* 35:377–386.

Zhang, Z., and Zhang, J. (2011). "Characterizing Failure-Causing Parameter Interactions by Adaptive Testing." In *Proceedings of the 2011 International Symposium on Software Testing and Analysis*, 331–341.

Appendix D

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

  - You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - You must cause any modified files to carry prominent notices stating that You changed the files; and
  - You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
  - You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS”, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Fitting Linear Models

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Fitting Linear Models*. Cary, NC: SAS Institute Inc.

JMP® 16 Fitting Linear Models

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Fitting Linear Models

1 Learn about JMP 15

Documentation and Additional Resources

Formatting Conventions in JMP Documentation 17
JMP Help 18
JMP Documentation Library 18
Additional Resources for Learning JMP 24
JMP Tutorials 24
Sample Data Tables 24
Learn about Statistical and JSL Terms 25
Learn JMP Tips and Tricks 25
JMP Tooltips 25
JMP User Community 26
Free Online Statistical Thinking Course 26
JMP New User Welcome Kit 26
Statistics Knowledge Portal 26
JMP Training 26
JMP Books by Users 27
The JMP Starter Window 27
JMP Technical Support 27

2 Model Specification 29

Specify Linear Models

Overview of the Fit Model Platform 31
Example of a Regression Analysis Using Fit Model 32
Launch the Fit Model Platform 35
Fit Model Launch Window Elements 36
Construct Model Effects 38
Fitting Personalities 45
Model Specification Options 47
Informative Missing 49
Validity Checks 50
Examples of Model Specifications and Their Model Fits 50
Simple Linear Regression 51

Polynomial in X to Degree k ... 52
Polynomial in X and Z to Degree k ... 53
Multiple Linear Regression ... 53
One-Way Analysis of Variance ... 54
Two-Way Analysis of Variance ... 55
Two-Way Analysis of Variance with Interaction ... 56
Three-Way Full Factorial ... 58
Analysis of Covariance, Equal Slopes ... 60
Analysis of Covariance, Unequal Slopes ... 61
Two-Factor Nested Random Effects Model ... 63
Three-Factor Fully Nested Random Effects Model ... 65
Simple Split Plot or Repeated Measures Model ... 66
Two-Factor Response Surface Model ... 68
Knotted Spline Effect ... 70

3 Standard Least Squares Models ... 73
Analyze Common Classes of Models

Example Using Standard Least Squares ... 76
Launch the Standard Least Squares Personality ... 79
Fit Model Launch Window ... 80
Standard Least Squares Options in the Fit Model Launch Window ... 81
Validation in Standard Least Squares ... 83
Missing Values ... 83
Fit Least Squares Report ... 84
Single versus Multiple Responses ... 84
Report Structure Related to Emphasis ... 85
Special Reports ... 85
Least Squares Fit Options ... 89
Fit Group Options ... 89
Response Options ... 90
Regression Reports ... 91
Summary of Fit ... 92
Analysis of Variance ... 93
Parameter Estimates ... 94
Effect Tests ... 96
Effect Details ... 97
Lack of Fit ... 111
Estimates ... 113
Show Prediction Expression ... 115
Sorted Estimates ... 116

Expanded Estimates . . . 119
Indicator Parameterization Estimates . . . 121
Sequential Tests . . . 122
Custom Test . . . 123
Multiple Comparisons . . . 126
Compare Slopes . . . 140
Joint Factor Tests . . . 140
Inverse Prediction . . . 141
Cox Mixtures . . . 145
Parameter Power . . . 147
Correlation of Estimates . . . 149
Coding for Nominal Effects . . . 150
Effect Screening . . . 151
Scaled Estimates and the Coding of Continuous Terms . . . 152
Plot Options . . . 153
Normal Plot Report . . . 158
Bayes Plot Report . . . 159
Pareto Plot Report . . . 161
Factor Profiling . . . 162
Profiler . . . 163
Interaction Plots . . . 164
Contour Profiler . . . 165
Mixture Profiler . . . 166
Cube Plots . . . 167
Box Cox Y Transformation . . . 168
Surface Profiler . . . 170
Row Diagnostics . . . 171
Leverage Plots . . . 173
Press . . . 177
Save Columns . . . 178
Prediction Formula . . . 181
Effect Summary Report . . . 181
Mixed and Random Effect Model Reports and Options . . . 186
Mixed Models and Random Effect Models . . . 186
Restricted Maximum Likelihood (REML) Method . . . 190
EMS (Traditional) Model Fit Reports . . . 195
Models with Linear Dependencies among Model Terms . . . 198
Singularity Details . . . 198
Parameter Estimates Report . . . 199
Effect Tests Report . . . 200

Examples ... 200
Statistical Details for the Standard Least Squares Personality ... 202
Emphasis Rules ... 203
Details of Custom Test Example ... 203
Correlation of Estimates ... 204
Leverage Plot Details ... 204
The Kackar-Harville Correction ... 207
Power Analysis ... 208

4 Standard Least Squares Examples ... 219
Analyze Common Classes of Models

One-Way Analysis of Variance Example ... 221
Analysis of Covariance with Equal Slopes Example ... 224
Analysis of Covariance with Unequal Slopes Example ... 227
Response Surface Model Example ... 229
Fit the Full Response Surface Model ... 230
Reduce the Model ... 231
Examine the Response Surface Report ... 232
Find the Critical Point Using the Prediction Profiler ... 233
View the Surface Using the Contour Profiler ... 235
Split Plot Design Example ... 237
Estimation of Random Effect Parameters Example ... 240
Knotted Spline Effect Example ... 242
Bayes Plot for Active Factors Example ... 244

5 Stepwise Regression Models ... 247
Find a Model Using Variable Selection

Overview of Stepwise Regression ... 249
Example Using Stepwise Regression ... 249
The Stepwise Report ... 251
Stepwise Platform Options ... 251
Stepwise Regression Control Panel ... 252
Current Estimates Report ... 259
Step History Report ... 260
Models with Crossed, Interaction, or Polynomial Terms ... 261
Example of the Combine Rule ... 261
Models with Nominal and Ordinal Effects ... 263
Construction of Hierarchical Terms ... 263
Example of a Model with a Nominal Term ... 264
Example of the Restrict Rule for Hierarchical Terms ... 269
Performing Binary and Ordinal Logistic Stepwise Regression ... 271

Example Using Logistic Stepwise Regression ... 271
The All Possible Models Option ... 273
Example Using the All Possible Models Option ... 273
The Model Averaging Option ... 275
Example Using the Model Averaging Option ... 275
Validation Options in Stepwise Regression ... 276
Validation Set with Two or Three Values in Stepwise Regression ... 276
K-Fold Cross Validation in Stepwise Regression ... 279

6 Generalized Regression Models ... 281
Build Models Using Variable Selection Techniques

Overview of the Generalized Regression Personality ... 283
Example of Generalized Regression ... 285
Launch the Generalized Regression Personality ... 288
Distribution ... 290
Generalized Regression Report Window ... 298
Generalized Regression Report Options ... 298
Model Launch Control Panel ... 299
Response Distribution ... 300
Estimation Method Options ... 300
Advanced Controls ... 305
Validation Method Options ... 308
Early Stopping ... 309
Go ... 310
Model Fit Reports ... 310
Regression Plot ... 310
Model Summary ... 311
Estimation Details ... 314
Solution Path ... 315
Parameter Estimates for Centered and Scaled Predictors ... 319
Parameter Estimates for Original Predictors ... 320
Active Parameter Estimates ... 321
Effect Tests ... 321
Model Fit Options ... 322
Statistical Details for the Generalized Regression Personality ... 333
Statistical Details for Estimation Methods ... 333
Statistical Details for Advanced Controls ... 335
Statistical Details for Distributions ... 336

7 Generalized Regression Examples 343
Build Models Using Regularization Techniques

Poisson Generalized Regression Example 345
Binomial Generalized Regression Example 347
Zero-Inflated Poisson Regression Example 349
Generalized Regression Example for Wide Data 352

8 Mixed Models 357
Jointly Model the Mean and Covariance

Overview of the Mixed Model Personality 359
Example Using Mixed Model 360
Launch the Mixed Model Personality 364
Fit Model Launch Window 364
Data Format 370
The Fit Mixed Report 370
Fit Statistics 373
Random Effects Covariance Parameter Estimates 375
Fixed Effects Parameter Estimates 376
Repeated Effects Covariance Parameter Estimates 377
Random Coefficients 377
Random Effects Predictions 377
Fixed Effects Tests 378
Sequential Tests 378
Multiple Comparisons 379
Compare Slopes 380
Marginal Model Inference 380
Actual by Predicted Plot 380
Residual Plots 381
Profilers 381
Conditional Model Inference 382
Actual by Conditional Predicted Plot 382
Conditional Residual Plots 382
Conditional Profilers 383
Variogram 384
Covariance and Correlation Matrices 385
Save Columns 385
Additional Examples of the Mixed Models Personality 386
Repeated Measures Example 387
Split Plot Example 402
Spatial Example: Uniformity Trial 408

Correlated Response Example ... 417
Statistical Details for the Mixed Models Personality ... 424
Convergence Score Test ... 424
Random Coefficient Model ... 425
Repeated Measures ... 426
Repeated Covariance Structures ... 427
Spatial and Temporal Variability ... 432
The Kackar-Harville Correction ... 435

9 Multivariate Response Models ... 437
Fit Relationships Using MANOVA

Example of a Multiple Response Model ... 439
Launch the Manova Personality ... 441
The Manova Report ... 441
The Manova Fit Options ... 442
Response Specification ... 443
Choose Response Options ... 443
Custom Test Option ... 444
Options for Model Effects ... 444
Multivariate Tests ... 448
The Extended Multivariate Report ... 449
Comparison of Multivariate Tests ... 450
Univariate Tests and the Test for Sphericity ... 451
Multivariate Model with Repeated Measures ... 452
Repeated Measures Example ... 453
Example of a Compound Multivariate Model ... 455
Discriminant Analysis ... 457
Example of the Save Discrim Option ... 458
Statistical Details for the Manova Personality ... 459
Multivariate Tests ... 459
Approximate F-Tests ... 460
Canonical Details ... 461

10 Loglinear Variance Models ... 465
Model the Variance and the Mean of the Response

Overview of the Loglinear Variance Model ... 467
Dispersion Effects ... 467
Model Specification ... 467
Notes ... 468
Launch the Loglinear Variance Personality ... 468
Example Using Loglinear Variance ... 469

The Loglinear Report ... 471
Loglinear Platform Options ... 472
Save Columns ... 473
Row Diagnostics ... 473
Examining the Residuals ... 474
Profiling the Fitted Model ... 475
Example of Profiling the Fitted Model ... 475

11 Logistic Regression Models ... 477
Fit Regression Models for Nominal or Ordinal Responses

Overview of the Nominal and Ordinal Logistic Personalities ... 479
Nominal Logistic Regression ... 479
Ordinal Logistic Regression ... 479
Other JMP Platforms That Fit Logistic Regression Models ... 480
Examples of Logistic Regression ... 480
Example of Nominal Logistic Regression ... 480
Example of Ordinal Logistic Regression ... 482
Launch the Nominal and Ordinal Logistic Personalities ... 485
Validation in Logistic Regression Models ... 486
The Logistic Fit Report ... 486
Whole Model Test ... 488
Fit Details ... 489
Lack of Fit Test ... 490
Logistic Fit Platform Options ... 490
Options for Nominal and Ordinal Fits ... 490
Options for Nominal Fits ... 492
Options for Ordinal Fits ... 494
Additional Examples of Logistic Regression ... 495
Example of Inverse Prediction ... 495
Example of Using Effect Summary for a Nominal Logistic Model ... 497
Example of a Quadratic Ordinal Logistic Model ... 499
Example of Stacking Counts in Multiple Columns ... 502
Statistical Details for the Nominal and Ordinal Logistic Personalities ... 503
Logistic Regression Model ... 504
Odds Ratios ... 504
Relationship of Statistical Tests ... 505

12 Generalized Linear Models ... 507
Fit Models for Nonnormal Response Distributions

Overview of the Generalized Linear Model Personality ... 509
Example of a Generalized Linear Model ... 510

Launch the Generalized Linear Model Personality . . . 513
Generalized Linear Model Fit Report . . . 515
Whole Model Test . . . 516
Generalized Linear Model Fit Report Options . . . 517
Additional Examples of the Generalized Linear Models Personality . . . 520
Using Contrasts to Compare Differences in the Levels of a Variable . . . 520
Poisson Regression with Offset . . . 521
Normal Regression with a Log Link . . . 522
Statistical Details for the Generalized Linear Model Personality . . . 525
Model Selection and Deviance . . . 527

A Statistical Details . . . 531
Fitting Linear Models
The Response Models . . . 533
Continuous Responses . . . 533
Nominal Responses . . . 534
Ordinal Responses . . . 535
The Factor Models . . . 537
Continuous Factors . . . 537
Nominal Factors . . . 537
Ordinal Factors . . . 549
Frequencies . . . 554
The Usual Assumptions . . . 555
Assumed Model . . . 555
Relative Significance . . . 555
Multiple Inferences . . . 555
Validity Assessment . . . 556
Alternative Methods . . . 556
Key Statistical Concepts . . . 557
Uncertainty, a Unifying Concept . . . 557
The Two Basic Fitting Machines . . . 558
Likelihood, AICc, and BIC . . . 561
Power Calculations . . . 562
Computations for the LSN . . . 563
Computations for the LSV . . . 563
Computations for the Power . . . 564
Computations for the Adjusted Power . . . 565
Inverse Prediction with Confidence Limits . . . 566

B References 569
C Technology License Notices 575

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 17
JMP Help . . . 18
JMP Documentation Library . . . 18
Additional Resources for Learning JMP . . . 24
JMP Tutorials . . . 24
Sample Data Tables . . . 24
Learn about Statistical and JSL Terms . . . 25
Learn JMP Tips and Tricks . . . 25
JMP Tooltips . . . 25
JMP User Community . . . 26
Free Online Statistical Thinking Course . . . 26
JMP New User Welcome Kit . . . 26
Statistics Knowledge Portal . . . 26
JMP Training . . . 26
JMP Books by Users . . . 27
The JMP Starter Window . . . 27
JMP Technical Support . . . 27

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Model Specification

Specify Linear Models

Using the Fit Model platform, you can specify complex models efficiently. Your task is simplified by Macros, Attributes, and transformations. Fit Model is your gateway to fitting a broad variety of models and effect structures.

These include:

- simple and multiple linear regression
- analysis of variance and covariance
- random effect, nested effect, mixed effect, repeated measures, and split plot models
- nominal and ordinal logistic regression
- multivariate analysis of variance (MANOVA)
- canonical correlation and discriminant analysis
- loglinear variance (to model the mean and the variance)
- generalized linear models (GLM)
- parametric survival and proportional hazards
- response screening, for studying a large number of responses

JMP PRO In JMP Pro, you can also fit the following:

- mixed models with a range of covariance structures
- generalized regression models including the elastic net, lasso, and ridge regression
- partial least squares

The Fit Model platform lets you fit a large variety of types of models by selecting the desired personality. This chapter focuses on the elements of the Model Specification window that are common to most personalities.

Contents

Overview of the Fit Model Platform . . . 31
Example of a Regression Analysis Using Fit Model . . . 32
Launch the Fit Model Platform . . . 35
Fit Model Launch Window Elements . . . 36
Construct Model Effects . . . 38
Fitting Personalities . . . 45
Model Specification Options . . . 47
Informative Missing . . . 49
Validity Checks . . . 50
Examples of Model Specifications and Their Model Fits . . . 50
Simple Linear Regression . . . 51
Polynomial in X to Degree k . . . 52
Polynomial in X and Z to Degree k . . . 53
Multiple Linear Regression . . . 53
One-Way Analysis of Variance . . . 54
Two-Way Analysis of Variance . . . 55
Two-Way Analysis of Variance with Interaction . . . 56
Three-Way Full Factorial . . . 58
Analysis of Covariance, Equal Slopes . . . 60
Analysis of Covariance, Unequal Slopes . . . 61
Two-Factor Nested Random Effects Model . . . 63
Three-Factor Fully Nested Random Effects Model . . . 65
Simple Split Plot or Repeated Measures Model . . . 66
Two-Factor Response Surface Model . . . 68
Knotted Spline Effect . . . 70

Overview of the Fit Model Platform

The Fit Model platform gives you an efficient way to specify models that have complex effect structures. These effect structures are linear in the model parameters. Once you have specified your model, you can select the appropriate fitting technique from a number of fitting *personalities*. Once you choose a personality, the Fit Model window provides choices that are relevant for the chosen personality. This chapter focuses on the elements of the Model Specification window that are common to most personalities. For a description of all personalities, see "Fit Model Launch Window Elements".

Fit Model can be used to specify a wide variety of models that can be fit using various methods. Table 2.1 lists some typical models that can be defined using Fit Model. In the table, the effects X and Z represent columns with a continuous modeling type, while A, B, and C represent columns with a nominal or ordinal modeling type.

See "Examples of Model Specifications and Their Model Fits" for the clicking sequences that produce these model effects, plots of the model fits, and some examples.

Table 2.1 Standard Model Types

Type of Model	Model Effects
Simple Linear Regression	X
Polynomial in X to Degree k	X, X*X,..., X^k
Polynomial in X and Z to Degree k	X, X*X,..., X^k, Z, Z*Z,..., Z^k
Multiple Linear Regression	X, Z, and other continuous columns
One-Way Analysis of Variance	A
Two-Way Analysis of Variance	A, B
Two-Way Analysis of Variance with Interaction	A, B, A*B
Three-Way Full Factorial	A, B, C, A*B, A*C, B*C, A*B*C
Analysis of Covariance, Equal Slopes	A, X
Analysis of Covariance, Unequal Slopes	A, X, A*X
Two-Factor Nested Random Effects Model	A, B[A]&Random
Three-Factor Fully Nested Random Effects Model	A, B[A]&Random, C[A,B]&Random

Table 2.1 Standard Model Types *(Continued)*

Type of Model	Model Effects
Simple Split Plot or Repeated Measures Model	A, B[A]&Random, C, C*A
Two-Factor Response Surface Model	X&RS, Z&RS, X*X, X*Z, Z*Z

Example of a Regression Analysis Using Fit Model

You have data resulting from an aerobic fitness study, and you want to predict the oxygen uptake from several continuous variables.

1. Select **Help > Sample Data Library** and open Fitness.jmp.
2. Select **Analyze > Fit Model**. Note that the Personality box is empty.
3. Select Oxy and click **Y**.

   When you specify a continuous response, the Personality defaults to Standard Least Squares, but you are free to choose another personality. Also, the Emphasis defaults to Effect Leverage.
4. Press Ctrl and select Sex, Age, Weight, Runtime, RunPulse, RstPulse, and MaxPulse. Click **Add** to add these to the Construct Model Effects list. Note that you can select **Keep dialog open** if you want to have this window available later on. Your Model Specification window should appear as shown in Figure 2.1.
5. Click **Run**. Figure 2.2 gives a partial view of the report.

Figure 2.1 Model Specification Window for Fitness Regression Model

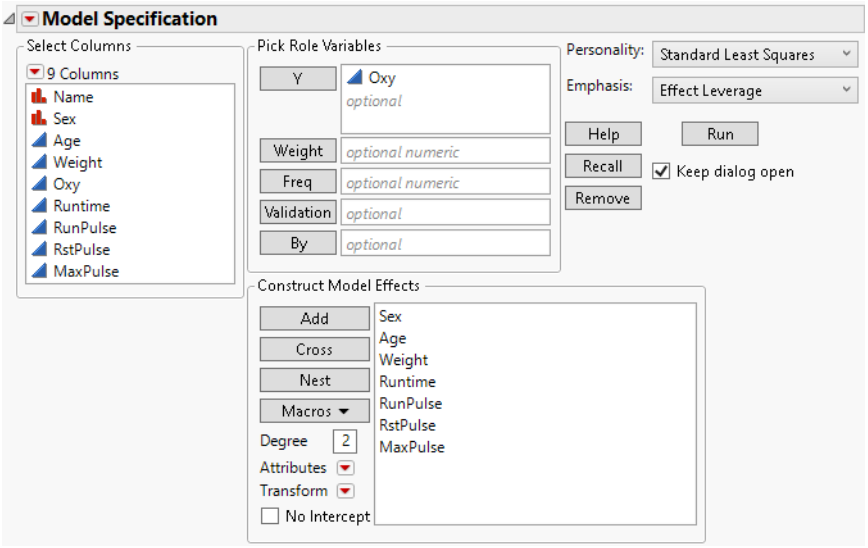
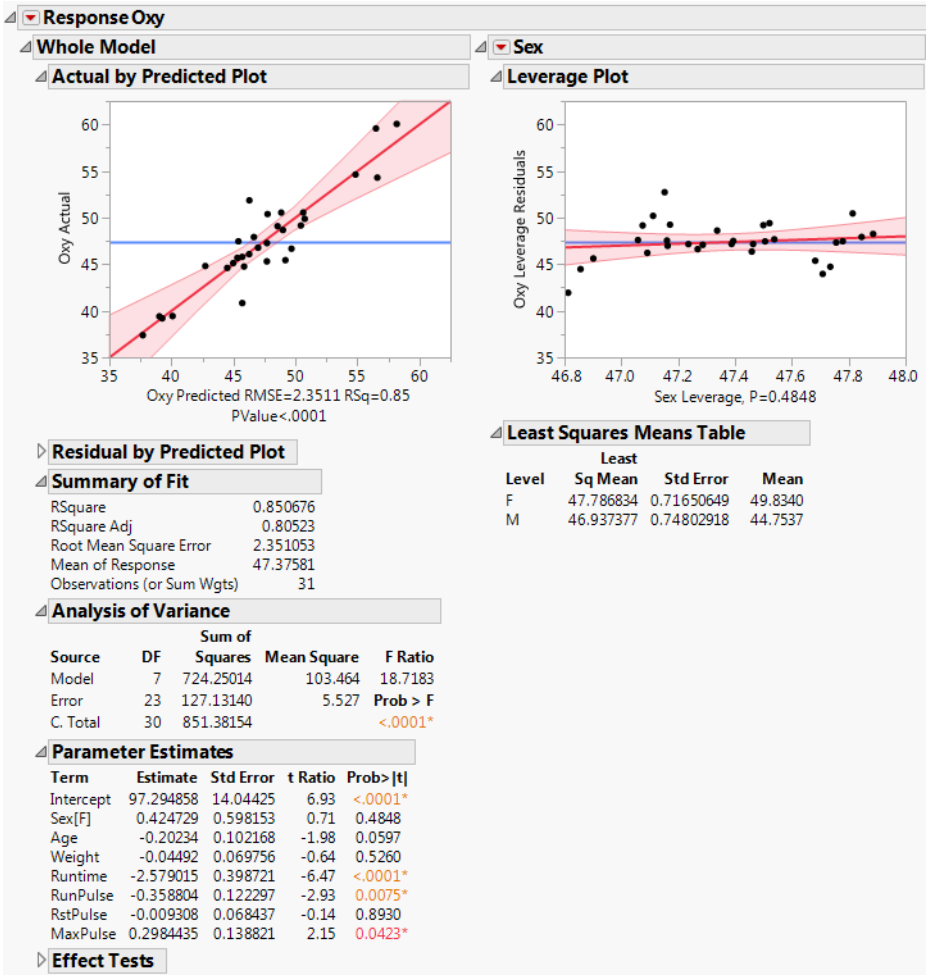


Figure 2.2 Partial View of Standard Least Squares Report for Fitness Data



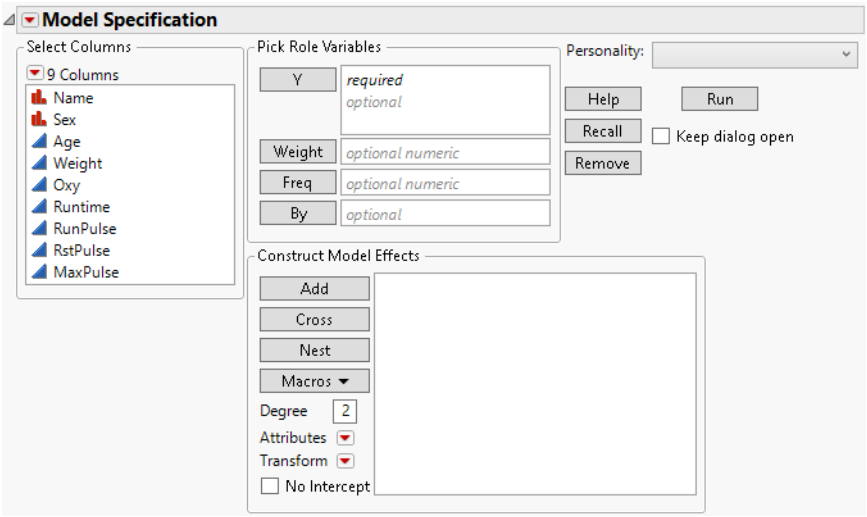
The plot and reports for the whole model appear in the left-most report column. The columns to the right show leverage plots for each of the effects that you specified in the model. Due to space limitations, Figure 2.2 shows only the column for Sex, but the report shows columns for the other six effects as well. The red triangle menus contain additional options that add reports and plots to the report window. For more information about the Standard Least Squares report window, see "Fit Least Squares Report".

Looking at the *p*-values in the Parameter Estimates report, you can see that Runtime, RunPulse, and MaxPulse appear to be significant predictors of oxygen uptake. The next step might be to reduce the model by removing insignificant predictors. See "Stepwise Regression Models".

Launch the Fit Model Platform

Launch the Fit Model platform by selecting **Analyze > Fit Model**. Figure 2.3 shows an example of the launch window for the Fitness.jmp sample data table. The Fit Model launch window is a common launch window for many platforms that are called Personalities. See "Fitting Personalities".

Figure 2.3 Fit Model Launch Window



Note: When you select **Analyze > Fit Model** in a data table that has a script named *Model* (or *model*), the launch window is automatically filled in based on the script.

The Fit Model launch window contains the following major areas:

- Select Columns is a list of the columns in the current data table. If the current data table contains excluded columns, they do not appear in the list. For more information about the options in the Select Columns red triangle menu, see *Using JMP*.
- Pick Role Variables contains standard JMP launch window buttons. For a description of these buttons, see "Fit Model Launch Window Elements".
- Construct Model Effects contains options that you use to enter effects into your model. See "Construct Model Effects".
- Personality is a list of the model types that you can choose from. Once you have selected a personality, different options appear, depending on the personality that you have selected. See "Fitting Personalities".

Fit Model Launch Window Elements

The following Fit Model launch window elements are common to most personalities:

Model Specification The Model Specification menu contains the following options:

Center Polynomials Centers the effects when polynomials are included in the model.

Informative Missing Creates a coding system that accommodates missing values in effects.

Set Alpha Level Sets the alpha level for confidence intervals in the model reports.

Save to Data Table Saves the model specifications to a script in the current data table.

Save to Script Window Saves the model specifications to a script window.

Create SAS Job Saves SAS code for the current model specification to a SAS Program window.

Submit to SAS Submits SAS code for the current model specification to a SAS session.

Convergence Settings Contains options for setting the maximum iterations and convergence limit used in the convergence of models in some personalities.

For more information about any of these options, see "Model Specification Options".

Select Columns Lists the unexcluded columns in the current data table.

Y Identifies one or more response variables (the dependent variables) for the model.

Note: Response columns with vector values are not supported in Fit Model.

Weight Identifies a column whose values assign a weight to each row for the analysis. See "Weight".

Freq Identifies a column whose values assign a frequency to each row for the analysis. In general terms, the effect of a frequency column is to expand the data table, so that any row with integer frequency *k* is expanded to *k* rows. You are allowed to specify fractional frequencies. See "Frequency".

JMP PRO **Validation** In JMP Pro, for some personalities, you can enter a Validation column. See the appropriate Personality chapter for more information about how each personality handles a Validation column. If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see *Predictive and Specialized Modeling*. For more information about how a Validation column is used in JMP modeling platforms, see *Predictive and Specialized Modeling*.

By Identifies a column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Add Adds effects to the model. See "Add".

Cross Creates interaction and polynomial effects by crossing two or more variables. See "Cross".

Nest Creates nested effects. See "Nest".

Macros Generates effects for commonly used models. See "Macros".

Degree Applies the specified degree to models with factorial or polynomial effects generated using Macros. See **Factorial to Degree** and **Polynomial to Degree** in "Macros".

Attributes Applies attributes to model effects. These attributes determine how the effects are treated. See "Attributes".

Transform Transforms selected continuous effects or *Y* columns. See "Transform".

No Intercept Excludes the intercept term from the model.

Personality Specifies the fitting methodology. See "Fit Model Launch Window Elements". Different options appear depending on the personality that you select.

Target Level (Available only in certain personalities and when *Y* is binary and has a nominal modeling type.) Specifies the level whose probability you want to model. The default value is the higher of the two levels based on the order of the levels.

Help Takes you to Help topics for the Fit Model launch window.

Recall Populates the launch window with the last model specification that you ran.

Remove Removes the selected variable from the assigned role. Alternatively, double-click the effect or select the effect and press Backspace.

Run Generates the report window for the specified model and personality.

Keep dialog open Keeps the launch window open after you run the analysis, enabling you to alter and re-run the analysis at any time.

Frequency

Frequency variables, entered in the **Freq** text box, are supported in most Fit Model personalities. In general, a frequency is interpreted in the following manner. Suppose that a row has a frequency *f*. Then the computed results are identical to those for a data table containing *f* copies of that row, each having a frequency of one.

Rows with zero or missing frequency values are excluded from analyses. Rows with negative frequency values are permitted only for censored observations, otherwise they are excluded from analyses. When used with censored observations, negative frequency values can be used to fit truncated distributions.

Frequency values do not need to be integers. The technical details describing how frequency columns, including those with non-integer values, are handled are given in "Frequencies".

Weight

Weight variables can be useful in situations where there are observations with different variances. For example, this can happen when one performs regression modeling on data where each row consists of pre-summarized means. Here, rows involving a larger number of observations (smaller variance) should contribute more heavily to the loss function than rows involving a smaller number of observations (larger variance). You can ensure that this occurs by using appropriately defined weights.

Weight variables are supported in many Fit Model personalities. Each personality that supports weight variables uses one of the following methods:

- Variance Scaling
- Frequency Symmetry

Variance Scaling

When estimation is performed using least squares or normal theory maximum likelihood, the weight w for a given row scales that row's contribution to the loss function by $w^{-1/2}$.

Weight variables have an impact on estimates and standard errors. However, unlike frequency variables, they do not affect the degrees of freedom used in hypothesis tests.

Rows with negative or zero values for Weight are excluded from analyses.

Frequency Symmetry

In the Nominal Logistic and Ordinal Logistic personalities, weight variables are handled as if they were frequency variables. If weight and frequency variables are both specified, these personalities handle each observation as if it has a frequency value equal to the product of the weight and frequency values.

Construct Model Effects

This section describes the options that you can use to facilitate entering effects into your model. Examples of how these options can be used to obtain specific types of models are given in "Examples of Model Specifications and Their Model Fits".

Add

Adds effects to the model. These effects can either be added directly from the Select Columns list or they can be selected in that list and modified using Macros or Attributes. Effects can also be created and added, or modified, using Cross and Nest. The modeling types of the variables involved in the effect, as well as any Attribute assigned to the effect, determine how that effect is treated in the model.

Note: To remove an effect from the Construct Model Effects list, double-click the effect, or select it and click **Remove** or press Backspace or Delete.

Cross

Creates interaction or polynomial effects. Select two or more variables in the Select Columns list and click **Cross**. Or, select one or more variables in the Select Columns list and one or more effects in the Construct Model Effects list and click **Cross**.

See "Statistical Details", for a discussion of how crossed effects are parameterized and coded.

Note: You can construct effects that combine up to ten columns as crossed and nested.

Example of Crossed Effects

Suppose that a product coating requires a dye to be applied. Both Dye pH and Dye Concentration are suspected to have an effect on the coating color. To understand their effects, you design an experiment where Dye pH and Dye Concentration are each set at a high and low level. It is possible that the effect of Dye pH on the color is more pronounced at the high level of Dye Concentration than at its low level. This is known as an *interaction*. To model this possible interaction, you include the crossed term, Dye pH * Dye Concentration, in the Construct Model Effects list. This enables JMP to test for an interaction.

Nest

Creates nested effects. If the levels of one effect (B) occur only within a single level of another effect (A), then B is said to be *nested* within A. The notation B[A], which is read as "B nested within A," is typically used. Note that nesting defines a hierarchical relationship. A is called the *outside* effect and B is called the *inside* effect. Nested terms must be categorical.

Note: The nesting terms must be specified in order from outer to inner. For example, if B is nested within A, and C is nested within B, then the model is specified as A, B[A], C[B,A] (or, equivalently, A, B[A], C[A,B]). You can construct effects that combine up to ten columns as crossed and nested.

Example of Nested Effects

As an illustration of nesting, consider the math teachers in each of two schools. One school has three math teachers; the other school has two math teachers. Each teacher in each school teaches two or three classes consisting of non-overlapping groups of students. In this example, classes (C) are nested within teachers (B), and teachers (B) are nested within schools (A). Enter these effects in the Fit Model launch window:

1. Add both A and B to the Construct Model Effects panel.
2. In the Construct Model Effects panel, select B.
3. In the Select Columns list, select A.
4. Click **Nest**. This converts B to the effect B[A].
5. Add C to the Construct Model Effects panel.
6. In the Construct Model Effects panel, select C.
7. In the Select Columns list, select A and B.
8. Click **Nest**. The converts C to the effect C[A, B].

Macros

In the Macros list, select options to automatically generate the effects for commonly used models and enter them into the Construct Model Effects list:

Full Factorial Creates all main effects and interactions for the columns selected in the Select Columns list. These are entered in an order that is based on the order in which the main effects are listed in the Select Columns list. For an alternate ordering, see **Factorial Sorted**, in this table.

Factorial to Degree Creates all main effects, but only interactions up to a specified degree (order). Specify the degree in the **Degree** box beneath the **Macros** button.

Factorial Sorted Creates the same set of effects as the **Full Factorial** option but lists them in order of degree. All main effects are listed first, followed by all two-way interactions, then all three-way interactions, and so on.

Response Surface Creates main effects, two-way interactions, and quadratic terms. The selected main effects are given the response surface attribute, denoted **RS**. When the RS attribute is applied to main effects and the Standard Least Squares personality is selected, a Response Surface report is provided. This report gives information about critical values and the shape of the response surface.

See also Response Surface Effect in "Attributes" and the *Design of Experiments Guide*.

Mixture Response Surface Creates main effects and two-way interactions. Main effects have the response surface (**RS**) and mixture (**Mixture**) attributes. In the Standard Least Squares

personality, the **Mixture** attribute causes a mixture model to be fit. The **RS** attribute creates a Response Surface report that is specific to mixture models.

See also Mixture Effect in "Attributes" and the *Design of Experiments Guide*.

Polynomial to Degree Creates main effects and polynomial terms up to a specified degree. Specify the degree in the **Degree** box beneath the **Macros** button.

Scheffé Cubic Creates main effects, interactions, and Scheffé cubic terms, which are useful in specifying response surfaces for mixture experiments. This macro creates a complete cubic model.

When you fit a third-degree polynomial model to a mixture, you must not introduce even-powered terms, for example, X1*X1*X2, because they are not estimable. However, it turns out that a complete polynomial specification of the surface can include terms of the form X1*X2*(X1 – X2), which are called *Scheffé cubic* terms.

Scheffé cubic terms are also included if you enter a 3 in the **Degree** box and then select the **Mixture Response Surface** macro command.

JMP PRO **Grouped Regressors** Creates a single effect for a set of continuous factors that are treated as one effect in various parts of the report. You can add or remove the grouped effect in the Effect Summary. There is one leverage plot and one effect test for each group effect, rather than individual plots and tests for each factor in the group. This can be useful if your data table contains indicator columns that represent levels of a categorical effect.

Note: Non-continuous factors included in a group of regressors are not included in the analysis.

Attributes

In the Attributes list, select attributes that you can assign to an effect selected in the Construct Model Effects list.

Random Effect Assigns the Random attribute to an effect. For more information about random effects, see "Specifying Random Effects and Fitting Method".

Response Surface Effect Assigns the **RS** attribute to an effect. Note that the relevant model terms must be included in the Construct Model Effects list. The Response Surface option in the Macros list automatically generates these terms and assigns the RS attribute to the main effects. To obtain the Response Surface report, interaction and polynomial terms do not need to have the RS attribute assigned to them. You need only assign this attribute to main effects.

LogVariance Effect Assigns the LogVariance attribute to an effect. This attribute indicates that the effect is to be included in a model of the variance of the response.

To include an effect in models for *both* the mean and variance of the response, you must specify the effect twice. In the tabbed interface, it must appear on both the Mean Effects and Variance Effects tabs. Otherwise, you can enter it twice on the Mean Effects tab, once without the LogVariance Effect attribute and once with the LogVariance Effect attribute.

Mixture Effect Assigns the Mixture attribute to main effects. This is used to specify the main effects involved in the mixture. Note that the Mixture Response Surface option in the Macros list automatically assigns the mixture attribute to selected effects, and provides a Response Surface report when possible.

Excluded Effect Assigns the Excluded attribute to an effect. This excludes the effect from the model fit. However, the effect is used to group observations for lack-of-fit tests. In the Standard Least Squares personality, a table of least square means is provided for this effect.

Knotted Spline Effect Assigns the Knotted attribute to a continuous main effect. This implicitly adds cubic splines for the effect to the model specification. See "Knotted Spline Effect".

Knotted Spline Effect

Knotted splines are used to fit a response Y using a flexible function of a predictor. Consider the single predictor X. When the Knotted Spline Effect is assigned to X, and k knots are specified, then k - 2 additional effects are implicitly added to the set of predictors. Each of these effects is a piecewise cubic polynomial spline whose segments are defined by the knots. See Stone and Koo (1985).

The number of splines is determined by the number of knots, which you are asked to specify. The coefficients associated with the splines are estimated based on the method used by the personality.

The placement of knots follows guidance given in the literature. In particular, if there are 100 or fewer points, the first and last knots are the fifth smallest and largest points, respectively. Otherwise, the first and last knots are placed at the 0.05 and 0.95 quantiles for 5 or fewer knots, or the 0.025 and 0.975 quantiles for more than 5 knots. The default number of knots is 5 for more than 30 observations, and 3 for fewer than 30 observations.

Note: Knotted splines are implemented only for main-effect continuous terms.

Transform

The Transform options transform selected Y columns or main effects that are selected in the Construct Model Effects text box.

Note: You can also transform a column by right-clicking it in the Select Columns list and selecting Transform. A reference to the transformed column appears in the Select Columns list. You can then use the column in the Fit Model window as you would any data table column. See *Using JMP*.

None Removes any Transform options that have been applied.

Log Applies the natural logarithm transformation to the selected variable.

Sqrt Applies the square root of the values of the selected variable.

Square Applies the square of the values of the selected variable.

Reciprocal Applies the transformation 1/X to the variable X.

Exp Applies the exponential transformation to the selected variable.

Arrhenius Applies the Arrhenius transformation to the variable T (temperature in degrees Centigrade):

$$X = \frac{11604.5181215503}{T + 273.15}$$

This is the component of the Arrhenius relationship that is multiplied by the activation energy.

ArrheniusInv Applies the inverse of the Arrhenius transformation to the variable X:

$$T = \frac{11604.5181215503}{X} - 273.15$$

Logit Calculates the inverse of the logistic function for the selected column (where p is in the range of 0 to 1):

$$\text{Logit}(p) = \log\left(\frac{p}{1-p}\right)$$

Logistic Calculates the logistic (also known as Squish and Logist) function for the selected column (where the result is in the range of 0 to 1):

$$\text{Logistic}(x) = \frac{1}{(1 + e^{-x})}$$

LogitPct Calculates the logit as a percent for the selected column (where *pct* is a percent in the range of 0 to 100):

$$\text{LogitPct}(pct) = \log\left(\frac{\left(\frac{pct}{100}\right)}{1-\left(\frac{pct}{100}\right)}\right)$$

LogisticPct Calculates the logistic (or logist) as a percent for the selected column (where the result is in the range of 0 to 100):

$$\text{LogisticPct}(x) = \frac{100}{(1+e^{-x})}$$

No Intercept

Select No Intercept if you want to fit a model with no intercept term. Certain modeling structures require no intercept models. For these, the No Intercept box is checked by default.

Construct Model Effects Tabs

For the following personalities, you can enter model effects using a tabbed interface:

Note: If you apply Attributes to effects on the first (main) tab, the attributes determine how the effects are treated in the model. If you run the model and then request Model Dialog from the report's red triangle menu, you find that those effects appear on the appropriate tabs.

Standard Least Squares Enter model effects:

Fixed Effects tab Enter effects to be modeled as fixed effects. A fixed effect is one whose specific treatment levels are of interest. You want to compare the mean response across its treatment levels.

Random Effects tab Enter effects to be modeled as random effects. A random effect is one whose levels are considered a random sample from a larger population. You want to estimate the variation in the response that is attributable to this effect.

Mixed Model Enter model effects:

Fixed Effects tab Enter effects to be modeled as fixed effects. See Standard Least Squares in this table.

Random Effects tab Enter effects to be modeled as random effects. Use for variance component models and random coefficients models.

Repeated Structure tab Use to select a covariance structure for repeated effects.

Loglinear Variance Enter model effects:

Mean Effects tab Enter effects for which you want to model expected values.

Variance Effects tab Enter effects for which you want to model variance.

If you want to model both the expected value and variance of an effect, you must enter it on both tabs.

Parametric Survival Enter model effects:

Location Effects tab Enter effects that you want to use in modeling the location parameter, mu, or in the case of the Weibull distribution, the shape parameter.

Scale Effects tab Enter effects that you want to use in modeling the scale parameter.

Fitting Personalities

In the Fit Model launch window, you select your fitting and analysis method by specifying a Personality. Based on the response (or responses) and the factors that you enter, JMP makes an initial context-based guess at the desired personality, but you can alter this selection in the Personality menu.

The following fitting personalities are available:

Standard Least Squares Fits models where the response is continuous. Techniques include regression, analysis of variance, analysis of covariance, mixed models, and analysis of designed experiments. See "Standard Least Squares Models" and "Emphasis Options for Standard Least Squares".

Stepwise Facilitates variable selection for standard least squares and ordinal logistic analyses (or nominal with a binary response). For continuous responses, cross validation, *p*-value, BIC, and AICc criteria are provided. Also provided are options for fitting all possible models and for model averaging. For logistic fits, *p*-value, BIC, and AICc criteria are provided. See "Stepwise Regression Models".

JMP PRO **Generalized Regression** Fits generalized linear models using regularized, also known as penalized, regression techniques. The regularization techniques include ridge regression, the lasso, the adaptive lasso, the elastic net, and the adaptive elastic net. The response distributions include the normal, binomial, Poisson, zero-inflated Poisson, negative binomial, zero-inflated negative binomial, and gamma. See "Generalized Regression Models" and "Distribution".

JMP PRO **Mixed Model** Fits a wide variety of linear models for continuous-responses with complex covariance structures. The situations addressed include:

- Split plot experiments
- Random coefficients models
- Repeated measures designs
- Spatial data
- Correlated response data

See "Mixed Models".

Manova Fits models that involve multiple continuous Y variables. Techniques include multivariate analysis of variance, repeated measures, discriminant analysis, and canonical correlations. See "Multivariate Response Models".

Loglinear Variance For a continuous Y variable, constructs models for both the mean and the variance. You can specify different sets of effects for the two models. See "Loglinear Variance Models".

Nominal Logistic Fits a logistic regression model to a nominal response. See "Logistic Regression Models".

Ordinal Logistic Fits a logistic regression model to an ordinal response. See "Logistic Regression Models".

Proportional Hazard Fits a semiparametric regression model (the Cox proportional hazards model) to assess the effect of explanatory variables on survival times, taking censoring into account.

You can also launch this personality by selecting **Analyze > Reliability and Survival > Fit Proportional Hazards**. See *Reliability and Survival Methods*.

Parametric Survival Fits a general linear regression model to survival times. Use this option if you have survival times that can be expressed as a function of one or more explanatory variables. Takes into account various survival distributions and censoring.

You can also launch this personality by selecting **Analyze > Reliability and Survival > Fit Parametric Survival**. See *Reliability and Survival Methods*.

Generalized Linear Model Fits generalized linear models using various distribution and link functions. Techniques include logistic, Poisson, and exponential regression. See "Generalized Linear Models".

JMP PRO **Partial Least Squares** Fits models to one or more *Y* variables using latent factors. This permits models to be fit when explanatory variables (*X* variables) are highly correlated, or when there are more *X* variables than observations.

You can also launch a partial least squares analysis by selecting **Analyze > Multivariate Methods > Partial Least Squares**. See *Multivariate Methods*.

Response Screening Automates the process of conducting tests for linear model effects across a large number of responses. Test results and summary statistics are presented in data tables and plots. A False-Discovery Rate (FDR) approach guards against incorrect declarations of significance. A robust estimation method reduces the sensitivity of tests to outliers.

Note: This personality allows only continuous responses. Response Screening for individual factors is also available by selecting **Analyze > Screening > Response Screening**. This platform supports categorical responses, and also provides equivalence tests and tests of practical significance. See *Predictive and Specialized Modeling*.

Model Specification Options

The Model Specification red triangle menu contains the following options:

Center Polynomials Centers by its mean any continuous term that is involved in an effect with a degree greater than one. This option is checked by default, except when a term involved in the effect is assigned the Mixture Effect attribute or has the Mixture column property. Terms with the Coding column property are centered midway between their specified High and Low values.

Centering is useful in making regression coefficients more interpretable and in reducing collinearity between low-order and high-order effects.

Informative Missing Provides a coding system for missing values. This system allows estimation of a predictive model despite the presence of missing values. It is useful in situations where missing data are informative. See "Informative Missing".

This option is available for the following personalities: Standard Least Squares, Stepwise, Generalized Regression, MANOVA, Loglinear Variance, Nominal Logistic, Ordinal Logistic, Proportional Hazard, Parametric Survival, Generalized Linear Model, and Response Screening.

Set Alpha Level Sets the alpha level for confidence intervals in the Fit Model analysis. The default alpha level is 0.05.

Error Specification (Available only for the Standard Least Squares personality when there are no random effects.) Specifies the error variance and the error degrees of freedom that are used for standard errors and tests in the Fit Least Squares report. Note that the Studentized Residuals plot and the Box Cox Transformations report are not affected by changing the Error Specification. When the Error Specification is Pure Error or Specified,

an additional column appears in the Analysis of Variance report. See "Analysis of Variance".

Default Estimate Uses the standard root mean square error and error degrees of freedom from the model to calculate all tests and standard errors.

Pure Error Uses the Pure Error mean square and associated degrees of freedom from the Lack of Fit report to calculate all tests and standard errors. See "Lack of Fit".

Caution: If the pure error degrees of freedom is 1, a warning message is displayed indicating that tests are weak and confidence limits are large.

Specified Uses user-specified values for the error variance and error degrees of freedom to calculate all tests and standard errors.

Save to Data Table Saves your Fit Model launch window specifications as a script that is attached to the data table. The script is named Model. When a table contains a script called Model, this script automatically populates the launch window when you select **Analyze > Fit Model**. (Simply rename the script if this is not desirable.)

For more information about JSL scripting, see the *Scripting Guide*.

Save to Script Window Copies your Fit Model launch window specifications to a script window. You can save the script window and re-create the model at any time by running the script.

Create SAS job Creates a SAS program that can re-create the current analysis and data table in SAS in a script window. Once created, you have several options for submitting the code to SAS.

1. Copy and paste the code into the SAS Program Editor. This method is useful if you are running an older version of SAS (pre-version 8.2).
2. Select **Edit > Submit to SAS**.
3. Save the file and double-click it to open it in a local copy of SAS. This method is useful if you would like to take advantage of SAS ODS options, such as generating HTML or PDF output from the SAS code.

See *Using JMP*.

Submit to SAS Submits code to SAS and displays the results in JMP. If you are not connected to a SAS server, prompts guide you through the connection process.

See *Using JMP*.

Convergence Settings The Convergence Settings menu contains the following options:

Maximum Iterations Specifies the maximum number of iterations that are used in the model fitting. By default, the maximum number of iterations is 100. If your model does not readily converge, you might want to increase the number of iterations. If you have a very large data set or a complicated model, you might want to limit the number of iterations.

Convergence Limit Specifies the convergence limit for the model fitting. If you model does not readily converge, you might want to increase the convergence limit. By default, the convergence limit is 0.00000001.

Note: The Convergence Settings menu appears only for certain personalities. In the Standard Least Squares personality, the Convergence Settings menu appears only when there is a random effect and REML is selected as the Method in the launch window.

Informative Missing

The Informative Missing option constructs a coding system that allows estimation of a predictive model despite the presence of missing values. It codes both continuous and categorical model effects.

Continuous Effects

When a continuous main effect has missing values, a new design matrix column is created. This column is an indicator variable, with values of one if the main effect column is missing and zero if it is not missing. In addition, missing values for the continuous main effect are replaced with the mean of the nonmissing values for rows included in the analysis. The mean is a neutral value that maintains the interpretability of parameter estimates.

The parameter associated with the indicator variable estimates the difference between the response predicted by the missing value grouping and the predicted response if the covariate is set at its mean.

For a higher-order effect, missing values in the covariates are replaced by the covariate means. This makes the higher-order effect zero for rows with missing values, assuming that Center Polynomials is checked (the default setting). This is because Center Polynomials centers the individual terms involved in a polynomial by their means.

In the Effect Tests report, each continuous main effect with missing values has Nparm = 2. In the Parameter Estimates report, the parameter for a continuous main effect with missing values is labeled <colname> Or Mean if Missing and the indicator parameter is labeled <colname> Is Missing. Prediction formulas that you save to the data table are given in terms of expressions corresponding to these model parameters.

Categorical Effects

When a nominal or ordinal main effect has missing values, the missing values are coded as a separate level of that effect. As such, in the Effect Tests report, each categorical main effect with missing values has one additional parameter.

In the Parameter Estimates report, the parameter for a nominal effect is labeled <colname>[ ]. For an ordinal effect, the parameter is labeled <colname>[-x], where x denotes the level with highest value ordering.

As with continuous effects, prediction formulas that you save to the data table are given in terms of expressions corresponding to the model parameters.

Coding Table

When you are using the Standard Least Squares personality, you can view the design matrix columns used in the Informative Missing model by selecting **Save Columns > Save Coding Table.**

Validity Checks

Fit Model checks your model for errors such as duplicate effects or missing effects in a hierarchy. If you receive an alert message, you can either click **Continue** to proceed with fitting, or click **Cancel** to stop the fitting process.

Examples of Model Specifications and Their Model Fits

This section gives templates for entering the effects for various model types that you can specify using the Construct Model Effects panel in the Fit Model platform.

- The model effects X and Z represent continuous columns.
- The model effects A, B, and C represent nominal or ordinal columns.

For most models, visual views of their model fits are also given.

- "Simple Linear Regression"
- "Polynomial in X to Degree k"
- "Polynomial in X and Z to Degree k"
- "Multiple Linear Regression"
- "One-Way Analysis of Variance"
- "Two-Way Analysis of Variance"
- "Two-Way Analysis of Variance with Interaction"
- "Three-Way Full Factorial"
- "Analysis of Covariance, Equal Slopes"
- "Analysis of Covariance, Unequal Slopes"
- "Two-Factor Nested Random Effects Model"
- "Three-Factor Fully Nested Random Effects Model"
- "Simple Split Plot or Repeated Measures Model"
- "Two-Factor Response Surface Model"
- "Knotted Spline Effect"

Simple Linear Regression

Effects to be entered: X

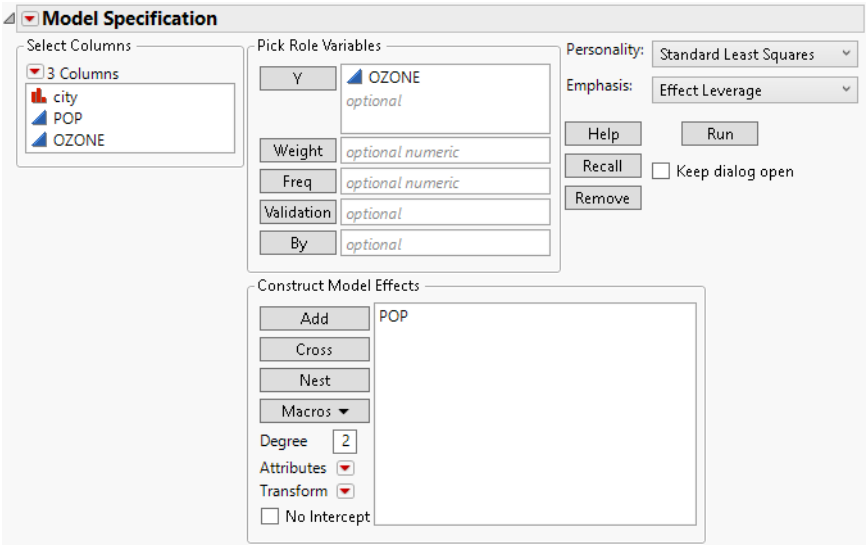
1. In the Select Columns list, select X.
2. Click **Add**.

Example of Simple Linear Regression Model

You are interested in the relationship between the population in thousands of the given city, and ozone levels. Ozone is the response of interest, and POP is the continuous model effect.

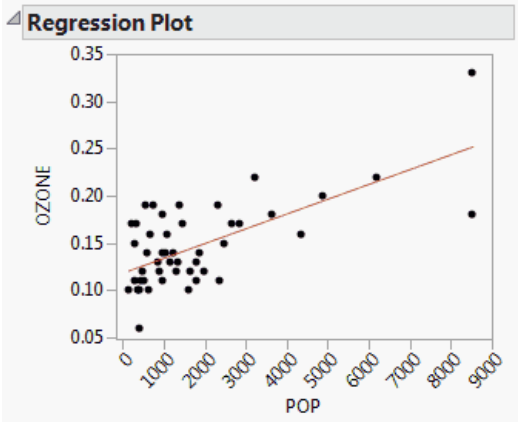
1. Select **Help > Sample Data Library** and open Polycity.jmp.
2. Select **Analyze > Fit Model**.
3. In the Select Columns list, select Ozone and click Y.
4. In the Select Columns list, select POP.
5. Click **Add**.

Figure 2.4 Fit Model Window for Simple Linear Regression



When you click **Run**, the Fit Least Squares report appears, showing various results, including a Regression Plot. The Regression Plot shows the data and a simple linear regression model fit to the data.

Figure 2.5 Model Fit for Simple Linear Regression



Polynomial in X to Degree k

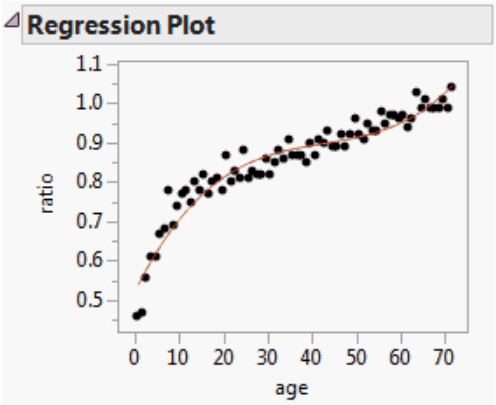
Effects to be entered: X, X*X,..., X^k

1. Type *k* into the text box for **Degree**.
2. In the Select Columns list, select X.

3. Select **Macros > Polynomial to Degree**.

Figure 2.6 shows a plot of the data and a cubic polynomial model fit to the data for the Growth.jmp sample data table. This is one of the fits produced when you run the **Bivariate** data table script.

Figure 2.6 Model Fit for a Degree-Three Polynomial in One Variable



Polynomial in X and Z to Degree k

Effects to be entered: X, X*X,..., X^k, Z, Z*Z,..., Z^k

1. Type *k* into the text box for **Degree**.
2. In the Select Columns list, select X and Z.
3. Select **Macros > Polynomial to Degree**.

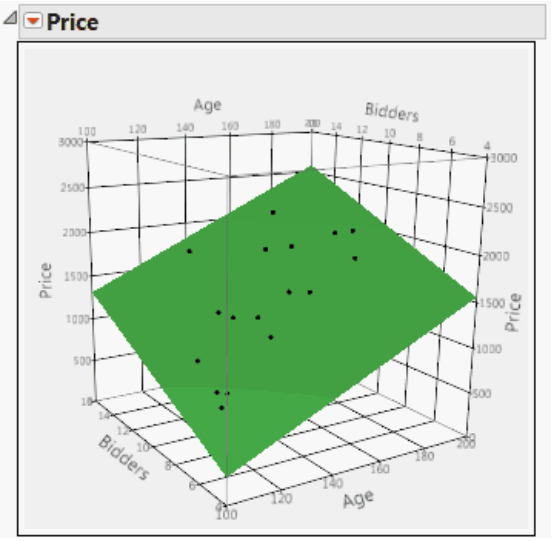
Multiple Linear Regression

Effects to be entered: Selected columns

1. In the Select Columns list, select the continuous effects of interest.
2. Click **Add**.

Figure 2.7 shows a surface profiler plot of the data and of the multiple linear regression fit to the data for the Grandfather Clocks.jmp sample data table. The model effects are Age and Bidders. The response is Price. You can obtain the plot by running the data table script **Fit Model with Surface Profiler Plot**.

Figure 2.7 Model Fit for a Multiple Linear Regression Model with Two Predictors



Example of Multiple Linear Regression Model

See also "Example of a Regression Analysis Using Fit Model" for an example with several predictors.

One-Way Analysis of Variance

Effects to be entered: A

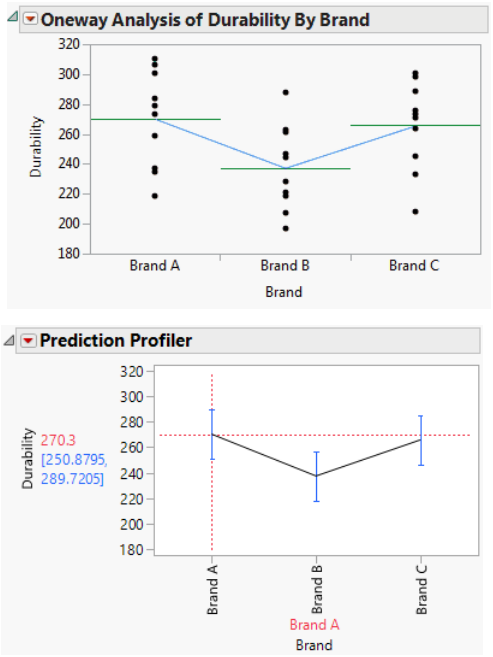
1. In the Select Columns list, select one nominal or ordinal effect, A.
2. Click **Add**.

Consider the Golf Balls.jmp sample data table. You are interested in whether Durability varies by Brand. Figure 2.8 shows two plots.

The first is a plot, obtained using Fit Y by X, that shows the data by brand. Horizontal lines are plotted at the mean for each brand and line segments connect the means. To produce this plot, run the script **Oneway: Durability by Brand** in the Golf Balls.jmp sample data table.

The second plot is a profiler plot obtained using Fit Model. This second plot shows the predicted responses for each brand, connected by line segments. To produce this plot, run the script Fit Model: Durability by Brand in the Golf Balls.jmp sample data table. Drag the vertical dashed red line to the brand of interest. The horizontal dashed red line updates to intersect the vertical axis at the predicted response.

Figure 2.8 Model Fit for One-Way Analysis of Variance



Two-Way Analysis of Variance

Effects to be entered: A, B

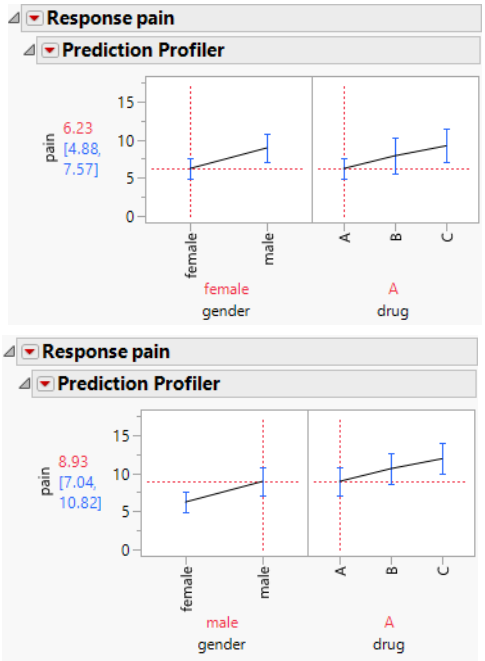
1. In the Select Columns list, select two nominal or ordinal effects, A and B.
2. Click **Add**.

Figure 2.9 shows two profiler plots of the fit to the data for the Analgesics.jmp sample data table. The model effects are gender and drug. The response is pain. To obtain this plot, select **Analyze > Fit Model**, select pain as Y, select gender and drug as model effects, and then click **Run**. Click the Response Pain red triangle and select **Factor Profiling > Profiler.**

The line segments in each plot connect the predicted values for the settings defined by the vertical dashed red lines. Move these to see predictions at other settings.

The top plot in Figure 2.9 shows predictions for females, whereas the bottom plot shows predictions for males. Note that the relative effects of the three drugs are consistent across the levels of gender. This is because there is no interaction term in the model. For an example with interaction, see Figure 2.11.

Figure 2.9 Model Fit for a Two-Way Analysis of Variance with No Interaction



Two-Way Analysis of Variance with Interaction

Effects to be entered: A, B, A*B

1. In the Select Columns list, select two nominal or ordinal effects, A and B.
2. Select **Macros > Full Factorial**.

Or:

1. In the Select Columns list, select two nominal or ordinal effects, A and B.
2. Click **Add**.
3. In the Select Columns list, select A and B again and click **Cross**.

Example of Two-Way Analysis of Variance with Interaction

You have data about the yield of popcorn and are interested in whether the type of popcorn and the size of the batch have an effect on yield.

1. Select **Help > Sample Data Library** and open Popcorn.jmp.
2. Select **Analyze > Fit Model**.
3. In the Select Columns list, select yield and click **Y**.
4. In the Select Columns list, select popcorn and batch.

5. Select **Macros > Full Factorial**.

Figure 2.10 Fit Model Window for Two-Way Analysis of Variance with Interaction

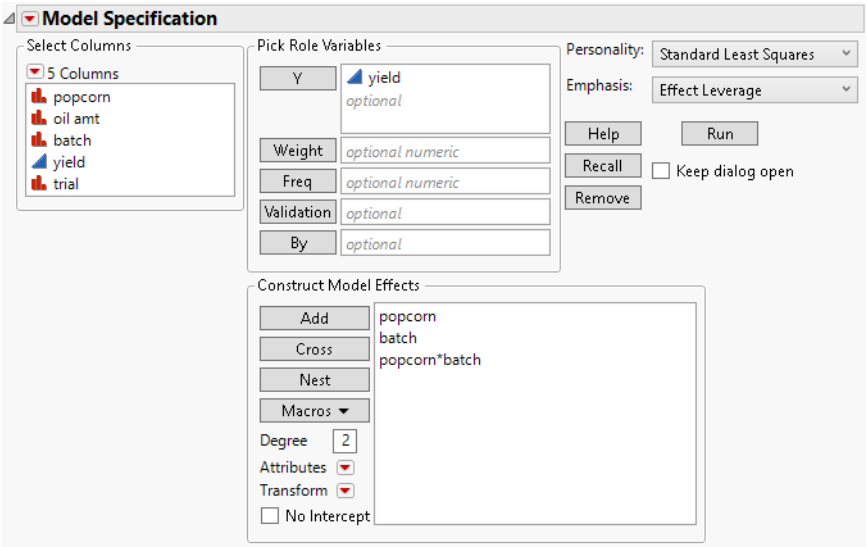
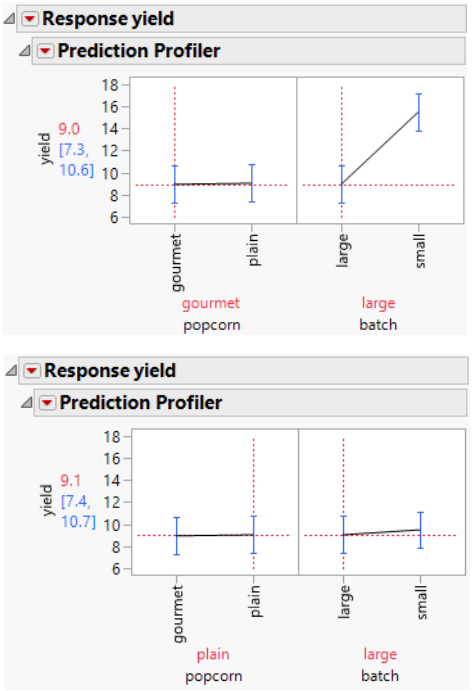


Figure 2.11 shows a profiler plot of the fit for this example. To obtain this plot, click **Run** in the Fit Model window shown in Figure 2.10. Then, click the Response yield red triangle and select **Factor Profiling > Profiler**.

In the top plot, popcorn is set to gourmet, and in the bottom plot, it is set to plain. Note how the predicted values for the settings of batch depend on the type of popcorn. This is a consequence of the interaction between popcorn and batch.

Figure 2.11 Model Fit for a Two-Way Analysis of Variance with Interaction



Three-Way Full Factorial

Effects to be entered: A, B, C, A*B, A*C, B*C, A*B*C

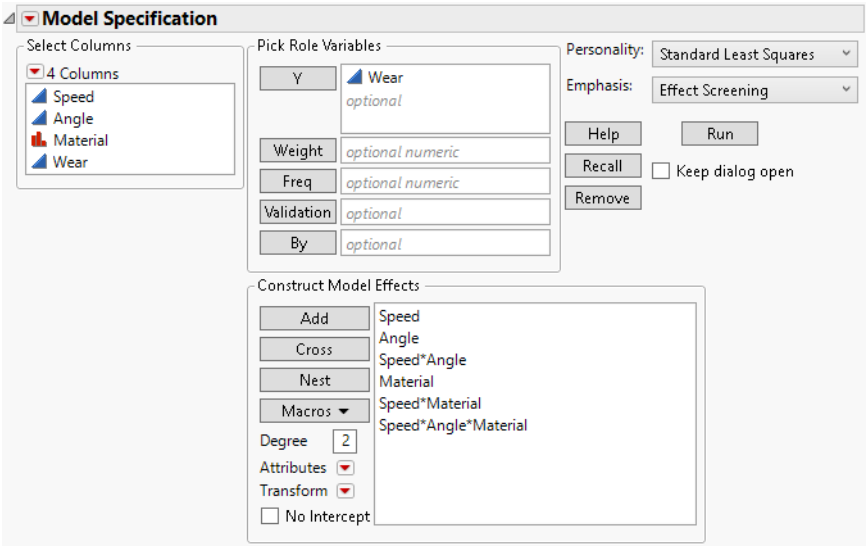
1. In the Select Columns list, select three nominal or ordinal effects, A, B, and C.
2. Select **Macros > Full Factorial**.

Example of Three-Way Full Factorial

You are interested in whether Speed, Angle, and Material, or their interactions, have an effect on the Wear of a cutting tool.

1. Select **Help > Sample Data Library** and open Tool Wear.jmp.
2. Select **Analyze > Fit Model**.
3. In the Select Columns list, select Wear and click **Y**.
4. In the Select Columns list, select Speed, Angle, and Material.
5. Select **Macros > Full Factorial**.

Figure 2.12 Fit Model Window for Three-Way Full Factorial

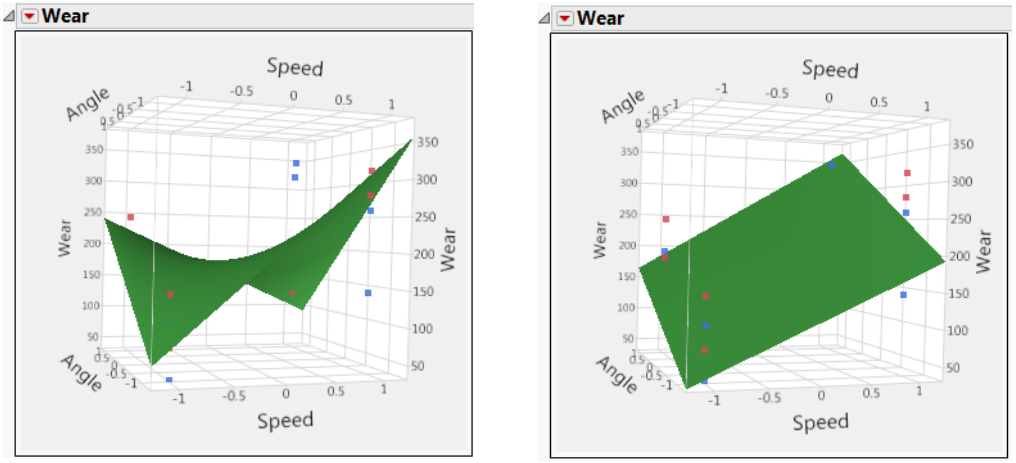


The Surface Profiler plots in Figure 2.13 show the predicted response for Wear in terms of the two continuous effects Speed and Angle. The plot on the left shows the predicted response when Material is A; the plot on the right shows the predicted response when Material is B. The points for which Material is A are colored red, whereas those for which Material is B are colored blue. The difference in the form of the response surfaces across the levels of Material is a consequence of the three-way interaction.

To obtain Surface Profiler plots, click **Run** in the Fit Model window shown in Figure 2.12. Click the Response Wear red triangle and select **Factor Profiling > Surface Profiler**. To add points to the plot, open the Appearance panel and click **Actual**. If you want to make the points appear larger, right-click in the plot, select **Settings** and adjust the **Marker Size**.

To show plots for both Materials A and B, use the slider marked Material in the Independent Variables panel, setting it at 0 for Material A and 1 for Material B. Note that the table contains two data table scripts that produce Surface Profiler plots: **Prediction and Surface Profilers** and **Surface Profilers for Two Materials**.

Figure 2.13 Model Fit for a Three-Way Full Factorial Design - Material A on Left, Material B on Right



Analysis of Covariance, Equal Slopes

Here you are interested in testing for the effect of A with X as a covariate. Suppose that you have reason to believe that the effect of X on the response does not depend on the level of A.

Effects to be entered: A, X

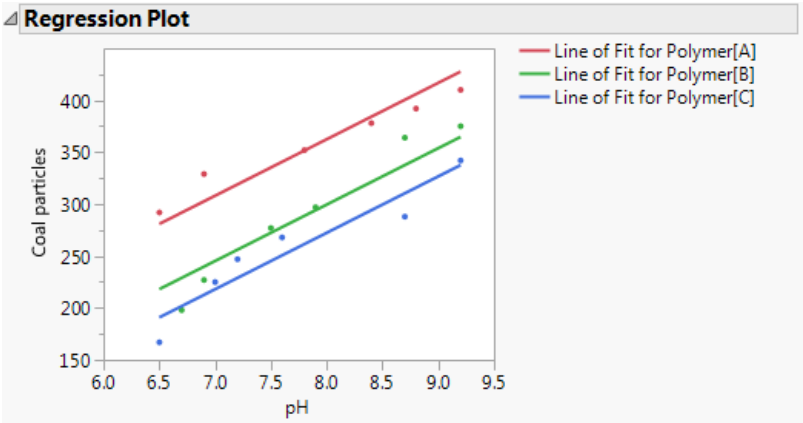
1. In the Select Columns list, select one nominal or ordinal effect, A, and one continuous effect, X.
2. Click **Add**.

Figure 2.14 shows the data from the Cleansing.jmp sample data table. You are interested in which Polymer removes the most Coal particles from a cleansing tank. However, you suspect that the pH of the tank also has an effect on removal. The plot shows an analysis of covariance fit. Here the slopes relating pH and Coal particles are assumed equal across the levels of Polymer.

You can obtain the plot in Figure 2.14 using the following steps. In the Fit Model window, enter Coal particles as **Y** and both pH and Polymer in the **Construct Model Effects** box. Click **Run**. The Regression Plot appears in the Fit Least Squares report. To color the points, select **Rows > Color or Mark by Column**, select Polymer from the **Mark by Column** list, and click **OK**.

However, a more complete analysis indicates that pH and Polymer *do* interact in their effect on Coal particles. The appropriate model fit is shown in "Analysis of Covariance, Unequal Slopes".

Figure 2.14 Model Fit for Analysis of Covariance, Equal Slopes



Analysis of Covariance, Unequal Slopes

Here you are again interested in testing for the effect of A with X as a covariate. But you construct your model so as to admit the possibility that the effect of X on the response depends on the level of A.

Effects to be entered: A, X, A*X

1. In the Select Columns list, select one nominal or ordinal effect, A, and one continuous effect, X.
2. Select **Macros > Full Factorial**.

Or:

1. In the Select Columns list, select one nominal or ordinal effect, A, and one continuous effect, X.
2. Click **Add**.
3. In the Select Columns list, select A and X again and click **Cross**.

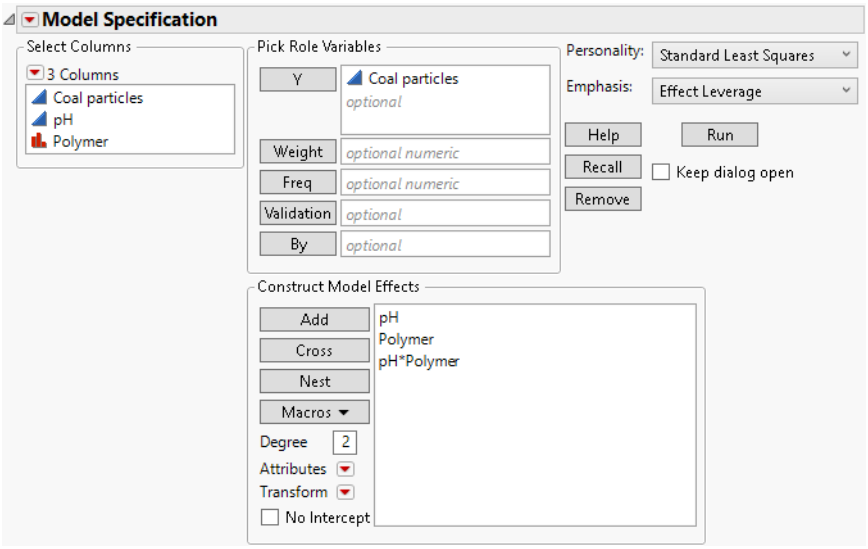
Example of Analysis of Covariance, Unequal Slopes

You are interested in whether any of the three polymers (Polymer) has an effect on coal particle removal (Coal particles). The tank pH is included as a covariate as it might affect a polymer's ability to clean the tank. You allow for possibly different slopes when modeling the relationship between pH and Coal particles for the three Polymer types.

1. Select **Help > Sample Data Library** and open Cleansing.jmp.
2. Select **Analyze > Fit Model**.
3. In the Select Columns list, select Coal particles and click **Y**.

4. In the Select Columns list, select pH and Polymer.
5. Select **Macros > Full Factorial**.

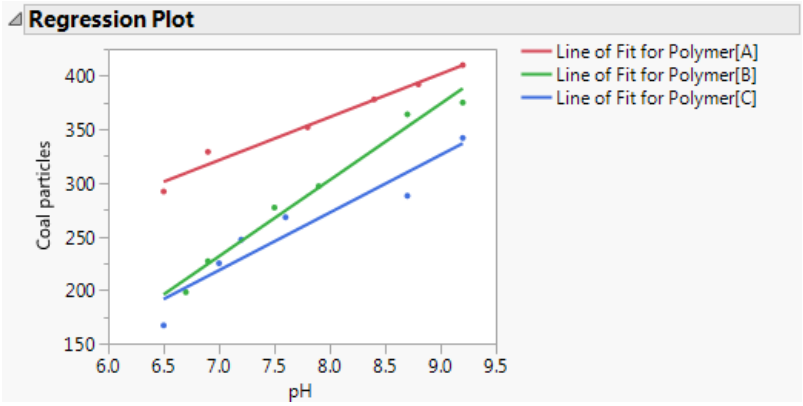
Figure 2.15 Fit Model Window for Analysis of Covariance, Unequal Slopes



When you click **Run**, the Fit Least Squares report appears. The Effect Tests report indicates that the interaction between pH and Polymer is significant and should be included in the model.

The Regression Plot given in the report is shown in Figure 2.16. This plot shows the points and the model fit. The interaction allows the slopes of the lines that relate pH to Coal particles to depend on the Polymer. Note that, despite this interaction, over the range of interest, Polymer A consistently has the highest removal. If you want to color the points as shown in Figure 2.16, select **Rows > Color or Mark by Column**, select Polymer from the **Mark by Column** list, and click **OK**.

Figure 2.16 Model Fit for Analysis of Covariance, Unequal Slopes



Two-Factor Nested Random Effects Model

Consider a model with two factors, A and B, but where B is nested within A. Although there are situations where a nested effect is treated as a fixed effect, in most situations a nested effect is treated as a random effect. For this reason, in the model described below, the nested effect is entered as a random effect.

Effects to be entered: A, B[A]&Random

1. In the Select Columns list, select two nominal or ordinal effects, A and B.
2. Click **Add**.
3. To nest B within A: In the Construct Model Effects list, select B. In the Select Columns list, select A. The two effects should be highlighted.
4. Click **Nest**.
5. With B[A] highlighted in the Construct Model Effects list, select **Attributes > Random Effect**.

Example of Two-Factor Nested Random Effects Model

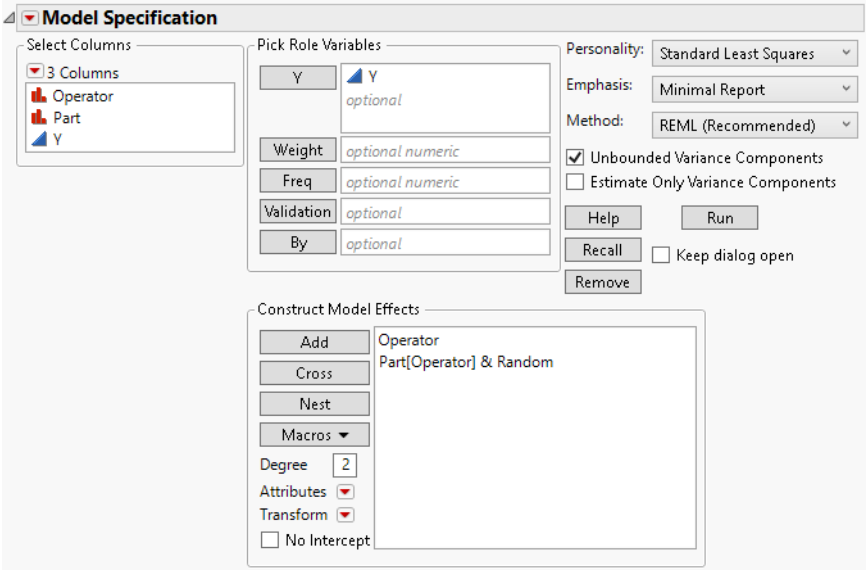
As part of a measurement systems analysis study, 24 randomly chosen parts are measured. These parts are evenly divided among the six operators who typically measure these parts. Each operator makes three independent measurements of each of the four assigned parts.

Since the parts measured by one operator are measured only by that specific operator, Part is nested within Operator. Since the parts are a random sample of production, Part is considered a random effect. Since these specific six operators are of interest, Operator is treated as a fixed effect. Specify the appropriate model.

1. Select **Help > Sample Data Library** and open Variability Data/2 Factors Nested.jmp.

2. Select **Analyze > Fit Model**.
3. In the Select Columns list, select Y and click **Y**.
4. In the Select Columns list, select Operator and Part.
5. Click **Add**.
6. To nest Part within Operator: In the Construct Model Effects list, select Part. In the Select Columns list, select Operator. The two effects should be highlighted.
7. Click **Nest**.
8. With Part[Operator] highlighted in the Construct Model Effects list, select **Attributes > Random Effect**.

Figure 2.17 Fit Model Window for Two-Factor Nested Random Effects Model



9. Click **Run** to obtain the Fit Least Squares report.

Figure 2.18 shows two plots. The first is a Variability Chart showing the three measurements by each Operator on each of the four parts. Horizontal line segments show the mean measurement for each Operator.

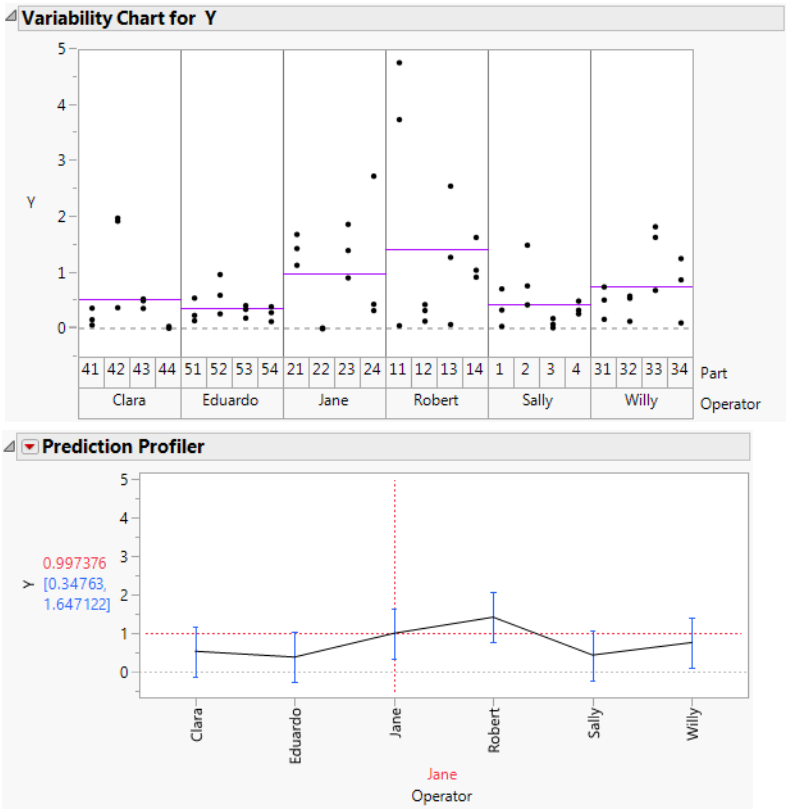
To construct the Variability Chart in Figure 2.18, in the 2 Factors Nested.jmp sample data table, run the data table script **Variability Chart - Nested**. Click the Variability Gauge red triangle, deselect **Show Range Bars**, and select **Show Group Means**.

The second plot is the Fit Least Squares report Prediction Profiler plot for Operator. This plot shows the predicted response for each operator. The vertical dashed red line set at Jane indicates that Jane's predicted response is 0.997. You can see the correspondence between the model predictions given in the Prediction Profiler plot and the raw data in the Variability Chart.

To obtain the Prediction Profiler plot, click the Response Y red triangle and select **Factor Profiling > Profiler**.

These plots show how the predicted measurements for each Operator are modeled. However, keep in mind that you are not only interested in whether the operators differ in how they measure parts. You are also interested in the variability of the part measurements themselves, which requires estimation of the variance component associated with Part.

Figure 2.18 Model Fit for Two-Factor Nested Random Effects Model



Three-Factor Fully Nested Random Effects Model

Consider a model with three factors, A, B, and C, but where B is nested within A and C is nested within both A and B. Also consider B and C to be random effects.

Effects to be entered: A, B[A]&Random, C[A,B]&Random

1. In the Select Columns list, select three nominal or ordinal effects, A, B, and C.
2. Click **Add**.
3. To nest B within A: In the Construct Model Effects list, select B. In the Select Columns list, select A. The two effects should be highlighted.
4. Click **Nest**.
5. To nest C within A and B: In the Construct Model Effects list, select C. In the Select Columns list, select A and B. The three effects should be highlighted.
6. Click **Nest**.
7. With both B[A] and C[A,B] highlighted in the Construct Model Effects list, select **Attributes > Random Effect**.

Simple Split Plot or Repeated Measures Model

Here A is the whole plot variable, B[A] is the whole plot ID, and C is the split plot, or repeated measures, variable.

Effects to be entered: A, B[A]&Random, C, C*A

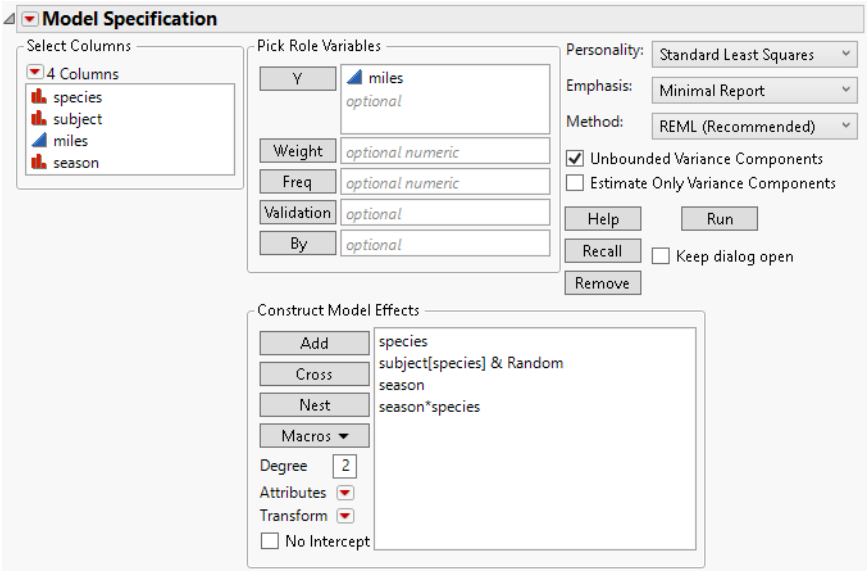
1. In the Select Columns list, select two nominal or ordinal effects, A and B.
2. Click **Add**.
3. To nest B within A: In the Construct Model Effects list, select B. In the Select Columns list, select A. The two effects should be highlighted.
4. Click **Nest**.
5. In the Construct Model Effects list, select B[A].
6. Select **Attributes > Random Effect**.
7. In the Select Columns list, select a third nominal or ordinal effect, C.
8. Click **Add**.
9. In the Construct Model Effects list, select C. In the Select Columns list, click A. Both effects should be highlighted.
10. Click **Cross**.

Example of a Simple Repeated Measures Model

You have data on the distance traveled by each of six animals in each of the four seasons. There are two species and the animal identifier is nested within species. Since these six animals are representatives of larger species populations, you decide to treat subject as a random effect. You want to model the response, miles, as a function of species and season, accounting for the fact that there are repeated measures for each animal.

1. Select **Help > Sample Data Library** and open Animals.jmp.
2. Select **Analyze > Fit Model**.
3. In the Select Columns list, select miles and click **Y**.
4. In the Select Columns list, select species and subject.
5. Click **Add**.
6. To nest subject within species: In the Construct Model Effects list, select subject. In the Select Columns list, select species. The two effects should be highlighted.
7. Click **Nest**.
8. In the Construct Model Effects list, select subject[species].
9. Select **Attributes > Random Effect**.
10. In the Select Columns list, select season.
11. Click **Add**.
12. In the Construct Model Effects list, select season. In the Select Columns list, click species. Both effects should be highlighted.
13. Click **Cross**.

Figure 2.19 Fit Model Window for Simple Repeated Measures Model



Two-Factor Response Surface Model

Effects to be entered: X&RS, Z&RS, X*X, X*Z, Z*Z

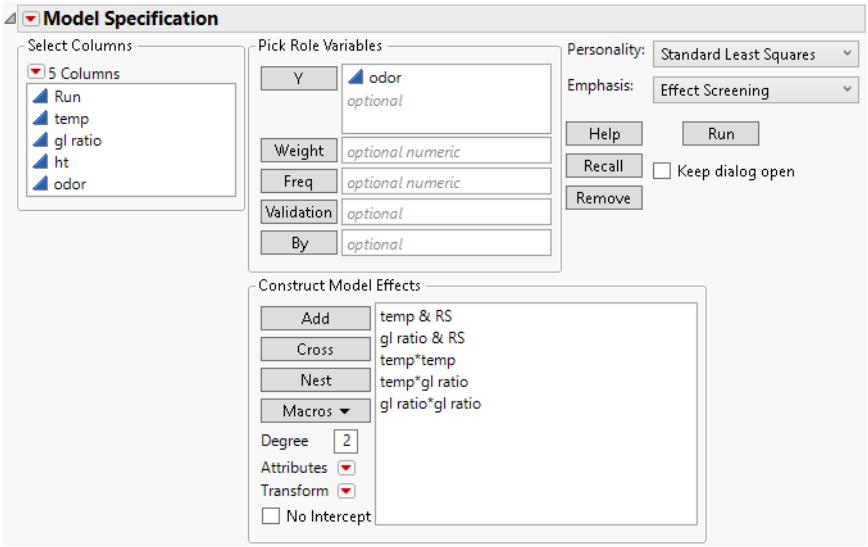
1. In the Select Columns list, select two continuous effects, X and Z.
2. Select **Macros > Response Surface**.

Example of Two-Factor Response Surface Model

In the Odor Control Original.jmp sample data table, you want to fit a response surface to model the response, odor, as a function of temp and gl ratio. (Although you could include ht, as shown in the data table script Response Surface, for this illustration do not.)

1. Select **Help > Sample Data Library** and open Odor Control Original.jmp.
2. Select **Analyze > Fit Model**.
3. In the Select Columns list, select odor and click **Y**.
4. In the Select Columns list, select temp and gl ratio.
5. Select **Macros > Response Surface**.

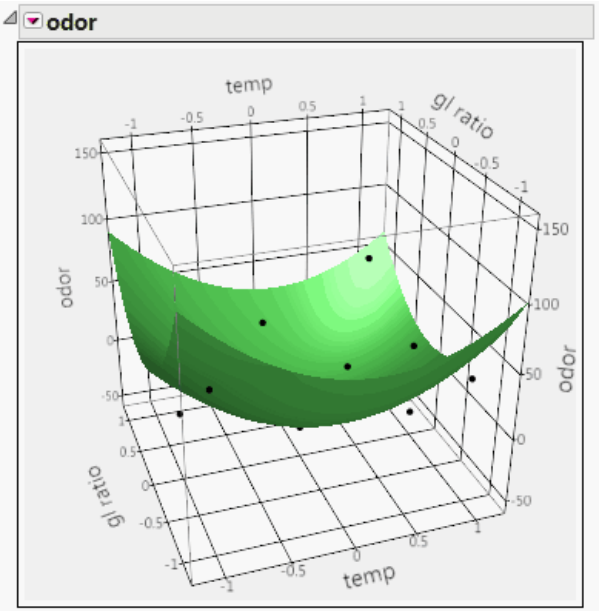
Figure 2.20 Fit Model Window for Two-Factor Response Surface Model



6. Click **Run**.

Figure 2.21 shows a Surface Profiler plot of the data and a quadratic response surface fit to the data for the Odor Control Original.jmp sample data table. To obtain this plot, click the Response odor red triangle and select **Factor Profiling > Surface Profiler**. To show the points, click the disclosure icon to open the **Appearance** panel and click **Actual**. If you want to make the points appear larger, right-click in the plot, select **Settings** and adjust the **Marker Size**.

Figure 2.21 Model Fit for a Degree-Two Polynomial in Two Variables



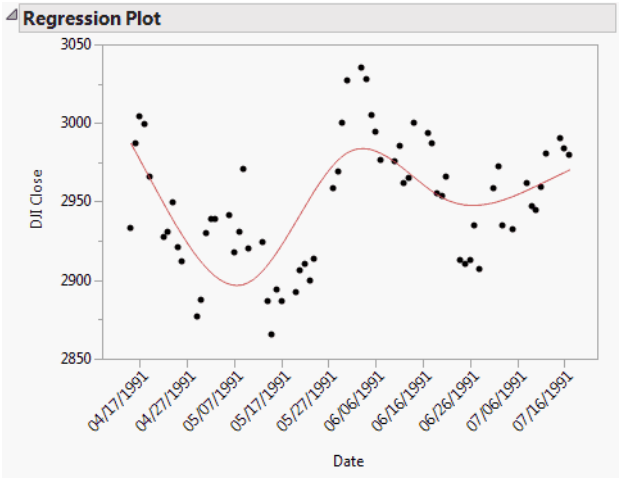
Knotted Spline Effect

Effects to be entered: X&Knotted

1. In the Select Columns list, select a continuous effect, X.
2. Click **Add**.
3. Select X in the Construct Model Effects list.
4. Select **Attributes > Knotted Spline Effect**.
5. In the menu that appears, specify the number of knots or accept the default number.
6. Click **OK**.

Figure 2.22 shows the Regression Plot for a model fit to the data in the XYZ Stock Averages (plots).jmp sample data table. Here Date is assigned the Knotted Spline Effect and five knots are specified. DJI Close is the response.

Figure 2.22 Model Fit for a Knotted Spline with Five Knots



Chapter 3

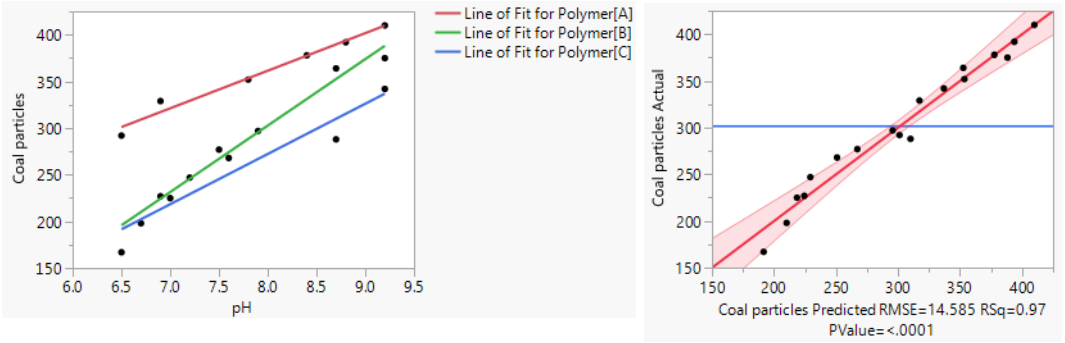
Standard Least Squares Models

Analyze Common Classes of Models

The Standard Least Squares personality within the Fit Model platform fits a wide spectrum of standard models. These models include regression, analysis of variance, analysis of covariance, and mixed models, as well as the models typically used to analyze designed experiments. Use the Standard Least Squares personality to construct linear models for continuous-response data using least squares or, in the case of random effects, restricted maximum likelihood (REML).

Analytic results are supported by compelling dynamic visualization tools such as profilers, contour plots, and surface plots (see *Profilers*). These visual displays stimulate, complement, and support your understanding of the model. They enable you to optimize several responses simultaneously and to explore the effect of noise.

Figure 3.1 Examples of Standard Least Squares Plots



Contents

Example Using Standard Least Squares . . . 76
Launch the Standard Least Squares Personality . . . 79
Fit Model Launch Window . . . 80
Standard Least Squares Options in the Fit Model Launch Window . . . 81
Validation in Standard Least Squares . . . 83
Missing Values . . . 83
Fit Least Squares Report . . . 84
Single versus Multiple Responses . . . 84
Report Structure Related to Emphasis . . . 85
Special Reports . . . 85
Least Squares Fit Options . . . 89
Fit Group Options . . . 89
Response Options . . . 90
Regression Reports . . . 91
Summary of Fit . . . 92
Analysis of Variance . . . 93
Parameter Estimates . . . 94
Effect Tests . . . 96
Effect Details . . . 97
Lack of Fit . . . 111
Estimates . . . 113
Show Prediction Expression . . . 115
Sorted Estimates . . . 116
Expanded Estimates . . . 119
Indicator Parameterization Estimates . . . 121
Sequential Tests . . . 122
Custom Test . . . 123
Multiple Comparisons . . . 126
Compare Slopes . . . 140
Joint Factor Tests . . . 140
Inverse Prediction . . . 141
Cox Mixtures . . . 145
Parameter Power . . . 147
Correlation of Estimates . . . 149
Coding for Nominal Effects . . . 150
Effect Screening . . . 151
Scaled Estimates and the Coding of Continuous Terms . . . 152
Plot Options . . . 153
Normal Plot Report . . . 158

Bayes Plot Report . . . 159
Pareto Plot Report . . . 161
Factor Profiling . . . 162
Profiler . . . 163
Interaction Plots . . . 164
Contour Profiler . . . 165
Mixture Profiler . . . 166
Cube Plots . . . 167
Box Cox Y Transformation . . . 168
Surface Profiler . . . 170
Row Diagnostics . . . 171
Leverage Plots . . . 173
Press . . . 177
Save Columns . . . 178
Prediction Formula . . . 181
Effect Summary Report . . . 181
Mixed and Random Effect Model Reports and Options . . . 186
Mixed Models and Random Effect Models . . . 186
Restricted Maximum Likelihood (REML) Method . . . 190
EMS (Traditional) Model Fit Reports . . . 195
Models with Linear Dependencies among Model Terms . . . 198
Singularity Details . . . 198
Parameter Estimates Report . . . 199
Effect Tests Report . . . 200
Examples . . . 200
Statistical Details for the Standard Least Squares Personality . . . 202
Emphasis Rules . . . 203
Details of Custom Test Example . . . 203
Correlation of Estimates . . . 204
Leverage Plot Details . . . 204
The Kackar-Harville Correction . . . 207
Power Analysis . . . 208

Example Using Standard Least Squares

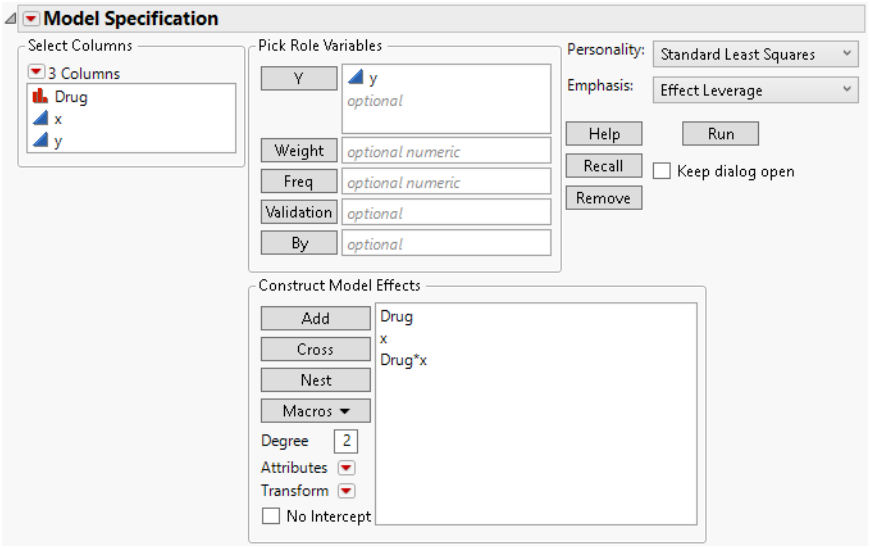
In a study of the effect of drugs in treating a disease, thirty patients are randomly divided into three groups of ten. Two of these groups are administered drugs (Drug a and Drug d), whereas the third group is administered a placebo (Drug f). A pretreatment measure, x, is taken on each patient, as well as a posttreatment measure, y. The pretreatment score, x, is included as a covariate, to account for differences in the stage of the disease among patients. This example is taken from Snedecor and Cochran (1967, p. 422).

You are interested in determining if there is a difference in the three Drug groups. You construct a model with response y and model effects Drug, x, and the interaction of Drug and x. The interaction might account for a situation where drugs have differential effects, based on the stage of the disease. For background on the Fit Model window and the various personalities, see "Model Specification".

1. Select **Help > Sample Data Library** and open Drug.jmp.
2. Select **Analyze > Fit Model**.
3. Select y and click **Y**.

   When you add this column as **Y**, the fitting **Personality** becomes Standard Least Squares. An **Emphasis** option is added with a selection of Effect Leverage, which you can change if desired.
4. Select Drug and x. With these two effects highlighted in the Select Columns list, click **Macros** and select **Full Factorial**. The macro adds the two effects and their two-way interaction to the Construct Model Effects list.

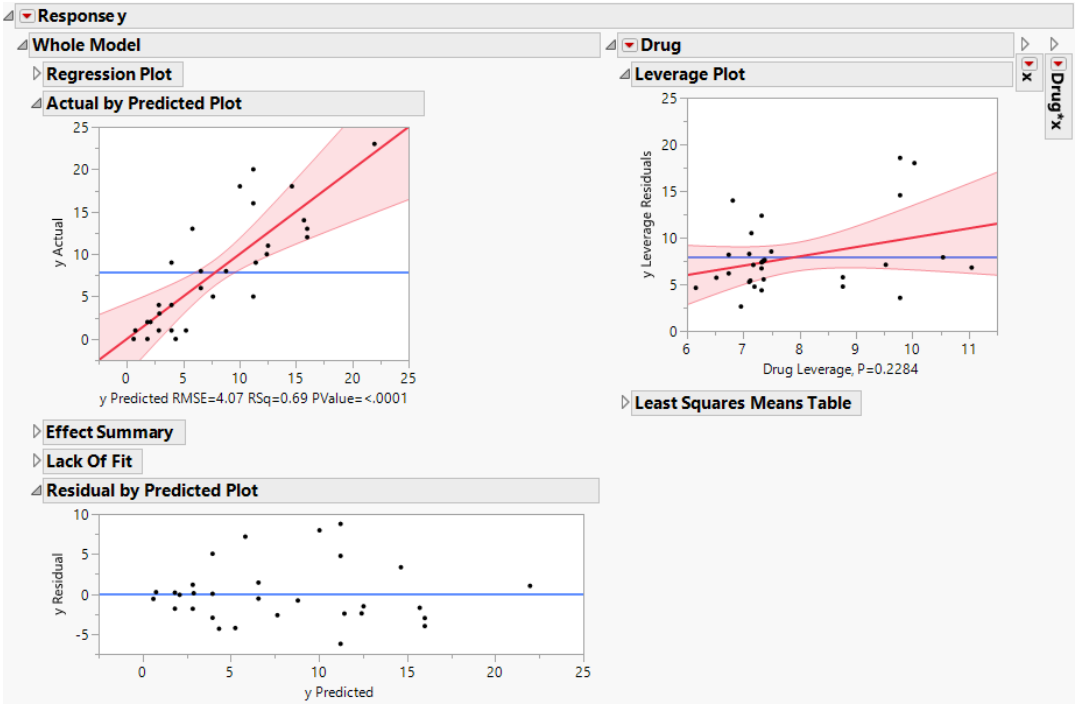
Figure 3.2 Completed Fit Model Launch Window



5. Click **Run**.

The Fit Least Squares report is shown in Figure 3.3. Note that some of the constituent reports are closed because of space considerations. The Actual by Predicted, Residual by Predicted, and Leverage plots show no discrepancies in terms of model fit and underlying assumptions.

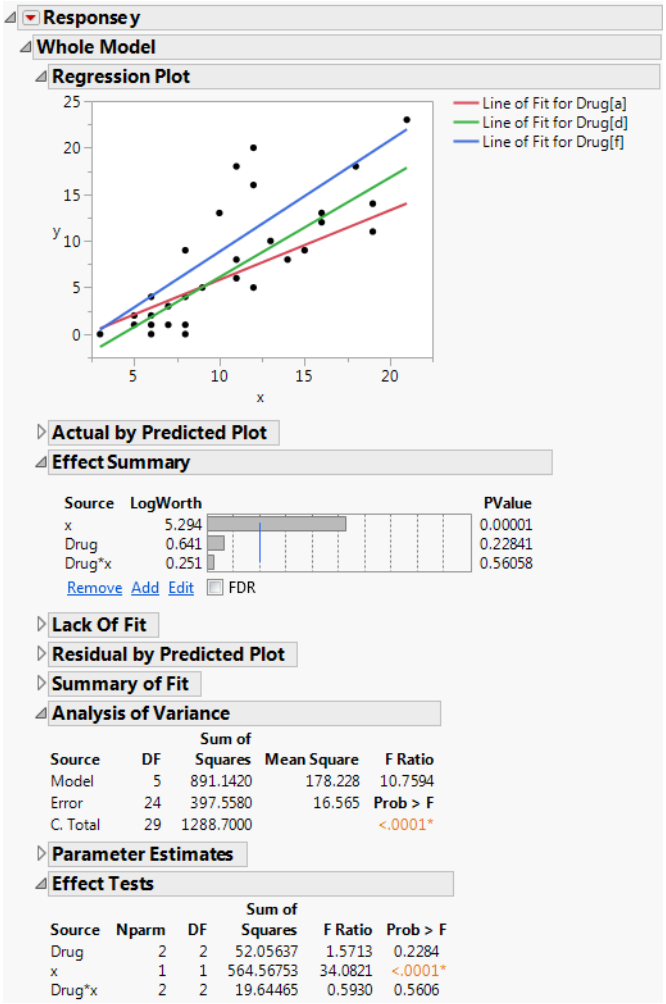
Figure 3.3 Fit Least Squares Report Showing Plots to Assess Model Fit



Since there are no apparent problems with the model fit, you can now interpret the statistical tests. Figure 3.4 shows the relevant reports. The overall model is significant, as shown in the Analysis of Variance report.

Although the Regression Plot suggests that Drug and the pretreatment measure, x, interact, the Prob > F value in the Effect Tests report does not support that conclusion. The Effect Tests report also shows that x is significant in explaining y, but Drug is not significant. The study does not detect a difference among the three groups. However, you cannot conclude that Drug has no effect. The drugs might have different effects, but the study size was not large enough to detect that difference.

Figure 3.4 Fit Least Squares Report Showing Reports to Assess Significance



Launch the Standard Least Squares Personality

Standard least squares is one of several analytic techniques that you can select in the Fit Model launch window.

This section describes how you select standard least squares as your fitting methodology in the Fit Model launch window. Options that are specific to this selection are also covered. For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

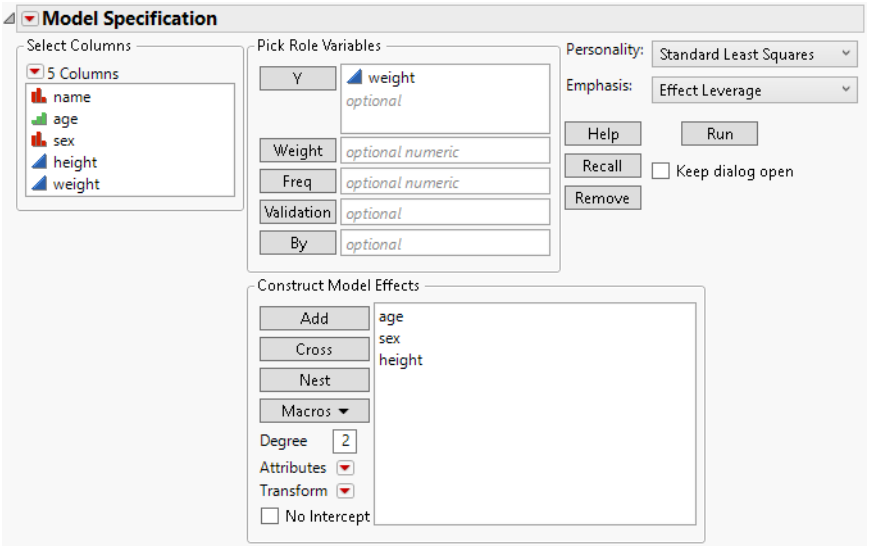
Fit Model Launch Window

You can specify models with both fixed and random effects in the Fit Model launch window. The options differ based on the nature of the model that you specify.

Fixed Effects Only

To fit models using the standard least squares personality, select **Analyze > Fit Model** and then select **Standard Least Squares** from the Personality list. When you enter one or more continuous variables in the Y list, the Personality defaults to Standard Least Squares. Note, however, that other selections are available for continuous Y variables. When you specify only fixed effects for a Standard Least Squares fit, the Fit Model launch window appears as shown in Figure 3.5. This example illustrates the launch window using the Big Class.jmp sample data table.

Figure 3.5 Fit Model Launch Window for a Fixed Effects Model



When the Standard Least Squares personality is selected in the Personality list, an Emphasis option also appears. Emphasis options control the reports that are provided in the initial report window. Based on the model effects that are included, JMP infers which reports you are likely to want. However, any report not shown as part of the initial report can be shown by selecting the appropriate option from the default report's red triangle menu.

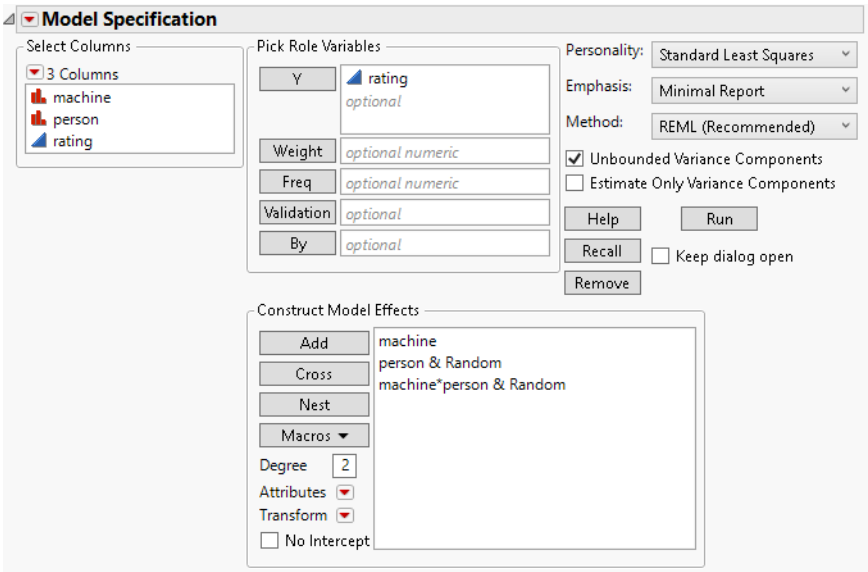
For more information about reports that are available for each Emphasis option, see "Emphasis Options for Standard Least Squares".

Random Effects

If the specified model contains one or more random effects, then additional options become available in the Fit Model launch window. Consider the Machine.jmp sample data table. Each of six randomly chosen workers performs work at each of three machines and their output is rated. You are interested in estimating the variation in ratings across the workforce, rather than in determining whether these six specific workers' ratings differ. You need to treat person and machine*person as random effects when you specify the model.

The Fit Model launch window for this model is shown in Figure 3.6. When the Random Effect attribute is applied to person, a Method option and two options relating to variance components appear in the Fit Model Launch window.

Figure 3.6 Fit Model Launch Window for a Model Containing a Random Effect



Standard Least Squares Options in the Fit Model Launch Window

The following Fit Model launch window options are specific to the Standard Least Squares personality.

Emphasis Controls the types of reports and plots that appear in the initial report window. See "Emphasis Options for Standard Least Squares".

Method (Appears only when random effects are specified.) Estimates the model using one of these methods:

REML See "REML Variance Component Estimates".

EMS Expected Mean Squares, also called the Method of Moments. See "EMS (Traditional) Model Fit Reports".

Unbounded Variance Components (Appears only when REML is selected as the Method.) Allows variance component estimates to be negative. This option is selected by default. This option should remain selected if you are interested in fixed effects, since bounding the variance estimates at zero leads to bias in the tests for fixed effects. See "Negative Variances".

Estimate Only Variance Components (Appears only when REML is selected as the Method.) Provides a report that shows only variance component estimates. See "Estimate Only Variance Components".

Fit Separately (Appears only for models with multiple Y variables and no random effects.) Fits a separate model for each Y variable using all rows that are nonmissing. See "Missing Values".

Emphasis Options for Standard Least Squares

The three options in the Emphasis list control the types of plots and reports that you see as part of the initial report for the Standard Least Squares personality. See the descriptions below. JMP chooses a default emphasis based on the number of rows in the data table, the number of effects entered in the Construct Model Effects list, and the attributes applied to effects. You can change this choice of emphasis based on your needs. For more information about how JMP chooses the emphasis, see "Emphasis Rules".

After the initial report opens, you can add other reports and plots from the red triangle menu in the platform report window.

The following emphasis options are available:

Effect Leverage Shows leverage and residual plots, as well as reports with details about the model fit. This option is useful when your main focus is model fitting.

Effect Leverage is the most comprehensive option. This emphasis divides reports into those that relate to the Whole Model and those that relate to individual model effects. The Whole Model reports are in the left corner of the report window under the Whole Model title, with effect reports to the right.

Effect Screening Shows a sorted or scaled parameter estimates report along with a graph (when appropriate), the Prediction Profiler, and reports with details about the model fit. This option is useful when you have many effects and your initial focus is to discover which effects are active, as in screening designs.

When Effect Screening is selected, a Box-Cox transformation is calculated. If the confidence interval for the estimated λ does not contain 1, the Box-Cox Transformations report appears. See "Box Cox Y Transformation".

Minimal Report Shows only the regression plot and reports with details about the model fit. This Emphasis is the default when the Random Effect attribute is applied to any model effect.

This option is the least detailed and most concise. You can request reports of specific interest to you from the red triangle menus.

To change which reports or plots appear for all of the Emphasis options, use platform preferences. Go to **File > Preferences > Platforms > Fit Least Squares**, and use the **Set** check boxes:

- To prevent an option from appearing in the report, next to an option, select **Set** but do not select the option.
- To ensure that an option appears in the report, select **Set** and select the option.

JMP PRO Validation in Standard Least Squares

In JMP Pro, you can specify a Validation column in the Fit Model window. A validation column must have a numeric data type and should contain at least two distinct values.

- If the column contains two values, the smaller value defines the training set and the larger value defines the validation set.
- If the column contains three values, the values define the training, validation, and test sets in order of increasing size.
- If the column contains four or more distinct values, the Validation column is ignored.

The Standard Least Squares personality in JMP Pro supports the use of a Validation column. If you enter a Validation column, a Crossvalidation report is provided. See "Crossvalidation Report".

If you enter a Validation column, observations from the Validation and Test sets are marked as 'v' and 't', respectively, in plots in the report.

For more information about how a Validation column is used in JMP modeling platforms, see *Predictive and Specialized Modeling*.

Missing Values

By default, rows that have missing values for Y or any model effects are excluded from the analysis.

Note: JMP provides an Informative Missing option in the Fit Model window in the Model Specification red triangle menu. Informative Missing enables you to fit models using rows where model effects are missing. See "Informative Missing".

When your model contains a random effect, Y values are fit separately by default. The individual reports appear in the Fit Group report.

Suppose that your model contains only fixed effects, and the following statements are true:

- You specified more than one Y response.
- Some of these Y responses have missing values.
- You did not select the Fit Separately option in the Fit Model launch window.

Then, JMP prompts you to select one of the following options:

- Fit Separately fits each Y using all rows that are nonmissing for that particular Y.
- Fit Together fits each Y uses only those rows that are nonmissing for all of the Y variables.

When you select Fit Separately, a Fit Group report contains the individual reports for the Y variables. You can select profilers from the Fit Group red triangle menu to view all the Y variables in the same profiler. Alternatively, you can select a profiler from an individual Y variable report to view only that variable in the profiler.

When you select Fit Together, a Fit Least Squares report contains individual reports for each of the Y variables. However, some parts of the report are combined for all Y variables: the Effect Summary and the Profilers.

Fit Least Squares Report

When you fit a model using the Standard Least Squares personality, you obtain a Fit Least Squares report. The content of the report is driven by the nature of the data and your selections in the Fit Model launch window.

Tip: To always see reports that do not appear by default, select them using **File > Preferences > Platforms > Fit Least Squares**.

Single versus Multiple Responses

When you fit a single response variable Y, the Fit Least Squares window organizes detailed reports in a report entitled “Response Y”. When you fit several responses, reports for individual responses are usually organized in a report entitled “Least Squares Fit”. However, if there is missing response data, and you select the option to Fit Separately, reports for individual responses are organized in a report titled “Fit Group”.

Report Structure Related to Emphasis

When you select the Effect Leverage Emphasis in the Fit Model launch window, the report for a given response is arranged in columns. The left column consists of the Whole Model report, which contains additional reports that pertain to the model. Reports for each effect in the model are shown in the columns to the right of the Whole Model report.

When you select either the Effect Screening or Minimal Report Emphasis in the Fit Model launch window, all reports for each response are arranged in the left column.

Special Reports

This section describes the reports that are available based on the data structure or choices that you made regarding effect attributes.

Singularity Details

When there are linear dependencies among model effects, the Singularity Details report appears as the first report under the Response report title. It contains a table of the linear functions that the model terms satisfy. These functions define the aliasing relationships among model terms. Figure 3.7 shows an example for the Singularity.jmp sample data table.

Figure 3.7 Singularity Details Report

Singularity Details

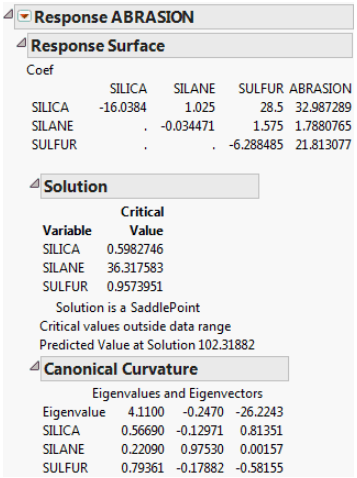
Term	Details
X1	= - X2 + X3=2*Intercept + 3*X2 - 4*A[a] + 2*A[b]= - X2 + 2*A[a] - 2*A[c]

When there are linear dependencies among effects, estimates of some model terms are not unique. See "Models with Linear Dependencies among Model Terms".

Response Surface Report

When an effect in a model has the response surface (&RS) or mixture response surface (&RS&Mixture) attribute, a Response Surface report is provided. See Figure 3.8 for an example of a Response Surface report for the Tiretread.jmp sample data table.

Figure 3.8 Response Surface Report



Coef Table

The Coef table shown as the first part of the Response Surface report gives a concise summary of the estimated model parameters. The first columns give the coefficients of the second-order terms. The last column gives the coefficients of the linear terms. To see the prediction expression in its entirety, select **Estimates** > **Show Prediction Expression** from the report's red triangle menu.

Solution Report

The Solution report gives a critical value (maximum, minimum, or saddle point), if one exists, along with the predicted value at that point. It also alerts you if the solution falls outside the range of the data.

Canonical Curvature Report

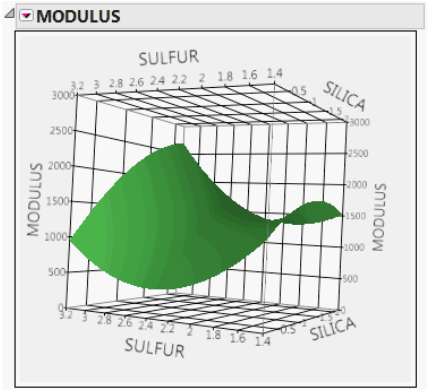
The eigenvalues and eigenvectors of the matrix of second-order parameter estimates determine the type of curvature. The eigenvectors show the principal directions of the surface, including the directions of greatest and smallest curvature.

The eigenvalues are provided in the first row of the Canonical Curvature table.

- If the eigenvalues are negative, the response surface curves downward from a maximum.

- If the eigenvalues are positive, the surface shape curves upward from a minimum.
- If there are both positive and negative eigenvalues, the surface is saddle shaped, curving up in one direction and down in another direction. See Figure 3.9 for an example using the Tiretread.jmp sample data table.

Figure 3.9 Surface Profiler Plot with Saddle-Shaped Surface



The eigenvectors listed below the eigenvalues show the orientation of the principal axes. The larger the absolute value of an eigenvalue, the greater the curvature of the response surface in its associated direction. Sometimes a zero eigenvalue occurs. This eigenvalue means that, along the direction described by the corresponding eigenvector, the fitted surface is flat.

Note: The response surface report is not shown for response surface models consisting of more than 20 factors. No error message or alert is given. For more information about response surface designs, see the *Design of Experiments Guide*.

Mixed and Random Effect Model Reports

When you specify a random effect in the Fit Model launch window, the Method list appears. This list provides two fitting methods: REML (Recommended) and EMS (Traditional). Additional reports as well as Save Columns and Profiler options are shown, based on the model and the method that you select.

For more information about the REML method reports, see "Restricted Maximum Likelihood (REML) Method". For more information about the EMS method reports, see "EMS (Traditional) Model Fit Reports".

Crossvalidation Report

JMP PRO When you enter a Validation column in the Fit Model launch window, a Crossvalidation report is provided. The report gives the following for each of the sets used in validation:

Source Identifies the set as the Training, Validation, or Test set.

RSquare The RSquare value calculated for observations in the given set relative to the model derived using the Training Set. For the Training Set, this is the usual RSquare value.

For each of the Training, Validation, and Test sets, the RSquare value is computed as follows:

- For each observation in the given set, compute the prediction error. This is the difference between the actual response and the response predicted by the Training set model.
- Square and sum the prediction errors to obtain SSE_{Source}, where the subscript *Source* denotes any of the Training, Validation, or Test sets.
- Square and sum the differences between the actual responses for observations in the *Source* set and their mean. Denote this value by SST_{Source}.
- RSquare for the *Source* set is:

$$\text{RSquare}_{Source} = 1 - \frac{SSE_{Source}}{SST_{Source}}$$

Note: It is possible for RSquare values for the Validation and Test sets to be negative.

RASE The square root of the mean squared prediction error. For each of the Training, Validation, and Test sets, RASE is computed as follows:

- For each observation in the given set, calculate the prediction error. This is the difference between the actual response and the response predicted by the Training set model.
- Square and sum the prediction errors to obtain SSE_{Source}, where the subscript *Source* denotes any of the Training, Validation, or Test sets.
- Denote the number of observations by n.
- RASE is:

$$\text{RASE}_{Source} = \sqrt{\frac{SSE_{Source}}{n}}$$

Freq The number of observations in the Source set.

Least Squares Fit Options

You might have more than one Y and no missing response values, or more than one Y with missing values. When you select Fit Together, the responses are grouped in a report called Least Squares Fit. The red triangle menu includes the following options:

Profilers Shows all responses in a single profiler. You can view the effects of model terms on all responses simultaneously and perform multiple optimization. See "Factor Profiling".

Model Dialog Shows the completed launch window for the current analysis.

Effect Summary Shows the interactive Effect Summary report that enables you to add or remove effects from the model. See "Effect Summary Report".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Fit Group Options

When you have more than one Y and you select Fit Separately, the responses are grouped in a report called Fit Group. The red triangle menu includes the following options:

Profiler Shows all responses in a single profiler. You can view the effects of model terms on all responses simultaneously and perform multiple optimization. See "Profiler".

Contour Profiler Shows all responses in a single contour profiler. You can explore the effects of model terms on all responses simultaneously.

Surface Profiler Shows separate surface profiler reports for each response.

Arrange in Rows Rearranges the reports for the platform analyses in a specified number of rows. This would be used mostly to arrange reports so that more reports fit in a window or on the page of output.

Order by Goodness of Fit Sorts the reports by significance of fit (RSquare). This option is available only for platforms that surface the RSquare to the platform level. For example, if you generate hundreds of Oneway analyses from one launch window, they appear in a `FitGroup` and you can sort them so that the strongest relationships appear first.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Response Options

Red triangle menu options for the response give you the ability to customize reports according to your needs.

Regression Reports Provides basic reports and report options. See "Regression Reports".

Estimates Provides options for further analyses relating to parameter estimates. See "Estimates".

Effect Screening Provides reports and plots for identifying significant effects. See "Effect Screening".

Factor Profiling Provides profilers, interaction, and cube plots to examine how the response is related to the model terms. Also provides a plot and report for fitting a Box-Cox transformation. See "Factor Profiling".

Row Diagnostics Provides plots and reports for examining residuals. Also reports the PRESS statistic and provides a Durbin-Watson test. See "Row Diagnostics".

Save Columns Saves model results as columns in the data table, except for Save Coding Table, which saves results in a separate data table. See "Save Columns".

Model Dialog Shows the completed Fit Model launch window for the current analysis.

Effect Summary Shows the interactive Effect Summary report that enables you to add or remove effects from the model. See "Effect Summary Report".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Regression Reports

The Regression Reports menu provides summary information about model fit, effect significance, and model parameters.

Summary of Fit Shows or hides a summary of model fit. See "Summary of Fit".

Analysis of Variance Shows or hides calculations for comparing the fitted model to a simple mean model. See "Analysis of Variance".

Parameter Estimates Shows or hides a report containing the parameter estimates and *t* tests for the hypothesis that each parameter is zero. See "Parameter Estimates".

Effect Tests Shows or hides tests for the fixed effects in the model. See "Effect Tests".

Effect Details Shows or hides a report containing details, plots, and tests for individual effects. See "Effect Details".

When the Effect Leverage Emphasis option is selected, each effect has its own report at the top of the Fit Least Squares report window. This report includes effect details options as well as a leverage plot. See "Leverage Plots".

Lack of Fit Shows or hides a test assessing if the model has the appropriate effects, when that test can be conducted. See "Lack of Fit".

Show All Confidence Intervals Shows or hides confidence intervals for the following statistics:

- Parameter estimates in the Parameter Estimates report
- Least squares means in the Least Squares Means Table

AICc Shows or hides AICc and BIC values in the Summary of Fit report. See "Likelihood, AICc, and BIC".

Summary of Fit

The Summary of Fit report provides details such as RSquare calculations and the AICc and BIC values.

RSquare Estimates the proportion of variation in the response that can be attributed to the model rather than to random error. Using quantities from the corresponding Analysis of Variance table, RSquare (also called the *coefficient of multiple determination)* is calculated as follows:

$$\frac{\text{Sum of Squares(Model)}}{\text{Sum of Squares(C. Total)}}$$

An RSquare closer to 1 indicates a better fit to the data than does an RSquare closer to 0. An RSquare near 0 indicates that the model is not a much better predictor of the response than is the response mean.

Note: A low RSquare value suggests that there might be variables not in the model that account for the unexplained variation. However, if your data are subject to a large amount of inherent variation, even a useful regression model can have a low RSquare value. Read the literature in your research area to learn about typical RSquare values.

Rsquare Adj Adjusts the RSquare statistic for the number of parameters in the model. Rsquare Adj facilitates comparisons among models with different numbers of parameters. The computation uses the degrees of freedom. Using quantities from the corresponding Analysis of Variance table, RSquare Adj is calculated as follows:

$$1 - \frac{\text{Mean Square(Error)}}{\text{Sum of Squares (C. Total)}/\text{DF(C. Total)}}$$

Root Mean Square Error Estimates the standard deviation of the random error. This quantity is the square root of the Mean Square for Error in the Analysis of Variance report.

Note: Root Mean Square Error is commonly known as *RMSE*.

Mean of Response Shows the overall mean of the response values.

Observations (or Sum Wgts) Shows the number of observations used in the model.

– This value is the same as the number of rows in the data table under the following conditions: there are no missing values, no excluded rows, and no column assigned to the role of Weight or Freq.

- This value is the sum of the positive values in the Weight column if there is a column assigned to the role of Weight.
- This value is the sum of the positive values in the Freq column if there is a column assigned to the role of Freq.

AICc (Appears only if you have selected the AICc option from the Regression Reports menu or if you have set AICc as a Fit Least Squares preference.) Shows the corrected Akaike Information Criterion value (AICc). See "Likelihood, AICc, and BIC".

BIC (Appears only if you have selected the AICc option from the Regression Reports menu or if you have set AICc as a Fit Least Squares preference.) Shows the Bayesian Information Criterion value (BIC). See "Likelihood, AICc, and BIC".

Analysis of Variance

The Analysis of Variance report provides the calculations for comparing the fitted model to a model where all predicted values equal the response mean.

Note: If either a Frequency or a Weight variable is entered in the Fit Model launch window, the entries in the Analysis of Variance report are adjusted in keeping with the descriptions in "Frequency" and "Weight".

The Analysis of Variance report contains the following columns:

Source The three sources of variation: Model, Error, and C. Total (Corrected Total).

DF The associated *degrees of freedom* (DF) for each source of variation. The C. Total DF is always one less than the number of observations, and it is partitioned into degrees of freedom for the Model and Error as follows:

- The Model DF is the number of parameters (other than the intercept) used to fit the model.
- The Error DF is the difference between the C. Total DF and the Model DF.

Sum of Squares The associated Sum of Squares (SS) for each source of variation.

- The total (C. Total) SS is the sum of the squared differences between the response values and the sample mean. It represents the total variation in the response values.
- The Error SS is the sum of the squared differences between the fitted values and the actual values. It represents the variability that remains unexplained by the fitted model.
- The Model SS is the difference between C. Total SS and Error SS. It represents the variability explained by the model.

Mean Square The mean square statistics for the Model and Error sources of variation. Each Mean Square is the sum of squares divided by its corresponding DF.

Note: The square root of the Mean Square for Error is the same as RMSE in the Summary of Fit report.

MSE Used (Appears only when the Error Specification is Pure Error or Specified.) The mean square error used when the default error specification is not selected. This value is used to calculate the F Ratio instead of the mean square error in the Mean Square column.

DFE Used (Appears only when the Error Specification is Pure Error or Specified.) The error degrees of freedom used when the default error specification is not selected. This value is used to calculate the Prob > F value instead of the Error DF in the DF column.

F Ratio The model mean square divided by the error mean square. The F Ratio is the test statistic for a test of whether the model differs significantly from a model where all predicted values are the response mean.

Prob > F The *p*-value for the test. The Prob > F value measures the probability of obtaining an F Ratio as large as what is observed, given that all parameters except the intercept are zero. Small values of Prob > F indicate that the observed *F* Ratio is unlikely. Such values are considered evidence that there is at least one significant effect in the model.

Parameter Estimates

The Parameter Estimates report shows the estimates of the model parameters and, for each parameter, gives a *t* test for the hypothesis that it equals zero.

Note: Estimates are obtained and tested, if possible, even when there are linear dependencies among the model terms. Such estimates are labeled Biased or Zeroed. See "Models with Linear Dependencies among Model Terms".

Term The model term corresponding to the estimated parameter. The first term is always the intercept, unless the No Intercept option was checked in the Fit Model launch window. Continuous effects appear with the name of the data table column. Note that continuous columns that are part of higher order terms might be centered. Nominal or ordinal effects appear with values of levels in brackets. See "Coding for Nominal Effects" and "The Factor Models" for information about the coding of nominal and ordinal terms.

Estimate The parameter estimates for each term. These are the estimates of the model coefficients. When there are linear dependencies among model terms, these might be labeled as Biased or Zeroed. See "Models with Linear Dependencies among Model Terms".

Std Error The estimates of the standard errors for each of the estimated parameters.

***t* Ratio** The tests of whether the true value of each parameter is zero. The *t* Ratio is the ratio of the estimate to its standard error. Given the usual assumptions about the model, the *t* Ratio has a Student's *t* distribution under the null hypothesis.

Prob>|t| The *p*-value for the test that the true parameter value is zero, against the two-sided alternative that it is not.

Lower 95% The lower 95% confidence limit for the parameter estimate. This column appears only if you have the Regression Reports > Show All Confidence Intervals option selected or if you right-click in the report and select Columns > Lower 95%.

Upper 95% The upper 95% confidence limit for the parameter estimate. This column appears only if you have the Regression Reports > Show All Confidence Intervals option selected or if you right-click in the report and select Columns > Upper 95%.

Std Beta The parameter estimates for a regression model where all of the terms have been standardized to a mean of 0 and a variance of 1. This column appears only if you right-click in the report and select Columns > Std Beta.

VIF The variance inflation factor for each term in the model. High VIFs indicate a collinearity issue among the terms in the model.

The VIF for the i^{th} term, x_i, is defined as follows:

$$VIF_i = \frac{1}{1 - R_i^2}$$

where R_i^2 is the RSquare, or *coefficient of multiple determination*, for the regression of x_i as a function of the other explanatory variables. This column appears only if you right-click in the report and select Columns > VIF.

Design Std Error The square roots of the *relative variances* of the parameter estimates (Goos and Jones 2011, p. 25):

$$\sqrt{diag(X'X)^{-1}}$$

These are the standard errors divided by RMSE. This column appears only if you right-click in the report and select Columns > Design Std Error.

Effect Tests

The Effect Tests report appears only when there are fixed effects in the model. The effect test for a given effect tests the null hypothesis that all parameters associated with that effect are zero. An effect might have only one parameter as for a single continuous explanatory variable. In this case, the test is equivalent to the *t* test for that term in the Parameter Estimates report. A nominal or ordinal effect can have several associated parameters, based on its number of levels. The effect test for such an effect tests whether all of the associated parameters are zero.

Note the following:

- Effect tests are conducted, when possible, for effects whose terms are involved in linear dependencies. See "Models with Linear Dependencies among Model Terms".
- Parameterization and handling of singularities differ from the SAS GLM procedure. For more information about parameterization and handling of singularities, see "The Factor Models".

The Effects Test report contains the following columns:

Source The effects in the model.

Nparm The number of parameters associated with the effect. A continuous effect has one parameter. The number of parameters for a nominal or ordinal effect is one less than its number of levels. The number of parameters for a crossed effect is the product of the number of parameters for each individual effect.

DF The degrees of freedom for the effect test. Ordinarily, Nparm and DF are the same. They can differ if there are linear dependencies among the predictors. In such cases, DF might be less than Nparm, indicating that at least one parameter associated with the effect is not testable. Whenever DF is less than Nparm, the note LostDFs appears to the right of the line in the report. If there are degrees of freedom for error, the test is conducted. See "Effect Tests Report".

Sum of Squares The sum of squares for the hypothesis that the effect is zero.

F Ratio The F statistic for testing that the effect is zero. The F Ratio is the ratio of the mean square for the effect divided by the mean square for error. The mean square for the effect is the sum of squares for the effect divided by its degrees of freedom.

Prob > F The *p*-value for the effect test.

Mean Square The mean square for the effect, which is the sum of squares for the effect divided by its DF.

Note: Appears only if you right-click in the report and select **Columns > Mean Square**.

Effect Details

The Effect Details report provides details, plots, and tests for individual effects. It consists of separate reports based on the emphasis that you select in the Fit Model launch window.

- Effect Leverage emphasis: Each effect has its own report at the top of the Fit Least Squares report window to the right of the Whole Model report. In this case, the report includes a Leverage Plot for the effect.
- Effect Screening or Minimal Report emphases: The Effect Details report is provided but is initially closed. Click the disclosure icon to show the report.

The initial content of the report is the Table of Least Squares Means. Depending on the nature of the effect, this table might not be appropriate, and the default report might initially show no content. However, certain red triangle options are available.

Table of Effect Options

The red triangle menu next to an effect name provides the following options. For certain modeling types, some of these options might not be appropriate and are therefore not available.

LSMeans Table Shows the statistics that are compared when effects are tested. See "LSMeans Table".

This option is not enabled for continuous effects.

LSMeans Plot Shows plots of least squares means for nominal and ordinal effects. If the effect is an interaction, this option displays the Least Squares Means Plot Options window. See "LSMeans Plot".

LSMeans Contrast Shows the Contrast Specification window, which enables you to specify and test contrasts to compare levels for nominal and ordinal effects and their interactions. See "LSMeans Contrast".

LSMeans Student's t Shows tests and confidence intervals for pairwise comparisons of least squares means using Student's *t* tests. See "LSMeans Student's t and LSMeans Tukey HSD".

Note: The significance level applies to individual comparisons and *not* to all comparisons collectively. The error rate for the collection of comparisons is greater than the error rate for individual tests.

LSMeans Tukey HSD Shows tests and confidence intervals for pairwise comparisons of least squares means using the *Tukey-Kramer HSD* (Honestly Significant Difference) test (Tukey 1953; Kramer 1956). See "LSMeans Student's t and LSMeans Tukey HSD".

Note: The significance level applies to the collection of pairwise comparisons. The significance level is exact if the sample sizes are equal and conservative if the sample sizes differ (Hayter 1984).

LSMeans Dunnett Shows tests and confidence intervals for pairwise comparisons against a control level that you specify. Also provides a plot of test results. See "LSMeans Dunnett".

Test Slices For each level of each column in the interaction, jointly tests pairwise comparisons among all the levels of the other classification columns in the interaction. See "Test Slices".

Note: Available only for interactions involving nominal and ordinal effects.

Power Analysis Shows the Power Details report, which enables you to analyze the power for the effect test. See "Power Analysis".

LSMeans Table

Least squares means are values predicted by the model for the levels of a categorical effect where the other model factors are set to *neutral* values. The neutral value for a continuous effect is defined to be its sample mean. The neutral value for a nominal effect that is not involved in the effect of interest is the average of the coefficients for that effect. The neutral value for an uninvolved ordinal effect is defined to be the first level of the effect in the value ordering.

Least squares means are also called *adjusted means* or *population marginal means*. Least squares means can differ from simple means when there are other effects in the model. In fact, it is common for the least squares means to be closer together than the sample means. This situation occurs because of the nature of the neutral values where these predictions are made.

Because least squares means are predictions at specific values of the other model factors, you can compare them. When effects are tested, comparisons are made using the least squares means. For more information about least squares means, see "Least Squares Means across Nominal Factors" and "Ordinal Least Squares Means".

For main effects, the Least Squares Means Table also includes the sample mean (Figure 3.10).

Example of a Least Squares Means Table

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Model**.
3. Select weight and click **Y**.
4. Select age, sex, and height and click **Add**.
5. From the **Emphasis** list, select **Effect Screening**.

6. Click **Run**.
7. The Effect Details report appears near the bottom of the Fit Least Squares report and is initially closed. Click the disclosure icon next to the Effect Details report title to show the report.

The Effect Details report, shown in Figure 3.10, shows reports for each of the three effects. Least Squares Means tables are given for age and sex, but not for the continuous effect height. Notice how the least squares means differ from the sample means.

Figure 3.10 Least Squares Mean Table

Response weight

Effect Details

age

Least Squares Means Table

Level	Least Sq Mean	Std Error	Mean
12	119.38671	5.5302002	99.000
13	105.70112	5.2695143	94.714
14	93.53945	3.9461891	100.833
15	99.61686	5.1729356	108.286
16	109.22769	7.7788438	118.333
17	121.90696	8.0768318	140.667

sex

Least Squares Means Table

Level	Least Sq Mean	Std Error	Mean
F	121.43592	6.1844278	100.944
M	117.33750	5.7895918	108.318

height

The Least Squares Means report contains the following columns:

Level The categorical levels or combination of levels.

Least Sq Mean An estimate of the least squares mean for each level.

Estimability (Appears only when a least squares mean is not estimable.) A warning if a least squares mean is not estimable.

Std Error Shows the standard error of the least squares mean for each level.

Lower 95% Shows the lower 95% confidence limit for the least squares mean. This column appears only if you have the Regression Reports > Show All Confidence Intervals option selected or if you right-click in the report and select Columns > Lower 95%.

Upper 95% Shows the upper 95% confidence limit for the least squares mean. This column appears only if you have the Regression Reports > Show All Confidence Intervals option selected or if you right-click in the report and select Columns > Upper 95%.

Mean Shows the response sample mean for the given level. This mean differs from the least squares mean if the values for other effects in the model do not balance out across this effect.

LSMeans Plot

The LSMeans Plot option produces a Least Squares Means Plot for nominal and ordinal main effects and their interactions. If the effect is an interaction, this option displays the Least Squares Means Plot Options window. See "Least Squares Means Plot Options".

The Least Squares Means Plot red triangle menu contains the following options:

Show Confidence Limits Shows or hides confidence limits for each estimate in the plot.

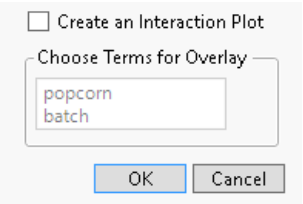
Show Connected Points Shows or hides one or more lines that connect the least squares means for each level in the plot.

Remove Removes the Least Squares Means Plot report for the specified effect.

Least Squares Means Plot Options

When you select the LSMeans Plot option from the red triangle menu of an interaction effect, the Least Squares Means Plot Options window appears.

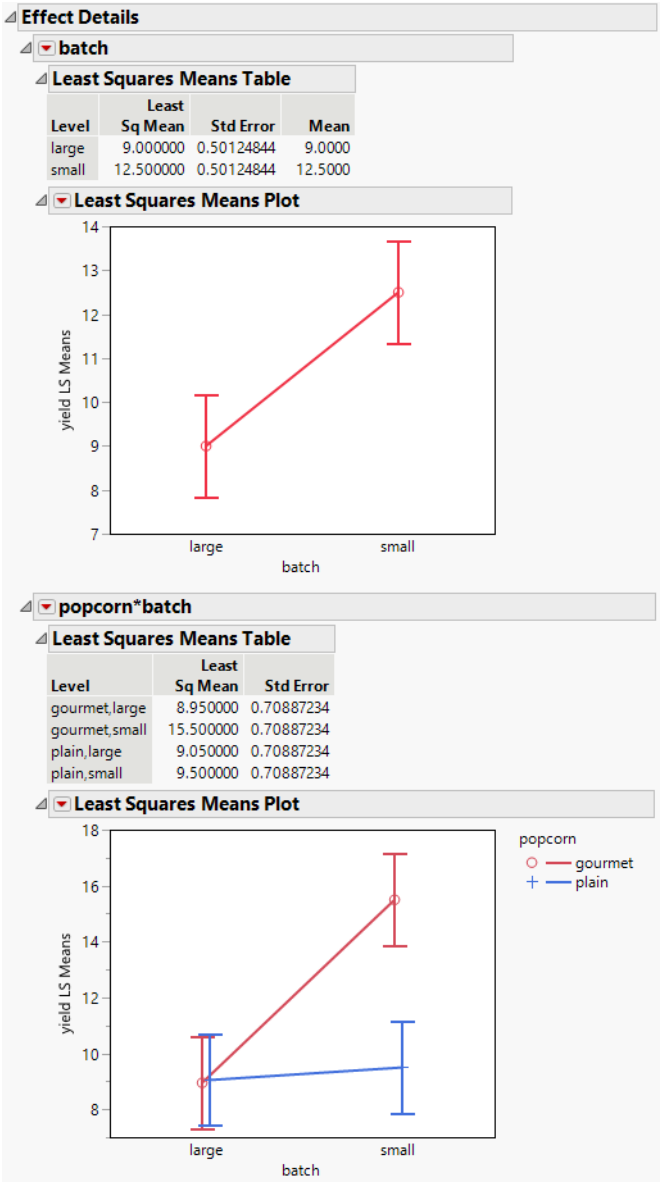
Figure 3.11 Least Squares Means Plot Options Window



If you click OK without selecting anything in the window, one LSMeans Plot appears. The horizontal axis of the plot consists of the levels of the factors nested to obtain a separate effect for each combination. To create an interaction plot, select the box next to Create an Interaction Plot. The Choose Terms for Overlay option enables you to select which effect is displayed as the overlay variable in the interaction plot.

For a three-way interaction term, a second panel of options appears after you choose an overlay variable for the interaction plot. If you click OK without selecting anything in the second panel, one interaction plot appears. Alternatively, use the second panel of options to create separate interaction plots for each level of the effect that you select under Choose Terms for Separate Plots.

Figure 3.12 Least Squares Means Tables and Plots for Two Effects



Example of an LS Means Plot

To create the report in Figure 3.12, follow these steps:

1. Select **Help > Sample Data Library** and open Popcorn.jmp.
2. Select **Analyze > Fit Model**.
3. Select yield and click **Y**.

4. Select popcorn, oil amt, and batch and click **Macros > Full Factorial**. Note that the Emphasis changes to Effect Screening.
5. Click **Run**.
6. Click the Effect Details disclosure icon to show the details for the seven model effects.
7. Click the batch red triangle and select **LSMeans Plot**.
8. Click the popcorn*batch red triangle and select **LSMeans Plot**. The Least Squares Means Plot Options window appears.
9. In the Least Squares Means Plot Options window, click the box next to Create an Interaction Plot.
10. Under Choose Terms for Overlay, select popcorn.
11. Click **OK**.
12. To transpose the factors in the plot for popcorn*batch, repeat step 8 and step 9.
13. Under the Choose Terms for Overlay, select batch and click **OK**.

Figure 3.13 LSMeans Plot for Interaction with Factors Transposed

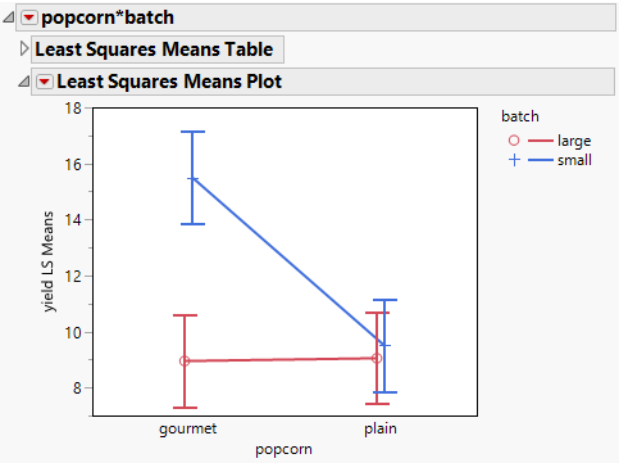


Figure 3.13 shows the popcorn*batch interaction plot with the factors transposed. Compare it with the plot in Figure 3.12. These plots depict the same information but, depending on your interest, one might be more intuitive than the other.

LSMeans Contrast

A *contrast* is a linear combination of parameter values. In the Contrast Specification window, you can specify multiple contrasts and jointly test whether they are zero (Figure 3.14).

JMP builds contrasts in terms of the least squares means of the effect. Each column of the contrast is normalized to have sum zero and so that the sum of the absolute values equals two. If a contrast involves a covariate, you can specify the value of the covariate at which to test the contrast.

The Contrast Specification box shows the name of the effect and the names of the levels in the effect. The contrast values are initially set to zero and appear next to cells containing + and - signs. Click these buttons to compare levels.

Each time you click the + or - button, the contrast coefficients are normalized to make their sum zero and their absolute sum equal to two, if possible. To compare additional levels, click the **New Column** button. A new column appears in which you define a new contrast. After you are finished, click **Done**. The Contrast report appears (Figure 3.15). The overall test is a joint *F* test for all contrasts.

Note: If you attempt to specify more than the maximum number of contrasts possible, the test automatically evaluates.

The Contrast report provides the following details about the joint *F* test:

SS The sum of squares for the joint test.

NumDF The numerator degrees of freedom.

DenDF The denominator degrees of freedom.

F Ratio The ratio of SS divided by NumDF divided by the mean square error.

Prob > F The *p*-value for the significance test.

Test Detail Report

The Test Detail report (Figure 3.15) shows a column for each contrast that you tested. For each contrast, the report gives its estimated value, its standard error, a *t* ratio for a test of that single contrast, the corresponding *p*-value, its sum of squares, and a confidence interval for the contrast estimate.

Parameter Function Report

The Parameter Function report (Figure 3.15) shows the contrasts that you specified expressed as linear combinations of the terms of the model.

Example of LSMeans Contrast

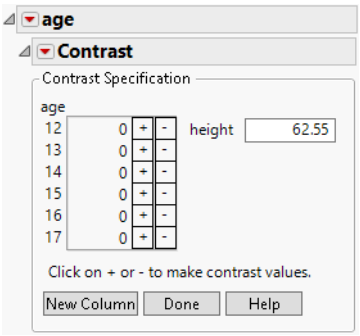
To illustrate the LSMeans Contrast option, form a contrast that compares the first two age levels with the next two levels.

Follow these steps to create the report shown in Figure 3.14.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Model**.
3. Select weight and click **Y**.
4. Select age, sex, and height, and click **Add**.
5. Select age in the Select Columns list, select height in the Construct Model Effects list, and click **Cross**.
6. Click **Run**.

   The Fit Least Squares report appears.
7. Click the age red triangle and select **LSMeans Contrast**.

Figure 3.14 LSMeans Contrast Specification for age

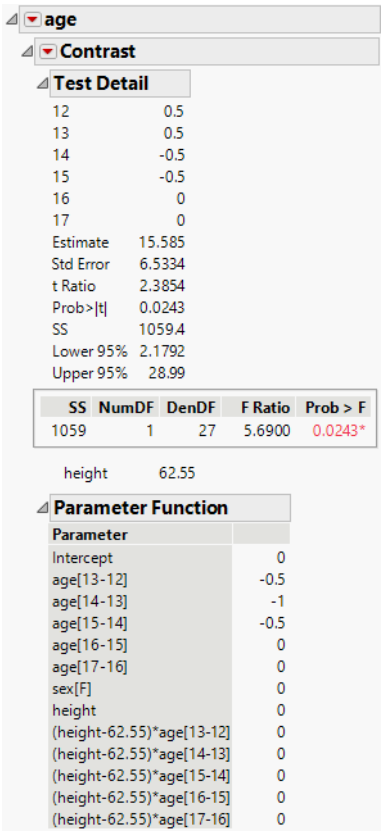


8. Click "+" for the ages 12 and 13.
9. Click "-" for ages 14 and 15.

   This contrast tests whether the mean weights differ for the two age groups, based on predicted values at a height of 62.55.
10. Note that there is a text box next to the continuous effect height. The default value is the mean of the continuous effect.
11. Click **Done**.
12. Open the Test Detail and Parameter Function reports.

The Contrast report is shown in Figure 3.15. The test for the contrast is significant at the 0.05 level. You conclude that the predicted weight for age 12 and 13 children differs statistically from the predicted weight for age 14 and 15 children at the mean height of 62.55.

Figure 3.15 LSMeans Contrast Report



age

Contrast

Test Detail

12	0.5
13	0.5
14	-0.5
15	-0.5
16	0
17	0
Estimate	15.585
Std Error	6.5334
t Ratio	2.3854
Prob>|t|	0.0243
SS	1059.4
Lower 95%	2.1792
Upper 95%	28.99

SS	NumDF	DenDF	F Ratio	Prob > F
1059	1	27	5.6900	0.0243*

height 62.55

Parameter Function

Parameter	
Intercept	0
age[13-12]	-0.5
age[14-13]	-1
age[15-14]	-0.5
age[16-15]	0
age[17-16]	0
sex[F]	0
height	0
(height-62.55)*age[13-12]	0
(height-62.55)*age[14-13]	0
(height-62.55)*age[15-14]	0
(height-62.55)*age[16-15]	0
(height-62.55)*age[17-16]	0

LSMeans Student's t and LSMeans Tukey HSD

The LSMeans Student's t and LSMeans Tukey HSD (*honestly significant difference*) options test pairwise comparisons of model effects.

- The LSMeans Student's t option is based on the usual independent samples, equal variance *t* test. Each comparison is based on the specified significance level. The overall error rate resulting from conducting multiple comparisons exceeds that specified significance level.
- The LSMeans Tukey HSD option conducts Tukey HSD tests. For these comparisons, the significance level applies to the entire collection of pairwise comparisons. For this reason, confidence intervals for LS Means Tukey HSD are wider than those for LSMeans Student's t. The significance level is exact if the sample sizes are equal and conservative if the sample sizes differ (Hayter 1984).

Figure 3.16 shows the LSMeans Tukey report for the effect age in the Big Class.jmp sample data table. (To obtain this report, run the **Fit Model** data table script, click the age red triangle, and select **LS Means Tukey HSD**.) By default, the report shows the Crosstab Report and the Connecting Letters Report.

Figure 3.16 LSMeans Tukey HSD Report

LSMeans Differences Tukey HSD

α= 0.050 Q= 3.02917

LSMean[i]	Mean[i]-Mean[j] Std Err Dif Lower CL Dif Upper CL Dif	LSMean[j] 12	13	14	15	16	17
12		0	13.6856	25.8473	19.7698	10.159	-2.5203
		0	6.96659	7.24776	8.04937	9.96733	10.5955
		0	-7.4174	3.89261	-4.613	-20.034	-34.616
		0	34.7886	47.8019	44.1527	40.3517	29.5754
13		-13.686	0	12.1617	6.08426	-3.5266	-16.206
		6.96659	0	6.82786	7.59798	9.69566	10.0859
		-34.789	0	-8.521	-16.931	-32.896	-46.758
		7.41737	0	32.8444	29.0998	25.8432	14.346
14		-25.847	-12.162	0	-6.0774	-15.688	-28.368
		7.24776	6.82786	0	6.28014	8.57237	8.65355
		-47.802	-32.844	0	-25.101	-41.655	-54.581
		-3.8926	8.52105	0	12.9462	10.2789	-2.1545
15		-19.77	-6.0843	6.07741	0	-9.6108	-22.29
		8.04937	7.59798	6.28014	0	9.23282	9.21068
		-44.153	-29.1	-12.946	0	-37.579	-50.191
		4.61302	16.9313	25.101	0	18.3569	5.61058
16		-10.159	3.52657	15.6882	9.61083	0	-12.679
		9.96733	9.69566	8.57237	9.23282	0	10.8916
		-40.352	-25.843	-10.279	-18.357	0	-45.672
		20.0337	32.8963	41.6554	37.5786	0	20.3131
17		2.52026	16.2058	28.3675	22.2901	12.6793	0
		10.5955	10.0859	8.65355	9.21068	10.8916	0
		-29.575	-14.346	2.15448	-5.6106	-20.313	0
		34.6159	46.7577	54.5806	50.1908	45.6716	0

Level		Least Sq Mean
17	A	121.90696
12	A	119.38671
16	A B	109.22769
13	A B	105.70112
15	A B	99.61686
14	B	93.53945

The Crosstab Report

Both options display a matrix, called the Crosstab Report, where each cell contains the difference in means, the standard error of the difference, and lower and upper confidence limits. The significance level and corresponding critical value are given above the matrix. The default significance level is 0.05, but you can specify a different significance level in the Fit Model launch window. Cells that correspond to pairs of means that differ statistically are shown in red.

The Connecting Letters Report

A Connecting Letters Report appears by default beneath the Crosstab matrix. Levels that share, or are connected by, the same letter do not differ statistically. Levels that are not connected by a common letter do differ statistically.

In Figure 3.16, levels 17, 12, 16, 13, and 15 are connected by the letter A. The connection indicates that these levels do not differ at the 0.05 significance level. Also, levels 16, 13, 15, and 14 are connected by the letter B, indicating that they do not differ statistically. However, ages 17 and 14, and ages 12 and 14, are not connected by a common letter, indicating that these two pairs of levels are statistically different.

Tip: Right-click in the connecting letters report and select Columns to add columns containing connecting letters (Letters), standard errors (Std Error), and confidence interval limits (Lower X% and Upper X%). In the Letters column, the connecting letters are concatenated into a single column. The significance and confidence levels are determined by the significance level that you specify in the Fit Model launch window using the Set Alpha Option.

LSMeans Student's t and LSMeans Tukey HSD Options

The red triangle options that appear in each report window show or hide optional reports. All of the options below are available for LSMeans Student's t. The first four options are available for LSMeans Tukey HSD. For both LSMeans Student's t and LSMeans Tukey HSD, the Crosstab Report and the Connecting Letters Report are shown by default.

Crosstab Report Shows a two-way table that provides, for each pair of levels, the difference in means, the standard error of the difference, and confidence limits for the difference. The contents of cells containing significant differences are highlighted in red.

Connecting Letters Report Illustrates significant and non-significant comparisons with connecting letters. Levels not connected by the same letter are significantly different. Levels connected by the same letter are not significantly different.

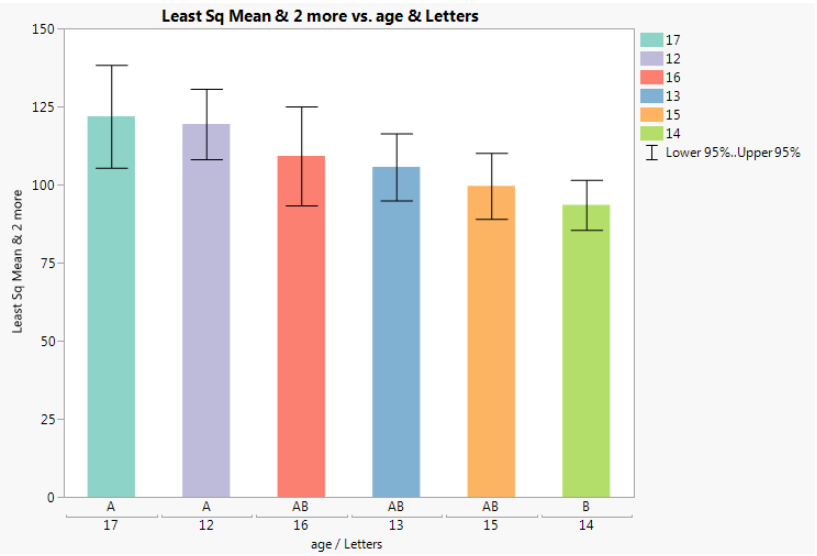
Save Connecting Letters Table Creates a data table whose columns give the levels of the effect, the connecting letters, the least squares means, their standard errors, and confidence intervals. The table contains a script called Bar Chart that produces a colored bar chart of the least squares means with their confidence intervals superimposed. The levels are arranged in decreasing order of least squares means.

Figure 3.17 shows the bar chart for an example based on Big Class.jmp. Run the **Fit Model** data table script, click the age red triangle, and select **LSMeans Tukey HSD**. Select **Save Connecting Letters Table** from the LSMeans Differences Tukey HSD report. Run the **Bar Chart** script in the data table that appears.

Ordered Differences Report Ranks the differences from largest to smallest, giving standard errors, confidence limits, and *p*-values. Also plots the differences on a bar chart with overlaid confidence intervals.

Detailed Comparisons Gives individual detailed reports for each comparison. For a given comparison, the report shows the estimated difference, standard error, confidence interval, *t* ratio, degrees of freedom, and *p*-values for one- and two-sided tests. Also shown is a plot of the *t* distribution, which illustrates the significance test for the comparison. The area of the shaded portion is the *p*-value for a two-sided test.

Figure 3.17 Bar Chart from LSMeans Differences HSD Connecting Letters Table



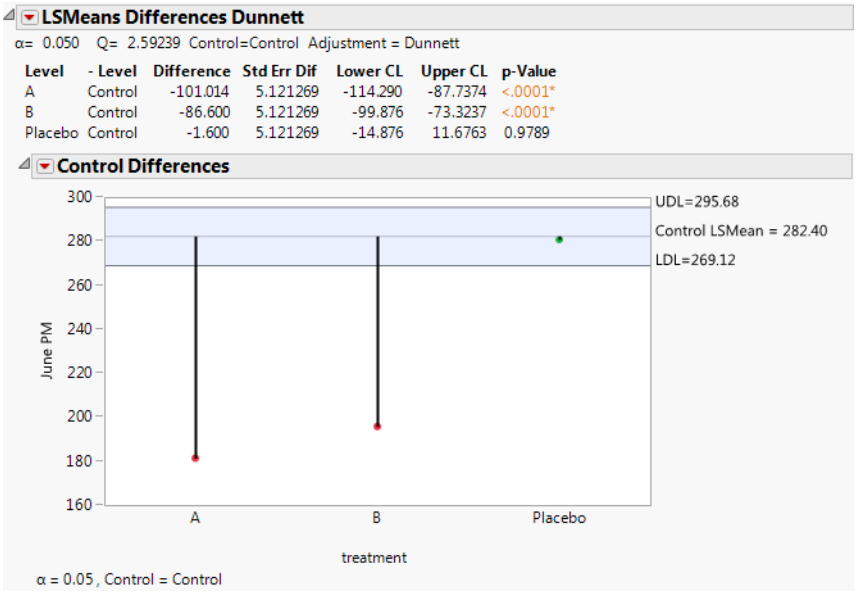
LSMeans Dunnett

Dunnett's test (Dunnett 1955) compares a set of means against the mean of a control group. The error rate applies to the collection of pairwise comparisons. The LSMeans Dunnett option conducts Dunnett's test for the levels of the given effect. Hsu's factor analytical approximation is used for the calculation of *p*-values and confidence intervals (Hsu 1992).

When you select LSMeans Dunnett, you are prompted to enter a control level for the effect. The LS Means Differences Hsu-Dunnett report shows the significance level, the value of the test statistic (Q), and the control level.

A report for the LSMeans Dunnett option for effect treatment in the Cholesterol.jmp sample data table is shown in Figure 3.18. Here, the response is June PM and the level of treatment called Control is specified as the control level.

Figure 3.18 LSMeans Dunnett Report



The report has two options:

Control Differences Report The Control Differences report is shown by default. For each level of the effect, a table shows the following information: the level being compared to the control level, the estimated difference, the standard error of the difference, a confidence interval, and the *p*-value for the comparison.

Control Differences Chart For each level other than the control, a point shows the difference between the LS Mean for that level and the LS Mean for the control level. Upper and lower decision limits (UDL, LDL) are plotted. The report has a Show Summary Report option and Display options. The Show Summary Report option gives the plot detail. The Display options enable you to modify the plot appearance.

Test Slices

The Test Slices option is enabled for interaction effects composed of nominal or ordinal columns. For each level of each nominal or ordinal column in the interaction, this option produces a report that jointly tests all pairwise comparisons of settings involving that level. The test is effectively a test of differences within the specified “slice” of the interaction.

Suppose that you are interested in an A*B*C interaction, where one of the levels of A is “Small”. The Test Slice report for the slice A = Small jointly tests all pairwise comparisons of the B*C levels when A = Small. It enables you to detect differences in levels within an interaction.

The Test Slice reports follow the same format as do the LSMeans Contrast reports. See "LSMeans Contrast".

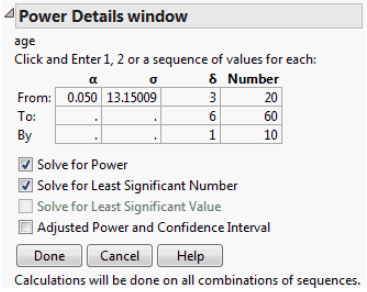
Power Analysis

Opens the Power Details window, where you can enter information to obtain retrospective or prospective details for the *F* test of a specific effect.

Note: To ensure that your study includes sufficiently many observations to detect the required differences, use information about power when you *design* your experiment. Such an analysis is called a *prospective* power analysis. Consider using the DOE platform to design your study. Both DOE > Sample Size and Power and DOE > Evaluate Design are useful for prospective power analysis. For an example of a prospective power analysis using standard least squares, see "Prospective Power Analysis".

Figure 3.19 shows an example of the Power Details window for the Big Class.jmp sample data table. Using the Power Details window, you can explore power for values of alpha (α), sigma (σ), delta (δ), and Number (study size). Enter a single value (From only), two values (From and To), or the start (From), stop (To), and increment (By) for a sequence of values. Power calculations are reported for all possible combinations of the values that you specify.

Figure 3.19 Power Details Window



See "Power Analysis".

The Power Details window report contains the following columns and options:

Alpha (α) The significance level of the test. This value is between 0 and 1, and is often 0.05, 0.01, or 0.10. The initial value for Alpha, shown in the first row, is 0.05, unless you have selected Set Alpha Level and set a different value in the Fit Model launch window.

Sigma (σ) An estimate of the residual error in the model. The initial value shown in the first row, provided for guidance, is the RMSE (the square root of the mean square error).

Delta (δ) The effect size of interest. See "Effect Size". The initial value, shown in the first row, is the square root of the sum of squares for the hypothesis divided by the square root of the number of observations in the study (that is, $\delta = \sqrt{SS/n}$).

Number (*n*) The sample size. The initial value, shown in the first row, is the number of observations in the current study.

Solve for Power Solves for the power as a function of α, σ, δ, and *n*. The power is the probability of detecting a difference of size δ by seeing a test result that is significant at level α, for the specified σ and *n*. See "Computations for the Power".

Solve for Least Significant Number Solves for the smallest number of observations required to obtain a test result that is significant at level α, for the specified δ and σ. See "Computations for the LSN".

Solve for Least Significant Value Solves for the smallest positive value of a parameter or linear function of the parameters that produces a *p*-value of α. The least significant value is a function of α, σ, and *n*. This option is available only for one-degree-of-freedom tests. See "Computations for the LSV".

Adjusted Power and Confidence Interval Retrospective power calculations use estimates of the standard error and the test parameters in estimating the *F* distribution's noncentrality parameter. *Adjusted power* is retrospective power calculation based on an estimate of the noncentrality parameter from which positive bias has been removed (Wright and O'Brien 1988).

The confidence interval for the adjusted power is based on the confidence interval for the noncentrality estimate.

The adjusted power deals with a sample estimate, so it and its confidence limits are computed only for the δ estimated in the current study. See "Computations for the Adjusted Power".

Lack of Fit

The Lack of Fit report gives details for a test that assesses whether the model fits the data well. The Lack of Fit report appears only when it is possible to conduct this test. The test relies on the ability to estimate the variance of the response using an estimate that is independent of the model. Constructing this estimate requires that response values are available at replicated values of the model effects. The test involves computing an estimate of *pure error*, based on a sum of squares, using these replicated observations.

In the following situations, the Lack of Fit report does not appear because the test statistic cannot be computed:

- There are no replicated points with respect to the *X* variables, so it is impossible to calculate a pure error sum of squares.

- The model is *saturated*, meaning that there are as many estimated parameters as there are observations. Such a model fits perfectly, so it is impossible to assess lack of fit.

The difference between the error sum of squares from the model and the pure error sum of squares is called the *lack of fit* sum of squares. The lack of fit variation can be significantly greater than pure error variation if the model is not adequate. For example, you might have the wrong functional form for a predictor, or you might not have enough, or the correct, interaction effects in your model.

The Lack of Fit report contains the following columns:

Source The three sources of variation: Lack of Fit, Pure Error, and Total Error.

DF The degrees of freedom (DF) for each source of error:

– The DF for Total Error is the same as the DF value found on the Error line of the Analysis of Variance table. Based on the sum of squares decomposition, the Total Error DF is partitioned into degrees of freedom for Lack of Fit and for Pure Error.

– The Pure Error DF is pooled from each replicated group of observations. In general, if there are g groups and if each group has identical settings for each effect, the pure error DF, denoted DF_{PE}, is defined as follows:

$$\mathrm{DF}_{PE} = \sum_{i=1}^{g} (n_i - 1)$$

where n_i is the number of replicates in the i^{th} group.

– The Lack of Fit DF is the difference between the Total Error and Pure Error DFs.

Sum of Squares The associated sum of squares (SS) for each source of error:

– The Total Error SS is the sum of squares found on the Error line of the corresponding Analysis of Variance table.

– The Pure Error SS is the total of the sum of squares values for each replicated group of observations. The Pure Error SS divided by its DF estimates the variance of the response at a given predictor setting. This estimate is unaffected by the model. In general, if there are g groups and if each group has identical settings for each effect, the Pure Error SS, denoted SS_{PE}, is defined as follows:

$$\mathrm{SS}_{PE} = \sum_{i=1}^{g} \mathrm{SS}_i$$

where SS_i is the sum of the squared differences between each observed response and the mean response for the i^{th} group.

– The Lack of Fit SS is the difference between the Total Error and Pure Error sum of squares.

Mean Square The mean square for the Source, which is the Sum of Squares divided by the DF. A Lack of Fit mean square that is large compared to the Pure Error mean square suggests that the model is not fitting well. The *F* ratio provides a formal test.

F Ratio The ratio of the Mean Square for Lack of Fit to the Mean Square for Pure Error. The *F* Ratio tests the hypothesis that the variances estimated by the Lack of Fit and Pure Error mean squares are equal, which is interpreted as representing "no lack of fit".

Prob > F The *p*-value for the Lack of Fit test. A small *p*-value indicates a significant lack of fit.

Max RSq The maximum RSquare that can be achieved by a model based only on these effects. The Pure Error Sum of Squares is invariant to the form of the model. So the largest amount of variation that a model with these replicated effects can explain equals:

$$\frac{SS(\text{C. Total}) - SS(\text{Pure Error})}{SS(\text{C. Total})} = 1 - \frac{SS(\text{Pure Error})}{SS(\text{C. Total})}$$

This formula defines the Max RSq.

Estimates

The Estimates menu provides additional detail about model parameters. To better understand estimates, you might want to review how JMP codes nominal and ordinal effects. See "Details of Custom Test Example", "Nominal Factors", and "Ordinal Factors".

If your model contains random effects, then only the options below that are appropriate are available from the Estimates menu.

The Estimates menu provides the following options:

Show Prediction Expression Shows or hides the Prediction Expression report, which contains the equation for the estimated model. See "Show Prediction Expression" for an example.

Sorted Estimates Shows or hides the Sorted Parameter Estimates report, which can be useful in screening situations. If the design is not saturated, this report is the Parameter Estimates report with the terms, other than the Intercept, sorted in decreasing order of significance. If the design is saturated, then Pseudo *t* tests are provided. See "Sorted Estimates".

Expanded Estimates (Available only when at least one of the effects is not continuous.) Shows or hides the Expanded Estimates report, which expands the Parameter Estimates report by giving parameter estimates for all levels of nominal effects. See "Expanded Estimates".

Indicator Parameterization Estimates (Available only when there are nominal columns among the model effects.) Shows or hides the Indicator Function Parameterization report, which contains parameter estimates with the nominal effects in the model parametrized using the classical indicator functions. See "Indicator Parameterization Estimates".

Sequential Tests Shows or hides the Sequential (Type 1) Tests report that contains the sums of squares as effects are added to the model sequentially. Conducts *F* tests based on the sequential sums of squares. See "Sequential Tests".

Custom Test Enables you to test a custom hypothesis. See "Custom Test".

Multiple Comparisons Enables you to specify comparisons among effect levels. These comparisons can involve a single effect or you can define flexible custom comparisons. You can compare to the overall mean, to a control mean, or you can obtain all pairwise comparisons using Tukey HSD or Student's *t*. When you specify the Student's *t* method, you can also perform equivalence tests to identify pairwise differences that are of practical importance. See "Multiple Comparisons".

Compare Slopes (Available only when there is one nominal term, one continuous term, and their interaction effect for the fixed effects.) Produces a report that enables you to compare the slopes of each level of the interaction effect in an analysis of covariance (ANCOVA) model. See "Compare Slopes".

Joint Factor Tests (Available only when the model contains interactions.) For each main effect in the model, shows or hides a joint test on all of the parameters involving that main effect. See "Joint Factor Tests".

Inverse Prediction Enables you to predict values of explanatory variables for one or more values of the response. See "Inverse Prediction".

Cox Mixtures (Available only when the model contains mixture effects.) Produces parameter estimates for the Cox mixture model. Using these to derive factor effects and estimate the response surface shape relative to a reference point in the design space. See "Cox Mixtures".

Parameter Power Adds columns to the Parameter Estimates report that give power and other details relating to the corresponding hypothesis tests. See "Parameter Power".

Correlation of Estimates Shows or hides a correlation matrix for all parameter estimates in the model. See "Correlation of Estimates".

Error Specification (Available only when there are no random effects.) Specifies the error variance and the error degrees of freedom that are used for standard errors and tests in the Fit Least Squares report. Note that the Studentized Residuals plot and the Box Cox Transformations report are not affected by changing the Error Specification. When the Error Specification is Pure Error or Specified, an additional column appears in the Analysis of Variance report. See "Analysis of Variance".

Default Estimate Uses the standard root mean square error and error degrees of freedom from the model to calculate all tests and standard errors.

Pure Error Uses the Pure Error mean square and associated degrees of freedom from the Lack of Fit report to calculate all tests and standard errors. See "Lack of Fit".

Caution: If the pure error degrees of freedom is 1, a warning message is displayed indicating that tests are weak and confidence limits are large.

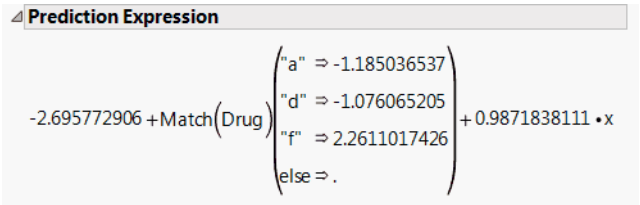
Specified Uses user-specified values for the error variance and error degrees of freedom to calculate all tests and standard errors.

Show Prediction Expression

The Show Prediction Expression option shows the equation used to predict the response. Figure 3.20 shows an example for the Drug.jmp sample data table. This expression is given as a typical JMP formula. For example, to predict the response for someone on Drug a with x = 10, you would calculate, with some rounding: –2.696 - 1.185 + 0.987(10) = 5.99.

Tip: To specify the number of digits in the prediction formula, go to **File > Preferences > Tables** and change the **Default Field Width** value.

Figure 3.20 Prediction Expression



To obtain the preceding report, follow these steps:

Example of a Prediction Expression

1. Select **Help > Sample Data Library** and open Drug.jmp.
2. Select **Analyze > Fit Model**.
3. Select y and click **Y**.
4. Select Drug and x, and then click **Add**.
5. Click **Run**.
6. Click the Response y red triangle and select **Estimates > Show Prediction Expression**. The report in Figure 3.20 appears.

Sorted Estimates

The Sorted Estimates option produces a version of the Parameter Estimates report that is useful in screening situations. If the design is not saturated, the Sorted Estimates report gives the information found in the Parameter Estimates report, but with the terms, other than the Intercept, sorted in decreasing order of significance (second report in Figure 3.21). If the design is saturated, then Pseudo *t* tests are provided. These are based on Lenth's *pseudo standard error* (Lenth 1989). See "Lenth's PSE".

Example of a Sorted Estimates Report

1. Select **Help > Sample Data Library** and open Reactor.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Make sure that 2 appears in the **Degree** box near the bottom of the window.
5. Select F, Ct, A, T, and Cn and click **Macros > Factorial to Degree**.
6. Click **Run**.
7. Click the Response Y red triangle and select **Estimates > Sorted Estimates**.

Figure 3.21 Sorted Parameter Estimates

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	65.5	0.565962	115.73	<.0001*
F	-0.6875	0.565962	-1.21	0.2421
Ct	9.75	0.565962	17.23	<.0001*
A	-0.3125	0.565962	-0.55	0.5885
T	5.375	0.565962	9.50	<.0001*
Cn	-3.125	0.565962	-5.52	<.0001*
F*Ct	0.6875	0.565962	1.21	0.2421
F*A	0.375	0.565962	0.66	0.5170
F*T	-0.4375	0.565962	-0.77	0.4508
F*Cn	0.0625	0.565962	0.11	0.9134
Ct*A	0.4375	0.565962	0.77	0.4508
Ct*T	6.625	0.565962	11.71	<.0001*
Ct*Cn	1	0.565962	1.77	0.0963
A*T	1.0625	0.565962	1.88	0.0788
A*Cn	0.4375	0.565962	0.77	0.4508
T*Cn	-5.5	0.565962	-9.72	<.0001*

Effect Tests

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Ct	9.75	0.565962	17.23	<.0001*
Ct*T	6.625	0.565962	11.71	<.0001*
T*Cn	-5.5	0.565962	-9.72	<.0001*
T	5.375	0.565962	9.50	<.0001*
Cn	-3.125	0.565962	-5.52	<.0001*
A*T	1.0625	0.565962	1.88	0.0788
Ct*Cn	1	0.565962	1.77	0.0963
F	-0.6875	0.565962	-1.21	0.2421
F*Ct	0.6875	0.565962	1.21	0.2421
F*T	-0.4375	0.565962	-0.77	0.4508
Ct*A	0.4375	0.565962	0.77	0.4508
A*Cn	0.4375	0.565962	0.77	0.4508
F*A	0.375	0.565962	0.66	0.5170
A	-0.3125	0.565962	-0.55	0.5885
F*Cn	0.0625	0.565962	0.11	0.9134

The Sorted Parameter Estimates report also appears automatically if the Emphasis is set to Effect Screening and all of the effects have only one parameter.

Note the following differences between the Parameter Estimates report and the Sorted Parameter Estimates report (both shown in Figure 3.21):

- The Sorted Parameter Estimates report does not show the intercept.
- The effects are sorted by the absolute value of the *t* ratio, showing the most significant effects at the top.
- A bar chart shows the *t* ratio with vertical lines showing critical values for the 0.05 significance level.

Sorted Estimates Report for Saturated Models

Screening experiments often involve fully saturated models, where there are not enough degrees of freedom to estimate error. In these cases, the Sorted Estimates report (Figure 3.21) gives relative standard errors and constructs *t* ratios and *p*-values using Lenth's pseudo standard error (PSE). These quantities are labeled with *Pseudo* in their names. See "Lenth's PSE" and "Pseudo t-Ratios". A note explains the change and shows the PSE.

The report contains the following columns:

Term The model term whose coefficient is of interest.

Estimate The parameter estimates are presented in sorted order, with smallest *p*-values listed first.

Relative Standard Error If there are no degrees of freedom for residual error, the report gives relative standard errors. The relative standard error is computed by setting the root mean square error equal to 1.

Pseudo t-Ratio A *t* ratio for the estimate, computed using pseudo standard error. The value of Lenth PSE is shown in a note at the bottom of the report.

Pseudo p-Value A *p*-value computed using an error degrees of freedom value (DFE) of $m/3$, where m is the number of parameters other than the intercept. The value of DFE is shown in a note at the bottom of the report.

Lenth's PSE

Lenth's *pseudo standard error* (PSE) is an estimate of residual error due to Lenth (1989). It is based on the principle of effect sparsity: in a screening experiment, relatively few effects are active. The inactive effects represent random noise and form the basis for Lenth's estimate.

The value is computed as follows:

1. Consider the absolute values of all non-intercept parameters.

2. Remove all parameter estimates whose absolute values exceed 3.75 times the median absolute estimate.
3. Multiply the median of the remaining absolute values of parameter estimates by 1.5.

Pseudo t-Ratios

When relative standard errors are equal, Lenth's PSE is shown in a note at the bottom of the report. The Pseudo *t*-Ratio is calculated as follows:

$$\text{Pseudo } t\text{-Ratio} = \frac{\text{Estimate}}{\text{PSE}}$$

When relative standard errors are not equal, the TScale Lenth PSE is computed. This value is the PSE of the estimates divided by their relative standard errors. The Pseudo *t*-Ratio is calculated as follows:

$$\text{Pseudo } t\text{-Ratio} = \frac{\text{Estimate}}{\text{TScale Lenth PSE} \times \text{Relative Std Error}}$$

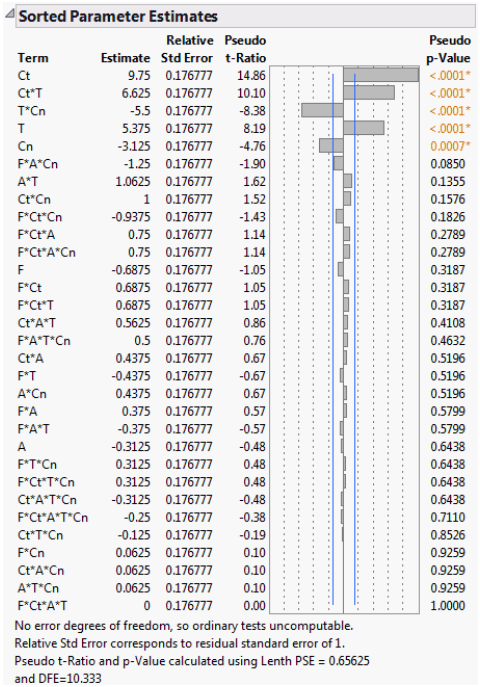
Note that, to estimate the standard error for a given estimate, TScale Lenth PSE is adjusted by multiplying it by the estimate's relative standard error.

Example of a Saturated Model

1. Select **Help > Sample Data Library** and open Reactor.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Select the following five columns: F, Ct, A, T, and Cn.
5. Click the **Macros** button and select **Full Factorial**.
6. Click **Run**.
7. Click the Response Y red triangle and select **Estimates > Sorted Estimates**.

   Note that Lenth's PSE and the degrees of freedom used are given at the bottom of the report. The report indicates that, based on their Pseudo p-Values, the effects Ct, Ct*T, T*Cn, T, and Cn are highly significant.

Figure 3.22 Sorted Parameter Estimates Report for Saturated Model



Sorted Parameter Estimates

Term	Estimate	Relative Std Error	Pseudo t-Ratio	Pseudo p-Value
Ct	9.75	0.176777	14.86	<.0001*
Ct*T	6.625	0.176777	10.10	<.0001*
T*Cn	-5.5	0.176777	-8.38	<.0001*
T	5.375	0.176777	8.19	<.0001*
Cn	-3.125	0.176777	-4.76	0.0007*
F*A*Cn	-1.25	0.176777	-1.90	0.0850
A*T	1.0625	0.176777	1.62	0.1355
Ct*Cn	1	0.176777	1.52	0.1576
F*Ct*Cn	-0.9375	0.176777	-1.43	0.1826
F*Ct*A	0.75	0.176777	1.14	0.2789
F*Ct*A*Cn	0.75	0.176777	1.14	0.2789
F	-0.6875	0.176777	-1.05	0.3187
F*Ct	0.6875	0.176777	1.05	0.3187
F*Ct*T	0.6875	0.176777	1.05	0.3187
Ct*A*T	0.5625	0.176777	0.86	0.4108
F*A*T*Cn	0.5	0.176777	0.76	0.4632
Ct*A	0.4375	0.176777	0.67	0.5196
F*T	-0.4375	0.176777	-0.67	0.5196
A*Cn	0.4375	0.176777	0.67	0.5196
F*A	0.375	0.176777	0.57	0.5799
F*A*T	-0.375	0.176777	-0.57	0.5799
A	-0.3125	0.176777	-0.48	0.6438
F*T*Cn	0.3125	0.176777	0.48	0.6438
F*Ct*T*Cn	0.3125	0.176777	0.48	0.6438
Ct*A*T*Cn	-0.3125	0.176777	-0.48	0.6438
F*Ct*A*T*Cn	-0.25	0.176777	-0.38	0.7110
Ct*T*Cn	-0.125	0.176777	-0.19	0.8526
F*Cn	0.0625	0.176777	0.10	0.9259
Ct*A*Cn	0.0625	0.176777	0.10	0.9259
A*T*Cn	0.0625	0.176777	0.10	0.9259
F*Ct*A*T	0	0.176777	0.00	1.0000

No error degrees of freedom, so ordinary tests uncomputable.
Relative Std Error corresponds to residual standard error of 1.
Pseudo t-Ratio and p-Value calculated using Lenth PSE = 0.65625 and DFE=10.333

Expanded Estimates

In dealing with parameter estimates, you must understand how JMP codes nominal and ordinal columns. For more information about how nominal columns are coded, see "Details of Custom Test Example". For more information about how ordinal columns are coded and modeled, see "Nominal Factors" and "Ordinal Factors".

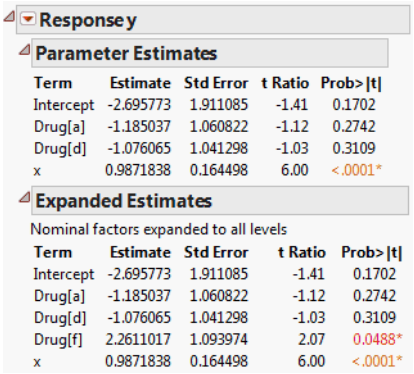
Use the Expanded Estimates option when there are nominal terms in the model and you want to see details for the full set of estimates. The Expanded Estimates option provides the estimates, their standard errors, *t* ratios, and *p*-values.

Example of an Expanded Estimates Report

1. Select **Help > Sample Data Library** and open Drug.jmp.
2. Select **Analyze > Fit Model**.
3. Select y and click **Y**.
4. Select Drug and x, and then click **Add**.
5. Click **Run**.
6. Click the Response y red triangle and select **Estimates > Expanded Estimates**.

The Expanded Estimates report, along with the Parameter Estimates report, is shown in Figure 3.23. Note that an estimate for the term Drug[f] appears in the Expanded Estimates report. The null hypothesis for the test is that the mean for the Drug f group does not differ from the overall mean. The test for Drug[f] is significant at the 0.05 level, suggesting that the mean response for the Drug f group differs from the overall response. See "Interpretation of Tests for Expanded Estimates".

Figure 3.23 Comparison of Parameter Estimates and Expanded Estimates



Response y

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	-2.695773	1.911085	-1.41	0.1702
Drug[a]	-1.185037	1.060822	-1.12	0.2742
Drug[d]	-1.076065	1.041298	-1.03	0.3109
x	0.9871838	0.164498	6.00	<.0001*

Expanded Estimates

Nominal factors expanded to all levels

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	-2.695773	1.911085	-1.41	0.1702
Drug[a]	-1.185037	1.060822	-1.12	0.2742
Drug[d]	-1.076065	1.041298	-1.03	0.3109
Drug[f]	2.2611017	1.093974	2.07	0.0488*
x	0.9871838	0.164498	6.00	<.0001*

Interpretation of Tests for Expanded Estimates

Suppose that your model consists of a single nominal factor that has *n* levels. That factor is represented by *n-1* indicator variables, one for each of *n-1* levels. The parameter estimate corresponding to any one of these *n-1* indicator variables is the difference between the mean response for that level and the average response across all levels. This representation is due to how JMP codes nominal variables. See "Details of Custom Test Example". The parameter estimate is often interpreted as the *effect* of that level.

For example, in the Cholesterol.jmp sample data table, consider the single factor treatment and the response June PM. The parameter estimate associated with the term, or indicator variable, treatment[A] is the difference between the mean of June PM for treatment A and the overall mean of June PM.

The effects across *all* levels of a nominal variable are constrained to sum to zero. Consider the effect of the last level in the level ordering, namely, the level that is coded with –1s. The effect of this level is the negative of the sum of the effects across the other *n-1* levels. It follows that the effect of the last level is the negative of the sum of the parameter estimates across the other *n-1* levels.

The Expanded Estimates option in the Estimates menu calculates missing estimates, tests for all effects that involve nominal columns, and shows them in a text report. You can verify that the mean (or sum) of the estimates across the levels of any such effect is zero. In particular, this relationship indicates that these estimates, and their associated tests, are not independent of each other.

In the Drug.jmp report shown in Figure 3.23, the estimates for the terms associated with Drug are based on a model that includes the covariate x.

Notes:

- The estimate for Drug[a] is the difference between the least squares mean for Drug a and the overall mean of y.
- The estimate for Drug[f], given in the Expanded Estimates report, is the negative of the sum of the estimates for Drug[a] and Drug[d].
- The *t* test for Drug [f] presented in the Expanded Estimates report tests whether the response for the Drug f group differs from the overall mean response.
- If nominal factors are involved in high-degree interactions, the Expanded Estimates report can be lengthy. For example, a five-way interaction of two-level nominal factors produces only one parameter estimate but has $2^5 = 32$ expanded effects, which are all identical up to sign changes.

Indicator Parameterization Estimates

This option displays the Indicator Function Parameterization report, which gives parameter estimates for the model where nominal columns are coded using indicator (SAS GLM) parameterization and are treated as continuous. Ordinal columns remain coded using the usual JMP coding scheme. The SAS GLM and JMP coding schemes are described in "The Factor Models".

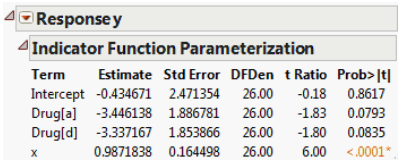
In the JMP coding scheme, the estimate that corresponds to the indicator for a level of a nominal variable is an estimate of the difference between the mean response at that level and the mean response over all the levels. To see the JMP coding, select **Save Columns > Save Coding Table** from the Standard Least Squares red triangle menu.

In the indicator coding scheme, the estimate that corresponds to the indicator for a level of a nominal variable is an estimate of the difference between the mean response at that level and the mean response at the last level. The last level is the level with the highest value order coding; it is the level whose indicator function is not included in the model.

Caution: Standard errors and t-ratios given in the Indicator Function Parameterization report differ from those in the Parameter Estimates report. This is because the estimates are estimating different parameters.

To create the report in Figure 3.24, follow the steps in "Example of an Expanded Estimates Report". But instead of selecting Expanded Estimates, select **Indicator Parameterization Estimates**.

Figure 3.24 Indicator Parameterization Estimates



Responsey

Indicator Function Parameterization

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|
Intercept	-0.434671	2.471354	26.00	-0.18	0.8617
Drug[a]	-3.446138	1.886781	26.00	-1.83	0.0793
Drug[d]	-3.337167	1.853866	26.00	-1.80	0.0835
x	0.9871838	0.164498	26.00	6.00	<.0001*

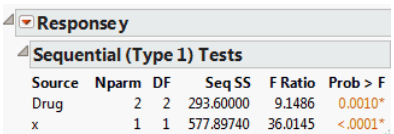
The JMP coding scheme is used for nominal variables throughout JMP with the exception of the Generalized Regression personality of Fit Model. For more information about the coding scheme for nominal variables in the Generalized Regression personality, see "Launch the Generalized Regression Personality".

Note that there might be differences in models derived using the JMP versus the SAS GLM parameterization. Some models are equivalent. Other models (such as no-intercept models, models with missing cells, models with nominal or ordinal effects, and mixture models) might show differences.

Sequential Tests

The Sequential Tests report shows sums of squares and tests as effects are added to the model sequentially. The order of entry is defined by the order of effects as they appear in the Fit Model launch window's Construct Model Effects list. The report in Figure 3.25 is for the Drug.jmp sample data table.

Figure 3.25 Sequential Tests Report



Responsey

Sequential (Type 1) Tests

Source	Nparm	DF	Seq SS	F Ratio	Prob > F
Drug	2	2	293.60000	9.1486	0.0010*
x	1	1	577.89740	36.0145	<.0001*

The sums of squares that form the basis for sequential tests are also called *Type I Sums of Squares*. They are computed by fitting models in steps following the specified entry order of effects. Consider a specific effect. Compute the model sum of squares for a model containing all effects entered *prior* to that effect. Then compute the model sum of squares for a model containing those effects *and* the specified effect. The sequential sum of squares for the specified effect is the increase in the model sum of squares.

Refer to Figure 3.25, showing sequential sums of squares for the Drug.jmp sample data table. In the Fit Model launch window, Drug was entered first, followed by x. A model consisting only of Drug has model sum of squares equal to 293.6. When x is added to the model, the model sum of squares becomes 871.4974. The increase of 577.8974 is the sequential sum of squares for x.

The tests shown in the Sequential (Type 1) Tests report are *F* tests based on sequential sums of squares, also called *Type I Tests*. The *F* Ratio tests the specified effect, where the model contains only that effect and the effects listed above it in the Source column.

The sequential sums of squares sum to the model sum of squares. Another nice feature is that, under the usual model assumptions, the values are statistically independent of each other. However, they do depend on the order of terms in the model and, as such, are not appropriate in many situations.

Sequential tests are considered appropriate in the following situations:

- balanced analysis of variance models specified in proper sequence (that is, two-way interactions follow main effects in the effects list, and so on)
- purely nested models specified in the proper sequence
- polynomial regression models specified in the proper sequence.

The tests given in the Parameter Estimates and Effect Tests reports are based on *Type III Sums of Squares*. Here the sum of squares for an effect is the extra sum of squares explained by the effect after all other effects have been entered in the model.

Custom Test

To test one or more custom hypotheses involving any model parameters, select **Custom Test** from the Estimates menu. In this window, you can specify one or more linear functions, or *contrasts*, of the model parameters.

The results include individual tests for each contrast and a joint test for all contrasts. See Figure 3.26. The report for the individual contrasts gives the estimated value of the specified linear function of the parameters and its standard error. A *t* ratio, its *p*-value, and the associated sum of squares are also provided. Below the individual contrast results, the joint test for all contrasts gives the sum of squares, the numerator degrees of freedom, the *F* ratio, and its *p*-value.

Caution: These tests are conducted using residual error. If you have random effects in your model and if you use EMS instead of REML, then these tests might not be appropriate.

Note: If you are testing for effects that are involved in higher-order effects, consider using a test for least squares means, rather than a custom test. Least squares means are adjusted for other model effects. You can test least squares means contrasts under Effect Details.

Custom Test Report Components

The Custom Test specification window has the following components:

Editable text box The space beneath the Custom Test title bar is an editable area for entering a test name.

Parameter Lists the model terms. To the right of the list of terms are columns of zeros corresponding to the corresponding parameters. Enter values in these cells to specify the linear functions for your tests.

The "=" sign The last line in the Parameter list is labeled =. Enter a constant into this cell to complete the specification for each contrast.

Add Column Adds columns of zeros so that you can jointly test several linear functions of the parameters.

Done Click the Done button to perform the tests. The report changes to show the test statistic value, the standard error, and other statistics for each test column. The joint *F* test for all columns is given in a box at the bottom of the report.

Custom Test Report Options

The Custom Test red triangle menu contains the following options:

Power Analysis Provides a power analysis for the joint test. This option is available only after the test has been conducted. See "Parameter Power".

Remove Removes the Custom Test report.

Note: Select **Estimates > Custom Test** repeatedly to conduct several joint custom tests.

Figure 3.26 shows an example of the specification window with three contrasts, using the Cholesterol.jmp sample data table. Note that the constant is set to zero for all three tests. The report for these tests is shown in Figure 3.27.

Example of a Custom Test

The Cholesterol.jmp sample data table gives repeated measures on 20 patients at six time periods. Four treatment groups are studied. Typically, this data should be properly analyzed using all repeated measures as responses. This example considers only the response for June PM.

Suppose that you want to test three contrasts. You want to compare the mean responses for the following:

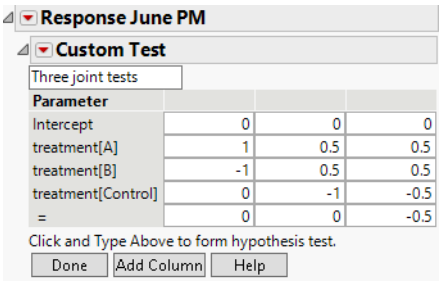
- treatment A to treatment B
- treatments A and B to the control group
- treatments A and B to the control and placebo groups

To test these contrasts using Custom Test:

1. Select **Help > Sample Data Library** and open Cholesterol.jmp.
2. Select **Analyze > Fit Model**.
3. Select June PM and click **Y**.
4. Select treatment and click **Add**.
5. Click **Run**.
6. Click the red triangle next to Response June PM and select **Estimates > Custom Test**.
7. In the Custom Test specification report, click **Add Column** twice to create two additional columns.
8. Fill in the editable area with a test name and enter values in the three columns as shown in Figure 3.26.

   To see how to obtain these values, particularly those in the third column, see "Interpretation of Parameters".

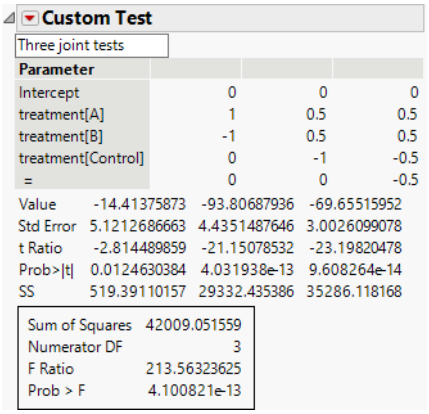
Figure 3.26 Custom Test Specification Report for Three Contrasts



9. Click **Done**.

The results shown in Figure 3.27 indicate that all three hypotheses are individually, as well as jointly, significant.

Figure 3.27 Custom Test Report Showing Tests for Three Contrasts



Custom Test

Three joint tests

Parameter			
Intercept	0	0	0
treatment[A]	1	0.5	0.5
treatment[B]	-1	0.5	0.5
treatment[Control]	0	-1	-0.5
=	0	0	-0.5
Value	-14.41375873	-93.80687936	-69.65515952
Std Error	5.1212686663	4.4351487646	3.0026099078
t Ratio	-2.814489859	-21.15078532	-23.19820478
Prob>|t|	0.0124630384	4.031938e-13	9.608264e-14
SS	519.39110157	29332.435386	35286.118168

Sum of Squares	42009.051559
Numerator DF	3
F Ratio	213.56323625
Prob > F	4.100821e-13

Multiple Comparisons

Use this option to obtain tests and confidence levels that compare means defined by levels of your model effects. The goal of multiple comparisons methods is to determine whether group means differ, while controlling the probability of reaching an incorrect conclusion. The Multiple Comparisons option lets you compare group means with the overall average (Analysis of Means) and with a control group mean. You can also conduct pairwise comparisons using either Tukey HSD or Student's *t*. When you specify the Student's *t* method, you can also perform equivalence tests to identify pairwise differences that are of practical importance.

The Student's *t* method controls only the error rate for an individual comparison. As such, it is not a true multiple comparison procedure. All other methods provided control the overall error rate for all comparisons of interest. Each of these methods uses a multiple comparison *adjustment* in calculating *p*-values and confidence limits.

If your model contains nominal and ordinal effects, you can conduct comparisons using Least Squares Means estimates, or you can define specific comparisons using User-Defined Estimates. If your model contains only continuous effects, you can compare means using User-Defined Estimates.

Tip: Suppose that a continuous effect consists of relatively few levels. If you are interested in comparisons using Least Squares Means Estimates, consider assigning that effect an ordinal (or nominal) modeling type.

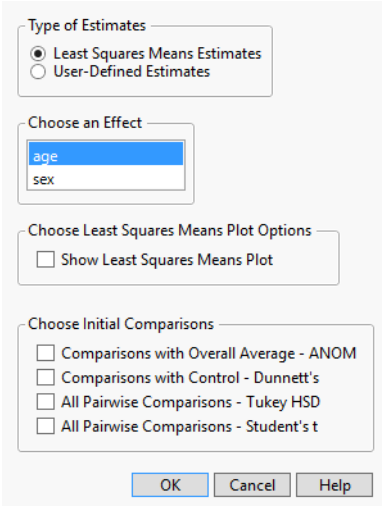
Launch the Option

An example of the control window for the Multiple Comparisons option is shown in Figure 3.28. This example is based on the Big Class.jmp data table, with weight as Y and age, sex, and height as model effects. Two classes of estimates are available for comparisons: Least Squares Means Estimates and User-Defined Estimates.

Least Squares Means Estimates

This option compares least squares means and is available only if there are nominal or ordinal effects in the model. Recall that least squares means are means computed at some neutral value of the other effects in the model. (For a definition of least squares means, see "LSMeans Table".) You must select the effect of interest. In Figure 3.28, Least Squares Means Estimates for age are specified. There is an option to show the least squares means plot. See "Least Squares Means Plot Options".

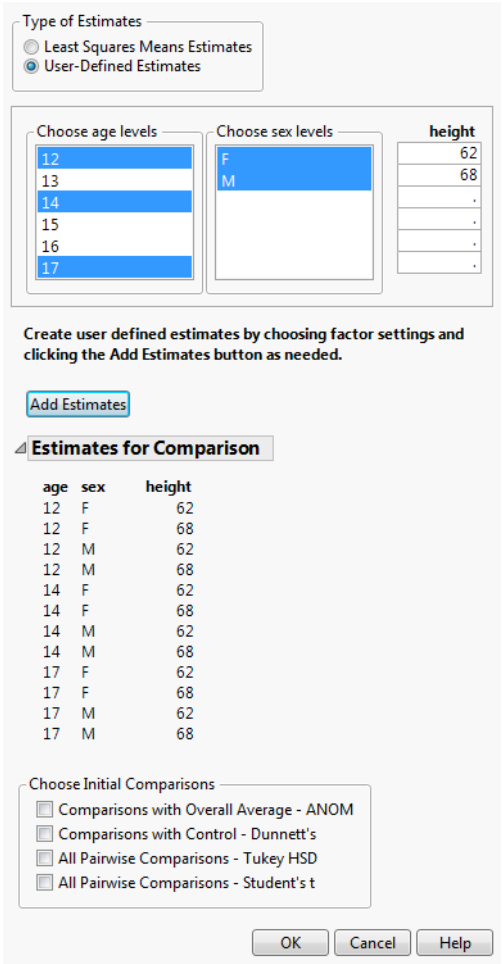
Figure 3.28 Launch Window for Least Squares Means Estimates



User-Defined Estimates

The specification of User-Defined Estimates is illustrated in Figure 3.29. Three levels of age and both levels of sex have been selected. Also, two values of height have been manually entered. The Add Estimates button has been clicked, resulting in the listing of all possible combinations of the specified levels. At this point, you can specify more estimates and click the Estimates button again to add them to the list of Estimates for Comparison.

Figure 3.29 Launch Window for User-Defined Estimates



When you use User-Defined Estimates, effects with no specified levels are set according to the modeling type:

- Continuous effects are set to the mean of the effect.
- Nominal and ordinal effects are set to the first level in the value ordering.

Note: In this section, we use the term *mean* to refer to either estimates of least squares means or user-defined estimates.

Choose Least Squares Means Plot Options

Select **Show Least Squares Means Plot** to obtain a least square means plot. If your effect is an interaction term, then you have the option to create an interaction plot. You select the term for the overlay. If you do not select the interaction plot, then the least squares plot will nest the effect terms. See "Least Squares Means Plot Options".

Choose Initial Comparisons

Once you have specified estimates, you can choose the types of comparisons that you would like to see in your initial report by making selections under Choose Initial Comparisons. Or click OK without making any selections.

Comparisons with Overall Average - ANOM Compares each effect least squares mean with the overall least squares mean. (Analysis of Means).

Comparisons with Control - Dunnett's Compares each effect least squares mean with the least squares mean of a control level.

All Pairwise Comparisons - Tukey HSD Tests all pairwise comparisons of the effect least squares means using the Tukey HSD adjustment for multiplicity.

All Pairwise Comparisons - Student's t Tests all pairwise comparisons of the effect least squares means with no multiplicity adjustment.

Each of these selections opens a report with an area at the top that shows details specific to the report. This information includes the quantile, or critical value. For the true multiple comparisons procedures, the method used for the multiple comparison adjustment is shown. If you have specified User-Defined Estimates, the report displays a list of effects that do not vary relative to the specified estimates and the levels at which these effects are set. Unless you have specified otherwise, any continuous effect is set to its mean. Any nominal or ordinal effect is set to the first level in its value ordering.

If you click OK without selecting from the Choose Initial Comparisons list, the Multiple Comparisons report opens, showing the Least Squares Means Estimates table or the User-Defined Estimates table. From the Multiple Comparison red triangle menu, all of the options listed above are available. The available reports and options are described below.

Least Squares Means or User-Defined Estimates Report

By default, the Multiple Comparisons option displays a Least Squares Means Estimates report or a User-Defined Estimates report, depending on the type of estimates that you selected in the launch window. For each combination of levels of interest, this table gives an estimate of the mean, as well as a test and confidence interval. Specifically, this table contains the following:

Levels of the Categorical Effects The first columns in the report identify the effect or effects of interest. The values in the columns specify the groups being analyzed.

Estimate An estimate of the mean for each group.

Std Error The standard error of the mean for each group.

DF The degrees of freedom for a test of whether the mean is 0.

Lower 95% The lower confidence limit for the mean. You can change the confidence level by selecting Set Alpha Level in the Fit Model window.

Upper 95% The upper confidence limit for the mean.

t Ratio The *t* ratio for the significance test. This column appears only if you right-click in the report and select Columns > t Ratio.

Prob>|t| The *p*-value for the significance test. This column appears only if you right-click in the report and select Columns > Prob>|t|.

Arithmetic Mean Estimate (Appears only in the Least Squares Means Estimates report.) An estimate of the arithmetic mean for each group.

N (Appears only in the Least Squares Means Estimates report.) The number of observations used to calculate the mean for each group.

Note: You can obtain *t* ratios and *p*-values by right-clicking in the table and selecting Columns.

Comparisons with Overall Average

This option compares the means for the specified levels specified to the overall mean for these levels. It displays a table showing confidence intervals for differences from the overall mean and a chart showing decision limits. The method used to make the comparisons is called analysis of means (ANOM) (Nelson et al. 2005). ANOM is a multiple comparison procedure that controls the joint error rate for all pairwise comparisons to the overall mean. See Figure 3.30 for a report based on the Lipid Data.jmp sample data table.

ANOM might appear similar to analysis of variance. However, it is fundamentally different in that it identifies levels with means that differ from the overall mean for all levels. In contrast, analysis of variance tests for differences in the means themselves.

At the top of the Comparisons with Overall Average report, you find:

Quantile The value of Nelson's h statistic used in constructing the decision limits.

Adjusted DF The degrees of freedom used in constructing the decision limits.

Avg The average mean. For least squares estimates, the average mean is a weighted average of the group least squares means. This weighted average represents the overall mean at the neutral settings where the group least squares means are calculated.

Specifically, the average least squares mean is a weighted average with weights inversely proportional to the diagonal entries of the matrix $L(X'X)^{-1}L'$. Here L is the matrix of coefficients used to compute the group least squares means. For a technical definition of least squares means, see the GLM Procedure chapter in SAS Institute Inc. (2020b).

For user-defined estimates, the average mean is defined similarly. However, in this case, L is the matrix of coefficients used to define the estimates.

Adjustment Describes the method used to obtain the critical value:

Nelson Provides exact critical values and p-values. Used whenever possible, in particular, when the estimates are uncorrelated.

Nelson-Hsu Provides approximate critical values and p-values based on Hsu's factor analytical approximation is used (Hsu 1992). Used when exact values cannot be obtained.

Sidak Used when both Nelson and Nelson-Hsu fail.

For technical details, see the GLM Procedure chapter in SAS Institute Inc. (2020b).

Three options are available from the Comparisons with Overall Average report menu:

Differences from Overall Average

For each comparison of a group's mean to the overall mean, this report provides the following details:

- The levels being compared
- Difference - the estimated difference
- Std Error - the standard error of the difference
- Lower and Upper limits for the confidence interval
- t Ratio - the ratio of the Difference and Std Error columns

Comparisons with Overall Average Decision Chart

This decision chart plots a point at the mean for each group. A horizontal line is plotted at the average mean. Upper and lower decision limits are plotted. Suppose that a point corresponding to a group mean falls outside these limits. This occurrence indicates that the group mean differs from the overall mean, based on the analysis of means test at the specified significance level. The significance level is shown below the chart.

The Comparisons with Overall Average Decision Chart report menu has these options:

Show Summary Report Produces a table showing the estimate, decision limits, and the limit exceeded for each group

Display Options Gives several options for controlling the display of the chart.

Calculate Adjusted P-Values

Adds a column that contains *p*-values (Prob>|t|) to the Comparisons with Overall Average report. Note that computing exact critical values and *p*-values for unbalanced designs requires complex integration and can be computationally challenging. When calculations for such a quantile fail, the Sidak quantile is computed but *p*-values are not available.

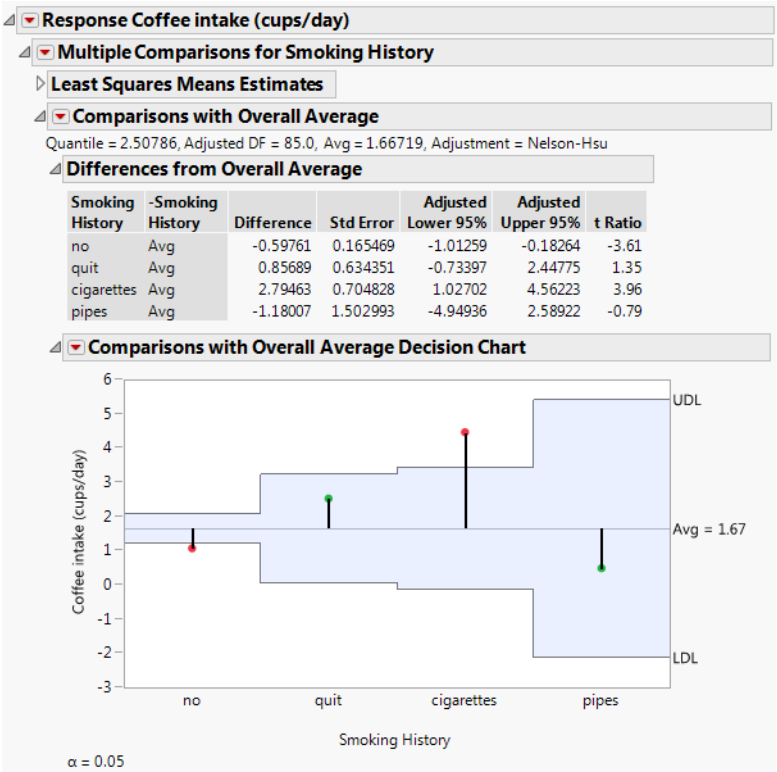
Example of Comparisons with Overall Average

Consider the Lipid Data.jmp sample data table. You are interested in whether any of the four Smoking History categories are unusual in that their mean Coffee intake (cups/day) differ from the overall average coffee intake while controlling for alcohol use and heart history. You specify a model with Coffee intake (cups/day) as the response and Smoking History, Alcohol Use, and Heart History as model effects.

1. Select **Help > Sample Data Library** and open Lipid Data.jmp.
2. Select **Analyze > Fit Model**.
3. Select Coffee intake (cups/day) and click **Y**.
4. Select Smoking History, Alcohol Use, and Heart History, and click **Add**.
5. Click **Run**.
6. Click the red triangle next to Response Coffee intake (cups/day) and select **Estimates > Multiple Comparisons**.
7. From the Choose an Effect list, select Smoking History.
8. In the Choose Initial Comparisons list, select **Comparisons with Overall Average - ANOM**.
9. Click **OK**.

The results shown in Figure 3.30 indicate that the least squares means for non-smokers and cigarette smokers differ significantly from the overall average in terms of coffee intake.

Figure 3.30 Comparisons with Overall Average for Ratings



Comparisons with Control

If you select Comparisons with Control - Dunnett's, a window opens, asking you to specify a control group. If you selected Least Squares Means Estimates, the list consists of all levels of the effect you that you selected. If you selected User-Defined Estimates, the list consists of the combinations of effect levels that you specified.

After you choose a control group and click OK, the Comparisons with Control report appears in your Fit Least Squares report. This option compares the means for the specified settings to the control group mean. It displays a table showing confidence intervals for differences from the control group and a chart showing decision limits. Dunnett's method is used to make the comparisons. Dunnett's method is a multiple comparison procedure that controls the error rate over all comparisons (Hsu 1996; Westfall et al. 2011).

When exact calculation of *p*-values and confidence intervals is not possible, Hsu's factor analytical approximation is used (Hsu 1992). Note that computing exact critical values and *p*-values for unbalanced designs requires complex integration and can be computationally intensive. When calculations for such a quantile fail, the Sidak quantile is computed.

In addition to the list of effects that do not vary for the specified estimates, at the top of the Comparisons with Control report you also find:

Quantile The critical value for Dunnett's test.

Adjusted DF The degrees of freedom used in constructing the confidence intervals.

Control The setting that defines the control group. This is a single level if you have selected a single effect; it is a combination of levels if you specified a user-defined combination of more than one effect.

Adjustment The method used to obtain the critical value:

Dunnett Provides exact critical values and *p*-values. Used whenever possible, in particular, when the estimates are uncorrelated.

Dunnett-Hsu Provides approximate critical values and *p*-values based on Hsu's factor analytical approximation (Hsu 1992). Used when exact values cannot be obtained.

Sidak Used when both Dunnett and Dunnett-Hsu fail.

For technical details, see the GLM Procedure chapter in SAS Institute Inc. (2020b).

Three options are available from the Comparisons with Control report menu:

Differences from Control

For each comparison of a group mean to the control mean, this report provides the following details:

- The levels being compared
- Difference - the estimated difference
- Std Error - the standard error of the difference
- Lower and Upper limits for the confidence interval
- t Ratio - the ratio of the Difference and Std Error columns

Comparisons with Control Decision Chart

This decision chart plots a point at the mean for each group being compared to the control group. A horizontal line shows the mean for the control group. Upper and lower decision limits are plotted. When a point falls outside these limits, it corresponds to a group whose mean differs from the control group mean based on Dunnett's test at the specified significance level. That level is shown beneath the chart.

The Comparisons with Control Decision Chart report menu has these options:

Show Summary Report Produces a table showing the estimate, decision limits, and the limit exceeded for each group

Display Options Gives several options for controlling the display of the chart.

Calculate Adjusted P-Values

Adds a column that contains *p*-values (Prob>|t|) to the Comparisons with Control report. Note that computing exact critical values and *p*-values for unbalanced designs requires complex integration and can be computationally challenging. When calculations for such a quantile fail, the Sidak quantile is computed but *p*-values are not available.

All Pairwise Comparisons

The All Pairwise Comparisons option shows either a Tukey HSD All Pairwise Comparisons or Student's *t* All Pairwise Comparisons report (Hsu 1996; Westfall et al. 2011). Tukey HSD comparisons are constructed so that the significance level applies jointly to all pairwise comparisons. In contrast, for Student's *t* comparisons, the significance level applies to each individual comparison. When making several pairwise comparisons using Student's *t* tests, the risk that one of the comparisons incorrectly signals a difference can well exceed the stated significance level.

At the top of the Tukey HSD All Pairwise Comparisons report you find:

Quantile The critical value for the test. Note that, for Tukey HSD, the quantile is $q/(\sqrt{2})$, where q is the appropriate percentage point of the Studentized range statistic.

Adjusted DF The degrees of freedom used in constructing the confidence intervals.

Adjustment Describes the method used to obtain the critical value:

Tukey Provides exact critical values and *p*-values. Used when the means are uncorrelated and have equal variances, or when the design is variance-balanced.

Tukey-Kramer Provides approximate critical values and *p*-values. Used when exact values cannot be obtained.

For technical details, see the GLM Procedure chapter in SAS Institute Inc. (2020b).

At the top of the Student's t All Pairwise Comparisons report you find the Quantile, or critical value, for the *t* test and DF, the degrees of freedom used for the *t* test.

All Pairwise Differences Report

Both Tukey HSD and Student's *t* compare all pairs of levels. For each pairwise comparison, the All Pairwise Differences report shows:

- The levels being compared
- Difference - the estimated difference between the means
- Std Error - the standard error of the difference

- t Ratio - the *t* ratio for the test of whether the difference is zero
- Prob > |t| - the *p*-value for the test
- Lower and Upper limits for a confidence interval for the difference in means

All Pairwise Comparisons Scatterplot

This plot, sometimes called a *diffogram* or a *mean-mean scatterplot*, displays the confidence intervals for all means pairwise differences. (See Figure 3.32 for an example.) Colors indicate which differences are significant.

The plot shows a reference line as an upwardly sloping line on the diagonal. This line represents points where the two means are equal. Each line segment corresponds to a confidence interval for a pairwise comparison. The coordinates of the point displayed on the line segment are the means for the corresponding groups. Hover over one of these points to show a tooltip that identifies the groups being compared and shows the estimated difference. If a line segment crosses the line on the diagonal, then the means can be equal and the comparison is not significant.

The Pairwise Comparisons Scatterplot has the following option:

Show Reference Lines Displays reference grid lines for the points on the scatterplot. This is not recommended if there are many points in the scatterplot. If there are many points, it is better to hover over the points to view the tooltip labels.

All Pairwise Differences Connecting Letters

Use this option to display a report that illustrates significant and non-significant comparisons with connecting letters. Levels not connected by the same letter are significantly different. Levels connected by the same letter are not significantly different.

Save All Pairwise Differences Connecting Letters Table

This option creates a data table whose columns contain the levels of the effect, the connecting letters, the least squares means, their standard errors, and confidence intervals. The data table contains a script called Bar Chart that produces a colored bar chart of the least squares means with their confidence intervals superimposed. The levels are arranged in decreasing order of least squares means.

Equivalence Tests

Use this option to conduct one or more equivalence tests. Equivalence tests are useful when you want to detect differences that are of *practical* interest. You must specify a threshold difference for group means for which smaller differences are considered practically equivalent. In other words, if two group means differ by this amount or less, you are willing to consider them equivalent.

Once you have specified this value, the Equivalence Tests report appears. The bounds that you have specified are given at the top of the report. The report consists of a table giving the equivalence tests and a scatterplot that displays them. The equivalence tests and confidence intervals are based on Student's *t* critical values.

Note: Equivalence tests are available only for the Student's *t* method.

Equivalence TOST Tests

The Two One-Sided Tests (TOST) method is used to test for a practical difference between the means (Schuirmann 1987). Two one-sided pooled-variance *t* tests are constructed for the null hypotheses that the true difference exceeds the threshold values. If both tests reject, the difference in the means does not statistically exceed either threshold value. Therefore, the groups are considered practically equivalent. If only one or neither test rejects, then the groups might not be practically equivalent.

For each comparison, the Equivalence TOST Tests report gives the following information:

- Difference - the estimated difference in the means
- Lower Bound t Ratio, Upper Bound t Ratio - the lower and upper bound *t* ratios for the two one-sided pooled-variance significance tests
- Lower Bound p-Value, Upper Bound p-Value - *p*-values corresponding to the lower and upper bound *t* ratios
- Maximum p-Value - the maximum of the lower and upper bound *p*-values
- Lower and Upper limits for a 1–2α confidence interval for the difference in the means.

Note: Equivalence TOST tests are available only for the Student's *t* method.

Equivalence Tests Scatterplot

Using colors, this scatterplot indicates which means are practically equivalent and which are not practically equivalent as determined by the equivalence test. This plot is sometimes called a *diffogram* or a mean-mean scatterplot.

The plot shows a solid reference line on the diagonal as well as a shaded reference band. The width of the band is twice the practical difference. The coordinates of the point on the line segment are the means for the corresponding groups. There is an implied third axis on the diagonal where each line segment corresponds to a 1–2α confidence interval for a pairwise comparison. Hover over one of these points to show a tooltip that indicates the groups being compared and the estimated difference. When a line segment is entirely contained within the diagonal band, it follows that the means are practically equivalent.

Note: Equivalence tests scatterplots are available only for the Student's *t* method.

The Equivalence Tests Scatterplot has the following option:

Show Reference Lines Displays reference lines for the points on the scatterplot. This is not recommended if there are many points in the scatterplot. If there are many points, it is better to hover over the points to view the tooltip labels.

Remove

This option removes the Equivalence Tests report from the Student's t All Pairwise Comparisons report.

Example of Tukey HSD All Pairwise Comparisons

Consider the Lipid Data.jmp sample data table. You are interested in Cholesterol differences for gender and non-smokers versus former smokers (Smoking History equal to no and quit, respectively) across two ages (25 and 35) and average height.

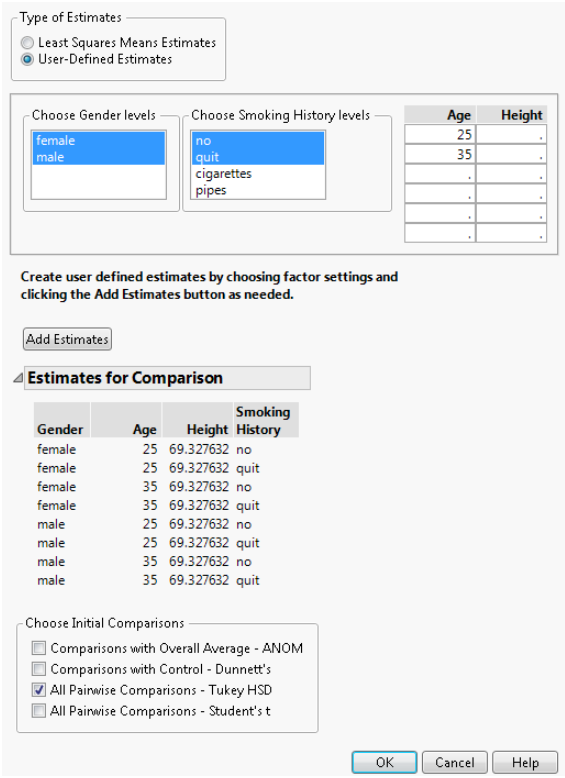
1. Select **Help > Sample Data Library** and open Lipid Data.jmp.
2. Select **Analyze > Fit Model**.
3. Select Cholesterol and click **Y**.
4. Select Gender, Age, Height, and Smoking History, and click **Add**.
5. Click **Run**.
6. Click the red triangle next to Response Cholesterol and select **Estimates > Multiple Comparisons**.
7. From the Type of Estimates list, click **User-Defined Estimates**.
8. From the Choose Gender levels list, select female (it should already be selected by default) and male.
9. From the Choose Smoking History levels list, select no and quit.
10. In the Age list, enter the ages 25 and 35 in the first two rows.

    Do not enter any values in the list entitled Height. Because no values for Height are specified, the mean value of the Height column is used in the multiple comparisons report.
11. Click **Add Estimates**.

    Note that all possible combinations of the levels that you specified appear in the Estimates for Comparison report.
12. In the Choose Initial Comparisons list, select **All Pairwise Comparisons - Tukey HSD**.

    Check that your window is completed as shown in Figure 3.31.

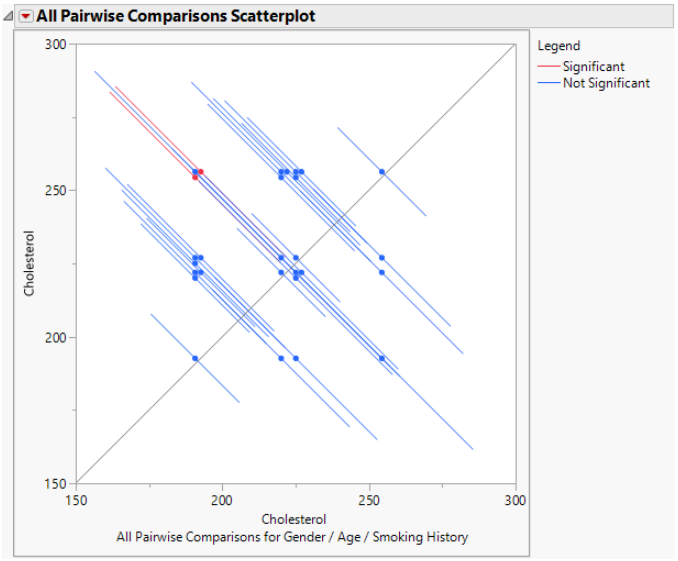
Figure 3.31 Populated Used-Defined Estimates Window



13. Click **OK**.

    The All Pairwise Differences report indicates that two of the 28 pairwise comparisons are significant. The All Pairwise Comparisons Scatterplot, shown in Figure 3.32, shows the confidence intervals for these comparisons in red. You can hover over any of the points to determine which pairwise comparison the point represents. The tooltips also contain the difference between the two levels in the comparison. The two red points in Figure 3.32 represent the points comparing 35-year-old former smokers to 25-year-old non-smokers, for both females and males.

Figure 3.32 All Pairwise Comparisons Scatterplot for User-Defined Comparisons



Compare Slopes

The Compare Slopes option appears when there is one nominal term, one continuous term, and their interaction effect for the fixed effects. This option produces a report that enables you to compare the slopes in an analysis of covariance (ANCOVA) model. The report compares the slopes of each level of the interaction effect to the overall slope. The comparison uses analysis of means (ANOM) with the overall average. For more information about the analysis of means (ANOM) report, see "Comparisons with Overall Average".

The overall average slope is a weighted average of the slopes, where the weights are inversely proportional to the variances of the slope estimates. These variances are the squared values of the Std Error column in the Differences from Overall Average Slope table.

Joint Factor Tests

The Joint Factor Test option appears when interaction effects are present. For each main effect in the model, JMP produces a joint test of whether all the coefficients for terms involving that main effect are zero. This test is conditional on all other effects being in the model. Specifically, the joint test is a general linear hypothesis test of a restricted model. In that model, all parameters that correspond to the specified effect and the interactions that contain it are set to zero.

Example of a Joint Factor Tests Report

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Model**.
3. Select weight and click **Y**.
4. Verify that 2 appears in the **Degree** box.
5. Select age, sex, and height and click **Macros > Factorial to Degree**.
6. Click **Run**.
7. Click the Response weight red triangle and select **Estimates > Joint Factor Tests**.

Figure 3.33 Joint Factor Tests Report

Joint Factor Tests

Term	DF	Sum of Squares	F Ratio	Prob > F
age	15	6116.2127	2.8488	0.0139*
sex	7	2113.1080	2.1091	0.0879
height	7	9217.8156	9.2002	<.0001*

Note that the test for age has 15 degrees of freedom. This test involves the five parameters for age, the five parameters for age*sex, and the five parameters for height*age. The null hypothesis for this test is that all 15 parameters are zero.

Inverse Prediction

Inverse prediction occurs when you use a statistical model to infer the value of an explanatory variable, given a value of the response variable. Inverse prediction is sometimes referred to as *calibration*.

By selecting Inverse Prediction on the Estimates menu, you can estimate values of an independent variable, *X*, that correspond to specified values of the response (Figure 3.37). In addition, you can specify values for other explanatory variables in the model (Figure 3.37). The inverse prediction computation provides confidence limits for values of X that correspond to the specified response value. You can specify the response value to be the mean response or simply an individual response. For an example, see "Example of Inverse Prediction".

Analyzing Multiple Explanatory Variables

When the model includes multiple explanatory variables, you can predict the value of X for the specified values of the other variables. You might want to predict the amount of running time that results in an oxygen uptake of 50 when one's resting pulse rate is 60. You might want separate inverse predictions for both males and females. Specify these requirements using the inverse prediction option.

The inverse prediction window shows the list of explanatory variables to the left. (See Figure 3.37 for an example.) Each continuous variable is initially set to its mean. Each nominal or ordinal variable is set to its lowest level (in terms of value ordering). You must remove the value for the variable that you want to predict, setting it to missing. Also, you must specify the values of the other variables for which you want your inverse prediction to hold (if these differ from the default settings). In the list to the right in the window, you can supply one or more response values of interest. For an example, see "Example of Predicting a Single X Value with Multiple Model Effects".

Note: The confidence limits for inverse prediction can sometimes result in a one-sided or even an infinite interval. For technical details, see "Inverse Prediction with Confidence Limits".

Example of Inverse Prediction

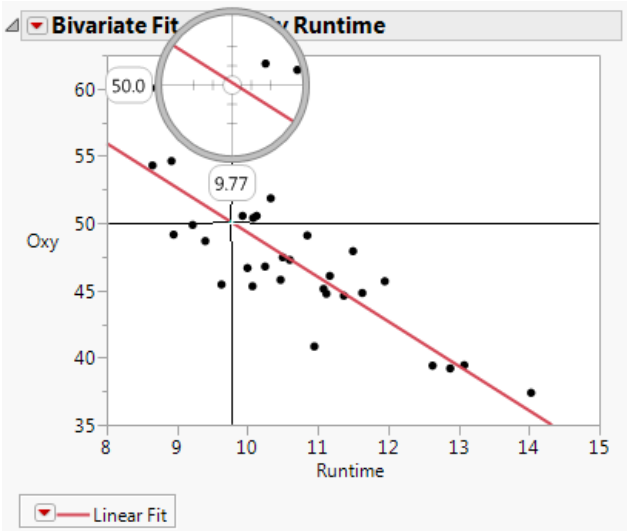
In this example, you fit a regression model that predicts oxygen uptake from Runtime. Then you estimate the Runtime values that result in specified oxygen uptake values. There is only a single *X*, Runtime, so you start by using the Fit Y by X platform to obtain a visual approximation of the inverse prediction values.

1. Select **Help > Sample Data Library** and open Fitness.jmp.
2. Select **Analyze > Fit Y by X**.
3. Select Oxy and click **Y, Response**.
4. Select Runtime and click **X, Factor**.
5. Click **OK**.
6. Click the Bivariate Fit of Oxy By Runtime red triangle and select **Fit Line**.

   Use the crosshairs tool as described below to approximate the Runtime value that results in a mean Oxy value of 50.
7. Select **Tools > Crosshairs**.
8. Click on the prediction line and then drag the crosshairs tool to find the inverse prediction for Oxy = 50.

Figure 3.34 shows that a Runtime of about 9.77 gives an inverse prediction of about 50 for Oxy.

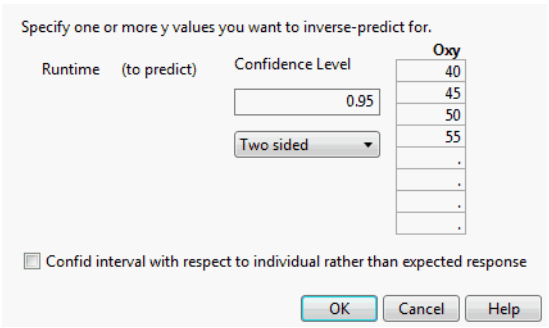
Figure 3.34 Bivariate Fit for Fitness.jmp



To obtain an exact prediction for Runtime, along with a confidence interval, use the Fit Model launch window:

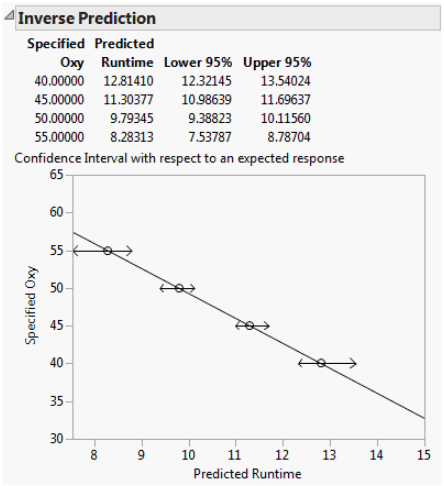
1. From the Fitness.jmp sample data table, select **Analyze > Fit Model**.
2. Select Oxy and click **Y**.
3. Select Runtime and then click **Add**.
4. Click **Run**.
5. Click the Response Oxy red triangle and select **Estimates > Inverse Prediction**.
6. Enter four values for Oxy as shown in Figure 3.35.
7. Click **OK**.

Figure 3.35 Completed Inverse Prediction Specification Window



The Inverse Prediction report gives predicted Runtime values that correspond to each specified Oxy value. The report also shows upper and lower 95% confidence limits for these Runtime values, relative to obtaining the mean response.

Figure 3.36 Inverse Prediction Report



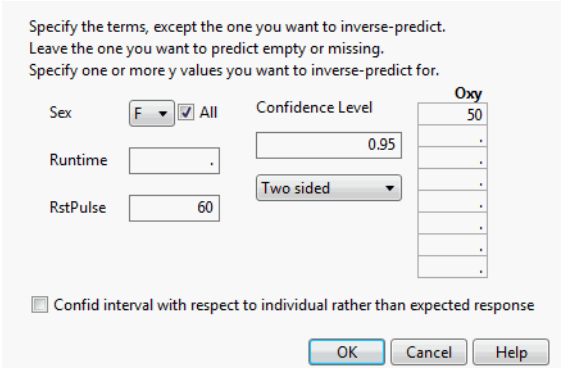
The exact predicted Runtime resulting in an Oxy value of 50 is 9.7935. This value is close to the approximate Runtime value of 9.77 found in the Bivariate Fit report shown in Figure 3.34. The Inverse Prediction report also gives a plot showing the linear relationship between Oxy and Runtime and the confidence intervals.

Example of Predicting a Single X Value with Multiple Model Effects

This example predicts the Runtime that results in oxygen uptake of 50 when RstPulse is 60. The Runtime is predicted for both males and females.

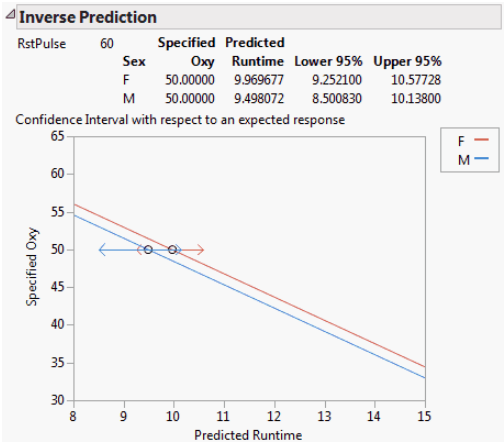
1. From the Fitness.jmp sample data table, select **Analyze > Fit Model**.
2. Select Oxy and click **Y**.
3. Select Sex, Runtime, and RstPulse and then select **Add**.
4. Click **Run**.
5. Click the Response Oxy red triangle and select **Estimates > Inverse Prediction**.
6. Delete the value for Runtime, because you want to predict that value.
7. Select the **All** box next to Sex to estimate Runtime for all levels of Sex.
8. Replace the mean for RstPulse with 60.
9. Enter the value 50 for Oxy as shown in Figure 3.37.
10. Click **OK**.

Figure 3.37 Inverse Prediction Specification for a Multiple Regression Model



The report, shown in Figure 3.38, gives the predicted values of Runtime for both females and males. The report also includes 95% confidence intervals for Runtime values that give a mean response of 50.

Figure 3.38 Inverse Prediction Report for a Multiple Regression Model



The plot shows the linear fits for females and males, given that RstPulse is 60. The two confidence intervals are shown in red and blue, respectively.

Cox Mixtures

Note: This option is available only for mixture models.

Standard least squares fits mixture models using the parameterization suggested in Scheffé (1958). The parameters for this model cannot easily be used to judge the effects of the mixture components. The Cox Mixture model is a reparameterized and constrained version of the Scheffé model. Using its parameter estimates, you can derive factor effects and the response surface shape relative to a reference point in the design space. See Cornell (1990) for a complete discussion.

The Cox Mixture option opens a window where you enter the reference mixture. If you enter components for the reference mixture that do not sum to one, then the components are proportionately scaled so that they do sum to one. The rescaled mixture is shown in the report as the Reference Mixture. The component effects also appear in the report. A Cox component effect is the difference in the predicted response as the factor goes from its minimum to maximum values along the Cox effect direction.

Example of Cox Mixtures

1. Select **Help > Sample Data Library** and open Five Factor Mixture.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y1 and click **Y**.
4. Select X1 through X5.
5. Select **Macros > Mixture Response Surface**.
6. Click **Run**.
7. Click the Response Y1 red triangle and select **Estimates > Cox Mixtures**.
8. In the Reference Mixture specification window, enter the following values:
   - Next to X1, type 0.15.
   - Next to X2, type 0.25.
   - Next to X3, type 0.4.
   - Next to X4, type 0.1.
   - Next to X5, type 0.1.
9. Click **OK**.

Figure 3.39 Cox Mixtures

Cox Reference Mixture Model

Cox Parameter Estimates

Parameter	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	2.071	1.2541	1.651	0.1035
X1	3.120	7.7538	0.402	0.6887
X2	-5.828	7.0104	-0.831	0.4089
X3	-0.696	3.5822	-0.194	0.8465
X4	3.935	12.6619	0.311	0.7570
X5	8.740	12.5405	0.697	0.4883
X1^2	-15.897	16.3627	-0.972	0.3349
X2^2	17.877	13.5932	1.315	0.1931
X3^2	5.234	8.2498	0.634	0.5280
X4^2	-91.239	159.9347	-0.570	0.5703
X5^2	-145.693	158.3991	-0.920	0.3611
X1*X2	-2.938	24.9012	-0.118	0.9065
X1*X3	-7.617	19.1697	-0.397	0.6924
X2*X3	-16.250	18.7489	-0.867	0.3893
X1*X4	53.690	50.5711	1.062	0.2923
X2*X4	-17.702	49.9082	-0.355	0.7240
X3*X4	-7.808	44.0229	-0.177	0.8598
X1*X5	31.813	50.6186	0.628	0.5319
X2*X5	-2.279	49.3761	-0.046	0.9633
X3*X5	17.983	43.5145	0.413	0.6808
X4*X5	177.431	97.3862	1.822	0.0731

Cox Reference Mixture

Factor	Reference Mixture
X1	0.1500000
X2	0.2500000
X3	0.4000000
X4	0.1000000
X5	0.1000000

Component Effects

Component	Cox Effect	Component Range
X1	0.80746	0.2200000
X2	-2.33103	0.3000000
X3	-0.48747	0.4200000
X4	0.65585	0.1500000
X5	1.45663	0.1500000

The report shows the parameter estimates for the Cox mixture model, along with standard errors and hypothesis tests. The reference mixture appears on the right. The component effects appear below, along with the component ranges.

Parameter Power

The *power* of a statistical test is the probability that the test will be significant, if a difference actually exists. The power of the test indicates how likely your study is to declare a true effect to be significant. The Parameter Power option addresses *retrospective* power analysis.

Note: To ensure that your study includes sufficiently many observations to detect the required differences, use information about power when you *design* your experiment. This type of analysis is called *prospective* power analysis. Consider using the DOE platform to design your study. Both DOE > Sample Size and Power and DOE > Evaluate Design are useful for prospective power analysis. For an example of a prospective power analysis using standard least squares, see "Prospective Power Analysis".

The power of a test to detect a difference is affected by the following factors:

- the sample size
- the unknown residual error variance
- the significance level of the test
- the size of the effect to be detected

Suppose that you have already conducted your study, analyzed your data, and found that an effect of interest is not significant. You might be interested in the size of the difference that you might have been likely to detect or the power of the test that you conducted. Or you might want to know the number of observations that you would have needed to detect a difference of a given size with high probability.

The Parameter Power option inserts three columns of values relating to retrospective power analysis in the Parameter Estimates report. The least significant value (LSV0.05), the least significant number (LSN0.05), and a power calculation (AdjPower0.05) are provided.

The Parameter Power calculations apply to a new sample that has the same variability profile as the observed sample.

Caution: The results provided by the LSV0.05, LSN, and AdjPower0.05 should not be used in prospective power analysis. They do not reflect the uncertainty inherent in a future study.

- LSV0.05 is the *least significant value*. This number is the smallest absolute value of the estimate that would make this test significant at significance level 0.05. To be more specific, suppose that the number of observations, the mean square error and that the sum of squares and cross-products matrix for the design remain unchanged. Then, if the absolute value of the estimate had been less than LSV0.05, the Prob>|t| value would have exceeded 0.05. See "The Least Significant Value (LSV)".
- LSN is the *least significant number*. This number is the number of observations that would make this test significant at significance level 0.05. Specifically, suppose that the estimate of the parameter, the mean square error, and the sum of squares and cross-products matrix for the design remain unchanged. Then, if the number of observations had been less than the LSN, the Prob>|t| value would have exceeded 0.05. See "The Least Significant Number (LSN)".
- AdjPower0.05 is the *adjusted power* value. This number is an estimate of the probability that this test will be significant. Sample values from the current study are substituted for the parameter values typically used in a power calculation. The *adjusted* power calculation adjusts for bias that results from direct substitution of sample estimates into the formula for the non-centrality parameter (Wright and O'Brien 1988). See "The Adjusted Power and Confidence Intervals".

The LSV, LSN, and adjusted power are useful in assessing a test's sensitivity. These retrospective calculations also provide an enlightening instructional tool. However, you must be cautious in interpreting these values (Hoenig and Heisey 2001).

For more information about LSV, LSN, and adjusted power, see "Power Analysis". For an example of a retrospective analysis, see "Example of Retrospective Power Analysis".

Correlation of Estimates

The Correlation of Estimates command on the Estimates menu computes the correlation matrix for the parameter estimates. These correlations indicate whether collinearity is present.

For insight on the construction of this matrix, consider the typical least squares regression formulation. Here, the response (Y) is a linear function of predictors (x's) plus error (ε):

$$Y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \varepsilon$$

Each row of the data table contains a response value and values for the p predictors. For each observation, the predictor values are considered fixed. However, the response value is considered to be a realization of a random variable.

Considering the values of the predictors fixed, for any set of Y values, the coefficients, $\beta_0, \beta_1, \ldots, \beta_p$, can be estimated. In general, different sets of Y values lead to different estimates of the coefficients. The Correlation of Estimates option calculates the theoretical correlation of these parameter estimates. For technical details, see "Details of Custom Test Example".

The correlations of the parameter estimates depend solely on the predictor values and a term representing the intercept. The correlation between two parameter estimates is not affected by the values of the response.

A high positive correlation between two estimates suggests that a collinear relationship might exist between the two corresponding predictors. Note, though, that you need to interpret these correlations with caution (Belsley et al. 1980, p. 185, 92–94). Also, a rescaling of a predictor that shifts its mean changes the correlation of its parameter estimate with the intercept's value.

Example of Correlation of Estimates

1. Select **Help > Sample Data Library** and open Socioeconomic.jmp.
2. Select **Analyze > Fit Model**.
3. Select Median House Value and click **Y**.
4. Select Total Population, Median School Years, Total Employment, and Professional Services and click **Add**.
5. In the Emphasis list, select **Minimal Report**.

6. Click **Run**.
7. Click the red triangle next to Response Median House Value and select **Estimates > Correlation of Estimates**.

Figure 3.40 Correlation of Estimates Report

Correlation of Estimates

Corr

	Intercept	Total Population	Median School Years	Total Employment	Professional Services
Intercept	1.0000	-0.5743	-0.9818	0.4719	0.6903
Total Population	-0.5743	1.0000	0.5871	-0.9746	-0.2398
Median School Years	-0.9818	0.5871	1.0000	-0.5132	-0.7085
Total Employment	0.4719	-0.9746	-0.5132	1.0000	0.1094
Professional Services	0.6903	-0.2398	-0.7085	0.1094	1.0000

The report shows high negative correlations between the parameter estimates for the Intercept and Median School Years (–0.9818). High negative correlations also exist between Total Population and Total Employment (–0.9746).

Coding for Nominal Effects

When you enter a column with a nominal modeling type into your model, JMP represents it internally as a set of continuous indicator variables. Each variable assumes only the values –1, 0, and 1. (Note that this coding is one of many ways to use indicator variables to code nominal variables.) If your nominal column has *n* levels, then *n-1* of these indicator variables are needed to represent it. (The need for *n-1* indicator variables relates directly to the fact that the main effect associated with the nominal column has *n-1* degrees of freedom.) Full details are covered in "Nominal Factors".

Tip: You can view the coding by selecting Save Columns > Save Coding Table from the red triangle menu for the main report. See "Save Coding Table".

Suppose that you have a nominal column with four levels. Take, as an example, the treatment column in the Cholesterol.jmp sample data table. The treatment column has four levels: A, B, Control, and Placebo. Each of the first three levels is represented by an indicator variable. These indicator variables are named treatment[A], treatment[B], and treatment[Control].

The indicator variable for a given level assigns the values 1 to that level, –1 to the last level, and 0 to the remaining levels. Table 3.1 shows the definitions of the treatment[A], treatment[B], and treatment[Control] indicator variables for this example. For example, consider the indicator variable treatment[A]. As shown in Table 3.1, this variable assigns the following values:

- The value 1 is assigned to rows that have treatment = A
- The value 0 is assigned to rows that have treatment = B or Control

- The value –1 is assigned to rows that have treatment = Placebo

Table 3.1 Illustration of Indicator Variables for treatment in Cholesterol.jmp

Treatment Assigned to Row	treatment[A]	treatment[B]	treatment[Control]
A	1	0	0
B	0	1	0
Control	0	0	1
Placebo	-1	-1	-1

The order of the levels is determined either by the Value Order column property, if you have assigned one, or by the default ordering assigned by JMP. The default ordering is typically the numeric sorting order for numbers and the alphanumeric sorting order for character data. However, certain categorical values, such as the names of months, are sorted appropriately by default. For more information about value ordering, see *Using JMP*.

These variables are used to parametrize the model. They do not typically appear in the data table, but the estimated coefficients for these variables are given in the Parameter Estimates and other reports. Although many other codings are possible, this coding has proven to be practical and interpretable.

For information about the coding of ordinal effects, see "Ordinal Factors".

Effect Screening

A screening design often provides no degrees of freedom for error. So classical tests for effects are not available. In such cases, Effect Screening is particularly useful.

For these designs, most inferences about effect sizes assume that the estimates for non-intercept parameters are uncorrelated and have equal variances. These assumptions hold for the models associated with many classical experimental designs. However, there are situations where these assumptions do not hold. In both of these situations, the Effect Screening platform guides you in determining which effects are significant.

The Effect Screening platform uses the principle of *effect sparsity* (Box and Meyer 1986). This principle asserts that relatively few of the effects that you study in a screening design are active. Most are inactive, meaning that their true effects are zero and that their estimates are random error.

The following Effect Screening options are available:

Scaled Estimates Gives parameter estimates corresponding to factors that are scaled to have a mean of zero and a range of two. See "Scaled Estimates and the Coding of Continuous Terms".

Normal Plot Identifies parameter estimates that deviate from normality, helping you determine which effects are active. See "Normal Plot Report".

Bayes Plot Computes posterior probabilities for all model terms using a Bayesian approach. See "Bayes Plot Report".

Pareto Plot Plots the absolute values of the orthogonalized and standardized parameter estimates, relating these to the sum of their absolute values. See "Pareto Plot Report".

Scaled Estimates and the Coding of Continuous Terms

A parameter estimate is highly dependent on the scale of its corresponding factor. When you convert a factor from grams to kilograms, its parameter estimate changes by a multiple of a thousand. When you apply the same change to a squared (quadratic) term, its parameter estimate changes by a multiple of a million.

To better understand and compare effect sizes, you should examine parameter estimates in a scale-invariant fashion. It makes sense to use a scale that relates the size of a parameter estimate to the size of the effect of its corresponding term on the response. There are many approaches to doing this.

The Effect Screening > Scaled Estimates option in the report's red triangle menu gives coefficients corresponding to scaled factors. The factors are scaled to have a mean of zero and a range of two. Figure 3.41 shows a report for Drug.jmp.

If the sample values for the factor are such that the maximum and minimum values are equidistant from the sample mean, then the scaled factor ranges from –1 to 1. This scaling corresponds to the traditional coding used in the design of experiments. In the case of regression with a single factor, the scaled parameter estimate is half of the predicted response change as the factor travels its whole range.

Scaled estimates are important in assessing the impact of model terms when the data involve uncoded values. For orthogonal designs, the scaled estimates are identical to the estimates for the uncoded data.

Note: The Coding column property scales factor values linearly so that their coded values range from –1 to 1. Parameter estimates are given in terms of these coded values, so that scaled estimates are not required in this situation. (Unlike the transformation used to obtain scaled estimates, the coded values might not have mean zero.)

Example of Scaled Estimates

1. Select **Help > Sample Data Library** and open Drug.jmp.
2. Select **Analyze > Fit Model**.
3. Select y and click **Y**.
4. Select Drug and x and add these to the **Construct Model Effects** list.
5. Change the Emphasis to **Minimal Report**.
6. Click **Run**.
7. Click the Response y red triangle and select **Effect Screening > Scaled Estimates**.

The report indicates that the continuous factor, x, is centered by its mean and scaled by its half-range.

Figure 3.41 Scaled Estimates Report

Scaled Estimates

Nominal factors expanded to all levels
Continuous factors centered by mean, scaled by range/2

Term	Scaled Estimate	Std Error	t Ratio	Prob>|t|
Intercept	7.9	0.731352	10.80	<.0001*
Drug[a]	-1.185037	1.060822	-1.12	0.2742
Drug[d]	-1.076065	1.041298	-1.03	0.3109
Drug[f]	2.2611017	1.093974	2.07	0.0488*
x	8.8846543	1.480478	6.00	<.0001*

The model fits three parallel lines, one for each Drug group. The x values range from 3 to 21. The Scaled Estimate for x, 8.8846543, is half the difference between the predicted value for x = 21 and the predicted value for x = 3 for any one of the Drug groups. You can verify this fact by selecting **Save Columns > Prediction Formula** from the report's red triangle menu. Then add rows to obtain predicted values for each of the Drug groups at x = 21 and x = 3.

So, over the range of x values in this particular data set, the impact of x is to vary the response over a range of about 17.8 units. Note that the parameter estimate for x based on the raw data is 0.9871838; it does not permit direct interpretation in terms of the response.

Plot Options

The Normal, Bayes, and Pareto Plot options provide reports that appear as part of the Effect Screening report. These reports can be constructed so that they correct for unequal variances and correlations among the estimates.

Note: The Normal, Bayes, and Pareto Plot options require that your model involves at least four parameters. One of these parameters can be the intercept.

Transformations

When you select any of the plot options, the following information appears directly beneath the Effect Screening report title:

- If the estimates have equal variances and are uncorrelated, these two notes appear:
  - The parameter estimates have equal variances.
  - The parameter estimates are not correlated.
- If the estimates have unequal variances or are correlated, then an option list replaces the relevant note. The list items selected by default show that JMP has transformed the estimates. If you want to undo either or both transformations, select the appropriate list items.

Lenth PSE Values

A Lenth PSE (*pseudo standard error*) table appears directly beneath the notes or option lists. For a description of the PSE, see "Lenth's PSE". The statistics that appear in the Lenth table depend on the variances and correlation of the parameter estimates.

When the parameter estimates have equal variances and are uncorrelated, the Lenth table provides the following statistic:

Lenth PSE The Lenth pseudo standard error for the estimates.

When the parameter estimates have unequal variances or are correlated or both, the Lenth table provides the following statistics:

t-Test Scale Lenth PSE The Lenth pseudo standard error computed for the transformed parameter estimates divided by their standard errors in the transformed scale.

Coded Scale Lenth PSE The Lenth pseudo standard error for the transformed parameter estimates.

Parameter Estimate Population Report

The Parameter Estimate Population report gives tests for the parameter estimates. The tests are based on transformations as specified in the option lists.

- The option **Using estimates standardized to have equal variances** applies a normalizing transformation to standardize the variances. This option is selected by default when the variances are unequal.
- The option **Using estimates orthogonalized to be uncorrelated** applies a transformation to remove correlation. This option is selected by default when the estimates are correlated. The transformation that is applied is identical to the transformation that is used to

calculate sequential sums of squares. The estimates measure the additional contribution of the variable after all previous variables have been entered into the model.

- If the notes indicate that the estimates have equal variances and are not correlated, no transformation is applied.

The columns that appear in the table depend on the selections initially described in the notes or option lists. The report highlights any row corresponding to an estimate with a *p*-value of 0.20 or less. All versions of the report give Term, Estimate, and either *t*-Ratio and Prob>|t| or Pseudo *t*-Ratio and Pseudo *p*-Value.

Term Gives the model term whose parameter estimate is of interest.

Estimate Gives the estimate for the parameter. Estimate sizes can be compared only when the model effects have identical scaling.

t-Ratio Appears if there are degrees of freedom for error. This value is the parameter estimate divided by its standard error.

Prob>|t| Gives the *p*-value for the test. If a transformation is applied, this option gives the *p*-value for a test using the transformed data.

Pseudo t-Ratio Appears when there are no degrees of freedom for error. If the relative standard errors of the parameters are equal, Pseudo t-Ratio is the parameter estimate divided by Lenth's PSE. If the relative standard errors vary, it is calculated as shown in "Pseudo t-Ratios".

Pseudo p-Value Appears when there are no degrees of freedom for error. The *p*-value is derived using a *t* distribution. The degrees of freedom are *m*/3, rounded down to the nearest integer, where *m* is the number of parameters.

If **Using estimates standardized to have equal variances** is selected and the note indicating that the parameter estimates are not correlated appears, the report shows a column called Standardized Estimate. This column provides estimates of the parameters resulting from the transformation used to transform the estimates to have equal variances.

If both **Using estimates standardized to have equal variances** and **Using estimates orthogonalized to be uncorrelated** are selected, the report gives a column called Orthog Coded. The following information is provided:

Orthog Coded Gives the estimate of the parameter resulting from the transformation that is used to orthogonalize the estimates.

Orthog t-Ratio Appears if there are degrees of freedom for error. Gives the *t* ratio for the transformed estimates.

Pseudo Orthog t-Ratio Appears if there are no degrees of freedom for error. It is a *t* ratio computed by dividing the orthogonalized estimate, Orthog Coded, by Coded Scale Lenth PSE.

Effect Screening Report

Figure 3.42 shows the Effect Screening report that you create by running the Fit Model script in the Bicycle.jmp sample data table. Note that you would select **Effect Screening > Normal Plot** in order to obtain this form of the report. The notes directly beneath the report title indicate that no transformation is required. Consequently, the Lenth PSE is displayed. Because there are no degrees of freedom for error, no estimate of residual error can be constructed. For this reason, Lenth's PSE is used as an estimate of residual error to obtain pseudo *t* ratios. Pseudo *p*-values are given for these *t* ratios. Rows for non-intercept terms corresponding to the three estimates with *p*-values of 0.20 or less are highlighted.

Figure 3.42 Effect Screening Report for Equal Variance and Uncorrelated Estimates

Effect Screening

The parameter estimates have equal variances.
The parameter estimates are not correlated.

Lenth PSE
0.75

Parameter Estimate Population

Term	Estimate	Pseudo t-Ratio	Pseudo p-Value
Intercept	66.5000	88.6667	<.0001*
HBars[down]	0.5000	0.6667	0.5649
Dynamo[off]	-6.0000	-8.0000	0.0096*
Seat[down]	1.7500	2.3333	0.1266
Tires[hard]	-1.2500	-1.6667	0.2195
Gear[low]	-11.2500	-15.0000	0.0023*
Raincoat[off]	0.2500	0.3333	0.7665
Brkfast[no]	0.5000	0.6667	0.5649

Orthog t Test used Pseudo Standard Error

Normal Plot

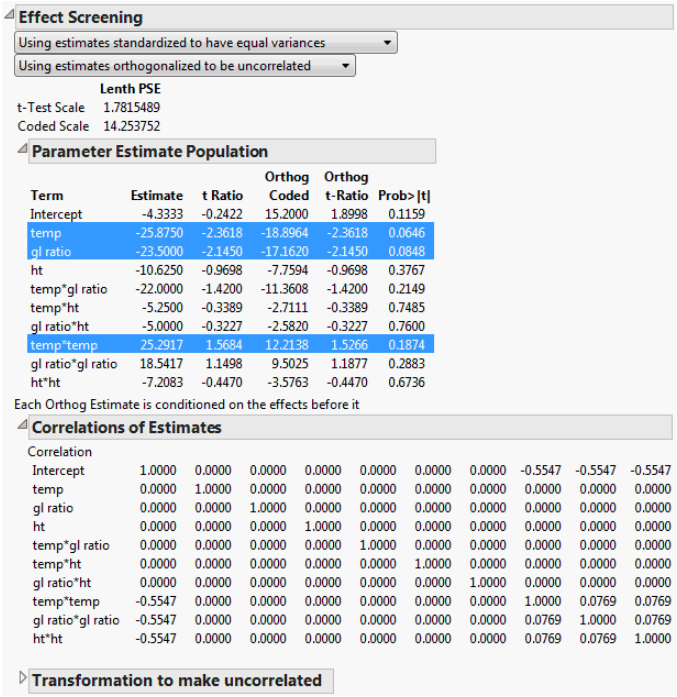
Effect Screening Report for Unequal Variances and Correlated Estimates

In the Odor.jmp sample data table, run the Model script and click **Run**. To create the report shown in Figure 3.43, click the Response Y red triangle and select **Effect Screening > Normal Plot**. You can also create the report by clicking the Response Y red triangle and selecting the Bayes Plot or Pareto Plot options.

The notes directly beneath the report title indicate that transformations were required both to standardize and orthogonalize the estimates. The correlation matrix is shown in the Correlation of Estimates report.

The report shows the *t*-Test Scale and Coded Scale Lenth PSEs. But, because there are degrees of freedom for error, the tests in the Parameter Estimate Population report do not use the Lenth PSEs. Rows for non-intercept terms corresponding to the three estimates with *p*-values of 0.20 or less are highlighted. A note at the bottom of the Parameter Estimate Population report indicates that orthogonalized estimates depend on their order of entry into the model.

Figure 3.43 Effect Screening Report for Unequal Variances and Correlated Estimates



Effect Screening

Using estimates standardized to have equal variances

Using estimates orthogonalized to be uncorrelated

	Lenth PSE
t-Test Scale	1.7815489
Coded Scale	14.253752

Parameter Estimate Population

Term	Estimate	t Ratio	Orthog Coded	Orthog t-Ratio	Prob>|t|
Intercept	-4.3333	-0.2422	15.2000	1.8998	0.1159
temp	-25.8750	-2.3618	-18.8964	-2.3618	0.0646
gl ratio	-23.5000	-2.1450	-17.1620	-2.1450	0.0848
ht	-10.6250	-0.9698	-7.7594	-0.9698	0.3767
temp*gl ratio	-22.0000	-1.4200	-11.3608	-1.4200	0.2149
temp*ht	-5.2500	-0.3389	-2.7111	-0.3389	0.7485
gl ratio*ht	-5.0000	-0.3227	-2.5820	-0.3227	0.7600
temp*temp	25.2917	1.5684	12.2138	1.5266	0.1874
gl ratio*gl ratio	18.5417	1.1498	9.5025	1.1877	0.2883
ht*ht	-7.2083	-0.4470	-3.5763	-0.4470	0.6736

Each Orthog Estimate is conditioned on the effects before it

Correlations of Estimates

Correlation										
Intercept	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.5547	-0.5547	-0.5547
temp	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
gl ratio	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
ht	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
temp*gl ratio	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000
temp*ht	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000
gl ratio*ht	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000
temp*temp	-0.5547	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0769	0.0769
gl ratio*gl ratio	-0.5547	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0769	1.0000	0.0769
ht*ht	-0.5547	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0769	0.0769	1.0000

Transformation to make uncorrelated

Correlations of Estimates Report

The Correlations of Estimates report appears only if the estimates are correlated (Figure 3.43). The report provides the correlation matrix for the parameter estimates. This matrix is similar to the one that you obtain by selecting the Estimates > Correlation of Estimates red triangle option. However, to provide a more compact representation, the report does not show column headings. See “Correlation of Estimates”.

“Transformation to make uncorrelated” Report

The “Transformation to make uncorrelated” report appears only if the estimates are correlated. The report gives the matrix used to transform the design matrix to produce uncorrelated parameter estimates. The transformed, or *orthogonally coded*, estimates are obtained by pre-multiplying the original estimates with this matrix and multiplying the result by a scale factor. The scale factor is a function of the root mean square error (RMSE) and the number of rows. It is defined as follows:

RMSE / sqrt(NRows)

The transformation matrix can be obtained using the Cholesky decomposition. Express X′X as LL′, where L is the lower triangular matrix in the Cholesky decomposition. Then the transformation matrix is given by L′.

This transformation orthonormalizes each regressor with respect to the regressors that precede it in the ordering of model terms. The transformation produces a diagonal covariance matrix with equal diagonal elements. The coded estimates are a result of this iterative process.

Note: The orthogonally coded estimates depend on the order of terms in the model. Each effect's contribution is estimated only after it is made orthogonal to previously entered effects. Consider entering main effects first, followed by two-way interactions, then three-way interactions, and so on.

Normal Plot Report

Below the Normal Plot report title, select either a normal plot or a half-normal plot (Daniel 1959). Both plots are predicated on the principle of effect sparsity, namely, the idea that relatively few effects are active. Those effects that are inactive represent random noise. Their estimates can be assumed to have a normal distribution with mean 0 and variance σ^2, where σ^2 represents the residual error variance. It follows that, on a normal probability plot, estimates representing inactive effects fall close to a line with slope σ.

Normal Plot

If no transformation is required, the vertical coordinate of the normal plot represents the value of the estimate and the horizontal coordinate represents its normal quantile. Points that represent inactive effects should follow a line with slope of σ. Lenth's PSE is used to estimate σ and a blue line with this slope is shown on the plot.

If a transformation to orthogonality has been applied, the vertical axis represents the Normalized Estimates. These are the Orthog *t*-Ratio values found in the Parameter Estimate Population report. (The Orthog *t*-Ratio values are the Orthog Coded estimates divided by the Coded Scale Lenth PSE.)

Because the estimates are normalized by an estimate of σ, the points corresponding to inactive effects should fall along a line of slope 1. A red line with slope 1 is shown on the plot, as well as a blue line with slope equal to the t-Test Scale Lenth PSE.

In all cases, estimates that deviate from normality at the 0.20 level, based on the *p*-values in the Parameter Estimate Population report, are labeled on the plot.

Half-Normal Plot

The half normal plot shows the absolute values of effects. The construction of the axes and the lines displayed mirror those aspects of normal plot.

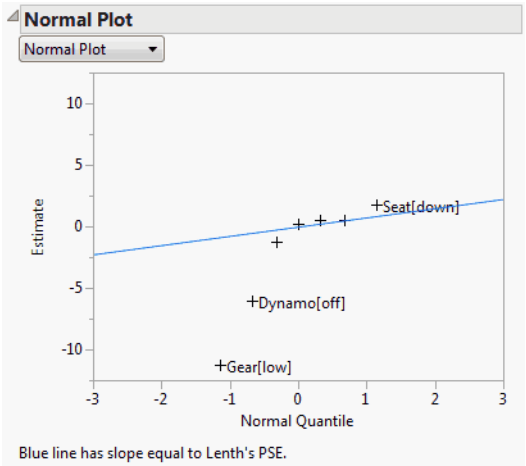
Figure 3.44 shows the Normal Plot report for the Bicycle.jmp sample data table. No transformation is needed for this model, so the plot shows the raw estimates plotted against their normal quantiles. A line with slope equal to Lenth's PSE is shown on the plot. The plot suggests that Gear, Dynamo, and Seat are active factors.

Example of a Normal Plot

1. Select **Help > Sample Data Library** and open Bicycle.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Select HBars through Brkfast and click **Add**.
5. Click **Run**.
6. Click the Response Y red triangle and select **Effect Screening > Normal Plot**.

   The following Normal Plot appears.

Figure 3.44 Normal Plot



Bayes Plot Report

The Bayes Plot report gives another approach to determining which effects are active. This report helps you compute posterior probabilities using a Bayesian approach. This method, due to Box and Meyer (1986), assumes that the estimates are a mixture from two distributions. The majority of the estimates, corresponding to inactive effects, are assumed to be pure random normal noise with variance σ^2. The remaining estimates, the active ones, are assumed to come from a *contaminating* distribution that has a variance K times larger than σ^2.

Term Gives the model term corresponding to the parameter estimate.

Estimate Gives the parameter estimate. The Bayes plot is constructed with respect to estimates that have estimated standard deviation equal to 1. If the estimates are not correlated, the *t*-Ratio is used. If the estimates are correlated, the Orthog *t*-Ratio is used.

Prior Prob Enables you to specify a probability that the estimate is nonzero (equivalently, that the estimate is in the contaminating distribution). Prior probabilities for estimates are usually set to equal values. The commonly recommended value of 0.2 appears initially, though you can change it.

K Contam The value of the contamination coefficient, representing the ratio of the contaminating distribution variance to the error variance. *K* is commonly set to 10, which is the default value.

Std Err Scale If there are degrees of freedom for an estimate of standard error, this value is set to 1. JMP uses this value because the estimates used in the report are transformed and scaled to unit variance. The value is set to 0 for a saturated model with no estimate of error. If you specify a different value, think of it in terms of a scale factor of the RMSE estimate.

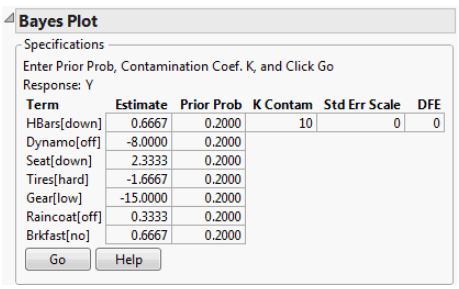
DFE Gives the error degrees of freedom.

The specifications window, showing default settings for a Bayes Plot for the Bicycle.jmp sample data table, is shown in Figure 3.45. Clicking Go in this window updates the report to show Posterior probabilities for each of the terms and a bar chart (Figure 3.46).

Example of a Bayes Plot

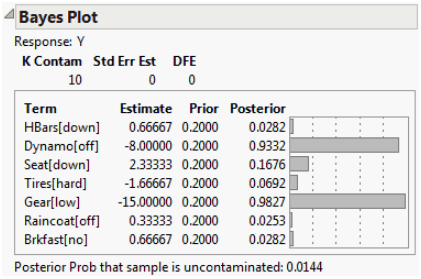
1. Select **Help > Sample Data Library** and open Bicycle.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Select HBars through Brkfast and click **Add**.
5. Click **Run**.
6. Click the Response Y red triangle and select **Effect Screening > Bayes Plot**.

Figure 3.45 Bayes Plot Specifications



7. Click **Go** to calculate the posterior probabilities.

Figure 3.46 Bayes Plot Report



The note beneath the plot in the Bayes Plot report gives the Posterior Prob that the sample is uncontaminated. Posterior Prob is the probability, based on the priors and the data, that there are no active effects whatsoever. The probability is small, 0.0144, indicating that it is likely that there are active effects. The Posterior probability column suggests that at least Dynamo and Gear are active effects.

Pareto Plot Report

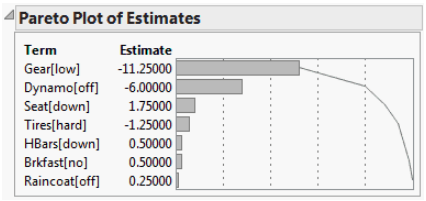
The Pareto Plot report presents a Pareto chart of the absolute values of the estimates. Figure 3.47 shows a Pareto Plot report for the Bicycle.jmp sample data table.

- If the original estimates have equal variances and are not correlated, the original estimates are plotted.
- If the original estimates have unequal variances and are not correlated, the *t* ratios are plotted.
- If the original estimate have unequal variances and are correlated, the Orthog Coded estimates are plotted.

The cumulative sum line in the plot sums the absolute values of the estimates. Keep in mind that the orthogonal estimates depend on the order of entry of terms into the model.

Note: The estimates that appear in the plot are standardized to have equal variances and are transformed to be orthogonal. You have the option of undoing these transformations. See "Transformations".

Figure 3.47 Pareto Plot



Factor Profiling

The Factor Profiling menu helps you explore and visualize your estimated model. You can explore the shape of the response surface, find optimum settings, simulate response data based on your specified noise assumptions, and transform the response if needed.

The Profiler, Contour Profiler, Mixture Profiler, and Surface Profiler are extremely versatile tools whose use extends beyond standard least squares models. For more information about their interpretation and use, see *Profilers*.

If the response column in your model contains an invertible transformation of one variable, the profilers and Interaction Plots show the response on the untransformed scale.

The following Factor Profiling options are available:

Note: If your model contains an expression or vector as an effect, most of these options are not available.

Profiler Shows prediction traces for each X variable. Enables you to find optimum settings for one or more responses and to explore response distributions using simulation. See "Profiler" and *Profilers*.

Interaction Plots Shows a matrix of interaction plots, when there are interaction effects in the model. See "Interaction Plots".

Contour Profiler Provides an interactive contour profiler, which is useful for optimizing response surfaces graphically. See "Contour Profiler" and *Profilers*.

Mixture Profiler Shows response contours of mixture experiment models on a ternary plot. See "Mixture Profiler" and *Profilers*.

Note: This option appears only if you apply the Mixture Effect attribute to three or more effects or the Mixture property to three or more columns.

Cube Plots Shows predicted values for the extremes of the factor ranges laid out on the vertices of cubes. See "Cube Plots".

Box Cox Y Transformation Finds a Box-Cox power transformation of the response that is best in terms of satisfying the normality and homogeneity of variance assumptions. See "Box Cox Y Transformation".

Surface Profiler Shows a three-dimensional surface plot of the response surface. See "Surface Profiler" and *Profilers*.

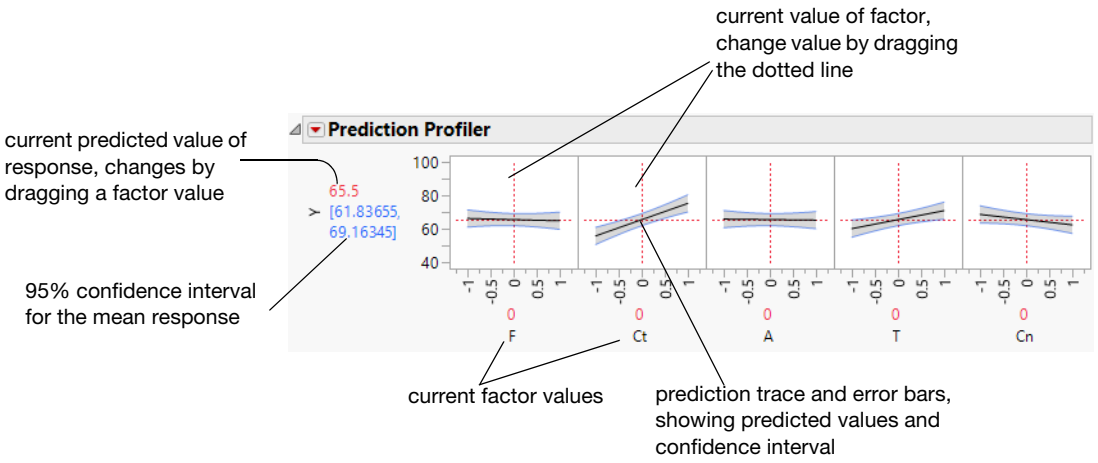
Profiler

Note: For more information about the profiler, see *Profilers*.

The **Profiler** (or Prediction Profiler) shows prediction traces for each *X* variable.

Figure 3.48 illustrates part of the profiler for the Reactor.jmp sample data table. The vertical dotted line for each *X* variable shows its *current value* or *current setting*. Use the Profiler to change one variable at a time in order to examine the effect on the predicted response.

Figure 3.48 Illustration of Prediction Traces



The factors F and Ct in Figure 3.48 are continuous. If the factor is nominal, the levels are displayed on the horizontal axis.

For each X variable, the value above the factor name is its current value. You change the current value by clicking in the graph or by dragging the dotted line where you want the new current value to be.

- The horizontal dotted line shows the *current predicted value* of each Y variable for the current values of the X variables.
- The black lines within the plots show how the predicted value changes when you change the current value of an individual X variable. The 95% confidence interval for the predicted values is shown by a dotted curve surrounding the prediction trace (for continuous variables) or an error bar (for categorical variables).

Interaction Plots

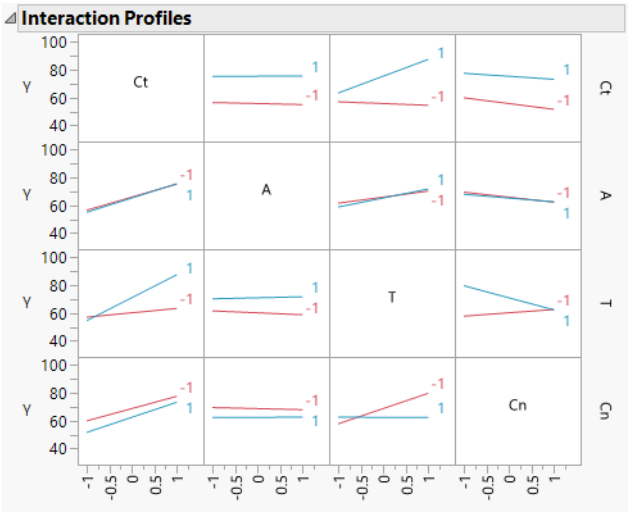
When there are interaction effects in the model, the Interaction Plots option shows a matrix of interaction plots. Each cell of the matrix contains a plot whose horizontal axis is scaled for the effect displayed in the column in which the plot appears. Line segments are plotted for the interaction of that effect with the effect displayed in the corresponding row. So, an interaction plot shows the interaction of the row effect with the column effect.

A line segment is plotted for each level of the row effect. Response values predicted by the model are joined by line segments. Non-parallel line segments give visual evidence of possible interactions. However, the *p*-value for such a suggested interaction should be checked before concluding that it exists. Figure 3.49 gives an interaction plot matrix for the Reactor.jmp sample data table.

Example of Interaction Plots

1. Select **Help > Sample Data Library** and open Reactor.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Make sure that the **Degree** box has a 2 in it.
5. Select Ct, A, T, and Cn and click **Macros > Factorial to Degree**.
6. Click **Run**.
7. Click the Response Y red triangle and select **Factor Profiling > Interaction Plots**.

Figure 3.49 Interaction Plots



The plot corresponding to the T*Cn interaction is the third plot in the bottom row of plots or equivalently, the third plot in the last column of plots. Either plot shows that the effect of Cn on Y is fairly constant at the low level of T, whether Cn is set at a high or low level. However, at the high level of T, the effect of Cn on Y differs based on its level. Cn at –1 leads to a higher predicted Y than Cn at 1. Note that this interaction is significant with a p-value < 0.0001.

In certain designs, two-way interactions are aliased with other two-way interactions. When this aliasing occurs, cells in the Interaction Profiles plot corresponding to these interactions are dimmed.

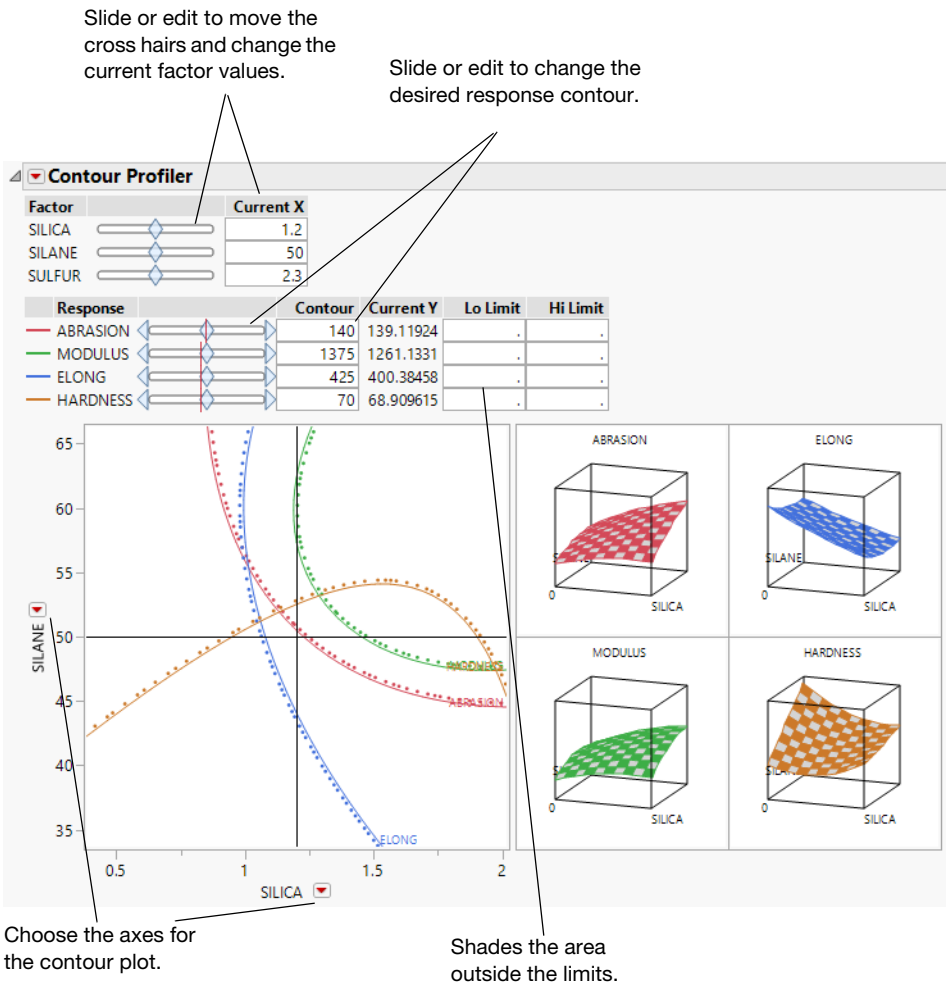
Contour Profiler

Note: For more information about the contour profiler, see *Profilers*.

Use the interactive Contour Profiler for optimizing response surfaces graphically. The Contour Profiler shows contours for the fitted model for two factors at a time. The report also includes a surface plot.

Figure 3.50 shows a contour profiler view for the Tiretread.jmp sample data table. Run the data table script **RSM for 4 Responses** and select **Profilers > Contour Profiler** from the Least Squares Fit report menu.

Figure 3.50 Contour Profiler



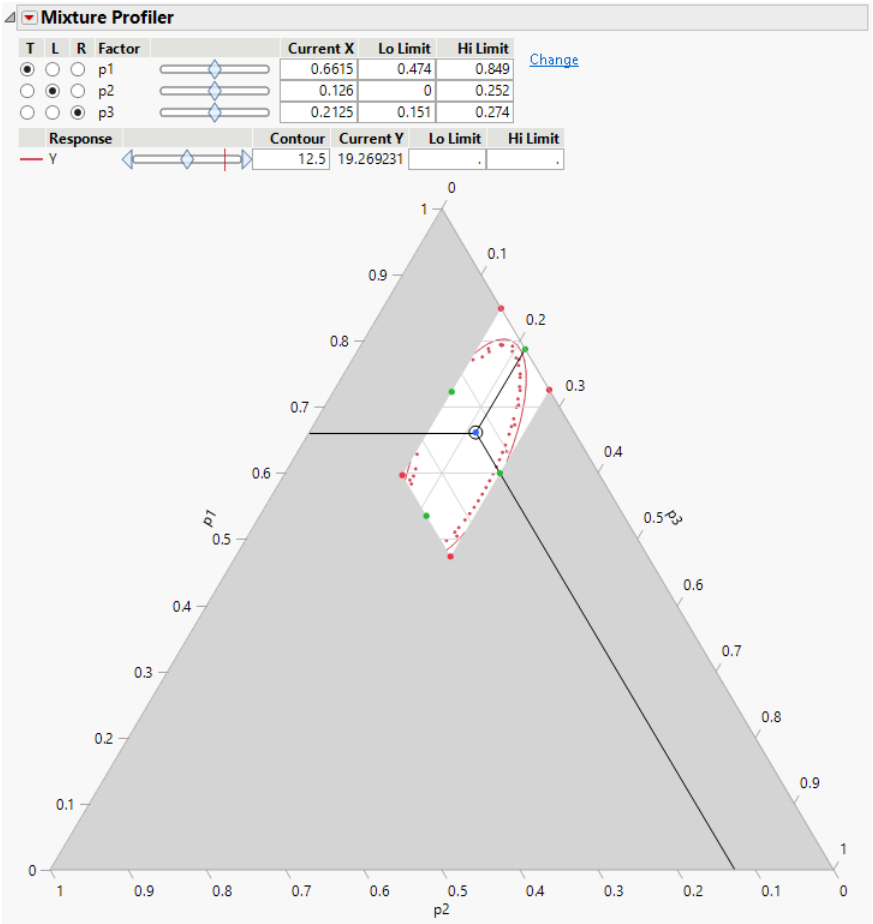
Mixture Profiler

Note: This option appears only if you specify the **Macros > Mixture Response Surface** option for an effect. For more information about the mixture profiler, see *Profilers*.

The Mixture Profiler shows response contours of mixture experiment models on a ternary plot. Use the Mixture Profiler when three or more factors in your experiment are components in a mixture. The Mixture Profiler helps you visualize and optimize the response surfaces of your experiment.

Figure 3.51 shows the Mixture Profiler for the model in the Plasticizer.jmp sample data table. Run the **Model** data table script and then select **Factor Profiling > Mixture Profiler** from the report's red triangle menu. You modify plot axes for the factors by selecting different radio buttons at the top left of the plot. The Lo and Hi Limit columns at the upper right of the plot let you enter constraints for both the factors and the response.

Figure 3.51 Mixture Profiler



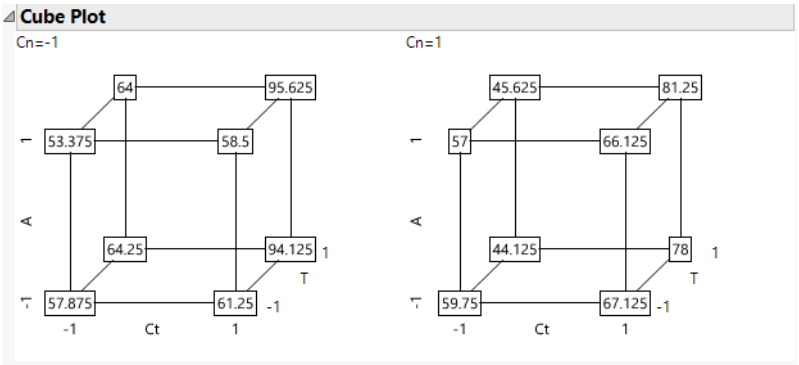
Cube Plots

The Cube Plots option displays predicted values for the extremes of the factor ranges. These values appear on the vertices of cubes (Figure 3.52). The vertices are defined by the smallest and largest observed values of the factor. When you have multiple responses, the multiple responses are shown stacked at each vertex.

Example of Cube Plots

1. Select **Help > Sample Data Library** and open Reactor.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Make sure that the **Degree** box has a 2 in it.
5. Select Ct, A, T, and Cn and click **Macros > Factorial to Degree**.
6. Click **Run**.
7. Click the Response Y red triangle and select **Factor Profiling > Cube Plots**.

Figure 3.52 Cube Plots



Note that there is one cube for Cn = –1 and one for Cn = 1. To change the layout so that the factors are mapped to different cube coordinates, click a factor name in the first cube. Drag it to cover the factor name for the desired axis. For example, in Figure 3.52, if you click T and drag it over Ct, then T and Ct (and their corresponding coordinates) exchange places. To see the levels of Cn in a single cube, exchange it with another factor in the first cube by dragging it over that factor.

Box Cox Y Transformation

A Box-Cox power transformation is used to transform the response so that the usual regression assumptions of normality and homogeneity of variance are more closely satisfied. The transformed response can then be fit using a regression model. However, you can also use the Box-Cox power transformation to transform a variable for other reasons. This transformation is appropriate only when the response, Y, is strictly positive.

A commonly used transformation raises the response to some power. Box and Cox (1964) formalized and described this family of power transformations. The formula for the transformation is constructed to provide a continuous definition in terms of the parameter λ, and so that the error sums of squares are comparable. Specifically, the following equation provides the family of transformations:

$$Y_\lambda = \begin{cases} \dfrac{y^\lambda - 1}{\lambda \dot{y}^{\lambda-1}} & \text{if } \lambda \neq 0 \\ \dot{y}\ln(y) & \text{if } \lambda = 0 \end{cases}$$

Here, $\dot{y}$ denotes the geometric mean.

The Box Cox Y Transformation option fits transformations from λ = –2 to 2 in increments of 0.2. To choose a value of λ, the likelihood function for each of these transformations is computed. They are computed under the assumption that the errors are independent and normal with mean zero and variance σ^2. The value of λ that maximizes the likelihood is selected. This value also minimizes the SSE over the values of λ. The value of λ that minimizes the SSE is found using a quadratic interpolation between the two incremental grid points surrounding the grid point with the smallest SSE. If this interpolation results in a negative SSE value, then the grid value of λ that minimizes the SSE is reported as the best λ.

The Box-Cox Transformations report displays a plot showing the sum of squared errors (SSE) values against the values of λ. The horizontal red line on the plot represents a one-sided 95% confidence interval for λ. This confidence interval is based on the confidence region defined in Box and Cox (1964, p. 216). The confidence region is defined by the following inequality:

$$SSE(\lambda) < SSE(\lambda_{best}) * \exp(ChiSquareQuantile(0.95,1) / dfe)$$

where

$SSE(\lambda_{best})$ is the SSE calculated using the reported Best λ

ChiSquareQuantile(0.95,1) is the 0.95^{th} quantile of a χ^2 distribution with 1 degree of freedom

dfe is the error degrees of freedom in the Analysis of Variance table for the regression model

The Box Cox Transformations report provides the following options:

Refit with Transform Enables you to specify a value for lambda to define a transformed Y variable and then provides a least squares fit to the transformed variable.

Replace with Transform Enables you to specify a value for lambda to define a transformed Y variable and then replaces the existing least squares fit with a fit to the transformed variable. If you have multiple responses, Replace with Transform replaces only the report for the response that you are transforming.

Save Best Transformation Creates a new column in the data table and saves the formula for the best transformation.

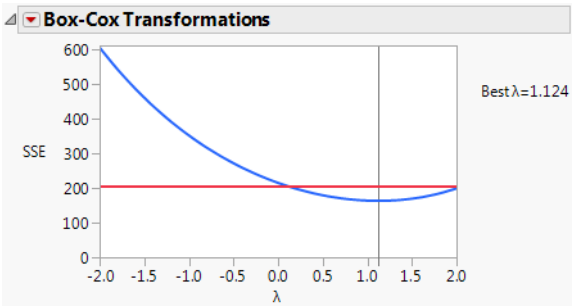
Save Specific Transformation Enables you to specify a value for lambda and creates a column in the data table with the formula for your specified transformation.

Table of Estimates Creates a new data table containing parameter estimates and SSE values for all λ from –2 to 2, in increments of 0.2.

Example of Box-Cox Y Transformation

1. Select **Help > Sample Data Library** and open Reactor.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Make sure that the **Degree** box has a 2 in it.
5. Select F, Ct, A, T, and Cn and click **Macros > Factorial to Degree**.
6. Click **Run**.
7. Click the Response Y red triangle and select **Factor Profiling > Box Cox Y Transformation**.

Figure 3.53 Box Cox Y Transformation



The plot shows that the best values of λ are between 0.1 and 2.0. The value that JMP selects, using interpolation between the best two values in the 0.2-unit grid of λ values, is 1.124.

8. (Optional) To see the SSE values used to construct the graph, select **Table of Estimates**.

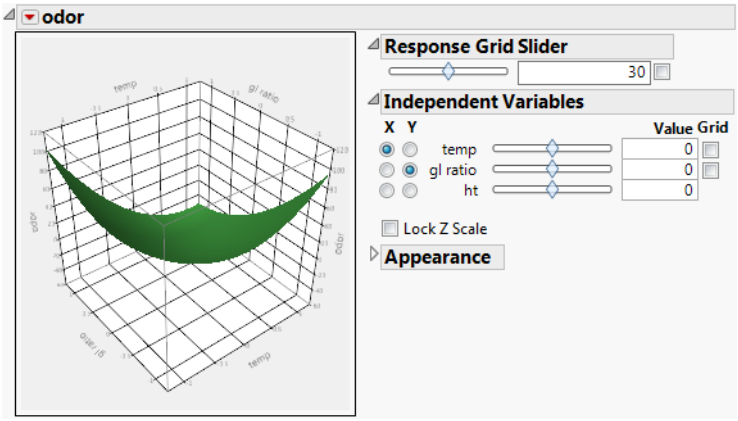
Surface Profiler

Note: For more information about the surface profiler, see *Profilers*.

The Surface Profiler shows a three-dimensional surface plot of the response surface.

Figure 3.54 shows the Surface Profiler for the model in the Odor.jmp sample data table. Run the **Model** data table script and then select **Factor Profiling > Surface Profiler** from the report's red triangle menu. You can change the variables on the axes using the radio buttons under Independent Variables. Also, you can plot points by clicking Actual under Appearance.

Figure 3.54 Surface Plot



Row Diagnostics

The row diagnostics menu addresses issues specific to rows, or observations.

Plot Regression Shows a Regression Plot report, displaying a scatterplot of the data and regression lines for each level of the categorical effect.

This option appears only if there is exactly one continuous effect and no more than one categorical effect in the model. In that case, the Regression Plot report is provided by default.

Plot Actual by Predicted Shows an Actual by Predicted plot, which plots the observed values of Y against the predicted values of Y. This plot is the leverage plot for the whole model. See "Leverage Plots".

Note: The Actual by Predicted Plot is shown by default when Effect Leverage or Effect Screening is selected as the Emphasis in the Fit Model launch window and the RSquare value is less than 0.999.

Plot Effect Leverage Shows a Leverage Plot report for each effect in the model. The plot shows how observations influence the test for that effect and gives insight on multicollinearity. See "Leverage Plots".

Note: Effect Leverage Plots are shown by default when Effect Leverage is selected as the Emphasis in the Fit Model launch window and the RSquare value is less than 0.999. They appear to the right of the Whole Model report. When another Emphasis is selected, the Effect Leverage Plots appear in the Effect Details report. In all cases, the option Regression Reports > Effect Details must be selected in order for Effect Leverage plots to display.

Plot Residual by Predicted Shows a Residual by Predicted Plot report. The plot shows the residuals plotted against the predicted values of Y. You typically want to see the residual values scattered randomly about zero.

Note: The Residual by Predicted Plot is shown by default when Effect Leverage or Effect Screening is selected as the Emphasis in the Fit Model launch window and the RSquare value is less than 0.999.

Plot Residual by Row Shows a Residual by Row Plot report. The residual values are plotted against the row numbers. This plot can help you detect patterns that result from the row ordering of the observations.

Plot Studentized Residuals Shows a Studentized Residuals plot. Each point on the plot is computed using an estimate of its standard deviation obtained with the current observation deleted. These residuals are also called *RStudent* or *externally Studentized* residuals.

The plot contains two sets of limits:

- The outer limits that appear in red on the plot are 95% Bonferroni limits. These limits are placed at ± tQuantile(0.025/n, n–p–1), where n is the number of observations and p is the number of predictors.
- The inner limits that appear in green on the plot are 95% individual t distribution limits. These limits are placed at ± tQuantile(0.025, n–p–1), where n is the number of observations and p is the number of predictors.

Points that fall outside the red limits should be treated as probable outliers. Points that fall outside the green limits but within the red limits should be treated as possible outliers, but with less certainty. The confidence level of 95% for these limits is not affected by your selection in the Set Alpha Level option in the Model Specification window.

Caution: The residuals saved using Save Columns > Studentized Residuals are *not* externally Studentized.

Note: If the model contains random effects and REML is the specified Method in the launch window, the Studentized Residuals plot does not contain limits and the points that are plotted are *not* externally Studentized.

Plot Residual by Normal Quantiles (Not available when REML is the specified Method in the launch window.) Shows a Residual Normal Quantile Plot. The residual values are plotted against quantiles of the normal distribution. This plot can help you assess the assumption of normality of the residuals.

Press Shows a Press Report giving the Press statistic and its root mean square error (RMSE). The Press statistic is useful when comparing multiple models. Models with lower Press statistics are favored. See "Press".

Durbin-Watson Test (Not available when you specify a Frequency column.) Shows the Durbin-Watson report, which gives a statistic to test whether the residuals have first-order autocorrelation. The report also displays the autocorrelation of the residuals and Prob<DW, which is the exact probability associated with the statistic. This option is appropriate only for time series data and assumes that your observations are in time order.

Leverage Plots

An effect leverage plot for X is useful in the following ways:

- You can see which points might be exerting influence on the hypothesis test for X.
- You can spot unusual patterns and violations of the model assumptions.
- You can spot multicollinearity issues.

Construction

A leverage plot for an effect shows the impact of adding this effect to the model, given the other effects already in the model. For illustration, consider the construction of an effect leverage plot for a single continuous effect X. See "Horizontal Axis Scaling" for information about the scaling of the horizontal axis in more general situations.

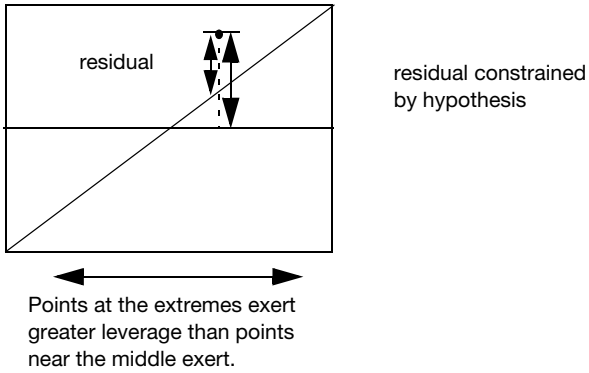
The response Y is regressed on all the predictors except X, and the residuals are obtained. Call these residuals the Y-residuals. Then X is regressed on all the other predictors in the model and the residuals are computed. Call these residuals the X-residuals. The X-residuals might contain information beyond what is present in the Y-residuals, which were obtained without X in the model.

The effect leverage plot for X is essentially a scatterplot of the X-residuals against the Y-residuals (Figure 3.57). To help interpretation and comparison with other plots that you might construct, JMP adds the mean of Y to the Y-residuals and the mean of X to the X-residuals. The translated Y-residuals are called the Y Leverage Residuals and the translated X-residuals are called X Leverage values. The points on the Effect Leverage plots are these X Leverage and Y Leverage Residual pairs.

JMP fits a least squares line to these points as well as confidence bands for the mean; the line of fit is solid red and the confidence bands are shaded red. The slope of the least squares line is precisely the estimate of the coefficient on X in the model where Y is regressed on X and the other predictors. The dashed horizontal blue line is set at the mean of the Y Leverage Residuals. This line describes a situation where the X residuals are not linearly related to the Y residuals. If the line of fit has nonzero slope, then adding X to the model can be useful in terms of explaining variation.

Figure 3.55 shows how residuals are depicted in the leverage plot. The distance from a point to the line of fit is the residual for a model that includes the effect. The distance from the point to the horizontal line is what the residual error would be without the effect in the model. In other words, the mean line in the leverage plot represents the model where the hypothesized value of the parameter (effect) is constrained to zero.

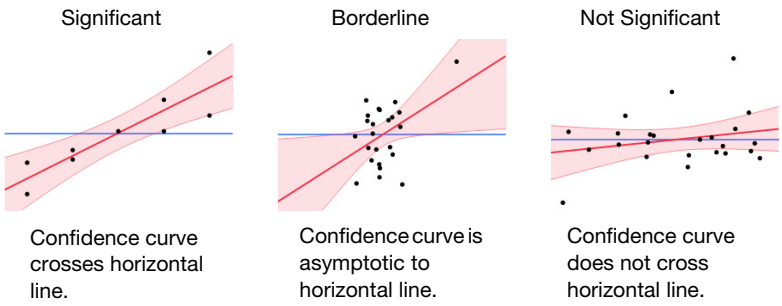
Figure 3.55 Illustration of a Generic Leverage Plot



Confidence Curves

Confidence curves for the line of fit are shown on leverage plots. These curves provide a visual indication of whether the test of interest is significant at the 5% level (or at the Set Alpha Level that you specified in the Fit Model launch window). If the confidence region between the curves contains the horizontal line representing the hypothesis, then the effect is not significant. If the curves cross the line, the effect is significant. See the examples in Figure 3.56.

Figure 3.56 Comparison of Significance Shown in Leverage Plots



Horizontal Axis Scaling

If the modeling type of a predictor X is continuous, then the horizontal axis is scaled in terms of the units of the X. The horizontal axis range mirrors the range of X values. The slope of the line of fit in the leverage plot is the parameter estimate for X. See the left illustration in Figure 3.57.

If the effect is nominal or ordinal, or if the effect is a complex effect such as an interaction, then the horizontal axis cannot represent the values of the effect directly. In this case the horizontal axis is scaled in units of the response, and the line of fit is a diagonal with a slope of 1. The Whole Model leverage plot, where the hypothesis of interest is that all parameter values are zero, uses this scaling. See "Leverage Plot Details". For this plot, the horizontal axis is scaled in terms of predicted response values for the whole model, as illustrated by the right-hand plot in Figure 3.57.

The leverage plot for the linear effect in a simple regression is the same as the traditional plot of actual response values against the predictor.

Leverage

The term *leverage* is used because these plots help you visualize the influence of points on the test for including the effect in the model. A point that is horizontally distant from the center of the plot exerts more influence on the effect test than does a point that is close to the center. Recall that the test for an effect involves comparing the sum of squared residuals to the sum of squared residuals of the model with that effect removed. At the extremes, the differences of the residuals before and after being constrained by the hypothesis tend to be comparatively larger. Therefore, these residuals tend to have larger contributions to the sums of squares for that effect's hypothesis test.

Multicollinearity

Multicollinearity is a condition where two or more predictors are highly related, or more technically, involved in a nearly linear dependent relationship. When multicollinearity is present, standard errors can be inflated and parameters estimates can be unstable. If an effect is collinear with other predictors, the horizontal values of the points tend to cluster toward the middle of the plot. This situation indicates that the slope of the line of fit is unstable.

The Whole Model Actual by Predicted Plot

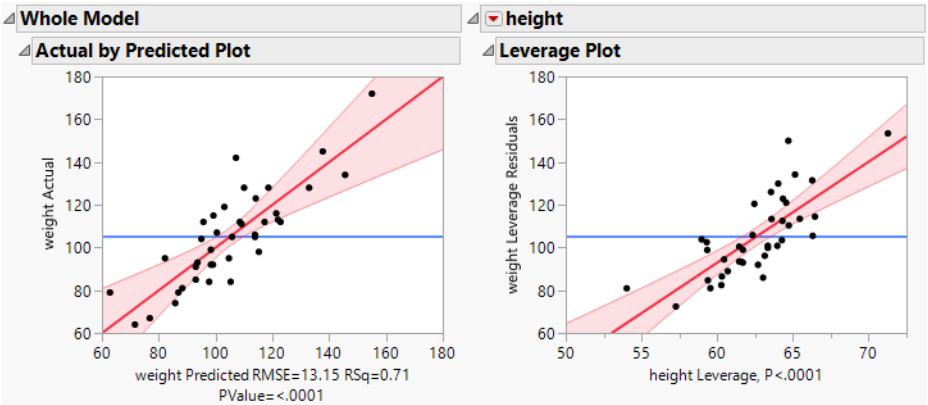
The Plot Effect Leverage option produces a leverage plot for each effect in the model. In addition, the Actual by Predicted plot can be considered to be a leverage plot. This plot lets you visualize the test that all the parameters in the model (except the intercept) are zero. The same test is conducted analytically in the Analysis of Variance report. See "Leverage Plot Details" for more information about this plot.

Example of a Leverage Plot for a Linear Effect

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Model**.
3. Select weight and click **Y**.
4. Select height, age, and sex, and click **Add**.
5. Click **Run**.

The Whole Model Actual by Predicted Plot and the effect Leverage Plot for height are shown in Figure 3.57. The Whole Model plot, on the left, tests for all effects. You can infer that the model is significant because the confidence curves cross the horizontal line at the mean of the response, weight. The Leverage Plot for height, on the right, also shows that height is significant, even with age and sex in the model. Neither plot suggests concerns relative to influential points or multicollinearity.

Figure 3.57 Whole Model and Effect Leverage Plots



Press

The *Press*, or *prediction error sum of squares*, statistic is an estimate of prediction error computed using leave-one-out cross validation. In leave-one-out cross validation, each observation, in turn, is removed. Consider a specific observation. The model is fit with that observation withheld and then a predicted value is obtained for that observation. The residual for that observation is computed. This procedure is applied to all observations and the residuals are squared and summed to give the Press value.

Specifically, the Press statistic is given by the following:

$$\text{Press} = \sum_{i=1}^{n} (\hat{y}_{(i)} - y_i)^2$$

where n is the number of observations, y_i is the observed response value for the i^{th} observation, and $\hat{y}_{(i)}$ is the predicted response value for the i^{th} observation. These values are based on a model fit without including that observation.

The Press RMSE is defined as $\sqrt{Press/n}$.

The Press RSquare is defined as 1 - $Press/SS_{\text{Total}}$.

Save Columns

Each Save Columns option adds one or more new columns to the current data table. Additional Save Columns options appear when the fitting method is REML. These are detailed in "REML Save Columns Options".

Note the following:

- When formulas are created, they are entered as Formula column properties.
- For many of the new columns, a Notes column property is added describing the column and indicating that Fit Least Squares created it.
- For the Predicted Formula and Predicted Values options, a Predicting column property is added. This property is used internally by JMP in conducting model comparisons (Analyze > Predictive Modeling > Model Comparison). When you fit many models, it is also useful to you because it documents the origin of the column.

The following Save Columns options are available:

Prediction Formula Creates a new column called Pred Formula <colname> that contains both the formula and the predicted values. A Predicting column property is added, noting the source of the prediction.

Note: Pred Formula <colname> inherits certain properties from <colname>. These include Response Limits, Spec Limits, and Control Limits. If you change these properties for <colname> after saving Pred Formula <colname>, they will not update in Pred Formula <colname>.

See "Prediction Formula".

Predicted Values Creates a new column called Predicted <colname> that contains the predicted values computed by the specified model. Both a Notes and a Predicting column property are added, noting the source of the prediction.

Note: Predicted <colname> inherits certain properties from <colname>. These include Response Limits, Spec Limits, and Control Limits. If you change these properties for <colname> after saving Predicted <colname>, they will not update in Predicted <colname>.

Residuals Creates a new column called Residual <colname> that contains the observed response values minus their predicted values.

Mean Confidence Interval Creates two new columns called Lower 95% Mean <colname> and Upper 95% Mean <colname>. These columns contain the lower and upper 95% confidence limits for the mean response.

Note: If you press Shift while selecting the option, you are prompted to enter an α level for the computations.

Indiv Confidence Interval Creates two new columns called Lower 95% Indiv <colname> and Upper 95% Indiv <colname>. These columns contain lower and upper 95% confidence limits for individual response values.

Note: If you press Shift while selecting the option, you are prompted to enter an α level for the computations.

Studentized Residuals Creates a new column called Studentized Resid <colname> that contains the residuals divided by their standard errors.

Externally Studentized Residuals (Not available when the fitting method is REML.) Creates a new column called Externally Studentized Residuals <colname> that contains the residuals divided by standard error estimates that exclude the current row. See "Plot Studentized Residuals".

Hats Creates a new column called h <colname>. The column values are the diagonal values of the matrix $X(X'X)^{-1}X'$, sometimes called hat values.

Std Error of Predicted Creates a new column called StdErr Pred <colname> that contains the standard errors of the predicted mean response.

Std Error of Residual Creates a new column called StdErr Resid <colname> that contains the standard errors of the residual values.

Std Error of Individual Creates a new column called StdErr Indiv <colname> that contains the standard errors of the individual predicted response values.

Effect Leverage Pairs Creates a set of new columns that contain the X Leverage values and Y Leverage Residuals for each leverage plot. For each effect in the model, two columns are added. If the response column name is R and the effect is X, the new column names are:

- X Leverage of X for R
- Y Leverage of X for R

In the columns panel, these columns are organized in a columns group called Leverage.

Cook's D Influence Creates a new column called Cook's D Influence <colname>, which contains values of the Cook's *D* influence statistic.

StdErr Pred Formula Creates a new column called PredSE <colname> that contains both the formula and the values for the standard error of the predicted values.

Note: The saved formula can be large. If you do not need the formula, use the Std Error of Predicted option.

Mean Confidence Limit Formula Creates two new columns called Lower 95% Mean <colname> and Upper 95% Mean <colname>. These columns contain both the formulas and the values for lower and upper 95% confidence limits for the mean response.

Note: If you press Shift while selecting the option, you are prompted to enter an α level for the computations.

Indiv Confidence Limit Formula Creates two new columns called Lower 95% Indiv <colname> and Upper 95% Indiv <colname>. These columns contain both the formulas and the values for lower and upper 95% confidence limits for individual response values.

Note: If you press Shift while selecting the option, you are prompted to enter an α level for the computations.

Save Coding Table Creates a new data table whose first columns show the JMP coding for all model parameters. The last column gives the values of the response. If you entered more than one response column, all of these columns appear as the last columns in the coding table.

Note: The coding data table contains a table variable called Original Data that gives the name of the data table that was used for the analysis. In the case where a By variable is specified, the Original Data table variable gives the By variable and its level.

JMP PRO **Publish Prediction Formula** Creates a prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

JMP PRO **Publish Standard Error Formula** Creates a standard error formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

JMP PRO **Publish Mean Confid Limit Formula** Creates confidence limit formulas for the mean response and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

JMP PRO **Publish Indiv Confid Formula** Creates confidence formulas for individual response values and saves them as formula column scripts in the Formula Depot platform. If a

Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Prediction Formula

Pred Formula <colname> differs from Predicted <colname> in that it contains the prediction formula. Right-click in the Pred Formula <colname> column heading and select **Formula** to see the prediction formula. The prediction formula can require considerable space if the model is large. If you do not need the formula with the column of predicted values, use the **Save Columns > Predicted Values** option. For information about formulas, see *Using JMP*.

The Prediction Formula option is useful for predicting values in new rows or for use with the profilers. The profilers are available in the Fit Least Squares report (Factor Profiling). However, when your data table includes formula columns, you can also use the profilers provided in the **Graph** menu. When you are analyzing multiple responses, accessing the profilers from the **Graph** menu can be useful.

Note: If you select **Graph > Profiler** to access the profilers, first save the formula columns to the data table using Prediction Formula and StdErr Pred Formula. Then place both of these formulas into the Y, Prediction Formula role in the Profiler window. After you click **OK**, specify whether you want to use PredSE <colname> to construct confidence intervals for Pred Formula <colname>. Otherwise, JMP creates a separate profiler plot for PredSE <colname>.

Effect Summary Report

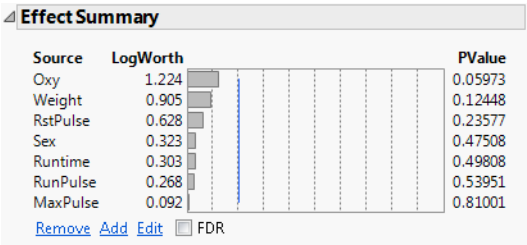
The Effect Summary option shows an interactive report. It gives a plot of the LogWorth (or FDR LogWorth) values for the effects in the model. The report also provides controls that enable you to add or remove effects from the model. The model fit report updates automatically based on the changes made in the Effects Summary report.

The Effect Summary report is available in the following personalities:

- Standard Least Squares
- Nominal Logistic
- Ordinal Logistic
- Proportional Hazard
- Parametric Survival
- Generalized Linear Model

Figure 3.58 shows the initial view of the Effect Summary report for the Fitness.jmp data table. The check box labeled **FDR** controls the columns that appear in the summary table.

Figure 3.58 Effect Summary Report



Effect Summary Table Columns

The Effect Summary table contains the following columns:

Source Lists the model effects, sorted by ascending *p*-values.

LogWorth Shows the LogWorth for each model effect, defined as $-\log_{10}(p\text{-value})$. This transformation adjusts *p*-values to provide an appropriate scale for graphing. A value that exceeds 2 is significant at the 0.01 level (because $-\log_{10}(0.01) = 2$).

FDR LogWorth Shows the False Discovery Rate LogWorth for each model effect, defined as $-\log_{10}(\text{FDR PValue})$. This is the best statistic for plotting and assessing significance. However, it is highly dependent on the ordering of the significances, is conservative for positively correlated tests, and does not give experimentwise protection at the alpha level. Select the **FDR** check box to replace the LogWorth column with the **FDR LogWorth** column.

Bar Graph Shows a bar graph of the LogWorth (or FDR LogWorth) values. The graph has dashed vertical lines at integer values and a blue reference line at 2.

PValue Shows the *p*-value for each model effect. This is generally the *p*-value corresponding to the significance test displayed in the Effect Tests table or Effect Likelihood Ratio Tests table of the model report.

FDR PValue Shows the False Discovery Rate *p*-value for each model effect calculated using the Benjamini-Hochberg technique. This technique adjusts the *p*-values to control the false discovery rate for multiple tests. Select the **FDR** check box to replace the **PValue** column with the **FDR PValue** column.

For more information about the FDR correction, see Benjamini and Hochberg (1995). For more information about the false discovery rate, see *Predictive and Specialized Modeling* or Westfall et al. (2011).

Effect Heredity Column Identifies lower-order effects that are components of more significant higher-order effects. The lower-order effects are identified with a caret. See "Effect Heredity".

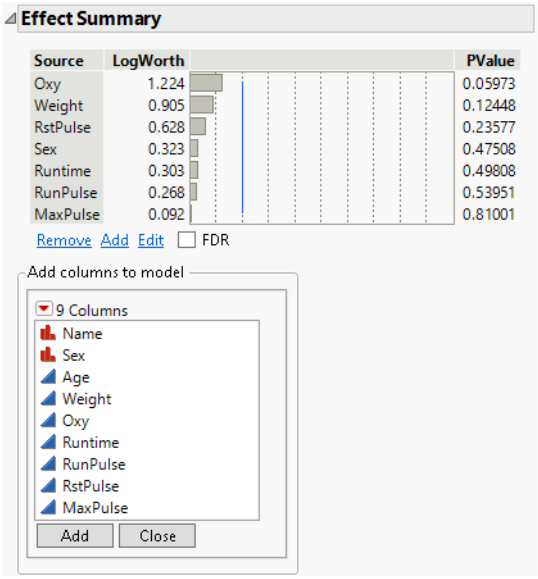
Effect Summary Table Options

The options below the summary table enable you to add and remove effects:

Remove Removes the selected effects from the model. To remove one or more effects, select the rows corresponding to the effects and click the **Remove** button.

Add Opens a panel that contains a list of all columns in the data table. Select columns that you want to add to the model, and then click **Add** below the column selection list to add the columns to the model. Click **Close** to close the panel. Figure 3.59 shows the Add Columns panel.

Figure 3.59 Effect Summary Add Columns Panel



Edit Opens the Edit Model panel, which contains a Select Columns list and an Effects specification panel. The Effects panel resembles the Construct Model Effects panel in the Fit Model launch window. The Edit Model panel enables you to add individual, crossed, nested, and transformed effects. You can also add multiple effects using the Macros menu. For more information about how to construct effects using Add, Cross, Nest, Macros, and Transform, see "Construct Model Effects".

The following options are available to the right of the Effects panel:

Commit applies your updates to the model.

Close closes the panel without making changes to the model.

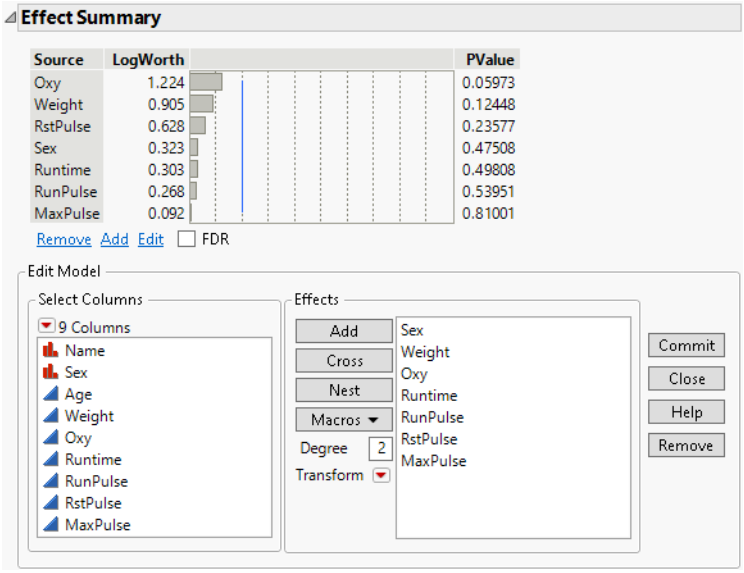
Remove removes one or more selected effects from the Effects list.

Figure 3.60 shows the Edit Model panel.

Tip: The Edit button gives you the greatest degree of control over updates to your model. It includes the functionality of the Remove and Add buttons and enables you to construct effects to add to your model.

Undo Enables you to undo changes to the effects in the model.

Figure 3.60 Effect Summary Edit Model Panel



Effect Heredity

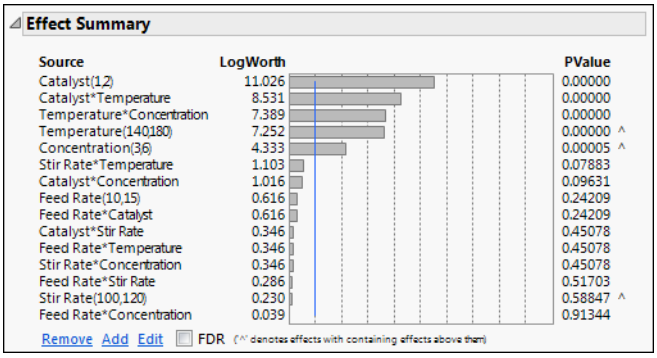
When a model contains significant higher-order effects, you might want to retain some or all of their lower-order components, even though these are not significant. The principle of *strong effect heredity* states that, if a higher-order effect is included in the model, all of its lower-order components should be included as well. The principle of *weak effect heredity* indicates that a chain of components should be included.

When a lower-order component of a higher-order effect appears in the Effect Summary table below the higher-order effect, a caret appears in the right-most column. The caret indicates that the containing higher-order effect is more significant than the lower-order effect. If all higher-order effects that contain a lower-order effect are less significant than the lower-order effect, no caret appears in the row for the lower-order effect.

If you remove an effect marked with a caret, you can choose one of two approaches for removing effects. Choose **Remove all selected effects** to remove all the selected effects, including the ones marked with a caret. Choose **Remove only non-contained effects** to remove only the selected effects that do not have a higher-order effect that still remains in the model.

Figure 3.61 show an example of an Effect Summary table where three lower-order effects appear below higher-order effects that contain the lower-order effects. For example, Stir Rate(100,120) appears below Stir Rate*Temperature.

Figure 3.61 Effect Summary Table with Effect Heredity for Reactor 32 Runs.jmp



Multiple Responses

In the case of multiple responses, each effect appears in each response model, but only one Effect Summary report appears. For each effect, the table shows the minimum *p*-value among the *p*-values for that effect. Adding or removing an effect applies to the models for all of the responses.

Mixed and Random Effect Model Reports and Options

Mixed and random effect models can be specified in the Fit Model launch window. The Standard Least Squares personality fits the variance component covariance structure. Two methods, REML and EMS, are provided for fitting such models.

Note: JMP PRO JMP Pro users are encouraged to use the Mixed Model personality in the Fit Model window. The Mixed Model personality offers a broader set of covariance structures than does Standard Least Squares.

- "Mixed Models and Random Effect Models"
- "Restricted Maximum Likelihood (REML) Method"
- "EMS (Traditional) Model Fit Reports"

Mixed Models and Random Effect Models

A *random effect model* is a model all of whose factors represent random effects. See "Random Effects". Such models are also called *variance component models*. Random effect models are often hierarchical models. A model that contains both fixed and random effects is called a *mixed model*. Repeated measures and split-plot models are special cases of mixed models. Often the term *mixed model* is used to subsume random effect models.

To fit a mixed model, you must specify the random effects in the Fit Model launch window. However, if all of your model effects are random, you can also fit your model in the Variability / Attribute Gauge Chart platform. Only certain models can be fit in this platform. Note that the fitting methods used in the Variability / Attribute Gauge Chart platform do not allow variance component estimates to be negative. For more information about how the Variability / Attribute Gauge Chart platform fits variance components models, see *Quality and Process Methods*.

Random Effects

A random effect is a factor whose levels are considered a random sample from some population. Often, the precise levels of the random effect are not of interest, rather it is the variation reflected by the levels that is of interest (the *variance components*). However, there are also situations where you want to predict the response for a given level of the random effect. Technically, a random effect is considered to have a normal distribution with mean zero and nonzero variance.

Suppose that you are interested in whether two specific ovens differ in their effect on mold shrinkage. An oven can process only one batch of 50 molds at a time. You design a study where three randomly selected batches of 50 molds are consecutively placed in each of the two ovens. Once the batches are processed, shrinkage is measured for five parts randomly selected from each batch.

Note that Batch is a factor with six levels, once for each batch. So, in your model, you include two factors, Oven and Batch. Because you are specifically interested in comparing the effect of each oven on shrinkage, Oven is a fixed effect. But you are not interested in the effect of these specific six batches on the mean shrinkage. These batches are representative of a whole population of batches that could have been chosen for this experiment and to which the results of the analysis must generalize. Batch is considered a random effect. In this experiment, the Batch factor is of interest in terms of the variation in shrinkage among all possible batches. Your interest is in estimating the amount of variation in shrinkage that it explains. (Note that Batch is also nested within Oven, because only one batch can be processed once in one oven.)

Now suppose that you are interested in the weight of eggs for hens subjected to two feed regimes. Ten hens are randomly assigned to feed regimes: Five are given Feed regime A and five are given Feed regime B. However, these ten hens have some genetic differences that are not accounted for in the design of the study. In this case, you are interested the predicted weight of the eggs from certain specific hens as well as in the variance of the weights of eggs among hens.

The Classical Linear Mixed Model

JMP fits the classical linear mixed effects model:

$$Y = X\beta + Z\gamma + \varepsilon$$
$$\gamma \sim N(0, G)$$
$$\varepsilon \sim N(0, \sigma^2 I_n)$$

Here,

- Y is an n x 1 vector of responses
- $\mathbf{X}$ is the n x p design matrix for the fixed effects
- β is a p x 1 vector of unknown fixed effects with design matrix X
- $\mathbf{Z}$ is the n x s design matrix for the random effects
- γ is an s x 1 vector of unknown random effects with design matrix Z
- ε is an n x 1 vector of unknown random errors
- $\mathbf{G}$ is an s x s diagonal matrix with identical entries for each level of the random effect

- I_n is an n x n identity matrix
- γ and ε are independent

The diagonal elements of **G**, as well as σ^2, are called *variance components*. These variance components, together with the vector of fixed effects β and the vector of random effects γ, are the model parameters that must be estimated.

The covariance structure for this model is sometimes called the *variance component* structure (SAS Institute Inc. 2020d, ch. 83). This covariance structure is the only one available in the Standard Least Squares personality.

JMP PRO The Mixed Model personality fits a variety of covariance structures, including Residual, First-order Autoregressive (or *AR(1)*), Unstructured, and Spatial. See "Repeated Structure Tab".

REML versus EMS for Fitting Models with Random Effects

JMP provides two methods for fitting models with random effects:

- REML, which stands for *restricted maximum likelihood* (always the recommended method)
- EMS, which stands for *expected mean square*s (use only for teaching from old textbooks)

The REML method is now the mainstream fitting methodology, replacing the traditional EMS method. REML is considerably more general in terms of applicability than the EMS method. The REML approach was pioneered by Patterson and Thompson (1974). See also Wolfinger et al. (1994) and Searle et al. (1992).

The EMS method, also called the *method of moments*, was developed before the availability of powerful computers. Researchers restricted themselves to balanced situations and used the EMS methodology, which provided computational shortcuts to compute estimates for random effect and mixed models. Because many textbooks still in use today use the EMS method to introduce models containing random effects, JMP provides an option for EMS. (See, for example, McCulloch et al., 2008; Poduri, 1997; Searle et al., 1992.)

The REML methodology performs maximum likelihood estimation of a restricted likelihood function that does not depend on the fixed-effect parameters. This yields estimates of the variance components that are then used to obtain estimates of the fixed effects. Estimates of precision are based on estimates of the covariance matrix for the parameters. Even when the data are unbalanced, REML provides useful estimates, tests, and confidence intervals.

The EMS methodology solves for estimates of the variance components by equating observed mean squares to expected mean squares. For balanced designs, a complex set of rules specifies how estimates are obtained. There are problems in applying this technique to unbalanced data.

For balanced data, REML estimates are identical to EMS estimates. But, unlike EMS, REML performs well with unbalanced data.

Specifying Random Effects and Fitting Method

Models with random effects are specified in the Fit Model launch window. To specify a random effect, highlight it in the Construct Model Effects list and select **Attributes > Random Effect**. This appends &Random to the effect name in the model effect list. For a definition of random effects, see "Random Effects". Random effects can also be specified in a separate effects tab. See "Construct Model Effects Tabs".

In the Fit Model launch window, once the &Random attribute has been appended to an effect, you are given a choice of fitting Method: REML (Recommended) or EMS (Traditional).

Caution: You must declare crossed and nested relationships explicitly. For example, a subject ID might also identify the group that contains the subject, as when each subject is in only one group. In such a situation, subject ID must still be declared as nested within group. Take care to be explicit in defining the design structure.

Unrestricted Parameterization for Variance Components

There are two different approaches to parameterizing the variance components: the *unrestricted* and the *restricted* approaches. The issue arises when there are mixed effects in the model, such as the interaction of a fixed effect with a random effect. Such an interaction term is considered to be a random effect.

In the restricted approach, for each level of the random effect, the sum of the interaction effects across the levels of the fixed effect is assumed to be zero. In the unrestricted approach, the mixed terms are simply assumed to be independent random realizations of a normal distribution with mean 0 and common variance. (This assumption is analogous to the assumption typically applied to residual error.)

JMP and SAS use the unrestricted approach. This distinction is important because many statistics textbooks use the restricted approach. Both approaches have been widely taught for 60 years. For a discussion of both approaches, see Cobb (1998, Section 13.3).

Negative Variances

Though variances are always positive, it is possible to have a situation where the unbiased estimate of the variance is negative. Negative estimates can occur in experiments when an effect is very weak or when there are very few levels corresponding to a variance component. By chance, the observed data can result in an estimate that is negative.

Unbounded Variance Components

JMP can produce negative estimates for both REML and EMS. For REML, there are two options in the Fit Model launch window: Unbounded Variance Components and Estimate Only Variance Components. The Unbounded Variance Components option is selected by default. Deselecting this option constrains variance component estimates to be nonnegative.

You should leave the Unbounded Variance Components option selected if you are interested in fixed effects. *Constraining the variance estimates to be nonnegative leads to bias in the tests for the fixed effects.*

Estimate Only Variance Components

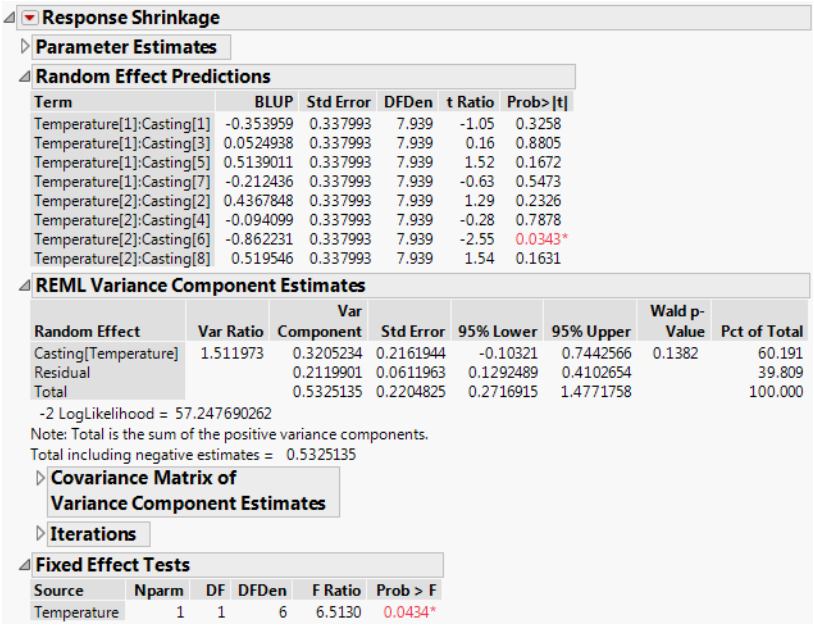
Select this option if you want to see only the REML Variance Component Estimates report. If you are interested only in variance components, you might want to constrain variance components to be nonnegative. Deselecting the Unbounded Variance Components option and selecting the Estimate Only Variance Components option might be appropriate.

Restricted Maximum Likelihood (REML) Method

Based on the fitting method selected, the Fit Least Squares report provides different analysis results and provide additional menu options for Save Columns and Profiler. In particular, the analysis of variance report is not shown because variances and degrees of freedom do not partition in the usual way. You can obtain the residual variance estimate from the REML Variance Component Estimates report. See "REML Variance Component Estimates". The Effect Tests report is replaced by the Fixed Effect Tests report where fixed effects are tested. Additional reports give predicted values for the random effects and details about the variance components.

Figure 3.62 shows the report obtained for a fit to the Investment Castings.jmp sample data using the REML method. Run the script **Model - REML**, and then fit the model. Note that Casting is a random effect and is nested within Temperature.

Figure 3.62 Fit Least Squares Report for REML Method



Response Shrinkage

Parameter Estimates

Random Effect Predictions

Term	BLUP	Std Error	DFDen	t Ratio	Prob>|t|
Temperature[1]:Casting[1]	-0.353959	0.337993	7.939	-1.05	0.3258
Temperature[1]:Casting[3]	0.0524938	0.337993	7.939	0.16	0.8805
Temperature[1]:Casting[5]	0.5139011	0.337993	7.939	1.52	0.1672
Temperature[1]:Casting[7]	-0.212436	0.337993	7.939	-0.63	0.5473
Temperature[2]:Casting[2]	0.4367848	0.337993	7.939	1.29	0.2326
Temperature[2]:Casting[4]	-0.094099	0.337993	7.939	-0.28	0.7878
Temperature[2]:Casting[6]	-0.862231	0.337993	7.939	-2.55	0.0343*
Temperature[2]:Casting[8]	0.519546	0.337993	7.939	1.54	0.1631

REML Variance Component Estimates

Random Effect	Var Ratio	Var Component	Std Error	95% Lower	95% Upper	Wald p-Value	Pct of Total
Casting[Temperature]	1.511973	0.3205234	0.2161944	-0.10321	0.7442566	0.1382	60.191
Residual		0.2119901	0.0611963	0.1292489	0.4102654		39.809
Total		0.5325135	0.2204825	0.2716915	1.4771758		100.000

-2 LogLikelihood = 57.247690262
Note: Total is the sum of the positive variance components.
Total including negative estimates = 0.5325135

Covariance Matrix of Variance Component Estimates

Iterations

Fixed Effect Tests

Source	Nparm	DF	DFDen	F Ratio	Prob > F
Temperature	1	1	6	6.5130	0.0434*

Random Effect Predictions

For each term in the model, this report gives an empirical estimate of its *best linear unbiased predictor* (BLUP) and a test for whether the corresponding coefficient is zero.

Note: The Regression Reports > Parameter Estimates option must be selected for the Random Effect Predictions report to appear.

Term The terms in the model that correspond to random effects.

BLUP The empirical estimate of the *best linear unbiased predictor* (BLUP) for each random effect. See "Best Linear Unbiased Predictors".

Std Error The standard error of the BLUP.

DFDen The denominator degrees of freedom for a test that the effect is zero. In most cases, the degrees of freedom for the *t* test is fractional.

t Ratio The *t* ratio for testing that the effect is zero. The *t* ratio is obtained by dividing the BLUP by its standard error.

Prob>|t| The *p*-value for the test.

Lower 95% The lower 95% confidence limit for the BLUP. This column appears only if you have the Regression Reports > Show All Confidence Intervals option selected or if you right-click in the report and select Columns > Lower 95%.

Upper 95% The upper 95% confidence limit for the BLUP. This column appears only if you have the Regression Reports > Show All Confidence Intervals option selected or if you right-click in the report and select Columns > Upper 95%.

Best Linear Unbiased Predictors

The term *best linear unbiased predictor* (BLUP) refers to an estimator of a random effect. Specifically, it is an estimator that, among all unbiased estimators, minimizes mean square prediction error. The Random Effect Predictions report gives estimates of the BLUPs, or *empirical* BLUPs. These are empirical because the BLUPs depend on the values of the variance components, which are unknown. The estimated values of the variance components are substituted into the formulas for the BLUPs, resulting in the estimates shown in the report.

REML Variance Component Estimates

When REML is selected as the fitting method in the Fit Model launch window, the REML Variance Component Estimates report is provided. This report contains the following columns:

Random Effect The random effects in the model.

Var Ratio The ratio of the variance component for the effect to the variance component for the residual. It compares the effect's estimated variance to the model's estimated error variance.

Var Component The estimated variance component for the effect. Note that the variance component for the Total is the sum of the positive variance components only. The sum of all variance components is given beneath the table.

Std Error The standard error for the variance component estimate.

95% Lower The lower 95% confidence limit for the variance component. See "Confidence Intervals for Variance Components".

95% Upper The upper 95% confidence limit for the variance component. See "Confidence Intervals for Variance Components".

Wald p-Value The *p*-value for the test that the covariance parameter is equal to zero. This column appears only when you have selected Unbounded Variance Components in the Fit Model launch window.

Pct of Total The ratio of the variance component for the effect to the variance component for the total as a percentage.

Sqrt Variance Component The square root of the corresponding variance component. It is an estimate of the standard deviation for the effect. This column appears only if you right-click in the report and select Columns > Sqrt Variance Component.

CV The coefficient of variation for the variance component. It is 100 times the square root of the variance component, divided by the mean response. This column appears only if you right-click in the report and select Columns > CV.

Norm KHC The Kackar-Harville correction. See "Kackar-Harville Correction". This column appears only if you right-click in the report and select Columns > Norm KHC.

Confidence Intervals for Variance Components

The method used to calculate the confidence limits depends on whether you have selected Unbounded Variance Components in the Fit Model launch window. Note that Unbounded Variance Components is selected by default.

- If Unbounded Variance Components is selected, Wald-based confidence intervals are computed. These are valid asymptotically but note that they can be unreliable with small samples.
- If Unbounded Variance Components is not selected, meaning that parameters have a lower boundary constraint of zero, a Satterthwaite approximation is used (Satterthwaite 1946).

Kackar-Harville Correction

In the REML method, the standard errors of the fixed effects are estimated using estimates of the variance components. However, if variability in these estimates is not taken into account, the standard error is underestimated. To account for the increased variability, the covariance matrix of the fixed effects is adjusted using the Kackar-Harville correction (Kackar and Harville 1984; Kenward and Roger 1997). All calculations that involve the covariance matrix of the fixed effects use this correction. These include least squares means, fixed effect tests, confidence intervals, and prediction variances. For statistical details, see "The Kackar-Harville Correction".

Norm KHC is the Frobenius (matrix) norm of the Kackar-Harville correction. In cases where the design is fairly well balanced, Norm KHC tends to be small.

Covariance Matrix of Variance Components Estimates

This report gives an estimate of the asymptotic covariance matrix for the variance components. It is the inverse of the observed Fisher information matrix.

Iterations

The estimates of σ^2 and the variance components in *G* are obtained by maximizing a residual log-likelihood function that depends on only these parameters. An iterative procedure attempts to maximize the residual log-likelihood function, or equivalently, to minimize twice the negative of the residual log-likelihood (–2LogLike). The Iterations report provides details about this procedure.

Iter Iteration number.

-2LogLike Twice the negative log-likelihood. It is the objective function. See "Likelihood, AICc, and BIC".

Norm Gradient The norm of the gradient (first derivative) of the objective function

Parameters The column labeled Parameters and the remaining columns each correspond to a random effect. The order of the columns follows the order in which random effects are listed in the REML Variance Component Estimates report. At each iteration, the value in the column is the estimate of the variance component at that point.

The convergence criterion is based on the gradient, with a default tolerance of 10^{-8}. You can change the criterion in the Fit Model launch window by selecting the option Convergence Settings > Convergence Limit and specifying the desired tolerance.

Fixed Effect Tests

When REML is used, the Effect Tests report provides tests for the fixed effects. This report contains the following columns:

Source The fixed effects in the model.

Nparm The number of parameters associated with the effect.

DF The degrees of freedom associated with the effect.

DFDen The denominator degrees of freedom. These are based on an approximation to the distribution of the statistic obtained when the covariance matrix is adjusted using the Kenward-Roger correction. See "Kackar-Harville Correction" and "Random Effects".

FRatio The computed *F* ratio.

Prob > F The *p*-value for the effect test.

REML Save Columns Options

When you use the REML method, six additional options appear in the Save Columns menu. These option names start with the adjective *Conditional*. This prefix indicates that the calculations for these columns use the predicted values for the terms associated with the random effects, rather than their expected values of zero.

Conditional Pred Formula Saves the prediction formula to a new column in the data table.

Conditional Pred Values Saves the predicted values to a new column in the data table.

Conditional Residuals Saves the residuals to a new column in the data table.

Conditional Mean CI Saves the confidence interval for the mean.

Conditional Indiv CI Saves the confidence interval for individuals.

JMP PRO **Publish Conditional Formula** Creates a conditional prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

REML Profiler Option

When you use the REML method and select Factor Profiling > Profiler, a new option, Conditional Predictions, appears on the red triangle menu next to Prediction Profiler. Note that the conditional values use the predicted values for the random effects, rather than their zero expected values.

Note: The profiler displays conditional predicted values and conditional mean confidence intervals for all combinations of factors levels. Some of these combinations might not be meaningful due to nesting.

EMS (Traditional) Model Fit Reports

Caution: The use of EMS is not recommended. REML is the recommended method.

When EMS is selected as the fitting method, four new reports are displayed. The Effect Tests report is not shown, as tests for both fixed and random effects are conducted in the Tests wrt Random Effects report.

Expected Mean Squares

The expected mean square for a model effect is a linear combination of variance components and fixed effect values, including the residual error variance. This table gives the coefficients that define each model effect's expected mean square. The rows of the matrix correspond to the effects, listed on the left. The columns correspond to the variance components, identified across the top. Each expected mean square includes the residual variance with a coefficient of one. This information is given beneath the table.

Figure 3.63 shows the Expected Mean Squares report for the Investment Castings.jmp sample data table. Run the Model - EMS script and then run the model.

Figure 3.63 Expected Mean Squares Report

Expected Mean Squares

The Mean Square per row by the Variance Component per column

EMS

	Intercept	Temperature	Casting[Temperature]&Random
Intercept	0	0	0
Temperature	0	16	4
Casting[Temperature]&Random	0	0	4

plus 1.0 times Residual Error Variance

As indicated by the table, the expected mean square for Casting[Temperature] is

$$4\sigma^2_{Casting[Temperature]} + \sigma^2_{Error}$$

Variance Component Estimates

Estimates of the variance components are obtained by equating the expected mean squares to the corresponding observed mean squares and solving. The Variance Component Estimates report gives the estimated variance components.

Component Lists the random effects.

Var Comp Est Gives the estimate of the variance component.

Percent of Total Gives the ratio of the variance component to the sum of the variance components.

CV Gives the coefficient of variation for the variance component. It is 100 times the square root of the variance component, divided by the mean response.

Note: Appears only if you right-click in the report and select Columns > CV.

Test Denominator Synthesis

For each effect to be tested, an F statistic is constructed. The denominator for this statistic is the mean square whose expectation is that of the numerator mean square under the null hypothesis. This denominator is constructed, or *synthesized*, from variance components and values associated with fixed effects.

Source Shows the effect to be tested.

MS Den Gives the estimated mean square for the denominator of the *F* test.

DF Den Gives the degrees of freedom for the synthesized denominator. These are constructed using Satterthwaite's method (Satterthwaite 1946).

Denom MS Synthesis Gives the variance components used in the denominator synthesis. The residual error variance is always part of this synthesis.

Tests wrt Random Effects

Tests for fixed and random effects are presented in this report.

Source Lists the effects to be tested. These include fixed and random effects.

SS Gives the sum of squares for the effect.

MS Num Gives the numerator mean square.

DF Num Gives the numerator degrees of freedom.

F Ratio Gives the *F* ratio for the test. It is the ratio of the numerator mean square to the denominator mean square. The denominator mean square can be obtained from the Test Denominator Synthesis report.

Prob > F Gives the *p*-value for the effect test.

Caution: Standard errors for least squares means and denominators for contrast *F* tests use the synthesized denominator. In certain situations, such as tests involving crossed effects compared at common levels, these tests might not be appropriate. Custom tests are conducted using residual error, and leverage plots are constructed using the residual error, so these also might not be appropriate.

EMS Profiler

When you use the EMS method and select Factor Profiling > Profiler, the profiler gives predictions and conditional mean confidence intervals based on the fixed-effects model. These values are not based on the predicted values for the random effects.

Models with Linear Dependencies among Model Terms

When there are linear dependencies among the columns of the matrix of predictors, several standard least squares reports are affected.

- "Singularity Details"
- "Parameter Estimates Report"
- "Effect Tests Report"
- "Examples"

Singularity Details

The linear regression model is formulated as $Y = X\beta + \varepsilon$. Here X is a matrix whose first column consists of *1*s, and whose remaining columns are the values of the non-intercept terms in the model. If the model consists of p terms, including the intercept, then X is an n by p matrix, where n is the number of observations. The parameter estimates, denoted by the vector b, are typically given by the formula:

$$b = (X'X)^{-1}X'Y$$

However, this formula presumes that $X'X^{-1}$ exists, in other words, that the $p \times p$ matrix $X'X$ is invertible, or equivalently, of full rank. Situations often arise when $X'X$ is not invertible because there are linear dependencies among the columns of X.

In such cases, the matrix $X'X$ is singular, and the Fit Least Squares report contains the Singularity Details report. This report contains a table of expressions that describe the linear dependencies. The terms involved in these linear dependencies are aliased (confounded).

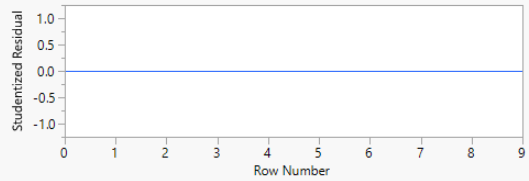
Figure 3.64 shows reports for the Reactor 8 Runs.jmp sample data table. To obtain these reports, fit a model with Percent Reacted as Y. Enter Feed Rate, Catalyst, Stir Rate, Temperature, Concentration, Catalyst*Stir Rate, Catalyst*Concentration, and Feed Rate*Catalyst as model effects.

Figure 3.64 Singularity and Parameter Estimates Report for Model with Linear Dependencies

Singularity Details

Term	Details
Concentration(3,6)	=Catalyst*Stir Rate
Stir Rate(100,120)	=Catalyst*Concentration

Studentized Residuals



Externally Studentized Residuals with 95% Simultaneous Limits (Bonferroni)

Parameter Estimates

Term		Estimate	Std Error	t Ratio	Prob>|t|
Intercept		65.875	6.125	10.76	0.0590
Feed Rate(10,15)		-4.875	6.125	-0.80	0.5720
Catalyst(1,2)		10.125	6.125	1.65	0.3463
Stir Rate(100,120)	Biased	1.875	6.125	0.31	0.8109
Temperature(140,180)		5.875	6.125	0.96	0.5133
Concentration(3,6)	Biased	-3.375	6.125	-0.55	0.6794
Catalyst*Stir Rate	Zeroed	0	0	.	.
Catalyst*Concentration	Zeroed	0	0	.	.
Feed Rate*Catalyst		0.375	6.125	0.06	0.9611

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F	
Feed Rate(10,15)	1	1	190.12500	0.6335	0.5720	
Catalyst(1,2)	1	1	820.12500	2.7326	0.3463	
Stir Rate(100,120)	1	0	0.00000	.	.	LostDFs
Temperature(140,180)	1	1	276.12500	0.9200	0.5133	
Concentration(3,6)	1	0	0.00000	.	.	LostDFs
Catalyst*Stir Rate	1	0	0.00000	.	.	LostDFs
Catalyst*Concentration	1	0	0.00000	.	.	LostDFs
Feed Rate*Catalyst	1	1	1.12500	0.0037	0.9611	

Parameter Estimates Report

When $X'X$ is singular, a generalized inverse is used to obtain estimates. This approach permits some, but not all, of the parameters involved in a linear dependency to be estimated. Parameters are estimated based on the order of entry of their associated terms into the model, so that the last terms entered are the ones whose parameters are not estimated. Estimates are given in the Parameter Estimates report, and parameters that cannot be estimated are given estimates of 0.

However, estimates of parameters for terms involved in linear dependencies are not unique. Because the associated terms are aliased, there are infinitely many vectors of estimates that satisfy the least squares criterion. In these cases, “Biased” appears to the left of these estimates in the Parameter Estimates report. “Zeroed” appears to the left of the estimates of 0 in the Parameter Estimates report for terms involved in a linear dependency whose parameters cannot be estimated. For an example, see Figure 3.64.

If there are degrees of freedom available for an estimate of error, t tests for parameters estimated using biased estimates are conducted. These tests should be interpreted with caution, though, given that the estimates are not unique.

Effect Tests Report

In a standard least squares fit, only as many parameters are estimable as there are model degrees of freedom. In conducting the tests in the Effect Tests report, each effect is considered to be the last effect entered into the model.

- If all the Model degrees of freedom are used by the other effects, an effect shows DF equal to 0. When DF equals 0, no sum of squares can be computed. Therefore, the effect cannot be tested.
- If not all Model degrees of freedom are used by the other effects, then that effect has nonzero DF. However, its DF might be less than its number of parameters (Nparm), indicating that only some of its associated parameters are testable.

An *F* test is conducted if the degrees of freedom for an effect are nonzero, assuming that there are degrees of freedom for error. Whenever DF is less than Nparm, the description LostDFs is displayed to the far right in the row corresponding to the effect (Figure 3.64). These effects have the opportunity to explain only model sums of squares that have not been attributed to the aliased effects that have absorbed their lost degrees of freedom. It follows that the sum of squares given in the Effect Tests report most likely under represents the "true" sum of squares associated with the effect. If the test is significant, its significance is meaningful. But lack of significance should be interpreted with caution.

For statistical details, see the section "Statistical Background" in the "Introduction to Statistical Modeling with SAS/STAT Software" chapter in SAS Institute Inc. (2020c).

Examples

Open the Singularity.jmp sample data table. There is a response Y, four predictors X1, X2, X3, and A, and five observations. The predictors are continuous except for A, which is nominal with four levels. Also note that there is a linear dependency among the continuous effects, namely, X3 = X1 + X2.

Non-Uniqueness of Estimates

To see that estimates are not unique when there are linear dependencies:

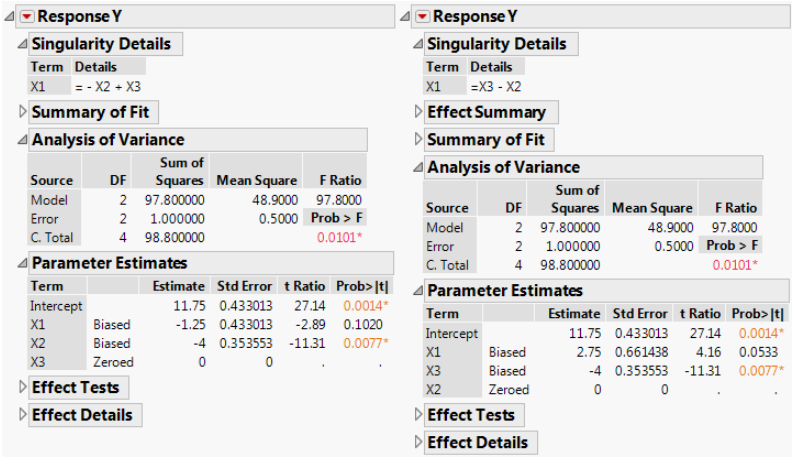
1. Select **Help > Sample Data Library** and open Singularity.jmp.
2. Run the script **Model 1**. The script opens a Fit Model launch window where the effects are entered in the order X1, X2, X3.
3. Click **Run** and leave the report window open.
4. Run the script **Model 2**. The script opens a Fit Model launch window where the effects are entered in the order X1, X3, X2.
5. Click **Run** and leave the report window open.

Compare the two reports (Figure 3.65). The Singularity Details report at the top of both reports displays the linear dependency, indicating that X1 = X3 - X2.

Now compare the Parameter Estimates reports for both models. Note, for example, that the estimate for X1 for Model 1 is –1.25 while for Model 2 it is 2.75. In both models, only two of the terms associated with effects are estimated, because there are only two model degrees of freedom. See the Analysis of Variance report. The estimates of the two terms that are estimated are labeled Biased while the remaining estimate is set to 0 and labeled Zeroed.

The Effect Tests report shows that no tests are conducted. Each row is labeled LostDFs. The reason this happens is that the effect test for any one of these effects requires it to be entered into the model last. However, the other two effects entirely account for the model sum of squares associated with the two model degrees of freedom. So there are no degrees of freedom or associated sum of squares left for the effect of interest.

Figure 3.65 Fit Least Squares Reports for Model 1 (on left) and Model 2 (on right)



Response Y

Singularity Details

Term	Details
X1	= - X2 + X3

Summary of Fit

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	2	97.800000	48.9000	97.8000
Error	2	1.000000	0.5000	Prob > F
C. Total	4	98.800000		0.0101*

Parameter Estimates

Term		Estimate	Std Error	t Ratio	Prob>|t|
Intercept		11.75	0.433013	27.14	0.0014*
X1	Biased	-1.25	0.433013	-2.89	0.1020
X2	Biased	-4	0.353553	-11.31	0.0077*
X3	Zeroed	0	0	.	.

Effect Tests

Effect Details

Response Y

Singularity Details

Term	Details
X1	=X3 - X2

Effect Summary

Summary of Fit

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	2	97.800000	48.9000	97.8000
Error	2	1.000000	0.5000	Prob > F
C. Total	4	98.800000		0.0101*

Parameter Estimates

Term		Estimate	Std Error	t Ratio	Prob>|t|
Intercept		11.75	0.433013	27.14	0.0014*
X1	Biased	2.75	0.661438	4.16	0.0533
X3	Biased	-4	0.353553	-11.31	0.0077*
X2	Zeroed	0	0	.	.

Effect Tests

Effect Details

LostDFs

To gain more insight on LostDFs, follow the steps below or run the data table script Fit Model Report:

1. Select **Help > Sample Data Library** and open Singularity.jmp.
2. Click **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Select X1 and A and click **Add**.
5. Set the **Emphasis** to **Minimal Report**.
6. Click **Run**.

Portions of the report are shown in (Figure 3.66). The Singularity Details report shows that there is a linear dependency involving X1 and the three terms associated with the effect A. For more information about how a nominal effect is coded, see "Details of Custom Test Example". The Analysis of Variance report shows that there are three model degrees of freedom. The Parameter Estimates report shows Biased estimates for the three terms X1, A[a], and A[b] and a Zeroed estimate for the fourth, A[c].

The Effect Tests report shows that X1 cannot be tested, because A must be entered first and A accounts for the three model degrees of freedom. However, A can be tested, but with only two degrees of freedom. (X1 must be entered first and it accounts for one of the model degrees of freedom.) The test for A is partial, so it must be interpreted with care.

Figure 3.66 Fit Least Squares Report for Model with X1 and A

Response Y

Singularity Details

Term	Details
Intercept	=2*X1 - A[a] - A[b] + 3*A[c]

Regression Plot

Summary of Fit

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	3	98.300000	32.7667	65.5333
Error	1	0.500000	0.5000	Prob > F
C. Total	4	98.800000		0.0905

Parameter Estimates

Term		Estimate	Std Error	t Ratio	Prob>|t|
Intercept	Biased	14.416667	0.399653	36.07	0.0176*
X1	Biased	-2.583333	0.399653	-6.46	0.0977
A[a]	Biased	-5.333333	0.471405	-11.31	0.0561
A[b]	Biased	3.1666667	0.552771	5.73	0.1100
A[c]	Zeroed	0	0	.	.

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F	
X1	1	0	0.000000	.	.	LostDFs
A	3	2	64.500000	64.5000	0.0877	LostDFs

Effect Details

Statistical Details for the Standard Least Squares Personality

- "Emphasis Rules"
- "Details of Custom Test Example"
- "Correlation of Estimates"
- "Leverage Plot Details"
- "The Kackar-Harville Correction"
- "Power Analysis"

Emphasis Rules

The default emphasis in the Fit Model launch window is based on the number of rows, n, the number of effects (k) entered in the Construct Model Effects list, and the attributes applied to effects.

- If $n > 1000$, the Emphasis is set to Minimal Report.
- If $n \leq 1000$ and $k \leq 4$, the Emphasis is set to Effect Leverage
- If $n \leq 1000$ and $k \geq 10$, the Emphasis is set to Effect Screening.
- If $n \leq 1000$ and $4 < k < 10$ and $n - k > 20$, the Emphasis is set to Effect Leverage.
- If any effect has a Random Effect attribute, the Emphasis is set to Minimal Report.
- If none of these conditions hold, the Emphasis is set to Effect Screening.

Details of Custom Test Example

In "Example of a Custom Test", you are interested in testing three contrasts using the Cholesterol.jmp sample data table. Specifically, you want to compare:

- the mean responses for treatments A and B,
- the mean response for treatments A and B combined to the mean response for the control group,
- the mean response for treatments A and B combined to the mean response for the combined control and placebo groups.

To derive the contrast coefficients that you enter into the Custom Test columns, do the following. Denote the theoretical effects for the four treatment groups as α_A, α_B, $\alpha_{Control}$, and $\alpha_{Placebo}$. These are the treatment effects, so they are constrained to sum to 0. Because the parameters associated with the indicator variables represent only the first three effects, you need to formulate your contrasts in terms of these first three effects. See "Details of Custom Test Example" and "Interpretation of Parameters".

The hypotheses that you want to test can be written in terms of model effects as follows:

- Compare treatment A to treatment B:

$$\alpha_A - \alpha_B = 0$$

- Compare treatments A and B to the control group:

$$0.5(\alpha_A + \alpha_B) - \alpha_{Control} = 0$$

- Compare treatments A and B to the control and placebo groups:

$$0.5(\alpha_A + \alpha_B) - 0.5(\alpha_{Control} + \alpha_{Placebo}) = \alpha_A + \alpha_B = 0$$

To obtain contrast coefficients for this contrast, you need to write the placebo effect in terms of the model effects. Specifically, use the fact that $\alpha_A + \alpha_B + \alpha_{Control} + \alpha_{Placebo} = 0$. Then $\alpha_{Placebo} = -\alpha_A - \alpha_B - \alpha_{Control}$. It follows that:

$$\begin{aligned} 0.5(\alpha_A + \alpha_B) - 0.5(\alpha_{Control} + \alpha_{Placebo}) &= 0.5(\alpha_A + \alpha_B) - 0.5(\alpha_{Control} - \alpha_A - \alpha_B - \alpha_{Control}) \\ &= 0.5(\alpha_A + \alpha_B) - 0.5\alpha_{Control} + 0.5(\alpha_A + \alpha_B + \alpha_{Control}) \\ &= \alpha_A + \alpha_B \\ &= 0 \end{aligned}$$

Correlation of Estimates

Consider a data set with n observations and $p - 1$ predictors. Define the matrix **X** to be the design matrix. That is, **X** is the n by p matrix whose first column consists of *1s* and whose remaining $p - 1$ columns consist of the $p - 1$ predictor values. (Nominal columns are coded in terms of indicator predictors. Each of these is a column in the matrix X.)

The estimate of the vector of regression coefficients is

$$\hat{\beta} = (X'X)^{-1}X'Y$$

where Y represents the vector of response values.

Under the usual regression assumptions, the covariance matrix of $\hat{\beta}$ is

$$Cov(\hat{\beta}) = \sigma^2(X'X)^{-1}$$

where σ^2 represents the variance of the response.

The correlation matrix for the estimates is obtained by dividing each entry in the covariance matrix by the product of the square roots of the diagonal entries. Define V to be the diagonal matrix whose entries are the square roots of the diagonal entries of the covariance matrix:

$$V = Sqrt(Diag(Cov(\hat{\beta})))$$

Then the correlation matrix for the parameter estimates is given by the following:

$$Corr(\hat{\beta}) = \sigma^2 V^{-1}(X'X)^{-1}V^{-1}$$

Leverage Plot Details

Effect leverage plots are also referred to as *partial-regression residual leverage plots* (Belsley et al. 1980) or *added variable plots* (Cook and Weisberg 1982). Sall (1990) generalized these plots to apply to any linear hypothesis.

JMP provides two types of leverage plots:

- Effect Leverage plots show observations relative to the hypothesis that the effect is not in the model, given that all other effects are in the model.
- The Whole Model leverage plot, given in the Actual by Predicted Plot report, shows the observations relative to the hypothesis of no factor effects.

In the Effect leverage plot, only one effect is hypothesized to be zero. However, in the Whole Model Actual by Predicted plot, all effects are hypothesized to be zero. Sall (1990) generalizes the idea of a leverage plot to arbitrary linear hypotheses, of which the Whole Model leverage plot is an example. The details from that paper, summarized in this section, specialize to the two types of plots found in JMP.

Construction

Suppose that the estimable hypothesis of interest is

$$L\beta = 0$$

The leverage plot characterizes this test by plotting points so that the distance of each point to the sloped regression line displays the unconstrained residual. The distance to the horizontal line at 0 displays the residual when the fit is constrained by the hypothesis. The difference between the sums of squares of these two sets of residuals is the sum of squares due to the hypothesis. This value becomes the main component of the *F* test.

The parameter estimates constrained by the hypothesis can be written

$$b_0 = b - (X'X)^{-1}L'\lambda$$

Here b is the least squares estimate

$$b = (X'X)^{-1}X'y$$

and lambda is the Lagrangian multiplier for the hypothesis constraint, calculated by

$$\lambda = (L(X'X)^{-1}L')^{-1}Lb)$$

The unconstrained and hypothesis-constrained residuals are, respectively,

$$r = y - Xb$$
$$r_0 = r + X(X'X)^{-1}L'\lambda$$

For each observation, consider the point with horizontal axis value v_x and vertical axis value v_y where:

- v_x is the constrained residual minus the unconstrained residual, r_0 - r, reflecting information left over once the constraint is applied
- v_y is the horizontal axis value plus the unconstrained residual

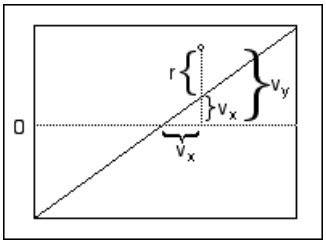
Thus, these points have x and y coordinates

$$v_x = \mathrm{X(X'X)^{-1}L'\lambda} \quad \text{and} \quad v_y = \mathrm{r} + v_x$$

These points form the basis for the leverage plot. This construction is illustrated in Figure 3.67, where the response mean is 0 and slope of the solid line is 1.

Leverage plots in JMP have a dotted horizontal line at the mean of the response, $\bar{y}$. The plotted points are given by ($v_x + \bar{y}$, v_y).

Figure 3.67 Construction of Leverage Plot



Superimposing a Test on the Leverage Plot

In simple linear regression, you can plot the confidence limits for the expected value of the response as a smooth function of the predictor variable **x**

$$\text{Upper (x)} = \mathbf{xb} + t_{\alpha/2} s \sqrt{\mathrm{x(X'X)^{-1}x'}}$$

$$\text{Lower (x)} = \mathbf{xb} - t_{\alpha/2} s \sqrt{\mathrm{x(X'X)^{-1}x'}}$$

where **x** = [1 x] is the 2-vector of predictors.

These confidence curves give a visual assessment of the significance of the corresponding hypothesis test, illustrated in Figure 3.56:

- Significant: If the slope parameter is significantly different from zero, the confidence curves cross the horizontal line at the response mean.
- Borderline: If the t test for the slope parameter is sitting right on the margin of significance, the confidence curve is asymptotic to the horizontal line at the response mean.

- Not Significant: If the slope parameter is not significantly different from zero, the confidence curve does not cross the horizontal line at the response mean.

Leverage plots mirror this thinking by displaying confidence curves. These are adjusted so that the plots are suitably centered. Denote a point on the horizontal axis by z. Define the functions

$$\text{Upper}(z) = z + \sqrt{s^2 t^2_{\alpha/2} \bar{h} + (F_\alpha / F) z^2}$$

and

$$\text{Lower}(z) = z - \sqrt{s^2 t^2_{\alpha/2} \bar{h} + (F_\alpha / F) z^2}$$

where F is the F statistic for the hypothesis and F_α is the reference value for significance level α.

And $\bar{h} = \bar{x}(X'X)^{-1}\bar{x}'$, where $\bar{x}$ is a row vector consisting of suitable middle values for the predictors, such as their means.

These functions behave in the same fashion as do the confidence curves for simple linear regression:

- If the F statistic is greater than the reference value, the confidence functions cross the horizontal axis.
- If the F statistic is equal to the reference value, the confidence functions have the horizontal axis as an asymptote.
- If the F statistic is less than the reference value, the confidence functions do not cross.

Also, it is important that Upper(z) - Lower(z) is a valid confidence interval for the predicted value at z.

The Kackar-Harville Correction

The variance matrix of the fixed effects is always modified to include a Kackar-Harville correction. The variance matrix of the BLUPs, and the covariances between the BLUPs and the fixed effects, are not Kackar-Harville corrected. The rationale for this approach is that corrections for BLUPs can be computationally and memory intensive when the random effects have many levels.

In SAS, the Kackar-Harville correction is done for both fixed effects and BLUPs only when the `DDFM=KENWARDROGER` is set.

- Standard errors for linear combinations involving only fixed effects parameters match `PROC MIXED DDFM=KENWARDROGER`. This case assumes that one has taken care to transform between the different parameterizations used by `PROC MIXED` and JMP.

- Standard errors for linear combinations involving only BLUP parameters match PROC MIXED DDFM=SATTERTHWAITE.
- Standard errors for linear combinations involving both fixed effects and BLUPS do not match PROC MIXED for any DDFM option if the data are unbalanced. However, these standard errors are between what you get with the DDFM=SATTERTHWAITE and DDFM=KENWARDROGER options. If the data are balanced, JMP matches SAS for balanced data, regardless of the DDFM option, because the Kackar-Harville correction is null.

Degrees of Freedom

The degrees of freedom for tests involving only linear combinations of fixed effect parameters are calculated using the Kenward and Roger correction. Therefore, the JMP results for these tests match PROC MIXED using the DDFM=KENWARDROGER option. If there are BLUPs in the linear combination, JMP uses a Satterthwaite approximation to get the degrees of freedom. The results then follow a pattern similar to what is described for standard errors in the preceding paragraph.

For more information about the Kackar-Harville correction and the Kenward-Roger DF approach, see Kenward and Roger (1997). The Satterthwaite method is described in detail in the MIXED Procedure chapter in SAS Institute Inc. (2020d, ch. 83).

Power Analysis

Options relating to power calculations are available only for continuous-response models. These are the contexts in which power and related test details are available:

Parameter Estimate

To obtain retrospective test details for each parameter estimate, select **Estimates > Parameter Power** from the report's red triangle menu. This option displays the least significant value, the least significant number, and the adjusted power for the 0.05 significance level test for each parameter based on current study data.

Effect or Effect Details

To obtain either prospective or retrospective details for the *F* test of a specific effect, select **Power Analysis** from the effect's red triangle menu. Keep in mind that, for the Effect Screening and Minimal Report personalities, the report for each effect is found under Effect Details. For the Effect Leverage personality, the report for an effect is found to the right of the first (Whole Model) column in the report.

LS Means Contrast

To obtain either prospective or retrospective details for a test of one or more contrasts, select **LSMeans Contrast** from the effect's red triangle menu. Define the contrasts of interest and click Done. From the Contrast red triangle menu, select **Power Analysis**.

Custom Test

To obtain either prospective or retrospective details for a custom test, select **Estimates > Custom Test** from the response's red triangle menu. Define the contrasts of interest and click **Done**. From the Custom Test red triangle menu, select **Power Analysis**.

In all cases except the first, selecting Power Analysis opens the Power Details window. You then enter information in the Power Details window to modify the calculations according to your needs.

Effect Size

The effect size, denoted by δ, is a measure of the difference between the null hypothesis and the true values of the parameters involved. The null hypothesis might be formulated in terms of a single linear contrast that is set equal to zero, or of several such contrasts. The value of δ reflects the difference between the true values of the contrasts and their hypothesized values of 0.

In general terms, the effect size is given by:

$$\delta = \sqrt{SS_{Hyp(Pop)}/n}$$

where $SS_{Hyp(Pop)}$ is the sum of squares for the hypothesis being tested given in terms of population parameters and n is the total number of observations.

When observations are available, the estimated effect size is calculated by substituting the calculated sum of squares for the hypothesis into the formula for δ.

Balanced One-Way Layout

For example, in the special case of a balanced one-way layout with k levels where the i^{th} group has mean response α_i,

$$\delta^2 = \frac{\sum(\alpha_i - \bar{\alpha})^2}{k}$$

Recall that JMP codes parameters so that, for i =1, 2,..., k-1

$$\beta_i = (\alpha_i - \bar{\alpha})$$

and

$$\beta_k = -\sum_{m=1}^{k-1} \alpha_m$$

So, in terms of these parameters, δ for a two-level balanced layout is given by:

$$\delta^2 = \frac{\beta_1^2 + (-\beta_1)^2}{2} = \beta_1^2$$

or $\delta = |\beta_1|$

Unbalanced One-Way Layout

In the case of an unbalanced one-way layout with k levels, and where the i^{th} group has mean response α_i and n_i observations, and where $n = \sum n_i$:

$$\delta^2 = \sum \frac{n_i}{n}(\alpha_i - \bar{\alpha})^2$$

Effect Size and Power

The power is the probability that the F test of a hypothesis is significant at the α significance level, when the true effect size is a specified value. If the true effect size equals δ, then the test statistic has a noncentral F distribution with noncentrality parameter

$$\lambda = (n\delta^2)/\sigma^2$$

When the null hypothesis is true (that is, when the effect size is zero), the noncentrality parameter is zero and the test statistic has a central F distribution.

The power of the test increases with λ. In particular, the power increases with sample size n and effect size δ, and decreases with error variance σ^2.

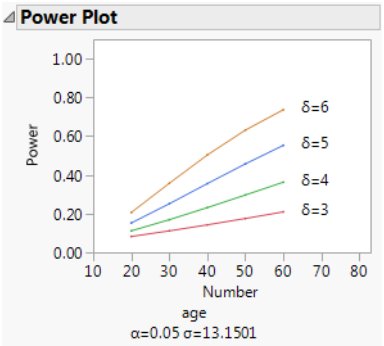
Some books, such as Cohen (1977), use a standardized effect size, $\Delta = \delta/\sigma$, rather than the raw effect size used by JMP. For the standardized effect size, the noncentrality parameter equals $\lambda = n\Delta^2$.

In the Power Details window, δ is initially set to $\sqrt{SS_{Hyp}/n}$. SS_{Hyp} is the sum of squares for the hypothesis, and n is the number of observations in the current study. SS_{Hyp} is an estimate of δ computed from the data, but such estimates are biased (Wright and O'Brien 1988). To calculate power using a sample estimate for δ, you might want to use the Adjusted Power and Confidence Interval calculation rather than the Solve for Power calculation. The adjusted power calculation uses an estimate of δ that is partially corrected for bias. See "Computations for the Adjusted Power".

Plot of Power by Sample Size

To see a plot of power by sample size, select the Power Plot option from the red triangle menu at the bottom of the Power report. JMP plots the Power and Number columns from the Power table. The plot shown in Figure 3.68 results from plotting the Power table obtained in "Example of Retrospective Power Analysis".

Figure 3.68 Plot of Power by Sample Size



The Least Significant Number (LSN)

The *least significant number* (LSN) is the smallest number of observations that leads to a significant test result, given the specified values of delta, sigma, and alpha. Recall that delta, sigma, and alpha represent, respectively, the effect size, the error standard deviation, and the significance level.

Note: LSN is *not* a recommendation of how large a sample to take because it does not take into account the probability of significance. It is computed based on specified values of delta and sigma.

The LSN has these characteristics:

- If the LSN is less than the actual sample size n, then the effect is significant.
- If the LSN is greater than n, the effect is not significant. If you believe that more data will show essentially the same structural results as does the current sample, the LSN suggests how much data you would need to achieve significance.
- If the LSN is equal to n, then the p-value is equal to the significance level alpha. The test is on the border of significance.
- The power of the test for the effect size, calculated when n = LSN, is always greater than or equal to 0.5. Note, however, that the power can be close to 0.5, which is considered low for planning purposes.

The Least Significant Value (LSV)

The LSV, or *least significant value*, is computed for single-degree-of-freedom hypothesis tests. These include tests for the significance of individual model parameters, as well as more general linear contrasts. The LSV is the smallest effect size, in absolute value, that would be significant at level alpha. The LSV gives a measure of the sensitivity of the test on the scale of the parameter, rather than on a probability scale.

The LSV has these characteristics:

- If the absolute value of the parameter estimate or contrast is greater than or equal to the LSV, then the *p*-value of the significance test is less than or equal to alpha.
- The absolute value of the parameter estimate or contrast is equal to the LSV if and only if its significance test has *p*-value equal to alpha.
- The LSV is the radius of a $1 - \alpha$ confidence interval for the parameter or linear combination of parameters. The $1 - \alpha$ confidence interval is centered at the estimate of the parameter or contrast.

Power

The power of a test is the probability that the test gives a significant result. The power is a function of the effect size δ, the significance level α, the error standard deviation σ, and the sample size n. The power is the probability that you will detect a specified effect size at a given significance level. In general, you would like to design studies that have high power of detecting differences that are of practical or scientific importance.

Power has these characteristics:

- If the true value of the parameter is in fact the hypothesized value, the power equals the significance level of the test. The significance level is usually a small value, such as 0.05. The small value is appropriate, because you want a low probability of seeing a significant result when the postulated hypothesis is true.
- If the true value of the parameter is *not* the hypothesized value, in general, you want the power to be as large as possible.
- Power increases as: sample size increases; error variance decreases; the difference between the true parameter value and the hypothesized value increases.

The Adjusted Power and Confidence Intervals

In retrospective power analysis, you typically substitute sample estimates for the population parameters involved in power calculations. This substitution causes the noncentrality parameter estimate to have a positive bias (Wright and O'Brien 1988). The adjusted power calculation is based on a form of the estimated noncentrality parameter that is partially corrected for this bias.

You can also construct a confidence interval for the adjusted power. Such confidence intervals tend to be wide. See Wright and O'Brien (1988).

Note that the adjusted power and confidence interval calculations are relevant only for the value of δ estimated from the data (the value provided by default). For other values of delta, the adjusted power and confidence interval are not provided.

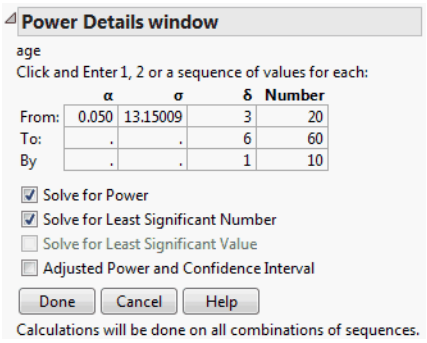
See "Computations for the Adjusted Power".

Example of Retrospective Power Analysis

This example illustrates a retrospective power analysis using the Big Class.jmp sample data table. The Power Details window (Figure 3.69) permits exploration of various quantities over ranges of values for α, σ, δ, and Number, or study size. Clicking Done replaces the window with the results of the calculations.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze > Fit Model**.
3. Select weight and click **Y**.
4. Add age, sex, and height as the effects.
5. Click **Run**.
6. Click the age red triangle and select **Power Analysis**.

Figure 3.69 Power Details Window for Age



7. Replace the δ value in the **From** box with 3, and enter 6 and 1 in the **To** and **By** boxes as shown in Figure 3.69.
8. Replace the **Number** value in the **From** box with 20, and enter 60 and 10 in the **To** and **By** boxes as shown in Figure 3.69.
9. Select **Solve for Power** and **Solve for Least Significant Number**.
10. Click **Done**.
11. The Power Details window is replaced by the Power Details report.

Figure 3.70 Power Details Report for Age

Power Details

Test age

Power

α	σ	δ	Number	Power
0.0500	13.15009	3	20	0.0828
0.0500	13.15009	3	30	0.1117
0.0500	13.15009	3	40	0.1426
0.0500	13.15009	3	50	0.1755
0.0500	13.15009	3	60	0.2099
0.0500	13.15009	4	20	0.1117
0.0500	13.15009	4	30	0.1694
0.0500	13.15009	4	40	0.2317
0.0500	13.15009	4	50	0.2969
0.0500	13.15009	4	60	0.3630
0.0500	13.15009	5	20	0.1524
0.0500	13.15009	5	30	0.2515
0.0500	13.15009	5	40	0.3554
0.0500	13.15009	5	50	0.4575
0.0500	13.15009	5	60	0.5529
0.0500	13.15009	6	20	0.2063
0.0500	13.15009	6	30	0.3572
0.0500	13.15009	6	40	0.5035
0.0500	13.15009	6	50	0.6320
0.0500	13.15009	6	60	0.7368

Least Significant Number

α	σ	δ	Number(LSN)
0.0500	13.15009	3	216.8578
0.0500	13.15009	4	123.8892
0.0500	13.15009	5	80.93391
0.0500	13.15009	6	57.6814

This analysis is a retrospective power analysis because the calculations assume a study with a structure identical to that of the Big Class.jmp sample data table. For example, the calculation of power in this example depends on the effects entered into the model and the number of participants in each age and sex grouping. Also, the value of σ was derived from the current study, though you could have replaced it with a value that would be representative of a future study.

For more information about the power results shown in Figure 3.70, see "Power". For more information about the least significant number (LSN), see "The Least Significant Number (LSN)".

Prospective Power Analysis

Prospective analysis helps you answer the question, "If differences of a specified size exist, will I detect them given my proposed sample size, alpha level, and estimate of error variance?" In a prospective power analysis, you must provide estimates of the group means and sample sizes in a data table. You must also provide an estimate of the error standard deviation σ in the Power Details window.

Equal Group Sizes

Consider a situation where you are comparing the means of three independent groups. To obtain sample sizes to achieve a given power, select **DOE > Design Diagnostics > Sample Size and Power** and then select **k Sample Means**. Next to Std Dev, enter your estimate of the error standard deviation. In the Prospective Means list, enter means that reflect the smallest differences that you want to detect. If, for example, you want to detect a difference of 8 units between any two means, enter the extreme values of the means (for example, 40, 40, and 48). Because the power is based on deviations from the grand mean, you can enter only values that reflect the desired differences (for example, 0, 0, and 8).

If you click **Continue**, you obtain a graph of power versus sample size. If instead you specify either power or sample size in the Sample Size window, the other quantity is computed and displayed in the Sample Size window. In particular, if you specify power, the sample size that is provided is the total required sample size. The k Sample Means calculation assumes equal group sizes. For three groups, you would divide the sample size by 3 to obtain the individual group sizes. For more information about k Sample Means, see the *Design of Experiments Guide*.

Unequal Group Sizes

Suppose that you want to design a study that uses groups of different sizes. You need to plan an experiment to study two treatments that reportedly reduce bacterial counts. You want to compare the effect of these treatments with results from a control group that receives no treatment. You also want to detect a difference of at least 8 units between the means of either treatment group and the control group. But the control group must be twice as large as either treatment group. The two treatment groups also must be equal in size. Previous studies suggest that the error standard deviation is on the order of 5 or 6.

To obtain a prospective power analysis for this situation, create a data table containing some basic information, as shown in the Bacteria.jmp sample data table.

Figure 3.71 Bacteria.jmp Data Table

	Group	Group Means	Relative Sizes
1	Control	40	2
2	Treatment A	40	1
3	Treatment B	48	1

- The Group column identifies the groups.
- The Means column reflects the smallest difference among the columns that it is important to detect. Here, it is assumed that the control group has a mean of about 40. You want the test to be significant if either treatment group has a mean that is at least 8 units higher than the mean of the control group. For this reason, you assign a mean of 48 to one of the two treatment groups. Set the mean of the other treatment group equal to that of the control group. (Alternatively, you could assign the control group and one of the treatment groups

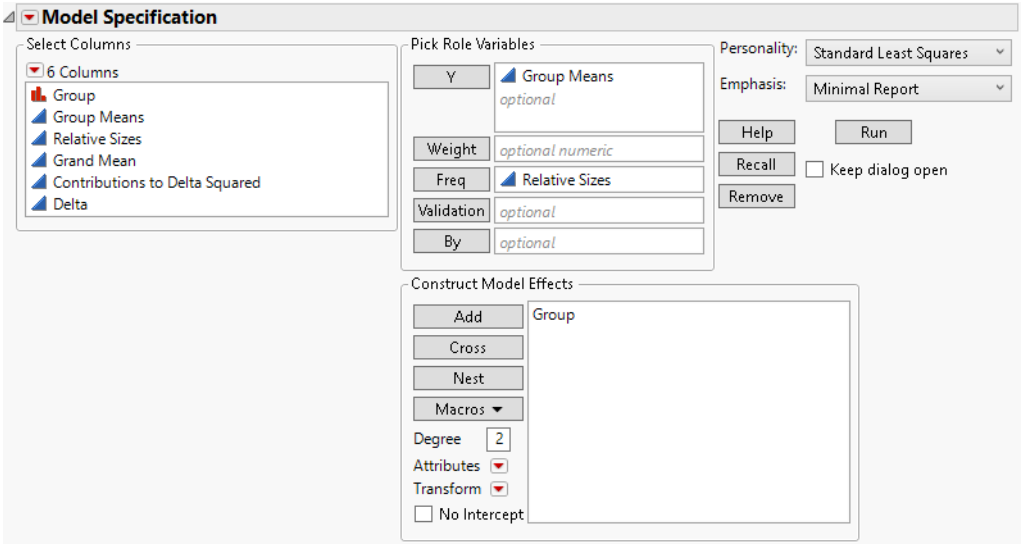
means of 0 and the remaining treatment group a mean of 8.) Note that the differences in the group means are population values.

- The Relative Sizes column shows the desired relative sizes of the treatment groups. This column indicates that the control group needs to be twice as large as each of the treatment groups. (Alternatively, you could start out with an initial guess for the treatment sizes that respects the relative size criterion.)

Note: The Relative Sizes column must be assigned the role of a Freq (frequency). See the symbol to the right of the column name in the Columns panel.

Next, use Fit Model to fit a one-way analysis of variance model (Figure 3.72). Note that Relative Sizes is declared as **Freq** in the launch window. Also, the Minimal Report emphasis option is selected.

Figure 3.72 Fit Model Launch Window for Bacteria Study

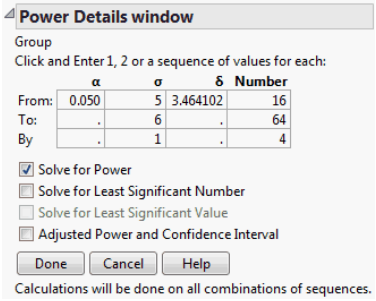


Click **Run** to obtain the Fit Least Squares report. The report shows Root Mean Square Error and Sum of Squares for Error as 0.0, because you specified a data table with no error variation within the groups. You must enter a proposed range of values for the error variation to obtain the power analysis. Specifically, you have information that the error variation will be about 5 but might be as large as 6.

1. Click the disclosure icon next to Effect Details to open this report.
2. Click the Group red triangle and select **Power Analysis**.
3. To explore the range of error variation suspected by the scientist, under σ, enter 5 in the first box and 6 in the second box (Figure 3.73).

4. Note that δ is entered as 3.464102. This is the effect size that corresponds to the specified difference in the group means. The data table contains three hidden columns that illustrate the calculation of the effect size. See "Unbalanced One-Way Layout".
5. To explore power over a range of study sizes, under **Number**, enter 16 in the first box, 64 in the second box, and an increment of 4 in the third box (Figure 3.73).
6. Select **Solve for Power**.
7. Click **Done**.

Figure 3.73 Power Details Window for Bacteria Study



The Power Details report, shown in Figure 3.74, replaces the Power Details window. This report gives power calculations for $\alpha = 0.05$, for all combinations of $\sigma = 5$ and 6, and sample sizes of 16 to 64 in increments of size 4. When σ is 5, to obtain about 90% power, you need a total sample size of about 32. You need 16 participants in the control group and 8 in each of the treatment groups. On the other hand, if σ is 6, then a total of 44 participants is required.

Figure 3.74 Power Details Report for Bacteria Study

Power Details

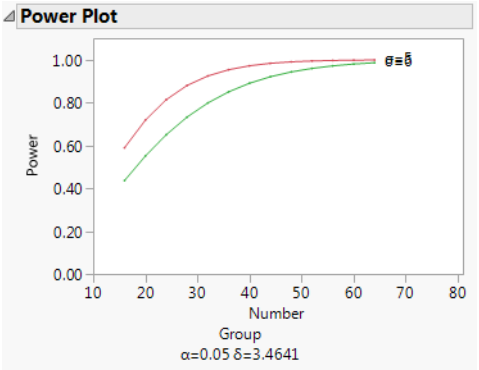
Test Group

Power

α	σ	δ	Number	Power
0.0500	5	3.464102	16	0.5894
0.0500	5	3.464102	20	0.7184
0.0500	5	3.464102	24	0.8134
0.0500	5	3.464102	28	0.8799
0.0500	5	3.464102	32	0.9246
0.0500	5	3.464102	36	0.9537
0.0500	5	3.464102	40	0.9721
0.0500	5	3.464102	44	0.9835
0.0500	5	3.464102	48	0.9903
0.0500	5	3.464102	52	0.9944
0.0500	5	3.464102	56	0.9968
0.0500	5	3.464102	60	0.9982
0.0500	5	3.464102	64	0.9990
0.0500	6	3.464102	16	0.4361
0.0500	6	3.464102	20	0.5510
0.0500	6	3.464102	24	0.6501
0.0500	6	3.464102	28	0.7323
0.0500	6	3.464102	32	0.7986
0.0500	6	3.464102	36	0.8506
0.0500	6	3.464102	40	0.8906
0.0500	6	3.464102	44	0.9208
0.0500	6	3.464102	48	0.9433
0.0500	6	3.464102	52	0.9598
0.0500	6	3.464102	56	0.9717
0.0500	6	3.464102	60	0.9803
0.0500	6	3.464102	64	0.9864

Click the arrow at the bottom of the table in the Power Details report to obtain a plot of power versus sample size for the two values of σ, shown in Figure 3.75. Here, the red markers correspond to σ = 5 and the green correspond to σ = 6.

Figure 3.75 Power Plot for Bacteria Study



Chapter 4

Standard Least Squares Examples

Analyze Common Classes of Models

This chapter provides examples with instructional material for several standard least squares topics. These include analysis of variance, analysis of covariance, a response surface model, a split plot design, estimation of random effect parameters, a knotted spline fit, and the identification of active factors using a script that provides a Bayesian approach.

Contents

One-Way Analysis of Variance Example 221
Analysis of Covariance with Equal Slopes Example 224
Analysis of Covariance with Unequal Slopes Example. 227
Response Surface Model Example 229
Fit the Full Response Surface Model 230
Reduce the Model 231
Examine the Response Surface Report. 232
Find the Critical Point Using the Prediction Profiler. 233
View the Surface Using the Contour Profiler 235
Split Plot Design Example 237
Estimation of Random Effect Parameters Example 240
Knotted Spline Effect Example 242
Bayes Plot for Active Factors Example. 244

One-Way Analysis of Variance Example

In a one-way analysis of variance, a different mean is fit to each of the different groups, as identified by a nominal variable. To specify the model for JMP, select a continuous response column and a nominal effect column. This example uses the data in Drug.jmp.

1. Select **Help > Sample Data Library** and open Drug.jmp.
2. Select **Analyze > Fit Model**.
3. Select y and click **Y**.
4. Select Drug and click **Add**.
5. Click **Run**.

In this example, Drug has three levels, a, d, and f. The standard least squares fitting method translates this specification into a linear model: The nominal variables define a sequence of indicator variables, which assume only the values 1, 0, and –1. The linear model is specified as follows:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \varepsilon_i$$

where:

- y_i is the observed response for the i^{th} observation
- x_{1i} is the value of the first indicator variable for the i^{th} observation
- x_{2i} is the value of the second indicator variable for the i^{th} observation
- β_0, β_1, and β_2 are parameters for the intercept, the first indicator variable, and the second indicator variable, respectively
- ε_i are the independent and normally distributed error terms

The first indicator variable, x_1, is defined as follows. Note that Drug = a contributes a value 1, Drug = d contributes a value 0, and Drug = f contributes a value –1 to the indicator variable:

$$x_{1i} = \begin{cases} 1, & \text{if Drug = a} \\ 0, & \text{if Drug = d} \\ -1, & \text{if Drug = f} \end{cases}$$

The second indicator variable, x_2, is given the following values:

$$x_{2i} = \begin{cases} 0, & \text{if Drug = a} \\ 1 & \text{if Drug = d} \\ -1, & \text{if Drug = f} \end{cases}$$

The estimates of the means for the three levels in terms of this parameterization are defined as follows:

$$\mu_a = \beta_0 + \beta_1$$

$$\mu_d = \beta_0 + \beta_2$$

$$\mu_f = \beta_0 - \beta_1 - \beta_2$$

Solving for β_i yields the following:

$$\beta_0 = \frac{(\mu_a + \mu_d + \mu_f)}{3} = \mu \quad \text{(the average over levels)}$$

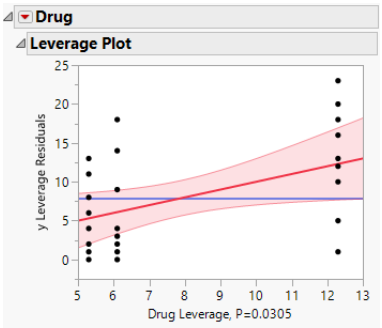
$$\beta_1 = \mu_a - \mu$$

$$\beta_2 = \mu_d - \mu$$

Therefore, if regressor variables are coded as indicators for each level minus the indicator for the last level, then the parameter for a level is interpreted as the difference between that level's response and the average response across all levels. See the appendix "Statistical Details" for additional information about the interpretation of the parameters for nominal factors.

Figure 4.1 shows the Leverage Plot and the LS Means Table for the Drug effect. Figure 4.2 shows the Parameter Estimates and the Effect Tests reports for the one-way analysis of the drug data.

Figure 4.1 Leverage Plot and LS Means Table for Drug



Least Squares Means Table

Level	Least Sq Mean	Std Error	Mean
a	5.300000	1.9197801	5.3000
d	6.100000	1.9197801	6.1000
f	12.300000	1.9197801	12.3000

Figure 4.2 Parameter Estimates and Effect Tests for Drug.jmp

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	7.9	1.108386	7.13	<.0001*
Drug[a]	-2.6	1.567494	-1.66	0.1088
Drug[d]	-1.8	1.567494	-1.15	0.2609

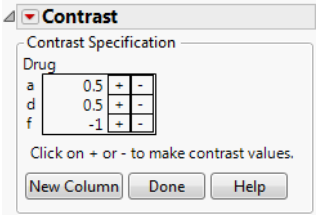
Effect Tests

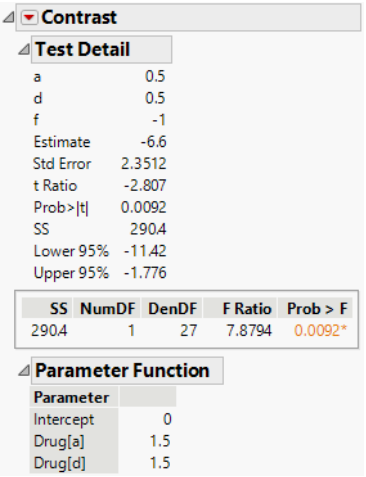
Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F
Drug	2	2	293.60000	3.9831	0.0305*

The Drug effect can be studied in more detail by using a contrast of the least squares means:

1. Click the Drug red triangle and select **LSMeans Contrast**.
2. Click the + boxes for drugs a and d, and the - box for drug f to define the contrast that compares the average of drugs a and d to f (shown in Figure 4.3).
3. Click **Done**.

Figure 4.3 Contrast Example for the Drug Experiment





The Contrast report shows that the LSMean for drug f is significantly different from the average of the LSMeans of the other two drugs.

Analysis of Covariance with Equal Slopes Example

An analysis of variance model with a continuous regressor term is called an *analysis of covariance*. In the Drug.jmp sample data table, the column x is a covariate.

The covariate adds an additional term, x_{3i}, to the model equation. The model analysis of covariance model is written this way:

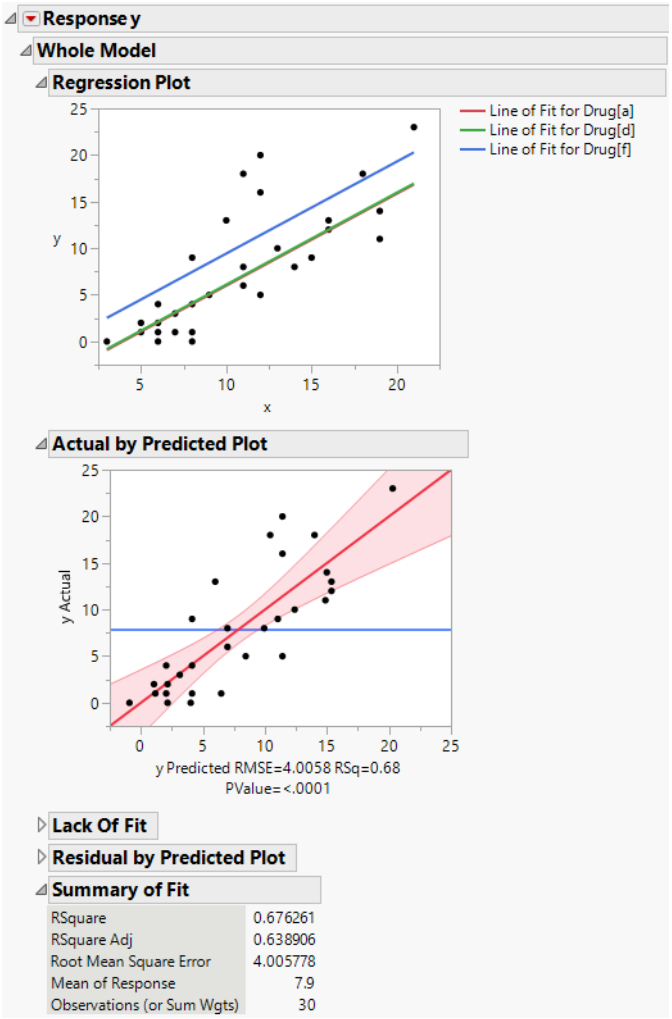
$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \varepsilon_i$$

There are two model effects: one is a nominal main effect involving two parameters, and the other is continuous covariate associated with one parameter.

1. Select **Help > Sample Data Library** and open Drug.jmp.
2. Select **Analyze > Fit Model**.
3. Select y and click **Y**.
4. Select both Drug and x and click **Add**.
5. Click **Run**.

The Regression Plot in the report shows that you have fit a model with equal slopes (Figure 4.4). Compared to the main effects model (Drug effect only), RSquare increases from 22.8% to 67.6%. The Root Mean Square Error decreases from 6.07 to 4.0. As shown in Figure 4.4, the *F* test significance probability for the whole model decreases from 0.03 to less than 0.0001.

Figure 4.4 Analysis of Covariance for Drug Data



The drug data table contains replicated observations. For example, rows 1 and 9 both have Drug = a and x = 11. When fitting models, replicated observations can be used to construct a *pure error* estimate of variation. Another estimate of error can be constructed for unspecified functional forms of covariates, or interactions of nominal effects. These estimates form the basis for a lack of fit test. If the lack of fit error is significant, this indicates that there is some effect in your data not explained by your model.

The Lack of Fit report shows the results of this test for the drug data. The lack of fit error is not significant, as seen by the Prob > F value of 0.7507.

The covariate, x, accounts for much of the variation in the response previously accounted for by the Drug variable. Thus, even though the model is fit with much less error, the Drug effect is no longer significant. The significance of Drug observed in the main effects model appears to be explained by the covariate.

The least squares means in the covariance model differ from the ordinary means. This is because they are adjusted for the effect of x, the covariate, on the response, y. The least squares means are values predicted for each of the three levels of Drug, when the covariate, x, is held at some neutral value. The neutral value is chosen to be the mean of the covariate, which is 10.7333.

The least squares means are calculated using the parameter estimates given in the Parameter Estimates report:

Prediction Expression: `-2.696 - 1.185*Drug[a] - 1.0761*Drug[d] + 0.98718*x`

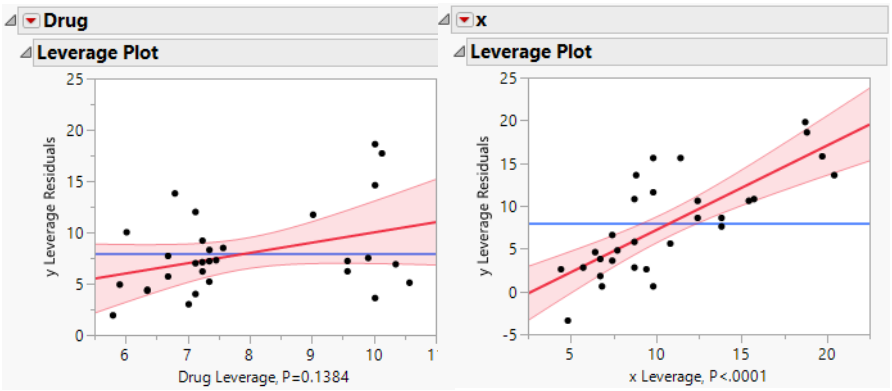
For a: `-2.696 - 1.185*(1) -1.0761*(0) + 0.98718*(10.7333) = 6.71`

For d: `-2.696 - 1.185*(0) -1.0761*(1) + 0.98718*(10.7333) = 6.82`

For f: `-2.696 - 1.185*(-1) -1.0761*(-1) + 0.98718*(10.7333) = 10.16`

Figure 4.5 shows a leverage plot for each effect. Because the covariate is significant, the leverage values for Drug are dispersed somewhat from their least squares means.

Figure 4.5 Comparison of Leverage Plots for Drug Test Data



Analysis of Covariance with Unequal Slopes Example

Continuing with the Drug.jmp sample data table, this example fits a model where the slope for the covariate depends on the level of Drug. After fitting the model, the example compares the least square means for the levels of Drug at a specific value of the covariate x.

1. Select **Help > Sample Data Library** and open Drug.jmp.
2. Select **Analyze > Fit Model**.
3. Select y and click **Y**.
4. Select both Drug and x and click **Macros > Factorial to Degree**.

   This adds terms up to the degree specified in the **Degree** box to the model. The default value for **Degree** is 2. Thus, the main effects of Drug and x, and their interaction, Drug*x, are added to the model effects list.
5. Click **Run**.

This specification adds two columns to the linear model (call them x_{4i} and x_{5i}) that allow the slopes for the covariate to differ by Drug level. The new variables are formed by multiplying the indicator variables for Drug by the covariate values, giving the following formula:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \beta_4 x_{4i} + \beta_5 x_{5i} + \varepsilon_i$$

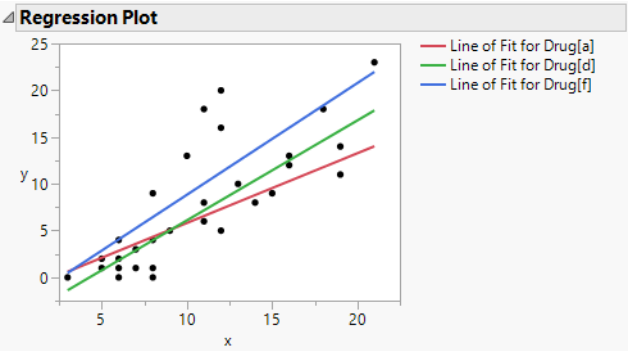
Table 4.1, shows the coding for this model. The mean of X is 10.7333. It is used in centering continuous terms.

Table 4.1 Coding of Analysis of Covariance with Separate Slopes

Regressor	Effect	Values
X_1	Drug[a]	+1 if a, 0 if d, –1 if f
X_2	Drug[d]	0 if a, +1 if d, –1 if f
X_3	X	the values of X
X_4	Drug[a]*(X - 10.733)	X – 10.7333 if a, 0 if d, –(X – 10.7333) if f
X_5	Drug[d]*(X - 10.733)	0 if a, X – 10.7333 if d, –(X – 10.7333) if f

A portion of the report is shown in Figure 4.6. The Regression Plot shows fitted lines with different slopes. The Effect Tests report gives a p-value for the interaction of 0.56. This is not significant, indicating the model does not need to include different slopes.

Figure 4.6 Plot with Interaction



Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F
Drug	2	2	52.05637	1.5713	0.2284
x	1	1	564.56753	34.0821	<.0001*
Drug*x	2	2	19.64465	0.5930	0.5606

Perform a Spotlight Analysis

You now want to compare the least square means for the levels of Drug at a specific value of the covariate x. This type of comparison in an analysis of covariance model is sometimes referred to as *spotlight analysis*. For more information about spotlight analysis, see Spiller et al. (2013).

1. Click the Response y red triangle and select **Estimates > Multiple Comparisons**.
2. In the Multiple Comparisons window, select **User-Defined Estimates**.
3. Select all three values below **Choose Drug Levels**.
4. In the first box below x, enter 12.5.
5. Click **Add Estimates**.

   This adds comparisons of the three levels of Drug at x = 12.5.
6. Click **OK**.

   The User-Defined Estimates report shows least square means estimates for each level of Drug with the covariate x set to 12.5. The red triangle next to Multiple Comparisons for User-Defined Estimates contains options that enable you to test for differences among the estimates.

Figure 4.7 User-Defined Estimates Report

Multiple Comparisons for User-Defined Estimates

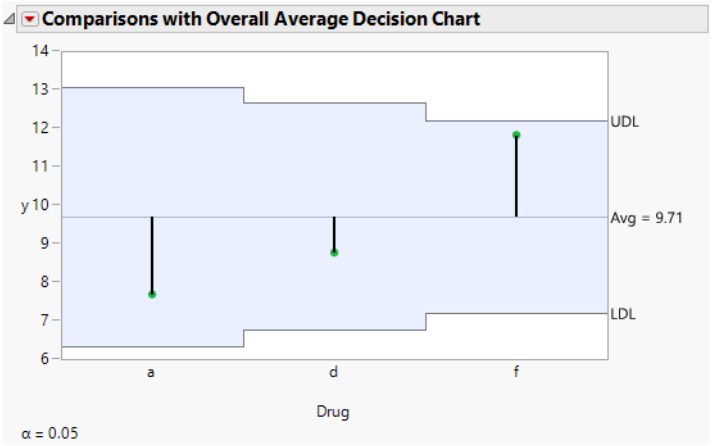
User-Defined Estimates

x = 12.5

Drug	Estimate	Std Error	DF	Lower 95%	Upper 95%
a	7.684713	1.5772057	24	4.4295208	10.939906
d	8.771371	1.4401229	24	5.7991033	11.743639
f	11.822214	1.2943345	24	9.1508392	14.493590

7. Click the red triangle next to Multiple Comparisons for User-Defined Estimates and select **Comparisons with Overall Average**.

Figure 4.8 Comparisons with Overall Average Decision Chart



The Comparisons with Overall Average option creates an analysis of means (ANOM) chart for differences between the average and the three least squares means. From the ANOM chart, you conclude that there is not a significant effect of Drug on the response at x = 12.5.

Response Surface Model Example

This example fits a response surface model. Your objective is to minimize the response.

- "Fit the Full Response Surface Model"
- "Reduce the Model"
- "Examine the Response Surface Report"
- "Find the Critical Point Using the Prediction Profiler"
- "View the Surface Using the Contour Profiler"

Fit the Full Response Surface Model

1. Select **Help > Sample Data Library** and open Design Experiment/Custom RSM.jmp.
2. Select **Analyze > Fit Model**.

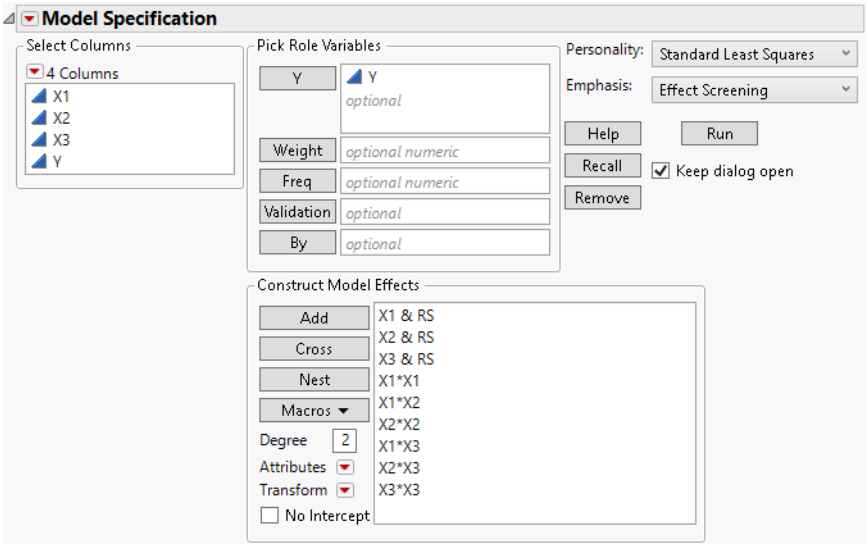
   Because the data table contains a **Model** script, the Model Specification window is filled out as specified in the **Model** script. Note the following:

   – Main effects appear in the **Construct Model Effects** list with a **&RS** suffix, indicating that the Response Surface macro has been applied.

   – The effects are those for a full response surface in the three predictors X1, X2, and X3.

   – Because the model contains terms with the **&RS** suffix, the analysis results will include a Response Surface report.

Figure 4.9 Fit Model Launch Window for the Response Surface Analysis

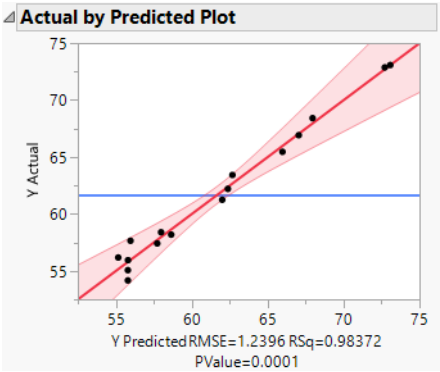


3. Click **Run**.

Reduce the Model

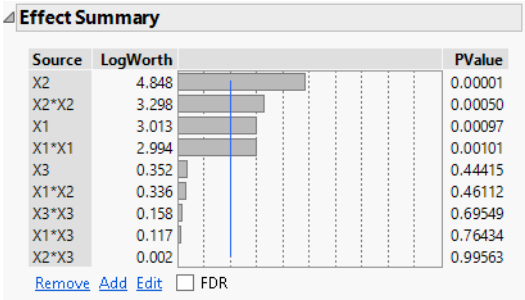
The Actual by Predicted Plot shows that the model is significant. There is no evidence of lack of fit.

Figure 4.10 Actual by Predicted Plot for Full Model



The Effect Summary report suggests that a number of effects are not significant. In particular, X2*X3 is the least significant effect with a PValue of 0.99563. Next, you will systematically reduce the model using the Effect Summary report interactively.

Figure 4.11 Effect Summary Report



1. In the Effect Summary report, click X2*X3 and click **Remove**.

   The model updates.

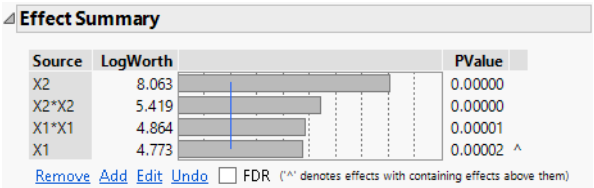
   The PValue column in the Effect Summary report indicates that X1*X3 is not significant.
2. Click X1*X3 and click **Remove**.

   The PValue for X3*X3 indicates that it is not significant.
3. Click X3*X3 and click **Remove**.
4. Click X1*X2 and click **Remove**.

Notice that X3 is not significant. It is not contained in any higher-order effects, so you can remove it without violating the Effect Heredity principle. See "Effect Heredity".

5. Click X3 and click **Remove**.

Figure 4.12 Effect Summary Report After Reducing Model

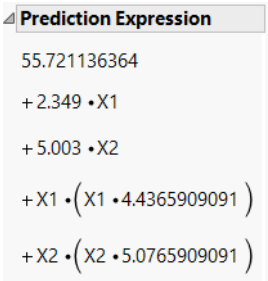


All remaining effects are significant.

Examine the Response Surface Report

1. Click the Response Y red triangle and select **Estimates > Show Prediction Expression**.

Figure 4.13 Prediction Expression

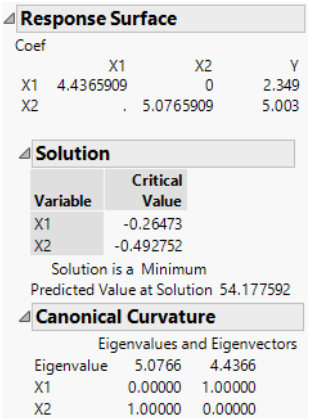


Tip: The Prediction Expression is the prediction formula. You can also obtain this formula by selecting **Save Columns > Prediction Formula**.

In the following steps, you can refer to the prediction expression to see the model coefficients.

2. Open the Response Surface report and then the Canonical Curvature report.

Figure 4.14 Response Surface Report



Response Surface

Coef

	X1	X2	Y
X1	4.4365909	0	2.349
X2	.	5.0765909	5.003

Solution

Variable	Critical Value
X1	-0.26473
X2	-0.492752

Solution is a Minimum

Predicted Value at Solution 54.177592

Canonical Curvature

Eigenvalues and Eigenvectors

Eigenvalue	5.0766	4.4366
X1	0.00000	1.00000
X2	1.00000	0.00000

The first table gives the second-order model coefficients in matrix form. The coefficient of X1*X1 is 4.4365909, the coefficient of X2*X2 is 5.0765909, and the coefficient of X1*X2 is 0. The coefficients of the linear effects, 2.349 for X1 and 5.003 for X2, are given in the column labeled Y.

The Solution report shows the critical values. These are the values where a maximum, a minimum, or a saddle point occur. In this example, the Solution report indicates that the response surface achieves a minimum of 54.18 at the critical value, where X1 = -0.265 and X2 = -0.493.

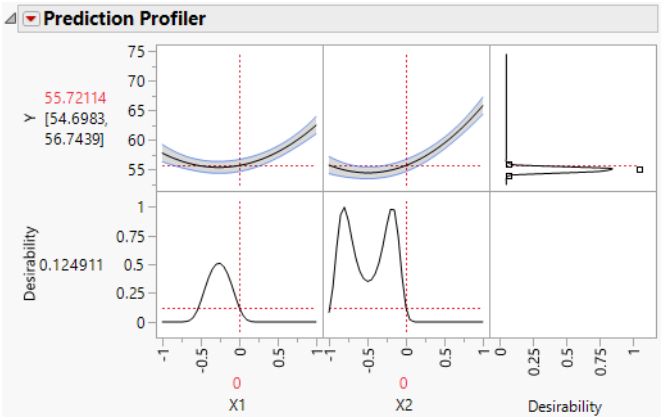
The Canonical Curvature report shows the eigenstructure of the matrix of second-order parameter estimates. The eigenstructure is useful for identifying the shape and orientation of the curvature. See "Canonical Curvature Report".

In this example, both eigenvalues are positive, which indicates that the surface achieves a minimum. The direction of greatest curvature corresponds to the largest eigenvalue (5.0766). That direction is defined by the corresponding eigenvector components. For the first direction, X2, with an eigenvector value of 1.00, determines the direction. The second direction is determined by X1, also with an eigenvector value of 1.00.

Find the Critical Point Using the Prediction Profiler

The Prediction Profiler report shows the quadratic behavior of the response surface along traces for X1 and X2. Because the Response Limits column property is set for Y, the profiler also shows desirability functions.

Figure 4.15 Prediction Profiler with Match Target as Goal

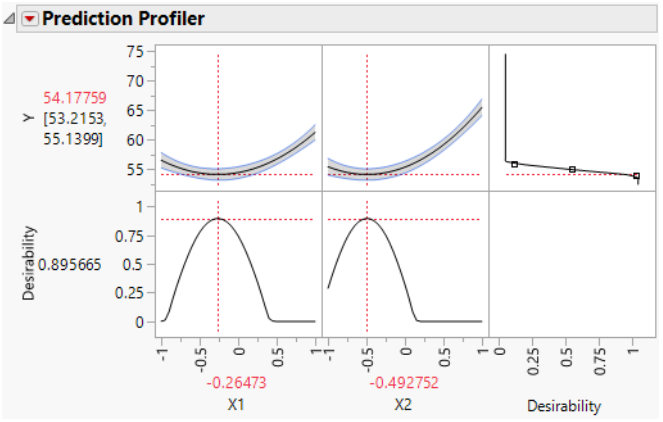


The goal for the Response Limits column property is set to Match Target. For this example, you are interested in minimizing Y, not matching a target. Therefore, you must change the setting:

1. Press Ctrl and click in the top right cell of the Prediction Profiler.
2. In the Response Goal dialog, select **Minimize** from the list of options.
3. Click **OK**.

   The desirability function now reflects your goal of minimizing Y.
4. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 4.16 Prediction Profiler with Minimize as Goal and Desirability Maximized



Settings within the design region that minimize Y appear under the profiler. Note that these are precisely the Critical Values given in the Solution report.

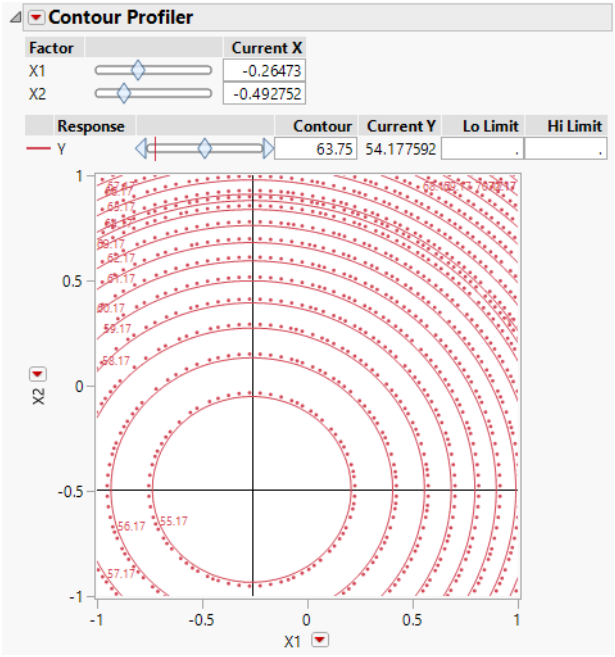
View the Surface Using the Contour Profiler

The Contour Profiler shows contours of the response surface. It gives an alternate profiler visualization of the predicted response in the area of the critical point.

1. Click the Response Y red triangle and select **Factor Profiling > Contour Profiler**.
2. Click the Contour Profiler red triangle and select **Contour Grid**.
3. For **Increment**, type 1.
4. Click **OK**.

   The contours are plotted at one unit intervals. See Figure 4.17.
5. Click the Prediction Profiler red triangle and select **Factor Settings > Link Profilers**.

Figure 4.17 Contour Profiler with Crosshairs at Critical Point



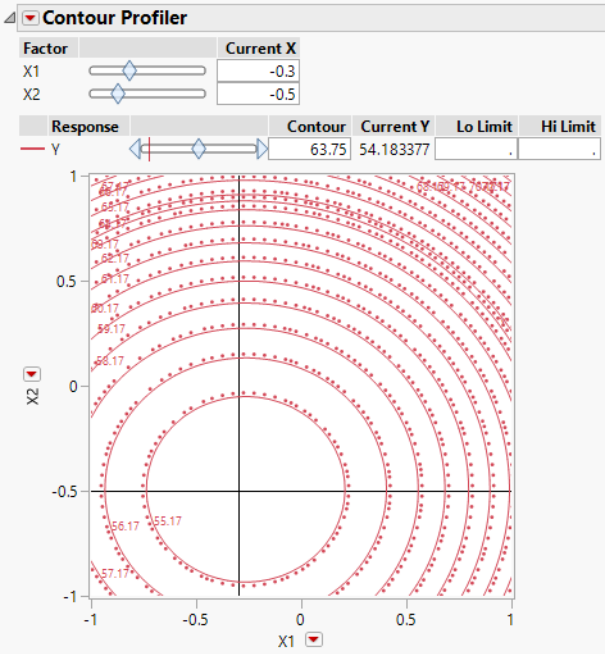
Linking the Contour Profiler to the Prediction Profiler links the **Current X** values in the Contour Profiler to the X values shown in the Prediction Profiler. The X values in the Prediction Profiler give the critical point where Y is minimized. The crosshairs in the Contour Profiler show the critical point. Notice that the **Current Y** value is 54.177592, the predicted minimum value according to the Prediction Profiler.

Often, it is not possible to set your process factors to exactly the values that optimize the response. The Contour Profiler can help you identify alternate settings of the process factors. In the next steps, suppose that you can set your process to X1 and X2 values with

only one decimal place precision, and that your process settings can vary by one decimal place in either direction of those settings.

6. In the Contour Profiler report, under **Current X**, type -0.3 next to X1 and -0.5 next to X2.

Figure 4.18 Contour Profiler Showing X1 = -0.3 and X2 = -0.5



The crosshairs are well within the innermost contour and the Current Y (the predict Y value at the Current X settings) is 54.183377, only slightly different from the predicted minimum of 54.177592.

7. In the Contour Profiler, click and drag the crosshairs to explore Current X values within a 0.1 unit radius of X1 = -0.3 and X2 = -0.5.

   The predicted Y values are all below 54.4. In fact, if the settings wander to any point within the innermost contour, the predicted Y is less than the contour value of 55.17.

Split Plot Design Example

Levels of random effects are randomly selected from a larger population of levels. For the purpose of inference, the distribution of a random effect is assumed to be normal, with mean zero and some variance (called a *variance component*).

In a sense, every model has at least one random effect, which is the effect that makes up the residual error. The individual observations are assumed to be randomly selected from a much larger population, and the error term is assumed to have a mean of zero and variance σ^2.

The most common random effects model is the repeated measures or split plot model. Table 4.2 lists the types of effects in a split plot model. In these models, the experiment has two layers. Some effects are applied on the whole plots or subjects of the experiment. Then these plots are divided or the subjects are measured at different times and other effects are applied within those subunits. The effects describing the whole plots or subjects are whole plot effects, and the subplots or repeated measures are subplot effects. Usually, the subunit effect is omitted from the model and absorbed as residual error.

Table 4.2 Types of Effects in a Split Plot Model

Split Plot Model	Type of Effect	Repeated Measures Model
whole plot treatment	fixed effect	across subjects treatment
whole plot ID	random effect	subject ID
subplot treatment	fixed effect	within subject treatment
subplot ID	random effect	repeated measures ID

Each of these cases can be treated as a layered model, and there are several traditional ways to fit them in a fair way. The situation is treated as two different experiments:

1. The whole plot experiment has whole plot or subjects as the experimental unit to form its error term.
2. Subplot treatment has individual measurements for the experimental units to form its error term (left as residual error).

The older, traditional way to test whole plots is to do any one of the following:

- Take means across the measurements and fit these means to the whole plot effects.
- Form an *F*-ratio by dividing the whole plot mean squares by the whole plot ID mean squares.

- Organize the data so that the split or repeated measures form different columns. Fit a MANOVA model, and use the univariate statistics.

These approaches work if the structure is simple and the data are complete and balanced. However, there is a more general model that works for any structure of random effects. This more generalized model is called the *mixed model*, because it has both fixed and random effects.

The most common type of layered design is a balanced split plot, often in the form of repeated measures across time. One experimental unit for some of the effects is subdivided (sometimes by time period) and other effects are applied to these subunits.

Consider the data in the Animals.jmp sample data table (the data are fictional). The study collected information about differences in the seasonal hunting habits of foxes and coyotes. Each season for one year, three foxes and three coyotes were marked and observed periodically. The average number of miles that they wandered from their dens during different seasons of the year was recorded (rounded to the nearest mile). The model is defined by the following aspects:

- The continuous response variable called miles
- The species effect with values fox or coyote
- The season effect with values fall, winter, spring, and summer
- An animal identification code called subject, with nominal values 1, 2, and 3 for both foxes and coyotes

There are two layers to the model:

1. The top layer is the between-subject layer, in which the effect of being a fox or coyote (species effect) is tested with respect to the variation from subject to subject.
2. The bottom layer is the within-subject layer, in which the repeated-measures factor for the four seasons (season effect) is tested with respect to the variation from season to season within a subject. The within-subject variability is reflected in the residual error.

The season effect can use the residual error for the denominator of its *F* statistics. However, the between-subject variability is not measured by residual error and must be captured with the subject within species (subject[species]) effect in the model. The *F* statistic for the between-subject effect species uses this nested effect instead of residual error for its *F* ratio denominator.

Note: JMP Pro users can construct this model using the Mixed Model personality.

To specify the split plot model for this data, follow these steps:

1. Select **Help > Sample Data Library** and open Animals.jmp.
2. Select **Analyze > Fit Model**.

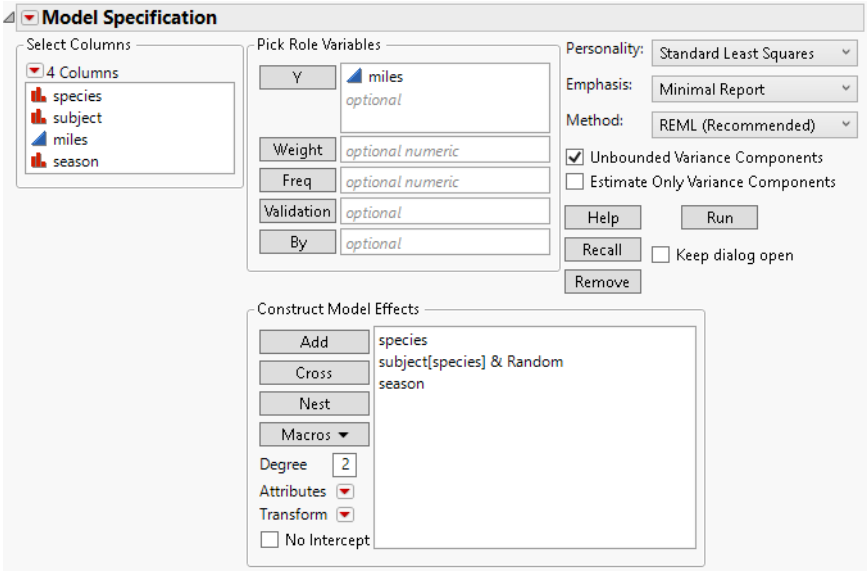
3. Select miles and click **Y**.
4. Select species and subject and click **Add**.
5. In the Select Columns list, select species.
6. In the Construct Model Effects list, select subject.
7. Click **Nest.**

   This adds the subject within species (subject[species]) effect to the model.
8. Select the nested effect subject[species].
9. Select **Attributes > Random Effect**.

   This nested effect is now identified as an error term for the species effect and appears as subject[species]&Random.
10. In the Select Columns list, select season and click **Add**.

    When you define an effect as random using the **Attributes** menu, the Method options (**REML** and **EMS**) appear at the top right of the dialog. The **REML** option is selected as the default. The completed launch window is shown in Figure 4.19.

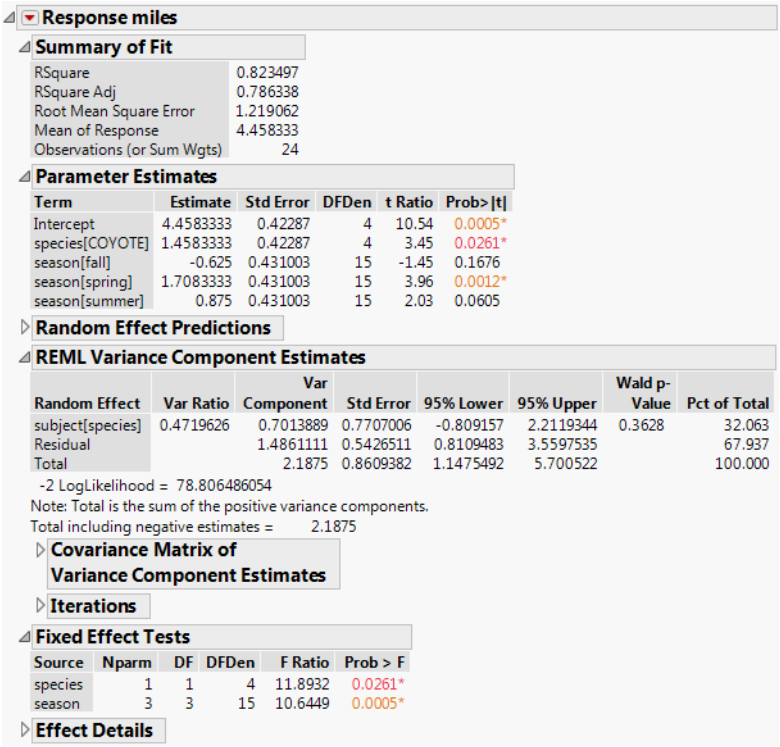
Figure 4.19 Fit Model Dialog



11. Click **Run**.

    The report is shown in Figure 4.20. Both fixed effects, species and season, are significant. The REML Variance Component Estimates report gives estimates of the subject within species and residual variances.

Figure 4.20 Partial Report of REML Analysis



Response miles

Summary of Fit

RSquare	0.823497
RSquare Adj	0.786338
Root Mean Square Error	1.219062
Mean of Response	4.458333
Observations (or Sum Wgts)	24

Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|
Intercept	4.4583333	0.42287	4	10.54	0.0005*
species[COYOTE]	1.4583333	0.42287	4	3.45	0.0261*
season[fall]	-0.625	0.431003	15	-1.45	0.1676
season[spring]	1.7083333	0.431003	15	3.96	0.0012*
season[summer]	0.875	0.431003	15	2.03	0.0605

Random Effect Predictions

REML Variance Component Estimates

Random Effect	Var Ratio	Var Component	Std Error	95% Lower	95% Upper	Wald p-Value	Pct of Total
subject[species]	0.4719626	0.7013889	0.7707006	-0.809157	2.2119344	0.3628	32.063
Residual		1.4861111	0.5426511	0.8109483	3.5597535		67.937
Total		2.1875	0.8609382	1.1475492	5.700522		100.000

-2 LogLikelihood = 78.806486054
Note: Total is the sum of the positive variance components.
Total including negative estimates = 2.1875

Covariance Matrix of Variance Component Estimates

Iterations

Fixed Effect Tests

Source	Nparm	DF	DFDen	F Ratio	Prob > F
species	1	1	4	11.8932	0.0261*
season	3	3	15	10.6449	0.0005*

Effect Details

Estimation of Random Effect Parameters Example

Random effects have a dual character. In one characterization, they represent residual error, such as the error associated with a whole-plot experimental unit. In another characterization, they are like fixed effects, associating a parameter with each level of the random effect. As parameters, you have extra information about them—they are derived from a normal distribution with mean zero and the variance estimated by the variance component. The effect of this extra information is that the estimates of the parameters are shrunken toward zero.

The parameter estimates associated with random effects are called *BLUPs* (Best Linear Unbiased Predictors). Some researchers consider these BLUPs as parameters of interest, and others consider them uninteresting by-products of the methodology.

BLUP parameter estimates are used to estimate random-effect least squares means, which are therefore also shrunken toward the grand mean. The degree of shrinkage depends on the variance of the effect and the number of observations per level in the effect. With large variance estimates, there is little shrinkage. If the variance component is small, then more shrinkage takes place. If the variance component is zero, the effect levels are shrunk to exactly zero. It is even possible to obtain highly negative variance components where the shrinkage is reversed. You can consider fixed effects as a special case of random effects where the variance component is very large.

The REML method balances the information about each individual level with the information about the variances across levels. If the number of observations per level is large, the estimates shrink less. If there are very few observations per level, the estimates shrink more. If there are infinitely many observations, there is no shrinkage and the estimates are identical to fixed effects.

Suppose that you have batting averages for different baseball players. The variance component for the batting performance across players describes how much variation is typical between players in their batting averages. Suppose that the player plays only a few times and that the batting average is unusually small or large. Then you tend not to trust that estimate, because it is based on only a few at-bats. But if you mix that estimate with the grand mean, that is, shrink the estimate toward the grand mean, you would trust the estimate more. For players who have a long batting record, you would shrink much less than those with a short record.

You can explore this behavior yourself.

1. Select **Help > Sample Data Library** and open Baseball.jmp.
2. Select **Analyze > Fit Model**.
3. Select Batting and click **Y, Response**.
4. Select Player and click **Add**.
5. Select Player in the Construct Model Effects box, and select **Random Effect** from the Attributes list.
6. Click **Run**.

Table 4.3 shows the Least Squares Means from the Player report for a REML (Recommended) fit. Also shown are the Method of Moment estimates, obtained using the EMS Method. The Method of Moment estimates are the ordinary Player means. Note that the REML estimate for Suarez, who has only three at-bats, is shrunken more toward the grand mean than estimates for other players with more at-bats.

Table 4.3 Comparison of Estimates between Method of Moments and REML

	Method of Moments	REML	N
Variance Component	0.01765	0.019648	
Anderson	0.29500000	0.29640407	6
Jones	0.20227273	0.20389793	11
Mitchell	0.32333333	0.32426295	6
Rodriguez	0.55000000	0.54713393	6
Smith	0.35681818	0.35702094	11
Suarez	0.55000000	0.54436227	3
Least Squares Means	same as ordinary means	shrunken from means	

Knotted Spline Effect Example

Use the **Knotted Spline Effect** option to fit a segmentation of smooth polynomials to a specified effect. When you select this attribute, a window appears, enabling you to specify the number of knot points. (Knotted splines are implemented only for main-effect continuous terms.)

JMP follows the advice in the literature in positioning the points. The knotted spline is also referred to as a Stone spline or a Stone-Koo spline. See Stone and Koo (1985). If there are 100 or fewer points, the first and last knots are the fifth point inside the minimum and maximum, respectively. Otherwise, the first and last knots are placed at the 0.05 and 0.95 quantiles if there are 5 or fewer knots, or the 0.025 and 0.975 quantiles for more than 5 knots. The default number of knots is 5 unless there are 30 or fewer points. In that case, the default is 3 knots.

Knotted splines have the following properties in contrast to smoothing splines:

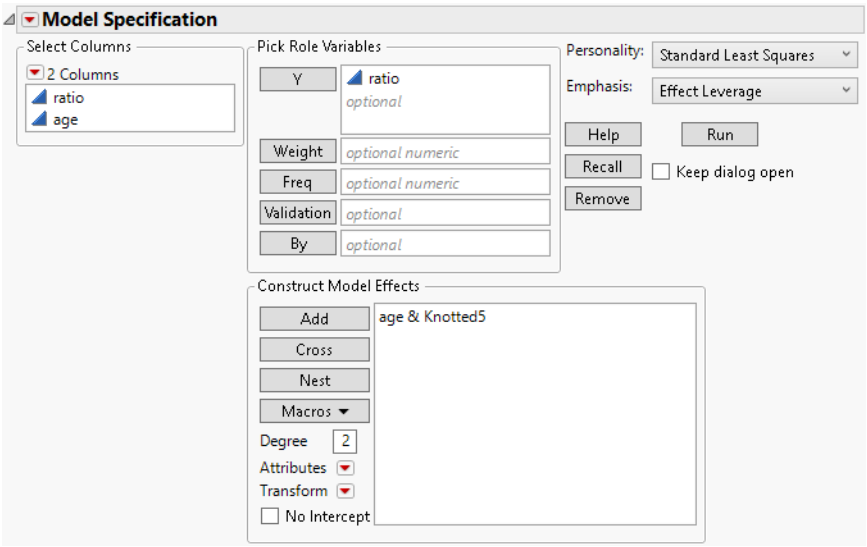
- Knotted splines work inside general models with many terms, whereas smoothing splines are for bivariate regressions.
- The regression basis is not a function of the response.
- Knotted splines are parsimonious, adding only $k - 2$ terms for curvature for k knot points.
- Knotted splines are conservative compared to pure polynomials in the sense that the extrapolation outside the range of the data is a straight line, rather than a polynomial.
- There is an easy test for curvature.

Example Using the Knotted Spline Effect to Test for Curvature

To test for curvature, follow these steps:

1. Select **Help > Sample Data Library** and open Growth.jmp.
2. Select **Analyze > Fit Model**.
3. Select the ratio column and click **Y**.
4. Select the age column and click **Add**.
5. Select age in the **Effects** pane and select **Attributes > Knotted Spline Effect**.
6. For the number of knots, type 5 and click **OK**.

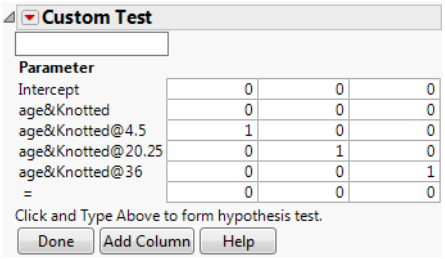
Figure 4.21 Fit Model Launch Window



7. Click **Run**.
8. Click the Response ratio red triangle and select **Estimates > Custom Test**.

   In the Custom Test report, notice that there is only one column. You want three columns.
9. Click the **Add Column** button twice to produce a total of three columns.
10. In the first column, type 1 for **age&Knotted@4.5**.
11. In the second column, type 1 for **age&Knotted@20.25**.
12. In the third column, type 1 for **age&Knotted@36**.

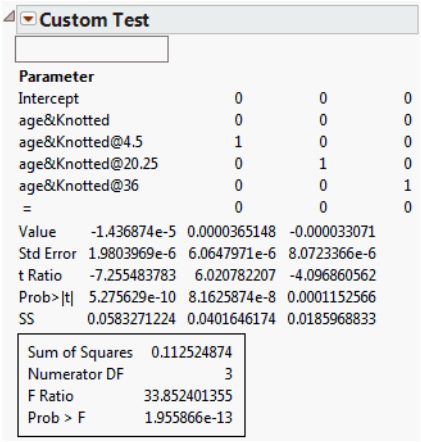
Figure 4.22 Values for the Custom Test for Curvature



13. Click **Done**.

    The small Prob > F value in the Custom Test report indicates that there is curvature.

Figure 4.23 Curvature Report

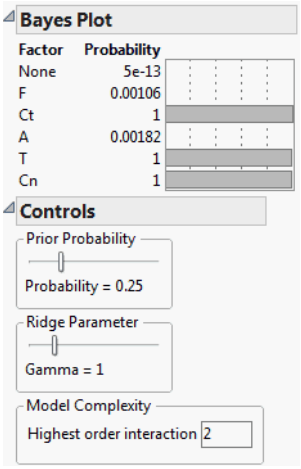


Bayes Plot for Active Factors Example

Suppose that you conduct an experimental design and want to determine which factors are active. You can address this in several ways using JMP. This example illustrates a script that can be used to identify active factors using a Bayesian approach.

1. Select **Help > Sample Data Library** and open Reactor.jmp.
2. In the Samples > Scripts folder, open the BayesPlotforFactors.jsl sample script.
3. Select **Edit > Run Script**.
4. Select Y and click **Y, Response**.
5. Select F, Ct, A, T, and Cn and click **X, Factor**.
6. Click **OK**.

Figure 4.24 Bayes Plot for Factor Activity



The Model Complexity indicates that the highest order interaction to consider is two. Therefore, all possible models that include up to second-order interactions are constructed. Based on the value assigned to Prior Probability, a posterior probability is computed for each of the possible models. The probability for a factor is the sum of the probabilities for each of the models where it was involved.

This approach identifies Ct, T, and Cn as active factors, and A and F as inactive.

If the ridge parameter were zero (not allowed), all the models would be fit by least squares. As the ridge parameter increases, the parameter estimates for any model shrink toward zero. For more information on the ridge parameter, and why it cannot be zero, see Box and Meyer (1993).

Chapter 5

Stepwise Regression Models

Find a Model Using Variable Selection

Stepwise regression is an approach to selecting a subset of effects for a regression model. It can be useful in the following situations:

- There is little theory to guide the selection of terms for a model.
- You want to interactively explore which predictors seem to provide a good fit.
- You want to improve a model's prediction performance by reducing the variance caused by estimating unnecessary terms.

For categorical predictors, you can do the following:

- Choose from among various rules to determine how associated terms enter the model.
- Enforce effect heredity.

The Stepwise platform also enables you to explore all possible models and to conduct model averaging.

Contents

Overview of Stepwise Regression 249
Example Using Stepwise Regression 249
The Stepwise Report 251
Stepwise Platform Options 251
Stepwise Regression Control Panel 252
Current Estimates Report 259
Step History Report 260
Models with Crossed, Interaction, or Polynomial Terms 261
Example of the Combine Rule 261
Models with Nominal and Ordinal Effects 263
Construction of Hierarchical Terms 263
Example of a Model with a Nominal Term 264
Example of the Restrict Rule for Hierarchical Terms 269
Performing Binary and Ordinal Logistic Stepwise Regression 271
Example Using Logistic Stepwise Regression 271
The All Possible Models Option 273
Example Using the All Possible Models Option 273
The Model Averaging Option 275
Example Using the Model Averaging Option 275
Validation Options in Stepwise Regression 276
Validation Set with Two or Three Values in Stepwise Regression 276
K-Fold Cross Validation in Stepwise Regression 279

Overview of Stepwise Regression

In JMP, stepwise regression is a personality of the Fit Model platform. The **Stepwise** feature computes estimates that are the same as those of other least squares platforms, but it facilitates searching and selecting among many models.

The approach has side effects of which you need to be aware. The significance levels on the statistics for selected models violate the standard statistical assumptions because the model has been selected rather than tested within a fixed model. On the positive side, the approach has been helpful for 30 years in reducing the number of terms. The book *Subset Selection in Regression* (Miller 1990) brings statistical sense to model selection statistics.

This chapter uses the term *significance probability* in a mechanical way to represent that the calculation would be valid in a fixed model, recognizing that the true significance probability could be nowhere near the reported one.

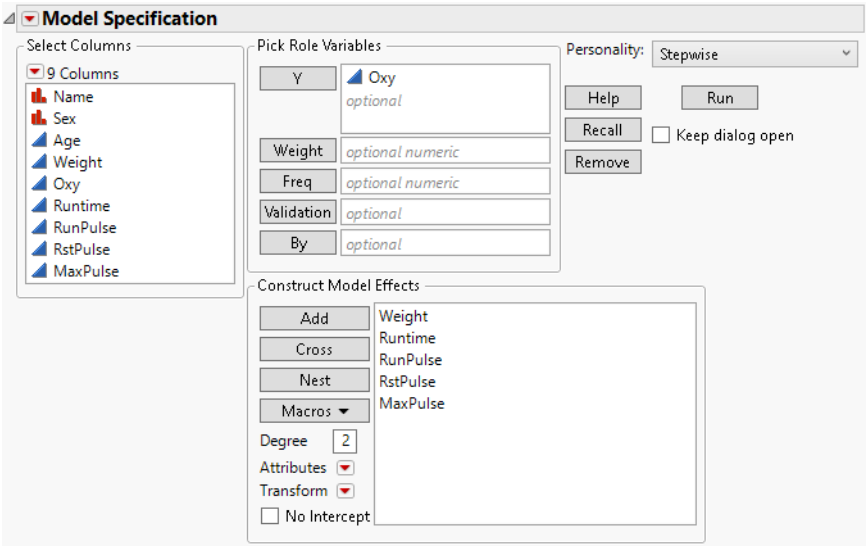
Example Using Stepwise Regression

The Fitness.jmp data table contains the results of an aerobic fitness study. Aerobic fitness can be evaluated using a special test that measures the oxygen uptake of a person running on a treadmill for a prescribed distance. However, it would be more economical to find a formula that uses simpler measurements that evaluate fitness and predict oxygen uptake. To identify such an equation, measurements of age, weight, run time, and pulse were taken for 31 participants who ran 1.5 miles.

Note: For purposes of illustration, certain values of MaxPulse and RunPulse have been changed from data reported by Rawlings (1988, p.105).

1. Select **Help > Sample Data Library** and open Fitness.jmp.
2. Select **Analyze > Fit Model**.
3. Select Oxy and click **Y**.
4. Select Weight, Runtime, RunPulse, RstPulse, MaxPulse, and click **Add**.
5. For **Personality**, select **Stepwise**.

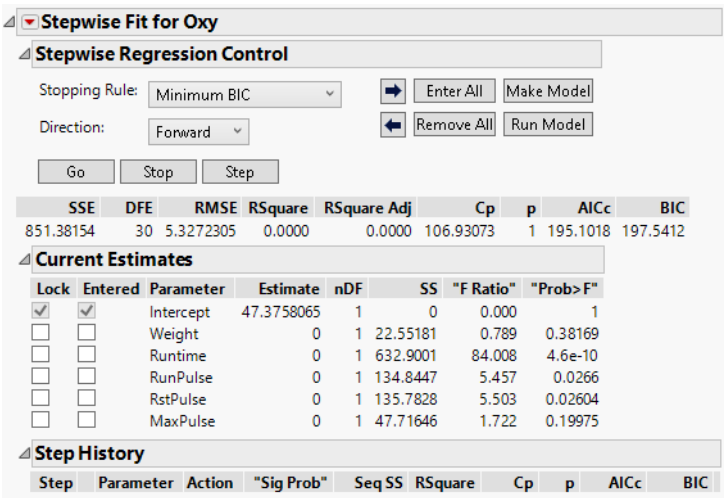
Figure 5.1 Completed Fit Model Launch Window



Validation is available only in JMP Pro.

6. Click **Run**.

Figure 5.2 Stepwise Report Window



To find a good oxygen uptake prediction equation, you need to compare many different regression models. Use the options in the Stepwise report window to search through models with combinations of effects and choose the model that you want.

The Stepwise Report

The Stepwise report window contains the following elements:

Platform options The Stepwise Fit red triangle menu contains options that affect all of the variables. See "Stepwise Platform Options".

Stepwise Regression Control Controls that limit regressor effect probabilities, determine the method of selecting effects, start or stop the selection process, and create a model. See "Stepwise Regression Control Panel".

Current Estimates A report that enables you to add, remove, and lock in model effects. See "Current Estimates Report".

Step History A report that records the effect of adding a term to the model. See "Step History Report".

Stepwise Platform Options

The Stepwise Fit red triangle menu contains the following options:

K-Fold Crossvalidation (Available only for continuous responses.) Performs K-Fold cross validation in the selection process. When selected, this option enables the Max K-Fold RSquare stopping rule ("Stepwise Regression Control Panel"). For more information about validation, see "Validation Options in Stepwise Regression".

All Possible Models (Available only for continuous responses.) Fits all possible models up to specified limits and shows the best models for each number of terms. Enter values for the maximum number of terms to fit in any one model. Also enter values for the maximum number of best model results to show for each number of terms in the model. Categorical variables are represented using indicator variables. See "Models with Nominal and Ordinal Effects". You can restrict the models that appear to those that satisfy strong effect heredity. See "The All Possible Models Option".

Model Averaging (Available only for continuous responses.) Enables you to average the fits for a number of models, instead of selecting a single best model. See "The Model Averaging Option".

Plot Criterion History Creates a plot of AICc and BIC versus the number of parameters. The Criterion History plot contains two shaded zones. Define the minimum AICc value as V^{best}. The green zone is defined by the range [V^{best}, V^{best}+4]. The yellow zone is defined by the range (V^{best}+4, V^{best}+10].

Plot RSquare History (Available only for continuous responses.) Creates a plot of training and validation R-square versus the number of parameters.

Clear History Clears and resets the step history.

JMP PRO **Export Model with Validation** (Available only when you have entered a Validation column in the Stepwise launch window.) Adds the Validation column to the Model Specification window when you select Make Model. Runs the model with the Validation column when you select Run Model.

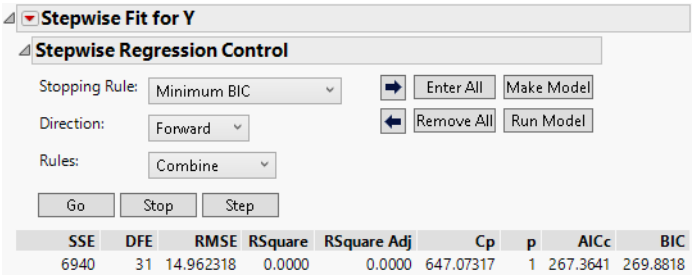
This option is selected by default.

Model Dialog Shows the completed Fit Model launch window for the current model.

Stepwise Regression Control Panel

Use the Stepwise Regression Control panel to limit regressor effect probabilities, determine the method of selecting effects, begin or stop the selection process, and run a model. A note appears beneath the Go button to indicate whether you have excluded or missing rows.

Figure 5.3 Stepwise Regression Control Panel



Stopping Rule

The Stopping Rule determines which model is selected. For all stopping rules other than P-value Threshold, only the Forward and Backward directions are allowed. The only stopping rules that use validation are Max Validation RSquare and Max K-Fold RSquare. See "Validation Options in Stepwise Regression".

P-value Threshold Uses *p*-values (significance levels) to enter and remove effects from the model. Two other options appear when you choose P-value Threshold:

Prob to Enter Specifies the maximum *p*-value that an effect must have to be entered into the model during a forward step.

Prob to Leave Specifies the minimum *p*-value that an effect must have to be removed from the model during a backward step.

Minimum AICc Uses the minimum corrected Akaike Information Criterion to choose the best model. For more details, see "Likelihood, AICc, and BIC".

Minimum BIC Uses the minimum Bayesian Information Criterion to choose the best model. For more details, see "Likelihood, AICc, and BIC".

JMP PRO **Max Validation RSquare** Uses the maximum R-square from the validation set to choose the best model. This is available only when you use a validation column with two or three distinct values. For more information about validation, see "Validation Set with Two or Three Values in Stepwise Regression".

Max K-Fold RSquare Uses the maximum RSquare from K-fold cross validation to choose the best model. You can access the Max K-Fold RSquare stopping rule by selecting this option from the Stepwise red triangle menu. JMP Pro users can access the option by using a validation set with four or more values. When you select this option, you are asked to specify the number of folds. For more information about validation, see "K-Fold Cross Validation in Stepwise Regression".

Direction

The Direction you choose controls how effects enter and leave the model. Select one of the following options:

Forward Enters the term with the smallest *p*-value. If the P-value Threshold stopping rule is selected, that term must be significant at the level specified by **Prob to Enter**. See "Forward Selection Example".

Backward Removes the term with the largest *p*-value. If the P-value Threshold stopping rule is selected, that term must not be significant at the level specified in **Prob to Leave**. See "Backward Selection Example".

Note: When Backward is selected as the Direction, you must click Enter All before clicking Go or Step.

Mixed Available only when the P-value Stopping Rule is selected. It alternates the forward and backward steps. It includes the most significant term that satisfies **Prob to Enter** and removes the least significant term satisfying **Prob to Leave**. It continues removing terms until the remaining terms are significant and then it changes to the forward direction.

Go, Stop, Step Buttons

The Go, Stop, and Step buttons enable you to control how terms are entered or removed from the model.

Note: All Stopping Rules consider only models defined by *p*-value entry (Forward direction) or removal (Backward direction). Stopping rules do not consider all possible models.

Go Automates the process of entering (Forward direction) or removing (Backward direction) terms. Among the fitted models, the model that is considered best based on the selected Stopping Rule is listed last. Except for the P-value Threshold stopping rule, the model selected as Best is one that overlooks local dips in the behavior of the stopping rule statistic. The button to the right the Best model selects it for the Make Model and Run Model options, but you are free to change this selection.

- For P-value Threshold, the best model is based on the Prob to Enter and Prob to Leave criteria. See "P-value Threshold".
- For Min AICc and Min BIC, the automatic fits continue until a Best model is found. The Best model is one with a minimum AICc or BIC that can be followed by as many as ten models with larger values of AICc or BIC, respectively. This model is designated by the terms Best in the Parameter column and Specific in the Action column.
- For Max Validation RSquare (JMP Pro only) and Max K-Fold RSquare, the automatic fits continue until a Best model is found. The Best model is one with an RSquare Validation or RSquare K-Fold value that can be followed by as many as ten models with smaller values of RSquare Validation or RSquare K-Fold, respectively. This model is designated by the terms Best in the Parameter column and Specific in the Action column.

Stop Stops the automatic selection process started by the Go button.

Step Enters terms one-by-one in the Forward direction or removes them one-by one in the Backward direction. At any point, you can select a model by clicking its button on the right in the Step History report. The selection of model terms is updated in the Current Estimates report. This is the model that is used once you click Make Model or Run Model.

Rules

Note: Appears only if your model contains related terms. When you have a nominal or ordinal variable, related terms are constructed and appear in the Current Estimates table.

Use Rules to change the rules that are applied when there is a hierarchy of terms in the model. A hierarchy can occur in the following ways:

- A hierarchy results when a variable is a component of another variable. For example, if your model contains variables A, B, and A*B, then A and B are *precedent* terms to A*B in the hierarchy.
- A hierarchy also results when you include nominal or ordinal variables. A term that is above another term in the tree structure is a *precedent* term. See "Construction of Hierarchical Terms".

Select one of the following options:

Combine Calculates *p*-values for two separate tests when considering entry for a term that has precedents. The first *p*-value, p_1, is calculated by grouping the term with its precedent terms and testing the group's significance probability for entry as a joint *F* test. The second *p*-value, p_2, is the result of testing the term's significance probability for entry after the precedent terms have already entered into the model. The final significance probability for entry for the term that has precedents is max(p_1, p_2).

Tip: The Combine rule avoids including non-significant interaction terms, whose precedent terms can have particularly strong effects. In this scenario, the strong main effects might make the group's significance probability for entry, p_1, very small. However, the second test finds that the interaction by itself is not significant. As a result, p_2 is large and is used as the final significance probability for entry.

Caution: The degrees of freedom value for a term that has precedents depends on which of the two significance probabilities for entry is larger. The test used for the final significance probability for entry determines the degrees of freedom, nDF, in the Current Estimates table. Therefore, if p_1 is used, nDF equals the number of terms in the group for the joint test, and if p_2 is used, nDF equals 1.

The Combine option is the default rule. See "Models with Crossed, Interaction, or Polynomial Terms".

Restrict Restricts the terms that have precedents so that they cannot be entered until their precedents are entered. See "Models with Nominal and Ordinal Effects" and "Example of the Restrict Rule for Hierarchical Terms".

No Rules Gives the selection routine complete freedom to choose terms, regardless of whether the routine breaks a hierarchy or not.

Whole Effects Enters only whole effects, when terms involving that effect are significant. This rule applies only when categorical variables with more than two levels are entered as possible model effects. See "Rules".

Buttons

The Stepwise Control Panel contains the following buttons:

Go Automates the selection process to completion.

Stop Stops the selection process.

Step Increments the selection process one step at a time.

Arrow buttons Step forward and backward one step in the selection process.

Enter All Enters all unlocked terms into the model.

Remove All Removes all unlocked terms from the model.

Make Model Creates a model for the Fit Model window from the model currently showing in the Current Estimates table. In cases where there are nominal or ordinal terms, **Make Model** creates temporary transform columns that contain terms that are needed for the model.

Run Model Runs the model currently showing in the Current Estimates table. In cases where there are nominal or ordinal terms, **Run Model** creates temporary transform columns that contain terms that are needed for the model.

Statistics

The following statistics appear below the Stepwise Regression Control panel.

SSE Sum of squared errors for the current model.

DFE Error degrees of freedom for the current model.

RMSE Root mean square error (residual) for the current model.

RSquare Proportion of the variation in the response that can be attributed to terms in the model rather than to random error.

RSquare Adj Adjusts R^2 to make it more comparable over models with different numbers of parameters by using the degrees of freedom in its computation. The adjusted R^2 is useful in stepwise procedure because you are looking at many different models and want to adjust for the number of terms in the model.

Cp Mallow's C_p criterion for selecting a model. It is an alternative measure of total squared error and can be defined as follows:

$$C_p = \left(\frac{\text{SSE}_p}{s^2}\right) - (N - 2p)$$

where s^2 is the MSE for the full model and SSE_p is the sum-of-squares error for a model with p variables, including the intercept. Note that p is the number of x-variables+1. If C_p is graphed with p, Mallows (1973) recommends choosing the model where C_p first approaches p.

p Number of parameters in the model, including the intercept.

AICc Corrected Akaike's Information Criterion. For more details, see "Likelihood, AICc, and BIC".

BIC Bayesian Information Criterion. For more details, see "Likelihood, AICc, and BIC".

Forward Selection Example

In forward selection, terms are entered into the model and most significant terms are added until all of the terms are significant.

1. Complete the steps in "Example Using Stepwise Regression".

   Notice that the default selection for **Direction** is Forward.

2. Click **Step**.

   In Figure 5.4, you can see that after one step, the most significant term, Runtime, is entered into the model.

3. Click **Go**.

   In Figure 5.5 you can see that all of the terms have been added, except RstPulse and Weight.

Figure 5.4 Current Estimates Table for Forward Selection after One Step

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	82.4217727	1	0	0.000	1
☐	☐	Weight	0	1	1.323628	0.171	0.68267
☐	☑	Runtime	-3.3105554	1	632.9001	84.008	4.6e-10
☐	☐	RunPulse	0	1	15.36208	2.118	0.15673
☐	☐	RstPulse	0	1	0.130138	0.017	0.89814
☐	☐	MaxPulse	0	1	1.567361	0.202	0.65632

Figure 5.5 Current Estimates Table for Forward Selection after Three Steps

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	80.9007896	1	0	0.000	1
☐	☐	Weight	0	1	4.989591	0.827	0.37137
☐	☑	Runtime	-2.9701867	1	443.2028	73.971	3.25e-9
☐	☑	RunPulse	-0.3751142	1	55.14175	9.203	0.00529
☐	☐	RstPulse	0	1	0.350744	0.056	0.81399
☐	☑	MaxPulse	0.35421891	1	41.34703	6.901	0.01403

Backward Selection Example

In backward selection, all terms are entered into the model and then the least significant terms are removed until all of the remaining terms are significant.

1. Complete the steps in "Example Using Stepwise Regression".
2. Click **Enter All**.

Figure 5.6 All Effects Entered into the Model

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	82.3936054	1	0	0.000	1
☐	☑	Weight	-0.0509071	1	4.83788	0.772	0.38784
☐	☑	Runtime	-2.9518165	1	366.3375	58.489	5.29e-8
☐	☑	RunPulse	-0.3970425	1	59.51519	9.502	0.00495
☐	☑	RstPulse	0.01239004	1	0.199033	0.032	0.85995
☐	☑	MaxPulse	0.38479281	1	45.83023	7.317	0.01212

3. For **Direction**, select Backward.
4. Click **Step** two times.

   The first backward step removes RstPulse and the second backward step removes Weight.

Figure 5.7 Current Estimates with Terms Removed and Step History Table

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	80.9007896	1	0	0.000	1
☐	☐	Weight	0	1	4.989591	0.827	0.37137
☐	☑	Runtime	-2.9701867	1	443.2028	73.971	3.25e-9
☐	☑	RunPulse	-0.3751142	1	55.14175	9.203	0.00529
☐	☐	RstPulse	0	1	0.350744	0.056	0.81399
☐	☑	MaxPulse	0.35421891	1	41.34703	6.901	0.01403

Step History

Step	Parameter	Action	"Sig Prob"	Seq SS	RSquare	Cp	p	AICc	BIC	
1	All	Entered	.	.	0.8161	6	6	157.051	162.22	○
2	RstPulse	Removed	0.8600	0.199033	0.8158	4.0318	5	153.721	158.825	○
3	Weight	Removed	0.3714	4.989591	0.8100	2.8284	4	151.592	156.362	◉

The Current Estimates and Step History tables shown in Figure 5.7 summarize the backward stepwise selection process. Note the BIC value of 156.362 for the third step in the Step History table. If you click Step again to remove another parameter from the model, the BIC value increases to 159.984. For this reason, you choose the step 3 model. This is also the model that the Go button produces.

Current Estimates Report

Use the Current Estimates report to enter, remove, and lock in model effects. (The intercept is permanently locked into the model.) This report contains the following columns:

Figure 5.8 Current Estimates Table

SSE	DFE	RMSE	RSquare	RSquare Adj	Cp	p	AICc	BIC
851.38154	30	5.3272305	0.0000	0.0000	106.93073	1	195.1018	197.5412

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	47.3758065	1	0	0.000	1
☐	☐	Weight	0	1	22.55181	0.789	0.38169
☐	☐	Runtime	0	1	632.9001	84.008	4.6e-10
☐	☐	RunPulse	0	1	134.8447	5.457	0.0266
☐	☐	RstPulse	0	1	135.7828	5.503	0.02604
☐	☐	MaxPulse	0	1	47.71646	1.722	0.19975

Lock Locks a term in or out of the model. A checked term cannot be entered or removed from the model.

Entered Indicates whether a term is currently in the model. You can click a term's check box to manually bring an effect in or out of the model.

Parameter The effect names.

Estimate The current parameter estimate, which is zero if the effect is not currently in the model.

nDF The number of degrees of freedom for a term. A term has more than one degree of freedom if its entry into a model also forces other terms into the model.

Wald/Score ChiSq (Shown only when response is categorical.) The test statistic for each term in the model. For terms not already included in the model, the test statistic is based on a score test of including the term in the model. For terms that are already included in the model, the test statistic is based on a Wald test of removing the term from the model.

"Sig Prob" (Shown only when response is categorical.) The *p*-value associated with the Wald/Score ChiSq test statistic based on nDF degrees of freedom. The "Sig Prob" is used to determine the next term to be included in the model.

SS (Shown only when response is continuous.) The reduction in the error (residual) sum of squares (SS) if the term is entered into the model or the increase in the error SS if the term is removed from the model. If a term is restricted in some fashion, it could have a reported SS of zero.

"F Ratio" (Shown only when response is continuous.) The traditional test statistic to test that the term effect is zero. It is the square of a *t*-ratio. It is in quotation marks because it does not have an *F*-distribution for testing the term because the model was selected as it was fit.

"Prob>F" (Shown only when response is continuous.) The *p*-value associated with the *F* statistic. Like the "F Ratio," it is in quotation marks because it is not to be trusted as a real significance probability.

R The multiple correlation with the other effects in the model. This column appears only if you right-click in the report and select Columns > R.

Step History Report

As each step is taken, the Step History report records the effect of adding a term to the model. For example, the Step History report for the Fitness.jmp example shows the order in which the terms entered the model and shows the statistics for each model. See "Example Using Stepwise Regression".

Use the radio buttons on the right to choose a model.

Figure 5.9 Step History Report

Step History

Step	Parameter	Action	"Sig Prob"	Seq SS	RSquare	Cp	p	AICc	BIC	
1	Runtime	Entered	0.0000	632.9001	0.7434	7.8825	2	155.397	158.81	○
2	RunPulse	Entered	0.1567	15.36208	0.7614	7.4298	3	155.787	159.984	○
3	MaxPulse	Entered	0.0140	41.34703	0.8100	2.8284	4	151.592	156.362	◉

Step History for Categorical Responses

The Step History report for models with a categorical response contains four additional columns: L-R ChiSquare, "Sig Prob", Entry ChiSquare, and Entry "Sig Prob".

The L-R ChiSquare and "Sig Prob" columns contain the full-versus-reduced likelihood ratio test statistic and *p*-value. Here, the full model is the one that contains the specified term and the reduced model does not contain the specified term.

The Entry ChiSquare and Entry "Sig Prob" columns contain the Wald/Score ChiSquare and "Sig Prob" values that were used to choose the most recent term to include in the model.

Figure 5.10 Step History for a Categorical Response before First Step

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	Wald/Score ChiSq	"Sig Prob"
☑	☑	Intercept[M]	-0.0645385	1	0	1
☐	☐	Oxy	0	1	7.275453	0.00699
☐	☐	Weight	0	1	10.13425	0.00146
☐	☐	Runtime	0	1	5.764797	0.01635
☐	☐	RunPulse	0	1	1.865024	0.17205
☐	☐	RstPulse	0	1	2.693096	0.10078
☐	☐	MaxPulse	0	1	1.467541	0.22573

Step History

Step	Parameter	Action	L-R ChiSquare	"Sig Prob"	Entry ChiSquare	Entry "Sig Prob"	RSquare	p	AICc	BIC

Figure 5.11 Step History for a Categorical Response after First Step

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	Wald/Score ChiSq	"Sig Prob"
☑	☑	Intercept[M]	-16.228294	1	0	1
☐	☐	Oxy	0	1	8.750599	0.0031
☐	☑	Weight	0.20792272	1	6.994297	0.00818
☐	☐	Runtime	0	1	6.371395	0.0116
☐	☐	RunPulse	0	1	1.428737	0.23197
☐	☐	RstPulse	0	1	3.510878	0.06097
☐	☐	MaxPulse	0	1	0.365726	0.54534

Step History

Step	Parameter	Action	L-R ChiSquare	"Sig Prob"	Entry ChiSquare	Entry "Sig Prob"	RSquare	p	AICc	BIC
1	Weight	Entered	12.16669	0.0005	10.1342	0.00146	0.2833	2	35.2047	37.6441

Note that the Entry ChiSquare and Entry "Sig Prob" values for Weight in the Step History table in Figure 5.11 match the Wald/Score ChiSq and "Sig Prob" values for Weight in Figure 5.10.

Models with Crossed, Interaction, or Polynomial Terms

Some models, especially those associated with experimental designs, involve interaction terms. For continuous factors, these are products of the columns representing the effects. For nominal and ordinal factors, interactions are defined by model terms that involve products of terms representing the categorical levels.

When there are interaction terms, you often want to impose a restriction on the model selection process so that lower-order components of higher-order effects are included in the model. This is suggested by the principle of Effect Heredity. See the *Design of Experiments Guide*. For example, if a two-way interaction is included in a model, its component main effects (*precedents*) should be included as well.

Example of the Combine Rule

1. Select **Help > Sample Data Library** and open Reactor.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. In the Degree box, type 2.
5. Select F, Ct, A, T, and Cn and click **Macros > Factorial to Degree**.
6. For **Personality**, select **Stepwise**.
7. Click **Run**.

Figure 5.12 Initial Current Estimates Report Using Combine Rule

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	65.5	1	0	0.000	1
☐	☐	F	0	1	15.125	0.066	0.79972
☐	☐	Ct	0	1	3042	23.412	3.67e-5
☐	☐	A	0	1	3.125	0.014	0.90823
☐	☐	T	0	1	924.5	4.611	0.03998
☐	☐	Cn	0	1	312.5	1.415	0.24363
☐	☐	F*Ct	0	1	15.125	0.066	0.79972
☐	☐	F*A	0	3	22.75	0.031	0.9926
☐	☐	F*T	0	1	6.125	0.027	0.87178
☐	☐	F*Cn	0	1	0.125	0.001	0.98161
☐	☐	Ct*A	0	1	6.125	0.027	0.87178
☐	☐	Ct*T	0	1	1404.5	7.612	0.00979
☐	☐	Ct*Cn	0	1	32	0.139	0.71193
☐	☐	A*T	0	1	36.125	0.157	0.69476
☐	☐	A*Cn	0	1	6.125	0.027	0.87178
☐	☐	T*Cn	0	1	968	4.863	0.03525

The model in Figure 5.12 contains all terms for up to two-factor interactions for the five continuous factors. The Combine, Restrict, and Whole Effects rules described in "Rules" enable you to control entry of interaction terms.

The Combine rule determines the entry of interaction terms based on two tests. See "Combine". You can determine which of the two tests was used for the *p*-value based on the degrees of freedom, nDF. For example, the interaction term F*A has an nDF value of 3. This means that F*A is grouped with its precedent terms F and A, and is considered for entry based on the 3 degree of freedom joint *F* test. In contrast, the interaction term Ct*T has an nDF value of 1. This means that Ct*T is considered for entry based on the 1 degree of freedom *F* test that tests the significance of Ct*T after its precedent terms, Ct and T, are already included in the model. Click **Step** once to see that Ct is entered by itself.

8. Click **Step** again to see that Ct*T is entered, along with T (Ct is already in the model).

Figure 5.13 Current Estimates Report Using Combine Rule, One Step

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	65.5	1	0	0.000	1
☐	☐	F	0	1	15.125	0.263	0.61236
☐	☑	Ct	9.75	2	4446.5	39.676	6.74e-9
☐	☐	A	0	1	3.125	0.054	0.81819
☐	☑	T	5.375	2	2329	20.781	2.93e-6
☐	☐	Cn	0	1	312.5	6.715	0.01524
☐	☐	F*Ct	0	2	30.25	0.256	0.7764
☐	☐	F*A	0	3	22.75	0.123	0.9459
☐	☐	F*T	0	2	21.25	0.178	0.83755
☐	☐	F*Cn	0	1	0.125	0.002	0.96335
☐	☐	Ct*A	0	2	9.25	0.077	0.92601
☐	☑	Ct*T	6.625	1	1404.5	25.064	2.72e-5
☐	☐	Ct*Cn	0	1	32	0.562	0.45989
☐	☐	A*T	0	2	39.25	0.334	0.7194
☐	☐	A*Cn	0	1	6.125	0.106	0.74747
☐	☐	T*Cn	0	1	968	43.488	4.49e-7

When there are significant interaction terms, several terms can enter at the same step. If the **Step** button is clicked twice, Ct*T is entered along with its two contained effects Ct and T. However, a step back is not symmetric because a crossed term can be removed without removing its two component terms. Notice that Ct and T now each have 2 degrees of freedom. This is because if Stepwise removes Ct or T, it must also remove Ct*T. If you change the Direction to **Backward** and click **Step**, Ct*T is removed and the degrees of freedom for Ct and T change to 1.

Models with Nominal and Ordinal Effects

Traditionally, stepwise regression has not addressed the situation where there are categorical effects in the model. Note the following:

- When a regression model contains nominal or ordinal effects, those effects are represented by sets of indicator columns.
- When a categorical effect has only two levels, that effect is represented by a single column.
- When a categorical effect has k levels, where $k > 2$, then it must be represented by k-1 columns.

The convention in JMP for standard platforms is to represent nominal variables by terms whose parameter estimates average to zero across all the levels.

In the Stepwise platform, categorical variables (nominal and ordinal) are coded in a *hierarchical* fashion. This differs from coding in other least squares fitting platforms. In hierarchical coding, the levels of the categorical variable are successively split into groups of levels that most separate the means of the response. The splitting process achieves the goal of representing a k-level categorical variable by k - 1 terms.

Note: In hierarchical coding, the initial terms that are constructed represent the groups responsible for the greatest separation. The advantage of this coding scheme is that these informative terms have the potential to enter the model early.

Construction of Hierarchical Terms

Hierarchical terms are constructed using a tree structure that is analogous to a Partition analysis. However, the criterion that is maximized is the sum of squares between groups (SSB).

For a nominal variable with *k* levels, the *k* levels are split into two groups of levels that have maximum SSB. Call these two groups of levels A1 and A2, where A1 has the smaller mean and A2 has the larger mean. The two groups of levels in A1 and A2 are used to define an indicator variable with values of 1 for the levels in A1 and -1 for the levels in A2. This variable is the first hierarchical term for the nominal variable.

For the levels within each of the initial two groups A1 and A2, the split into two groups of levels with the maximum SSB is identified. Suppose that the groups of levels with maximum SSB are among the levels in A1. Call the two groups B1 and B2, where B1 has the smaller mean and B2 has the larger mean. The two groups of levels in B1 and B2 are used to define a hierarchical variable with values of 1 for the levels in B1, -1 for the levels in B2, and 0 for the levels in A2. To construct the next variable, splits of the levels in B1, B2, and A2 are considered. The split that maximizes SSB defines the next hierarchical variable. The process continues until *k-1* hierarchical terms are constructed.

For an ordinal variable, the groups of levels considered in splitting contain only levels that are contiguous in the ordering. This ensures that the constructed terms respect the level ordering.

Rules and Hierarchical Terms

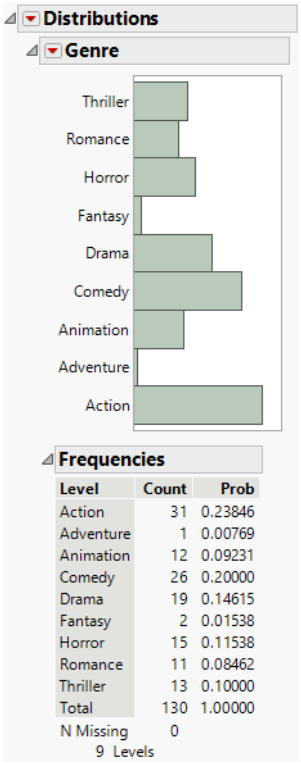
When you use the **Combine** rule or the **Restrict** rule, a term cannot enter the model unless all the terms above it in the hierarchy have been entered. When you use the **Whole Effects** rule and enter a term for a categorical variable, all of its associated terms are entered. For an example, see "Construction of Hierarchical Terms in Example".

Example of a Model with a Nominal Term

This example uses data on movies that were released in 2011. You are particularly interested in the World Gross values, which represent the gross receipts. Your potential predictors are Rotten Tomatoes Score, Audience Score, and Genre. The two score variables are continuous, but Genre is nominal. Before you attempt to reduce your model using Stepwise, you want to explore the variables of interest.

1. Select **Help > Sample Data Library** and open Hollywood Movies.jmp.
2. Select **Analyze > Distribution**.
3. Select Genre and click **Y, Columns**.
4. Click **OK**.

Figure 5.14 Distribution of Genre

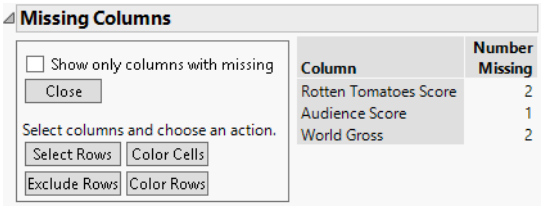


Level	Count	Prob
Action	31	0.23846
Adventure	1	0.00769
Animation	12	0.09231
Comedy	26	0.20000
Drama	19	0.14615
Fantasy	2	0.01538
Horror	15	0.11538
Romance	11	0.08462
Thriller	13	0.10000
Total	130	1.00000
N Missing	0	
9 Levels		

Note that Genre has nine levels, so it is represented by eight model terms. Further data exploration will reveal that, because of missing data, only eight levels are considered by Stepwise.

5. In the data table's Columns panel, select the columns of interest: Rotten Tomatoes Score, Audience Score, and World Gross.
6. Select **Analyze > Screening > Explore Missing Values**.
7. Click **Y, Columns** and click **OK**.

Figure 5.15 Missing Columns Report



Column	Number Missing
Rotten Tomatoes Score	2
Audience Score	1
World Gross	2

Note that Rotten Tomatoes Score is missing in 2 rows, Audience Score is missing in 1 row, and World Gross is missing in 2 rows.

8. In the Missing Columns report, select the three columns listed under **Column**.
9. Click **Select Rows**.

   In the data table's Rows panel, you can see that three rows are selected. Because these three rows contain missing data on the predictors or response, they are automatically excluded from the Stepwise analysis. Note that row 128 is the only entry in the Adventure category, which means that category is entirely removed from the analysis. For the purposes of the Stepwise analysis, it follows that Genre has only eight categories. Now that you have seen the effect of the missing data, you will conduct the Stepwise analysis.
10. Select **Analyze > Fit Model**.
11. Select Rotten Tomatoes Score, Audience Score, and Genre and click **Add**.

    If you fit a standard least squares model to World Gross using Rotten Tomatoes Score, Audience Score, and Genre as predictors, the residuals are highly heteroscedastic. (This is typical of financial data.) Use a log transformation to better satisfy the regression assumption of equal variance.
12. Right-click World Gross in the Select Columns list and select **Transform > Log**.

    The transformed variable *Log[World Gross]* appears at the bottom of the Select Columns list.
13. Select *Log[World Gross]* and click **Y**.
14. Select **Stepwise** from the Personality list.
15. Click **Run**.

Figure 5.16 Current Estimates Table Showing List of Model Terms

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	4.09998703	1	0	0.000	1
☐	☐	Rotten Tomatoes Score	0	1	10.78976	3.303	0.07154
☐	☐	Audience Score	0	1	8.715114	2.655	0.10577
☐	☐	Genre{Drama&Horror&Thriller&Fantasy&Romance&Comedy-Action&Animation}	0	1	49.78148	16.850	7.25e-5
☐	☐	Genre{Drama&Horror&Thriller-Fantasy&Romance&Comedy}	0	1	9.560063	2.918	0.09008
☐	☐	Genre{Drama-Horror&Thriller}	0	1	2.027141	0.608	0.43718
☐	☐	Genre{Horror-Thriller}	0	1	0.013011	0.004	0.95043
☐	☐	Genre{Fantasy&Romance-Comedy}	0	1	1.428612	0.428	0.51439
☐	☐	Genre{Fantasy-Romance}	0	1	0.000376	0.000	0.99157
☐	☐	Genre{Action-Animation}	0	1	1.362765	0.408	0.52426

In the Current Estimates table, note that Genre is represented by 7 terms. You will construct a model using two of these to see how these terms are defined.

16. Check the boxes under **Entered** next to the first two terms for Genre:
    - Genre{Drama&Horror&Thriller&Fantasy&Romance&Comedy-Action&Animation}
    - Genre{Drama&Horror&Thriller-Fantasy&Romance&Comedy}

17. Click **Make Model**.

    Notice that the two terms are added as temporary transform columns to the Model Effects list in the Model Specification window. These columns are discussed in the next section.

Construction of Hierarchical Terms in Example

Recall that because of missing values, Genre is a nominal variable with eight levels. In the Current Estimates table, Genre is represented by seven terms. This is appropriate, because Genre has eight levels. The first two terms that represent Genre are described below. Subsequent terms are defined in a similar fashion.

First Term

The first term that appears is Genre{Drama&Horror&Thriller&Fantasy&Romance&Comedy-Action&Animation}. This variable has the form Genre{A1 - A2}, where A1 and A2 are separated by a minus sign. The notation indicates that the maximum separation in terms of sum of squares between groups occurs between the following two sets of levels:

- Drama, Horror, Thriller, Fantasy, Romance, and Comedy (represented by A1)
- Action and Animation (represented by A2)

If you include the term Genre{Drama&Horror&Thriller&Fantasy&Romance&Comedy-Action&Animation} in a model, a temporary transform column representing that term is used in the model. The column contains the following values:

- 1 for Drama, Horror, Thriller, Fantasy, Romance, and Comedy
- -1 for Action and Animation

Second Term

The second term that appears is Genre{Drama&Horror&Thriller-Fantasy&Romance&Comedy}. This set of levels is entirely contained in the first split for the first term (A1). The notation contrasts the levels:

- Drama, Horror, and Thriller
- Fantasy, Romance, and Comedy

Among all the splits of the levels of Drama, Horror, Thriller, Fantasy, Romance, and Comedy (A1) and of the levels of Action and Animation (A2), the algorithm determines that this split has the largest sum of squares between groups.

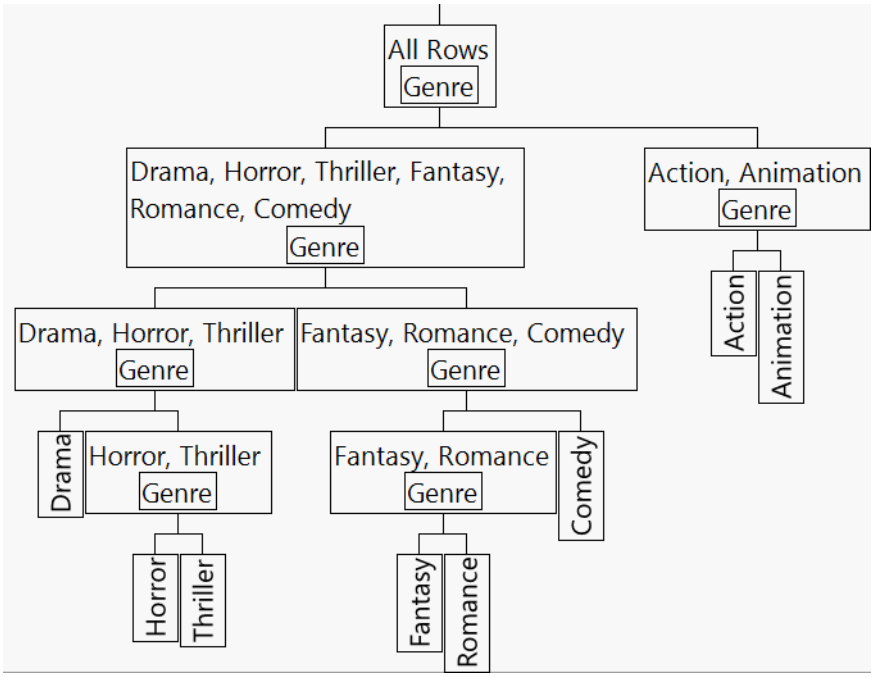
If you include this term in a model, a temporary transform column representing that term is used in the model. The column contains the following values:

- 1 for Drama, Horror, and Thriller
- -1 for Fantasy, Romance, and Comedy
- 0 for Action and Animation

Hierarchy of Terms

The splitting of terms continues, based on the sum of squares between groups criterion. The hierarchy that leads to the definition of the terms is illustrated in Figure 5.17.

Figure 5.17 Tree Showing Splits Used in Hierarchical Coding



Rules

When you use the **Combine** rule or the **Restrict** rule, a term cannot enter the model unless all the terms above it in the hierarchy have been entered. For example, if you enter Genre{Action-Animation}, then JMP enters Genre{Drama&Horror&Thriller&Fantasy&Romance&Comedy-Action&Animation} as well.

When you use the **Whole Effects** rule and enter any one of the Genre terms, all of the Genre terms are entered.

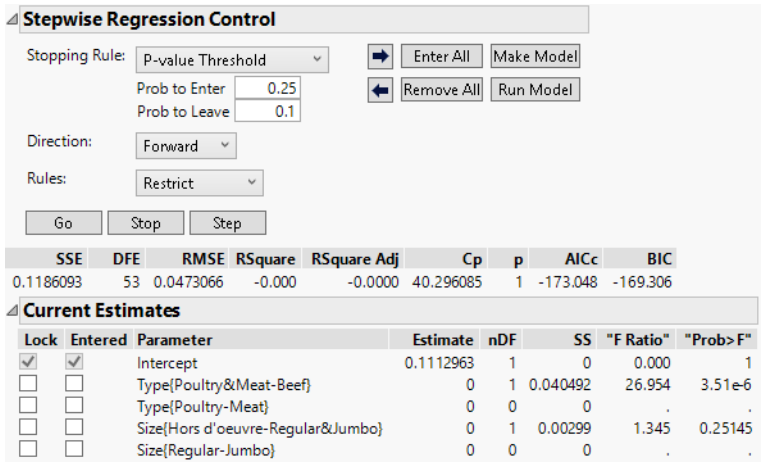
Example of the Restrict Rule for Hierarchical Terms

If you have a model with nominal or ordinal terms, when you make or run the model, temporary transform columns containing the hierarchical terms involved in the model are used in the model fit. The model itself appears in a new Fit Model window. This example further illustrates how **Stepwise** constructs a model with hierarchical effects.

A simple model examines the cost per ounce ($/oz) of hot dogs as a function of the Type of hot dog (Meat, Beef, Poultry) and the Size of the hot dog (Jumbo, Regular, Hors d'oeuvre).

1. Select **Help > Sample Data Library** and open Hot Dogs2.jmp.
2. Select **Analyze > Fit Model**.
3. Select $/oz and click **Y**.
4. Select Type and Size and click **Add**.
5. For **Personality**, select **Stepwise**.
6. Click **Run**.
7. For **Stopping Rule**, select **P-value Threshold**.
8. For **Rules**, select **Restrict**.

Figure 5.18 Stepwise Control Panel with P-value Threshold and Restrict Rule



Notice that when you change from the default Rule of Combine to Restrict, the F Ratio and Prob > F values for two terms are shown as missing. These are the terms Type{Poultry-Meat} and Size{Regular-Jumbo}. This is because these two terms cannot enter the model until their precedent terms enter.

9. Click **Step**.

The term Type{Poultry&Meat-Beef} enters the model. This term has the smallest Prob>F value, and that value falls below the Prob to Enter threshold of 0.25.

Figure 5.19 Stepwise Control Panel with One Term Entered

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	0.11864706	1	0	0.000	1
☐	☑	Type{Poultry&Meat-Beef}	-0.0283529	1	0.040492	26.954	3.51e-6
☐	☐	Type{Poultry-Meat}	0	1	0.012426	9.648	0.00309
☐	☐	Size{Hors d'oeuvre-Regular&Jumbo}	0	1	0.001038	0.687	0.4112
☐	☐	Size{Regular-Jumbo}	0	0	0	.	.

The F Ratio and Prob > F values for the term Type{Poultry-Meat} appear. Since its precedent term has entered the model, Type{Poultry-Meat} is now allowed to enter.

10. Click **Step**.

    Since Type{Poultry-Meat} has the smallest Prob>F value among the remaining terms, and that value is below the Prob to Enter threshold, it is the next term to enter the model.

11. Click **Step**.

    The term Size{Hors d'oeuvre-Regular&Jumbo} enters the model, since its Prob>F value is 0.1577. Because its precedent term is now in the model, the term Size{Regular-Jumbo} is allowed to enter the model and its Prob>F value appears.

    However, the Prob>F value for the term Size{Regular-Jumbo} is 0.7566, which exceeds the Prob to Enter value of 0.25. For this reason, if you click Step again, it is not entered into the model.

Figure 5.20 Current Estimates Report for the Final Model

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	SS	"F Ratio"	"Prob>F"
☑	☑	Intercept	0.10849381	1	0	0.000	1
☐	☑	Type{Poultry&Meat-Beef}	-0.0275278	0	0	.	.
☐	☑	Type{Poultry-Meat}	-0.0205527	1	0.013985	11.083	0.00164
☐	☑	Size{Hors d'oeuvre-Regular&Jumbo}	-0.0121982	1	0.002596	2.057	0.1577
☐	☐	Size{Regular-Jumbo}	0	1	0.000125	0.097	0.7566

Tip: Use the Go button to run the entire stepwise process automatically. To see this in action, click **Remove All**. Then click **Go**.

12. Click **Make Model**.

    After you click Make Model, a Fit Model launch window appears, containing only the three model effects that were selected in the stepwise process. In the launch window, temporary transform columns that define the three hierarchical effects entered into the model are included in the model.

Performing Binary and Ordinal Logistic Stepwise Regression

The Stepwise personality of Fit Model performs ordinal logistic stepwise regression when the response is ordinal or nominal. Nominal responses are treated as ordinal responses in the logistic stepwise regression fitting procedure. When a response has only two levels, ordinal logistic regression models are equivalent to nominal logistic regression models. To run a logistic stepwise regression, specify an ordinal or nominal response, add terms to the model as usual, and choose Stepwise from the Personality menu.

The Stepwise reports for a logistic model are similar to those provided when the response is continuous. The following elements are specific to logistic regression results:

- When the response is categorical, the overall fit of the model is given by its negative log-likelihood (-LogLikelihood). This value is calculated based on the full iterative maximum likelihood fit. See "Likelihood, AICc, and BIC".
- The Current Estimates report shows chi-square statistics (Wald/Score ChiSq) and their *p*-values (Sig Prob). The test statistic column shows score statistics for parameters that are not in the current model and shows Wald statistics for parameters that are in the current model. The regression estimates (Estimate) are based on the full iterative maximum likelihood fit.
- The Step History report shows the L-R ChiSquare. This value is the test statistic for the likelihood ratio test of the hypothesis that the corresponding regression parameter is zero, given the other terms in the model. The Sig Prob is the *p*-value for this test.

Note: If the response is nominal, you can fit the current model using the Nominal Logistic personality of Fit Model by clicking the Make Model button. In the Fit Model launch window that appears, click Run.

Example Using Logistic Stepwise Regression

1. Select **Help > Sample Data Library** and open Fitness.jmp.
2. Select **Analyze > Fit Model**.
3. Select Sex and click **Y**.
4. Select Weight, Runtime, RunPulse, RstPulse, and MaxPulse and click **Add**.
5. For **Personality**, select **Stepwise**.
6. Click **Run**.
7. Click **Go**.

Figure 5.21 Logistic Stepwise Report

Current Estimates

Lock	Entered	Parameter	Estimate	nDF	Wald/Score ChiSq	"Sig Prob"
☑	☑	Intercept[M]	-35.807846	1	0	1
☐	☑	Weight	0.28229655	1	5.471429	0.01933
☐	☑	Runtime	1.32680366	1	2.545732	0.11059
☐	☐	RunPulse	0	1	0.470408	0.4928
☐	☐	RstPulse	0	1	0.007216	0.9323
☐	☐	MaxPulse	0	1	0.076497	0.7821

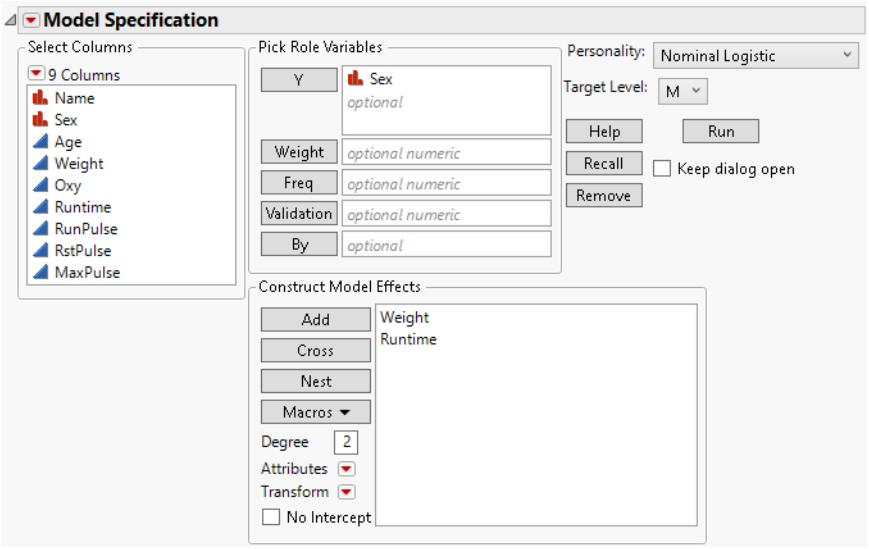
Step History

Step	Parameter	Action	L-R ChiSquare	"Sig Prob"	Entry ChiSquare	Entry "Sig Prob"	RSquare	p	AICc	BIC
1	Weight	Entered	12.16669	0.0005	10.1342	0.00146	0.2833	2	35.2047	37.6441
2	Runtime	Entered	8.017826	0.0046	6.37139	0.0116	0.4700	3	29.6472	33.0603
3	RunPulse	Entered	1.440088	0.2301	1.44684	0.22903	0.5036	4	30.8567	35.0542
4	MaxPulse	Entered	0.071809	0.7887	0.06994	0.79142	0.5052	5	33.6464	38.4164
5	RstPulse	Entered	0.007156	0.9326	0.00722	0.93228	0.5054	6	36.7393	41.8432
6	Best	Specific	.	.	0.00722	0.93228	0.4700	3	29.6472	33.0603

The two variables Weight and Runtime are entered into the model based on the Stopping Rule.

8. Click **Make Model**.

Figure 5.22 Model Specification Window for Reduced Model



A model specification window appears containing the two variables as model effects. Note that the Personality is Nominal Logistic. If the response had been ordinal, the Personality would be Ordinal Logistic.

The All Possible Models Option

Use the All Possible Models option to investigate all models that can be constructed using your predictors. This option is accessed from the Stepwise red triangle.

Note the following:

- This option is not practical for large problems, when the number of models is greater than 5 million.
- Categorical predictors are represented by indicator variables. See "Models with Nominal and Ordinal Effects".

The following options restrict the number of models that appear:

Maximum number of terms in a model Enter a value for the maximum number of terms in a model.

Number of best models to see Enter the maximum number of models of each size to display. The best models according to RSquare value appear.

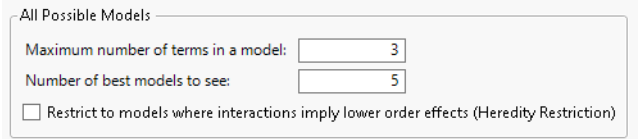
Restrict to models where interactions imply lower order effects (Heredity Restriction)

Shows only models that contain all lower-order effects when a higher-order effect is included. These models satisfy strong effect heredity. This option is useful when your predictors include interaction or polynomial terms.

Example Using the All Possible Models Option

1. Select **Help > Sample Data Library** and open Fitness.jmp.
2. Select **Analyze > Fit Model**.
3. Select Oxy and click **Y**.
4. Select Runtime, RunPulse, RstPulse, and MaxPulse and click **Add**.
5. For **Personality**, select **Stepwise**.
6. Click **Run**.
7. Click the Stepwise red triangle menu and select **All Possible Models**.
8. Enter 3 for the maximum number of terms, and enter 5 for the number of best models.

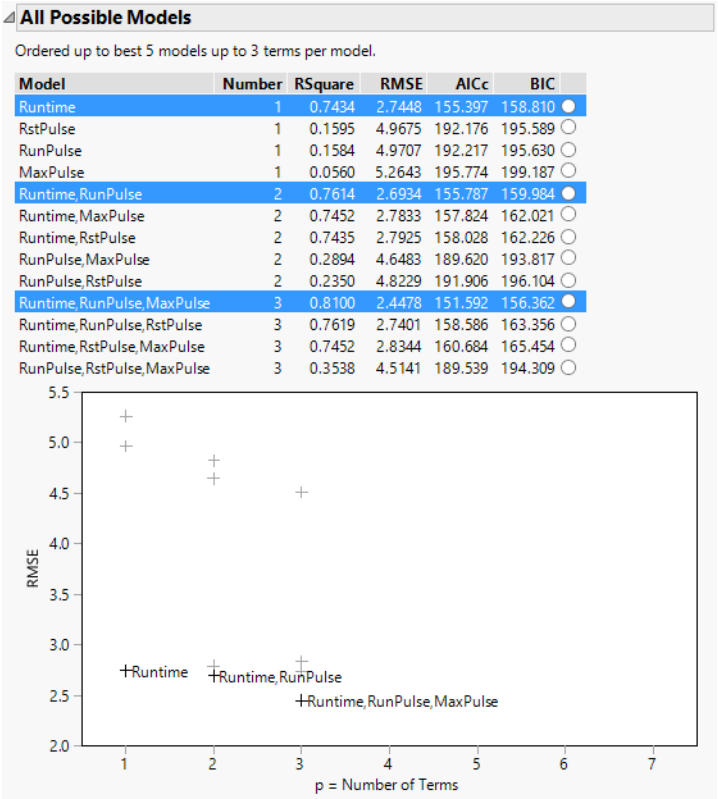
Figure 5.23 All Possible Models Pop-up Dialog



9. Click **OK**.

All possible models (up to three terms in a model) are fit.

Figure 5.24 All Possible Models Report



The models are listed in increasing order of the number of parameters that they contain. The model with the highest R^2 for each number of parameters is highlighted. The radio button column at the right of the table enables you to select one model at a time and check the results.

Note: The recommended criterion for selecting a model is to choose the one corresponding to the smallest BIC or AICc value. Some analysts also want to see the C_p statistic. Mallow's C_p statistic is computed, but initially hidden in the table. To make it visible, right-click in the table and select **Columns > Cp**.

The Model Averaging Option

The model averaging technique enables you to average the fits for several models, instead of selecting a single model. Often, the resulting average model has better prediction capability than each of the single models that were averaged. The Model Averaging feature is useful for avoiding a model that over fits your data. When many terms are selected into a model, the fit tends to inflate the parameter estimates. Model averaging tends to shrink the estimates on the weaker terms, which yields better predictions. The models are averaged with respect to the AICc Weight of each model, which is calculated as follows:

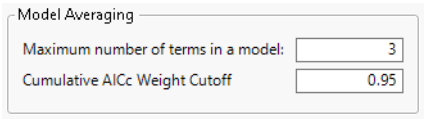
$$AICcWeight = \exp[-0.5(AICc - AICcBest)]$$

AICcBest is the smallest AICc value among the fitted models. The AICc Weight values are calculated for each model, sorted in decreasing order, and scaled to sum to 1. The scaled AICc Weight values that sum to less than one minus the specified Cumulative AICc Weight Cutoff value are set to zero. This eliminates the use of weak models in the averaged model. The parameters for the averaged model are the weighted averages of the parameter estimates across the models that have nonzero AICcWeights.

Example Using the Model Averaging Option

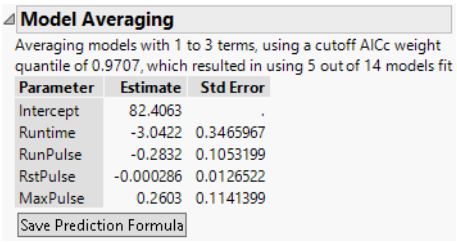
1. Select **Help > Sample Data Library** and open Fitness.jmp.
2. Select **Analyze > Fit Model**.
3. Select Oxy and click **Y**.
4. Select Runtime, RunPulse, RstPulse, and MaxPulse and click **Add**.
5. For **Personality**, select **Stepwise**.
6. Click **Run**.
7. Click the Stepwise red triangle menu and select **Model Averaging**.
8. Enter 3 for the Maximum number of terms in a model.
9. Keep 0.95 for the Cumulative AICc Weight Cutoff.

Figure 5.25 Model Averaging Window



10. Click **OK**.

Figure 5.26 Model Averaging Report



Model Averaging

Averaging models with 1 to 3 terms, using a cutoff AICc weight quantile of 0.9707, which resulted in using 5 out of 14 models fit

Parameter	Estimate	Std Error
Intercept	82.4063	.
Runtime	-3.0422	0.3465967
RunPulse	-0.2832	0.1053199
RstPulse	-0.000286	0.0126522
MaxPulse	0.2603	0.1141399

Save Prediction Formula

In the Model Averaging report, average estimates and standard errors appear for each parameter. The standard errors shown reflect the bias of the estimates toward zero.

11. Click **Save Prediction Formula** to save the prediction formula in the original data table.

Validation Options in Stepwise Regression

To perform cross validation for stepwise regression models in JMP, click the Stepwise Fit red triangle and select **K-Fold Crossvalidation**. See "K-Fold Cross Validation in Stepwise Regression".

JMP PRO In JMP Pro, you can specify a Validation column in the Fit Model window. A validation column must have a numeric data type and should contain at least two distinct values.

- If the column contains two values, the smaller value defines the training set and the larger value defines the validation set.
- If the column contains three values, the values define the training, validation, and test sets in order of increasing size.
- If the column contains four or more distinct values and the response is continuous, these values define folds for k-fold validation.

For more information about using a Validation column, see "Validation Set with Two or Three Values in Stepwise Regression".

Validation Set with Two or Three Values in Stepwise Regression

JMP PRO If you specify a Validation column with two or three values, Stepwise fits models based on the training set. Model fit statistics are reported for the validation and test sets. See "Validation and Test Set Statistic Definitions" for details about how these statistics are defined.

If the response is continuous, the following statistics appear in the Stepwise Regression Control panel:

- RSquare Validation (also shown in the Step History report)
- RASE Validation
- RSquare Test (if there is a test set)
- RASE Test (if there is a test set)

If the response is binary nominal or ordinal, the following statistics appear in the Stepwise Regression Control panel:

- RSquare Validation (also shown in the Step History report)
- Avg Log Error Validation
- RSquare Test (if there is a test set)
- Avg Log Error Test (if there is a test set)

Max Validation RSquare

If you specify a validation column with two or three values in the Fit Model window, the Stopping Rule defaults to Max Validation RSquare. This rule attempts to find a model that maximizes the RSquare statistic for the validation set. The rule can be applied with the Direction set to Forward or Backward.

Note: Max Validation RSquare considers only the models defined by *p*-value entry (Forward direction) or removal (Backward direction). It does not consider all possible models.

You can use the Step button to enter terms one-by-one in the Forward direction or to remove them one-by one in the Backward direction. At any point, you can select a model by clicking the button to the right of RSquare Validation in the Step History report. The selection of model terms is updated in the Current Estimates report. This is the model that is used once you click Make Model or Run Model.

Forward Direction

In the Forward direction, Stepwise constructs successive models by adding terms based on the next smallest *p*-value.

If you click Go rather than Step, the process of entering terms proceeds automatically. Among the fitted models, the model that is considered best is listed last. This model is obtained by overlooking local dips in RSquare Validation. Specifically, it is the model with the largest RSquare Validation that can be followed by as many as ten models with lower RSquare Validation values. This model is designated by the terms Best in the Parameter column and Specific in the Action column. The button to the right of RSquare Validation selects this Best model, though you are free to change this selection.

Backward Direction

In the Backward direction, Stepwise constructs successive models by removing terms based on the next largest *p*-value.

To use the Backward direction, you must first click Enter All to enter all of the terms into the model. The Backward direction behaves in a similar fashion to the Forward direction. If you click Go rather than Step, the process of removing terms proceeds automatically. The model designated as Best is the one with the largest RSquare Validation that can be followed by as many as ten models with lower RSquare Validation values.

Validation and Test Set Statistic Definitions

RSquare Validation and RASE Validation are defined in this section. RSquare Test and RASE Test are computed for the test set in a completely analogous fashion.

Continuous Response

RSquare Validation An RSquare measure for the validation set computed as follows:

- For each observation in the validation set, compute the prediction error. This is the difference between the actual response and the response predicted by the training set model.
- Square and sum the prediction errors to obtain $SSE_{Validation}$.
- Square and sum the differences between the actual responses in the validation set and their mean. This is the $SST_{Validation}$.
- RSquare Validation is:

$$\text{RSquare Validation} = 1 - \frac{SSE_{Validation}}{SST_{Validation}}$$

Note: It is possible for RSquare Validation to be negative.

RASE Validation The square root of the mean squared prediction error for the validation set. This is computed as follows:

- For each observation in the validation set, compute the prediction error. This is the difference between the actual response and the response predicted by the training set model.
- Square and sum the prediction errors to obtain the $SSE_{Validation}$.
- Denote the number of observations in the validation set by $n_{Validation}$.
- RASE Validation is:

$$\text{RASE Validation} = \sqrt{\frac{SSE_{Validation}}{n_{Validation}}}$$

Binary Nominal or Ordinal Response

RSquare Validation An Entropy RSquare measure (also known as McFadden's R^2) for the validation set computed as follows:

- A model is fit using the training set.
- Predicted probabilities are obtained for all observations.
- Using the predicted probabilities based on the training set model, the likelihood for the model is computed for observations in the validation set. Call this quantity $Likelihood_Full_{Validation}$.
- Using the data in the validation set, the likelihood of the reduced model (no predictors) is computed. Call this quantity $Likelihood_Reduced_{Validation}$.
- RSquare Validation is:

$$\text{RSquare Validation} = 1 - \frac{\log(Likelihood_Full_{Validation})}{\log(Likelihood_Reduced_{Validation})}$$

Note: It is possible for RSquare Validation to be negative.

Avg Log Error Validation The average log error for the validation set is computed as follows:

- For each observation in the validation set, compute the log of its predicted probability as determined by the model based on the training set.
- Sum these logs, divide by the number of observations in the validation set, and take the negative of the resulting value.

Tip: Smaller values of Avg Log Error Validation are desirable.

K-Fold Cross Validation in Stepwise Regression

K-fold cross validation randomly divides the data into k subsets. In turn, each of the k sets is used as a validation set while the remaining data are used as a training set to fit the model. In total, k models are fit and k validation statistics are obtained. The model giving the best validation statistic is chosen as the final model. This method is useful for small data sets, because it makes efficient use of limited amounts of data.

Note: K-fold cross validation is available only for continuous responses.

In JMP, click the Stepwise Fit red triangle and select **K-Fold Crossvalidation**.

In JMP Pro, you can access *k*-fold cross validation in two ways:

- Click the Stepwise Fit red triangle and select **K-Fold Crossvalidation**.
- Specify a validation column with four or more distinct values.

RSquare K-Fold Statistic

If you conduct k-fold cross validation, the RSquare K-Fold statistic appears to the right of the other statistics in the Stepwise Regression Control panel. RSquare K-Fold is the average of the RSquare Validation values for the *k* folds.

Max K-Fold RSquare

When you use *k*-fold cross validation, the Stopping Rule defaults to Max K-Fold RSquare. This rule attempts to maximize the RSquare K-Fold statistic.

Note: Max K-Fold RSquare considers only the models defined by *p*-value entry (Forward direction) or removal (Backward direction). It does not consider all possible models.

The Max K-Fold RSquare stopping rule behaves in a fashion similar to the Max Validation RSquare stopping rule. See "Max Validation RSquare". Replace references to RSquare Validation with RSquare K-Fold.

Chapter 6

JMP PRO Generalized Regression Models

Build Models Using Variable Selection Techniques

The Generalized Regression personality of the Fit Model platform is available only in JMP Pro.

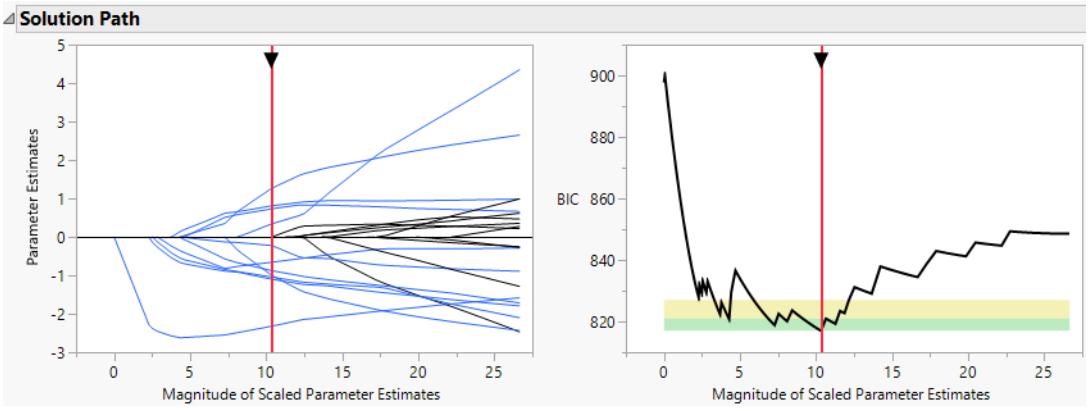
In JMP Pro, the Fit Model platform's Generalized Regression personality provides variable selection techniques, including shrinkage techniques, that specifically address modeling correlated and high-dimensional data. Two of these techniques, the Lasso and the Elastic Net, perform variable selection as part of the modeling procedure.

Large data sets that contain many variables typically exhibit multicollinearity issues. Modern data sets can include more variables than observations, requiring variable selection if traditional modeling techniques are to be used. The presence of multicollinearity and a profusion of predictors exposes the shortcomings of classical techniques.

Even for small data sets with little or no correlation, including designed experiments, the Lasso and Elastic Net are useful. They can be used to build predictive models or to select variables for model reduction or for future study.

The Generalized Regression personality is useful for many modeling situations. This personality enables you to specify a variety of distributions for your response variable. Use it when your response is continuous, binomial, a count, or zero-inflated. Use it when you are interested in variable selection or when you suspect collinearity in your predictors. More generally, use it to fit models that you compare to models obtained using other techniques.

Figure 6.1 The Solution Path for an Elastic Net Fit



Contents

Overview of the Generalized Regression Personality 283
Example of Generalized Regression 285
Launch the Generalized Regression Personality 288
Distribution 290
Generalized Regression Report Window 298
Generalized Regression Report Options 298
Model Launch Control Panel 299
Response Distribution 300
Estimation Method Options 300
Advanced Controls 305
Validation Method Options 308
Early Stopping 309
Go 310
Model Fit Reports 310
Regression Plot 310
Model Summary 311
Estimation Details 314
Solution Path 315
Parameter Estimates for Centered and Scaled Predictors 319
Parameter Estimates for Original Predictors 320
Active Parameter Estimates 321
Effect Tests 321
Model Fit Options 322
Statistical Details for the Generalized Regression Personality 333
Statistical Details for Estimation Methods 333
Statistical Details for Advanced Controls 335
Statistical Details for Distributions 336

JMP PRO Overview of the Generalized Regression Personality

The Generalized Regression personality features regularized, or penalized, regression techniques. Such techniques attempt to fit better models by shrinking the model coefficients toward zero. The resulting estimates are biased. This increase in bias can result in decreased prediction variance, thus lowering overall prediction error compared to non-penalized models. Two of these techniques, the Elastic Net and the Lasso, include variable selection as part of the modeling procedure.

Modeling techniques such as the Elastic Net and the Lasso are particularly useful for large data sets, where collinearity is typically a problem. In addition, modern data sets often include more variables than observations. This situation is sometimes referred to as the $p > n$ problem, where n is the number of observations and p is the number of predictors. Such data sets require variable selection if traditional modeling techniques are to be used.

The Elastic Net and Lasso can also be used for small data sets with little correlation, including designed experiments. They can be used to build predictive models or to select variables for model reduction or for future study.

The personality provides the following classes of modeling techniques:

- Maximum Likelihood
- Step-Based Estimation
- Penalized Regression

The Elastic Net and Lasso are relatively recent techniques (Tibshirani 1996; Zou and Hastie 2005). Both techniques penalize the size of the model coefficients, resulting in a continuous shrinkage. The amount of shrinkage is determined by a *tuning parameter*. An optimal level of shrinkage is determined by one of several validation methods. Both techniques have the ability to shrink coefficients to zero. In this way, variable selection is built into the modeling procedure. The Elastic Net model subsumes both the Lasso and ridge regression as special cases. See "Statistical Details for Estimation Methods".

Details about Generalized Regression Modeling Techniques

- The Maximum Likelihood method is a classical approach. It provides a baseline to which you can compare the other techniques, and it is the most appropriate place for traditional inference techniques such as hypothesis testing.
- Forward Selection is a method of stepwise regression. In forward selection, terms are entered into the model. The most significant terms are added until all of the terms are in the model or there are no degrees of freedom left.

- The Lasso has two shortcomings. When several variables are highly correlated, it tends to select only one variable from that group. When the number of variables, p, exceeds the number of observations, n, the Lasso selects at most n predictors.
- The Elastic Net, on the other hand, tends to select all variables from a correlated group, fitting appropriate coefficients. It can also select more than n predictors when $p > n$.
- Ridge regression was among the first of the penalized regression methods proposed (Hoerl 1962; Hoerl and Kennard 1970). Ridge regression does not shrink coefficients to zero, so it does not perform variable selection.
- The Double Lasso attempts to separate the selection and shrinkage steps by performing variable selection with an initial Lasso model. The variables selected in the initial model are then used as the input variables for a second Lasso model.
- Two-Stage Forward Selection performs two stages of forward stepwise regression. It performs variable selection on the main effects in the first stage. Then, higher-order effects are allowed to enter the model in the second stage.

The Generalized Regression personality also fits an *adaptive* version of the Lasso and the Elastic Net. These adaptive versions attempt to penalize variables in the true active set less than variables not contained in the true active set. The *true active set* refers to the set of terms in a model that have an actual effect on the response. The adaptive versions of the Lasso and Elastic Net were developed to ensure that the oracle property holds. The oracle property guarantees the following: Asymptotically, your estimates are what they would have been had you fit the model to the true active set of predictors. More specifically, your model correctly identifies the predictors that should have zero coefficients. Your estimates converge to those that would have been obtained had you started with only the true active set. See "Adaptive Methods".

The Generalized Regression personality enables you to specify a variety of distributions for your response variable. The distributions fit include normal, Cauchy, Student's t, exponential, gamma, Weibull, lognormal, beta, binomial, beta binomial, Poisson, negative binomial, zero-inflated binomial, zero-inflated beta binomial, zero-inflated Poisson, zero-inflated negative binomial, and zero-inflated gamma. This flexibility enables you to fit categorical and count responses, as well as continuous responses, and specifically, right-skewed continuous responses. You can also fit quantile regression and Cox proportional hazards models. For some of the distributions, you can fit models to censored data. The personality provides a variety of validation criteria for model selection and supports training, validation, and test columns. See "Distribution".

JMP PRO Example of Generalized Regression

The data in the Diabetes.jmp sample data table consist of measurements on 442 diabetics. The response of interest is Y, disease progression measured one year after a baseline measure was taken. Ten variables thought to be related to disease progression are also measured at baseline. This example shows how to develop a predictive model using generalized regression techniques.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y from the Select Columns list and click **Y**.
4. Select Age through Glucose and click **Macros > Factorial to Degree**.

   This adds all terms up to degree 2 (the default in the **Degree** box) to the model.
5. Select Validation from the Select Columns list and click **Validation**.
6. From the Personality list, select **Generalized Regression**.
7. Click **Run**.

   The Generalized Regression report that appears contains a Model Comparison report, a Model Launch control panel, and a Normal Standard Least Squares with Validation Column report.

   In the Model Launch control panel, note the following:

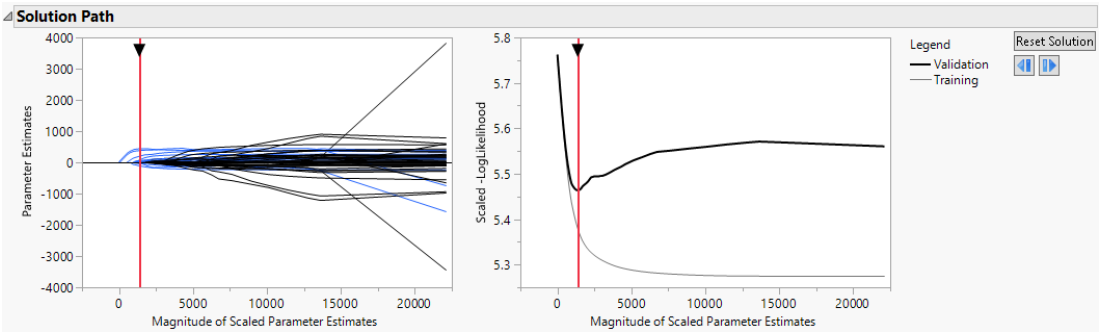
   – The Response Distribution is set to Normal because you specified Normal as the Distribution in the Fit Model launch window.

   – The default Estimation Method is the Lasso.

   – The Validation Method is set to Validation Column because you specified a validation column in the Fit Model window.
8. Click **Go**.

   A Normal Lasso with Validation Column report appears. The Solution Path report (Figure 6.2) shows plots of the parameter estimates and scaled negative log-likelihood. The shrinkage increases as the Magnitude of Scaled Parameter Estimates decreases. The estimates at the far right of the plot are the maximum likelihood estimates. A vertical red line indicates those parameter values selected by the validation criterion, in this case, the holdback sample defined by the column Validation.

Figure 6.2 Solution Path Plot



9. Click the red triangle next to Normal Lasso with Validation Column and select **Select Nonzero Terms**.

   This option highlights the nonzero terms in the Parameter Estimates for Original Predictors report (Figure 6.3) and their paths in the Solution Path Plot. The corresponding columns in the data table are also selected. Note that only 11 of the 55 parameter estimates are nonzero. The scale parameter for the normal distribution (sigma) is also estimated and shown in a separate table at the bottom of the Parameter Estimates for Original Data report. Note that not all of the 55 parameter estimates appear in Figure 6.3.

Figure 6.3 Portion of Parameter Estimates for Original Predictors Report

Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	-247.8422	42.926975	33.334183	<.0001*	-331.9775	-163.7068
Age	0	0	0	1.0000	0	0
Gender[1-2]	8.7286488	6.9678967	1.5692446	0.2103	-4.928178	22.385475
BMI	5.702121	0.9303784	37.562422	<.0001*	3.8786128	7.5256293
BP	0.8156913	0.2526706	10.421787	0.0012*	0.320466	1.3109166
Total Cholesterol	0	0	0	1.0000	0	0
LDL	0	0	0	1.0000	0	0
HDL	-0.47425	0.3986141	1.4155011	0.2341	-1.25552	0.3070188
TCH	0	0	0	1.0000	0	0
LTG	41.286654	7.0844177	33.963392	<.0001*	27.401451	55.171858
Glucose	0	0	0	1.0000	0	0
(Age-48.5181)*Gender[1-2]	-0.082735	0.3621512	0.0521915	0.8193	-0.792538	0.6270682
(Age-48.5181)*(BMI-26.3758)	0	0	0	1.0000	0	0
(Age-48.5181)*(BP-94.647)	0.0043544	0.0171735	0.0642885	0.7998	-0.029305	0.0380137
(Age-48.5181)*(Total Cholesterol-189.14)	0	0	0	1.0000	0	0
(Age-48.5181)*(LDL-115.439)	0	0	0	1.0000	0	0
(Age-48.5181)*(HDL-49.7885)	0	0	0	1.0000	0	0
(Age-48.5181)*(TCH-4.07025)	0	0	0	1.0000	0	0
(Age-48.5181)*(LTG-4.64141)	0	0	0	1.0000	0	0
(Age-48.5181)*(Glucose-91.2602)	0.0231484	0.0233144	0.9858084	0.3208	-0.022547	0.0688439
Gender[1-2]*(BMI-26.3758)	0	0	0	1.0000	0	0
Gender[1-2]*(BP-94.647)	0	0	0	1.0000	0	0
Gender[1-2]*(Total Cholesterol-189.14)	0	0	0	1.0000	0	0
Gender[1-2]*(LDL-115.439)	0	0	0	1.0000	0	0
Gender[1-2]*(HDL-49.7885)	-0.221009	0.4728172	0.2184913	0.6402	-1.147714	0.7056955
Gender[1-2]*(TCH-4.07025)	0	0	0	1.0000	0	0
Gender[1-2]*(LTG-4.64141)	0	0	0	1.0000	0	0
Gender[1-2]*(Glucose-91.2602)	0	0	0	1.0000	0	0
(BMI-26.3758)*(BP-94.647)	0.0254032	0.0525392	0.2337819	0.6287	-0.077572	0.1283782
(BMI-26.3758)*(Total Cholesterol-189.14)	0	0	0	1.0000	0	0
(BMI-26.3758)*(LDL-115.439)	0	0	0	1.0000	0	0
(BMI-26.3758)*(HDL-49.7885)	0	0	0	1.0000	0	0
(BMI-26.3758)*(TCH-4.07025)	0	0	0	1.0000	0	0
(BMI-26.3758)*(LTG-4.64141)	0	0	0	1.0000	0	0
(BMI-26.3758)*(Glucose-91.2602)	0.0210214	0.0597655	0.1237153	0.7250	-0.096117	0.1381596

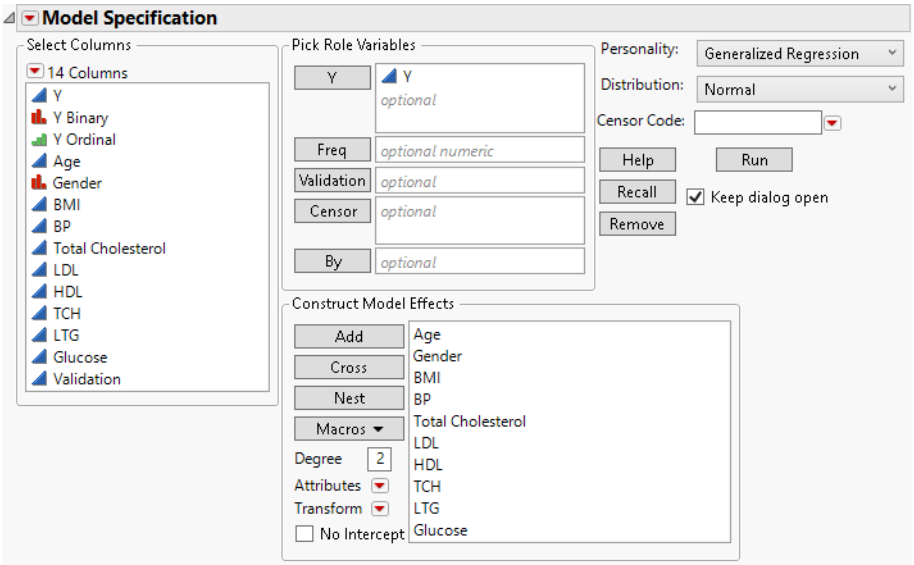
Normal Distribution Parameters	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Scale	52.512586	2.3230199	511.00007	<.0001*	47.959551	57.065621

To save the prediction formula, click the red triangle next to Normal Lasso with Validation Column report and select **Save Columns > Save Prediction Formula**.

JMP PRO Launch the Generalized Regression Personality

Launch the Generalized Regression personality by selecting **Analyze > Fit Model**, entering one or more columns for **Y**, and selecting **Generalized Regression** from the **Personality** menu.

Figure 6.4 Fit Model Launch Window with Generalized Regression Selected



For more information about aspects of the Fit Model window that are common to all personalities, see "Model Specification". For more information about the options in the Select Columns red triangle menu, see *Using JMP*. Information specific to the Generalized Regression personality is presented here.

The parameterization of nominal variables used in the Generalized Regression personality differs from their parameterization using other Fit Model personalities. The Generalized Regression personality uses indicator function parameterization. In this parameterization, the estimate that corresponds to the indicator for a level of a nominal variable is an estimate of the difference between the mean response at that level and the mean response at the last level. The last level is the level with the highest value order coding; it is the level whose indicator function is not included in the model.

If your model effects have missing values, you can treat these missing values as informative categories. Select the Informative Missing option from the Model Specification red triangle menu.

To specify a model without an intercept term, select the No Intercept option in the Construct Model Effects panel of the Fit Model window. If you select this option, note the following:

- The predictors are not centered and scaled.
- Odds ratios, hazard ratios, and incidence rate ratios are not available in the report window.
- The No Intercept option is not available for the Ordinal Logistic Distribution.

Caution: Using the No Intercept option with the Lasso or Elastic Net is not recommended because the results are sensitive to the scale of the model effects. The adaptive versions of these estimation methods are recommended instead.

Censoring

You can specify censoring for your response variable in one of the following ways:

- For interval-censored, right-censored, and left-censored responses, specify a column that contains a Detection Limits column property. The limits specified in the column property define the range of response values that are considered uncensored:
  - For interval-censored responses, specify a nonmissing value for both the Lower Detection Limit and the Upper Detection Limit in the Detection Limits column property.
  - For right-censored responses, specify a missing value for the Lower Detection Limit and a nonmissing value for the Upper Detection Limit in the Detection Limits column property.
  - For left-censored responses, specify a nonmissing value for the Lower Detection Limit and a missing value for the Upper Detection Limit in the Detection Limits column property.
- For right-censored responses, specify a column that contains indicators for right-censored observations as a Censor column in the launch window. Select the value in that column that designates right-censored observations from the Censor Code list.
- For interval-censored and left-censored responses, specify two columns that define the censoring interval in the Y column role:
  - For interval-censored responses, the first Y variable gives the lower limit.
  - For left-censored responses, the first Y variable contains a missing value.
  - For both interval-censored and left-censored responses, the second Y variable gives the upper limit for each response.

  If you specify two columns for Y and a Distribution that supports censoring, an Alert appears that asks whether the columns represent censoring. If you choose No, the columns are treated as separate responses.

Note: You can specify the default behavior for two responses using the Treatment of Two Response Columns preference in Generalized Regression platform preferences.

Censoring is available when the specified Distribution is Normal, Exponential, Gamma, Weibull, Lognormal, or Cox Proportional Hazards.

JMP PRO Distribution

When you select **Generalized Regression** from the **Personality** menu, the **Distribution** option appears. Here you can specify a distribution for Y. The abbreviation *ZI* means *zero-inflated*. The distributions are separated into three categories based on their response: continuous, discrete, and zero-inflated. The options are described below.

Note: If you specify multiple Y variables in the Model Specification window, the same response distribution must be used for all of the specified Y variables. If you want to fit separate distributions to different response variables in the same Generalized Regression report, you must use a script.

JMP PRO Continuous

Normal Y has a normal distribution with mean μ and standard deviation σ. The normal distribution is symmetric and with a large enough sample size, can approximate a large variety of other distributions using the Central Limit Theorem. The link function for μ is the identity. That is, the mean of Y is expressed as a linear model.

Note: When the specified Distribution is Normal, Standard Least Squares replaces the Maximum Likelihood Estimation method.

The scale parameter for the normal distribution is σ. When there is no penalty in the estimation method, the estimate of the scale parameter σ is the root mean square error (RMSE). The RMSE is the square root of the usual unbiased estimator of σ^2. The results shown are equivalent to a standard least squares fit unless censored observations are involved.

Note: The parameterization of nominal variables used in the Generalized Regression personality differs from their parameterization using the Standard Least Squares personality. Because of this difference, parameter estimates differ for models that contain nominal or ordinal effects.

See “Statistical Details for Distributions”.

Cauchy Y has a Cauchy distribution with location parameter μ and scale parameter σ. The Cauchy distribution has an undefined mean and standard deviation. The median and mode are both μ. Most data do not inherently follow a Cauchy distribution. However, it is useful for conducting a robust regression on data that contain a large proportion of outliers (up to 50%). The link function for μ is the identity. See "Statistical Details for Distributions".

t(5) Y has a Student's *t* distribution with 5 degrees of freedom, location parameter μ and scale parameter σ. The Student's *t* distribution is symmetric and is a robust option that spans the space between a normal distribution and a Cauchy distribution. As the degrees of freedom in the Student's *t* distribution approach infinity, the distribution is equivalent to the normal. When the degrees of freedom in the Student's *t* distribution equals 1, the distribution is equivalent to the Cauchy. The link function for μ is the identity. See "Statistical Details for Distributions".

Exponential Y has an exponential distribution with mean parameter μ. The exponential distribution is right-skewed and is often used to model lifetimes or the time between successive events. The link function for μ is the logarithm. See "Statistical Details for Distributions".

Gamma Y has a gamma distribution with mean parameter μ and dispersion parameter σ. The gamma is a flexible distribution and contains a family of other widely used distributions. For example, the exponential distribution is a special case of the gamma distribution where σ = μ. The chi-squared distribution can also be derived from the gamma distribution. The link function for μ is the logarithm. See "Statistical Details for Distributions".

Weibull Y has a Weibull distribution with mean parameter μ and scale parameter σ. The Weibull distribution is a flexible distribution and is often used to model lifetimes or the time until an event. The link function for μ is the identity. See "Statistical Details for Distributions".

LogNormal Y has a Lognormal distribution with mean parameter μ and scale parameter σ. The Lognormal distribution is right-skewed and is often used to model lifetimes or the time until an event. The link function for μ is the identity. See "Statistical Details for Distributions".

Beta Y has a beta distribution with mean parameter μ and dispersion parameter σ. The response for the beta is between 0 and 1 (not inclusive) and is often used to model proportions or rates. The link function for μ is the logit. See "Statistical Details for Distributions".

Quantile Regression Quantile regression models a specified conditional quantile of the response. No assumption is made about the form of the underlying distribution. When

you select Quantile Regression, a Quantile box appears beneath the Distribution menu. Specify the desired quantile.

If you specify 0.5 (the default) for the Quantile on the Model Dialog window, quantile regression models the conditional median of the response. Quantile regression is particularly useful when the rate of change in the conditional quantile, expressed by the regression coefficients, depends on the quantile. An advantage of quantile regression over least squares regression is its flexibility for modeling data with heterogeneous conditional distributions.

Quantile Regression is fit by minimizing an objective function using an iterative approach. For more information about quantile regression, see Koenker and Hallock (2001) and Portnoy and Koenker (1997).

When you choose Quantile Regression, Maximum Likelihood is the only available Estimation Method, and None is the only available Validation Method.

Note: If a quantile regression fit is time intensive, a progress bar appears. The progress bar shows the relative change in the objective function. When you click Accept Current Estimates, the calculation stops and the reported parameter estimates correspond to the best model fit at that point.

Cox Proportional Hazards The Cox proportional hazards model is a regression model for time-to-event data with predictors. It is based on a multiplicative relationship between the predictors and the hazard function. It can be used to examine the effect of predictors on survival times. The model involves an arbitrary baseline hazard function that is scaled by the predictors to give a general hazard function. The proportional hazards model produces parameter estimates and standard errors for each predictor. The Cox proportional hazards model was first proposed by D. R. Cox (1972). For more information about proportional hazards models, see Kalbfleisch and Prentice (2002).

When you choose Cox Proportional Hazards, the only available Validation Methods are BIC and AICc. Also, the Ridge Estimation Method is not available.

Note: When there are ties in the response, the Efron likelihood is used. See Efron (1977). This is a different method for handling ties than is used in the Proportional Hazard personality of the Fit Model platform or in the Fit Proportional Hazards platform.

JMP PRO Discrete

Binomial Y has a binomial distribution with parameters p and n. The response, Y, indicates the total number of successes in n independent trials with a fixed probability, p, for all trials. This distribution allows for the use of a sample size column. If no column is listed, it is assumed that the sample size is one. The link function for p is the logit. When you select

a binary response variable that has a Nominal modeling type, Binomial is the only available response distribution. See "Statistical Details for Distributions".

When you select Binomial as the Distribution, the response variable must be specified in one of the following ways.

- Unsummarized: If your data are not summarized as frequencies of events, specify a single binary column as the response. If this column has a modeling type of Nominal, you can designate one of the levels to be the Target Level. The default Target Level value is the higher of the two levels based on the order of the levels.
- Summarized with Freq column: If your data are summarized as frequencies of successes and failures, specify a single binary column as the response. If this column has a modeling type of Nominal, you can designate one of the levels to be the Target Level. The default Target Level value is the higher of the two levels based on the order of the levels. Assign the frequency column to the **Freq** role.
- Summarized with sample size column entered as second Y: If your data are summarized as frequencies of events (successes) and trials, specify two continuous columns as Y in this order: the count of the number of successes, and the count of the number of trials.

Note: When the specified Distribution is Binomial, Logistic Regression replaces the Maximum Likelihood Estimation method.

Beta Binomial Y has a beta binomial distribution with the probability of success, p, the number of trails, n, and overdispersion parameter, δ. This distribution is an overdispersed version of the binomial distribution.

Run demoBetaBinomial.jsl in the JMP Samples/Scripts folder to compare a beta binomial distribution with dispersion parameter δ to a binomial distribution with parameters p and $n = 20$.

The beta binomial distribution requires a sample size greater than one for each observation. Thus, the user must specify a sample size column. To insert a sample size column, specify two continuous columns as Y in this order: the count of the number of successes, and the count of the number of trials. The link function for p is the logit. See "Statistical Details for Distributions".

Multinomial Y has a multinomial distribution with three or more discrete levels. The response variable must have a nominal or ordinal modeling type. The model fits separate intercepts and effects parameters for each level of the response variable. If the response variable has k levels, the model contains k - 1 intercepts and effects parameters. The link function for the Multinomial distribution is the multinomial logit. See "Nominal Responses".

Ordinal Logistic Y has a multinomial distribution with ordinal levels. The response variable must have an ordinal modeling type. The model fits an intercept for each level of the response variable. The effects parameters are common across all levels of the response variable. The link function for the Ordinal Logistic distribution is the ordered logit. See "Ordinal Responses".

Note: The intercept parameterization for Ordinal Logistic in Generalized Regression differs from that in the Ordinal Logistic personality of Fit Model. The first Intercept term in Generalized Regression corresponds to the first Intercept term in the Ordinal Logistic personality. The subsequent Intercept terms in Generalized Regression are the successive differences between the intercept terms for the ordered levels of the response variable.

Poisson Y has a Poisson distribution with mean λ. The Poisson distribution typically models the number of events in a given interval and is often expressed as count data. The link function for λ is the logarithm. Poisson regression is permitted even if Y assumes non-integer values. See "Statistical Details for Distributions".

Negative Binomial Y has a negative binomial distribution with mean μ and dispersion parameter σ. The negative binomial distribution typically models the number of successes before a specified number of failures. The negative binomial distribution is also equivalent to the Gamma Poisson distribution under certain conditions. For more information about the connection between negative binomial and Gamma Poisson, see *Basic Analysis*.

Run demoGammaPoisson.jsl in the JMP Samples/Scripts folder to compare a Gamma Poisson distribution with mean λ and dispersion parameter σ to a Poisson distribution with mean λ.

The link function for μ is the logarithm. Negative binomial regression is permitted even if Y assumes non-integer values. See "Statistical Details for Distributions".

JMP PRO Zero-Inflated

ZI Binomial Y has a zero-inflated binomial distribution with parameters p, n, and zero-inflation parameter π. The response, Y, indicates the total number of successes in n independent trials with a fixed probability, p, for all trials. This distribution allows for the use of a sample size column. If no column is listed, it is assumed that the sample size is one. The link function for p is the logit. See "Statistical Details for Distributions".

ZI Beta Binomial Y has a beta binomial distribution with the probability of success, p, the number of trails, n, overdispersion parameter, δ, and zero-inflation parameter π. This distribution is an overdispersed version of the ZI binomial distribution. The ZI beta binomial distribution requires a sample size greater than one for each observation. Thus, the user must specify a sample size column. To insert a sample size column, specify two continuous columns as Y in this order: the count of the number of successes, and the count

of the number of trials. The link function for p is the logit. See "Statistical Details for Distributions".

ZI Poisson Y has a zero-inflated Poisson distribution with mean parameter λ and zero-inflation parameter π. The parameter λ is the conditional mean based on the observations coming from the Poisson distribution and not the inflating zeros. The link function for λ is the logarithm. ZI Poisson regression is permitted even if Y assumes no observed zeros or non-integer values. See "Statistical Details for Distributions".

ZI Negative Binomial Y has a zero-inflated negative binomial with location parameter μ, dispersion parameter σ, and zero-inflation parameter π. The parameter μ is the conditional mean based on the observations coming from the negative binomial distribution and not the inflating zeros. The link function for μ is the logarithm. ZI negative binomial regression is permitted even if Y assumes no observed zeros or non-integer values. See "Statistical Details for Distributions".

ZI Gamma Y has a zero-inflated gamma distribution with mean parameter μ and zero-inflation parameter π. Many times, we might believe that our nonzero responses are gamma distributed. This is true for insurance claims: claim values are approximately gamma distributed but there are also zeros in the data for policies that do not have any claims. The zero-inflated gamma could handle such data directly without having to split the data into zero and nonzero responses. The parameter μ is the conditional mean based on observations coming from the gamma distribution and not the inflating zeros. The link function for μ is the logarithm. See "Statistical Details for Distributions".

Table 6.1 gives the Data Types, Modeling Types, and other requirements for Y variables assigned the various distributions.

Table 6.1 Requirements for Y for Distributions

Distribution	Data Type	Modeling Type	Other
Normal	Numeric	Continuous	
Cauchy	Numeric	Continuous	
t(5)	Numeric	Continuous	
Exponential	Numeric	Continuous	Positive
Gamma	Numeric	Continuous	Positive
Weibull	Numeric	Continuous	Positive
LogNormal	Numeric	Continuous	Positive
Beta	Numeric	Continuous	Between 0 and 1

Table 6.1 Requirements for Y for Distributions *(Continued)*

Distribution	Data Type	Modeling Type	Other
Quantile Regression	Numeric	Continuous	
Cox Proportional Hazards	Numeric	Continuous	Nonnegative
Binomial, unsummarized	Any	Any	Binary
Binomial, summarized with Freq column	Any	Any	Binary
Binomial, summarized with count column entered as second Y	Numeric	Continuous	Nonnegative
Beta Binomial	Numeric	Continuous	Nonnegative
Multinomial	Any	Ordinal or Nominal	
Ordinal Logistic	Any	Ordinal	
Poisson	Numeric	Any	Nonnegative
Negative Binomial	Numeric	Any	Nonnegative
Zero-Inflated Binomial	Numeric	Any	Nonnegative
Zero-Inflated Beta Binomial	Numeric	Any	Nonnegative
Zero-Inflated Poisson	Numeric	Any	Nonnegative
Zero-Inflated Negative Binomial	Numeric	Any	Nonnegative
Zero-Inflated Gamma	Numeric	Continuous	Nonnegative

For more information about how these distributions are parameterized, see "Statistical Details for Distributions". Table 6.2 summarizes the details.

Table 6.2 Distributions, Parameters, and Link Functions

Distribution	Parameters	Mean Model Link Function
Normal	μ, σ	*Identity*(μ)
Cauchy	μ, σ	*Identity*(μ)
t(5)	μ, σ	*Identity*(μ)

Table 6.2 Distributions, Parameters, and Link Functions *(Continued)*

Distribution	Parameters	Mean Model Link Function
Exponential	μ	*Log*(μ)
Gamma	μ, σ	*Log*(μ)
Weibull	μ, σ	*Identity*(μ)
LogNormal	μ, σ	*Identity*(μ)
Beta	μ	*Logit*(μ)
Quantile Regression	μ	*Identity*(μ)
Cox Proportional Hazards	μ	*Log*(μ)
Binomial	*n*, *p*	*Logit*(*p*)
Beta Binomial	*n*, *p*, δ	*Logit*(*p*)
Multinomial	*n*, p_1, ..., p_k	*Multinomial Logit*(p_1, ..., p_k)
Ordinal Logistic	p_1, ..., p_{k-1}	*Ordinal Link*(p_1, ..., p_{k-1})
Poisson	λ	*Log*(μ)
Negative Binomial	μ, σ	*Log*(μ)
Zero-Inflated Binomial	*n*, *p*, π (zero-inflation)	*Logit*(*p*)
Zero-Inflated Beta Binomial	*n*, *p*, δ, π (zero-inflation)	*Logit*(*p*)
Zero-Inflated Poisson	λ, π (zero-inflation)	*Log*(μ)
Zero-Inflated Negative Binomial	μ, σ, π (zero-inflation)	*Log*(μ)
Zero-Inflated Gamma	μ, σ, π (zero-inflation)	*Log*(μ)

After selecting an appropriate Distribution, click **Run**. The Generalized Regression report window appears.

JMP PRO Generalized Regression Report Window

When you click **Run** in the Fit Model launch window, the Generalized Regression report window that appears contains the following items:

- A Model Comparison report that enables you to compare all of the models that have been fit in the report. Each time a new model is fit using the Model Launch control panel, it is added to the Model Comparison report. You can show or hide the reports for fitted models using the check boxes in the Show column. Other columns in the Model Comparison report contain information about each model that was fit, as well as fit statistics for each model. Click on the column headings to sort the models by any of the columns in the Model Comparison report. The first click sorts in ascending order; click the column heading a second time to sort in descending order.

Note: The Model Comparison report appears when one or more models have been fit.

- A Model Launch control panel for fitting models. See "Model Launch Control Panel". As you fit models, outlines are added with titles that describe the types of models that you have fit. See "Model Fit Reports" and "Model Fit Options".
  - If there are linear dependencies among the model terms, the Model Launch control panel contains a Singularity Details report that shows the linear functions that the model terms satisfy. See "Models with Linear Dependencies among Model Terms".
- A Maximum Likelihood report that shows the results of a model that was fit using maximum likelihood estimation. See "Maximum Likelihood". The Maximum Likelihood report appears only if the following conditions are met:
  - There are no linear dependencies among the predictors.
  - There are more observations than predictors.
  - There are no more than 250 predictors.

Note: When the specified Distribution is Normal, this report is labeled Standard Least Squares. When the specified Distribution is Binomial, this report is labeled Logistic Regression.

JMP PRO Generalized Regression Report Options

Model Dialog Shows the completed Fit Model launch window for the current analysis.

Set Random Seed Sets the seed for the randomization process used for KFold and Holdback validation. This is useful if you want to reproduce an analysis. Set the seed to a positive value, save the script, and the seed is automatically saved in the script. Running the script always produces the same cross validation analysis.

Save Coding Table Creates a new data table whose first columns show the JMP coding for all model parameters. The last column shows the values of the response variable. If you specified a sample size column for a binomial response, both the response and sample size columns are shown in the table. If you used two response columns to specify censoring, both response columns are shown in the table. If you used a response column that contains a Detection Limits column property to specify censoring, the response column in the coding table contains a Detection Limits column property.

Note: The coding data table contains a table variable called Original Data that gives the name of the data table that was used for the analysis. In the case where a By variable is specified, the Original Data table variable also gives the By variable and its level.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Model Launch Control Panel

The Model Launch control panel provides options for the following:

- "Response Distribution"
- "Estimation Method Options"
- "Advanced Controls"
- "Validation Method Options"
- "Early Stopping"
- "Go"

JMP PRO Response Distribution

The available response distributions in the control panel depend on attributes of the response variable specified in the Fit Model launch window. Appropriate response distributions are available in the Response Distribution option. See "Distribution" for more information about the response distributions.

JMP PRO Estimation Method Options

The available estimation methods can be grouped into techniques with no selection and no penalty, step-based model selection techniques, and penalized regression techniques.

The Maximum Likelihood, Standard Least Squares, and Logistic Regression methods fit the entire model that is specified in the Fit Model launch window. No variable selection is performed. These models can serve as baselines for comparison to other methods.

Note: Only one of Maximum Likelihood, Standard Least Squares, and Logistic Regression is available for a given report. The name of this estimation method depends on the Distribution specified in the Fit Model launch window.

The Backward Elimination, Forward Selection, Pruned Forward Selection, Best Subset, and Two Stage Forward Selection methods are based on variables entering or leaving the model at each step. However, they do not impose a penalty on the regression coefficients.

The Dantzig Selector, Lasso, Elastic Net, Ridge, and Double Lasso methods are penalized regression techniques. They shrink the size of regression coefficients and reduce the variance of the estimates, in order to improve predictive ability of the model.

Note: When your data are highly collinear, the adaptive versions of Lasso and Elastic Net might not provide good solutions. This is because the adaptive versions presume that the MLE provides a good estimate. The Adaptive option is not recommended in such cases.

Two types of penalties are used in these techniques:

- the l_1 penalty, which penalizes the sum of the *absolute values* of the regression coefficients
- the l_2 penalty, which penalizes the sum of the *squares* of the regression coefficients

The default Estimation Method for observational data is the Lasso. If the data table contains a DOE script and no singularities, the default Estimation Method is Forward Selection with the Effect Heredity option enabled. If the data table contains a DOE script and a singularity in the design matrix, the default Estimation Method is Two-Stage Forward Selection with the Effect Heredity option enabled.

The following methods are available for model fitting:

Estimation Methods with No Selection and No Penalty

Maximum Likelihood Computes maximum likelihood estimates (MLEs) for model parameters. No penalty is imposed. Maximum Likelihood is the only estimation method available for Quantile Regression. If you specified a Validation column in the Fit Model launch window, the maximum likelihood model is fit to the Training set. A maximum likelihood model report appears by default, as long as the following conditions are met:

- There are no linear dependencies among the predictors.
- There are more observations than predictors.
- There are no more than 250 predictors.

The Maximum Likelihood option gives you a way to construct classical models for the response distributions supported by the Generalized Regression personality. In addition, a model based on maximum likelihood can serve as a baseline for model comparison.

When the specified Distribution is Normal or Binomial, the Maximum Likelihood method is called Standard Least Squares or Logistic Regression, respectively.

Standard Least Squares When the Normal distribution is specified, the Maximum Likelihood estimation method is replaced with the Standard Least Squares estimation method. The default report is a Standard Least Squares report that gives the usual standard least squares results.

Logistic Regression When the Binomial distribution is specified, the Maximum Likelihood estimation method is replaced with the Logistic Regression estimation method. The default report is a Logistic Regression report. The logistic results are identical to maximum likelihood results.

Step-Based Estimation Methods

Note: Step-based estimation methods are not available when the specified Distribution is Multinomial.

Backward Elimination Computes parameter estimates using backward elimination regression. The model chosen provides the best solution relative to the selected Validation Method. Backward elimination starts by including all parameters in the model and removing one effect at each step until reaching the intercept-only model. At each step, the Wald tests for each parameter is used to determine which parameter is removed.

Caution: The horizontal axis of the Solution Path for Backward Elimination is the reverse of the same axis in other estimation methods. Therefore, as you move left to right in the Solution Path for the Backward Elimination estimation method, terms are being removed from the model, rather than added.

Forward Selection Computes parameter estimates using forward stepwise regression. At each step, the effect with the most significant score test is added to the model. The model chosen is the one that provides the best solution relative to the selected Validation Method.

When there are interactions and the Effect Heredity option is enabled, compound effects are handled in the following manner. If the effect with the most significant score test at a given step is one that would violate effect heredity, then a compound effect is created. The compound effect contains the effect with the most significant score test as well as any other inactive effects that are needed to satisfy effect heredity. If the compound effect has the most significant score test, then all of the effects in the compound effect are added to the model.

Pruned Forward Selection Computes parameter estimates using a mixture of forward and backward steps. The algorithm starts with an intercept-only model. At the first step, the effect with the most significant score test is added to the model. After the first step, the algorithm considers the following three possibilities at each step:

1. From the effects not in the model, add the effect that has the most significant score test.
2. From the effects in the model, remove the effect that has the least significant Wald test.
3. Do both of the above actions in a single step.

To choose the action taken at each step, the algorithm uses the specified Validation Method. For example, if the Validation Method is BIC, the algorithm chooses the action that results in the smallest BIC value. When there are interactions and the Effect Heredity option is enabled, compound effects are considered for adding effects, but they are not considered for removing effects.

When the model becomes saturated, the algorithm attempts a backward step to check if that improves the model. The maximum number of steps in the algorithm is 5 times the number of parameters. The model chosen is the one that provides the best solution relative to the selected Validation Method.

Pruned Forward Selection is an alternative to the Mixed Step option in the Stepwise Regression personality. However, it does not use the *p*-value to determine which variables enter or leave the model.

Tip: The Early Stopping option is not recommended for the Pruned Forward Selection Estimation Method.

Best Subset Computes parameter estimates by increasing the number of active effects in the model at each step. In each step, the model is chosen among all possible models with a number of effects given by the step number. The values on the horizontal axes of the Solution Path plots represent the number of active effects in the model. Step 0 corresponds to the intercept-only model. Step 1 corresponds to the best model of the ones that contain only one active effect. The steps continue up to the value of Max Number of Effects specified in the Advanced Controls in the Model Launch report. See "Advanced Controls".

Tip: The Best Subset Estimation Method is computationally intensive. It is not recommended for large problems.

Two Stage Forward Selection (Available only when there are second- or higher-order effects in the model.) Computes parameter estimates in two stages. In the first stage, a forward stepwise regression model is run on the main effects to determine which to retain in the model. In the second stage, a forward stepwise regression model is run on all of the higher-order effects that are composed entirely of the main effects chosen in the first stage. This method assumes strong effect heredity.

Terms that are not retained from the first stage still appear in the Parameter Estimates reports as zeroed terms. However, they are ignored in the fitting of the second stage model. Terms that are selected in the first stage are not forced into the second stage; they are available for selection in the second stage.

Penalized Estimation Methods

Dantzig Selector (Available only when the specified Distribution is Normal and the No Intercept option is not selected.) Computes parameter estimates by applying an l_1 penalty using a linear programming approach. See Candes and Tao (2007). The Dantzig Selector is useful for analyzing the results of designed experiments. For orthogonal problems, the Dantzig Selector and Lasso give identical results. See "Dantzig Selector".

Lasso Computes parameter estimates by applying an l_1 penalty. Due to the l_1 penalty, some coefficients can be estimated as zero. Thus, variable selection is performed as part of the fitting procedure. In the ordinary Lasso, all coefficients are equally penalized.

Adaptive Lasso Computes parameter estimates by penalizing a weighted sum of the absolute values of the regression coefficients. The weights in the l_1 penalty are determined by the data in such as way as to guarantee the oracle property (Zou 2006). This option uses the MLEs to weight the l_1 penalty. MLEs cannot be computed when the number of predictors exceeds the number of observations or when there are strict linear dependencies among the predictors. If MLEs for the regression parameters cannot be computed, a generalized inverse solution or a ridge solution is used for the l_1 penalty weights. See "Adaptive Methods".

The Lasso and the adaptive Lasso options generally choose parsimonious models when predictors are highly correlated. These techniques tend to select only one of a group of correlated predictors. High-dimensional data tend to have highly correlated predictors. For this type of data, the Elastic Net might be a better choice than the Lasso. See "Lasso Regression".

Elastic Net Computes parameter estimates by applying both an l_1 penalty and an l_2 penalty. The l_1 penalty ensures that variable selection is performed. The l_2 penalty improves predictive ability by shrinking the coefficients as ridge does.

Adaptive Elastic Net Computes parameter estimates using an adaptive l_1 penalty as well as an l_2 penalty. This option uses the MLEs to weight the l_1 penalty. MLEs cannot be computed when the number of predictors exceeds the number of observations or when there are strict linear dependencies among the predictors. If MLEs for the regression parameters cannot be computed, a generalized inverse solution or a ridge solution is used for the l_1 penalty weights. You can set a value for the Elastic Net Alpha in the Advanced Controls panel. See "Adaptive Methods".

The Elastic Net tends to provide better prediction accuracy than the Lasso when predictors are highly correlated. (In fact, both Ridge and the Lasso are special cases of the Elastic Net.) In terms of predictive ability, the adaptive Elastic Net often outperforms both the Elastic Net and the adaptive Lasso. The Elastic Net has the ability to select groups of correlated predictors and to assign appropriate parameter estimates to the predictors involved. See "Elastic Net".

Note: If you select an Elastic Net fit and set the Elastic Net Alpha to missing, the algorithm computes the Lasso, Elastic Net, and Ridge fits, in that order. If a fit is time intensive, a progress bar appears. When you click Accept Current Estimates, the calculation stops and the reported parameter estimates correspond to the best model fit at that point. The progress bar indicates when the algorithm is fitting Lasso, Elastic Net, and Ridge. You can use this information to decide when to click Accept Current Estimates.

Ridge Computes parameter estimates using ridge regression. Ridge regression is a biased regression technique that applies an l_2 penalty and does not result in zero parameter estimates. It is useful when you want to retain all predictors in your model. See "Ridge Regression".

Double Lasso Computes parameter estimates in two stages. In the first stage, a Lasso model is fit to determine the terms to be used in the second stage. In the second stage, a Lasso model is fit using the terms from the first stage. The Solution Path results and the parameter estimate reports that appear are for the second-stage fit. If none of the variables enters the model in the first stage, there is no second stage, and the results of the first stage appear in the report.

The Double Lasso is especially useful when the number of observations is less than the number of predictors. By breaking the variable selection and shrinkage operations into two stages, the Lasso in the second stage is less likely to overly penalize the terms that should be included in the model. The double lasso is similar to the relaxed lasso. The relaxed lasso is described in Hastie et al. (2009, p. 91).

Adaptive Double Lasso Computes parameter estimates in two stages. In the first stage, an adaptive Lasso model is fit to determine the terms to be used in the second stage. In the second stage, an adaptive Lasso model is fit using the terms from the first stage. The second stage considers only the terms that are included in the first stage model and uses weights based on the parameter estimates in the first stage. You can choose the method of calculating the weights using the Adaptive Penalty Weights option in the Advanced Controls. See "Advanced Control Options". The results that are shown are for the second-stage fit. If none of the variables enters the model in the first stage, there is no second stage, and the results of the first stage appear in the report. See "Adaptive Methods".

JMP PRO Advanced Controls

Use the Advanced Controls options to adjust various aspects of the model fitting process. A number of controls relate to the grid for the tuning parameter.

Tuning Parameter

The solution paths for the Lasso and Ridge Estimation Methods depend on a single tuning parameter. The solution path for the Elastic Net depends on a tuning parameter for the penalty on the likelihood as well as the Elastic Net Alpha. The penalty on the likelihood for the Elastic Net is a weighted sum of the penalties associated with the Lasso and Ridge Estimation Methods. The Elastic Net Alpha determines the weights of these two penalties. See "Statistical Details for Estimation Methods" and "Statistical Details for Advanced Controls".

When the tuning parameter is zero, the solution is unpenalized and maximum likelihood estimates are obtained. As the tuning parameter increases, the penalty increases.

The solution is the set of parameter estimates that minimizes the penalized negative log-likelihood function relative to the selected validation method. The current solution is designated by the solid red vertical line in the Solution Path Plots.

Note: The value of the tuning parameter increases as the Magnitude of Scaled Parameter Estimates in the Solution Path Plot decreases. Estimates close to the MLE are associated with large magnitudes and estimates that are heavily penalized are associated with small magnitudes.

It is important to be mindful of the following:

- When the tuning parameter is too small, the data are typically overfit and result in models with high variance.
- When the tuning parameter is too large, the data are typically underfit.

The Tuning Parameter Grid

To obtain a solution, the tuning parameter is increased over a fine grid.

- For the Lasso, Elastic Net with Elastic Net Alpha specified, and Ridge, the value of the tuning parameter that gives the solution is the one that provides the best fit over the grid of tuning parameters.

  Note: Elastic Net Alpha is set to 0.99 by default.

- If you do not set a value for the Elastic Net Alpha, the value of alpha is also increased over a fine grid. For a fixed value of the tuning parameter, alpha is varied until ten consecutive values of alpha fail to improve upon the best fit as determined by the validation method. This process is repeated for the entire grid of tuning parameter values. The final values of the tuning parameter and alpha are the values that provide the best fit over the grid of tuning parameters.

The grid of tuning parameter values ranges from zero, in most cases, to the smallest value for which all of the non-intercept terms are zero. Define the smallest value of the tuning parameter for which all non-intercept terms are zero to be its *upper bound*. The *lower bound* for the tuning parameter is zero except in the following two cases where it is set to 0.0001:

- If the design matrix is singular, the maximum likelihood estimates cannot be computed. The lower bound of 0.0001 allows estimates close to the MLEs to be computed.
- If the selected distribution is binomial or multinomial, the lower bound of 0.0001 helps prevent separation.

Advanced Control Options

Enforce effect heredity Requires lower-order effects to enter the model before their related higher order effects. In most cases, this means that X^2 is not in the model unless X is in the model. For estimation methods other than Forward Selection, however, it is possible for X^2 to enter the model and X to leave the model in the same step. If the data table contains a DOE script, this option is enabled, but it is off by default.

Elastic Net Alpha Sets the α parameter for the Elastic Net. This α parameter determines the mix of the l_1 and l_2 penalty tuning parameters in estimating the Elastic Net coefficients. The default value is α = 0.99, which sets the coefficient on the l_1 penalty to 0.99 and the coefficient on the l_2 penalty to 0.01. This option is available only when Elastic Net is selected as the Estimation Method. See "Statistical Details for Estimation Methods".

Number of Grid Points Specifies the number of grid points between the lower and upper bounds for the tuning parameter. At each grid point value, parameter estimates for that value of the tuning parameter are obtained. The default value is 150 grid points.

Minimum Penalty Fraction Indicates the minimum value for the ratio of the lower bound of the tuning parameter to its upper bound. When the lower bound for the tuning parameter is 0, the solution provides the MLE. In cases where you do not want to include the MLE or solutions very close to it, you can set the Minimum Penalty Fraction to a nonzero value. For the Double Lasso estimation method, the specified value of this option is used only in the first stage of the fit. When there is a singularity in the design matrix, the default value is 0.0001. Otherwise, the default value is 0.

Grid Scale Provides options for choosing the distribution of the grid scale. You can choose between a linear, square root, or log scale. Grid points equal in number to the specified Number of Grid Points are distributed according to the selected scale between the lower and upper bounds of the tuning parameter. The default grid scale is square root. See "Statistical Details for Advanced Controls".

First Stage Solution Provides options for choosing the solution in the first stage of the Double Lasso and Two Stage Forward Selection. By default, the solution that is the best fit according to the specified Validation Method is selected and is the solution initially shown (Best Fit). You can choose to initially display models with larger or smaller l_1 norm values that lie in the green or yellow zones. For example, if you choose Smallest in Yellow Zone, the initially displayed solution is the model in the yellow zone that has the smallest l_1 norm. See "Comparable Model Zones".

Max Number of Effects Specifies the maximum number of effects to consider in models for the Best Subset estimation method. You can use this to limit the number of computations needed to fit the model. The default value is 10.

Initial Displayed Solution Provides options for choosing the solution that is initially displayed as the current model in the Solution Path report. The current model is identified by a solid vertical line. See "Current Model Indicator". The best fit solution is identified by a dotted vertical line. By default, the displayed solution is the one that is considered the best fit according to the specified Validation Method.

You can choose to initially display models with larger or smaller l_1 norm values that still lie in the green or yellow zones. For example, if you choose Smallest in Yellow Zone, the initially displayed solution is the model in the yellow zone that has the smallest l_1 norm. See "Comparable Model Zones".

Adaptive Penalty Weights Provides options for the calculation of the penalty weights that are used in the second stage of the Adaptive Double Lasso. By default, the Inverse Solution option is selected. This option calculates the penalty weights using the parameter estimates from the first stage fit.

The Inverse Model Average option calculates the penalty weights using the parameter estimates from a solution that is the weighted average of the AICc or BIC models. The AICc models are used if the AICc Validation Method is selected. Otherwise, the BIC models are used. If you use the Inverse Model Average option, the maximum likelihood solution, if it exists, appears as the right-most point in the Solution Path for the Adaptive Double Lasso model.

Force Terms Enables you to select which terms, if any, you want to force into the model. The terms that are forced into the model are not included in the penalty.

JMP PRO Validation Method Options

The following methods are available for validation of the model fit.

Note: The only Validation Method allowed for Quantile Regression is None. The only Validation Methods allowed for the Maximum Likelihood Estimation Method are None and Validation Column. The only Validation Methods allowed for Cox Proportional Hazards are BIC, AICc, and None. The only Validation Methods allowed for the Dantzig Selector Estimation Method are BIC and AICc.

KFold For each value of the tuning parameter, the following steps are conducted:

- The observations are partitioned into k subsets, or *folds*.
- In turn, each fold is used as a validation set. A model is fit to the observations *not* in the fold. The log-likelihood based on that model is calculated for the observations *in* the fold, providing a *validation* log-likelihood.
- The mean of the validation log-likelihoods for the k folds is calculated. This value serves as a validation log-likelihood for the value of the tuning parameter.

The value of the tuning parameter that has the maximum validation log-likelihood is used to construct the final solution. To obtain the final model, all k models derived for the optimal value of the tuning parameter are fit to the entire data set. Of these, the model that has the highest validation log-likelihood is selected as the final model. The training set used for that final model is designated as the Training set and the holdout fold for that model is the Validation set. These are the Training and Validation sets used in plots and in the reported results for the final solution.

Holdback Randomly selects the specified proportion of the data for a validation set, and uses the other portion of the data to fit the model. The final solution is the one that

minimizes the negative log-likelihood for the validation set. This method is useful for large data sets. The random selection is based on stratified sampling across the model factors to attempt to create training and validation sets that are more balanced than ones based on simple random sampling.

Leave-One-Out Performs leave-one-out cross validation. This is equivalent to KFold, with the number of folds equal to the number of rows. This option should not be used on moderate or large data sets. It can require long processing time for even a moderate number of observations. The Training and (one-row) Validation sets used in plots and in the reported results for the final solution are determined as is done for KFold validation.

BIC Minimizes the Bayesian Information Criterion (BIC) over the solution path. See "Likelihood, AICc, and BIC".

AICc Minimizes the corrected Akaike Information Criterion (AICc) over the solution path. AICc is the default setting for Validation Method. See "Likelihood, AICc, and BIC".

Note: The AICc is not defined when the number of parameters approaches or exceeds the sample size.

ERIC Minimizes the Extended Regularized Information Criterion (ERIC) over the solution path. See "Model Fit Detail". Available only for exponential family distributions and for the Lasso and adaptive Lasso estimation methods.

None Does not use validation. Available only for the Maximum Likelihood Estimation Method and Quantile Regression.

Validation Column Uses the column specified in the Fit Model window as having the Validation role. The final solution is the one that minimizes the negative log-likelihood for the validation set. This option is not available when the specified Estimation Method is Dantzig Selector or when the specified Distribution is Quantile Regression or Cox Proportional Hazards.

JMP PRO Early Stopping

Early Stopping adds an early stopping rule:

- For Forward Selection, the algorithm terminates when 10 consecutive steps of adding variables to the model fail to improve upon the validation measure. The solution is the model at the step that precedes the 10 consecutive steps.
- For Lasso, Elastic Net, and Ridge, the algorithm terminates when 10 consecutive values of the tuning parameter fail to improve upon the best fit as determined by the validation method. The solution is the estimate corresponding to the tuning parameter value that precedes the 10 consecutive values.

Note: For the AICc and BIC validation methods, early stopping does not occur until at least four predictors have entered the model.

JMP PRO Go

When you click **Go**, a report opens. The title of the report specifies the response distribution, the estimation method, and the validation method that you selected. You can return to the Model Launch control panel to perform additional analyses and choose other response distributions, estimation methods, and validation methods.

JMP PRO Model Fit Reports

For each Estimation Method and Validation Method that you specify in the Model Launch panel, a report is produced. The report specifies your selected Distribution, Estimation method, and Validation method in its title.

The following reports are presented by default:

- Regression Plot
- Model Summary
- Estimation Details (shown only for Lasso, Elastic Net, and Ridge)
- Solution Path (shown for all but the Maximum Likelihood Estimation Method and Quantile Regression)
- Parameter Estimates for Original Predictors
- Effect Tests

JMP PRO Regression Plot

Note: The Regression Plot report appears only when there is one continuous predictor and no more than one categorical predictor. It is not available if the Distribution option is set to Multinomial, Ordinal Logistic, or Cox Proportional Hazards. The response must be continuous.

The Regression Plot report shows a plot of the response values on the vertical axis and the continuous predictor on the horizontal axis. A regression line is shown over the points. If there is a categorical predictor in the model, each level of the categorical predictor has a separate regression line and a legend appears next to the plot. If the response is specified as the counts of the number of successes and the number of trials, the number of successes divided by the number of trials is plotted on the vertical axis.

JMP PRO Model Summary

The Model Summary report describes the model that you have fit and provides summary information about the fit itself.

JMP PRO Model Description Detail

The first part of the Model Summary report gives information that describes the model that you have fit.

Response The column assigned to the Y role in the Fit Model window. When two columns are used to specify interval censoring, both column names are listed.

Distribution The Distribution selected in the Fit Model window. For Quantile Regression, the value of the specified quantile for the response is also displayed.

Estimation Method The Estimation Method selected in the Model Launch panel.

Validation Method The Validation Method selected in the Model Launch panel.

Mean Model Link The link function for the model for the mean, based on the Distribution selected in the Fit Model window.

Location Model Link The link function for the model for the location parameter, shown when either Cauchy or t(5) is selected as the Distribution in the Fit Model window.

Scale Model Link The link function for the model for the scale parameter, based on the Distribution selected in the Fit Model window.

Probability Model Link The link function for the model for the probability, based on the Distribution selected in the Fit Model window.

Dispersion Model Link The link function for the model for the dispersion parameter, based on the Distribution selected in the Fit Model window.

Zero Inflation Model Link The link function for the model for the zero inflation parameter, based on the Distribution selected in the Fit Model window.

Lower Detection Limit The lower detection limit specified in a Detection Limits column property assigned to the response column.

Upper Detection Limit The upper detection limit specified in a Detection Limits column property assigned to the response column.

Censor Column The column assigned to the Censor role in the Fit Model window.

Censor Code The value in the Censor column that designates right-censored observations. This is the value that was specified in the Censor Code list in the Fit Model window.

JMP PRO Model Fit Detail

The second part of the Model Summary report gives statistics related to the model fit. If either Holdback or Validation Column is selected as the Validation Method, these statistics are computed separately for the training and validation sets. This part of the Model Summary report is not available if either KFold or Leave-One-Out is selected as the Validation Method.

Number of rows The number of rows.

Sum of Frequencies The sum of the values of a column assigned to the Freq or Weight role in the Fit Model window.

Note: For -LogLikelihood, BIC, AICc, and ERIC, smaller is better. See "Likelihood, AICc, and BIC".

-LogLikelihood The negative of the natural logarithm of the likelihood function for the current model. See "Likelihood, AICc, and BIC".

Note: -LogLikelihood is not available for Quantile Regression.

Objective Function (Available only for Quantile Regression.) The value of the function that is minimized to fit the specified quantile regression model. The function that is minimized is the check-loss function.

Number of Parameters The number of nonzero parameters in the current model.

BIC The Bayesian Information Criterion, which is defined as follows:

$$\text{BIC} = -2LogLikelihood + k\ln(n)$$

See "Likelihood, AICc, and BIC".

AICc The corrected Akaike Information Criterion, which is defined as follows:

$$\text{AICc} = -2LogLikelihood + 2k + 2k(k+1)/(n-k-1)$$

See "Likelihood, AICc, and BIC".

ERIC (Available only for exponential family distributions and when the Lasso or adaptive Lasso estimation method is specified.) The Extended Regularization Information Criterion. See Hui et al. (2015). ERIC is defined as follows:

$$\text{ERIC} = -2LogLikelihood + (k-2)\ln(n\phi/\lambda)$$

where λ is the value of the tuning parameter and ϕ is the nuisance parameter.

Generalized RSquare (Not available for Quantile Regression.) An extension of the RSquare measure that can be applied to general regression models. Generalized RSquare compares the likelihood of the fitted model (L_M) to the likelihood of the intercept-only (constant) model (L_0). It is scaled to have a maximum of 1. For distributions other than Binomial, the Generalized RSquare is defined as follows:

$$\text{Generalized RSquare} = 1 - (L_0/L_M)^{2/n}$$

When Binomial is the specified distribution, the Generalized RSquare is defined as follows:

$$\text{Generalized RSquare} = \frac{1 - (L_0/L_M)^{2/n}}{1 - L_0^{2/n}}$$

A Generalized RSquare value of 1 indicates a perfect model; a value of 0 indicates a model that is no better than a constant model. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991).

Note: Generalized RSquare is replaced by RSquare when the Normal distribution is specified.

Caution: You should not compare Generalized RSquare values for models that use different response distributions. The comparison being made is to the intercept-only model with a given response distribution.

RSquare (Available only when the Normal distribution is specified.) Estimates the proportion of variation in the response that can be attributed to the model rather than to random error. An RSquare value of 1 indicates a perfect model; a value of 0 indicates a model that is no better than a constant model. The RSquare value is calculated as follows:

$$1 - \frac{\sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{N} (Y_i - \bar{Y})^2}$$

RSquare Adj (Available only when the Normal distribution is specified and the estimation method does not involve a penalty.) Adjusts the RSquare statistic for the number of parameters in the model. Rsquare Adj facilitates comparisons among models with different numbers of parameters. The computation uses the degrees of freedom. The RSquare Adj value is calculated as follows:

$$1 - \frac{\frac{1}{N-(p-1)} \sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2}{\frac{1}{N-1} \sum_{i=1}^{N} (Y_i - \bar{Y})^2}$$

where N is the number of observations and p is the number of parameters.

Note: When there is a Validation set, the adjusted RSquare statistic is reported only for the Training set.

RASE (Available only when the Normal distribution is specified.) The Root Average Square Error (RASE) is the square root of the mean squared prediction error in the current model. See "RASE".

Lambda Penalty (Available only for the Dantzig Selector, Lasso, Elastic Net, Ridge, and Double Lasso estimation methods.) The value of the tuning parameter λ for the current model. See "Statistical Details for Estimation Methods".

JMP PRO Estimation Details

The Estimation Details report shows the settings of the Advanced Controls for the Dantzig Selector, Lasso, Elastic Net, Ridge, and Double Lasso estimation methods. For more information about these controls, see "Advanced Controls".

JMP PRO Solution Path

Note: The Solution Path report appears for all Estimation Methods except Maximum Likelihood and all Distributions except Quantile Regression.

The Solution Path report shows two plots:

- The Solution Path Plot displays values of the estimated parameters.
- The Validation Plot displays values of the validation statistic corresponding to the selected validation method.

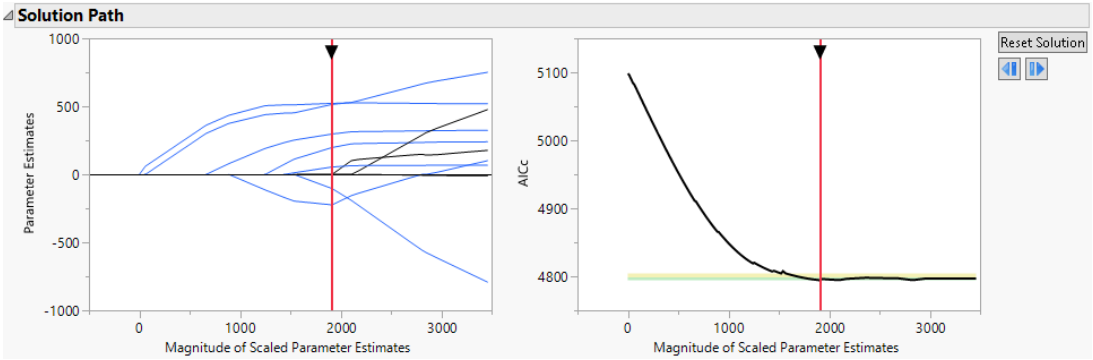
The horizontal scaling for both plots is given in terms of the Magnitude of Scaled Parameter Estimates. This is the l_1 norm, defined as the sum of the absolute values of the scaled parameter estimates for the model for the mean. (Estimates corresponding to the intercept, dispersion parameters, and zero-inflation parameters are excluded from the calculation of the l_1 norm.) Note the following:

- Estimates with large values of the l_1 norm are close to the MLE.
- Estimates with small values of the l_1 norm are heavily penalized.
- The value of the tuning parameter increases as the l_1 norm decreases.

JMP PRO Current Model Indicator

A solid vertical red line is placed in both plots at the value of the l_1 norm for the solution displayed in the Parameter Estimates for Original Predictors report. You can drag the arrow at the top of the vertical red line in either plot to change the magnitude of the penalty, indicating a new current model. In the Validation Plot, you can also click anywhere in the plot to change the model. As you drag the vertical red line to indicate a new model, the results in the report update to reflect the currently selected model. A dashed vertical line remains at the best fit model. You can click the **Reset Solution** button next to the Validation Plot to return the vertical red line and corresponding results to the initial solution. For some validation methods, the Validation Plot provides zones that identify comparable models. See "Comparable Model Zones".

Figure 6.5 Solution Path Report for Diabetes.jmp, Lasso with AICc Validation



For more information about the Solution Path Plot, see "Solution Path Plot". For more information about the Validation Plot, see "Validation Plot".

JMP PRO Solution Path Plot

You can select paths in the Solution Path Plot to highlight the corresponding terms in the Parameter Estimates reports. This action also selects the corresponding columns in the data table. Selecting rows in either of the reports highlights the corresponding rows in the other report and the corresponding paths in the Solution Path Plot. Press Shift and click to select multiple paths or rows.

The Parameter Estimates are plotted using the vertical axis of the Solution Path Plot. These are the *scaled* parameter estimates. They are derived for a model expressed in terms of centered and scaled predictors. See "Parameter Estimates for Centered and Scaled Predictors".

When the number of predictors is less than the number of observations, the Solution Path Plot usually shows the entire range of estimates from zero to the unpenalized fit given by the MLE. Otherwise, the plot extends to a magnitude that is close to the unpenalized solution. This occurs when the jump from the next-to-last grid point to the MLE solution is so large that the detail for solutions up to the next-to-last grid point is obscured. When this happens, as long as the MLE is not the final solution, the Solution Path Plot is rescaled so that the axis extends only to the next-to-last grid point.

JMP PRO The Solution ID

Internally, each solution in the Solution Path is assigned a Solution ID. When you adjust the tuning parameter to select a solution other than the one initially presented, the corresponding Solution ID appears in scripts created by the Save Script options. The Solution ID is the value N in the `Set Solution ID( N )` command. Saving the Solution ID ensures that you can re-create your selected solution when you run the script.

JMP PRO Validation Plot

The Validation Plot shows plots of statistics that describe how well models fit across the values of the tuning parameter, or equivalently, across the values of the Magnitude of the Scaled Parameter Estimates. The statistics plotted depend on the selected Validation Method. For each Validation Method, Table 6.3 lists the statistic that is plotted. For all validation methods, smaller values are better. For the KFold and Leave-One-Out validation methods, and for a Validation Column with more than three values, the statistic that is plotted is the mean of the scaled negative log-likelihood values across the folds.

The *Scaled -LogLikelihood* in Table 6.3 is the negative log-likelihood divided by the number of observations in the set for which the negative log-likelihood is computed.

Table 6.3 Validation Methods and Corresponding Validation Statistics and Zones

Validation Method	Validation Statistic	Tuning Parameter Regions
KFold	Mean of the Scaled -LogLikelihood values across the *K* folds	Two
Holdback	Scaled -LogLikelihood	None
Leave-One-Out	Mean of the Scaled -LogLikelihood values across all folds	Two
BIC	BIC for training data	Two
AICc	AICc for training data	Two
ERIC	ERIC for training data	Two
Validation Column with two or three values	Scaled -LogLikelihood	None
Validation Column with $K > 3$ values	Mean of the Scaled -LogLikelihood values across the *K* folds	Two

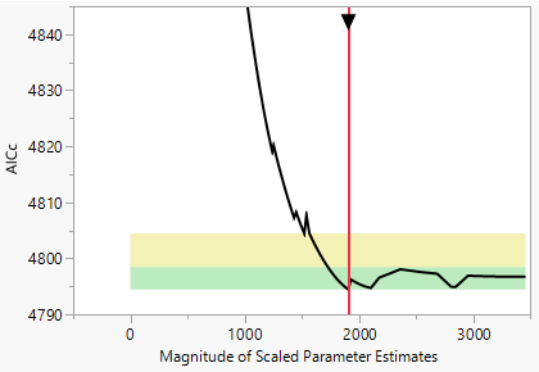
JMP PRO Comparable Model Zones

Although a model is estimated to be the best model, there can be uncertainty relative to this selection. Competing models might fit nearly as well and can contain useful information. For the AICc, BIC, KFold, and Leave-One-Out validation methods, and for a Validation Column with more than three values, the Validation Plot provides zones that identify competing models that might deserve consideration. Models that fall outside the zones are not recommended. See Burnham and Anderson (2004) and Burnham et al. (2011).

A zone is an interval of values of the validation statistics. The zones are plotted as green or yellow rectangles that span the entire horizontal axis. A model falls in a zone if the value of its validation statistic falls in the zone. You can drag the solid vertical red line to explore solutions within the zones. See "Current Model Indicator".

Figure 6.6 shows a Validation Plot for Diabetes.jmp with the vertical axis expanded to show the two zones.

Figure 6.6 Validation Plot for Diabetes.jmp, Lasso with AICc Validation



Zones for BIC, AICc, and ERIC Validation

For these validation methods, two regions are shown in the plot. Denote the validation BIC, AICc, and ERIC values for the best solutions by V^{best}.

- The green zone identifies models for which there is strong evidence that a model is comparable to the best model. The green zone is the interval [V^{best}, V^{best}+4].
- The yellow zone identifies models for which there is weak evidence that a model is comparable to the best model. The yellow zone is the interval (V^{best}+4, V^{best}+10].

Zones for KFold Validation, Leave-One-Out Validation, and Validation Column with More Than Three Values

For these validation methods, two regions are shown in the plot. At the solution for the best model, the scaled negative log-likelihood functions are evaluated for each validation set. Denote the standard error of these values as L^{SE}. Denote the scaled negative log-likelihood for the best solution by L^{best}.

- The green zone identifies models for which there is strong evidence that a model is comparable to the best model. The green zone is the interval $[L^{best}, L^{best}+L^{SE}]$.
- The yellow zone identifies models for which there is weak evidence that a model is comparable to the best model. The yellow zone is the interval $(L^{best}+L^{SE}, L^{best}+2.5*L^{SE}]$.

JMP PRO Parameter Estimates for Centered and Scaled Predictors

The Parameter Estimates for Centered and Scaled Predictors report gives estimates and other results for all parameters in the model. The initial table includes the coefficients for the predictors in the model. An additional table includes other model parameters such as scale, dispersion, or zero inflation parameters. See "Distribution". Both tables include the same columns of results.

Tip: You can click terms in the Parameter Estimates for Centered and Scaled Predictors report to highlight the corresponding paths in the Solution Path Plot. The corresponding columns in the data table are also selected. This is useful in terms of running further analyses. Press Shift and click the terms to select multiple rows.

For all fits in the Generalized Regression personality, every predictor is centered to have mean zero and scaled to have sum of squares equal to one:

- The mean is subtracted from each observation.
- Each difference is then divided by the square root of the sum of the squared differences from the mean.

This puts all predictors on an equal footing relative to the penalties applied.

Note: When the No Intercept option is selected in the launch window, the predictors are not centered and scaled.

The Parameter Estimates for Centered and Scaled Predictors report gives parameter estimates for the model expressed in terms of the centered and scaled predictors. The estimates are determined by the Validation Method that you specified. The estimates are depicted in the Solution Path Plots by a vertical red line.

The report provides the following information:

Term A list of the model terms. "Forced in" appears next to any terms that were forced into the model using the Advanced Controls option.

Estimate The parameter estimate corresponding to the centered and scaled model term.

Std Error The standard error of the estimate. This is obtained using M-estimation and a sandwich formula (Zou 2006; Huber and Ronchetti 2009).

Wald ChiSquare The ChiSquare value for a Wald test of whether the parameter is zero.

Prob > ChiSquare The *p*-value for the Wald test.

Lower 95% The lower bound for a 95% confidence interval for the parameter. You can change the α level in the Fit Model window by selecting Set Alpha Level from the Model Specification red triangle menu.

Upper 95% The upper bound for a 95% confidence interval for the parameter. You can change the α level in the Fit Model window by selecting Set Alpha Level from the Model Specification red triangle menu.

Singularity Details (Available only if there are linear dependencies among the model terms.) The linear function that the model term satisfies.

JMP PRO Parameter Estimates for Original Predictors

The Parameter Estimates for Original Predictors report gives estimates and other results for all parameters in the model. The initial table includes the coefficients for the predictors in the model. An additional table includes other model parameters such as scale, dispersion, or zero inflation parameters. See "Distribution". Both tables include the same columns of results.

Tip: You can click terms in the Parameter Estimates for Original Predictors report to highlight the corresponding paths in the Solution Path Plot. The corresponding columns in the data table are also selected. This is useful when running further analyses. Press Shift and click the terms to select multiple rows.

The Parameter Estimates for Original Predictors report gives parameter estimates for the model expressed in terms of the original (uncentered and unscaled) predictors.

The report provides the following information:

Term A list of the model terms. "Forced in" appears next to any terms that were forced into the model using the Advanced Controls option.

Estimate The parameter estimate corresponding to the model term given in terms of the original measurements.

Std Error The standard error of the estimate. This is obtained using M-estimation and a sandwich formula (Zou 2006 and Huber and Ronchetti 2009).

Wald ChiSquare The ChiSquare value for a Wald test of whether the parameter is zero.

Prob > ChiSquare The *p*-value for the Wald test.

Lower 95% The lower bound for a 95% confidence interval for the parameter. You can change the α level in the Fit Model window by selecting Set Alpha Level from the Model Specification red triangle menu.

Upper 95% The upper bound for a 95% confidence interval for the parameter. You can change the α level in the Fit Model window by selecting Set Alpha Level from the Model Specification red triangle menu.

Singularity Details (Available only if there are linear dependencies among the model terms.) The linear function that the model term satisfies.

VIF (Available only when the distribution is Normal. Appears only if you right-click in the parameter estimates table and select Columns > VIF.) The variance inflation factor (VIF) for each term in the model. High VIF values indicate a collinearity issue among the terms in the model.

The VIF for the i^{th} term, x_i, is calculated as follows:

$$VIF_i = \frac{1}{1 - R_i^2}$$

where $R_i{}^2$ is the RSquare for the regression of x_i as a function of the other explanatory variables.

JMP PRO Active Parameter Estimates

The Active Parameter Estimates report gives a subset of the Parameter Estimates for Original Predictors report. The Active Parameter Estimates report shows only the nonzero parameters.

JMP PRO Effect Tests

The Effect Tests report gives the following information:

Source A list of the effects in the model.

Nparm The number of parameters associated with the effect.

DF The degrees of freedom for the Wald ChiSquare test. This is the number of nonzero parameter estimates associated with the effect in the model.

Wald ChiSquare The ChiSquare value for a Wald test of whether all parameters associated with the effect are zero.

Prob > ChiSquare The *p*-value for the Wald ChiSquare test.

The following columns appear in the report in place of Wald ChiSquare and Prob > ChiSquare when the Distribution is Normal and there is no penalty in the estimation method:

Sum of Squares The sum of squares for the hypothesis that the effect is zero.

F Ratio The F statistic for testing that the effect is zero. The F Ratio is the ratio of the mean square for the effect divided by the mean square for error. The mean square for the effect is the sum of squares for the effect divided by its degrees of freedom.

Prob > F The *p*-value for the effect test.

If the coefficient for an effect has been estimated as zero, then:

- If the effect has one degree of freedom, the word "Removed" appears at the far right in the row for that effect.
- If the effect has multiple degrees of freedom, the phrase "Levels removed" appears, followed by the number of levels that correspond to terms with parameter estimates of zero.

JMP PRO Model Fit Options

Each model fit report has a red triangle menu with these options:

Caution: Many options in the platform are not available if you specify a column that has the Expression data type or Vector modeling type in the launch window.

Regression Reports Enables you to customize the reports that are shown for the specified model fit. All of the following reports are shown by default except for the Parameter Estimates for Centered and Scaled Parameter Estimates report and the Active Parameter Estimates report.

Model Summary Shows or hides the Model Summary report that includes information about the specification and goodness of fit statistics for the model. This option also displays the Estimation Details report for applicable models. See "Model Summary" and "Estimation Details".

Solution Path (Not available for Maximum Likelihood models.) Shows or hides the Solution Path and Validation Path plots. See "Solution Path".

Parameter Estimates for Centered and Scaled Predictors Shows or hides a table of centered and scaled parameter estimates. See "Parameter Estimates for Centered and Scaled Predictors".

Parameter Estimates for Original Predictors Shows or hides a table of parameter estimates in the original scale of the data. See "Parameter Estimates for Original Predictors".

Active Parameter Estimates (Not available for Maximum Likelihood or Ridge Regression models.) Shows or hides a table of active, or nonzero, parameter estimates for the currently selected model.

Show Solution Path Summary (Not available for Maximum Likelihood or Ridge Regression models.) Shows or hides a report that contains a table of fit statistics for the points on the Solution Path and Validation Path plots where the active set changes. The statistics that are available depend on the estimation method. For more information about the conditional model probabilities that are available for Normal Lasso with BIC Validation models, see Hu et al. (2019). When BIC, AICc, or ERIC Validation is specified, the cells in the BIC and AICc columns are colored in the same manner as the Validation Plot. See "Comparable Model Zones".

Effect Tests Shows or hides tests for each effect. The effect test for a given effect tests the null hypothesis that all parameters associated with that effect are zero. A nominal or ordinal effect can have several associated parameters, based on its number of levels. The effect test for such an effect tests whether all of the associated parameters are zero. When the Distribution is Multinomial, the effects are combined over the levels of the response. See "Effect Tests".

Show Prediction Expression Shows or hides the Prediction Expression report that contains the equation for the estimated model. See "Show Prediction Expression" for an example.

Select Nonzero Terms Highlights terms with nonzero coefficients in the report. Also selects all associated columns in the data table. This option is not available when Ridge Regression is selected as the Estimation Method.

Select Zeroed Terms Highlights terms with zero coefficients in the report. Also selects all associated columns in the data table. This option is not available when Ridge Regression is selected as the Estimation Method.

Relaunch Active Set (Not available for models that contain a predictor that has the Vector modeling type.) Contains options that open a Fit Model launch window where the Construct Model Effects list contains a set of terms based on the terms that have nonzero parameter estimates. These terms are the *active* effects. All other specifications in the launch window are those used in the original analysis.

Note: If you select any of the Relaunch Active Set options in a report that contains a By variable, the By variable is not added to the Fit Model launch window.

Relaunch with Active Effects Populates the Construct Model Effects list only with the active effects.

Relaunch Active Main Effects and Second Degree Factorial Populates the Construct Model Effects list with a second degree factorial constructed with the active effects.

Relaunch Active Main Effects and Third Degree Factorial Populates the Construct Model Effects list with a third degree factorial constructed with the active effects.

Relaunch Active Main Effects and Full Factorial Populates the Construct Model Effects list with a full factorial constructed with the active effects.

Relaunch Active Main Effects and Second Degree Polynomial Populates the Construct Model Effects list with a second degree polynomial constructed with the active effects.

Relaunch Active Main Effects and Third Degree Polynomial Populates the Construct Model Effects list with a third degree polynomial constructed with the active effects.

Relaunch Active Main Effects and Response Surface Model Populates the Construct Model Effects list with a response surface model constructed with the active effects.

Hide Inactive Paths Adjusts the transparency of the inactive paths in the Solution Path Parameter Estimates plot so that the paths that are not currently active appear faded.

Odds Ratios (Available only when the specified Distribution is Binomial and the model contains an intercept. Not available for models that contain a predictor that has the Vector modeling type.) Displays a report that contains odds ratios for categorical predictors, and unit odds ratios and range odds ratios for continuous predictors. An *odds ratio* is the ratio of the odds for two events. The *odds* of an event is the probability that the event of interest occurs versus the probability that it does not occur. The event of interest is defined by the Target Level in the Fit Model launch window.

For each categorical predictor, an Odds Ratios report appears. Odds ratios are shown for all combinations of levels of a categorical model term.

If there are continuous predictors, two additional reports appear:

– Unit Odds Ratios Report. The *unit odds ratio* is calculated over a one-unit change in a continuous model term.

– Range Odds Ratios Report. The *range odds ratio* is calculated over the entire range of a continuous model term.

The confidence intervals in the Odds Ratios report are Wald-based intervals. Note that the odds ratio for a model term is meaningful only if the model term is not involved in any higher-order effects.

Note: If there are interactions in the model, you can use the Multiple Comparisons option to obtain odds ratios. See "Multiple Comparisons".

Incidence Rate Ratios (Available only when the specified Distribution is Poisson or Negative Binomial and the model contains an intercept.) Displays a report that contains incidence rate ratios for categorical predictors, and unit incidence rate ratios and range incidence rate ratios for continuous predictors. An *incidence rate ratio* is the ratio of the incidence rate for two events. The *incidence rate* for a model term is the number of new events that occur over a given time period.

For each categorical predictor, an Incidence Rate Ratios report appears. Incidence rate ratios are shown for all combinations of levels of a categorical model term.

If there are continuous predictors, two additional reports appear:

– Unit Incidence Rate Ratios Report. The *unit incidence rate ratio* is calculated over a one-unit change in a continuous model term.

– Range Incidence Rate Ratios Report. The *range incidence rate ratio* is calculated over the entire range of a continuous model term.

The confidence intervals in the Incidence Rate Ratios report are Wald-based intervals. Note that the incidence rate ratio for a model term is meaningful only if the model term is not involved in any higher-order effects.

Hazard Ratios (Available only when the specified Distribution is Cox Proportional Hazards.) Displays a report that contains hazard ratios for categorical predictors, and unit hazard ratios and range hazard ratios for continuous predictors. A *hazard ratio* is the ratio of the hazard rate for two events. The *hazard rate* at time t for an event is the conditional probability that the event will not survive an additional amount of time, given that it has survived to time t.

For each categorical predictor, a Hazard Ratios report appears. Hazard ratios are shown for all combinations of levels of a categorical model term.

If there are continuous predictors, two additional reports appear:

– Unit Hazard Ratios Report. The *unit hazard ratio* is calculated over a one-unit change in a continuous model term.

– Range Hazard Ratios Report. The *range hazard ratio* is calculated over the entire range of a continuous model term.

The confidence intervals in the Hazard Ratios report are Wald-based intervals. Note that the hazard ratio for a model term is meaningful only if the model term is not involved in any higher-order effects.

Covariance of Estimates Displays a matrix showing the covariances of the parameter estimates. These are calculated using M-estimation and a sandwich formula (Zou 2006 and Huber and Ronchetti 2009). The covariance matrix does not contain zeroed terms.

Correlation of Estimates Displays a matrix showing the correlations of the parameter estimates. These are calculated using M-estimation and a sandwich formula (Zou 2006 and Huber and Ronchetti 2009). The correlation matrix does not contain zeroed terms.

Inverse Prediction (Not available for models that contain a predictor that has the Vector modeling type.) Predicts an X value, given specific values for Y and the other X variables. This can be used to predict continuous variables only. For more information about Inverse Prediction, see "Inverse Prediction".

Multiple Comparisons (Not available for models that contain a predictor that has the Vector modeling type or for models that do not contain any categorical predictors.) Displays the Multiple Comparisons launch window. For more information about the Multiple Comparisons launch window and report, see "Multiple Comparisons". Note that the multiple comparisons are performed on the linear predictor scale. When the specified Distribution is Binomial, the multiple comparisons are performed on the odds ratios. When the specified Distribution is Poisson, the multiple comparisons are performed on the incidence rate ratios. When the specified Distribution is Cox Proportional Hazards, the multiple comparisons are performed on the hazard ratios.

Confusion Matrix (Available only when the specified Distribution is Binomial, Multinomial, or Ordinal Logistic.) Displays a matrix that tabulates the actual response levels and the predicted response levels. For a good model, predicted response levels should be the same as the actual response levels. The confusion matrix enables you to assess how the predicted responses align with the actual responses. If you used validation, a confusion matrix is shown for each of the Training, Validation, and Test sets.

Set Probability Threshold (Available only when the specified Distribution is Binomial.) Specify a cutoff probability for classifying the response. By default, an observation is classified into the Target Level when its predicted probability exceeds 0.5. Change the threshold to specify a value other than 0.5 as the cutoff for classification into the Target Level. The Predicted Rate in the confusion matrix is updated to reflect classification according to the specified threshold.

If the response has a Profit Matrix column property, the initial value for the probability threshold is determined by the profit matrix.

Profilers (Not available for models that contain a predictor that has the Vector modeling type.) Provides various profilers that enable you to explore the fitted model.

Note: When the number of rows is less than or equal to 500 and the number of predictors is less than or equal to 30, the Profiler plots update continuously as you drag the current model indicator in either Solution Path plot. Otherwise, they update when you release the mouse button.

Profiler Displays the Prediction Profiler. Predictors that have parameter estimates of zero and that are not involved in any interaction terms with nonzero coefficients do not appear in the profiler. For more information about the prediction profiler, see *Profilers*.

Distribution Profiler (Not available when the specified distribution is Binomial or Quantile Regression.) Displays a profiler of the cumulative distribution function of the predictors and the response. The response is shown in the right-most cell.

Quantile Profiler (Not available when the specified distribution is Binomial or Quantile Regression.) Displays a profiler that shows the predicted response as a function of the predictors and the quantile of the cumulative distribution function. The quantile is called Probability and is shown in the right-most cell.

Survival Profiler (Available only when the specified Distribution is Normal, Exponential, Weibull, Lognormal, or Cox Proportional Hazards.) Displays a profiler that shows the survival function as a function of the predictors and the response. The response is shown in the right-most cell.

Hazard Profiler (Available only when the specified Distribution is Normal, Exponential, Weibull, Lognormal, or Cox Proportional Hazards.) Displays a profiler that shows the hazard rate as a function of the predictors and the response. The response is shown in the right-most cell.

Custom Test Displays a Custom Test report that enables you to test a custom hypothesis. If the model has a Solution Path, the custom test results update as you update the solution. For more information about custom tests, see "Custom Test". The Custom Test red triangle menu contains an option to remove the Custom Test report.

Diagnostic Plots Provides various plots to help assess how well the current model fits. If a Validation column is specified or if KFold, Holdback, or Leave-One-Out is selected as the Validation Method, the options below enable you to view the training, validation, and, if applicable, test sets, or they construct separate plots for these sets. If KFold or Leave-One-Out is selected, then the plots correspond to the validation set that optimizes prediction error, and its corresponding training set. See "KFold".

Note: All Diagnostic plots update continuously as you drag the current model indicator in either Solution Path plot.

Diagnostic Bundle (Not available when the specified Distribution is Binomial, Multinomial, Ordinal Logistic, or Cox Proportional Hazards.) Displays a set of four graphs including a plot of residuals by predicted values, residuals by row number, a histogram of the residuals, and a histogram of the probability of observing a response larger than the observed response.

The graphs are constructed using all observations. If you used a Validation Column or if you selected KFold, Holdback, or Leave-One-Out as the Validation Method, check boxes enable you to select the Training, Validation, and, if applicable, Test sets. Rows corresponding to these sets are selected in the data table and the corresponding points and areas are highlighted in the graphs. Use this option to determine whether the model fit is similar across the sets.

The Fitted Probability of Observing a Larger Response histogram helps you assess goodness of fit of the model. Different criteria apply based on the distribution:

- For distributions other than zero-inflated distributions and quantile regression, the "correct" model should display an approximately uniform distribution of values.
- For a zero-inflated distribution, the histogram should display a point mass at zero and an approximately uniform distribution elsewhere.
- For quantile regression, the histogram should display an approximately uniform distribution of values to the left of the specified quantile and an approximately uniform distribution of slightly higher values to the right of the specified quantile.

Plot Baseline Survival and Hazard (Available only when the specified Distribution is Cox Proportional Hazards.) Displays the Baseline Survival and Hazard plots, which plot the survival and hazard functions for the baseline proportional hazards function versus the response variable. Below the plots, there is a table that contains the plotted values.

Note: If the specified Distribution is Cox Proportional Hazards, the Plot Baseline Survival and Hazard option is the only available Diagnostic Plot.

ROC Curve (Available only when the specified Distribution is Binomial, Multinomial, or Ordinal Logistic.) Displays the Receiver Operating Characteristic (ROC) curve. If you used validation, an ROC curve is shown for each of the Training, Validation, and Test sets.

The ROC curve measures the ability of the fitted probabilities to classify response levels correctly. The further the curve from the diagonal, the better the fit. An introduction to ROC curves is found in *Basic Analysis*.

If the response has more than two levels, the ROC Curve plot displays an ROC curve for each response level. For a given response level, this curve is the ROC curve for correct classification into that level. See *Predictive and Specialized Modeling* for more information about ROC curves.

Lift Curve (Available only when the specified Distribution is Binomial, Multinomial, or Ordinal Logistic.) Displays the lift curve for the model. If you used validation, an ROC curve is shown for each of the Training, Validation, and Test sets.

A lift curve shows how effectively response levels are classified as their fitted probabilities decrease. The fitted probabilities are plotted along the horizontal axis in descending order. The vertical coordinate for a fitted probability is the proportion of correct classifications for that probability or higher, divided by the overall correct classification rate. Use the lift curve to see whether you can correctly classify a large proportion of observations if you select only those with a fitted probability that exceeds a threshold value.

If the response has more than two levels, the Lift Curve plot displays a lift curve for each response level. For a given response level, this curve is the lift curve for correct classification into that level. See *Predictive and Specialized Modeling* for more information about lift curves.

Plot Actual by Predicted (Not available when the specified Distribution is Binomial, Multinomial, Ordinal Logistic, or Cox Proportional Hazards.) Plots actual Y values on the vertical axis and predicted Y values on the horizontal axis. If you used validation, a plot is shown for each of the Training, Validation, and Test sets.

Plot Residual by Predicted (Not available when the specified Distribution is Binomial, Multinomial, Ordinal Logistic, or Cox Proportional Hazards.) Plots the residuals on the vertical axis and the predicted Y values on the horizontal axis. If you used validation, a plot is shown for each of the Training, Validation, and Test sets.

Plot Residual by Predictor (Not available when the specified Distribution is Binomial, Multinomial, Ordinal Logistic, or Cox Proportional Hazards. Not available for models that contain a predictor that has the Vector modeling type.) For each predictor in the model, plots the residuals on the vertical axis and the predictor values on the horizontal axis. There is a plot for each of the predictors in the model. If you used validation, a set of plots is shown for each of the Training, Validation, and Test sets.

Normal Quantile Plot (Available only when the specified Distribution is Normal and there is no censoring.) Plots normal quantiles on the vertical axis and standardized residuals on the horizontal axis. If you used validation, a plot is shown for each of the Training, Validation, and Test sets.

Save Columns Enables you to save columns based on the fitted model to the data table. See "Save Columns Options for Cox Proportional Hazards Models" for the options that are available if Cox Proportional Hazards is selected as the Distribution. For all other Distributions, the following columns can be saved to the data table:

Save Functional Prediction Formulas (Available only when the response columns contain the FDE FPC Num column property.) Saves a column for each FDE principal component response to the data table that contains a prediction formula for each functional principal component. This option also adds a model prediction formula that is a linear combination of the prediction formulas and the eigenfunction columns from the Functional Data Explorer platform. A script is added to the data table that enables you to use the model prediction formula to profile the original response, which is specified in the FDE Output column property of the FDE principal component response columns. For more information about functional principal components, see *Predictive and Specialized Modeling*.

Note: The Save Functional Prediction Formulas option saves formula columns for all FDE principal component responses in the report window. If multiple models are fit for a single response, the final model for each response is used to create the prediction formula for that response.

Save Prediction Formula Saves a column to the data table that contains the prediction formula, given in terms of the observed (unstandardized) data values. The prediction formula does not contain zeroed terms. See "Statistical Details for Distributions" for mean formulas.

When the response column is categorical, this option creates a probability column for each response level as well as a column that contains the most likely response. The Most Likely response column contains the level with the highest probability based on the model. If the Probability Threshold is a value other than 0.5, this option creates an additional column that contains the most likely response based on the probability threshold value.

Mean Confidence Interval Saves two columns to the data table that contain the lower and upper 95% confidence limits for the mean response.

Note: You can change the α level for the confidence interval in the Fit Model window by selecting Set Alpha Level from the Model Specification red triangle menu.

Std Error of Predicted Saves a column to the data table that contains the standard errors of the predicted mean response.

Save Residual Formula Saves a column to the data table that contains a formula for the residuals, given in the form Y minus the prediction formula. The residual formula does not contain zeroed terms. Not available if Binomial is selected as the Distribution.

Save Variance Formula Saves a column to the data table that contains a formula for the variance of the prediction. The variance of the prediction is calculated using the formula for the variance of the selected Distribution. The value of the parameter involved in the link function is estimated by applying the inverse of the link function to the estimated linear component. Other parameters are replaced by their estimates. See "Statistical Details for Distributions" for variance formulas. Not available if Binomial is selected as the Distribution.

Save Linear Predictor Saves a column to the data table that contains a formula for the product of the design matrix and the vector of parameter estimates. This is commonly referred to as $\mathbf{X}\beta$. The formula does not contain zeroed terms.

Save Validation Column (Available only if the specified Validation Method is KFold, Holdback, or Leave-One-Out.) Saves a column that describes the assignment of rows to folds. For KFold, the column lists the fold to which the row was assigned. For Holdback, each row is identified as belonging to the Training or Validation set. For Leave-One-Out, the row's value indicates its order in being left out.

Note: If you selected a Validation column in the launch window, the Save Validation Column option does not appear.

Save Distribution Formula (Not available when the specified distribution is Binomial or Quantile Regression.) Saves a column to the data table that contains a formula for the cumulative distribution function.

Save Survival Formula (Available only when the specified Distribution is continuous.) Saves a column to the data table that contains a formula for the probability of survival at the observed time. The survival function is equal to 1 minus the cumulative distribution function.

Save Simulation Formula Saves a column to the data table that contains a formula that generates simulated values using the estimated parameters for the model that you fit. This column can be used in the Simulate utility as a Column to Switch In. See *Basic Analysis*.

Cook's D Influence (Available only if the specified Distribution is Normal and the specified Estimation Method is Standard Least Squares.) Saves a column to the data table that contains the values for Cook's *D* Influence statistic.

Hats (Available only if the specified Distribution is Normal and the specified Estimation Method is Standard Least Squares.) Saves a column to the data table that contains the diagonal elements of $\mathbf{X(X'X)^{-1}X'}$. These values are sometimes called *hat values*.

Publish Prediction Formula Creates prediction formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Save Columns Options for Cox Proportional Hazards Models

Save Survival Formula Saves a column to the data table that contains a formula for the probability of survival at the observed time.

Save Cox Snell Residual Formula Saves a column to the data table that contains a formula for the Cox-Snell residuals. The Cox-Snell residuals are strictly positive. See Meeker and Escobar (1998, sec. 17.6.1) for a discussion of Cox-Snell residuals.

Save Martingale Residual Formula Saves a column to the data table that contains a formula for the martingale residuals. The martingale residual is defined as the difference between the observed number of events for an individual and a conditionally expected number of events. The martingale residuals have a mean of zero and range between negative infinity and 1. See Fleming and Harrington (1991).

Save Linear Predictor Saves a column to the data table that contains a formula for the product of the design matrix and the vector of parameter estimates. This is commonly referred to as $\mathbf{X}\beta$. The formula does not contain zeroed terms.

Remove Fit Removes the report for the fit.

JMP PRO Statistical Details for the Generalized Regression Personality

- "Statistical Details for Estimation Methods"
- "Statistical Details for Advanced Controls"
- "Statistical Details for Distributions"

JMP PRO Statistical Details for Estimation Methods

Penalized regression methods introduce bias to the regression coefficients by penalizing them.

JMP PRO Ridge Regression

An l_2 penalty is applied to the regression coefficients during ridge regression. Ridge regression coefficient estimates are defined as follows:

$$\hat{\beta}^{ridge} = \text{argmin}_{\beta}\left\{\sum_{i=1}^{N} -LogLikelihood(\beta;y_i) + \frac{\lambda}{2}\sum_{j=1}^{p}\beta_j^2\right\},$$

where $\sum_{j=1}^{p}\beta_j^2$ is the l_2 penalty, λ is the tuning parameter, N is the number of rows, and p is the number of variables.

JMP PRO Dantzig Selector

An l_∞ penalty is applied to the regression coefficients during Dantzig Selector. Coefficient estimates for the Dantzig Selector satisfy the following criterion:

$$\min_{\beta}\left\|X^T(y - X\beta)\right\|_\infty \quad \text{subject to } \|\beta\|_1 \le t$$

where $\|v\|_\infty$ denotes the l_∞ norm, which is the maximum absolute value of the components of the vector v.

JMP PRO Lasso Regression

An l_1 penalty is applied to the regression coefficients during Lasso. Coefficient estimates for the Lasso are defined as follows:

$$\hat{\beta}^{lasso} = \text{argmin}_{\beta}\left\{\frac{1}{2}\sum_{i=1}^{N} -LogLikelihood(\beta;y_i) + \lambda\sum_{j=1}^{p}|\beta_j|\right\},$$

where $\sum_{j=1}^{p}|\beta_j|$ is the l_1 penalty, λ is the tuning parameter, N is the number of rows, and p is the number of variables

JMP PRO Elastic Net

The Elastic Net combines both l_1 and l_2 penalties. Coefficient estimates for the Elastic Net are defined as follows:

$$\hat{\beta}^{enet} = \text{argmin}_{\beta}\left\{\sum_{i=1}^{N} -LogLikelihood(\beta;y_i) + \lambda\sum_{j=1}^{p}\left(\alpha|\beta_j| + \frac{(1-\alpha)}{2}\beta_j^2\right)\right\},$$

This is the notation used in the equation:

$\sum_{j=1}^{p}|\beta_j|$ is the l_1 penalty

$\sum_{j=1}^{p}\beta_j^2$ is the l_2 penalty

λ is the tuning parameter

α is a parameter that determines the mix of the l_1 and l_2 penalties

N is the number of rows

p is the number of variables

Tip: There are two sample scripts that illustrate the shrinkage effect of varying α and λ in the Elastic Net for a single predictor. Select **Help > Sample Data**, click **Open the Sample Scripts Directory**, and select demoElasticNetAlphaLambda.jsl or demoElasticNetAlphaLambda2.jsl. Each script contains a description of how to use it and what it illustrates.

JMP PRO Adaptive Methods

The adaptive Lasso method uses weighted penalties to provide consistent estimates of coefficients. The weighted form of the l_1 penalty is defined as follows:

$$\sum_{j=1}^{p} \frac{|\beta_j|}{\tilde{\beta}_j}$$

where $\tilde{\beta}_j$ is the MLE when the MLE exists. If the MLE does not exist and the response distribution is normal, estimation is done using least squares and $\tilde{\beta}_j$ is the solution obtained using a generalized inverse. If the response distribution is not normal, $\tilde{\beta}_j$ is the ridge solution.

For the adaptive Lasso, this weighted form of the l_1 penalty is used in determining the $\hat{\beta}^{lasso}$ coefficients.

The adaptive Elastic Net uses this weighted form of the l_1 penalty and also imposes a weighted form of the l_2 penalty. The weighted form of the l_2 penalty for the adaptive Elastic Net is defined as follows:

$$\sum_{j=1}^{p} \left(\frac{\beta_j}{\tilde{\beta}_j}\right)^2$$

where $\tilde{\beta}_j$ is the MLE when the MLE exists. If the MLE does not exist and the response distribution is normal, estimation is done using least squares and $\tilde{\beta}_j$ is the solution obtained using a generalized inverse. If the response distribution is not normal, $\tilde{\beta}_j$ is the ridge solution.

JMP PRO Statistical Details for Advanced Controls

JMP PRO Grid

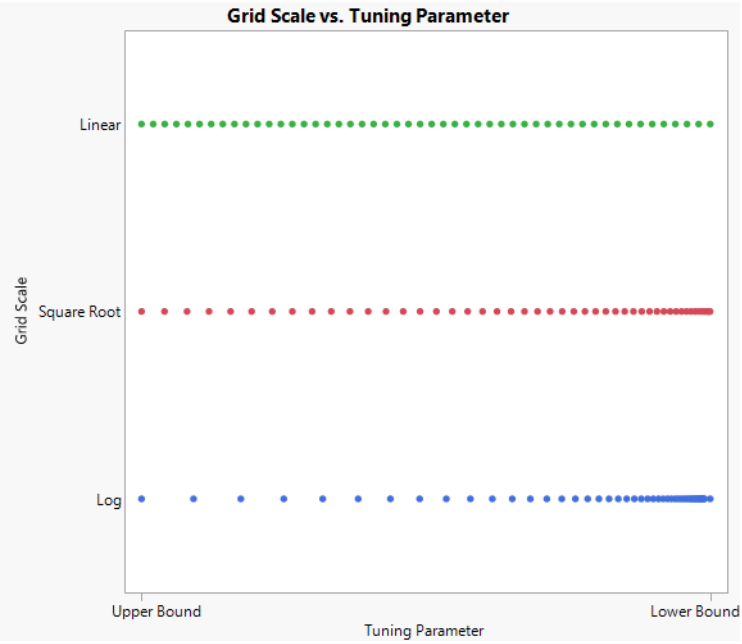
The tuning parameters for ridge regression and the Lasso that best minimize the penalized likelihood are found by searching a grid of tuning parameter values. This grid of values lies between a lower and an upper bound for the tuning parameter. You can specify the number of grid points under Advanced Controls.

The lower bound is zero except in special cases where it is set to 0.0001. See "Tuning Parameter". When the lower bound for the tuning parameter is zero, the solution is unpenalized and the coefficients are the MLEs. The upper bound is the smallest value for which all of the non-intercept terms are zero.

The grid of values between the lower and upper bounds is iteratively searched to determine the best value of the tuning parameter. The grid of possible tuning parameters can be set up in three different scales: linear, log, and square root. The default grid scale is square root.

In some cases, there is a large gap between the unpenalized estimates and the previous step. This large gap can distort the solution path. The log scale focuses its search on small tuning parameter values with few large values, whereas the linear scale evenly disperses the search from the minimum to the maximum value. The square root scale is a compromise between the other two scales. Figure 6.7 shows the different grid scales.

Figure 6.7 Options for Tuning Parameter Grid Scale



JMP PRO Statistical Details for Distributions

The distributions fit by the Generalized Regression personality are given below in terms of the parameters used in model fitting. Although it is not specifically stated as part of their descriptions, the Generalized Regression personality enables you to specify non-integer values for the discrete distributions.

JMP PRO Continuous Distributions

Normal Distribution

$$f(y|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{1}{2\sigma^2}(y-\mu)^2\right], -\infty < y < \infty$$

$$E(Y) = \mu$$

$$Var(Y) = \sigma^2$$

Cauchy Distribution

$$f(y|\mu,\sigma) = \left\{\pi\sigma\left[1+\left(\frac{y-\mu}{\sigma}\right)^2\right]\right\}^{-1}, -\infty < y < \infty$$

$$E(Y) = \text{undefined}$$

$$Var(Y) = \text{undefined}$$

t(5) Distribution

$$f(y|\mu,\sigma) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)}\frac{1}{\sqrt{\nu\pi\sigma^2}}\left[1+\frac{(x-\mu)^2}{\nu\sigma^2}\right]^{-\left(\frac{\nu+1}{2}\right)}, -\infty < y < \infty, \nu = 5$$

$$E(Y) = \mu$$

$$Var(Y) = \sigma^2\frac{\nu}{\nu-2}$$

Exponential Distribution

$$f(y|\theta) = \frac{1}{\mu}\exp\left[-\frac{y}{\mu}\right], y > 0$$

$$E(Y) = \mu$$

$$Var(Y) = \mu^2$$

Gamma Distribution

$$f(y|\mu,\sigma) = \frac{y^{(\mu/\sigma)-1}\exp[-y/\sigma]}{\Gamma[\mu/\sigma]\sigma^{\mu/\sigma}}, y > 0$$

$$E(Y) = \mu$$

$$Var(Y) = \mu\sigma$$

Weibull Distribution

$$f(y|\mu, \sigma) = \frac{1}{y\sigma}\exp\left[\frac{\log(y)-\mu}{\sigma}\right]\exp\left\{-\exp\left[\frac{\log(y)-\mu}{\sigma}\right]\right\}, \; y > 0$$

$$E(Y) = \exp(\mu)\Gamma[1+\sigma]$$

$$Var(Y) = \exp(2\mu)\{\Gamma[1+2\sigma] - (\Gamma[1+\sigma])^2\}$$

LogNormal Distribution

$$f(y|\mu, \sigma) = \frac{1}{y\sigma\sqrt{2\pi}}\exp\left\{-\frac{1}{2\sigma^2}[\log(y)-\mu]^2\right\}, \; y > 0$$

$$E(Y) = \exp\left(\mu + \frac{\sigma^2}{2}\right)$$

$$Var(Y) = [\exp(\sigma^2) - 1]\exp(2\mu + \sigma^2)$$

Beta Distribution

$$f(y|\mu, \sigma) = \frac{\Gamma[1/\sigma]}{\Gamma[\mu/\sigma]\Gamma[(1-\mu)/\sigma]}y^{\mu/\sigma - 1}(1-y)^{\frac{(1-\mu)}{\sigma} - 1}, \; y \in (0, 1)$$

$$E(Y) = \mu$$

$$Var(Y) = \mu(1-\mu)\frac{\sigma}{\sigma+1}$$

JMP PRO Discrete Distributions

Binomial Distribution

$$f(y|n, p) = \binom{n}{y}p^y(1-p)^{n-y}, \; y = 0, 1, 2, \ldots, n$$

$$E(Y) = np$$

$$Var(Y) = np(1-p)$$

Beta Binomial Distribution

$$f(y|n,p,\delta) = \binom{n}{y}\frac{\Gamma\left[\frac{1}{\delta}-1\right]\Gamma\left[y+p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[n-y+(1-p)\left(\frac{1}{\delta}-1\right)\right]}{\Gamma\left[p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[(1-p)\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[n-1+\frac{1}{\delta}\right]}, \; y = 0, 1, 2, \ldots, n$$

$$E(Y) = np$$

$$Var(Y) = np(1-p)[1+(n-1)\delta]$$

Poisson Distribution

$$f(y|\lambda) = \frac{\lambda^y}{y!}\exp(-\lambda), \; y = 0, 1, 2, \ldots$$

$$E(Y) = \lambda$$

$$Var(Y) = \lambda$$

Negative Binomial Distribution

$$f(y|\mu,\sigma) = \frac{\Gamma[y+(1/\sigma)]}{\Gamma[y+1]\Gamma[1/\sigma]}\left[\frac{(\mu\sigma)^y}{(1+\mu\sigma)^{y+(1/\sigma)}}\right], \; y = 0, 1, 2, \ldots$$

$$E(Y) = \mu$$

$$Var(Y) = \mu + \sigma\mu^2$$

JMP PRO Zero-Inflated Distributions

Zero-Inflated Binomial Distribution

$$f(y|n,p,\pi) = \begin{cases} \pi + (1-\pi)(1-p)^n, \text{ for } y = 0 \\ (1-\pi)\binom{n}{y}p^y(1-p)^{n-y}, \text{ for } y = 1, 2, \ldots, n \end{cases}$$

$$E(Y) = np(1-\pi)$$

$$Var(Y) = (1-\pi)np(1-p) + n^2p^2(1-\pi)^2$$

Zero-Inflated Beta Binomial Distribution

$$f(y|n,p,\delta,\pi) = \begin{cases} \pi + (1-\pi)\dfrac{\Gamma\left[\frac{1}{\delta}-1\right]\Gamma\left[p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[n+(1-p)\left(\frac{1}{\delta}-1\right)\right]}{\Gamma\left[p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[(1-p)\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[n-1+\frac{1}{\delta}\right]}, \text{ for } y=0 \\ (1-\pi)\binom{n}{y}\dfrac{\Gamma\left[\frac{1}{\delta}-1\right]\Gamma\left[y+p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[n-y+(1-p)\left(\frac{1}{\delta}-1\right)\right]}{\Gamma\left[p\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[(1-p)\left(\frac{1}{\delta}-1\right)\right]\Gamma\left[n-1+\frac{1}{\delta}\right]}, y=1,2,\ldots,n \end{cases}$$

$$E(Y) = np(1-\pi)$$

$$Var(Y) = (1-\pi)np\{1+(1-p)[1+(n-1)\delta]\} - [np(1-\pi)]^2$$

Zero-Inflated Poisson Distribution

$$f(y|\lambda,\pi) = \begin{cases} \pi + (1-\pi)\exp[-\lambda], \text{ for } y=0 \\ (1-\pi)\dfrac{\lambda^y}{y!}\exp[-\lambda], \text{ for } y=1,2,\ldots \end{cases}$$

$$E(Y) = (1-\pi)\lambda$$

$$Var(Y) = \lambda(1-\pi)(1+\lambda\pi)$$

Zero-Inflated Negative Binomial Distribution

$$f(y|\mu,\sigma,\pi) = \begin{cases} \pi + (1-\pi)(1+\mu\sigma)^{-(1/\sigma)}, \text{ for } y=0 \\ (1-\pi)\dfrac{\Gamma[y+(1/\sigma)]}{\Gamma[y+1]\Gamma[1/\sigma]}\left[\dfrac{(\mu\sigma)^y}{(1+\mu\sigma)^{y+(1/\sigma)}}\right], \text{ for } y=1,2,\ldots \end{cases}$$

$$E(Y) = (1-\pi)\mu$$

$$Var(Y) = \mu(1-\pi)[1+\mu(\sigma+\pi)]$$

Zero-Inflated Gamma Distribution

$$f(y|\mu,\sigma,\pi) = \begin{cases} \pi, \text{ for } y=0 \\ (1-\pi)\dfrac{\exp(-y/\sigma)}{\Gamma[\mu/\sigma]\sigma^{\mu/\sigma}y^{1-\mu/\sigma}}, \text{ for } y>0 \end{cases}$$

$$E(Y) = \mu(1-\mu)$$

$$Var(Y) = \mu(1-\pi)(\sigma+\mu) - (1-\pi)^2\mu^2$$

Chapter 7

Generalized Regression Examples

Build Models Using Regularization Techniques

This chapter provides examples with instructional material for several models fit using the Generalized Regression platform.

Contents

Poisson Generalized Regression Example 345
Binomial Generalized Regression Example 347
Zero-Inflated Poisson Regression Example 349
Generalized Regression Example for Wide Data 352

Poisson Generalized Regression Example

The Liver Cancer.jmp sample data table contains liver cancer Node Count values for 136 patients. It also includes measurements on six potentially related variables: BMI, Age, Time, Markers, Hepatitis, and Jaundice. These variables are described in the Notes column property in each column of the data table. To view the Notes column property, right-click a column name, select Column Info, and select Notes under Column Properties.

This example develops a prediction model for Node Count using the six predictors. Node Count is modeled using a Poisson distribution.

1. Select **Help > Sample Data Library** and open Liver Cancer.jmp.
2. Select **Analyze > Fit Model**.
3. Select Node Count from the Select Columns list and click **Y**.
4. Select BMI through Jaundice and click **Macros > Factorial to Degree**.

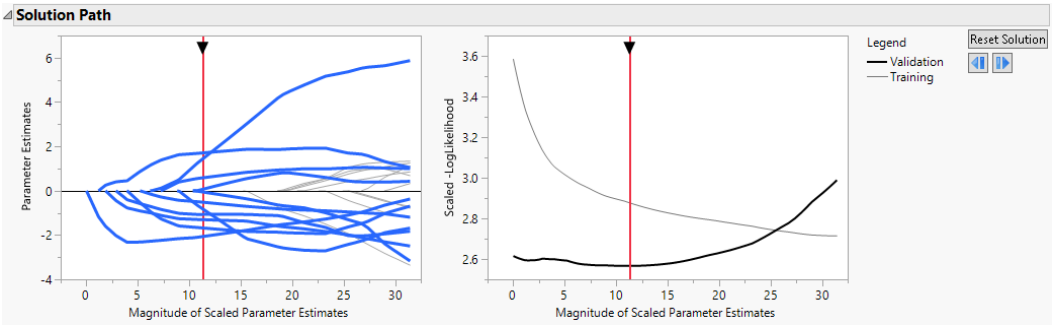
   This adds all terms up to degree 2 (the default in the **Degree** box) to the model.
5. Select Validation from the Select Columns list and click **Validation**.
6. From the Personality list, select **Generalized Regression**.
7. From the Distribution list, select **Poisson**.
8. Click **Run**.

   The Generalized Regression report that appears contains a Model Comparison report, a Model Launch control panel, and a Poisson Maximum Likelihood with Validation Column report. Note that the default estimation method is the Lasso.
9. Select the **Adaptive** box.
10. Click **Go**.
11. Click the red triangle next to Poisson Adaptive Lasso with Validation Column and select **Select Nonzero Terms**.

    The Solution Path is shown in Figure 7.1. The paths for terms that have nonzero coefficients are highlighted. Think of the solution paths as moving from right to left across the plot, as the solutions shrink farther from the MLE. A number of terms have paths that shrink them to zero fairly early.

    The vertical axis in the Solution Path Plot represents the values of the parameter estimates for the standardized predictors. The vertical red line indicates their values at the optimal shrinkage, as determined by cross validation. At this point, 11 terms have nonzero coefficients. Notice that the vertical red line indicates the minimum Scaled –LogLikelihood value in the Validation set.

Figure 7.1 Solution Path for Adaptive Lasso Fit with Nonzero Terms Highlighted



The Parameter Estimates for Original Predictors report (Figure 7.2) shows the parameter estimates for the uncentered and unscaled data. The 11 terms with nonzero parameter estimates are highlighted. These include interaction effects. In the data table, all six predictor columns are selected because every predictor column appears in a term that has a nonzero coefficient.

In the Effect Tests report, the 10 effects with zero coefficient estimates are designated as Removed. The Effect Tests report indicates that only one effect is significant at the 0.05 level: the Age*Markers interaction.

Figure 7.2 Parameter Estimates Report with Nonzero Terms Highlighted

Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	1.9658009	1.5268783	1.6575637	0.1979	-1.026826	4.9584273
BMI	-0.014557	0.0655911	0.049258	0.8244	-0.143114	0.1139989
Age	-0.000344	0.0052439	0.0043118	0.9476	-0.010622	0.0099335
Time	0	0	0	1.0000	0	0
Markers[0-1]	0.3853237	0.3347747	1.3247874	0.2497	-0.270823	1.0414702
Hepatitis[0-1]	0	0	0	1.0000	0	0
Jaundice[0-1]	-0.220763	0.1645355	1.8002539	0.1797	-0.543247	0.1017205
(BMI-23.1737)*(Age-56.3994)	0.0007336	0.0014188	0.2673515	0.6051	-0.002047	0.0035144
(BMI-23.1737)*(Time-9.00945)	0	0	0	1.0000	0	0
(BMI-23.1737)*Markers[0-1]	-0.066574	0.0511528	1.6938297	0.1931	-0.166832	0.0336837
(BMI-23.1737)*Hepatitis[0-1]	0.0269878	0.0827504	0.1063642	0.7443	-0.1352	0.1891755
(BMI-23.1737)*Jaundice[0-1]	0	0	0	1.0000	0	0
(Age-56.3994)*(Time-9.00945)	0	0	0	1.0000	0	0
(Age-56.3994)*Markers[0-1]	-0.023498	0.0083023	8.0106349	0.0047*	-0.03977	-0.007226
(Age-56.3994)*Hepatitis[0-1]	0	0	0	1.0000	0	0
(Age-56.3994)*Jaundice[0-1]	0	0	0	1.0000	0	0
(Time-9.00945)*Markers[0-1]	-0.007529	0.0104633	0.5178167	0.4718	-0.028037	0.0129784
(Time-9.00945)*Hepatitis[0-1]	0	0	0	1.0000	0	0
(Time-9.00945)*Jaundice[0-1]	0.0009619	0.0093476	0.0105893	0.9180	-0.017359	0.019283
Markers[0-1]*Hepatitis[0-1]	-0.422619	0.3974016	1.1309371	0.2876	-1.201512	0.356274
Markers[0-1]*Jaundice[0-1]	0	0	0	1.0000	0	0
Hepatitis[0-1]*Jaundice[0-1]	0	0	0	1.0000	0	0

12. Click the row for (Age - 56.3994)*Markers[0-1] in the Parameter Estimates for Original Predictors report.

This action highlights that effect's path in the Solution Path Plot and selects the columns Age and Markers in the data table.

13. Click the red triangle next to Poisson Adaptive Lasso with Validation Column and select **Save Columns > Save Prediction Formula** and **Save Columns > Save Variance Formula**.

    Two columns are added to the data table: Node Count Prediction Formula and Node Count Variance.

14. In the data table, right-click either column heading and select **Formula** to view the formula. Alternatively, click the plus sign to the right of the column name in the Columns panel.

    The prediction formula in the **Save Prediction Formula** column applies the exponential function to the estimated linear part of the model. The prediction variance formula in Node Count Variance is given by the identical formula, because the variance of a Poisson distribution is equal to its mean.

Binomial Generalized Regression Example

This example shows how to develop a prediction model for the binomial response, Severity, in the Liver Cancer.jmp sample data table.

1. Select **Help > Sample Data Library** and open Liver Cancer.jmp.
2. Select **Analyze > Fit Model**.
3. Select Severity from the Select Columns list and click **Y**.
4. Select BMI through Jaundice and click **Macros > Factorial to Degree**.

   All terms up to degree 2 (the default in the **Degree** box) are added to the model.

5. From the Personality list, select **Generalized Regression**.

   The Distribution list automatically shows the Binomial distribution. This is the only distribution available when Y is binary and has a Nominal modeling type.

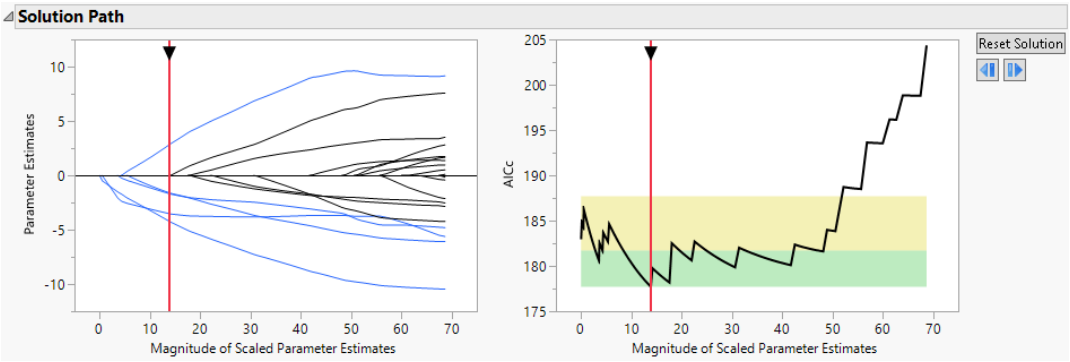
6. Click **Run**.

   The Generalized Regression report that appears contains a Model Comparison report, a Model Launch control panel, and a Logistic Regression report. Note that the default estimation method is the Lasso.

7. Select **Elastic Net** as the Estimation Method.
8. Select the **Adaptive** box.
9. Click **Go**.

   A Binomial Adaptive Elastic Net with AICc Validation report appears. The Solution Path is shown in Figure 7.3.

Figure 7.3 Solution Path Plot



The paths for terms that have nonzero coefficients are shown in blue. The optimal parameter values are substantially shrunken away from the MLE. The Validation Plot to the right indicates that several models can be considered as good as the best model. To view those models, slide the vertical red bar around in the region where the black line is in the green area.

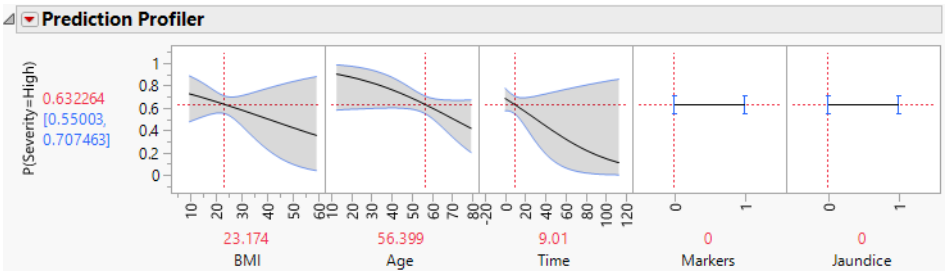
10. Click the red triangle next to Binomial Adaptive Elastic Net with AICc Validation and select the **Select Zeroed Terms** option.

    The 16 terms that have coefficient estimates of zero are highlighted in the Parameter Estimates for Original Predictors report. The Effect Tests report designates these terms as Removed.

    The Effect Tests report also shows that there are no significant terms at the 0.05 level. However, the Time*Markers interaction has a small *p*-value of 0.0626 and the Time effect has a small *p*-value of 0.1458.

11. Click the red triangle next to Binomial Adaptive Elastic Net with AICc Validation and select **Profilers > Profiler**.

Figure 7.4 Profiler for Probability That Severity = High, Time Low

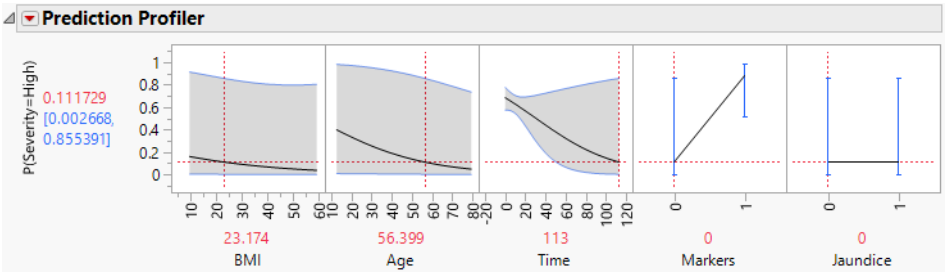


Examine the Prediction Profiler to see how Time and the Time*Markers interaction affect Severity.

Note: The predictor Hepatitis is not shown in the profiler because it does not appear in any active (nonzero) terms. Because Markers and Jaundice appear in active interaction terms, they appear in the profiler even though, as main effects, they are not active.

12. Move the red dashed line for Time from left to right to see its interaction with Markers (Figure 7.4 and Figure 7.5). For patients who enter the study with small values of Time since diagnosis, Markers have little impact on Severity. But for patients who enter the study having been diagnosed for a longer time, Markers are important. For those patients, normal markers suggest a lower probability of high Severity.

Figure 7.5 Profiler for Probability That Severity = High, Time High



Zero-Inflated Poisson Regression Example

The Fishing.jmp sample data table contains fictional data for a study of various factors that affect the number of fish caught by groups visiting a park. The data table contains 250 responses from families or groups of traveling companions. This example models the number of Fish Caught as a function of Live Bait, Fishing Poles, Camper, People, and Children. These variables are described in the Notes column property in each column of the data table. To view the Notes column property, right-click a column name, select Column Info, and select Notes under Column Properties.

The data table contains a hidden column called Fished. During data collection, it was never determined whether anyone in the group had actually fished. However, the Fished column is included in the table to emphasize the point that catching zero fish can happen in one of two ways: Either no one in the group fished, or everyone who fished in the group was unlucky.

Therefore, zero responses can come from two sources. To address this issue, you can fit a zero-inflated distribution. Because a Poisson distribution is appropriate for the count data resulting from people who fished, you fit a zero-inflated Poisson distribution.

1. Select **Help > Sample Data Library** and open Fishing.jmp.
2. Select **Analyze > Fit Model**.
3. Select Fish Caught from the Select Columns list and click **Y**.

4. Select Live Bait through Children and click **Macros > Factorial to Degree**.

   Terms up to degree 2 (the default in the **Degree** box) are added to the model.
5. Select Validation from the Select Columns list and click **Validation**.
6. From the Personality list, select **Generalized Regression**.
7. From the Distribution list, select **ZI Poisson**.
8. Click **Run**.

   The Generalized Regression report that appears contains a Model Comparison report, a Model Launch control panel, and a ZI Poisson Maximum Likelihood with Validation Column report. Note that the default estimation method is the Lasso.
9. From the Estimation Method List, select **Elastic Net**.
10. Click **Go**.

    A ZI Poisson Elastic Net with Validation Column report appears. The Solution Path, the Parameter Estimates for Original Predictors report, and the Effect Tests report indicate that one term is zeroed. The Zero Inflation parameter, whose estimate is shown on the last line of both Parameter Estimates reports, is highly significant. This indicates that some of the variation in the response, Fish Caught, might be due to the fact that some groups did not fish.

Figure 7.6 Parameter Estimates for Original Predictors Report

Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	1.3599777	0.3237335	17.647702	<.0001*	0.7254716	1.9944837
Live Bait[0-1]	-0.379555	0.1919937	3.9081951	0.0481*	-0.755856	-0.003255
Fishing Poles	0.3034685	0.0465239	42.547553	<.0001*	0.2122833	0.3946537
Camper[0-1]	-0.363094	0.1899392	3.6543491	0.0559	-0.735368	0.0091795
People	0.0776017	0.121377	0.4087617	0.5226	-0.160293	0.3154962
Children	-0.192656	0.1028943	3.5057658	0.0612	-0.394325	0.009013
Live Bait[0-1]*(Fishing Poles-0.856)	-0.026766	0.0668764	0.160184	0.6890	-0.157841	0.1043095
Live Bait[0-1]*Camper[0-1]	-0.336382	0.48759	0.4759452	0.4903	-1.292041	0.6192764
Live Bait[0-1]*(People-2.516)	-0.164164	0.1994702	0.6773258	0.4105	-0.555118	0.2267908
Live Bait[0-1]*(Children-0.936)	0.1025463	0.2362161	0.1884607	0.6642	-0.360429	0.5655213
(Fishing Poles-0.856)*Camper[0-1]	-0.446199	0.1120088	15.869111	<.0001*	-0.665732	-0.226666
(Fishing Poles-0.856)*(People-2.516)	-0.00611	0.0853344	0.0051267	0.9429	-0.173362	0.1611423
(Fishing Poles-0.856)*(Children-0.936)	-0.17587	0.080167	4.8127459	0.0282*	-0.332995	-0.018746
Camper[0-1]*(People-2.516)	-0.195261	0.3035369	0.4138148	0.5200	-0.790182	0.3996608
Camper[0-1]*(Children-0.936)	0.1899805	0.2998548	0.4014171	0.5264	-0.397724	0.7776852
(People-2.516)*(Children-0.936)	0	0	0	1.0000	0	0

ZI Poisson Distribution Parameters	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Zero Inflation	0.7815222	0.0354433	486.19905	<.0001*	0.7120545	0.8509898

The Effect Tests report indicates that four terms are significant at the 0.05 level: Live Bait, Fishing Poles, Fishing Poles*Camper, and Fishing Poles*Children.

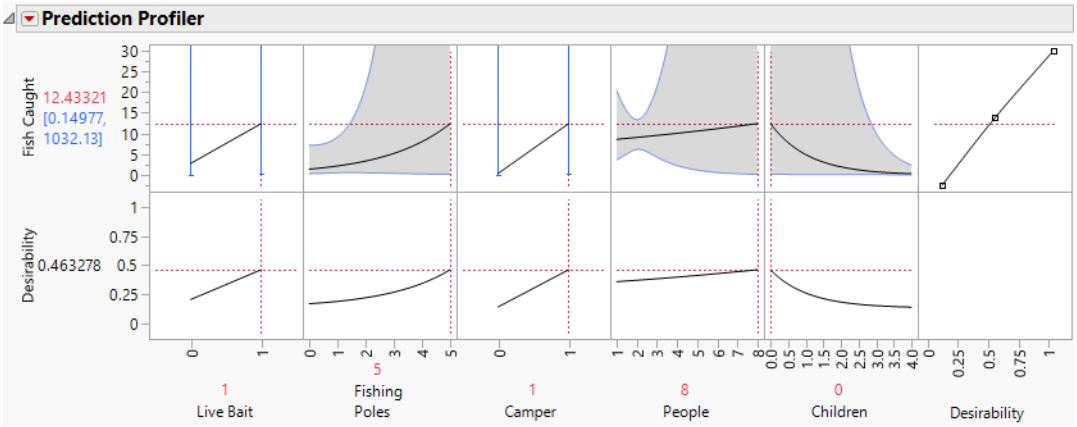
11. Click the red triangle next to ZI Poisson Elastic Net with Validation Column and select **Profilers > Profiler**.

12. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Desirability Functions**.

    A function is imposed on the response, which indicates that maximizing the number of Fish Caught is desirable. For more information about desirability functions, see *Profilers*.

13. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

Figure 7.7 Prediction Profiler with Fish Caught Maximized



You can vary the settings of the predictors to see the impact of the significant effects: Live Bait, Fishing Poles, Fishing Poles*Camper, and Fishing Poles*Children. For example, Live Bait is associated with more fish; a Camper tends to bring more fishing poles than someone who is not camping and therefore catches more fish.

14. Click the red triangle next to ZI Poisson Elastic Net with Validation Column and select **Save Columns > Save Prediction Formula** and **Save Columns > Save Variance Formula**.

    Two columns are added to the data table: Fish Caught Prediction Formula and Fish Caught Variance.

15. In the data table, right-click either column heading and select **Formula** to view the formula. Alternatively, click the plus sign to the right of the column name in the Columns panel. Note the appearance of the estimated zero-inflation parameter, 0.781522155, in both of these formulas.

Generalized Regression Example for Wide Data

When there are more predictors than there are observations, traditional regression methods are not practical. Regression methods that incorporate variable selection enable you to fit a regression model in these situations. In this example, you compare three models with varying degrees of variable selection. The three models are all located within the green zone on the Validation Plot, so there is strong evidence that any of these models are comparable to the best model.

The Prostate Cancer.jmp sample data table contains results of serum samples collected from 165 men, approximately half of whom had prostate cancer. There are 667 proteins measured in the serum samples.

1. Select **Help > Sample Data Library** and open Prostate Cancer.jmp.
2. Select **Analyze > Fit Model**.
3. Select Status from the Select Columns list and click **Y**.

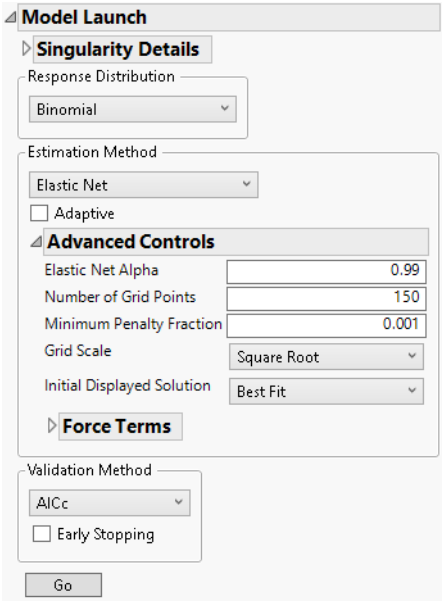
   Because this is a Nominal response column, the Personality changes to Nominal Logistic and the Target Level option appears. The default value for this option is CCD, because that is the value specified in the Target Level column property in the data table.
4. From the Personality list, select **Generalized Regression**.

   The Distribution list automatically shows the Binomial distribution. This is the only distribution available when Y is binary and has a Nominal modeling type.
5. Select the Proteins column group from the Select Columns list and click **Add**.

   This adds all 667 columns in the column group to the model.
6. Click **Run**.

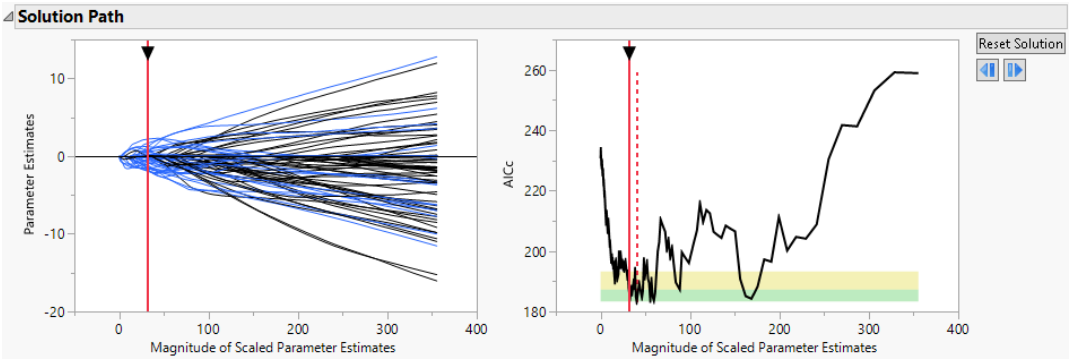
   The Generalized Regression report that appears contains a Model Launch control panel. There is no initial Logistic Regression model fit because the number of predictors is greater than the number of observations.
7. Select **Elastic Net** as the Estimation Method.
8. Click the gray disclosure icon next to Advanced Controls.

Figure 7.8 Advanced Controls



9. Select **Smallest in Green Zone** as the Initial Displayed Solution.
10. Click **Go**.

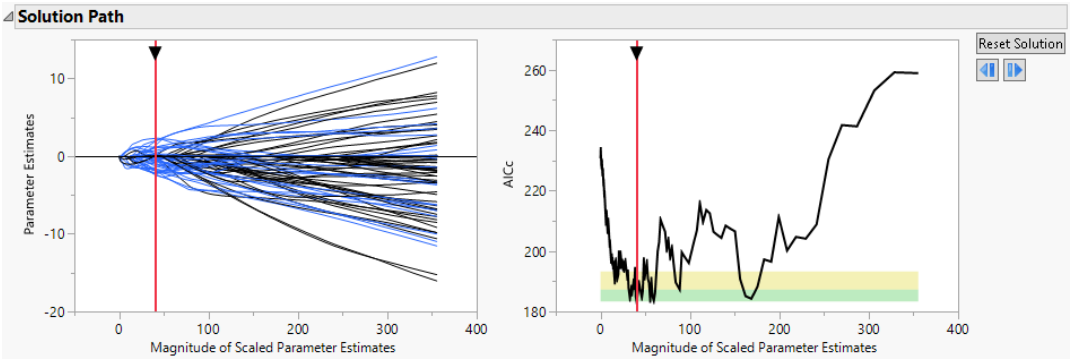
Figure 7.9 Smallest in Green Zone Model



The Solution Path shows the smallest model that is considered comparable to the minimum AICc model, where smallest model means the one with the fewest parameters.

11. Click the gray disclosure icon next to Binomial Elastic Net with AICc Validation.
12. Click the gray disclosure icon next to Model Launch.
13. Select **Best Fit** as the Initial Displayed Solution.
14. Click **Go**.

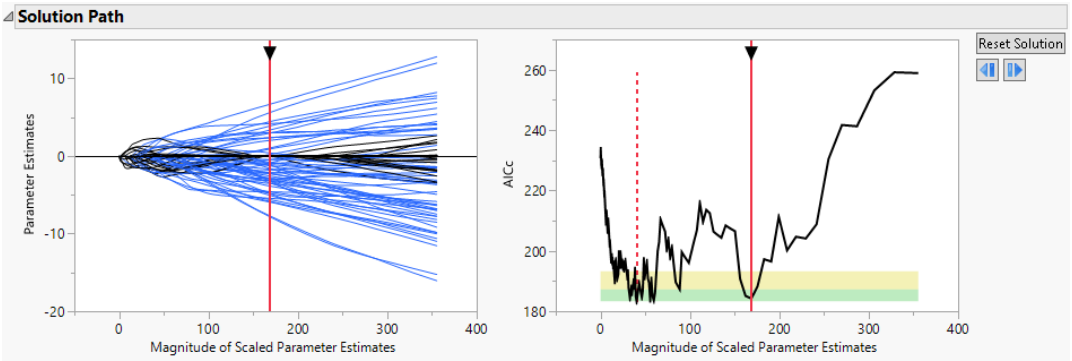
Figure 7.10 Best Fit Model



The Solution Path shows the best fit model, where best fit means the one with the minimum AICc value.

15. Click the gray disclosure icon next to Binomial Elastic Net with AICc Validation.
16. Click the gray disclosure icon next to Model Launch.
17. Select **Biggest in Green Zone** as the Initial Displayed Solution.
18. Click **Go**.

Figure 7.11 Biggest in Green Zone Model



The Solution Path shows the largest model that is considered comparable to the minimum AICc model, where largest model means the one with the most parameters.

19. Click the gray disclosure icon next to Binomial Elastic Net with AICc Validation.

Figure 7.12 Model Comparison Table

Model Comparison

Show	Response Distribution	Estimation Method	Validation Method	Nonzero Parameters	AICc	BIC	Generalized RSquare
☑	Binomial	Elastic Net	AICc	36	186.07102	277.07256	0.7466199
☑	Binomial	Elastic Net	AICc	40	183.29544	281.08164	0.8021925
☑	Binomial	Elastic Net	AICc	57	184.25399	299.49849	0.9824565

The Model Comparison report shows the three models. You can identify the size of each model using the Nonzero Parameters column. As the number of parameters in the models increases, the Generalized RSquare values increase. Because these models are all in the green zone, there is strong evidence that any of these models are comparable to the best model.

Chapter 8

JMP PRO Mixed Models

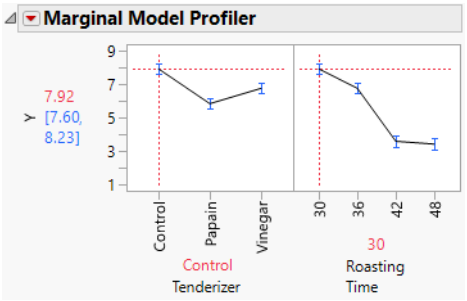
Jointly Model the Mean and Covariance

The Mixed Models personality of the Fit Model platform is available only in JMP Pro.

In JMP Pro, the Fit Model platform's Mixed Model personality fits a wide variety of linear models for continuous responses with complex covariance structures. These models include random coefficients, repeated measures, spatial data, and data with multiple correlated responses. Use the Mixed Model personality to specify linear mixed models and their covariance structures conveniently using an intuitive interface, and to fit these models using maximum likelihood methods.

The modeling results are supported by interactive visualization tools such as profilers, surface plots, and contour plots. You can use these tools to complement your understanding of the model.

Figure 8.1 Marginal Model Profiler for a Split Plot Experiment



Contents

Overview of the Mixed Model Personality . . . 359
Example Using Mixed Model . . . 360
Launch the Mixed Model Personality . . . 364
Fit Model Launch Window . . . 364
Data Format . . . 370
The Fit Mixed Report . . . 370
Fit Statistics . . . 373
Random Effects Covariance Parameter Estimates . . . 375
Fixed Effects Parameter Estimates . . . 376
Repeated Effects Covariance Parameter Estimates . . . 377
Random Coefficients . . . 377
Random Effects Predictions . . . 377
Fixed Effects Tests . . . 378
Sequential Tests . . . 378
Multiple Comparisons . . . 379
Compare Slopes . . . 380
Marginal Model Inference . . . 380
Actual by Predicted Plot . . . 380
Residual Plots . . . 381
Profilers . . . 381
Conditional Model Inference . . . 382
Actual by Conditional Predicted Plot . . . 382
Conditional Residual Plots . . . 382
Conditional Profilers . . . 383
Variogram . . . 384
Covariance and Correlation Matrices . . . 385
Save Columns . . . 385
Additional Examples of the Mixed Models Personality . . . 386
Repeated Measures Example . . . 387
Split Plot Example . . . 402
Spatial Example: Uniformity Trial . . . 408
Correlated Response Example . . . 417
Statistical Details for the Mixed Models Personality . . . 424
Convergence Score Test . . . 424
Random Coefficient Model . . . 425
Repeated Measures . . . 426
Repeated Covariance Structures . . . 427
Spatial and Temporal Variability . . . 432
The Kackar-Harville Correction . . . 435

JMP PRO Overview of the Mixed Model Personality

In JMP Pro, the Mixed Model personality lets you analyze models with complex covariance structures. The situations that can be analyzed include:

- Split plot experiments
- Random coefficients models
- Repeated measures designs
- Spatial data
- Correlated response data

Split plot experiments are experiments with two or more levels, or sizes, of experimental units resulting in multiple error terms. Such designs are often necessary when some factors are easy to vary and others are more difficult to vary. See the *Design of Experiments Guide*.

Random coefficients models are also known as hierarchical or multilevel models (Singer 1998; Sullivan et al. 1999). These models are used when batches or subjects are thought to differ randomly in intercept and slope. Drug stability trials in the pharmaceutical industry and individual growth studies in educational research often require random coefficient models.

Repeated measures designs, spatial data, and correlated response data share the property that observations are not independent, requiring that you model their correlation structure.

- Repeated measures designs, also known as within-subject designs, model changes in a response over time or space while allowing errors to be correlated.
- Spatial data are measurements made in two or more dimensions, typically latitude and longitude. Spatial measurements are often correlated as a function of their spatial proximity.
- Correlated response data result from making several measurements on the same experimental unit. For example, height, weight, and blood pressure readings taken on individuals in a medical study, or hardness, strength, and elasticity measured on a manufactured item, are likely to be correlated. Although these measurements can be studied individually, treating them as correlated responses can lead to useful insights.

Failure to account for correlation between observations can result in incorrect conclusions about treatment effects. However, estimating covariance structure parameters uses information in the data. The number of parameters being estimated impacts power and the Type I error rate. For this reason, you must choose covariance models judiciously. See "Repeated Measures Example".

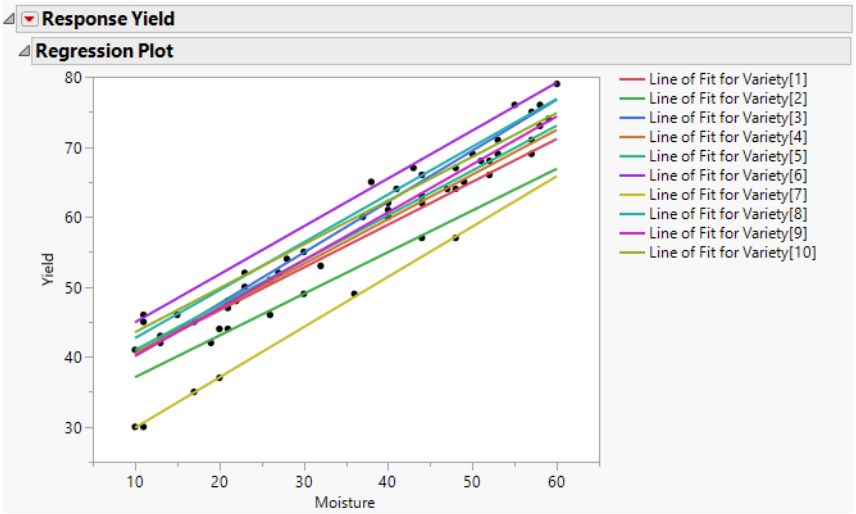
JMP PRO Example Using Mixed Model

In a study of wheat yield, 10 varieties of wheat are randomly selected from the population of varieties of hard red winter wheat adapted to dry climate conditions. These are randomly assigned to six one-acre plots of land. The preplanting moisture content of the plots could influence the germination rate and hence the eventual yield of the plots. Thus, the amount of preplanting moisture in the top 36 inches of soil is determined for each plot. You are interested in determining if the moisture content affects yield.

Because the varieties are randomly selected, the regression model for each variety is a random model selected from the population of variety models. The intercept and slope are random for each variety and might be correlated. The random coefficients are centered at the fixed effects. The fixed effects are the population intercept and the slope, which are the expected values of the population of the intercepts and slopes of the varieties. This example is taken from Littell et al. (2006, p. 320).

Fitting the model using REML in the Standard Least Squares personality lets you view the variation in intercepts and slopes (Figure 8.2). Note that the slopes do not have much variability, but the intercepts have quite a bit. The intercept and slope might be negatively correlated; varieties with lower intercepts seem to have higher slopes.

Figure 8.2 Standard Least Squares Regression



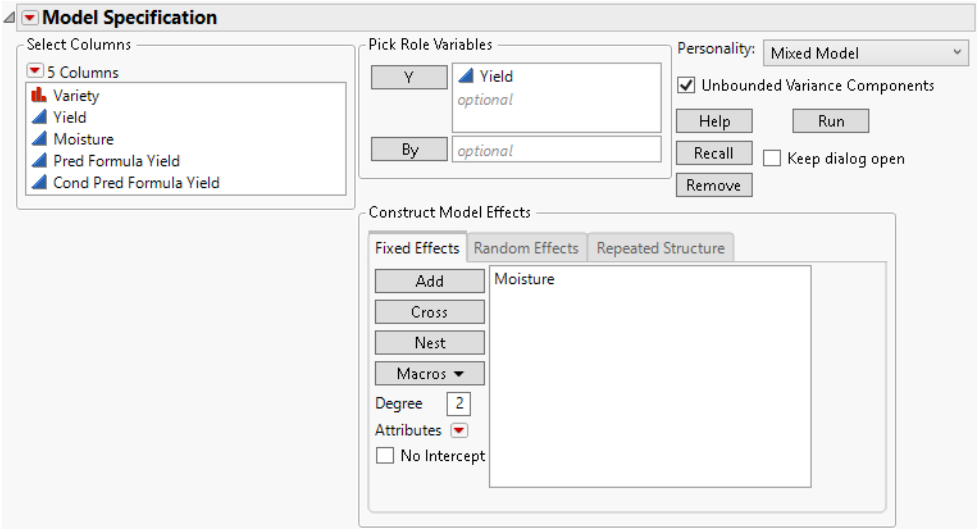
To model the correlation between the intercept and the slope, use the Mixed Models personality. You are interested in determining the population regression equation as well as variety-specific equations.

1. Select **Help > Sample Data Library** and open Wheat.jmp.

2. Select **Analyze > Fit Model**.
3. Select Yield and click **Y**.

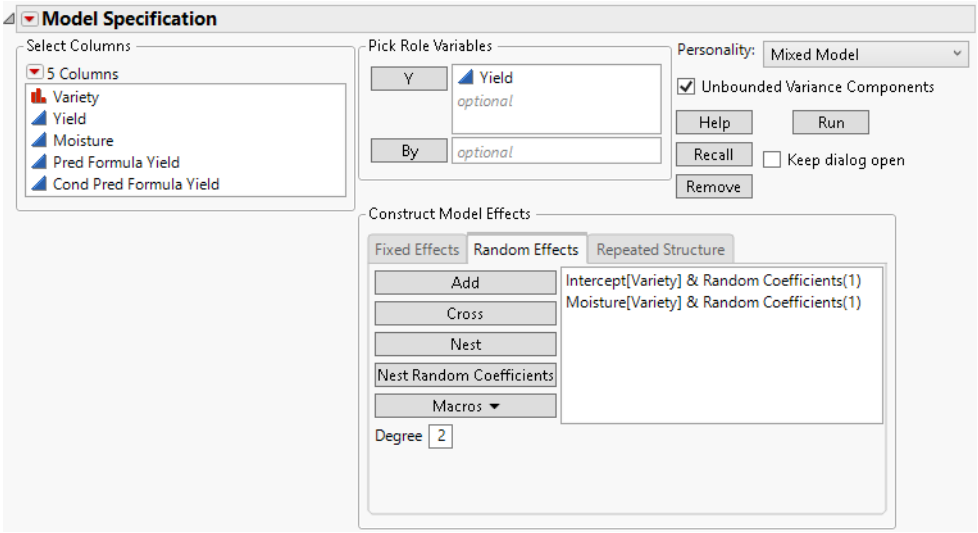
   When you add this column as Y, the fitting Personality becomes Standard Least Squares.
4. Select **Mixed Model** from the Personality list. Alternatively, you can select the Mixed Model personality first, and then click **Y** to add Yield.
5. Select Moisture and click **Add** on the Fixed Effects tab.

Figure 8.3 Completed Fit Model Launch Window Showing Fixed Effects



6. Select the **Random Effects** tab.
7. Select Moisture and click **Add**.
8. Select **Variety** from the Select Columns list, select Moisture from the Random Effects tab, and then click **Nest Random Coefficients**.

Figure 8.4 Completed Fit Model Launch Window Showing Random Effects Tab



Random effects are grouped by variety, and the intercept is included as a random component.

9. (Optional.) Click the Model Specification red triangle and check the setting of the **Center Polynomials** option.

   Even if the Center Polynomials option is selected, the Moisture effect will not be centered at its mean because it is involved in a random effect.

10. Click **Run**.

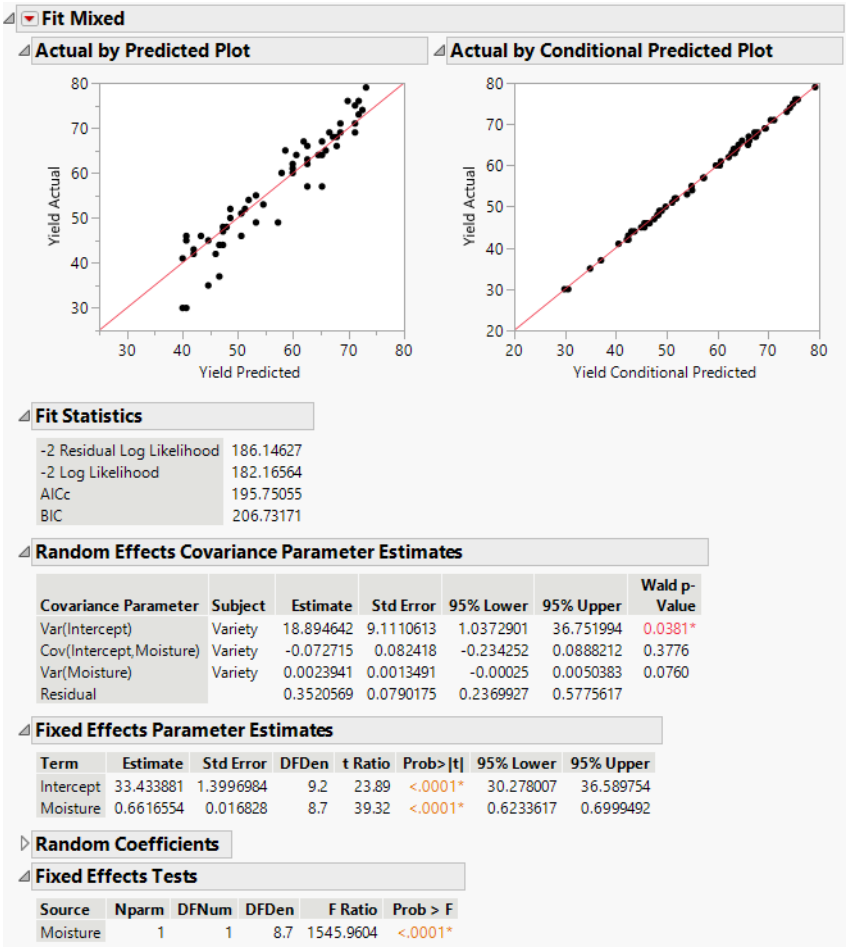
    The Fit Mixed report is shown in Figure 8.5. The Actual by Predicted plot shows no discrepancy in terms of model fit and underlying assumptions.

    Because there are no apparent problems with the model fit, you can now interpret the statistical tests and obtain the regression equation. The effect of moisture upon yield is significant, as shown in the Fixed Effects Tests report. The estimates given in the Fixed Effects Parameter Estimates indicate that the following equation is the estimated population regression equation:

    $$Yield = 33.43 + 0.66 * Moisture$$

    The Random Effects Covariance Parameter Estimates report gives estimates of the variance of the varieties' intercepts, Var(Intercept), and slopes, Var(Moisture), and their covariance, Cov(Moisture, Intercept). In this case, the intercept and slope are not significantly correlated, because the confidence interval for the estimate includes zero. The report also gives an estimate of the residual variance.

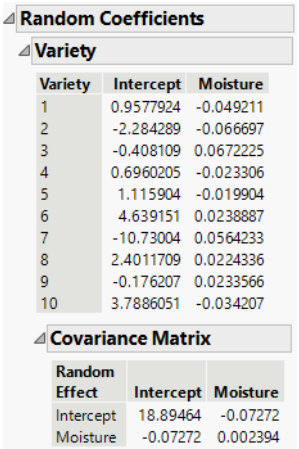
Figure 8.5 Fit Mixed Report



Although you have an estimate of the population regression equation, you are also interested in Variety 2's estimated yield.

11. Open the Random Coefficients report to see the estimates of the variety effects for Intercept and Moisture. These coefficients estimate how each variety differs from the population.

Figure 8.6 Random Coefficients Report



Random Coefficients

Variety

Variety	Intercept	Moisture
1	0.9577924	-0.049211
2	-2.284289	-0.066697
3	-0.408109	0.0672225
4	0.6960205	-0.023306
5	1.115904	-0.019904
6	4.639151	0.0238887
7	-10.73004	0.0564233
8	2.4011709	0.0224336
9	-0.176207	0.0233566
10	3.7886051	-0.034207

Covariance Matrix

Random Effect	Intercept	Moisture
Intercept	18.89464	-0.07272
Moisture	-0.07272	0.002394

From the Fixed Effects Parameter Estimates and Random Coefficients reports, you obtain the following prediction equation for Variety 2:

$$Yield = 33.433 + 0.662 * Moisture - 2.284 - 0.067 * Moisture$$

$$Yield = 31.149 + 0.595 * Moisture$$

Variety 2 starts with a lower yield than the population average and increases with Moisture at a slower rate than the population average.

JMP PRO Launch the Mixed Model Personality

Mixed Model is one of several personalities that you can select in the Fit Model launch window. This section describes how you select Mixed Model as your fitting methodology in the Fit Model launch window. Options that are specific to this selection are also covered.

JMP PRO Fit Model Launch Window

You can specify models with fixed effects, random effects, a repeated structure or a combination of those. The options differ based on the nature of the model that you specify.

To fit models using the mixed model personality, select **Analyze > Fit Model** and then select **Mixed Model** from the Personality list. Note that when you enter a continuous variable in the Y list *before* selecting a Personality, the Personality defaults to Standard Least Squares. For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

When fitting models using the Mixed Model personality, you can allow unbounded variance components. This means that variance components that have negative estimates are not reported as zero. This option is selected by default. It should remain selected if you are interested in fixed effects, because bounding the variance estimates at zero leads to bias in the tests for fixed effects. See "Negative Variances" for more information about the Unbounded Variance Components option.

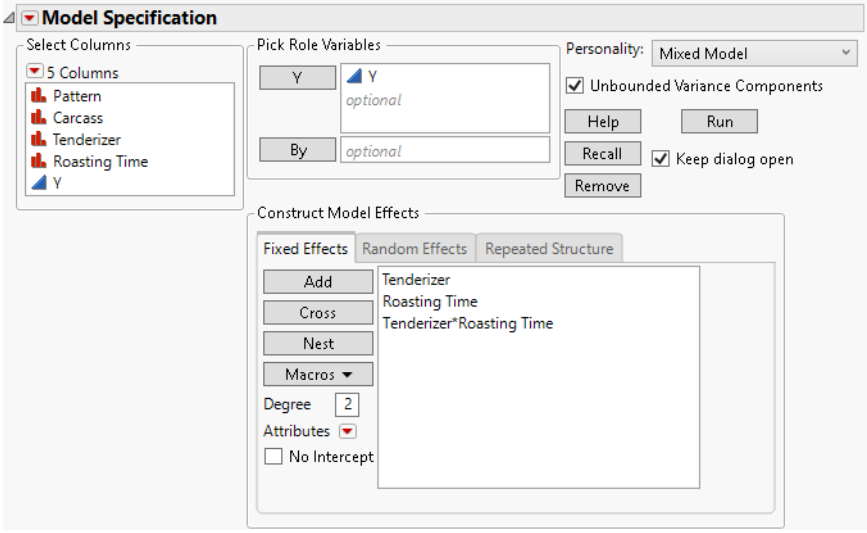
JMP PRO Fixed Effects Tab

Add all fixed effects on the Fixed Effects tab. Use the Add, Cross, Nest, Macros, and Attributes options as needed. For more information about these options, see "Model Specification".

Note: If a continuous column is involved in a random effect, that column is not centered, even if the Center Polynomials option in the Model Specifications red triangle menu is selected.

The fixed effects for analysis of the Split Plot.jmp sample data table appear in Figure 8.7. Note that it is possible to have no fixed effects in the model. For an example, see "Spatial Example: Uniformity Trial".

Figure 8.7 Fit Model Launch Window Showing Completed Fixed Effects



JMP PRO Random Effects Tab

Specify traditional variance component models and random coefficients models using the Random Effects tab.

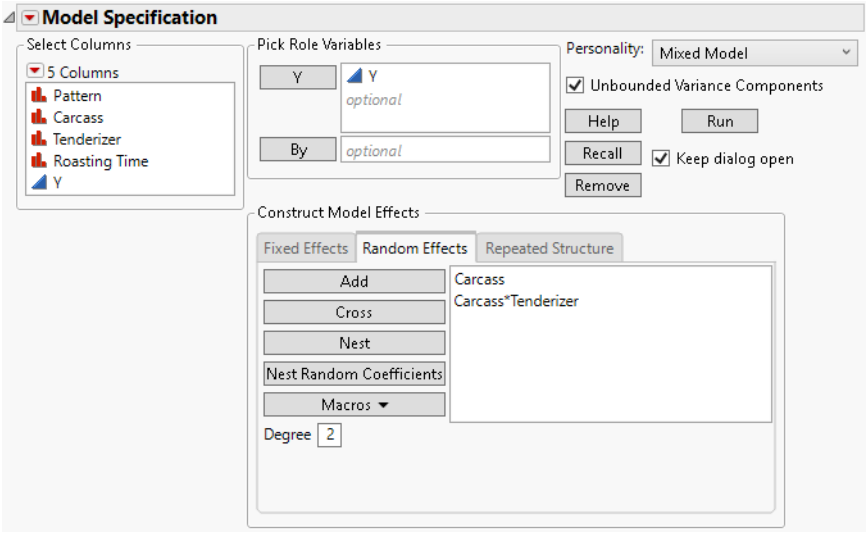
Note: If a continuous column is involved in a random effect, that column is not centered, even if the Center Polynomials option in the Model Specifications red triangle menu is selected.

Variance Components

For a traditional variance component model, specify terms such as random blocks, whole plot error terms, and subplot error terms using the Add, Cross, or Nest options. For more information about these options, see "Model Specification".

Figure 8.8 shows the random effects specification for the Split Plot.jmp sample data where Carcass is a random block. "Split Plot Example" describes the example in detail.

Figure 8.8 Fit Model Launch Window Showing Completed Random Effects Tab



Random Coefficients

To construct random coefficients models, use the Nest Random Coefficients button to create groups of random coefficients.

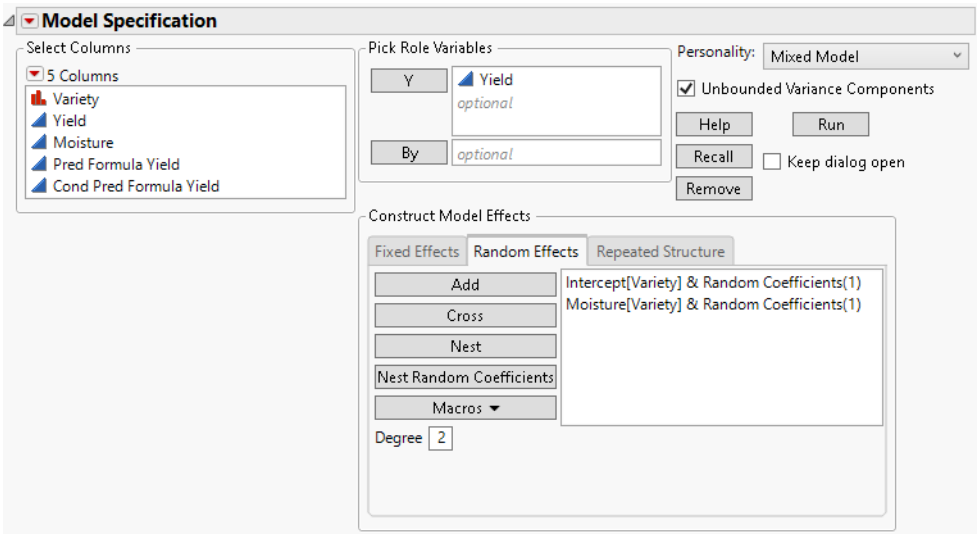
1. Select the continuous columns from the Select Columns list that are predictors.
2. Select the **Random Effects** tab and then **Add**.
3. Select these effects in the Random Effects tab. Also select the column that contains the random effect whose levels define the individual regression models. This column is essentially the subject in a random statement in SAS PROC MIXED.
4. Click the **Nest Random Coefficients** button.

This last step creates random intercept and random slope effects that are correlated within the levels of the random effect. The subject is nested within the other effects due to the variability among subjects. If you believed that the intercept might be fixed for all groups, you would select Intercept[<*group*>]&Random Coefficients(1) and then click **Remove**.

You can define multiple groups of random coefficients in this fashion, as in hierarchical linear models. This might be necessary when you have both a random batch effect and a random batch by treatment effect on the slope and intercept coefficients. This might also be necessary in a hierarchical linear model: when you have a random student effect and random school effect on achievement scores and students are nested within school.

Random coefficients are modeled using an unstructured covariance structure. Figure 8.9 shows the random coefficients specification for the Wheat.jmp sample data. See also "Example Using Mixed Model".

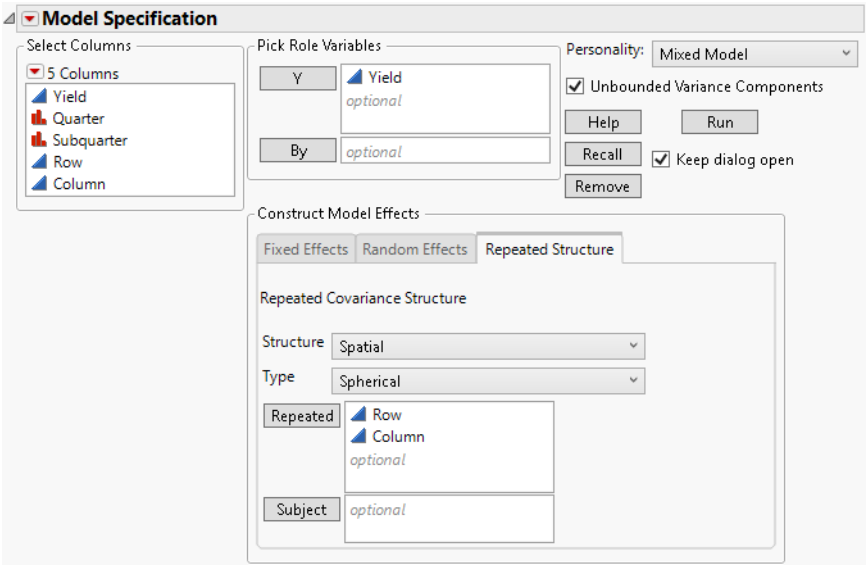
Figure 8.9 Completed Fit Model Launch Window Showing Random Coefficients



JMP PRO Repeated Structure Tab

Use the Repeated Structure tab to select a covariance structure for repeated effects in the model.

Figure 8.10 Completed Fit Model Launch Window Showing Repeated Structure Tab



Structure

The repeated structure is set to Residual by default. The Residual structure specifies that there is no covariance between observations, namely, the errors are independent. Besides the Residual and Unequal Variances structures, all other covariance structures model covariance between observations. For more information about the structures, see "Repeated Measures" and "Spatial and Temporal Variability" in the Statistical Details section.

Table 8.1 lists the covariance structures available, the requirements for using each structure, and the number of covariance parameters for the given structure. The number of observation times is denoted by *J*.

Table 8.1 Repeated Covariance Structure Requirements

Structure	Repeated Column Type	Required Number of Repeated Columns	Subject	Number of Parameters
Residual	not applicable	0	not applicable	0

Table 8.1 Repeated Covariance Structure Requirements *(Continued)*

Structure	Repeated Column Type	Required Number of Repeated Columns	Subject	Number of Parameters
Unequal Variances	categorical	1	optional	*J*
Unstructured	categorical	1	required	*J*(*J*+1)/2
AR(1)	continuous	1	optional	2
Compound Symmetry	categorical	1	required	2
Antedependent Equal Variance	categorical		required	*J*
Toeplitz	categorical	1	required	*J*
Compound Symmetry Unequal Variances	categorical	1	required	*J*+1
Antedependent	categorical		required	2*J*-1
Toeplitz Unequal Variances	categorical	1	required	2*J*-1
Spatial	continuous	2+	optional	
Spatial Anisotropic	continuous	2+	optional	
Spatial with Nugget	continuous	2+	optional	
Spatial Anisotropic with Nugget	continuous	2+	optional	

If you enter a Repeated or Subject column with the Residual structure, those columns are ignored. This alert appears: “Repeated columns and subject columns are ignored when the Residual covariance structure is selected.”

Type

When you select one of the spatial covariance structures, a Type list appears from which you select a type of spatial structure. Four Types are available: Power, Exponential, Gaussian, and Spherical. Figure 8.10 shows the Spatial Spherical selection for the Uniformity Trial.jmp sample data.

Repeated

Enter columns that define the repeated measures structure. The modeling types of Repeated columns depend on the covariance structure. See Table 8.1 for more information about the requirements for each repeated measures covariance structure.

Subject

Enter one or more columns that define the Subject. Subject columns must be categorical.

JMP PRO Data Format

The Mixed Models personality in the Fit Model platform requires that all response measurements be contained in one response column. Repeated measures data are sometimes recorded in multiple columns, where each row is a subject and the repeated measurements are recorded in separate response columns. Data that are in this format must be stacked before running the Mixed Models personality. The Cholesterol.jmp and Cholesterol Stacked.jmp sample data tables illustrate the wide format and the stacked format, respectively. Notice that each row in the wide table corresponds to one level of Patient in the stacked table.

JMP PRO The Fit Mixed Report

The Fit Mixed red triangle menu contains the following options:

Model Reports Produces reports that relate to the mixed model fit. These reports give estimates and tests for model parameters, as well as fit statistics. See "Model Reports".

Linear Combination of Variance Components Shows a report that enables you to compute confidence intervals for linear combinations of variance components. Initially, the report contains an editable text box and a table of variance components in the model. Use the text box to label the linear combination. Enter values in the cells in the right column of the table to specify the linear functions for your confidence intervals. After you specify a linear

combination of parameters and click Done, a table appears that contains confidence intervals for the specified linear combination.

The table contains an estimate and standard error, as well as two types of confidence intervals (Satterthwaite and Wald) and a Wald *p*-value. The Wald *p*-value corresponds to a hypothesis test that the estimate differs from zero.

Tip: The Satterthwaite confidence interval is restricted to positive values, so it is not recommended for cases where the specified coefficients are negative. If the estimate is negative, the Satterthwaite confidence interval cannot be constructed and is reported as missing.

Multiple Comparisons Opens the Multiple Comparisons dialog window where you can select one or more effects and initial comparisons. This report is available for categorical fixed effects. See "Multiple Comparisons".

Compare Slopes (Available only when there is one nominal term, one continuous term, and their interaction effect for the fixed effects.) Produces a report that enables you to compare the slopes of each level of the interaction effect in an analysis of covariance (ANCOVA) model. See "Compare Slopes".

Inverse Prediction For one or more values of the response, predicts values of explanatory variables. See "Inverse Prediction".

Marginal Model Inference Produces plots based on marginal predicted values and marginal residuals. These plots display the variation due to random effects. See "Marginal Model Inference".

Conditional Model Inference Produces plots based on conditional predicted values and conditional residuals. These plots display the variation that remains, once random effects are accounted for. See "Conditional Model Inference".

Save Columns Contains options to save various model results as columns in the data table. See "Save Columns".

Model Dialog Opens the completed Fit Model launch window for the current analysis. See "Fit Model Launch Window".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Model Reports

The reports available under Model Reports are determined by the type of analysis that you conduct. Several of these reports are shown by default.

Fit Statistics Shows report for model fit statistics. See "Fit Statistics".

Random Effects Covariance Parameter Estimates Shows report of random effects covariance parameter estimates. This report appears when you specify random effects in the launch window. See "Random Effects Covariance Parameter Estimates".

Fixed Effects Parameter Estimates Shows report of fixed effects parameter estimates. See "Fixed Effects Parameter Estimates".

Repeated Effects Covariance Parameter Estimates Shows report of repeated effects covariance parameter estimates. This report appears when you specify repeated effects in the launch window. See "Repeated Effects Covariance Parameter Estimates".

Indicator Parameterization Estimates (Available only when there are nominal columns among the fixed effects.) Displays the Indicator Function Parameterization report. This report gives parameter estimates for the fixed effects based on a model where nominal fixed effect columns are coded using indicator (SAS GLM) parameterization and are treated as continuous. Ordinal columns remain coded using the usual JMP coding scheme. The SAS GLM and JMP coding schemes are described in "The Factor Models".

Caution: Standard errors, t-ratios, and other results given in the Indicator Function Parameterization report differ from those in the Parameter Estimates report. This is because the estimates are estimating different parameters.

Random Coefficients Shows report of random coefficients. This report appears when you specify random effects in the launch window. See "Random Coefficients".

Random Effects Predictions Shows report of random effect predictions. This report appears when you specify random effects in the launch window. See "Random Effects Predictions".

Fixed Effects Test (Available only for models that contain at least one fixed effect.) Shows tests of fixed effects. This report appears when you specify fixed effects in the launch window. See "Fixed Effects Tests".

Sequential Tests (Available only for models that contain at least one fixed effect.) Shows the Sequential (Type 1) Tests report that contains the sums of squares as effects are added to the model sequentially. Conducts *F* tests based on the sequential sums of squares. See "Sequential Tests".

JMP PRO Fit Statistics

The Fit Statistics report gives statistics used for model comparison. For all fit statistics, smaller is better. A likelihood ratio test between two models can be performed if one model is contained within the other. If not, a cautious comparison of likelihoods can be informative. For an example, see "Fit a Spatial Structure Model".

"Description of the Fit Statistics Report" uses the following notation:

- Specify the mixed model:

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{Z}\gamma + \varepsilon$$

  Here $\mathbf{y}$ is the *nx1* vector of observations, β is a vector of fixed-effect parameters, γ is a vector of random-effect parameters, and ε is a vector of errors.

- The vectors γ and ε are assumed to have a multivariate normal distribution where

$$E\begin{bmatrix}\gamma \\ \varepsilon\end{bmatrix} = \begin{bmatrix}\mathbf{0} \\ \mathbf{0}\end{bmatrix}$$

  and

$$Var\begin{bmatrix}\gamma \\ \varepsilon\end{bmatrix} = \begin{bmatrix}\mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}\end{bmatrix}$$

- With these assumptions, the variance of $\mathbf{y}$ is calculated as follows:

$$\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$$

Description of the Fit Statistics Report

-2 Residual Log Likelihood The final evaluation of twice the negative residual log-likelihood, the objective function.

$$-2loglikelihood_{\mathbf{R}}(\mathbf{G},\mathbf{R}) = \log|\mathbf{V}| + \log\left|\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right| + \mathbf{r}'\mathbf{V}^{-1}\mathbf{r} + (n-p)\log(2\pi)$$

where

$$\mathbf{r} = \mathbf{y} - \mathbf{X}(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}(\mathbf{X}'\mathbf{V}^{-1}\mathbf{y})$$

and p is the rank of X. Use the residual likelihood only for model comparisons where the fixed effects portion of the model is identical. See "Likelihood, AICc, and BIC".

-2 Log Likelihood The evaluation of twice the negative log-likelihood function. See "Likelihood, AICc, and BIC".

Use the log-likelihood for model comparisons in which the fixed, random, and repeated effects differ in any of the models.

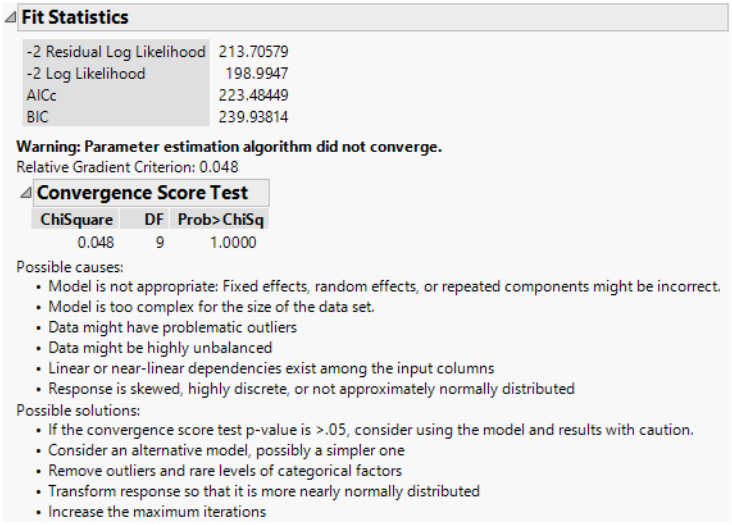
AICc Corrected Akaike's Information Criterion. See "Likelihood, AICc, and BIC".

BIC Bayesian Information Criterion. See "Likelihood, AICc, and BIC".

JMP PRO Convergence Score Test

If there are problems with model convergence, a warning message is displayed below the fit statistics. Figure 8.11 shows the warning that suggests the cause and possible solutions to the convergence issue. It also includes a test of the relative gradient at the final iteration. If this test is non-significant, the model might be correct but not fully reaching the convergence criteria. In this case, consider using the model and results with caution. See "Convergence Score Test".

Figure 8.11 Convergence Score Test



JMP PRO Random Effects Covariance Parameter Estimates

The Random Effects Covariance Parameter Estimates report provides details for the covariance parameters of the random effects that you specified in the model.

Covariance Parameter Lists all the covariance parameters of the random effects that you specified in the model.

> **Note:** This column is labeled Variance Component when the model contains only variance components.

Subject Lists the subject from which the block diagonal covariance matrix was formed.

Estimate Gives the estimated variance or covariance component for the effect.

> **Note:** When the model is equivalent to a REML model, a row for a Total covariance parameter is added to the table. The estimate for the Total covariance component is the sum of the positive variance components only.

Std Error Gives the standard error for the covariance component estimate.

95% Lower Gives the lower 95% confidence limit for the covariance component. See "Confidence Intervals for Variance Components".

95% Upper Gives the upper 95% confidence limit for the covariance component. See "Confidence Intervals for Variance Components".

Wald p-Value Gives the *p*-value for the test that the covariance parameter is equal to zero. This column appears only when you have selected Unbounded Variance Components in the Fit Model launch window.

JMP PRO Confidence Intervals for Variance Components

The method used to calculate the confidence limits depends on whether you have selected Unbounded Variance Components in the Fit Model launch window. Note that the Unbounded Variance Components is selected by default.

- If Unbounded Variance Components is selected, Wald-based confidence intervals are computed. These intervals are valid asymptotically, but note that they can be unreliable with small samples. The intervals are wider, which might lead you to mistakenly believe that an estimate is not significantly different from zero.
- If Unbounded Variance Components is not selected, meaning that the parameters have a lower boundary constraint of zero, a Satterthwaite approximation is used (Satterthwaite 1946). The confidence intervals are also bounded at zero.

JMP PRO Fixed Effects Parameter Estimates

The Fixed Effects Parameter Estimates report provides details for the fixed effect parameters specified in the model. For each parameter, the report provides the following details:

- the estimate
- the standard error (Std Error)
- a *t* test for the hypothesis that the estimate equals zero
- a 95% confidence interval on the estimate

The Fixed Effects Parameter Estimates report contains the following columns:

Term Gives the model term corresponding to the estimated parameter. The first term is always the intercept, unless you selected the No Intercept option in the Fit Model launch window. Continuous columns that are part of higher order terms are centered by default. Nominal or ordinal effects appear with values of levels in brackets. See "The Factor Models" for information about the coding of nominal and ordinal terms.

Note: If a continuous column is involved in a random effect, that column is not centered, even if the Center Polynomials option in the Model Specifications red triangle menu was selected.

Estimate Gives the parameter estimate for each term. This is the estimate of the term's coefficient in the model.

Std Error Gives an estimate of the standard error for the parameter estimate.

DFDen Gives the denominator degrees of freedom, that is, the degrees of freedom for error, for the effect test. DFDen is calculated using the Kenward-Roger first order approximation. See "The Kackar-Harville Correction".

t Ratio Tests whether the true value of the parameter is zero. The *t* Ratio is the ratio of the estimate to its standard error. Given the usual assumptions about the model, the *t* Ratio has a Student's *t* distribution under the null hypothesis.

Prob>|t| Lists the *p*-value for a two-sided test of the *t* Ratio.

95% Lower Shows the lower 95% confidence limit for the parameter.

95% Upper Shows the upper 95% confidence limit for the parameter.

JMP PRO Repeated Effects Covariance Parameter Estimates

The Repeated Effects Covariance Parameter Estimates report provides details for the covariance parameters of the repeated effects that you specified in the model. It includes the Estimate, Standard Error, and 95% confidence bounds for each parameter. For isotropic spatial models, the covariance parameter estimates have interpretations in terms of range, nugget, and sill. See "Variogram".

Note: Variances are covariances of effects with themselves.

The Repeated Effects Covariance Parameter Estimates report contains the following columns:

Covariance Parameter Lists all the covariance parameters for the repeated effects in the model.

Estimate Gives the estimated variance or covariance component for the effect.

Std Error Gives the standard error for the variance or covariance component estimate.

95% Lower Gives the lower 95% confidence limit for the covariance component. See "Confidence Intervals for Variance Components".

95% Upper Gives the upper 95% confidence limit for the covariance component. See "Confidence Intervals for Variance Components".

JMP PRO Random Coefficients

For each random effect, the Mixed Models personality provides a report showing estimated coefficients and a report showing the matrix of covariance estimates. Each row of the coefficients report corresponds to one level of the random effect. The row shows all coefficient estimates associated with that level of the random effect. The random coefficient estimates are used in conjunction with fixed effect estimates to create predictions for any specific level of the random effect.

JMP PRO Random Effects Predictions

For each random effect in the model, this report gives an estimate known as the *best linear unbiased predictor* (BLUP), its standard error, degrees of freedom, and a Satterthwaite-based confidence interval.

Estimation of the standard errors requires calculation of the BLUP covariance matrix, which can be time-intensive. If the calculation time is noticeable, a progress bar appears.

JMP PRO Fixed Effects Tests

This report shows a significance test for each fixed effect in the model. The test for a given effect tests the null hypothesis that all parameters associated with that effect are zero. An effect might have only one parameter as for a single continuous explanatory variable. In this case, the test is equivalent to the *t* test for that term in the Fixed Effects Parameter Estimates report. A nominal or ordinal effect can have several associated parameters, based on its number of levels. The effect test for such an effect tests whether all of the associated parameters are zero.

The Fixed Effects Tests report contains the following columns:

Source Lists the fixed effects in the model.

Nparm Shows the number of parameters associated with the effect. A continuous effect has one parameter. The number of parameters for a nominal or ordinal effect is one less than its number of levels. The number of parameters for a crossed effect is the product of the number of parameters for each individual effect.

DFNum Shows the numerator degrees of freedom for the effect test.

DFDen Shows the denominator degrees of freedom for the effect test (the degrees of freedom for error). DFDen is calculated using the Kenward-Roger first order approximation. See "The Kackar-Harville Correction".

F Ratio Gives the computed *F* ratio for testing that the effect is zero.

Prob > F Gives the *p*-value for the effect test.

JMP PRO Sequential Tests

This report shows sequential (type I) tests of the fixed effects. The report contains the sums of squares as effects are added to the model sequentially. The order of entry is defined by the order of effects as they appear in the Fit Model launch window's Construct Model Effects list.

The sums of squares that form the basis for sequential tests are also called *Type I Sums of Squares*. They are computed by fitting models in steps following the specified entry order of effects. Consider a specific effect. Compute the model sum of squares for a model containing all effects entered *prior* to that effect. Then compute the model sum of squares for a model containing those effects *and* the specified effect. The sequential sum of squares for the specified effect is the increase in the model sum of squares.

Sequential tests are considered appropriate in the following situations:

- balanced analysis of variance models specified in proper sequence (that is, two-way interactions follow main effects in the effects list, and so on)

- purely nested models specified in the proper sequence
- polynomial regression models specified in the proper sequence.

The Sequential (Type I) Tests report contains the following columns:

Source Lists the fixed effects in the model.

Nparm Shows the number of parameters associated with the effect. A continuous effect has one parameter. The number of parameters for a nominal or ordinal effect is one less than its number of levels. The number of parameters for a crossed effect is the product of the number of parameters for each individual effect.

DFNum Shows the numerator degrees of freedom for the effect test.

DFDen Shows the denominator degrees of freedom for the effect test (the degrees of freedom for error). DFDen is calculated using the Kenward-Roger first order approximation. See "The Kackar-Harville Correction".

F Ratio Gives the computed *F* ratio for testing that the effect is zero.

Prob > F Gives the *p*-value for the effect test.

JMP PRO Multiple Comparisons

The Multiple Comparisons option provides various methods for comparing least squares means of main effects and interaction effects. For more information about the multiple comparisons options, see "Multiple Comparisons". For mixed model examples, see "Compare All Treatments in June" and "Split Plot Example".

Only the fixed effect portion of the model is used in the multiple comparisons. The Multiple Comparisons report shows estimates of the least squares means, standard error, a *t* test of no effect, and a 95% confidence interval. This report is followed by the multiple comparisons test that you select. The All Pairwise Comparisons report provides equivalence tests.

JMP PRO Compare Slopes

The Compare Slopes option appears when there is one nominal term, one continuous term, and their interaction effect for the fixed effects. This option produces a report that enables you to compare the slopes in an analysis of covariance (ANCOVA) model. The report compares the slopes of each level of the interaction effect to the overall slope. The comparison uses analysis of means (ANOM) with the overall average. For more information about the analysis of means (ANOM) report, see "Comparisons with Overall Average".

The overall average slope is a weighted average of the slopes, where the weights are inversely proportional to the variances of the slope estimates. These variances are the squared values of the Std Error column in the Differences from Overall Average Slope table.

JMP PRO Marginal Model Inference

The marginal model plots are based on marginal predicted values and marginal residuals.

Actual by Predicted Plot Plots actual values versus values predicted by the model, but without accounting for the random effects. The Actual by Predicted Plot appears by default. See "Actual by Predicted Plot".

Residual Plots Provides residual plots that assess model fit, without accounting for the random effects. See "Residual Plots".

Profiler, Contour Profiler, Mixture Profiler, Surface Profiler Provides profilers to examine the relationship between the response and the model terms, without accounting for random effects. See "Profilers".

JMP PRO Actual by Predicted Plot

The Actual by Predicted plot appears by default. It provides a visual assessment of model fit that reflects variation due to random effects. It plots the observed values of Y against the marginal predicted values of Y. These are the predicted values obtained if you select Save Columns > Prediction Formula.

Denote the linear mixed model by $E[\mathbf{Y}|\gamma] = \mathbf{X}\beta + \mathbf{Z}\gamma$. Here β is the vector of fixed effect coefficients and γ is the vector of random effect coefficients. The *marginal predictions* are the predictions from the fixed effects part of the predictive model, given by $\mathbf{X}\hat{\beta}$.

JMP PRO Residual Plots

Marginal residuals reflect the prediction error based only on the fit of fixed effects. Marginal residuals are the differences between actual values and the predicted values obtained if you select Save Columns > Prediction Formula.

Denote the linear mixed model by $E[\mathbf{Y}|\gamma] = \mathbf{X}\beta + \mathbf{Z}\gamma$. Here β is the vector of fixed effect coefficients and γ is the vector of random effect coefficients. The *marginal residuals* are the residuals from the fixed effects part of the predictive model:

$$\mathbf{r} = \mathbf{Y} - \mathbf{X}\hat{\beta}$$

The Residual Plots option provides three visual methods to assess model fit:

Residual by Predicted Plot Shows the residuals plotted against the predicted values of *Y*. You typically want to see the residual values scattered randomly about zero.

Residual Quantile Plot Shows the quantiles of the residuals plotted against the quantiles of a standard normal distribution. Also shown is a bar chart of the residuals. If the residuals are normally distributed, the points on the normal quantile plot should approximately fall along the red diagonal line. This type of plot is also called a quantile-quantile plot, or Q-Q plot. The normal quantile plot also shows Lilliefors confidence bounds (Conover 1999).

Residual by Row Plot Shows residuals plotted against row numbers. This plot can help you detect patterns that result from the row ordering of the observations.

JMP PRO Profilers

The plots in the Marginal Model Profiler are based on marginal predicted values. These are the predicted values obtained if you select Save Columns > Prediction Formula.

Denote the linear mixed model by $E[\mathbf{Y}|\gamma] = \mathbf{X}\beta + \mathbf{Z}\gamma$. Here β is the vector of fixed effect coefficients and γ is the vector of random effect coefficients. The *marginal predictions* are the predictions from the fixed effects part of the predictive model, given by $\mathbf{X}\hat{\beta}$.

Note: Marginal model profiler plots show predictions for distinct settings of the fixed effects only. The Profiler shows only cells corresponding to fixed effects. In other profilers, where random effects can be displayed, only the settings of the fixed effects determine the predicted values.

Four types of profilers are provided:

- Profiler
- Contour Profiler
- Mixture Profiler

- Surface Profiler

Options that are appropriate for the model that you are fitting are enabled. See Figure 8.21 for an example of a profiler. See Figure 8.42 for an example of a Surface Profiler. For more information about the surface profiler, see *Profilers*.

JMP PRO Conditional Model Inference

The conditional model diagnostic plots are based on conditional residuals. Conditional residuals reflect the prediction error once both fixed and random effects have been fit.

Actual by Conditional Predicted Plot Plots actual values versus values predicted by the model, accounting for the random effects. When there are random effects, the Actual by Conditional Predicted Plot appears by default. See "Actual by Conditional Predicted Plot".

Conditional Residual Plots Provides residual plots that assess model fit, accounting for the random effects. See "Conditional Residual Plots".

Conditional Profiler, Conditional Contour Profiler, Conditional Mixture Profiler, Conditional Surface Profiler Provides profilers to examine the relationship between the response and the model terms, accounting for random effects. See "Conditional Profilers".

Variogram Provides a variogram plot that shows the change in covariance as the distance between observations increases. When the Residual structure is selected, you can select the columns to use as temporal or spatial coordinates. See "Variogram".

JMP PRO Actual by Conditional Predicted Plot

The Actual by Conditional Predicted plot appears by default. It provides a visual assessment of model fit that accounts for variation due to random effects. It plots the observed values of Y against the conditional predicted values of Y. These are the predicted values obtained if you select Save Columns > Conditional Prediction Formula.

Denote the linear mixed model by $E[\mathbf{Y}|\gamma] = \mathbf{X}\beta + \mathbf{Z}\gamma$. Here β is the vector of fixed effect coefficients and γ is the vector of random effect coefficients. The *conditional predictions* are the predictions obtained from the model given by $\mathbf{X}\hat{\beta} + \mathbf{Z}\hat{\gamma}$.

JMP PRO Conditional Residual Plots

Conditional residuals reflect the prediction error based on fitting both fixed and random effects. Conditional residuals are the differences between actual values and the conditional predicted values obtained if you select Save Columns > Conditional Prediction Formula.

Denote the linear mixed model by $E[\mathbf{Y}|\gamma] = \mathbf{X}\beta + \mathbf{Z}\gamma$. Here β is the vector of fixed effect coefficients and γ is the vector of random effect coefficients. The *conditional residuals* are calculated as follows:

$$\mathbf{r} = \mathbf{Y} - (\mathbf{X}\hat{\beta} + \mathbf{Z}\hat{\gamma})$$

The Conditional Residual Plots option provides three visual methods to assess model fit:

Conditional Residual by Predicted Plot Shows the conditional residuals plotted against the conditional predicted values of *Y*. You typically want to see the conditional residual scattered randomly about zero.

Conditional Residual Quantile Plot Shows the quantiles of the conditional residuals plotted against the quantiles of a standard normal distribution. Also shown is a bar chart of the conditional residuals. If the conditional residuals are normally distributed, the points on the normal quantile plot should approximately fall along the red diagonal line. This type of plot is also called a quantile-quantile plot, or Q-Q plot. The normal quantile plot also shows Lilliefors confidence bounds (Conover 1999).

Conditional Residual by Row Plot Shows conditional residuals plotted against row numbers. This plot can help you detect patterns that result from the row ordering of the observations.

JMP PRO Conditional Profilers

The conditional model profiler plots are based on conditional predicted values. These are the predicted values obtained if you select Save Columns > Conditional Prediction Formula.

Denote the linear mixed model by $E[\mathbf{Y}|\gamma] = \mathbf{X}\beta + \mathbf{Z}\gamma$. Here β is the vector of fixed effect coefficients and γ is the vector of random effect coefficients. The *conditional predictions* are the predictions obtained from the model given by $\mathbf{X}\hat{\beta} + \mathbf{Z}\hat{\gamma}$.

Four types of profilers are provided:

- Conditional Profiler
- Conditional Contour Profiler
- Conditional Mixture Profiler
- Conditional Surface Profiler

Options that are appropriate for the model that you are fitting are enabled. See Figure 8.21 for an example of a Profiler. See Figure 8.42 for an example of a Surface Profiler. For more information about the profiler, see *Profilers*.

JMP PRO Variogram

A Variogram plot describes the spatial or temporal correlation of observations in terms of their distance. The plot shows the semivariance as a function of distance or time. The theoretical semivariance is one half of the variance of the difference between response values at locations that are a given distance apart. Note that semivariance and correlation at a given distance are inversely related. If the correlation between values at a given distance is small, the semivariance between observations at that distance is large.

When you specify any isotropic Repeated Structure (AR(1), Spatial, or Spatial with Nugget) in the Fit Model window, a Variogram plot is shown by default. If you specify the Residual structure, selecting the Variogram option in the red triangle menu enables you to select the continuous columns to be used in calculating the variogram. You can include any number of columns that describe the spatial or temporal structure of your data.

The initial Variogram report shows a plot of the empirical semivariance against distance. For additional background and more information, see "Antedependent Covariance Structure".

Semivariance Curves for Isotropic Structures

Semivariance curves are provided for the isotropic covariance structures: AR(1), Power, Exponential, Gaussian, and Spherical. For the spatial structures, curves are provided for models with and without nuggets.

The curves for the theoretical models are fit using the covariance parameter estimates. For the underlying formulas, see Chilès and Delfiner (2012) and Cressie (1993).

Use the theoretical models to determine whether your data conform to your selected isotropic structure. If you have selected the Residual structure, you can use the empirical variogram to determine whether your data exhibit some temporal or spatial structure. If the points seem follow a horizontal line, this suggests that the correlation does not change with distance and that the Residual structure is appropriate. If the points show a pattern, fitting various isotropic models might suggest an appropriate Repeated structure with which to refit your model.

JMP PRO Nugget

The *nugget* is the vertical jump from the value of 0 at the origin of the variogram to the value of the semivariance at a very small separation distance. A variogram model with a nugget has a discontinuity at the origin. The value of the theoretical curve for distances just above 0 is the nugget.

JMP PRO Variogram Options

AR(1) Plots a variogram for an AR(1) covariance structure.

Spatial Plots a variogram for an Exponential, Gaussian, Power, or Spherical covariance structure.

Spatial with Nugget Plots a variogram for an Exponential, Gaussian, Power, or Spherical covariance structure with nugget.

JMP PRO Covariance and Correlation Matrices

The Covariance and Correlations Matrices menu contains options to view the covariance matrices that are associated with the model.

Covariance of Fixed Effects Shows or hides the covariance matrix for the fixed effects in the model.

Covariance of Covariance Parameters Shows or hides the covariance matrix for the random effects in the model. The effects in the matrix are ordered as follows: G-side random effects, R-side random effects, and residual effects.

Covariance of All Parameters Shows or hides the covariance matrix for all effects in the model. The effects in the matrix are ordered as follows: fixed effects, G-side random effects, R-side random effects, and residual effects.

Correlation of Fixed Effects Shows or hides the correlation matrix for the fixed effects in the model.

JMP PRO Save Columns

Each option in the Save Columns menu adds one or more new columns to the data table.

Prediction Formula Creates a new column called Pred Formula <colname> that contains both the formula and the marginal mean predicted values. A Predicting column property is added, noting the source of the prediction. See "Marginal Model Inference".

Standard Error of Predicted Creates a new column called StdErr Pred <colname> that contains standard errors for the predicted marginal mean responses.

Mean Confidence Interval Creates two new columns called Lower 95% Mean <colname> and Upper 95% Mean <colname>. These columns contain the lower and upper 95% confidence limits for the mean response. These intervals include the variation in the estimation, but not in the response.

Indiv Confidence Interval (Available for models that contain only G-side effects.) Creates two new columns called Lower 95% Indiv <colname> and Upper 95% Indiv <colname>.

These columns contain lower and upper 95% confidence limits for individual response values. These intervals include the variation in both the response and its estimation.

Residuals Creates a new column called Residual <colname> that contains the observed response values minus their marginal mean predicted values. See "Marginal Model Inference".

Conditional Prediction Formula Creates a new column called Cond Pred Formula <colname> that contains both the formula and the conditional mean predicted values. A Predicting column property is added, noting the source of the prediction. See "Conditional Profilers".

Standard Error of Conditional Predicted Creates a new column called StdErr Cond Pred <colname> that contains standard errors for the predicted conditional mean responses.

Conditional Mean CI (Available for models that contain a G-side effect.) Creates two new columns called Lower 95% Cond Mean <colname> and Upper 95% Cond Mean <colname>. These columns contain the lower and upper 95% confidence limits for the expected value from conditional prediction. The confidence intervals include random effect estimates for models with random effects. See "Conditional Model Inference".

Conditional Residuals Creates a new column called Cond Residual <colname> that contains the observed response values minus their conditional mean predicted values. See "Conditional Model Inference".

Save Simulation Formula (Available only for variance component and random coefficient models.) Saves a column to the data table that contains a formula that generates simulated values using the estimated parameters for the model that you fit. This column can be used in the Simulate utility as a Column to Switch In. See *Basic Analysis*.

JMP PRO Additional Examples of the Mixed Models Personality

- "Repeated Measures Example"
- "Split Plot Example"
- "Spatial Example: Uniformity Trial"
- "Correlated Response Example"

JMP PRO Repeated Measures Example

Consider the Cholesterol Stacked.jmp sample data table. A study was performed to test two new cholesterol drugs against a control drug. Twenty patients with high cholesterol were randomly assigned to each of four treatments (the two experimental drugs, the control, and a placebo). Each patient's total cholesterol was measured at six times during the study: the first day in April, May, and June in the morning and afternoon. You are interested in whether either of the new drugs is effective at lowering cholesterol and in whether time and treatment interact.

JMP PRO Background

Two methods have historically been used to analyze such a design:

- Multivariate analysis of variance (MANOVA)
- A split-plot in time univariate analysis of variance (ANOVA) with either the Huynh-Feldt (1976) or Greenhouse-Geisser (1959) correction

Both of these options are available using the MANOVA personality in Fit Model. These two options are the two extremes for modeling the covariance structure. The MANOVA analysis assumes an unstructured covariance structure where all variances and covariances are estimated individually. The independent split-plot in time analysis assumes that all errors are independent. In the Gaussian data case, this is equivalent to assuming a compound symmetry covariance structure.

These two models can result in vastly different conclusions about the treatment effects. When you assume a complex covariance structure, information in the data is used to estimate the covariance parameters. If you fit too many covariance parameters, you run the risk of overfitting your model. When you model repeated measures data, you must find a covariance structure that balances these issues.

- When the model is overfit, the power to detect differences is smaller than if you were to assume a less complex covariance structure.
- When the model is underfit, Type I error control is lost. In some cases, this leads to inflated rejection rates. In other cases, decreased rejection rates occur due to inflated variance.

JMP PRO Covariance Structures

The Mixed Model personality fits a variety of covariance structures. For repeated measures in time, both the Toeplitz covariance structure and the first-order autoregressive (AR(1)) covariance structures often provide appropriate correlation structures. These structures allow for correlated observations without overfitting the model. The AR(1) assumes a common variance parameter, whereas the Toeplitz covariance matrix with unequal variances estimates unique variances for each unit of the repeated measure variable. See "Repeated Covariance Structure Requirements".

In this example, you fit the four covariance structures. The number of observation times, *J*, is equal to six.

- "Covariance Structure: Unstructured". The Unstructured model fits all covariance parameters, $J(J+1)/2$ in total. In this example, the model fits 21 variances.
- "Covariance Structure: Residual". The Residual model is equivalent to the usual variance components structure. In this example, the model fits two variances.
- "Covariance Structure: Toeplitz". The Toeplitz model fits $2J-1$ covariance parameters. In this example, the model fits 11 variances.
- "Covariance Structure: AR(1)". This model fits two covariance parameters. One parameter determines the variance and the other determines how the covariance changes with time.

You use AICc to evaluate model fits. The BIC criterion can also be used. In this case, the same model is chosen by both criteria. You select a best covariance structure and then continue to do additional analysis:

- "Further Analysis Using AR(1) Structure"
- "Regression Model for AR(1) Model Example"

Tip: Leave the Fit Model launch window open as you work through this example.

JMP PRO Data Structure

The Cholesterol.jmp data table is in a format that is typically used for recording repeated measures data. To use the Mixed Model personality to analyze these data, each cholesterol measurement needs to be in its own row, as in Cholesterol Stacked.jmp. To construct Cholesterol Stacked.jmp, the data in Cholesterol.jmp were stacked using Tables > Stack.

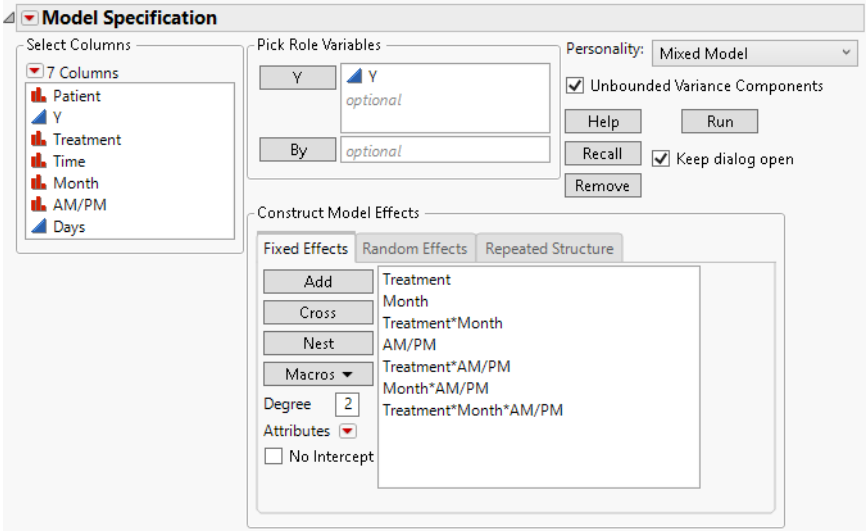
The Days column in the stacked table was constructed using a formula. The Days column gives the number days into the study when the cholesterol measurement was taken. Its modeling type is continuous. This is necessary because the AR(1) covariance structure requires the repeated effect be continuous.

JMP PRO Covariance Structure: Unstructured

Begin by fitting a model using an Unstructured covariance structure.

1. Select **Help > Sample Data Library** and open Cholesterol Stacked.jmp.
2. Select **Analyze > Fit Model**.
3. Select **Keep dialog open** so that you can return to the launch window in the next example.
4. Select Y and click **Y**.
5. Select **Mixed Model** from the Personality list.
6. Select Treatment, Month, and AM/PM, and then select **Macros > Full Factorial**.

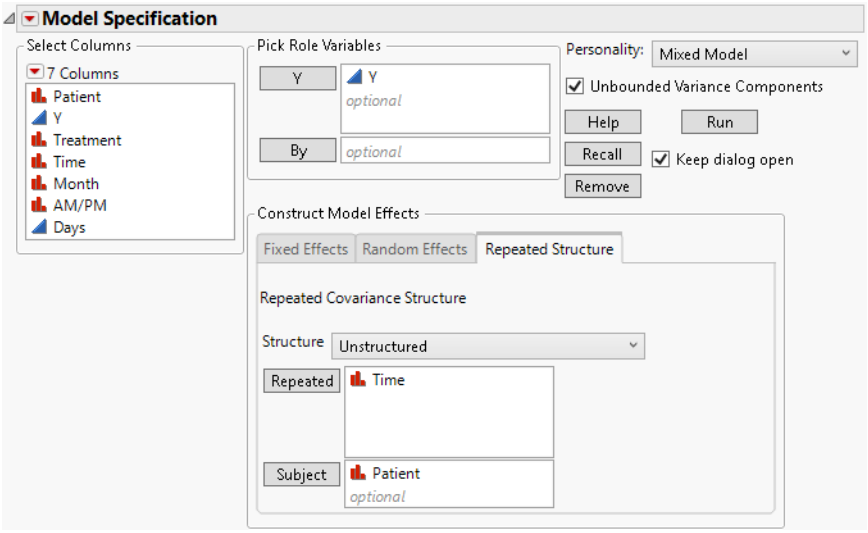
Figure 8.12 Fit Model Launch Window Showing Completed Fixed Effects Tab



7. Select the **Repeated Structure** tab.
8. Select **Unstructured** from the Structure list.
9. Select Time and click **Repeated**. The **Repeated** column defines the repeated measures within a subject.
10. Select Patient and click **Subject**.

Note: The Unstructured covariance model does not allow the repeated structure variables to assume duplicate values. Suppose that, in this example, the subject was nested within treatment, and that the patients had been numbered using the values 1, 2, 3, 4, and 5 within each treatment. A warning would be given when you run this analysis. You would need to renumber the patients to have different identifiers for each value of the Repeated variable. Or you would need to create a column in the data table that represents nesting within treatment and enter this effect as Subject.

Figure 8.13 Fit Model Launch Window Showing Completed Repeated Structure Tab



11. Click **Run**.

    The Fit Mixed report is shown in Figure 8.14. Because you want to compare your three models using AICc or BIC, you are interested in the Fit Statistics report. The AICc for the unstructured model is 703.84.

    The Repeated Effects Covariance Parameter Estimates report shows estimates of all 21 covariance parameters. As you would expect, observations taken closer in time have higher covariance than those farther apart. Also, variance increases with time.

Figure 8.14 Fit Mixed Report - Unstructured Covariance Structure

Fit Mixed

Actual by Predicted Plot

Fit Statistics

-2 Residual Log Likelihood	554.98818
-2 Log Likelihood	557.89101
AICc	703.83696
BIC	773.32814

Repeated Effects Covariance Parameter Estimates

Repeated Effect: Time
Subject: Patient

Covariance Parameter	Estimate	Std Error	95% Lower	95% Upper
Var(April AM)	18.725	6.6202872	5.7494754	31.700525
Cov(April PM,April AM)	18.354884	6.6035621	5.4121399	31.297628
Var(April PM)	19.268932	6.8125963	5.9164888	32.621376
Cov(May AM,April AM)	9.2756709	8.4629182	-7.311344	25.862686
Cov(May AM,April PM)	5.5074058	8.3704001	-10.89828	21.913089
Var(May AM)	56.603347	20.012305	17.379949	95.826744
Cov(May PM,April AM)	9.4147226	8.5038584	-7.252534	26.081979
Cov(May PM,April PM)	6.6230805	8.4532303	-9.944946	23.191108
Cov(May PM,May AM)	55.365523	19.83529	16.489068	94.241978
Var(May PM)	57.058089	20.173081	17.519577	96.596602
Cov(June AM,April AM)	1.1945478	8.6266098	-15.7133	18.102392
Cov(June AM,April PM)	0.3183447	8.7461245	-16.82374	17.460434
Cov(June AM,May AM)	1.106725	14.992149	-28.27735	30.490796
Cov(June AM,May PM)	0.6455101	15.050552	-28.85303	30.14405
Var(June AM)	63.512277	22.454981	19.501323	107.52323
Cov(June PM,April AM)	1.5810194	8.7687993	-15.60551	18.76755
Cov(June PM,April PM)	0.7647113	8.8882627	-16.65596	18.185386
Cov(June PM,May AM)	0.9262595	15.232066	-28.92804	30.780561
Cov(June PM,May PM)	0.6543895	15.292238	-29.31785	30.626625
Cov(June PM,June AM)	63.878686	22.700344	19.386829	108.37054
Var(June PM)	65.568482	23.181959	20.132677	111.00429

Fixed Effects Parameter Estimates

Fixed Effects Tests

Source	Nparm	DFNum	DFDen	F Ratio	Prob > F
Treatment	3	3	16.0	274.96713	<.0001*
Month	2	2	15.0	340.48166	<.0001*
Treatment*Month	6	6	18.3	123.47461	<.0001*
AM/PM	1	1	16.0	360.93593	<.0001*
Treatment*AM/PM	3	3	16.0	0.6339843	0.6038
Month*AM/PM	2	2	15.0	1.1988247	0.3289
Treatment*Month*AM/PM	6	6	18.3	1.1642781	0.3671

JMP PRO Covariance Structure: Residual

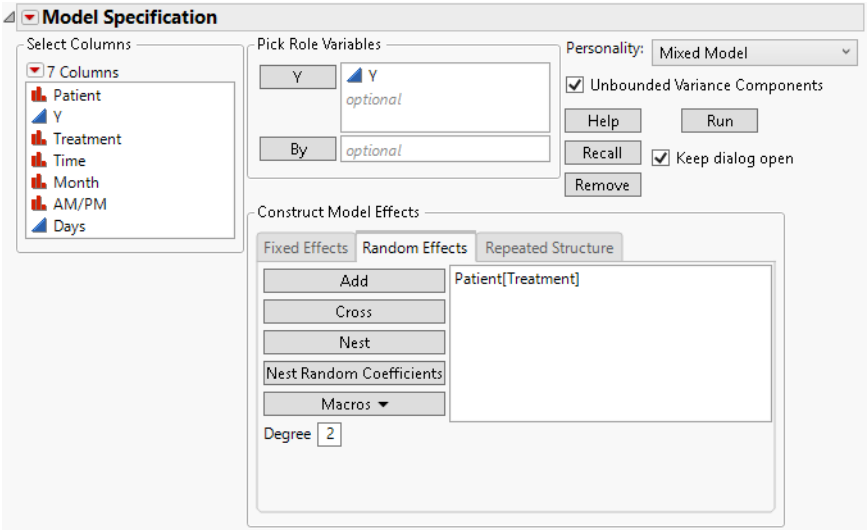
The Residual covariance structure is appropriate when you fit a split-plot model.

1. Complete step 1 through step 7 in "Covariance Structure: Unstructured".
2. On the **Repeated Structure** tab, select **Residual** from the Structure list.
3. If you are continuing from the previous example, remove Time and Patient.

   Otherwise, a warning appears: "Repeated columns and subject columns are ignored when the Residual covariance structure is selected." You are given the option to click **OK** to continue the analysis.

4. Select the **Random Effects** tab.
5. Select Patient and click **Add**.
6. Select Patient in the Random Effects area, select the Treatment column, and then click **Nest**.

Figure 8.15 Fit Model Launch Window Showing Completed Random Effects Tab



7. Click **Run**.

   The Fit Mixed report is shown in Figure 8.16. The Fit Statistics report shows that the AICc for the Residual model is 832.55, as compared to 703.84 for the Unstructured model.

   The estimates of the two covariance parameters are shown in the Random Effects Covariance Parameter Estimates report. These are estimates of the variance of Patient within Treatment, and of the Residual variance.

Figure 8.16 Fit Mixed Report - Residual Error Covariance Structure

Fit Mixed

Actual by Predicted Plot **Actual by Conditional Predicted Plot**

Fit Statistics

-2 Residual Log Likelihood	721.03949
-2 Log Likelihood	765.45515
AICc	832.55192
BIC	889.92993

Random Effects Covariance Parameter Estimates

Variance Component	Estimate	Std Error	95% Lower	95% Upper	Wald p-Value
Patient[Treatment]	11.707432	6.2749012	-0.591148	24.006012	0.0621
Residual	35.081923	5.546939	26.320843	49.105827	
Total	46.789355	7.7386523	34.678063	66.607248	

Fixed Effects Parameter Estimates

Random Coefficients

Fixed Effects Tests

Source	Nparm	DFNum	DFDen	F Ratio	Prob > F
Treatment	3	3	16.0	274.96713	<.0001*
Month	2	2	80.0	622.88066	<.0001*
Treatment*Month	6	6	80.0	226.49464	<.0001*
AM/PM	1	1	80.0	13.700114	0.0004*
Treatment*AM/PM	3	3	80.0	0.0240643	0.9949
Month*AM/PM	2	2	80.0	0.0324977	0.9680
Treatment*Month*AM/PM	6	6	80.0	0.030708	0.9999

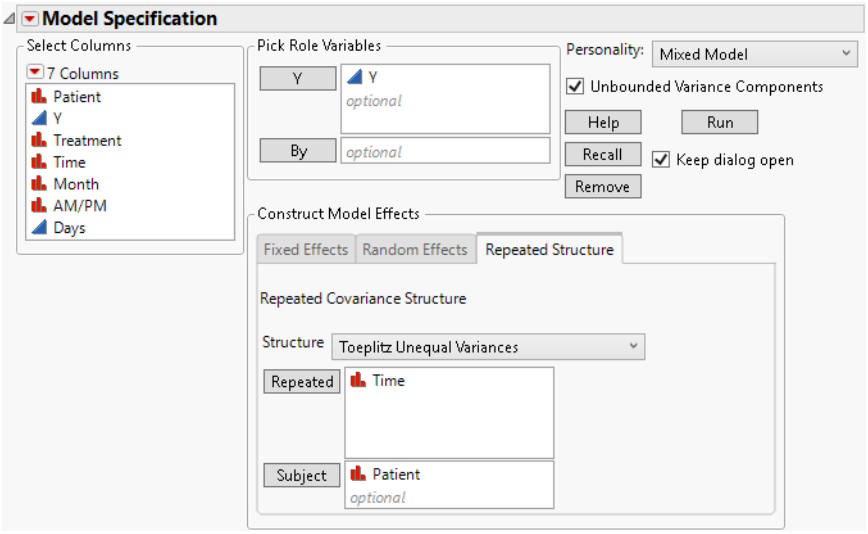
JMP PRO Covariance Structure: Toeplitz

Fit the model using the Toeplitz Unequal Variances structure.

1. Complete step 1 through step 6 in "Covariance Structure: Unstructured".
2. If you are continuing from the previous example, select Patient[Treatment] on the **Random Effects** tab and then click **Remove**.

   If you include both random effects and repeated effects, there is often insufficient data to estimate both effects.
3. Select the **Repeated Structure** tab.
4. Select **Toeplitz Unequal Variances** from the Structure list.
5. Select Time and click **Repeated**.
6. Select Patient and click **Subject**.

Figure 8.17 Fit Model Launch Window Showing Completed Repeated Structure Tab



7. Click **Run**.

Figure 8.18 Fit Mixed Report - Toeplitz Unequal Variances Structure

Fit Mixed

Actual by Predicted Plot

Fit Statistics

-2 Residual Log Likelihood	659.09915
-2 Log Likelihood	688.02972
AICc	788.02972
BIC	855.59193

Repeated Effects Covariance Parameter Estimates

Subject: Patient

Covariance Parameter	Estimate	Std Error	95% Lower	95% Upper
Toeplitz Correlation(1)	0.6230225	0.0856377	0.4551756	0.7908693
Toeplitz Correlation(2)	0.2555783	0.1463264	-0.031216	0.5423728
Toeplitz Correlation(3)	0.0962006	0.1887883	-0.273818	0.466219
Toeplitz Correlation(4)	-0.1112	0.2248683	-0.551934	0.3295338
Toeplitz Correlation(5)	0.2427218	0.1955609	-0.140571	0.6260141
Variance(April AM)	17.552396	6.9782256	3.8753253	31.229467
Variance(April PM)	24.652754	8.3215651	8.3427857	40.962721
Variance(May AM)	73.950898	25.930509	23.128035	124.77376
Variance(May PM)	63.668022	21.766899	21.005684	106.33036
Variance(June AM)	69.45679	22.578117	25.204495	113.70909
Variance(June PM)	52.484909	19.171565	14.909332	90.060486

Fixed Effects Parameter Estimates

Fixed Effects Tests

Source	Nparm	DFNum	DFDen	F Ratio	Prob > F
Treatment	3	3	14.1	230.25344	<.0001*
Month	2	2	23.7	363.8599	<.0001*
Treatment*Month	6	6	26.2	134.01838	<.0001*
AM/PM	1	1	5.2	106.73459	0.0001*
Treatment*AM/PM	3	3	5.2	0.1874794	0.9007
Month*AM/PM	2	2	26.9	0.036706	0.9640
Treatment*Month*AM/PM	6	6	34.6	0.0347301	0.9998

Note: The Mixed Models personality in JMP reports correlations, whereas PROC MIXED in SAS reports covariances.

The Fit Statistics report shows that the AICc for the Toeplitz with Unequal Variances model is 788.03. Compare this number to 832.55 for the Residual Model and 703.84 for the Unstructured model.

The Toeplitz Unequal Variances structure requires the estimation of eleven covariance parameters. These estimates are shown in the Repeated Effects Covariance Parameter Estimates report. The Toeplitz correlation estimates are shown, followed by the variance estimates for each time point. See "Repeated Measures" for information about how this matrix is parameterized.

JMP PRO Covariance Structure: AR(1)

Finally, fit the AR(1) structure.

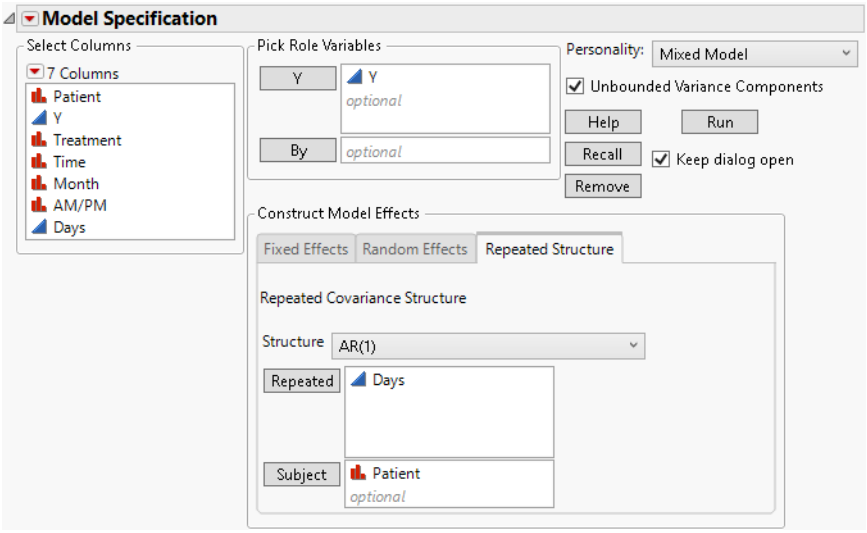
1. Complete step 1 through step 6 in "Covariance Structure: Unstructured".

2. If you are continuing from the previous example, select Time in the **Repeated** box and then click **Remove**.

   AR(1) requires a continuous variable for the repeated value.

3. Select **AR(1)** from the Structure list.
4. Select Days and click **Repeated**.

Figure 8.19 Fit Model Launch Window Showing Completed Repeated Structure Tab



5. Click **Run**.

   The Fit Mixed report is shown in Figure 8.20. The Fit Statistics report shows that the AICc for the AR(1) model is 652.63. Compare this number to 832.55 for the Residual model, 703.84 for the Unstructured model, and 788.03 for the Toeplitz Unequal Variances model. Based on the AICc criterion, the AR(1) model is the best of the four models.

   The AR(1) structure requires the estimation of two covariance parameters. These estimates are shown in the Repeated Effects Covariance Parameter Estimates report. The AR(1) Days parameter estimate is an estimate of ρ, the correlation parameter in the AR(1) structure.

   The Variogram plot shows the empirical semivariances and the curve for the AR(1) model. Since there are only five nonzero values for Days, only four distance classes are possible and only four points are shown. The AR(1) structure seems appropriate. To explore other structures, select options from the Variogram red triangle menu. For more information about Variogram options, see "Variogram".

Figure 8.20 Fit Mixed Report - AR(1) Covariance Structure

Fit Mixed

Actual by Predicted Plot

Fit Statistics

-2 Residual Log Likelihood	577.09954
-2 Log Likelihood	585.53021
AICc	652.62698
BIC	710.00499

Repeated Effects Covariance Parameter Estimates

Subject: Patient

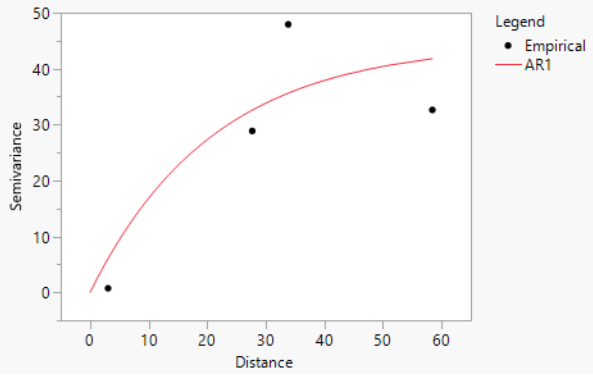
Covariance Parameter	Estimate	Std Error	95% Lower	95% Upper
AR(1) Days	0.9536532	0.0120225	0.9300895	0.9772169
Residual	44.579925	9.1281285	30.970217	69.685221

Fixed Effects Parameter Estimates

Fixed Effects Tests

Source	Nparm	DFNum	DFDen	F Ratio	Prob > F
Treatment	3	3	22.8	242.61161	<.0001*
Month	2	2	80.8	266.42269	<.0001*
Treatment*Month	6	6	82.7	98.224973	<.0001*
AM/PM	1	1	57.6	461.72981	<.0001*
Treatment*AM/PM	3	3	57.6	0.8110288	0.4930
Month*AM/PM	2	2	58.0	1.0871136	0.3440
Treatment*Month*AM/PM	6	6	58.0	1.0275595	0.4168

Variogram



JMP PRO Further Analysis Using AR(1) Structure

Because the AR(1) model gives the best fit, you adopt it as your model and proceed with your analysis. The Fixed Effects Tests report indicates that there is a significant interaction between Treatment and Month as well as a main effect of AM/PM. Here, we explore these significant effects.

1. Click the Fit Mixed red triangle and select **Marginal Model Inference > Profiler**.

   The Marginal Model Profiler report (Figure 8.21) enables you to see the effect on cholesterol levels (Y) for various settings of Treatment, Month, and AM/PM.

2. In the plot for Month, drag the vertical dotted red line from April to May and then to June.

Notice that the predicted AM measurements for Y decrease over the three months from a mean of 277.4 in April to a mean of 177.7 in June.

3. In the plot for Treatment, drag the vertical dotted red line from A to B.

   By dragging the line in the plot for Month from April to June, you see that, for Treatment B, the predicted AM mean for Y decreases from 276.8 in April to 191.2 in June.

4. In the plot for Treatment, drag the vertical dotted line to Control and then to Placebo.

   Notice that when you set Treatment to Control or Placebo, you see virtually no change over the three months (Figure 8.22).

   Next, you explore the effect of AM/PM.

5. Set Treatment and Month to all twelve combinations of their levels by dragging the vertical red lines.

   For all twelve combinations, the predicted cholesterol level is consistently higher in the afternoon than in the morning, demonstrating the main effect.

Note that Treatment A seems to result in lower cholesterol readings in May than Treatment B does. If this effect is significant, it might indicate that Treatment A acts more quickly than B. The next section, "Compare All Treatments in June", shows you how to evaluate the treatments.

Figure 8.21 Marginal Profiler Plot for Treatment A

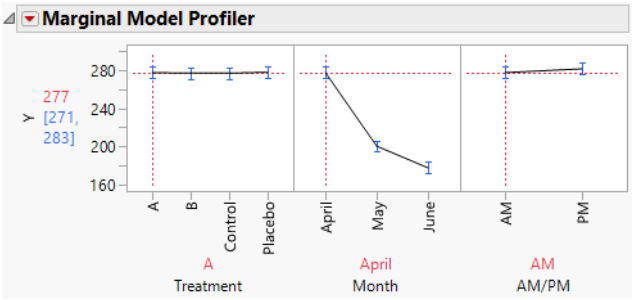
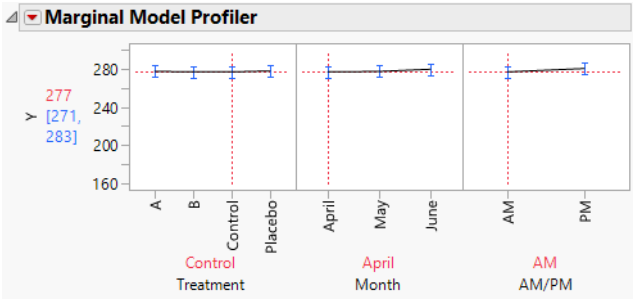


Figure 8.22 Marginal Profiler Plot for Control

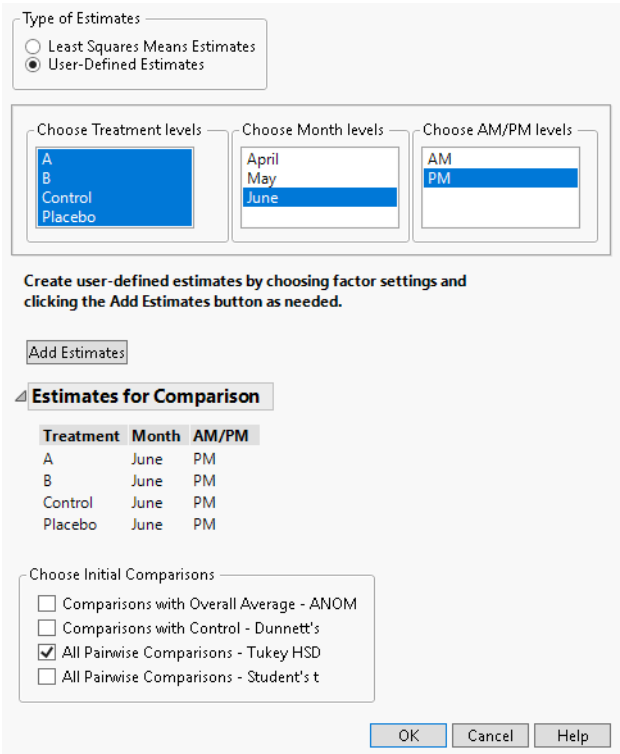


Compare All Treatments in June

The study is conducted over the months of April, May, and June. You are interested in which treatments differ on the PM measurement in June.

1. Click the Fit Mixed red triangle and select **Multiple Comparisons**.
2. Under Types of Estimates, select **User-Defined Estimates**.
3. From the Choose Treatment levels panel, select all four treatment types.
4. From the Choose Month levels panel, select June.
5. From the Choose AM/PM levels panel, select PM.
6. Click Add Estimates.
7. From the Choose Initial Comparisons list, select **All Pairwise Comparisons - Tukey HSD**.

Figure 8.23 Completed Multiple Comparisons Window



8. Click **OK**.

Figure 8.24 Tukey HSD All Pairwise Comparisons Report for All Treatments for June PM

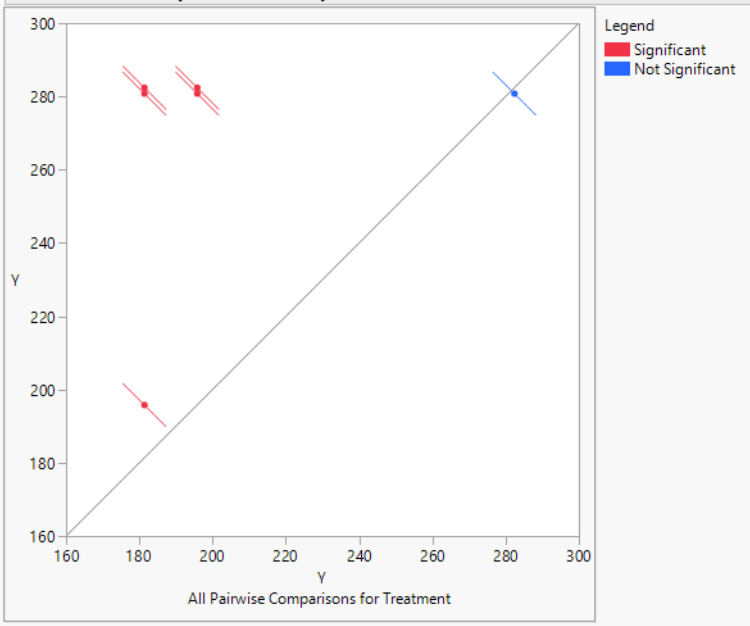
Tukey HSD All Pairwise Comparisons

Month = June, AM/PM = PM
Quantile = 2.76931, Adjusted DF = 22.8, Adjustment = Tukey

All Pairwise Differences

Treatment	-Treatment	Difference	Std Error	t Ratio	Prob>|t|	Lower 95%	Upper 95%
A	B	-14.414	4.222792	-3.41	0.0120*	-26.108	-2.7196
A	Control	-101.014	4.222792	-23.92	<.0001*	-112.708	-89.3196
A	Placebo	-99.414	4.222792	-23.54	<.0001*	-111.108	-87.7196
B	Control	-86.600	4.222792	-20.51	<.0001*	-98.294	-74.9058
B	Placebo	-85.000	4.222792	-20.13	<.0001*	-96.694	-73.3058
Control	Placebo	1.600	4.222792	0.38	0.9810	-10.094	13.2942

All Pairwise Comparisons Scatterplot



The Tukey HSD All Pairwise Comparisons report shows an All Pairwise Differences report and an All Pairwise Comparisons Scatterplot. All treatments other than the Control and Placebo differ significantly on the June PM measurements.

Consider the difference between treatments A and B. The difference in means is -14.414 and the confidence interval ranges from -26.108 to -2.7196. You conclude that the reduction in cholesterol measurements due to treatment A exceeds the reduction by treatment B by somewhere between 2.7 and 26.1 points. Both treatments A and B are highly effective compared to the Control and Placebo.

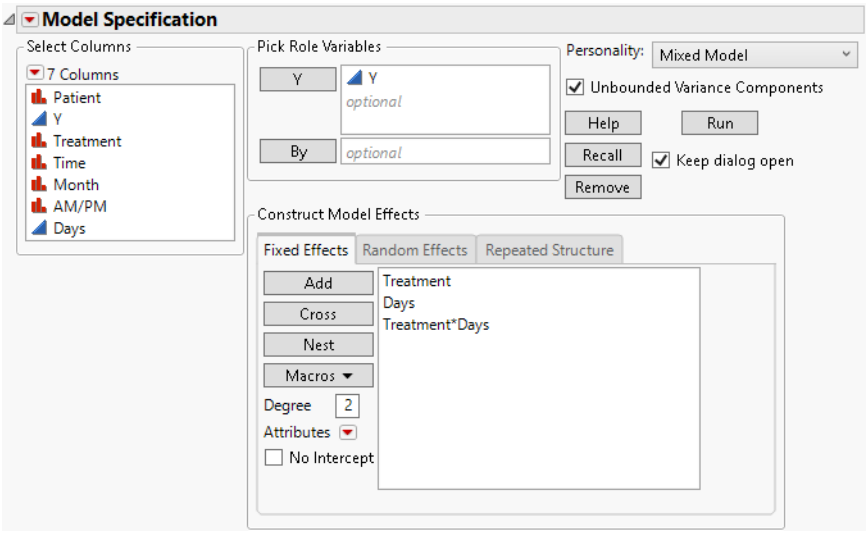
JMP PRO Regression Model for AR(1) Model Example

Using the Month and AM/PM categorical effects, you have compared four covariance structures for the cholesterol data. (Note that a categorical effect was required for the Unstructured fit.) You have decided to use an AR(1) covariance structure.

Suppose now that you want to model the effect of treatment in terms of the continuous effect Days instead of the categorical effects. You can then predict cholesterol levels at arbitrary time during treatment.

1. After following step 1 to step 4 in "Covariance Structure: AR(1)", return to the Fit Model launch window.
2. On the Fixed Effects tab, select the existing fixed effects and click **Remove**.
3. Select Treatment and Days then select **Macros > Full Factorial**.

Figure 8.25 Fit Model Launch Window Showing Fixed Effects Tab



4. Click the Model Specification red triangle and deselect **Center Polynomials**.

Note: In the default setting, the continuous effects used in interaction terms are centered. By turning off the Center Polynomials option, the continuous effects used in interaction terms are not centered.

5. Click **Run**.

The Fit Mixed report is shown in Figure 8.26. You see that the interaction of Treatment and Days is highly significant indicating different regressions for the drugs.

Note: To predict outcomes for the drugs at different days, use the profiler. See *Profilers*.

Figure 8.26 Fit Mixed Report - AR(1) Covariance Structure with Continuous Fixed Effect

Fit Mixed

Fit Statistics

-2 Residual Log Likelihood	791.54856
-2 Log Likelihood	789.76817
AICc	811.78652
BIC	837.64309

Repeated Effects Covariance Parameter Estimates

Subject: Patient

Covariance Parameter	Estimate	Std Error	95% Lower	95% Upper
AR(1) Days	0.7983047	0.0473108	0.7055772	0.8910321
Residual	100.5535	18.519317	72.239037	149.60179

Fixed Effects Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|	95% Lower	95% Upper
Intercept	276.17699	1.99461	52.7	138.46	<.0001*	272.17583	280.17815
Treatment[A]	-6.13599	3.4547658	52.7	-1.78	0.0815	-13.0662	0.794217
Treatment[B]	-0.352826	3.4547658	52.7	-0.10	0.9190	-7.283033	6.5773802
Treatment[Control]	1.9453259	3.4547658	52.7	0.56	0.5758	-4.984881	8.8755326
Days	-0.735803	0.0505394	53.3	-14.56	<.0001*	-0.83716	-0.634446
Treatment[A]*Days	-0.877856	0.0875368	53.3	-10.03	<.0001*	-1.053411	-0.702301
Treatment[B]*Days	-0.643666	0.0875368	53.3	-7.35	<.0001*	-0.819221	-0.468111
Treatment[Control]*Days	0.788161	0.0875368	53.3	9.00	<.0001*	0.6126061	0.963716

Fixed Effects Tests

Source	Nparm	DFNum	DFDen	F Ratio	Prob > F
Treatment	3	3	52.7	1.3028976	0.2832
Days	1	1	53.3	211.96431	<.0001*
Treatment*Days	3	3	53.3	76.472836	<.0001*

JMP PRO Split Plot Example

The Mixed Model personality offers a straightforward approach to specifying and analyzing split-plot experiments. Mixed Model provides tabs for specifying effects. The resulting analysis is targeted to random effects. Note, however, that split-plot experiments can also be analyzed using the Standard Least Squares personality.

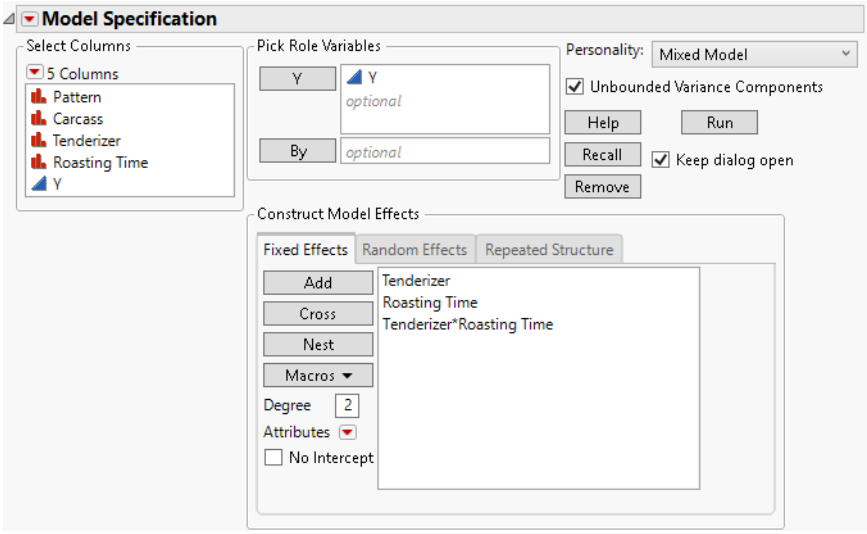
The data in the Split Plot.jmp sample data table come from a study of the effects of tenderizer and length of cooking time on meat. Six beef carcasses were randomly selected from carcasses at a meat packaging plant. From the right rib-eye muscle of each carcass, three rolled roasts were prepared under uniform conditions. Each of these three roasts was assigned a tenderizing treatment at random. After treatment, a coring device was used to mark four cores of meat near the center of each.

The three roasts from the same carcass were placed together in a preheated oven and allowed to cook. After 30 minutes, one of the cores was taken at random from each roast. Cores were removed in this fashion again after 36 minutes, 42 minutes, and 48 minutes. As each set cooled to serving temperature, the cores were measured for tenderness using the Warner-Bratzler device. Larger measurements indicate tougher meat.

Your interest centers on the effects of tenderizer, roasting time, and especially whether there is an interaction between tenderizer and roasting time. This design addresses that goal.

1. Select **Help > Sample Data Library** and open Split Plot.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Select **Mixed Model** from the Personality list.
5. Select Tenderizer and Roasting Time, and then select **Macros > Full Factorial**.

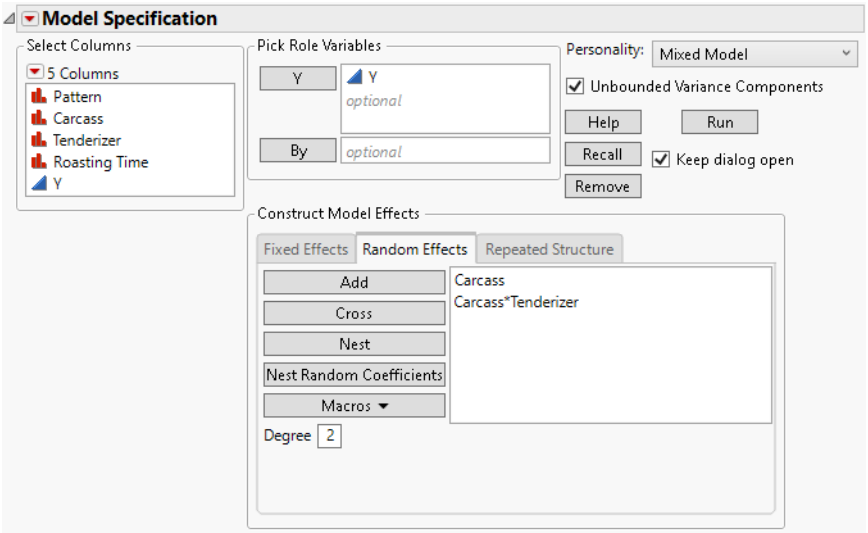
Figure 8.27 Fit Model Launch Window Showing Completed Fixed Effects Tab



6. Select the **Random Effects** tab.
7. Select Carcass and click **Add** to create the random carcass effect.
8. Select Carcass and Tenderizer and click **Cross**.

   The Carcass*Tenderizer interaction is the error term for the whole plot factor, Tenderizer. This is equivalent to the Carcass*Tenderizer&Random term in Standard Least Squares.

Figure 8.28 Fit Model Launch Window Showing Completed Random Effects Tab



9. Click **Run**.

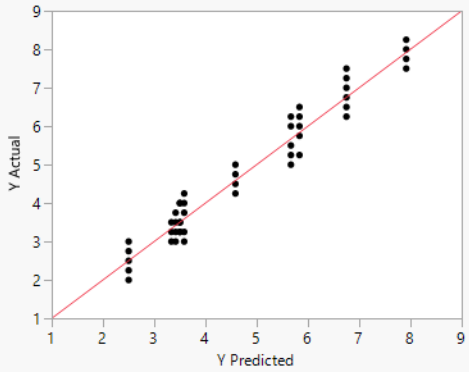
   The Fit Mixed report is shown in Figure 8.29.

   The Actual by Predicted Plot and the Actual by Conditional Predicted Plot show no issues with model fit, so you can proceed to interpret the results. The Fixed Effects Tests report indicates that there is a significant interaction between tenderizer and roasting time.

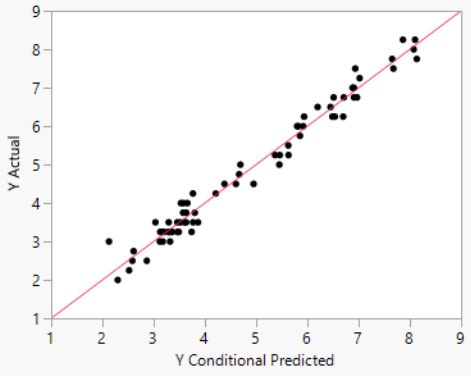
Figure 8.29 Fit Mixed Report

Fit Mixed

Actual by Predicted Plot



Actual by Conditional Predicted Plot



Fit Statistics

-2 Residual Log Likelihood	68.324206
-2 Log Likelihood	23.659713
AICc	62.231142
BIC	87.809705

Random Effects Covariance Parameter Estimates

Variance Component	Estimate	Std Error	95% Lower	95% Upper	Wald p-Value
Carcass	0.0605046	0.0411906	-0.020227	0.1412367	0.1419
Carcass*Tenderizer	-0.007957	0.0076066	-0.022865	0.006952	0.2956
Residual	0.0863419	0.0182025	0.0594003	0.1369726	
Total	0.1468465	0.0450332	0.0871674	0.2983132	

Fixed Effects Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|	95% Lower	95% Upper
Intercept	4.7777778	0.1041214	5.0	45.89	<.0001*	4.5101223	5.0454333
Tenderizer[Control]	0.6388889	0.0389142	10.0	16.42	<.0001*	0.552182	0.7255958
Tenderizer[Papain]	-0.673611	0.0389142	10.0	-17.31	<.0001*	-0.760318	-0.586904
Roasting Time[30]	2.0555556	0.0599798	45.0	34.27	<.0001*	1.93475	2.1763611
Roasting Time[36]	0.8888889	0.0599798	45.0	14.82	<.0001*	0.7680833	1.0096944
Roasting Time[42]	-1.25	0.0599798	45.0	-20.84	<.0001*	-1.370806	-1.129194
Tenderizer[Control]*Roasting Time[30]	0.4444444	0.0848243	45.0	5.24	<.0001*	0.2735996	0.6152893
Tenderizer[Control]*Roasting Time[36]	0.4444444	0.0848243	45.0	5.24	<.0001*	0.2735996	0.6152893
Tenderizer[Control]*Roasting Time[42]	-0.583333	0.0848243	45.0	-6.88	<.0001*	-0.754178	-0.412488
Tenderizer[Papain]*Roasting Time[30]	-0.326389	0.0848243	45.0	-3.85	0.0004*	-0.497234	-0.155544
Tenderizer[Papain]*Roasting Time[36]	-0.409722	0.0848243	45.0	-4.83	<.0001*	-0.580567	-0.238877
Tenderizer[Papain]*Roasting Time[42]	0.6458333	0.0848243	45.0	7.61	<.0001*	0.4749885	0.8166782

Random Coefficients

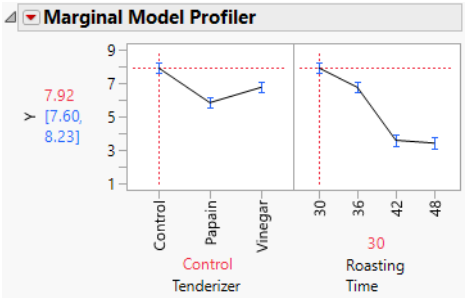
Fixed Effects Tests

Source	Nparm	DFNum	DFDen	F Ratio	Prob > F
Tenderizer	2	2	10.0	189.99483	<.0001*
Roasting Time	3	3	45.0	656.62735	<.0001*
Tenderizer*Roasting Time	6	6	45.0	18.458596	<.0001*

Explore the Interaction between Tenderizer and Roasting Time

1. Click the Fit Mixed red triangle and select **Marginal Model Inference > Profiler**.

Figure 8.30 Marginal Model Profiler with Roasting Time Set to 30 Minutes



2. Move the red dashed vertical line in the Roasting Time panel to 36, 42, and 48.

   In Figure 8.30, notice that both the papain and vinegar tenderizers result in significantly lower tenderness scores than the control when roasting time is either 30 or 36 minutes. However, at 42 minutes, there are no significant differences. At 48 minutes, papain gives a value lower than the control, but vinegar does not. Papain gives lower tenderness scores than does vinegar at all times except 42 minutes.

3. Click the Fit Mixed red triangle and select **Multiple Comparisons**.

4. Select **Tenderizer*Roasting Time**.

5. Select **All Pairwise Comparisons - Tukey HSD** and then click **OK**.

   Figure 8.31 shows a partial list of pairwise comparisons. Most of the differences between papain and vinegar that you observed in the profiler are statistically significant. Therefore, it appears that papain is the better tenderizer.

Figure 8.31 Multiple Comparisons, Partial View

Fit Mixed

Multiple Comparisons for Tenderizer*Roasting Time

Least Squares Means Estimates

Tenderizer	Roasting Time	Estimate	Std Error	DF	Lower 95%	Upper 95%
Control	30	7.9166667	0.15214568	20.355	7.5996506	8.2336827
Control	36	6.7500000	0.15214568	20.355	6.4329840	7.0670160
Control	42	3.5833333	0.15214568	20.355	3.2663173	3.9003494
Control	48	3.4166667	0.15214568	20.355	3.0996506	3.7336827
Papain	30	5.8333333	0.15214568	20.355	5.5163173	6.1503494
Papain	36	4.5833333	0.15214568	20.355	4.2663173	4.9003494
Papain	42	3.5000000	0.15214568	20.355	3.1829840	3.8170160
Papain	48	2.5000000	0.15214568	20.355	2.1829840	2.8170160
Vinegar	30	6.7500000	0.15214568	20.355	6.4329840	7.0670160
Vinegar	36	5.6666667	0.15214568	20.355	5.3496506	5.9836827
Vinegar	42	3.5000000	0.15214568	20.355	3.1829840	3.8170160
Vinegar	48	3.3333333	0.15214568	20.355	3.0163173	3.6503494

Tukey HSD All Pairwise Comparisons

Quantile = 3.44501, Adjusted DF = 45.0, Adjustment = Tukey-Kramer

All Pairwise Differences

Tenderizer	Roasting Time	-Tenderizer	-Roasting Time	Difference	Std Error	t Ratio	Prob>|t|	Lower 95%	Upper 95%
Control	30	Control	36	1.16667	0.1696485	6.88	<.0001*	0.58223	1.75111
Control	30	Control	42	4.33333	0.1696485	25.54	<.0001*	3.74889	4.91777
Control	30	Control	48	4.50000	0.1696485	26.53	<.0001*	3.91556	5.08444
Control	30	Papain	30	2.08333	0.1616429	12.89	<.0001*	1.52647	2.64019
Control	30	Papain	36	3.33333	0.1616429	20.62	<.0001*	2.77647	3.89019
Control	30	Papain	42	4.41667	0.1616429	27.32	<.0001*	3.85981	4.97353
Control	30	Papain	48	5.41667	0.1616429	33.51	<.0001*	4.85981	5.97353
Control	30	Vinegar	30	1.16667	0.1616429	7.22	<.0001*	0.60981	1.72353
Control	30	Vinegar	36	2.25000	0.1616429	13.92	<.0001*	1.69314	2.80686
Control	30	Vinegar	42	4.41667	0.1616429	27.32	<.0001*	3.85981	4.97353
Control	30	Vinegar	48	4.58333	0.1616429	28.35	<.0001*	4.02647	5.14019
Control	36	Control	42	3.16667	0.1696485	18.67	<.0001*	2.58223	3.75111
Control	36	Control	48	3.33333	0.1696485	19.65	<.0001*	2.74889	3.91777
Control	36	Papain	30	0.91667	0.1616429	5.67	<.0001*	0.35981	1.47353
Control	36	Papain	36	2.16667	0.1616429	13.40	<.0001*	1.60981	2.72353
Control	36	Papain	42	3.25000	0.1616429	20.11	<.0001*	2.69314	3.80686
Control	36	Papain	48	4.25000	0.1616429	26.29	<.0001*	3.69314	4.80686
Control	36	Vinegar	30	0.00000	0.1616429	0.00	1.0000	-0.55686	0.55686
Control	36	Vinegar	36	1.08333	0.1616429	6.70	<.0001*	0.52647	1.64019
Control	36	Vinegar	42	3.25000	0.1616429	20.11	<.0001*	2.69314	3.80686
Control	36	Vinegar	48	3.41667	0.1616429	21.14	<.0001*	2.85981	3.97353
Control	42	Control	48	0.16667	0.1696485	0.98	0.9974	-0.41777	0.75111
Control	42	Papain	30	-2.25000	0.1616429	-13.92	<.0001*	-2.80686	-1.69314
Control	42	Papain	36	-1.00000	0.1616429	-6.19	<.0001*	-1.55686	-0.44314
Control	42	Papain	42	0.08333	0.1616429	0.52	1.0000	-0.47353	0.64019
Control	42	Papain	48	1.08333	0.1616429	6.70	<.0001*	0.52647	1.64019
Control	42	Vinegar	30	-3.16667	0.1616429	-19.59	<.0001*	-3.72353	-2.60981
Control	42	Vinegar	36	-2.08333	0.1616429	-12.89	<.0001*	-2.64019	-1.52647
Control	42	Vinegar	42	0.08333	0.1616429	0.52	1.0000	-0.47353	0.64019
Control	42	Vinegar	48	0.25000	0.1616429	1.55	0.9187	-0.30686	0.80686
Control	48	Papain	30	-2.41667	0.1616429	-14.95	<.0001*	-2.97353	-1.85981
Control	48	Papain	36	-1.16667	0.1616429	-7.22	<.0001*	-1.72353	-0.60981
Control	48	Papain	42	-0.08333	0.1616429	-0.52	1.0000	-0.64019	0.47353
Control	48	Papain	48	0.91667	0.1616429	5.67	<.0001*	0.35981	1.47353
Control	48	Vinegar	30	-3.33333	0.1616429	-20.62	<.0001*	-3.89019	-2.77647
Control	48	Vinegar	36	-2.25000	0.1616429	-13.92	<.0001*	-2.80686	-1.69314
Control	48	Vinegar	42	-0.08333	0.1616429	-0.52	1.0000	-0.64019	0.47353
Control	48	Vinegar	48	0.08333	0.1616429	0.52	1.0000	-0.47353	0.64019

JMP PRO Spatial Example: Uniformity Trial

Consider the Uniformity Trial.jmp sample data table. An agronomic uniformity trial was conducted on an 8x8 grid of plots. In a uniformity trial, a test crop is grown on a field with no experimental treatments applied. The response variable, often yield, is measured. The idea is to characterize variability in the field as background for planning a designed experiment to be conducted on that field. (See Littell et al. 2006, p. 447.)

Your objective is to use the information from these data to design a yield trial with 16 treatments. Specifically, you want to decide whether to conduct future experiments on the field:

- a complete block design with 4 blocks (denoted Quarter in the data)
- an incomplete block design with 16 blocks (denoted Subquarter in the data)
- a completely randomized design with spatially correlated errors

With this objective, spatial data can be treated as repeated measures with two or more dimensions as repeated effects. So, you can compare and choose an appropriate model using the values in the Fit Statistics report. You start by determining if there is significant spatial variability, then you determine whether there is a nugget effect.

Once you have established whether there is a nugget effect, you determine the best fitting spatial covariance structure. Finally, you fit the blocking models and compare these to the best spatial structure. In this example, both AICc and BIC are used to select a best model. "Spatial Correlation Structure" provides more information about nugget effects and other spatial terminology.

Tip: This section walks you through many aspects of fitting spatial data (from fitting the model to deciding on the best covariance structure). Leave the Fit Model launch window open as you work through each example.

JMP PRO Fit a Spatial Structure Model

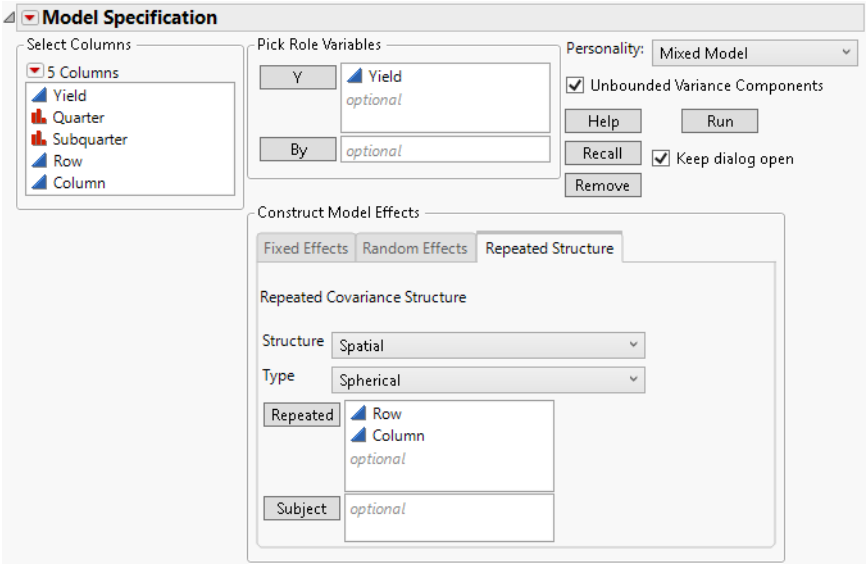
To determine whether there is significant spatial variability, you can fit a model that accounts for spatial variability. Then you can compare the likelihood for this spatial model to the likelihood for a model that does not account for spatial variability. You can do this because the independent errors model is nested within the spatial model family: The independent errors model is a spatial model with spatial correlation, ρ, equal to 0. This means that you can perform a formal likelihood ratio test of the two models.

First, fit the model that accounts for spatial structure.

1. Select **Help > Sample Data Library** and open Uniformity Trial.jmp.
2. Select **Analyze > Fit Model**.

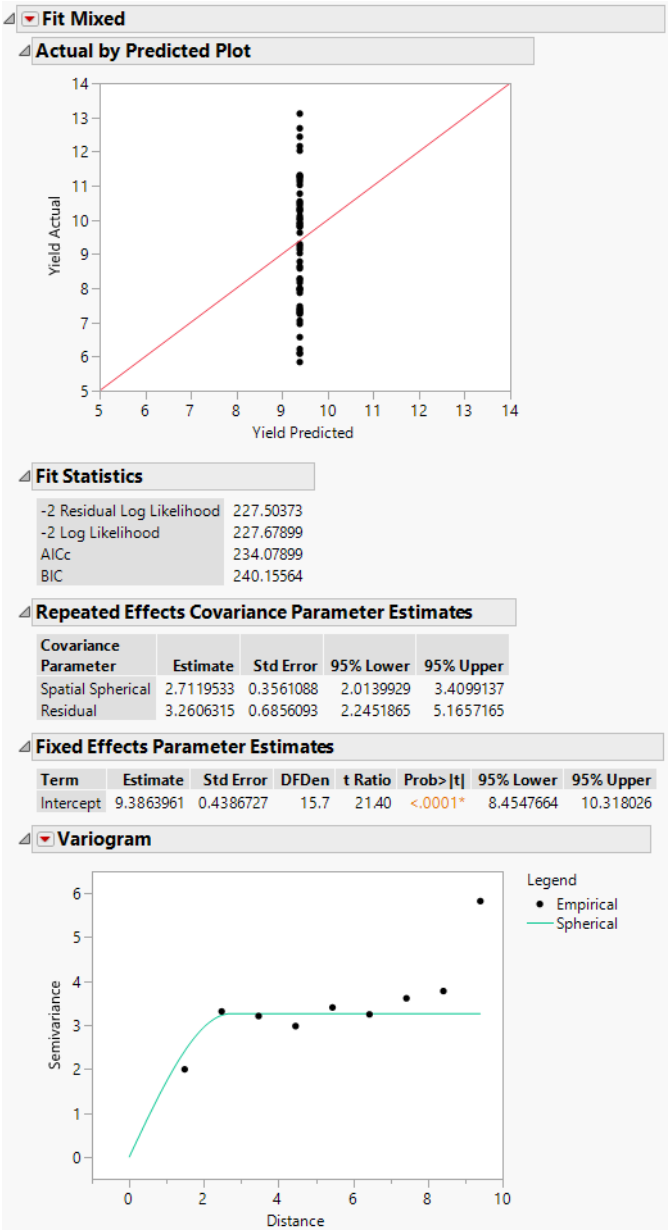
3. Select **Keep dialog open** so that you can return to the launch window in the next example.
4. Select Yield and click **Y**.
5. Select **Mixed Model** from the Personality list.
6. Select the **Repeated Structure** tab.
7. Choose **Spatial** from the list next to Structure.
8. Choose **Spherical** from the list next to Type.
9. Select Row and Column and click **Repeated**.

Figure 8.32 Completed Fit Model Launch Window Showing Repeated Structure Tab



10. Click **Run**.

Figure 8.33 Fit Mixed Report - Spatial Spherical



In the Fit Mixed report, the Actual by Predicted Plot shows that the predicted yield is a single value. This is because only spatial covariance was fit. The Fit Statistics report shows that -2 Log Likelihood is 227.68, and the AICc is 234.08.

Because an isotropic spatial structure was fit, a Variogram plot is shown. Because the trials are laid out in an 8 by 8 grid, there are more pairs of points at small distances than at very large distances. See Figure 8.37 for the layout. The Variogram shows that a spherical spatial structure is an excellent fit for distances up to about 8.4. The distance class for the final distance consists of only the two diagonal pairs of points.

The Repeated Effects Covariance Parameter Estimates report gives estimates of the range (Spatial Spherical = 2.71) and the sill (Residual = 3.26). See "Variogram".

JMP PRO Fit the Independent Errors Model

Next, fit the independent errors model.

1. Return to the Fit Model Launch Window.
2. Select **Repeated Structure** tab.
3. Select **Residual** from the Structure list.
4. Remove Row and Column from the Repeated effects list.

   Otherwise, a warning appears: "Repeated columns and subject columns are ignored when the Residual covariance structure is selected."
5. Click **Run**.

   The fit statistics for the independent errors model are: -2 Log Likelihood = 254.22, and AICc = 258.41. Each of these exceeds the corresponding value for the spatial correlation model, where -2 Log Likelihood is 227.68 and the AICc is 234.08. Because smaller values of these statistics indicate a better fit, the spatial model might provide a better fit.

JMP PRO Conduct a Likelihood Ratio Test (Optional)

A formal likelihood ratio test shows whether the spatial correlation model explains significant variation. One model must be nested in another model to create valid likelihood ratio tests.

Typically, spatial models are compared using AICc or BIC rather than through formal likelihood ratio testing. Evaluating the AICc or BIC is faster, and many spatial models are not nested.

You can conduct a likelihood ratio test in this example, because the independent errors model is nested within the spatial model family. The independent errors model is a spatial model with spatial correlation, ρ, equal to 0. This means that you can perform a formal likelihood ratio test of the two models.

In this example, the likelihood ratio test statistic is 254.22-227.68=26.54. Comparing this to a Chi-square distribution on one degree of freedom, the null hypothesis of no spatial correlation is rejected with a p-value < 0.0001. You can conclude that these data contain significant spatial variability.

JMP PRO Select the Type of Spatial Covariance

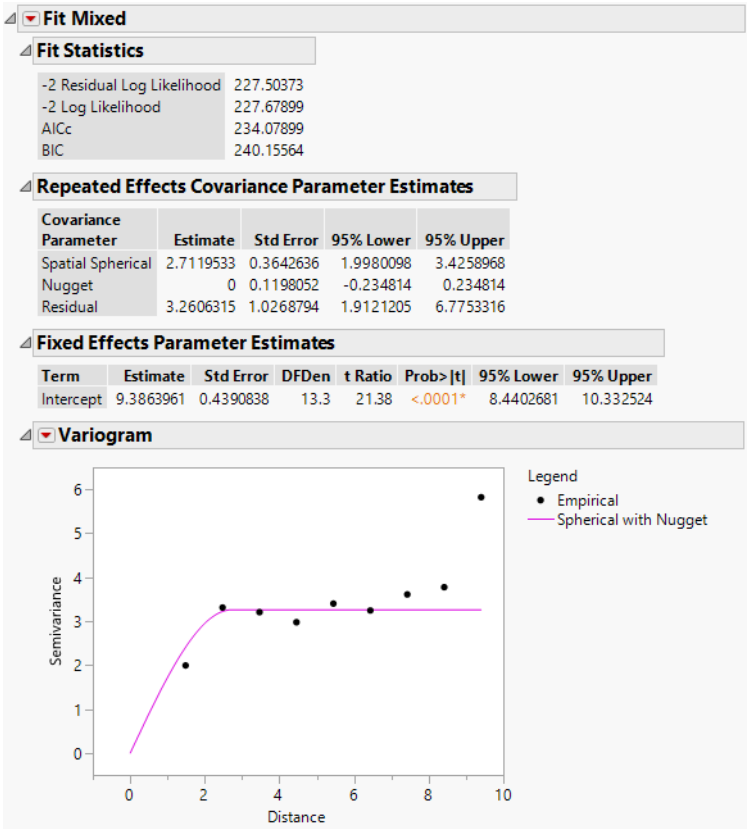
Next, you determine which spatial covariance structure best fits the data:

- with or without a nugget effect (variation over relatively small distances)
- isotropic (spatial correlation is equal in all directions) or anisotropic (spatial correlation differs in the two directions)
- type of structure, spherical, Gaussian, exponential, or power.

1. Return to the Fit Model launch window.
2. Select the **Repeated Structure** tab.
3. Select Row and Column and click **Repeated**.
4. Select **Spatial with Nugget** from the Structure list.
5. Select **Spherical** from the Type list.
6. Click **Run**.

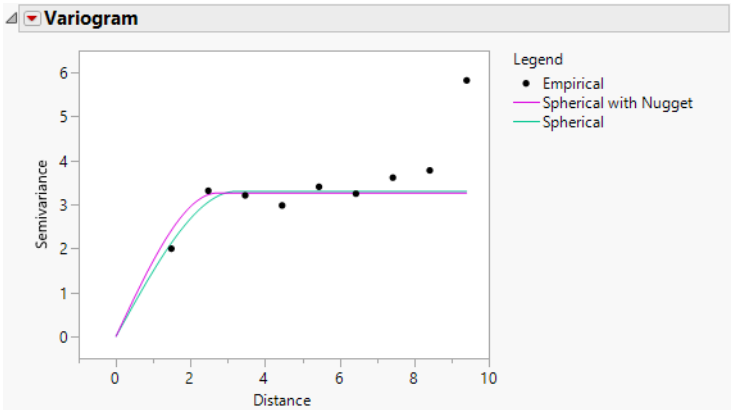
   The Fit Mixed report is shown in Figure 8.34. Notice that the log-likelihoods are essentially equal to those of the spherical with no nugget model, and the AICc is slightly higher (236.36 compared to 234.08). The Repeated Effects Covariance Parameter Estimates report shows that the Nugget covariance parameter has an estimate of zero. There does not appear to be any evidence for a nugget effect.

Figure 8.34 Fit Mixed Report - Spatial Spherical with Nugget



7. Click the Variogram red triangle and select **Spatial > Spherical**.

Figure 8.35 Fit Mixed Report - Variogram

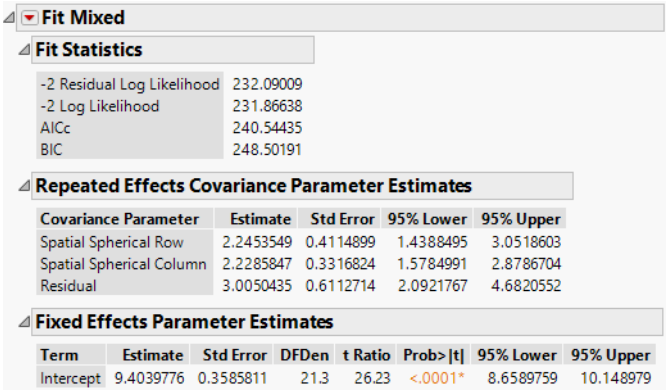


The two variograms are virtually identical. This also suggests that there is no evidence of a nugget effect.

8. Return to the Fit Model launch window.
9. Select **Repeated Structure** tab.
10. To test anisotropicity, select **Spatial Anisotropic** from the Structure list.
11. Select **Spherical** from the Type list.
12. Click **Run**.

    The Fit Mixed report is shown in Figure 8.36. The fit statistics indicate not as good a fit as the isotropic (spatial structure) spherical model (AICc 240.54 compared to 234.08). The Repeated Effects Covariance Parameter Estimates report shows that the estimates for the Row (Spatial Spherical Row) and Column (Spatial Spherical Column) covariances are very close. There is no evidence to suggest that spatial correlations within rows and columns of the grid differ.

Figure 8.36 Fit Mixed Report - Spatial Anisotropic Spherical



Fit Mixed

Fit Statistics

-2 Residual Log Likelihood	232.09009
-2 Log Likelihood	231.86638
AICc	240.54435
BIC	248.50191

Repeated Effects Covariance Parameter Estimates

Covariance Parameter	Estimate	Std Error	95% Lower	95% Upper
Spatial Spherical Row	2.2453549	0.4114899	1.4388495	3.0518603
Spatial Spherical Column	2.2285847	0.3316824	1.5784991	2.8786704
Residual	3.0050435	0.6112714	2.0921767	4.6820552

Fixed Effects Parameter Estimates

Term	Estimate	Std Error	DFDen	t Ratio	Prob>|t|	95% Lower	95% Upper
Intercept	9.4039776	0.3585811	21.3	26.23	<.0001*	8.6589759	10.148979

JMP PRO Determine the Type of the Spatial Structure

An isotropic spatial structure with no nugget is appropriate. To determine the type of the spatial structure, you can compare the available types.

1. Return to the Fit Model launch window.
2. Select the **Repeated Structure** tab.
3. Select Row and Column and click **Repeated**.
4. Select **Spatial** from the Structure list.
5. Select **Power** from the Type list.
6. Click **Run**.

   Note the AICc and BIC values for the model fit.

7. Return to the Fit Model launch window.
8. Select **Exponential** from the Type list.
9. Click **Run**.

   Note the AICc and BIC values for the model fit.
10. Return to the Fit Model launch window.
11. Select **Gaussian** from the Type list.
12. Click **Run**.

    Note the AICc and BIC values for the model fit.

The observed AICc values for these types and the other fits that you performed are summarized in Table 8.2.

Table 8.2 Fit Statistics for Spatial Models Fit

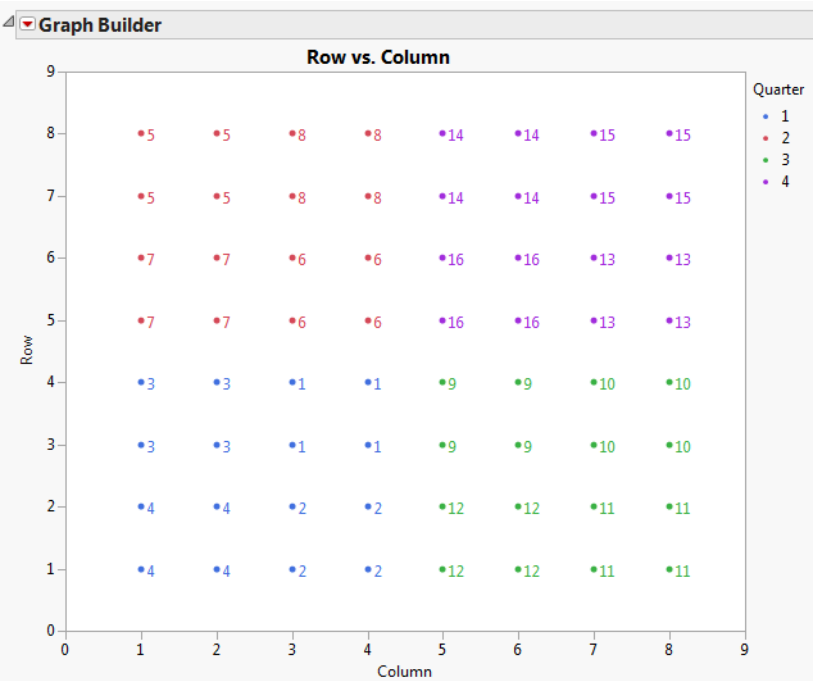
Structure	Type	AICc	BIC
Spatial	Spherical	234.08	240.16
Residual		258.41	262.53
Spatial with Nugget	Spherical	236.36	244.31
Spatial Anisotropic	Spherical	240.54	248.50
Spatial	Power	240.24	246.32
Spatial	Exponential	240.24	246.32
Spatial	Gaussian	238.37	244.44

The best fitting model (using the smallest AICc value) is the Spatial structure model with Spherical covariance structure. Now you will compare this model to the complete and incomplete block models to complete the objectives of the uniformity trial.

JMP PRO Compare the Model to Block Designs

1. Return to the Uniformity Trial data table.
2. In the Tables panel, run the Graph Builder script.

Figure 8.37 Graph Builder Plot of Proposed Complete and Incomplete Block Designs



This plot shows the proposed complete and incomplete block designs for the field. The color indicates the quarter fields that would serve as complete blocks. The numbered points represent the sub-quarter fields that would serve as incomplete blocks.

To fit the complete block model, follow these steps:

1. Return to the Fit Model launch window.
2. Select **Repeated Structure** tab.
3. Select **Residual** from the Structure list.
4. Remove Row and Column from the effect.

   Otherwise, a pop-up window appears, stating, "Repeated columns and subject columns are ignored when the Residual covariance structure is selected."
5. Select the **Random Effects** tab.
6. Select Quarter and click **Add**.
7. Click **Run**.

To fit the incomplete block model, follow these steps:

1. Return to the Fit Model launch window.
2. Select the **Random Effects** tab.

3. Select Quarter and click **Remove**.
4. Select Subquarter and click **Add**.
5. Click **Run**.

The following list shows both AICc and BIC values for the competing models. The spherical covariance structure results in the best model fit. This indicates that, for future studies using this field, a completely randomized design with spatially correlated errors is preferred.

- Spherical model (see "Determine the Type of the Spatial Structure")
  - AICc: 234.08
  - BICc: 240.16
- Complete block (RCBD) model
  - AICc: 259.90
  - BICc: 265.97
- Incomplete block model
  - AICc: 248.77
  - BICc: 254.85

JMP PRO Correlated Response Example

In this example, the effect of two layouts dealing with wafer production is studied for a characteristic of interest. Each of 50 wafers is partitioned into four quadrants and the characteristic is measured on each of these quadrants. Data of this type are usually presented in a format where each row contains all of the repeated measurements for one of the units of interest. Data of this type are often analyzed using separate models for each response. However, when repeated measurements are taken on a single unit, it is likely that there is within-unit correlation. Failure to account for this correlation can result in poor decisions and predictions. You can use the Mixed Model personality to account for and model the possible correlation.

For Mixed Model analysis of repeated measures data, each repeated measurement needs to be in its own row. If your data are in the typical format where all repeated measurements are in the same row, you can construct an appropriate data table for Mixed Model analysis by using Tables > Stack. See *Using JMP*.

In this example, you first fit univariate models using the usual data table format. Then you use the Mixed Model personality to fit models for the four responses while simultaneously accounting for possible correlation among the responses.

JMP PRO Fit Univariate Models

Use Standard Least Squares to fit a univariate model for each of the four quadrants.

1. Select **Help > Sample Data Library** and open Wafer Quadrants.jmp.

   This data table is structured for Mixed Model analysis, with one row for each Y measurement on each Quadrant. To conduct univariate analyses, you split the table using a saved script.

2. In the Wafer Quadrants.jmp data table, click the green triangle next to the **Split Y by Quadrant** script.

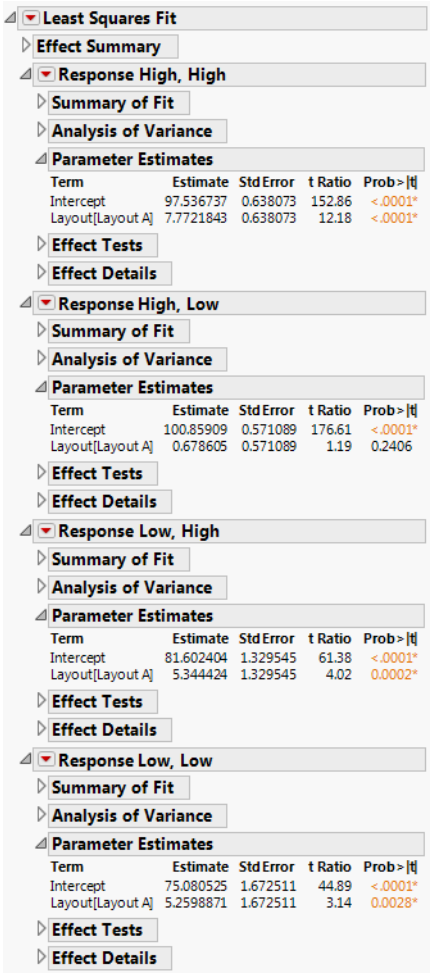
   The new data table is in the format often used to record repeated measures data. Each value of Wafer ID defines a row and the four measurements for that wafer are given in the single row.

3. Select **Analyze > Fit Model**.

   Since there is a Model script in the data table, the Model Specification window is already filled in. Note that the columns High, High through Low, Low are entered as Y and that Layout is the single model effect.

4. Click **Run**.

Figure 8.38 Four Univariate Models



The report indicates that Layout has a statistically significant effect for all quadrants except the High, Low quadrant.

JMP PRO Perform Mixed Model Analysis

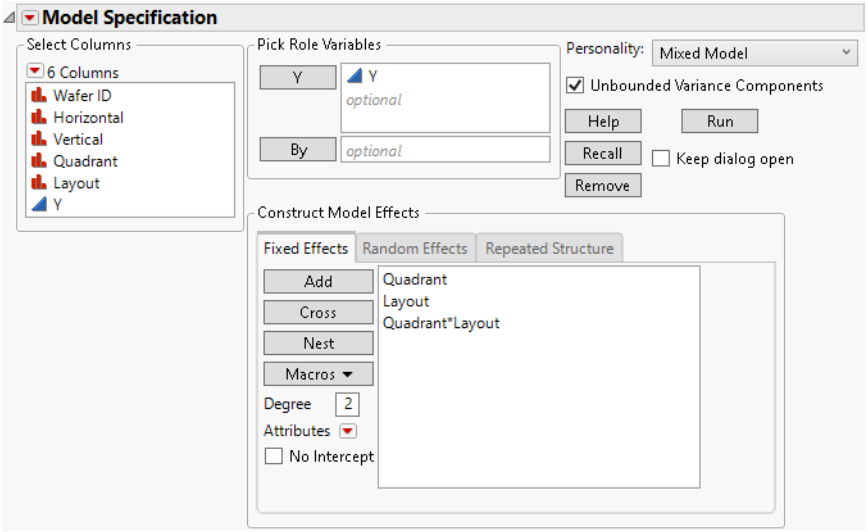
Using the Mixed Model analysis, you can obtain more information.

1. Return to Wafer Quadrants.jmp. Or, if you have closed it, select **Help > Sample Data Library** and open Wafer Quadrants.jmp.
2. Select **Analyze > Fit Model**.
3. Select Y and click **Y**.
4. Select **Mixed Model** from the Personality list.

5. Select Quadrant and Layout from the **Select Columns** list and click **Macros > Full Factorial**.

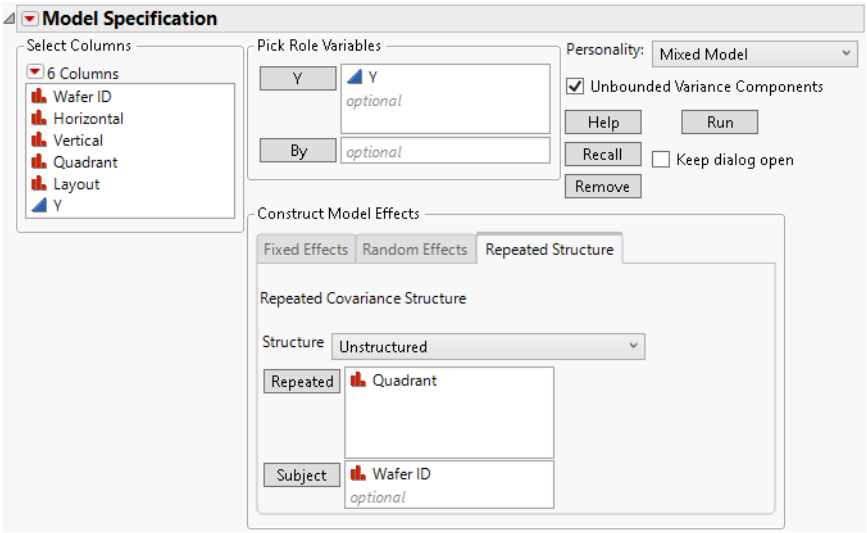
   This model specification enables you to explore the effect of Layout on the repeated measurements as well as the possible interaction of Layout with Quadrant.

Figure 8.39 Fit Model Launch Window Showing Completed Fixed Effects Tab



6. Select the **Repeated Structure** tab.
7. Select **Unstructured** from the Structure list.
8. Select Quadrant and click **Repeated**.
9. Select Wafer ID and click **Subject**.

Figure 8.40 Fit Model Launch Window Showing Repeated Structure Tab



10. Click **Run**.

Figure 8.41 Fit Mixed Report with Fixed Effects Parameter Estimates Report Closed

Fit Mixed

Repeated Effects Covariance Parameter Estimates

Repeated Effect: Quadrant
Subject: Wafer ID

Covariance Parameter	Estimate	Std Error	95% Lower	95% Upper
Var(High, High)	19.835702	4.0489457	11.899914	27.77149
Cov(High, Low,High, High)	0.0111321	2.5624804	-5.011237	5.0335013
Var(High, Low)	15.889659	3.2434631	9.5325884	22.24673
Cov(Low, High,High, High)	15.893138	6.3915311	3.3659676	28.420309
Cov(Low, High,High, Low)	1.0055411	5.3413847	-9.463381	11.474463
Var(Low, High)	86.121899	17.579559	51.666597	120.5772
Cov(Low, Low,High, High)	14.723606	7.7996739	-0.563474	30.010686
Cov(Low, Low,High, Low)	8.0424897	6.8163252	-5.317262	21.402242
Cov(Low, Low,Low, High)	2.1512894	15.640276	-28.50309	32.805667
Var(Low, Low)	136.28412	27.81888	81.76012	190.80813

Fixed Effects Tests

Source	Nparm	DFNum	DFDen	F Ratio	Prob > F
Quadrant	3	3	46.0	149.21352	<.0001*
Layout	1	1	48.0	51.757797	<.0001*
Quadrant*Layout	3	3	46.0	22.379307	<.0001*

The Repeated Effects Covariance Parameter Estimates report gives the estimated variances and covariances for the four responses. Note that the confidence interval for the covariance of Low, High with High, High does not include zero. This suggests that there is a positive covariance between measurements in these two quadrants. This is information that the Mixed Model analysis uses and that is not available when the responses are modeled independently.

The Fixed Effects Tests report indicates that there is a significant Layout by Characteristic interaction.

JMP PRO Explore the Layout by Characteristic Interaction with the Profiler

1. Click the Fit Mixed red triangle and select **Marginal Model Inference > Profiler**.
2. In the Profiler plot, compare the predicted values for Y across the quadrants by first setting the vertical red dotted line at Layout A and then at Layout B.

Figure 8.42 Profile for Quadrant for Layout A

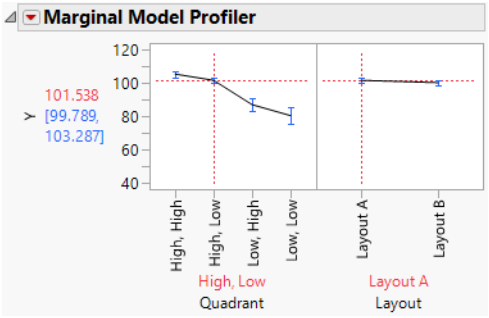
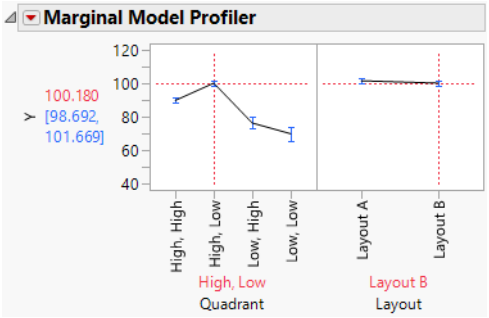


Figure 8.43 Profile for Quadrant for Layout B



The differences in the profiles at each setting of Layout give you insight into the significant interaction. It appears that the interaction is partially driven by the differences for the High, High quadrant.

JMP PRO Plot of Y by Layout and by Quadrant

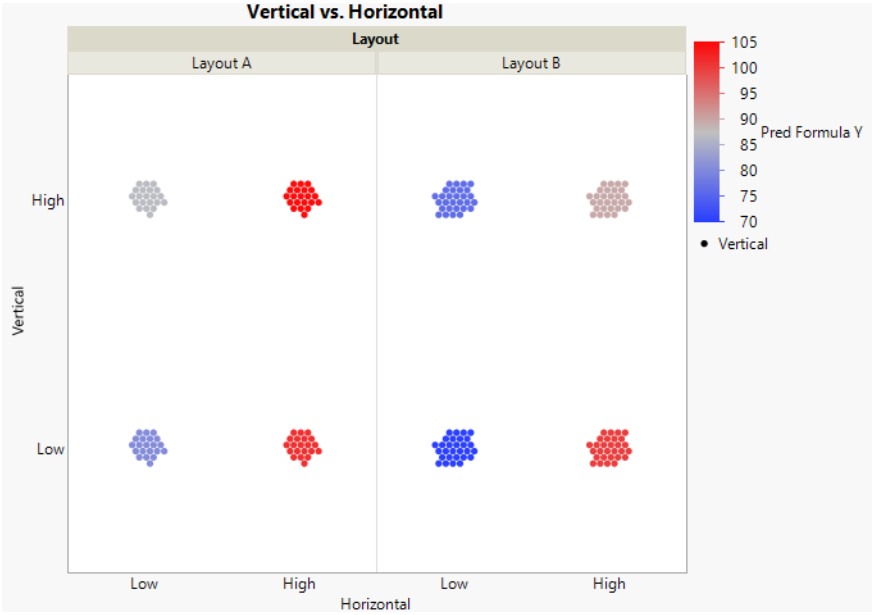
Use Graph Builder to explore the nature of the interaction.

1. Click the Fit Mixed red triangle and select **Save Columns > Prediction Formula**.

   The prediction formula is saved to the data table in the column Pred Formula Y.

2. Select **Graph > Graph Builder**.
3. Drag Horizontal to the **X** zone.
4. Drag Vertical to the **Y** zone.
5. Drag Pred Formula Y to the **Color** zone.
6. Drag Layout to the **Wrap** zone.
7. Click **Done** to hide the control panel.

Figure 8.44 Completed Graph Builder Plot



The plot shows the predicted differences for the eight Layout and Quadrant combinations using a color intensity scale. The predicted values for the High, Low quadrant are in the lower right. The color gradient shows relatively little difference for these predicted values. Other differences are clearly indicated.

JMP PRO Statistical Details for the Mixed Models Personality

- "Convergence Score Test"
- "Random Coefficient Model"
- "Repeated Measures"
- "Repeated Covariance Structures"
- "Spatial and Temporal Variability"
- "The Kackar-Harville Correction"

JMP PRO Convergence Score Test

The convergence failure warning shows the score test for the following hypothesis: that the unknown maximum likelihood estimate (MLE) is consistent with the parameter given in the final iteration of the model-fitting algorithm. This hypothesis test is possible because the relative gradient criterion is algebraically equivalent to the score test statistic. Remarkably, the score test does not require knowledge of the true MLE.

JMP PRO Score Test

Consider first the case of a single parameter, θ. Let l be the log-likelihood function for θ and let $\mathbf{x}$ be the data. The score is the derivative of the log-likelihood function with respect to θ:

$$U(\theta) = \frac{\partial l(\theta|\mathbf{x})}{\partial \theta}$$

The observed information is:

$$I(\theta) = -\frac{\partial^2}{\partial \theta^2} l(\theta|\mathbf{x})$$

The statistic for the score test of H_0: $\theta = \theta_0$ is:

$$S(\theta_0) = \frac{U(\theta_0)^2}{I(\theta_0)}$$

This statistic has an asymptotic Chi-square distribution with 1 degree of freedom under the null hypothesis.

The score test can be generalized to multiple parameters. Consider the vector of parameters θ. Then the test statistic for the score test of H_0: $\theta = \theta_0$ is:

$$S(\theta_0) = \mathbf{U}'(\theta_0)\mathbf{I}^{-1}(\theta_0)\mathbf{U}(\theta_0)$$

where

$$\mathbf{U}(\theta) = \frac{\partial l(\theta|\mathbf{x})}{\partial \theta}$$

and

$$\mathbf{I}(\theta) = -\frac{\partial^2 l(\theta|\mathbf{x})}{\partial \theta(\partial \theta')}$$

and $\mathbf{U}'$ denotes the transpose of the matrix $\mathbf{U}$.

The test statistic is asymptotically Chi-square distribution with k degrees of freedom. Here k is the number of unbounded parameters.

JMP PRO Relative Gradient

The convergence criterion for the Mixed Model fitting procedure is based on the relative gradient $g'\mathbf{H}^{-1}g$. Here, $g(\theta) = \mathbf{U}(\theta)$ is the gradient of the log-likelihood function and $\mathbf{H}(\theta) = -\mathbf{I}(\theta)$ is its Hessian.

Let θ_0 be the value of θ where the algorithm terminates. Note that the relative gradient evaluated at θ_0 is the score test statistic. A p-value is calculated using a Chi-square distribution with k degrees of freedom. This p-value gives an indication of whether the value of the unknown MLE is consistent with θ_0. The number of unbounded parameters listed in the Random Effects Covariance Parameter Estimates report equals k.

JMP PRO Random Coefficient Model

The standard random coefficient model specifies a random intercept and slope for each subject. Let y_{ij} denote the measurement of the j^{th} observation on the i^{th} subject. Then the random coefficient model can be specified as follows:

$$y_{ij} = a_i + b_i x_{ij} + e_{ij}$$

where

$$i = 1, 2, \ldots, t$$

$$j = 1, 2, \ldots, n_i$$

$$\begin{bmatrix} a_i \\ b_i \end{bmatrix} \sim iid\ N\left(\begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \mathbf{G} \right)$$

$$\mathbf{G} = \begin{bmatrix} \sigma_a^2 & \sigma_{ab} \\ \sigma_{ab} & \sigma_b^2 \end{bmatrix}$$

and

$$e_{ij} \sim iid\ N(0, \sigma^2)$$

You can reformulate the model to reflect the fixed and random components that are estimated:

$$y_{ij} = (\alpha + a_i^*) + (\beta + b_i^*)x_{ij} + e_{ij}$$

where

$$a_i^* = \alpha_i - \alpha$$

$$b_i^* = \beta_i - \beta$$

and

$$\begin{bmatrix} a_i^* \\ b_i^* \end{bmatrix} \sim iid\ N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \mathbf{G} \right)$$

with **G** and e_{ij} defined as above.

JMP PRO Repeated Measures

The form of the repeated measures model is $y_{ijk} = \alpha_{ij} + s_{ik} + e_{ijk}$, where

α_{ij} can be written as a treatment and time factorial

s_{ik} is the random effect of the k^{th} subject assigned to the i^{th} treatment

$j = 1,\ldots,m$ denotes the repeated measurements over time.

Assume that the s_{ik} are independent and identically distributed N(0, σ_s^2) variables. Denote the number of treatment factors by t and the number of subjects by s. Then the distribution of e_{ijk} is N(0, Σ), where

$$\Sigma = \mathbf{I}_{ts} \otimes Var(\mathbf{y}_{ik} | s_{ik})$$

and

$$\mathbf{y}'_{ik} | s_{ik} = \left(\left[y_{i1k} \; y_{i2k} \; \cdots \; y_{imk} \right] \middle| s_{ik} \right)$$

Denote the block diagonal component of the covariance matrix Σ corresponding to the ik^{th} subject within treatment by Σ_{ik}. In other words, $\Sigma_{ik} = Var(\mathbf{y}_{ik} | s_{ik})$. Because observations over time within a subject are not typically independent, it is necessary to estimate the variance of $y_{ijk} | s_{ik}$. Failure to account for the correlation leads to distorted inference.

See "Repeated Covariance Structures" and "Spatial and Temporal Variability" for more information about the covariance structures available for Σ_{ik}.

JMP PRO Repeated Covariance Structures

This section gives the parameterizations for the following covariance structures:

- "Unequal Variances Covariance Structure"
- "Unstructured Covariance Structure"
- "Compound Symmetry Covariance Structure"
- "AR(1) Covariance Structure"
- "Toeplitz Covariance Structure"
- "Antedependent Covariance Structure"

JMP PRO Unequal Variances Covariance Structure

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \cdots & 0 \\ & \sigma_2^2 & 0 & \cdots & 0 \\ & & \sigma_3^2 & \cdots & 0 \\ & & & \ddots & \vdots \\ & & & & \sigma_m^2 \end{bmatrix}$$

Here, the variance among observations taken at time j is:

$$\sigma_j^2 = Var(y_{ijk} | s_{ik})$$

The variances are allowed to differ across the levels of the repeated column. The covariances between observations at different levels are zero.

JMP PRO Unstructured Covariance Structure

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} & \cdots & \sigma_{1[m-1]} & \sigma_{1m} \\ & \sigma_2^2 & \sigma_{23} & \cdots & \sigma_{2[m-1]} & \sigma_{2m} \\ & & \sigma_3^2 & \cdots & \sigma_{3[m-1]} & \sigma_{3m} \\ & & & \ddots & \vdots & \vdots \\ & & & & \sigma_{m-1}^2 & \sigma_{[m-1]m} \\ & & & & & \sigma_m^2 \end{bmatrix}$$

Here, the variance among observations taken at time j is:

$$\sigma_j^2 = Var(y_{ijk} | s_{ik})$$

The covariance between observations taken at times j and j' is:

$$\sigma_{jj'} = Cov(y_{ijk}, y_{ij'k} | s_{ik})$$

The variances are allowed to differ across the levels of the repeated column. The covariances between observations at different levels is unique.

JMP PRO Compound Symmetry Covariance Structure

In JMP, a compound symmetry covariance structure is implemented using a mixed model with independent errors approach. Random effects are classified into two categories: G-side or R-side. See Searle et al. (1992).

The G-side random effects are associated with the design matrix for random effects. The R-side random effects are associated with residual error. Within-subject variance is part of the design structure and is modeled on the G-side. Between-subject variance falls into the residual structure and is modeled R-side. The variances in the independent structure are modeled in the following manner:

- The random effects G-side variance is modeled by $s_{ik} \sim$ iid N(0, σ_s^2).

- The R-side variance is modeled by $e_{ijk} \sim$ iid N(0, σ^2).

Then the covariance matrix is defined as follows:

$$\Sigma_{ik} = \sigma_s^2\mathbf{J} + \sigma^2\mathbf{I} = \begin{bmatrix} \sigma_s^2+\sigma^2 & \sigma_s^2 & \cdots & \sigma_s^2 \\ & \sigma_s^2+\sigma^2 & \cdots & \sigma_s^2 \\ & & \ddots & \vdots \\ & & & \sigma_s^2+\sigma^2 \end{bmatrix}$$

where **J** is a matrix consisting of *1*s and **I** is an identity matrix.

Alternatively, all variance could be modeled on the R-side. Under the Gaussian assumption, this compound-symmetry covariance structure is equivalent to the independence model (Type=CS in SAS). This structure is available in JMP by using the Compound Symmetry structure in the repeated structure tab. Here, the correlation between pairs of repeated observations is the same regardless of the time difference between the observations. Thus, the correlation matrix can be specified as follows:

$$C = \begin{bmatrix} 1 & \rho & \cdots & \rho \\ & 1 & \cdots & \rho \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix}$$

Using the Compound Symmetry structure in JMP also assumes a common variance, σ_e^2, among observations taken at any time point. The covariance structure is then $\Sigma = \sigma_e^2 C$ where

$$\sigma_e^2 = \sigma_s^2 + \sigma^2$$

and

$$\rho = \frac{\sigma_s^2}{\sigma_s^2 + \sigma^2}.$$

Here, ρ is the intra-class correlation coefficient and σ_e^2 is the residual variance. Another option is to use the Compound Symmetry Unequal Variances structure in JMP, which allows the variance to vary across time points. This leads to the following covariance matrix:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 & \rho\sigma_1\sigma_3 & \cdots & \rho\sigma_1\sigma_{m-1} & \rho\sigma_1\sigma_m \\ & \sigma_2^2 & \rho\sigma_2\sigma_3 & \cdots & \rho\sigma_2\sigma_{m-1} & \rho\sigma_2\sigma_m \\ & & \sigma_3^2 & \cdots & \rho\sigma_3\sigma_{m-1} & \rho\sigma_3\sigma_m \\ & & & \ddots & \vdots & \vdots \\ & & & & \sigma_{m-1}^2 & \rho\sigma_{m-1}\sigma_m \\ & & & & & \sigma_m^2 \end{bmatrix}$$

JMP PRO AR(1) Covariance Structure

$$\Sigma = \begin{bmatrix} \sigma^2 & \rho^{|t_1-t_2|}\sigma^2 & \rho^{|t_1-t_3|}\sigma^2 & \cdots & \rho^{|t_1-t_{m-1}|}\sigma^2 & \rho^{|t_1-t_m|}\sigma^2 \\ & \sigma^2 & \rho^{|t_2-t_3|}\sigma^2 & \cdots & \rho^{|t_2-t_{m-1}|}\sigma^2 & \rho^{|t_2-t_m|}\sigma^2 \\ & & \sigma^2 & \cdots & \rho^{|t_3-t_{m-1}|}\sigma^2 & \rho^{|t_3-t_m|}\sigma^2 \\ & & & \ddots & \vdots & \vdots \\ & & & & \sigma^2 & \rho^{|t_{m-1}-t_m|}\sigma^2 \\ & & & & & \sigma^2 \end{bmatrix}$$

Here t_j is the time of observation j. In this structure, observations taken at any given time have the same variance, σ^2. The parameter ρ, where $-1 < \rho < 1$, is the correlation between two observations that are one unit of time apart. As the time difference between observations increases, their covariance decreases because ρ is raised to a higher power. In many applications, AR(1) provides an adequate model of the within subject correlation, providing more power without sacrificing Type I error control.

JMP PRO Toeplitz Covariance Structure

In the Toeplitz structure, observations that are separated by a fixed number of time units have the same correlation. In contrast to the AR(1) correlation structure, the Toeplitz correlations at a fixed time difference are arbitrary. Denote the correlation for observations d units apart by ρ_d. The correlation matrix is defined as follows:

$$C = \begin{bmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{m-1} & \rho_m \\ & 1 & \rho_1 & \cdots & \rho_{m-2} & \rho_{m-1} \\ & & 1 & \cdots & \rho_{m-3} & \rho_{m-2} \\ & & & \ddots & \vdots & \vdots \\ & & & & 1 & \rho_1 \\ & & & & & 1 \end{bmatrix}$$

Two options in JMP use this correlation matrix:

- The Toeplitz structure assumes a common variance, σ^2, for observations from any time point. The covariance structure is $\Sigma = \sigma^2 C$.
- Alternatively, the Toeplitz Unequal Variances structure allows the variance to vary across time points:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_1\sigma_1\sigma_2 & \rho_2\sigma_1\sigma_3 & \cdots & \rho_{m-1}\sigma_1\sigma_{m-1} & \rho_m\sigma_1\sigma_m \\ & \sigma_2^2 & \rho_1\sigma_2\sigma_3 & \cdots & \rho_{m-2}\sigma_2\sigma_{m-1} & \rho_{m-1}\sigma_2\sigma_m \\ & & \sigma_3^2 & \cdots & \rho_{m-3}\sigma_3\sigma_{m-1} & \rho_{m-2}\sigma_3\sigma_m \\ & & & \ddots & \vdots & \vdots \\ & & & & \sigma_{m-1}^2 & \rho_1\sigma_{m-1}\sigma_m \\ & & & & & \sigma_m^2 \end{bmatrix}$$

JMP PRO Antedependent Covariance Structure

The antedependence model is a general model that is flexible and allows the correlation structure to change over time. In this model, the correlation between two observations at adjacent time points j - 1 and j is unique and is denoted $\rho_{j[j-1]}$.

The correlation between pairs of observations at non-adjacent time points j and j' is the product of all the adjacent correlations in between.

$$Corr(y_{ijk}, y_{ij'k} | s_{ik}) = \prod_{k=j}^{j'-1} \rho_{k[k-1]}$$

For example, the correlation between the pair of observations at time points j=2 and j'=6 would be $\rho_{21}\rho_{32}\rho_{43}\rho_{54}$.

The correlation matrix is defined as follows:

$$C = \begin{bmatrix} 1 & \rho_{10} & \rho_{10}\rho_{21} & \cdots & \rho_{10}\cdots\rho_{[m-1][m-2]} & \rho_{10}\cdots\rho_{m[m-1]} \\ & 1 & \rho_{21} & \cdots & \rho_{21}\cdots\rho_{[m-1][m-2]} & \rho_{21}\cdots\rho_{m[m-1]} \\ & & 1 & \cdots & \rho_{32}\cdots\rho_{[m-1][m-2]} & \rho_{32}\cdots\rho_{m[m-1]} \\ & & & \ddots & \vdots & \vdots \\ & & & & 1 & \rho_{m[m-1]} \\ & & & & & 1 \end{bmatrix}$$

Two options in JMP use this correlation matrix:

- The Antedependent Equal Variance structure assumes equal variances across observation times while still allowing for the correlations to change. The variance among observations at any time is σ^2 and the covariance matrix is $\Sigma = \sigma^2 C$.
- The Antedependent structure allows the variance among observations at any given time to vary. Denote the variance among observations taken at time j is σ_j^2 . Then the covariance matrix is defined as follows:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{10}\sigma_1\sigma_2 & \rho_{10}\rho_{21}\sigma_1\sigma_3 & \cdots & \rho_{10}\cdots\rho_{[m-1][m-2]}\sigma_1\sigma_{m-1} & \rho_{10}\cdots\rho_{m[m-1]}\sigma_1\sigma_m \\ & \sigma_2^2 & \rho_{21}\sigma_2\sigma_3 & \cdots & \rho_{21}\cdots\rho_{[m-1][m-2]}\sigma_2\sigma_{m-1} & \rho_{21}\cdots\rho_{m[m-1]}\sigma_2\sigma_m \\ & & \sigma_3^2 & \cdots & \rho_{32}\cdots\rho_{[m-1][m-2]}\sigma_3\sigma_{m-1} & \rho_{32}\cdots\rho_{m[m-1]}\sigma_3\sigma_m \\ & & & \ddots & \vdots & \vdots \\ & & & & \sigma_{m-1}^2 & \rho_{m[m-1]}\sigma_{m-1}\sigma_m \\ & & & & & \sigma_m^2 \end{bmatrix}$$

JMP PRO Spatial and Temporal Variability

Consider the simple model $y_i = \mu + e_i$. The spatial or temporal structure is modeled through the error term, e_i. In general, the spatial correlation model can be defined as $Var(e_i) = \sigma_i^2$ and $Cov(e_i, e_j) = \sigma_{ij}$.

Let $\mathbf{s}_i$ denote the location of y_i, where $\mathbf{s}_i$ is specified by coordinates reflecting space or time. The spatial or temporal structure is typically restricted by assuming that the covariance is a function of the Euclidean distance, d_{ij}, between $\mathbf{s}_i$ and $\mathbf{s}_j$. The covariance can be written as $Cov(e_i, e_j) = \sigma^2[f(d_{ij})]$, where $f(d_{ij})$ represents the correlation between observations y_i and y_j.

In the case of two or more location coordinates, if $f(d_{ij})$ does not depend on direction, then the covariance structure is *isotropic*. If it does, then the structure is *anisotropic*.

JMP PRO Spatial Correlation Structure

The correlation structures for spatial models available in JMP are shown below. These are parametrized by ρ, which is positive unless it is otherwise constrained.

- Spherical

$$f(d_{ij}) = [1 - 1.5(d_{ij}/\rho) + 0.5(d_{ij}/\rho)^3] \times 1_{\{d_{ij} < \rho\}}$$

where $1_{\{d_{ij} < \rho\}} = \begin{cases} 1, \text{ if } d_{ij} < \rho \\ 0, \text{ if } d_{ij} \geq \rho \end{cases}$

- Exponential

$$f(d_{ij}) = \exp(-d_{ij}/\rho)$$

- Gaussian

$$f(d_{ij}) = \exp(-d_{ij}^2/\rho^2)$$

- Power

$$f(d_{ij}) = \rho^{d_{ij}}$$

For an anisotropic model, the correlation function contains a parameter, ρ_κ, for each direction.

JMP PRO Variogram

When the spatial process is second-order stationary, the structures listed in "Spatial Correlation Structure" define *variograms*. Borrowed from geostatistics, the variogram is the standard tool for describing and estimating spatial variability. It measures spatial variability as a function of the distance, d_{ij}, between observations using the semivariance.

Let $Z(\mathbf{s})$ denote the value of the response at a location $\mathbf{s}$. The *semivariance* between observations at $\mathbf{s}_i$ and $\mathbf{s}_j$ is defined as follows:

$$\gamma(\mathbf{s}_i, \mathbf{s}_j) = (Var(Z(\mathbf{s}_i) - Z(\mathbf{s}_j)))/2$$

If the response has a constant mean, then the expression can be simplified to the following:

$$\gamma(\mathbf{s}_i, \mathbf{s}_j) = E[(Z(\mathbf{s}_i) - Z(\mathbf{s}_j))^2]/2$$

If the process is isotropic, the semivariance depends only on the distance h between points and the function can be specified as follows:

$$\gamma(h) = E[(Z(\mathbf{s}_i) - Z((\mathbf{s}_i + h)))^2]/2$$

The following terms are associated with variograms:

Nugget Defined as the intercept. This represents a jump discontinuity at $h = 0$.

Sill Defined as the value of the semivariogram at the plateau reached for larger distances. It corresponds to the variance of an observation. In models with no nugget effect, the sill is σ^2. In models with a nugget effect, the sill is $\sigma^2 + c_1$, where c_1 represents the nugget. The *partial sill* is defined as σ^2.

Range Defined as the distance at which the semivariogram reaches the sill. At distances less than the range, observations are spatially correlated. For distances greater than or equal to the range, spatial correlation is effectively zero. In spherical models, ρ is the range. In exponential models, 3ρ is the practical range. In Gaussian models, $\rho\sqrt{3}$ is the practical range. The practical range is defined as the distance where covariance is reduced to 95% of the sill.

In Figure 8.34, the repeated effects covariance parameter estimates represent the various semivariogram features:

Spatial Spherical An estimate of the range, ρ.

Nugget A scaled estimate of c_1. The Residual times the Nugget is c_1.

Residual The partial sill or the sill in no nugget models.

JMP PRO Variogram Estimate

For a given isotropic spatial structure, the estimated variogram is obtained using a nonlinear least squares fit of the observed data to the appropriate function in "Spatial Correlation Structure".

JMP PRO Empirical Semivariance

To compute the *empirical semivariance*, the distances between all pairs of points for the variables selected for the variogram covariance are computed. The range of the distances is divided into 10 equal intervals. If the data do not allow for 10 intervals, then as many intervals as possible are constructed.

Distance classes consisting of pairs of points are constructed. The h^{th} distance class consists of all pairs of points whose distances fall in the h^{th} interval.

Consider the following notation:

n total number of pairs of points

C_h distance class consisting of points whose distance falls into the h^{th} largest interval

$Z(\mathbf{x})$ value of the response at $\mathbf{x}$, where $\mathbf{x}$ is a vector of temporal or spatial coordinates

$\gamma(h)$ semivariance for distance class C_h

The semivariance function, γ, is defined as follows:

$$\gamma(h) = \begin{cases} \frac{1}{2n}\left\{ \sum_{(x,y) \text{ in } C_h} [Z(\mathbf{x}) - Z(\mathbf{y})]^2 \right\} & \text{for } h > 0 \\ \hat{c}_1 & \text{for } h = 0 \end{cases}$$

Here $\hat{c}_1$ is an estimate of the nugget effect.

JMP PRO The Kackar-Harville Correction

The variance matrix of the fixed effects is always modified to include a Kackar-Harville correction. The variance matrix of the BLUPs, and the covariances between the BLUPs and the fixed effects, are not Kackar-Harville corrected. The rationale for this approach is that corrections for BLUPs can be computationally and memory intensive when the random effects have many levels. In SAS, the Kackar-Harville correction is done for both fixed effects and BLUPs only when the `DDFM=KENWARDROGER` is set.

For covariance structures that have nonzero second derivatives with respect to the covariance parameters, the Kenward-Roger covariance matrix adjustment includes a second-order term. This term can result in standard error shrinkage. Also, the resulting adjusted covariance matrix can then be indefinite and is not invariant under reparameterization. The first-order Kenward-Roger covariance matrix adjustment eliminates the second derivatives from the calculation. All spatial structures and the AR(1) structure are covariance structures that generally lead to nonzero second derivatives.

Because JMP implements the Kenward-Roger first-order adjustment

- Standard errors for linear combinations involving only fixed effects parameters match `PROC MIXED DDFM=KENWARDROGER(FIRSTORDER)`. This presumes that one has taken care to transform between the different parameterizations used by `PROC MIXED` and JMP.
- Standard errors for linear combinations involving only BLUP parameters match `PROC MIXED DDFM=SATTERTHWAITE`.
- Standard errors for linear combinations involving both fixed effects and BLUPS do not match `PROC MIXED` for any `DDFM` option if the data are unbalanced. However, these standard errors are between those obtained using the `DDFM=SATTERTHWAITE` and `DDFM=KENWARDROGER` options. If the data are balanced, JMP matches SAS regardless of the `DDFM` option, because the Kackar-Harville correction is null.

JMP PRO Degrees of Freedom

The degrees of freedom for tests involving only linear combinations of fixed effect parameters are calculated using the first-order Kenward-Roger correction. Therefore, the JMP results for these tests match `PROC MIXED` using the `DDFM=KENWARDROGER(FIRSTORDER)` option. If there are BLUPs in the linear combination, JMP uses a Satterthwaite approximation to get the degrees of freedom. The results then follow a pattern similar to what is described for standard errors in the preceding paragraph.

For more information about the Kackar-Harville correction and the Kenward-Roger DF approach, see Kenward and Roger (1997). The Satterthwaite method is described in detail in the MIXED Procedure chapter in SAS Institute Inc. (2020d, ch. 83).

Chapter 9

Multivariate Response Models

Fit Relationships Using MANOVA

Multivariate models fit several responses (Y variables) to a set of effects. The functions across the Y variables can be tested with appropriate response designs.

In addition to creating standard MANOVA (Multivariate Analysis of Variance) models, you can use the following techniques:

- Repeated measures analysis when repeated measurements are taken on each subject and you want to analyze effects both between subjects and within subjects across the measurements. This multivariate approach is especially important when the correlation structure across the measurements is arbitrary.
- Canonical correlation to find the linear combination of the X and Y variables that has the highest correlation.
- Discriminant analysis to find distance formulas between points and the multivariate means of various groups so that points can be classified into the groups that they are most likely to be in. A more complete implementation of discriminant analysis is in the **Discriminant** platform.

The multivariate fit begins with a rudimentary preliminary analysis that shows parameter estimates and least squares means. You can then specify a response design across the Y variables and multivariate tests are performed.

Contents

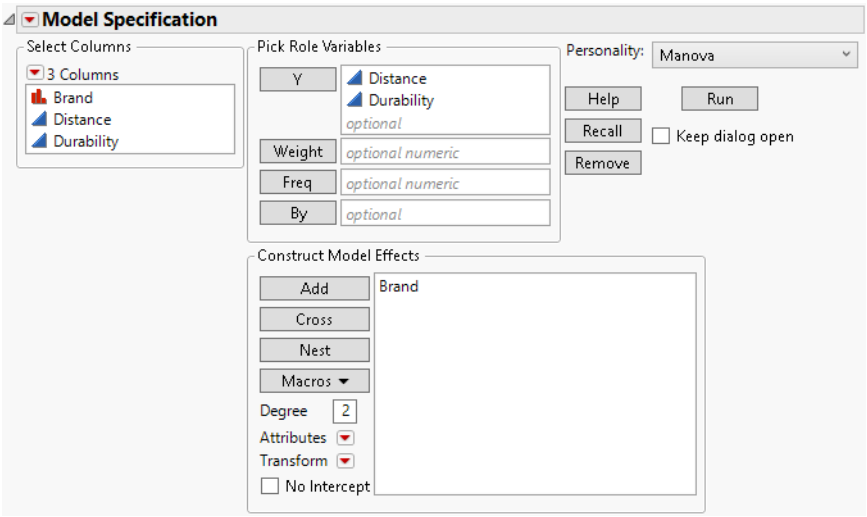
Example of a Multiple Response Model 439
Launch the Manova Personality 441
The Manova Report 441
The Manova Fit Options 442
Response Specification 443
Choose Response Options 443
Custom Test Option 444
Options for Model Effects 444
Multivariate Tests 448
The Extended Multivariate Report 449
Comparison of Multivariate Tests 450
Univariate Tests and the Test for Sphericity 451
Multivariate Model with Repeated Measures 452
Repeated Measures Example 453
Example of a Compound Multivariate Model 455
Discriminant Analysis 457
Example of the Save Discrim Option 458
Statistical Details for the Manova Personality 459
Multivariate Tests 459
Approximate F-Tests 460
Canonical Details 461

Example of a Multiple Response Model

This example uses the Golf Balls.jmp sample data table (McClave and Dietrich 1988). The data are a comparison of distances traveled and a measure of durability for three brands of golf balls. A robotic golfer hit a random sample of ten balls for each brand in a random sequence. The hypothesis to test is that distance and durability are the same for the three golf ball brands.

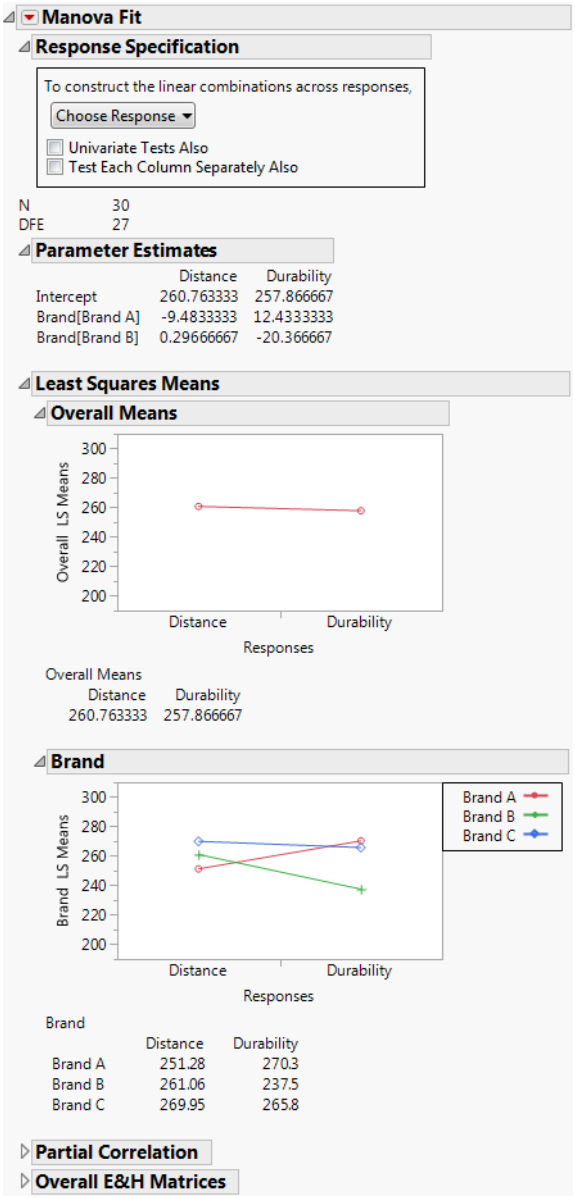
1. Select **Help > Sample Data Library** and open Golf Balls.jmp.
2. Select **Analyze > Fit Model**.
3. Select Distance and Durability and click **Y**.
4. Select Brand and click **Add**.
5. For **Personality**, select **Manova**.

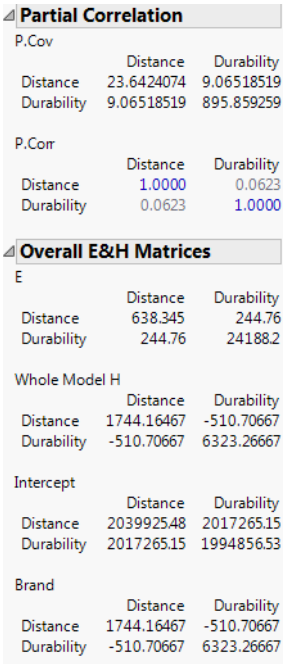
Figure 9.1 Manova Setup



6. Click **Run**.

Figure 9.2 Manova Report Window



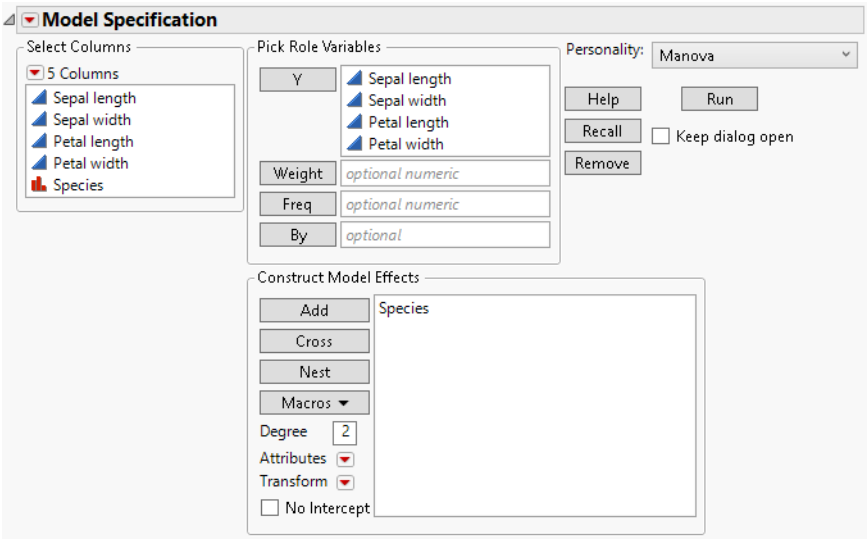


The initial results might not be very interesting in themselves, because no response design has been specified yet. After you specify a response design, the multivariate platform displays tables of multivariate estimates and tests. For more information about specifying a response design, see "Response Specification".

Launch the Manova Personality

Launch the Manova personality by selecting **Analyze > Fit Model**, entering one or more columns for **Y**, and selecting **Manova** from the **Personality** menu.

Figure 9.3 Fit Model Launch Window with Manova Selected



For more information about aspects of the Fit Model window that are common to all personalities, see "Model Specification". For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Manova Report

The Manova report window contains the following elements:

Manova Fit red triangle menu Contains save options. See "The Manova Fit Options".

Response Specification Enables you to specify the response designs for various tests. See "Response Specification".

Parameter Estimates Contains the parameter estimates for each response variable (without details like standard errors or *t tests*). There is a column for each response variable.

Least Squares Means Reports the overall least squares means of all of the response columns, least squares means of each nominal level, and least squares means plots of the means.

Partial Correlation Shows the covariance matrix and the partial correlation matrix of residuals from the initial fit, adjusted for the *X* effects.

Overall E&H Matrices Shows the E and H matrices:

- The elements of the **E** matrix are the cross products of the residuals.
- The **H** matrices correspond to hypothesis sums of squares and cross products.

There is an **H** matrix for the whole model and for each effect in the model. Diagonal elements of the **E** and **H** matrices correspond to the hypothesis (numerator) and error (denominator) sum of squares for the univariate *F* tests. New **E** and **H** matrices for any given response design are formed from these initial matrices, and the multivariate test statistics are computed from them.

The Manova Fit Options

The following Manova Fit options are available:

Save Discrim Performs a discriminant analysis and saves the results to the data table. See "Discriminant Analysis".

Save Predicted Saves the predicted responses to the data table.

Save Residuals Saves the residuals to the data table.

Model Dialog Shows the completed launch window for the current analysis.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

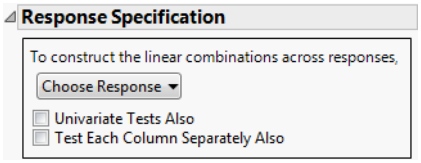
Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Response Specification

Specify the response designs for various tests using the Response Specification panel.

Figure 9.4 Response Specification Panel



Choose Response Provides choices for the M matrix. See "Choose Response Options".

Univariate Tests Also Obtains adjusted and unadjusted univariate repeated measures tests and multivariate tests. Use in repeated measures models.

Test Each Column Separately Also Obtains univariate ANOVA tests and multivariate tests on each response.

The following buttons are available only after you have chosen a response option:

Run Performs the analysis and shows the multivariate estimates and tests. See "Multivariate Tests".

Help Shows the Help for the Response Specification panel.

Orthogonalize Orthonormalizes the matrix. Orthonormalization is done after the column contrasts (sum to zero) for all response types except Sum.

Delete Last Column Reduces the dimensionality of the transformation.

Choose Response Options

The response design forms the M matrix. The columns of an M matrix define a set of transformation variables for the multivariate analysis. The **Choose Response** button contains the following options for the M matrix:

Repeated Measures Constructs and runs both Sum and Contrast responses.

Sum Sum of the responses that gives a single value.

Identity Uses each separate response, the identity matrix.

Contrast Compares each response and the first response.

Polynomial Constructs a matrix of orthogonal polynomials.

Helmert Compares each response with the combined responses listed below it.

Profile Compares each response with the following response.

Mean Compares each response with the mean of the others.

Compound Creates and runs several response functions that are appropriate if the responses are compounded from two effects.

Custom Uses any custom M matrix that you enter.

The most typical response designs are **Repeated Measures** and **Identity** for multivariate regression. There is little difference in the tests given by the **Contrast**, **Helmert**, **Profile**, and **Mean** options, since they span the same space. However, the tests and details in the Least Squares means and Parameter Estimates tables for them show correspondingly different highlights.

The **Repeated Measures** and the **Compound** options display dialogs to specify response effect names. They then fit several response functions without waiting for further user input. Otherwise, selections expand the control panel and give you more opportunities to refine the specification.

Custom Test Option

Set up custom tests of effect levels using the **Custom Test** option.

Note: For instructions on how to create custom tests, see "Custom Test".

Options for Model Effects

The menu icon beside each effect name gives you the following options to request additional information about the multivariate fit:

Test Details Displays the eigenvalues and eigenvectors of the $\mathbf{E}^{-1}\mathbf{H}$ matrix used to construct multivariate test statistics. See "Test Details".

Centroid Plot Plots the centroids (multivariate least squares means) on the first two canonical variables formed from the test space. See "Centroid Plot".

Save Canonical Scores Saves variables called Canon[1], Canon[2], and so on, as columns in the current data table. These columns have both the values and their formulas. For an example, see "Save Canonical Scores". For technical details, see "Canonical Details".

Contrast Performs the statistical contrasts of treatment levels that you specify in the contrasts dialog.

Note: The **Contrast** command is the same as for regression with a single response. See the "LSMeans Contrast", for a description and examples of the **LSMeans Contrast** commands.

Test Details

The Test Details report gives canonical details about the test for the whole model or the specified effect.

Eigenvalue The eigenvalues of the $\mathbf{E}^{-1}\mathbf{H}$ matrix used in computing the multivariate test statistics.

Canonical Corr The canonical correlations associated with each eigenvalue. This is the canonical correlation of the transformed responses with the effects, corrected for all other effects in the model.

Eigvec The eigenvectors of the $\mathbf{E}^{-1}\mathbf{H}$ matrix, or equivalently of $(\mathbf{E} + \mathbf{H})^{-1}\mathbf{H}$.

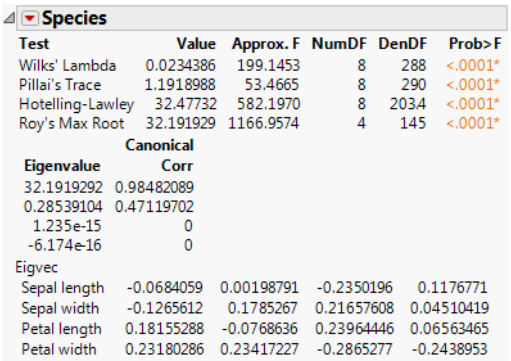
Example of Test Details

1. Select **Help > Sample Data Library** and open Iris.jmp.

   The Iris data (Mardia et al. 1979) have three levels of Species named Virginica, Setosa, and Versicolor. There are four measures (Petal length, Petal width, Sepal length, and Sepal width) taken on each sample.
2. Select **Analyze > Fit Model**.
3. Select Petal length, Petal width, Sepal length, and Sepal width and click **Y**.
4. Select Species and click **Add**.
5. For **Personality**, select **Manova**.
6. Click **Run**.
7. Click the **Choose Response** button and select **Identity**.
8. Click **Run**.
9. Click the Species red triangle and select **Test Details**.

   The eigenvalues, eigenvectors, and canonical correlations appear.

Figure 9.5 Test Details



Species

Test	Value	Approx. F	NumDF	DenDF	Prob>F
Wilks' Lambda	0.0234386	199.1453	8	288	<.0001*
Pillai's Trace	1.1918988	53.4665	8	290	<.0001*
Hotelling-Lawley	32.47732	582.1970	8	203.4	<.0001*
Roy's Max Root	32.191929	1166.9574	4	145	<.0001*

Eigenvalue	Canonical Corr
32.1919292	0.98482089
0.28539104	0.47119702
1.235e-15	0
-6.174e-16	0

Eigvec

Sepal length	-0.0684059	0.00198791	-0.2350196	0.1176771
Sepal width	-0.1265612	0.1785267	0.21657608	0.04510419
Petal length	0.18155288	-0.0768636	0.23964446	0.06563465
Petal width	0.23180286	0.23417227	-0.2865277	-0.2438953

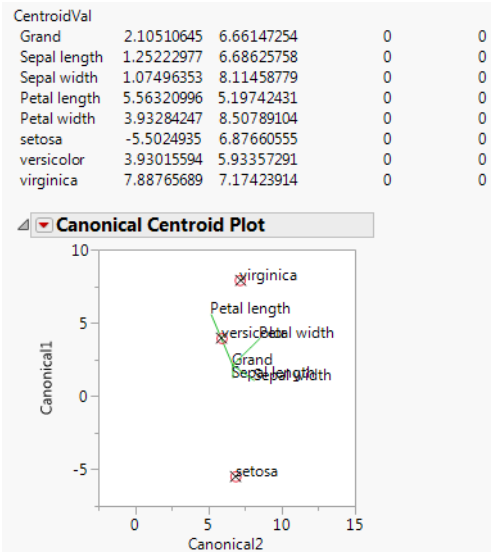
Centroid Plot

The **Centroid Plot** command (accessed from the Species red triangle) plots the centroids (multivariate least squares means) on the first two canonical variables formed from the test space, as in Figure 9.6. The first canonical axis is the vertical axis so that if the test space is only one dimensional the centroids align on a vertical axis. The centroid points appear with a circle corresponding to the 95% confidence region (Mardia et al. 1979). When centroid plots are created under effect tests, circles corresponding to the effect being tested appear in red. Other circles appear blue. Biplot rays show the directions of the original response variables in the test space. See "Details for Centroid Plot Option".

Click the **Centroid Val** disclosure icon to show additional information, shown in Figure 9.6.

The first canonical axis with an eigenvalue accounts for much more separation than does the second axis. The means are well separated (discriminated), with the first group farther apart than the other two. The first canonical variable seems to load the petal length variables against the petal width variables. Relationships among groups of variables can be verified with Biplot Rays and the associated eigenvectors.

Figure 9.6 Centroid Plot and Centroid Values



Save Canonical Scores

Saves columns called Canon[i] to the data table, where *i* refers to the i^{th} canonical score for the Y variables. The canonical scores are computed based on the $\mathbf{E}^{-1}\mathbf{H}$ matrix used to construct the multivariate test statistic. Canonical scores are saved for eigenvectors corresponding to nonzero eigenvalues.

Canonical Correlation

Canonical correlation analysis is not a specific command, but it can be performed using a sequence of commands in the multivariate fitting platform.

1. Follow step 1 through step 8 in "Example of Test Details".
2. Click the Whole Model red triangle and select **Test Details**.
3. Click the Whole Model red triangle and select **Save Canonical Scores**.

The details list the canonical correlations (Canonical Corr) next to the eigenvalues. The saved variables are called Canon[1], Canon[2], and so on. These columns contain both the values and their formulas.

To obtain the canonical variables for the *X* side, repeat the same steps, but interchange the *X* and *Y* variables. If you already have the columns Canon[n] appended to the data table, the new columns are called Canon[n] 2 (or another number) that makes the name unique.

This is an additional example of canonical correlation analysis.

1. Select **Help > Sample Data Library** and open Exercise.jmp.

2. Select **Analyze > Fit Model**.
3. Select chins, situps, and jumps and click Y.
4. Select weight, waist, and pulse and click **Add**.
5. For **Personality**, select **Manova**.
6. Click **Run**.
7. Click the **Choose Response** button and select **Identity**.
8. Click **Run**.
9. Click the Whole Model red triangle and select **Test Details**.
10. Click the Whole Model red triangle and select **Save Canonical Scores**.

Figure 9.7 Canonical Correlations

Whole Model

Test	Value	Approx. F	NumDF	DenDF	Prob>F
Wilks' Lambda	0.3503905	2.0482	9	34.223	0.0635
Pillai's Trace	0.6784815	1.5587	9	48	0.1551
Hotelling-Lawley	1.7719415	2.6397	9	19.053	0.0357*
Roy's Max Root	1.7247387	9.1986	3	16	0.0009*

Eigenvalue	Canonical Corr
1.72473874	0.79560815
0.0419084	0.20055604
0.00529433	0.07257029

Eigvec

chins	0.02503681	-0.016636	0.05641878
situps	0.00637953	0.0004622	-0.004547
jumps	-0.0052909	0.0048507	0.0018787

The output canonical variables use the eigenvectors shown as the linear combination of the Y variables. For example, Canon[1] is calculated as follows:

```
0.02503681*chins + 0.00637953*situps + -0.0052909*jumps
```

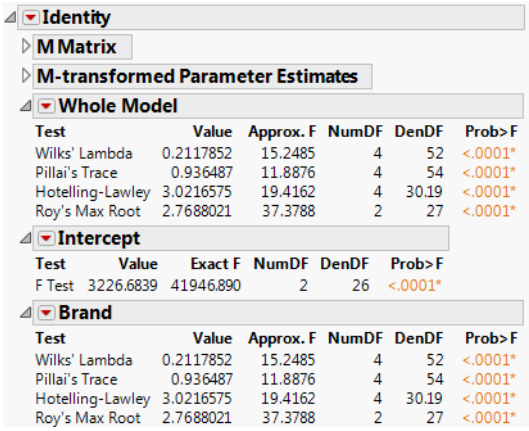
This canonical analysis does not produce a standardized variable with mean 0 and standard deviation 1, but it is easy to define a new standardized variable with the calculator that has these features.

Multivariate Tests

Example Choosing a Response

1. Complete step 1 through step 6 in "Example of a Multiple Response Model".
2. Click the **Choose Response** button and select **Identity**.
3. Click **Run**.

Figure 9.8 Multivariate Test Reports



Identity

M Matrix

M-transformed Parameter Estimates

Whole Model

Test	Value	Approx. F	NumDF	DenDF	Prob>F
Wilks' Lambda	0.2117852	15.2485	4	52	<.0001*
Pillai's Trace	0.936487	11.8876	4	54	<.0001*
Hotelling-Lawley	3.0216575	19.4162	4	30.19	<.0001*
Roy's Max Root	2.7688021	37.3788	2	27	<.0001*

Intercept

Test	Value	Exact F	NumDF	DenDF	Prob>F
F Test	3226.6839	41946.890	2	26	<.0001*

Brand

Test	Value	Approx. F	NumDF	DenDF	Prob>F
Wilks' Lambda	0.2117852	15.2485	4	52	<.0001*
Pillai's Trace	0.936487	11.8876	4	54	<.0001*
Hotelling-Lawley	3.0216575	19.4162	4	30.19	<.0001*
Roy's Max Root	2.7688021	37.3788	2	27	<.0001*

The **M Matrix** report gives the response design that you specified. The **M-transformed Parameter Estimates** report gives the original parameter estimates matrix multiplied by the transpose of the M matrix.

Note: Initially in this chapter, the matrix names **E** and **H** refer to the error and hypothesis cross products. After specification of a response design, **E** and **H** refer to those matrices transformed by the response design, which are actually **M′EM** and **M′HM**.

The Extended Multivariate Report

In multivariate fits, the sums of squares due to hypothesis and error are matrices of squares and cross products instead of single numbers. And there are lots of ways to measure how large a value the matrix for the hypothesis sums of squares and cross products (called **H** or **SSCP**) is compared to that matrix for the residual (called **E**). JMP reports the four multivariate tests that are commonly described in the literature. If you are looking for a test at an exact significance level, you might need to go hunting for tables in reference books. Fortunately, all four tests can be transformed into an approximate *F* test. If the response design yields a single value, or if the hypothesis is a single degree of freedom, the multivariate tests are equivalent and yield the same exact *F* test. JMP labels the test Exact F. Otherwise, JMP labels it Approx. F.

In the golf balls example, there is only one effect, so the Whole Model test and the test for Brand are the same, which show the four multivariate tests with approximate *F* tests. There is only a single intercept with two DF (one for each response), so the *F* test for it is exact and is labeled Exact F.

The red triangle menus on the Whole Model, Intercept, and Brand reports contain options to generate additional information, which includes eigenvalues, canonical correlations, a list of centroid values, a centroid plot, and a **Save** command that lets you save canonical variates.

The effect (Brand in this example) pop-up menu also includes the option to specify contrasts.

The custom test and contrast features are the same as those for regression with a single response. See "Standard Least Squares Models".

To see formulas for the MANOVA table tests, see "Multivariate Tests".

The extended Multivariate Report contains the following columns:

Test Labels each statistical test in the table. If the number of response function values (columns specified in the **M** matrix) is 1 or if an effect has only one degree of freedom per response function, the exact *F* test is presented. Otherwise, the standard four multivariate test statistics are given with approximate *F* tests: Wilks' Lambda (Λ), Pillai's Trace, the Hotelling-Lawley Trace, and Roy's Maximum Root.

Value Value of each multivariate statistical test in the report.

Approx. F (or Exact F) *F*-values corresponding to the multivariate tests. If the response design yields a single value or if the test is one degree of freedom, this is an exact *F* test.

NumDF Numerator degrees of freedom.

DenDF Denominator degrees of freedom.

Prob>F Significance probability corresponding to the *F*-value.

Note: For more information about the Sphericity Test table, see "Univariate Tests and the Test for Sphericity".

Comparison of Multivariate Tests

Although the four standard multivariate tests often give similar results, there are situations where they differ, and one might have advantages over another. Unfortunately, there is no clear winner. In general, here is the order of preference in terms of power:

1. Pillai's Trace
2. Wilks' Lambda
3. Hotelling-Lawley Trace
4. Roy's Maximum Root

When there is a large deviation from the null hypothesis and the eigenvalues differ widely, the order of preference is the reverse (Seber 1984).

Univariate Tests and the Test for Sphericity

There are cases, such as a repeated measures model, that allow transformation of a multivariate problem into a univariate problem (Huynh and Feldt 1970). Using univariate tests in a multivariate context is valid in the following situations:

- If the response design matrix **M** is orthonormal (**M′M** = Identity).
- If **M** yields more than one response the coefficients of each transformation sum to zero.
- If the *sphericity* condition is met. The sphericity condition means that the M-transformed responses are uncorrelated and have the same variance. **M′ΣM** is proportional to an identity matrix, where Σ is the covariance of the *Y* variables.

If these conditions hold, the diagonal elements of the **E** and **H** test matrices sum to make a univariate sums of squares for the denominator and numerator of an *F* test. Note that if the above conditions do not hold, then an error message appears. In the case of Golf Balls.jmp, an identity matrix is specified as the **M**-matrix. Identity matrices cannot be transformed to a full rank matrix after centralization of column vectors and orthonormalization. So the univariate request is ignored.

Example of Univariate and Sphericity Test

1. Select **Help > Sample Data Library** and open Dogs.jmp.
2. Select **Analyze > Fit Model**.
3. Select LogHist0, LogHist1, LogHist3, and LogHist5 and click **Y**.
4. Select drug and dep1 and click **Add**.
5. In the Construct Model Effects panel, select drug. In the Select Columns panel, select dep1. Click **Cross**.
6. For Personality, select **Manova**.
7. Click **Run**.
8. Select the check box next to **Univariate Tests Also**.
9. In the **Choose Response** menu, select **Repeated Measures**.

   Time should be entered for YName, and **Univariate Tests Also** should be selected.
10. Click **OK**.

Figure 9.9 Sphericity Test

Sphericity Test	
Mauchly Criterion	0.1752641
ChiSquare	16.930873
DF	5
Prob >Chisq	0.0046328

The sphericity test checks the appropriateness of an unadjusted univariate *F* test for the within-subject effects using the Mauchly criterion to test the sphericity assumption (Anderson 1958). The sphericity test and the univariate tests are always done using an orthonormalized **M** matrix. Use the following guidelines to interpret the sphericity test:

- If the true covariance structure is spherical, you can use the unadjusted univariate *F* tests.
- If the sphericity test is significant, the test suggests that the true covariance structure is not spherical. Therefore, you can use the multivariate or the adjusted univariate tests.

The univariate *F* statistic has an approximate *F* distribution even without sphericity, but the degrees of freedom for numerator and denominator are reduced by some fraction epsilon (ε). Box (1954), Greenhouse and Geisser (1959), and Huynh-Feldt (1976) offer techniques for estimating the epsilon degrees-of-freedom adjustment. Muller and Barton (1989) recommend the Greenhouse-Geisser version, based on a study of power.

The epsilon adjusted tests in the multivariate report are labeled G-G (Greenhouse-Geisser) or H-F (Huynh-Feldt). The epsilon adjustment is shown in the value column.

Multivariate Model with Repeated Measures

One common use of multivariate fitting is to analyze data with repeated measures, also called *longitudinal data*. A subject is measured repeatedly across time, and the data are arranged so that each of the time measurements form a variable. Because of correlation between the measurements, data should not be stacked into a single column and analyzed as a univariate model unless the correlations form a pattern termed *sphericity*. See the previous section, "Univariate Tests and the Test for Sphericity", for more information about this topic.

With repeated measures, the analysis is divided into two layers:

- Between-subject (or across-subject) effects are modeled by fitting the sum of the repeated measures columns to the model effects. This corresponds to using the **Sum** response function. This response function is an **M**-matrix that is a single vector of 1s.
- Within-subjects effects (repeated effects, or time effects) are modeled with a response function that fits differences in the repeated measures columns. This analysis can be done using the **Contrast** response function or any of the other similar differencing functions: **Polynomial**, **Helmert**, **Profile**, or **Mean**. When you model differences across the repeated measures, think of the differences as being a new within-subjects effect, usually time. When you fit effects in the model, interpret them as the interaction with the within-subjects effect. For example, the effect for Intercept becomes the Time (within-subject) effect, showing overall differences across the repeated measures. If you have an effect A, the within-subjects tests are interpreted to be the tests for the A*Time interaction, which model how the differences across repeated measures vary across the A effect.

Figure 9.1 shows the relationship between the response function and the model effects compared with what a univariate model specification would be. Using both the **Sum** (between-subjects) and **Contrast** (within-subjects) models, you should be able to reconstruct the tests that would have resulted from stacking the responses into a single column and obtaining a standard univariate fit.

There is a direct and an indirect way to perform the repeated measures analyses:

- The direct way is to use the pop-up menu item Repeated Measures. This prompts you to name the effect that represents the within-subject effect across the repeated measures. Then it fits both the **Contrast** and the **Sum** response functions. An advantage of this way is that the effects are labeled appropriately with the within-subjects effect name.
- The indirect way is to specify the two response functions individually. First, do the **Sum** response function and second, do either **Contrast** or one of the other functions that model differences. You need to remember to associate the within-subjects effect with the model effects in the contrast fit.

Repeated Measures Example

For example, consider a study by Cole and Grizzle (1966). The results are in the Dogs.jmp table in the sample data folder. Sixteen dogs are assigned to four groups defined by variables drug and dep1, each having two levels. The dependent variable is the blood concentration of histamine at 0, 1, 3, and 5 minutes after injection of the drug. The log of the concentration is used to minimize the correlation between the mean and variance of the data.

1. Select **Help > Sample Data Library** and open Dogs.jmp.
2. Select **Analyze > Fit Model**.
3. Select LogHist0, LogHist1, LogHist3, and LogHist5 and click **Y**.
4. Select drug and dep1 and select **Full Factorial** from the **Macros** menu.
5. For Personality, select **Manova**.
6. Click **Run**.
7. In the **Choose Response** menu, select **Repeated Measures**.

   Time should be entered for YName. If you check the **Univariate Tests Also** check box, the report includes univariate tests, which are calculated as if the responses were stacked into a single column.
8. Click **OK**.

Figure 9.10 Repeated Measures Window

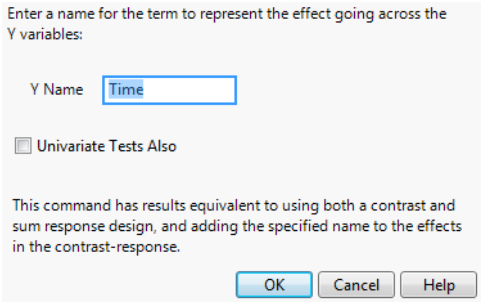


Table 9.1 shows how the multivariate tests for a **Sum** and **Contrast** response designs correspond to how univariate tests would be labeled if the data for columns LogHist0, LogHist1, LogHist3, and LogHist5 were stacked into a single *Y* column. The new rows are identified with a nominal grouping variable, Time.

Table 9.1 Corresponding Multivariate and Univariate Tests

Sum M-Matrix Between Subjects		**Contrast M-Matrix Within Subjects**	
Multivariate Test	**Univariate Test**	**Multivariate Test**	**Univariate Test**
intercept	intercept	intercept	time
drug	drug	drug	time*drug
depl	depl	depl	time*depl

The between-subjects analysis is produced first. This analysis is the same (except titling) as it would have been if **Sum** had been selected on the pop-up menu.

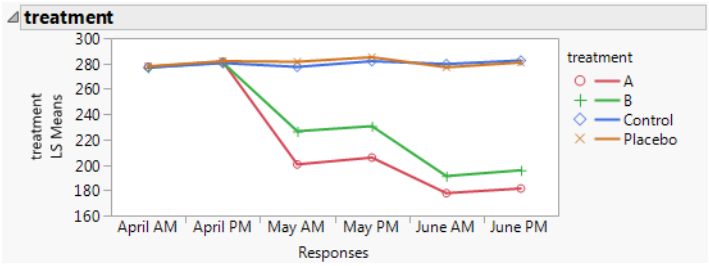
The within-subjects analysis is produced next. This analysis is the same (except titling) as it would have been if **Contrast** had been selected on the pop-up menu, though the within-subject effect name (Time) has been added to the effect names in the report. Note that the position formerly occupied by Intercept is Time, because the intercept term is estimating overall differences across the repeated measurements.

Example of a Compound Multivariate Model

JMP can handle data with layers of repeated measures. For example, see the Cholesterol.jmp data table. Groups of five participants belong to one of four treatment groups called A, B, Control, and Placebo. Cholesterol was measured in the morning and again in the afternoon once a month for three months (the data are fictional). In this example, the response columns are arranged chronologically with time of day within month.

1. Select **Help > Sample Data Library** and open Cholesterol.jmp.
2. Select **Analyze > Fit Model**.
3. Select April AM, April PM, May AM, May PM, June AM, and June PM and click **Y**.
4. Select treatment and click **Add**.
5. Next to Personality, select **Manova**.
6. Click **Run**.

Figure 9.11 Treatment Graph



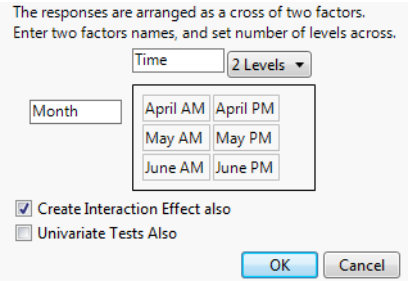
In the treatment graph, you can see that the four treatment groups began the study with very similar mean cholesterol values. The A and B treatment groups appear to have lower cholesterol values at the end of the trial period. The control and placebo groups remain unchanged.

7. Click the **Choose Response** menu and select **Compound**.

   Complete this window to tell JMP how the responses are arranged in the data table and the number of levels of each response. In the cholesterol example, the time of day columns are arranged within month. Therefore, you name time of day as one factor and the month effect as the other factor. Testing the interaction effect is optional.

8. Use the options in Figure 9.12 to complete the window.

Figure 9.12 Compound Window



9. Click **OK**.

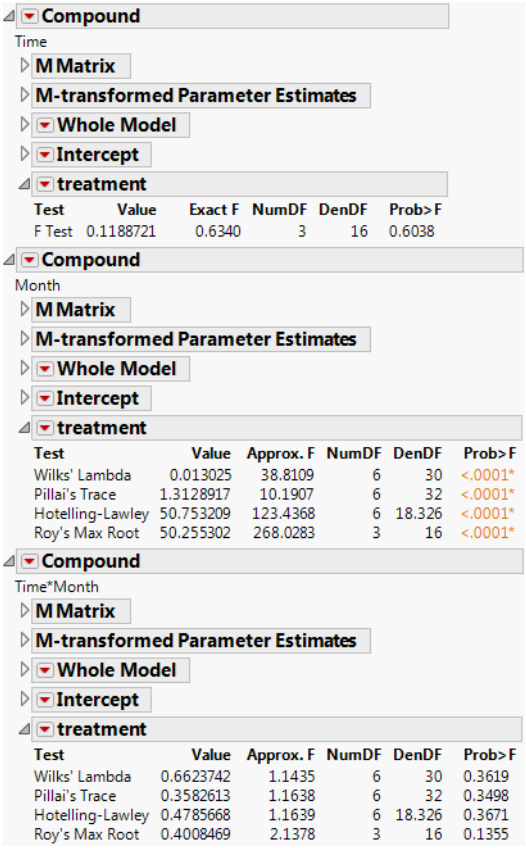
   The tests for each effect appear. Parts of the report are shown in Figure 9.13. Note the following:

   – The report for Time shows a *p*-value of 0.6038 for the interaction between Time and treatment, indicating that the interaction is not significant. This means that there is no evidence of a difference in treatment between AM and PM. Since Time has two levels (AM and PM), the exact *F* test appears.

   – The report for Month shows *p*-values of <.0001 for the interaction between Month and treatment, indicating that the interaction is significant. This suggests that the differences between treatment groups change depending on the month. The treatment graph in Figure 9.11 indicates no difference among the groups in April, but the difference between treatment types (A, B, Control, and Placebo) becomes large in May and even larger in June.

   – The report for Time*Month shows no significant *p*-values for treatment. This indicates that the three-way interaction effect involving Time, Month, and treatment is not statistically significant.

Figure 9.13 Cholesterol Study Results



Compound

Time

M Matrix

M-transformed Parameter Estimates

Whole Model

Intercept

treatment

Test	Value	Exact F	NumDF	DenDF	Prob>F
F Test	0.1188721	0.6340	3	16	0.6038

Compound

Month

M Matrix

M-transformed Parameter Estimates

Whole Model

Intercept

treatment

Test	Value	Approx. F	NumDF	DenDF	Prob>F
Wilks' Lambda	0.013025	38.8109	6	30	<.0001*
Pillai's Trace	1.3128917	10.1907	6	32	<.0001*
Hotelling-Lawley	50.753209	123.4368	6	18.326	<.0001*
Roy's Max Root	50.255302	268.0283	3	16	<.0001*

Compound

Time*Month

M Matrix

M-transformed Parameter Estimates

Whole Model

Intercept

treatment

Test	Value	Approx. F	NumDF	DenDF	Prob>F
Wilks' Lambda	0.6623742	1.1435	6	30	0.3619
Pillai's Trace	0.3582613	1.1638	6	32	0.3498
Hotelling-Lawley	0.4785668	1.1639	6	18.326	0.3671
Roy's Max Root	0.4008469	2.1378	3	16	0.1355

Discriminant Analysis

Discriminant analysis is a method of predicting some level of a one-way classification based on known values of the responses. The technique is based on how close the measurement variables are to the multivariate means of the levels being predicted. Discriminant analysis is more fully implemented using the Discriminant Platform. See *Multivariate Methods*.

In JMP, you specify the measurement variables as *Y* effects and the classification variable as a single *X* effect. The multivariate fitting platform gives estimates of the means and the covariance matrix for the data, assuming that the covariances are the same for each group. You obtain discriminant information with the **Save Discrim** option in the pop-up menu next to the MANOVA platform name. This command saves distances and probabilities as columns in the current data table using the initial **E** and **H** matrices.

For a classification variable with k levels, JMP adds k distance columns, k classification probability columns, the predicted classification column, and two columns of other computational information to the current data table.

Example of the Save Discrim Option

Examine Fisher's Iris data as found in Mardia et al. (1979). There are $k = 3$ levels of species and four measures on each sample.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Analyze > Fit Model**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y**.
4. Select Species and click **Add**.
5. Next to Personality, select **Manova**.
6. Click **Run**.
7. Click the Manova Fit red triangle and select **Save Discrim**.

The following columns are added to the Iris.jmp sample data table:

SqDist[0] Quadratic form needed in the Mahalanobis distance calculations.

SqDist[setosa] Mahalanobis distance of the observation from the Setosa centroid.

SqDist[versicolor] Mahalanobis distance of the observation from the Versicolor centroid.

SqDist[virginica] Mahalanobis distance of the observation from the Virginica centroid.

Prob[0] Sum of the negative exponentials of the Mahalanobis distances, used below.

Prob[setosa] Probability of being in the Setosa category.

Prob[versicolor] Probability of being in the Versicolor category.

Prob[virginica] Probability of being in the Virginica category.

Pred Species Species that is most likely from the probabilities.

Now you can use the new columns in the data table with other JMP platforms to summarize the discriminant analysis with reports and graphs. For example:

1. From the updated Iris.jmp data table (that contains the new columns) select **Analyze > Fit Y by X**.
2. Select Species and click **Y, Response**.
3. Select Pred Species and click **X, Factor**.
4. Click **OK**.

The Contingency Table summarizes the discriminant classifications. Three misclassifications are identified.

Figure 9.14 Contingency Table of Predicted and Actual Species

Contingency Table

Pred Species	Count / Total % / Col % / Row %	setosa	versicolor	virginica	Total
		Species			
setosa		50	0	0	50
		33.33	0.00	0.00	33.33
		100.00	0.00	0.00	
		100.00	0.00	0.00	
versicolor		0	48	1	49
		0.00	32.00	0.67	32.67
		0.00	96.00	2.00	
		0.00	97.96	2.04	
virginica		0	2	49	51
		0.00	1.33	32.67	34.00
		0.00	4.00	98.00	
		0.00	3.92	96.08	
Total		50	50	50	150
		33.33	33.33	33.33	

Statistical Details for the Manova Personality

- "Multivariate Tests"
- "Approximate F-Tests"
- "Canonical Details"

Multivariate Tests

In the following, **E** is the residual cross product matrix and **H** is the model cross product matrix. Diagonal elements of **E** are the residual sums of squares for each variable. Diagonal elements of **H** are the sums of squares for the model for each variable. In the discriminant analysis literature, **E** is often called **W**, where **W** stands for *within*.

Test statistics in the multivariate results tables are functions of the eigenvalues λ of $\mathbf{E}^{-1}\mathbf{H}$. The following list describes the computation of each test statistic.

Note: After specification of a response design, the initial **E** and **H** matrices are premultiplied by **M**′ and postmultiplied by **M**.

- Wilks' Lambda

$$\Lambda = \frac{\det(\mathbf{E})}{\det(\mathbf{H}+\mathbf{E})} = \prod_{i=1}^{n}\left(\frac{1}{1+\lambda_i}\right)$$

- Pillai's Trace

$$V = \text{Trace}[\mathbf{H}(\mathbf{H}+\mathbf{E})^{-1}] = \sum_{i=1}^{n}\frac{\lambda_i}{1+\lambda_i}$$

- Hotelling-Lawley Trace

$$U = \text{Trace}(\mathbf{E}^{-1}\mathbf{H}) = \sum_{i=1}^{n}\lambda_i$$

- Roy's Max Root

$\Theta = \lambda_1$, the maximum eigenvalue of $\mathbf{E}^{-1}\mathbf{H}$.

E and **H** are defined as follows:

$$\mathbf{E} = \mathbf{Y}'\mathbf{Y} - \mathbf{b}'(\mathbf{X}'\mathbf{X})\mathbf{b}$$

$$\mathbf{H} = (\mathbf{L}\mathbf{b})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{L}\mathbf{b})$$

where **b** is the estimated vector for the model coefficients and $\mathbf{A}^{-}$ denotes the generalized inverse of a matrix **A**.

The whole model **L** is a column of zeros (for the intercept) concatenated with an identity matrix having the number of rows and columns equal to the number of parameters in the model. **L** matrices for effects are subsets of rows from the whole model **L** matrix.

Approximate F-Tests

To compute *F*-values and degrees of freedom, let *p* be the rank of $\mathbf{H}+\mathbf{E}$. Let *q* be the rank of $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{L}'$, where the **L** matrix identifies elements of $\mathbf{X}'\mathbf{X}$ associated with the effect being tested. Let *v* be the error degrees of freedom and *s* be the minimum of *p* and *q*. Also let $m = 0.5(|p-q|-1)$ and $n = 0.5(v-p-1)$.

Table 9.2, gives the computation of each approximate F from the corresponding test statistic.

Table 9.2 Approximate *F*-statistics

Test	Approximate F	Numerator DF	Denominator DF
Wilks' Lambda	$F = \left(\frac{1-\Lambda^{1/t}}{\Lambda^{1/t}}\right)\left(\frac{rt-2u}{pq}\right)$	pq	$rt-2u$
Pillai's Trace	$F = \left(\frac{V}{s-V}\right)\left(\frac{2n+s+1}{2m+s+1}\right)$	$s(2m+s+1)$	$s(2n+s+1)$
Hotelling-Lawley Trace	$F = \frac{2(sn+1)U}{s^2(2m+s+1)}$	$s(2m+s+1)$	$2(sn+1)$
Roy's Max Root	$F = \frac{\Theta(v-\max(p,q)+q)}{\max(p,q)}$	$\max(p,q)$	$v-\max(p,q)+q$

Canonical Details

Details for the Test Details Option

When you select the Test Details option for a given test, eigenvalues, canonical correlations, and eigenvectors are shown in the report.

The canonical correlations produced by the Test Details option are computed as follows:

$$\rho_i = \sqrt{\frac{\lambda_i}{1+\lambda_i}}$$

where λ_i is the i^{th} eigenvalue of the $\mathbf{E}^{-1}\mathbf{H}$ matrix used in computing the multivariate test statistics

The matrix labeled Eigvec is the **V** matrix, which is the matrix of eigenvectors of $\mathbf{E}^{-1}\mathbf{H}$ for the given test.

Note: The **E** and **H** matrices for the given test refer to **M′EM** and **M′HM** in terms of the original **E** and **H** matrices. The **M** matrix is defined by the response design. The **E** and **H** used in this section are defined in "Multivariate Tests".

Details for Centroid Plot Option

The total sample centroid and centroid values for effects are computed as follows:

$$\text{Grand} = (c'_1\bar{y}, c'_2\bar{y}, \ldots, c'_g\bar{y})$$

$$\text{Effect}_j = (c'_1\bar{x}_j, c'_2\bar{x}_j, \ldots, c'_g\bar{x}_j)$$

where

$$c_i = \left(v'_i\left(\frac{E}{N-r}\right)v_i\right)^{-1/2} v_i$$

N is the number of observations

v_i is the i^{th} column of **V**, the eigenvector matrix of $\mathbf{E}^{-1}\mathbf{H}$ for the given test

$\bar{x}_j$ is the multivariate least squares mean for the j^{th} effect

$\bar{y}$ is the overall mean of the responses

g is the number of eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$ greater than 0

r is the rank of the **X** matrix

Note: The **E** and **H** matrices for the given test refer to **M′EM** and **M′HM** in terms of the original **E** and **H** matrices. The **M** matrix is defined by the response design. The **E** and **H** used in this section are defined in "Multivariate Tests".

The centroid radii for effects are calculated as follows:

$$d = \sqrt{\frac{\chi^2_g(0.95)}{L(X'X)^{-1}L'}}$$

where g is the number of eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$ greater than 0 and the **L** matrices in the denominator are from the multivariate least squares means calculations.

Details for the Save Canonical Scores Option

The canonical Y values are calculated as follows:

$$\tilde{Y} = \mathbf{YM'V}$$

where

Y is the matrix of response variables

M′ is the transpose of the response design matrix

V is the matrix of eigenvectors of $\mathbf{E}^{-1}\mathbf{H}$ for the given test

Note: The **E** and **H** matrices for the given test refer to **M′EM** and **M′HM** in terms of the original **E** and **H** matrices. The **M** matrix is defined by the response design. The **E** and **H** used in this section are defined in "Multivariate Tests".

Canonical Y values are saved for eigenvectors corresponding to eigenvalues larger than zero.

Chapter 10

Loglinear Variance Models

Model the Variance and the Mean of the Response

The Loglinear Variance personality of the Fit Model platform enables you to model both the expected value and the variance of a response using regression models. The log of the variance is fit to one linear model and the expected response is fit to a different linear model simultaneously.

Note: The estimates are demanding in their need for a lot of well-designed, well-fitting data. You need more data to fit variances than you do means.

For many engineers, the goal of an experiment is not to maximize or minimize the response itself, but to aim at a target response and achieve minimum variability. The loglinear variance model provides a very general and effective way to model variances, and can be used for unreplicated data, as well as data with replications.

Contents

Overview of the Loglinear Variance Model 467
Dispersion Effects 467
Model Specification 467
Notes 468
Launch the Loglinear Variance Personality 468
Example Using Loglinear Variance 469
The Loglinear Report 471
Loglinear Platform Options 472
Save Columns 473
Row Diagnostics 473
Examining the Residuals 474
Profiling the Fitted Model 475
Example of Profiling the Fitted Model 475

Overview of the Loglinear Variance Model

The loglinear-variance model provides a way to model the variance simply through a linear model. See Harvey (1976), Cook and Weisberg (1983), Aitken (1987), and Carroll and Ruppert (1988). In addition to having regressor terms to model the mean response, there are regressor terms in a linear model to model the log of the variance:

mean model: $E(y) = \mathbf{X}\beta$

variance model: $\log(Variance(y)) = \mathbf{Z}\ \lambda$,

or equivalently

$Variance(y) = \exp(\mathbf{Z}\ \lambda)$

where the columns of X are the regressors for the mean of the response, and the columns of Z are the regressors for the variance of the response. The regular linear model parameters are represented by β, and λ represents the parameters of the variance model.

Log-linear variance models are estimated using REML.

A *dispersion* or *log-variance* effect can model changes in the variance of the response. This is implemented in the Fit Model platform by a fitting personality called the Loglinear Variance personality.

Dispersion Effects

Modeling dispersion effects is not very widely covered in textbooks, with the exception of the Taguchi framework. In a Taguchi-style experiment, this is handled by taking multiple measurements across settings of an outer array, constructing a new response that measures the variability off-target across this outer array, and then fitting the model to find out the factors that produce minimum variability. This type of modeling requires a specialized design that is a complete Cartesian product of two designs. The method of this chapter models variances in a more flexible, model-based approach. The particular performance statistic that Taguchi recommends for variability modeling is $STD = -\log(s)$. In the methodology used in JMP, the $\log(s^2)$ is modeled and combined with a model that has a mean. The two are basically equivalent, since $\log(s^2)=2\log(s)$.

Model Specification

Log-linear variance effects are specified in the Fit Model dialog by highlighting them and selecting **LogVariance Effect** from the **Attributes** drop-down menu. **&LogVariance** appears at the end of the effect. When you use this attribute, it also changes the fitting **Personality** at the top to **LogLinear Variance**. If you want an effect to be used for both the mean and variance of the response, then you must specify it twice, once with the **LogVariance** option.

The effects that you specify with the log-variance attribute become the effects that generate the *Z*s in the model, and the other effects become the *X*s in the model.

Notes

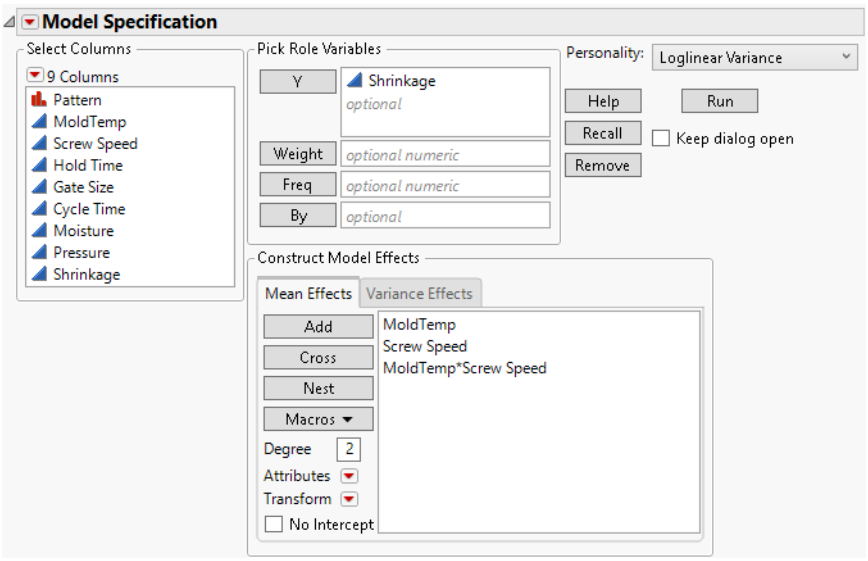
Every time another parameter is estimated for the mean model, at least one more observation is needed, and preferably more. But with variance parameters, several more observations for each variance parameter are needed to obtain reasonable estimates. It takes more data to estimate variances than it does means.

The log-linear variance model is a very flexible way to fit dispersion effects, and the method deserves much more attention than it has received so far in the literature.

Launch the Loglinear Variance Personality

Launch the Loglinear Variance personality by selecting **Analyze > Fit Model**, entering one or more columns for **Y**, and selecting **Loglinear Variance** from the **Personality** menu.

Figure 10.1 Fit Model Launch Window with Loglinear Variance Selected



For more information about aspects of the Fit Model window that are common to all personalities, see "Model Specification". For more information about the options in the Select Columns red triangle menu, see *Using JMP*. Information specific to the Loglinear Variance personality is presented here.

If your model effects have missing values, you can treat these missing values as informative categories. Select the Informative Missing option from the Model Specification red triangle menu.

Example Using Loglinear Variance

The data table InjectionMolding.jmp contains the experimental results from a 7-factor 2^{7-3} fractional factorial design with four added centerpoints (Montgomery 1991). Preliminary investigation determined that the mean response only seemed to vary with the first two factors, Mold Temperature, and Screw Speed, and the variance seemed to be affected by Holding Time.

Figure 10.2 Injection Molding Data

InjectionMolding
Locked File C:\Program
Model

Columns (9/0)

Pattern
MoldTemp *
Screw Speed *
Hold Time *
Gate Size *
Cycle Time + *
Moisture + *
Pressure + *
Shrinkage *

Rows
All rows 20
Selected 0
Excluded 0
Hidden 0
Labeled 0

	Pattern	MoldTemp	Screw Speed	Hold Time	Gate Size	Cycle Time	Moisture	Pressure	Shrinkage
1	-------	-1	-1	-1	-1	-1	-1	-1	6
2	+-----+	1	-1	-1	-1	1	-1	1	10
3	-+---+-	-1	1	-1	-1	1	1	-1	32
4	++--++-	1	1	-1	-1	-1	1	1	60
5	--+-++-	-1	-1	1	-1	1	1	1	4
6	+-+-+-+	1	-1	1	-1	-1	1	-1	15
7	-++--++	-1	1	1	-1	-1	-1	1	26
8	+++----	1	1	1	-1	1	-1	-1	60
9	---+++-	-1	-1	-1	1	-1	1	1	8
10	+--+-+-	1	-1	-1	1	1	1	-1	12
11	-+-+-+-	-1	1	-1	1	1	-1	1	34
12	++-+--+	1	1	-1	1	-1	-1	-1	60
13	--+++-+	-1	-1	1	1	1	-1	-1	16
14	+-+++--	1	-1	1	1	-1	-1	1	5
15	-++++--	-1	1	1	1	-1	1	-1	37
16	+++++++	1	1	1	1	1	1	1	52
17	0000000	0	0	0	0	0	0	0	25
18	0000000	0	0	0	0	0	0	0	29
19	0000000	0	0	0	0	0	0	0	24
20	0000000	0	0	0	0	0	0	0	27

1. Select **Help > Sample Data Library** and open InjectionMolding.jmp.
2. Select **Analyze > Fit Model**.

   Since the variables in the data table have been assigned preselected roles, the analysis runs automatically.

Figure 10.3 Loglinear Variance Report Window

Loglinear Variance Fit

Fit Statistics

-2 Residual Log Likelihood	90.712809
-2 Log Likelihood	87.777035
AICc	106.23857
BIC	105.75143

Mean Model for Shrinkage

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	27.582516	0.380697	72.45	<.0001*
MoldTemp	7.683713	0.392907	19.56	<.0001*
Screw Speed	18.673515	0.392907	47.53	<.0001*
MoldTemp*Screw Speed	5.765297	0.392907	14.67	<.0001*

Fixed Effect Tests

Source	Nparm	DF	DFDen	F Ratio	Prob > F
MoldTemp	1	1	5.758	382.4387	<.0001*
Screw Speed	1	1	5.758	2258.768	<.0001*
MoldTemp*Screw Speed	1	1	5.758	215.3094	<.0001*

Variance Model for Shrinkage

Likelihood Ratio Test for Equal Variance

Source	-2 Residual Log Likelihood
Equal Variances Initially	101.76083063
After Fitted Variances	90.712808548
Difference: Chi-sq	11.048022086
Degrees of Freedom	1
Prob > ChiSquare	0.0008878186

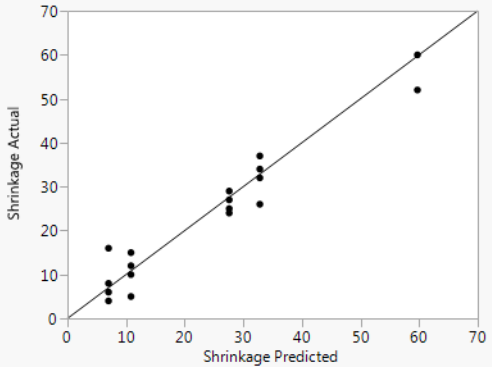
Variance Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq	Lower 95% Limit	Upper 95% Limit
Hold Time	1.5788108	0.412603	11.0480	0.0009*	0.7057476	2.37325
Residual	6.6912718	2.445524	7.4864	0.0062*	3.2689762	13.696373

Variance Effect Likelihood Ratio Tests

Source	Test Type	DF	ChiSquare	Prob>ChiSq
Hold Time	Likelihood	1	11.0480	0.0009*

Actual by Predicted Plot



The Mean Model for Shrinkage report gives the parameters for the mean model, and the Variance Model for Shrinkage report gives the parameters for the variance model.

The Loglinear Report

The top portion of the resulting report shows the fitting of the Expected response. "Description of the Fit Statistics Report" describes how the fit statistics are calculated. The Parameter Estimates and Fixed Effect Tests are similar to reports found in the standard least squares personality, though they are derived from restricted maximum likelihood (REML).

Figure 10.4 Mean Model Output

Loglinear Variance Fit

Fit Statistics

-2 Residual Log Likelihood	90.712809
-2 Log Likelihood	87.777035
AICc	106.23857
BIC	105.75143

Mean Model for Shrinkage

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	27.582516	0.380697	72.45	<.0001*
MoldTemp	7.683713	0.392907	19.56	<.0001*
Screw Speed	18.673515	0.392907	47.53	<.0001*
MoldTemp*Screw Speed	5.765297	0.392907	14.67	<.0001*

Fixed Effect Tests

Source	Nparm	DF	DFDen	F Ratio	Prob > F
MoldTemp	1	1	5.758	382.4387	<.0001*
Screw Speed	1	1	5.758	2258.768	<.0001*
MoldTemp*Screw Speed	1	1	5.758	215.3094	<.0001*

Figure 10.5 Variance Model Output

Variance Model for Shrinkage

Likelihood Ratio Test for Equal Variance

Source	-2 Residual Log Likelihood
Equal Variances Initially	101.76083063
After Fitted Variances	90.712808548
Difference: Chi-sq	11.048022086
Degrees of Freedom	1
Prob > ChiSquare	0.0008878186

Variance Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq	Lower 95% Limit	Upper 95% Limit
Hold Time	1.5788108	0.412603	11.0480	0.0009*	0.7057476	2.37325
Residual	6.6912718	2.445524	7.4864	0.0062*	3.2689762	13.696373

Variance Effect Likelihood Ratio Tests

Source	Test Type	DF	ChiSquare	Prob>ChiSq
Hold Time	Likelihood	1	11.0480	0.0009*

The second portion of the report shows the fit of the variance model. The **Variance Parameter Estimates** report shows the estimates and relevant statistics. Two hidden columns are provided:

- The hidden column **exp(Estimate)** is the exponential of the estimate. So, if the factors are coded to have +1 and -1 values, then the +1 level for a factor multiplies the variance by the **exp(Estimate)** value. Likewise, the -1 level multiplies the variance by the reciprocal of this

column. To see a hidden column, right-click the report and select the name of the column from the **Columns** menu that appears.

- The hidden column labeled **exp(2\|Estimate\|)**is the ratio of the higher to the lower variance if the regressor has the range -1 to +1.

The report also shows the standard error, chi-square, *p*-value, and profile likelihood confidence limits of each estimate. The residual parameter is the overall estimate of the variance, given all other regressors are zero.

Does the variance model fit significantly better than the original model? The likelihood ratio test for this question compares the fitted model with the model where all parameters are zero except the intercept, the model of equal-variance. In this case the *p*-value is highly significant. Changes in Hold Time change the variance.

The **Variance Effect Likelihood Ratio Tests** refit the model without each term in turn to create the likelihood ratio tests. These are generally more trusted than Wald tests.

Loglinear Platform Options

The Loglinear Variance Fit menu contains the following options:

Save Columns Creates one or more columns in the data table. See "Save Columns".

Row Diagnostics Plots row diagnostics. See "Row Diagnostics".

Profilers Opens the Profiler, Contour Profiler, or Surface Profiler. See "Factor Profiling".

Model Dialog Shows the completed launch window for the current analysis.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Save Columns

The following Save Columns options are available:

Prediction Formula Creates a new column called Mean. The new column contains the predicted values for the mean, as computed by the specified model.

Variance Formula Creates a new column called Variance. The new column contains the predicted values for the variance, as computed by the specified model.

Std Dev Formula Creates a new column called Std Dev. The new column contains the predicted values for the standard deviation, as computed by the specified model.

Residuals Creates a new column called Residual that contains the residuals, which are the observed response values minus predicted values. See "Examining the Residuals".

Studentized Residuals Creates a new column called Studentized Resid. The new column values are the residuals divided by their standard error.

Std Error of Predicted Creates a new column called Std Err Pred. The new column contains the standard errors of the predicted values.

Std Error of Individual Creates a new column called Std Err Indiv. The new column contains the standard errors of the individual predicted values.

Mean Confidence Interval Creates two new columns, Lower 95% Mean and Upper 95% Mean. The new columns contain the bounds for a confidence interval for the prediction mean.

Indiv Confidence Interval Creates two new columns, Lower 95% Indiv and Upper 95% Indiv. The new columns contain confidence limits for individual response values.

Row Diagnostics

The following Row Diagnostics options are available:

Plot Actual by Predicted Plots the observed values by the predicted values of *Y*. This is the leverage plot for the whole model.

Plot Studentized Residual by Predicted Plots the Studentized residuals by the predicted values of *Y*.

Plot Studentized Residual by Row Plots the Studentized residuals by row.

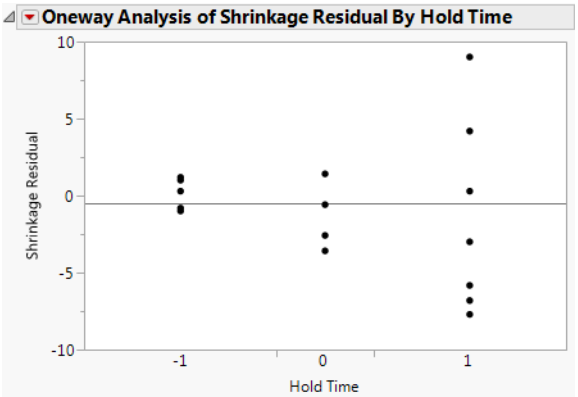
Examining the Residuals

To see the dispersion effect, follow these steps:

1. Select **Help > Sample Data Library** and open InjectionMolding.jmp.
2. Select **Analyze > Fit Model**.

   Since the variables in the data table have been assigned preselected roles, the analysis runs automatically.
3. Click the Loglinear Variance Fit red triangle and select **Save Columns > Residuals**.
4. In the InjectionMolding.jmp sample data table, right-click the continuous icon next to Hold Time in the Columns panel, and select **Nominal**.
5. Select **Analyze > Fit Y by X**.
6. Select Shrinkage Residual and click **Y, Response**.
7. Select Hold Time and click **X, Factor**.
8. Click **OK**.

Figure 10.6 Residual by Dispersion Effect



In this plot it is easy to see the variance go up as the Hold Time increases. This is done by treating Hold Time as a nominal factor.

Profiling the Fitted Model

Use the **Profiler**, **Contour Profiler**, or **Surface Profiler** to gain further insight into the fitted model. To select a profiler option, click the Loglinear Variance Fit red triangle and select one of the options under the **Profilers** menu.

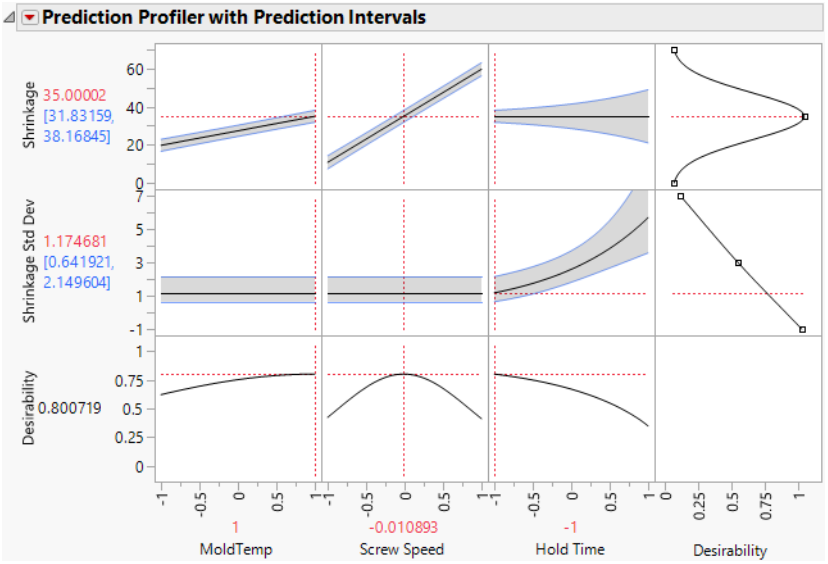
Example of Profiling the Fitted Model

For example, suppose that the goal was to find the factor settings that achieved a target of 35 for the response, but at the smallest variance. Fit the models and choose Profiler from the report menu. For example, Figure 10.7 shows the Profiler set up to match a target value for a mean and to minimize variance.

1. Select **Help > Sample Data Library** and open InjectionMolding.jmp.
2. Select **Analyze > Fit Model**.

   Since the variables in the data table have been assigned preselected roles, the analysis runs automatically.
3. Click the Loglinear Variance Fit red triangle and select **Profilers > Profiler**.
4. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Set Desirabilities**.
5. In the Response Goal window that appears, change Maximize to Match Target.
6. Click **OK**.
7. In the second Response Goal window, click **OK**.
8. Click the Prediction Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.
9. Click the Prediction Profiler red triangle and select **Prediction Intervals**.

Figure 10.7 Profiler to Match Target and Minimize Variance with Prediction Intervals



One of the best ways to see the relationship between the mean and the variance (both modeled with the LogVariance personality) is through looking at the individual prediction confidence intervals about the mean. Regular confidence intervals (those shown by default in the Profiler) do not show information about the variance model as well as individual prediction confidence intervals do. Prediction intervals show both the mean and variance model in one graph.

If Y is the modeled response, and you want a prediction interval for a new observation at x_n, then:

$$s^2\left|x_n = s^2_Y\right|x_n + s^2_{\hat{Y}}\left|x_n\right.$$

where:

$s^2\left|x_n\right.$ is the variance for the individual prediction at x_n

$s^2_Y\left|x_n\right.$ is the variance of the distribution of Y at x_n

$s^2_{\hat{Y}}\left|x_n\right.$ is the variance of the sampling distribution of $\hat{Y}$, and is also the variance for the mean.

Because the variance of the individual prediction contains the variance of the distribution of Y, the effects of the changing variance for Y can be seen. Not only are the individual prediction intervals wider, but they can change shape with a change in the variance effects.

Chapter 11

Logistic Regression Models

Fit Regression Models for Nominal or Ordinal Responses

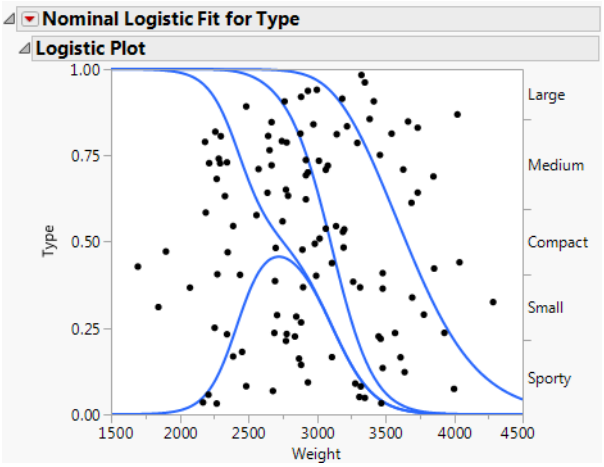
When your response variable has discrete values, you can use the Fit Model platform to fit a logistic regression model. The Fit Model platform provides two personalities for fitting logistic regression models. The personality that you use depends on the modeling type (Nominal or Ordinal) of your response column.

For nominal response variables, the Nominal Logistic personality fits a linear model to a multi-level logistic response function.

For ordinal response variables, the Ordinal Logistic personality fits the cumulative response probabilities to the logistic distribution function of a linear model.

Both personalities provide likelihood ratio tests for the model, a confusion matrix, and ROC and lift curves. When the response is binary, the Nominal Logistic personality also provides odds ratios (with corresponding confidence intervals).

Figure 11.1 Logistic Plot for a Nominal Logistic Regression Model



Contents

Overview of the Nominal and Ordinal Logistic Personalities 479
Nominal Logistic Regression 479
Ordinal Logistic Regression 479
Other JMP Platforms That Fit Logistic Regression Models 480
Examples of Logistic Regression 480
Example of Nominal Logistic Regression 480
Example of Ordinal Logistic Regression 482
Launch the Nominal and Ordinal Logistic Personalities 485
Validation in Logistic Regression Models 486
The Logistic Fit Report 486
Whole Model Test 488
Fit Details 489
Lack of Fit Test 490
Logistic Fit Platform Options 490
Options for Nominal and Ordinal Fits 490
Options for Nominal Fits 492
Options for Ordinal Fits 494
Additional Examples of Logistic Regression 495
Example of Inverse Prediction 495
Example of Using Effect Summary for a Nominal Logistic Model 497
Example of a Quadratic Ordinal Logistic Model 499
Example of Stacking Counts in Multiple Columns 502
Statistical Details for the Nominal and Ordinal Logistic Personalities 503
Logistic Regression Model 504
Odds Ratios 504
Relationship of Statistical Tests 505

Overview of the Nominal and Ordinal Logistic Personalities

Logistic regression models the probabilities of the levels of a categorical *Y* response variable as a function of one or more *X* effects. The Fit Model platform provides two personalities for fitting logistic regression models. The personality that you use depends on the modeling type (Nominal or Ordinal) of your response column.

For more information about fitting logistic regression models, see Walker and Duncan (1967), Nelson (1976), Harrell (1986), and McCullagh and Nelder (1989).

For more information about the parameterization of the logistic regression model, see "Logistic Regression Model".

Nominal Logistic Regression

When the response variable has a nominal modeling type, the platform fits a linear model to a multi-level logistic response function using maximum likelihood. Therefore, all but one response level is modeled by a logistic curve that represents the probability of the response level given the value of the *X* effects. The probability of the final response level is 1 minus the sum of the other fitted probabilities. As a result, at all values of the *X* effects, the fitted probabilities for the response levels sum to 1.

If the response variable is binary, you can set the Target Level in the Fit Model window to specify the level whose probability you want to model. By default, the model estimates the probability of the higher level of the response variable.

For more information about fitting models for nominal response variables, see "Nominal Responses".

Ordinal Logistic Regression

When the response variable has an ordinal modeling type, the platform fits the cumulative response probabilities to the logistic function of a linear model using maximum likelihood. Therefore, the cumulative probability of being at or below each response level is modeled by a curve. The curves are the same for each level except that they are shifted to the right or left.

Tip: If there are many response levels, the ordinal model is much faster to fit and uses less memory than the nominal model.

For more information about fitting models with ordinal response variables, see "Ordinal Responses".

Other JMP Platforms That Fit Logistic Regression Models

There are many other places in JMP where you can fit logistic regression models:

- To fit logistic regression models with a single continuous main effect, you can use the Fit Y by X platform to see a cumulative logistic probability plot for each effect. See *Basic Analysis*.
- To perform variable selection in logistic regression models, you can use the Stepwise personality of the Fit Model platform. See "Stepwise Regression Models".
- To fit logistic regression models that use a link function other than the Logit link, you can use the Generalized Linear Model personality of the Fit Model platform. See "Generalized Linear Models".
- JMP PRO To perform variable selection in logistic regression models and fit penalized logistic regression models, you can use the Generalized Regression personality of the Fit Model platform. See "Generalized Regression Models".

Examples of Logistic Regression

This section contains two examples, one for each of the logistic regression personalities in the Fit Model platform (Nominal and Ordinal):

- "Example of Nominal Logistic Regression"
- "Example of Ordinal Logistic Regression"

Example of Nominal Logistic Regression

An experiment was performed on metal ingots that were prepared with different heating and soaking times and then tested for readiness to roll. See Cox and Snell (1989). The data are contained in the Ingots.jmp sample data table. In this example, the Fit Model platform fits the probability of the Ready response using a logistic regression model with regressors heat and soak.

1. Select **Help > Sample Data Library** and open Ingots.jmp.

   The values of the categorical variable ready, Ready and Not Ready, indicate whether an ingot is ready to roll.
2. Select **Analyze > Fit Model**.
3. Select ready and click **Y**.

   Because you selected a column with the Nominal modeling type, the Fit Model Personality updates to Nominal Logistic.

Because ready is a Nominal column with only two levels, the Target Level option appears. This option enables you to specify the response level whose probability you want to model.

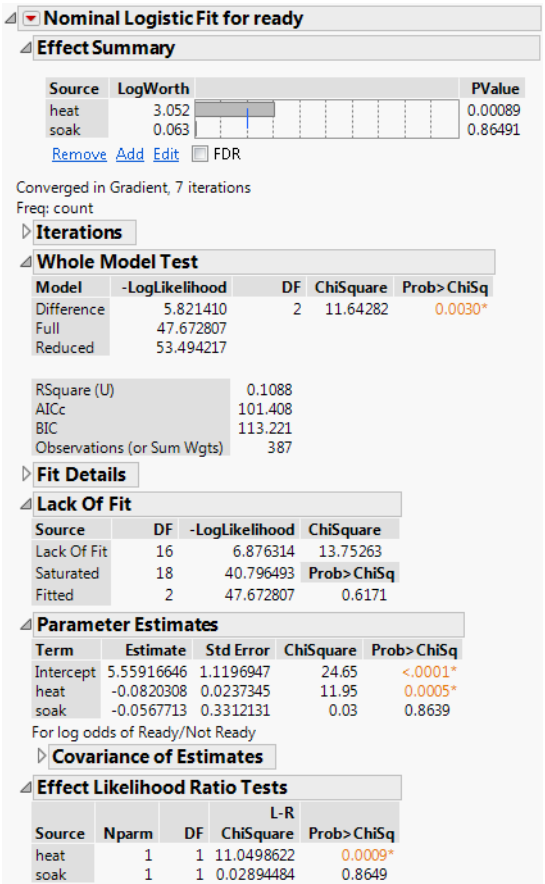
4. From the Target Level list, select Ready.

   In this model, the Target Level is Ready, so you are modeling the probability of the Ready response.

5. Select heat and soak and click **Add**.
6. Select count and click **Freq**.
7. Click **Run**.

   When the fitting process converges, the nominal regression report appears.

Figure 11.2 Nominal Logistic Fit Report



Nominal Logistic Fit for ready

Effect Summary

Source	LogWorth	PValue
heat	3.052	0.00089
soak	0.063	0.86491

Remove Add Edit FDR

Converged in Gradient, 7 iterations
Freq: count

Iterations

Whole Model Test

Model	-LogLikelihood	DF	ChiSquare	Prob>ChiSq
Difference	5.821410	2	11.64282	0.0030*
Full	47.672807			
Reduced	53.494217			

RSquare (U)	0.1088
AICc	101.408
BIC	113.221
Observations (or Sum Wgts)	387

Fit Details

Lack Of Fit

Source	DF	-LogLikelihood	ChiSquare
Lack Of Fit	16	6.876314	13.75263
Saturated	18	40.796493	Prob>ChiSq
Fitted	2	47.672807	0.6171

Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept	5.55916646	1.1196947	24.65	<.0001*
heat	-0.0820308	0.0237345	11.95	0.0005*
soak	-0.0567713	0.3312131	0.03	0.8639

For log odds of Ready/Not Ready

Covariance of Estimates

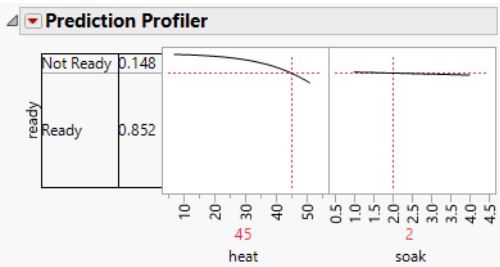
Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
heat	1	1	11.0498622	0.0009*
soak	1	1	0.02894484	0.8649

In the Whole Model Test report, the chi-square statistic (11.64) has a small *p*-value (0.0030), which indicates that the overall model is significant. However, the parameter estimate for soak has a *p*-value of 0.8639, which indicates that soaking time is not statistically significant.

8. Click the red triangle next to Nominal Logistic Fit for ready and select **Profiler**.

Figure 11.3 Prediction Profiler



When heat is set at 45 and soak is set at 2 the probability of ready is 0.85.

At this point, you might also be interested in using inverse prediction to find the heating time at a specific soaking time and given a particular probability of readiness to roll. See "Example of Inverse Prediction" for a continuation of this example.

Example of Ordinal Logistic Regression

An experiment was conducted to test whether various cheese additives (A to D) had an effect on cheese taste. Taste was measured by a tasting panel and recorded on an ordinal scale from 1 (strong dislike) to 9 (excellent taste). See McCullagh and Nelder (1989). The data are in the Cheese.jmp sample data table.

1. Select **Help > Sample Data Library** and open Cheese.jmp.
2. Select **Analyze > Fit Model**.
3. Select Response and click **Y**.

   Because you selected a column with the Ordinal modeling type, the Fit Model Personality updates to Ordinal Logistic.
4. Select Cheese and click **Add**.
5. Select Count and click **Freq**.
6. Click **Run**.

Figure 11.4 Ordinal Logistic Fit Report

Ordinal Logistic Fit for Response

Freq: Count

Whole Model Test

Model	-LogLikelihood	DF	ChiSquare	Prob>ChiSq
Difference	74.22695	3	148.4539	<.0001*
Full	355.67395			
Reduced	429.90090			

RSquare (U)	0.1727
AICc	734.695
BIC	770.061
Observations (or Sum Wgts)	208

Fit Details

Lack Of Fit

Source	DF	-LogLikelihood	ChiSquare
Lack Of Fit	21	10.15410	20.30819
Saturated	24	345.51986	Prob>ChiSq
Fitted	3	355.67395	0.5018

Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept[1]	-4.6051335	0.4267239	116.46	<.0001*
Intercept[2]	-3.5499488	0.3073868	133.37	<.0001*
Intercept[3]	-2.4503882	0.2398702	104.36	<.0001*
Intercept[4]	-1.381774	0.2001699	47.65	<.0001*
Intercept[5]	-0.0455233	0.1803299	0.06	0.8007
Intercept[6]	0.90648953	0.1886609	23.09	<.0001*
Intercept[7]	2.408151	0.2380517	102.34	<.0001*
Intercept[8]	3.96800057	0.3466551	131.02	<.0001*
Cheese[A]	-0.8622328	0.2289236	14.19	0.0002*
Cheese[B]	2.48960592	0.2703151	84.82	<.0001*
Cheese[C]	0.8476542	0.2280359	13.82	0.0002*

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
Cheese	3	3	148.453899	<.0001*

The model fit in this example reduces the –LogLikelihood of 429.9 for the intercept-only model to 355.67 for the full model. This reduction yields a likelihood ratio chi-square statistic for the whole model of 148.45 with 3 degrees of freedom. Therefore, the difference in perceived cheese taste is highly significant.

The most preferred cheese additive is the one with the most negative parameter estimate. Cheese[D] does not appear in the Parameter Estimates report, because it does not have its own column of the design matrix. However, Cheese D's effect can be computed as the negative sum of the others, and is shown in Table 11.1.

Table 11.1 Preferences for Cheese Additives in Cheese.jmp

Cheese	Estimate	Preference
A	–0.8622	2nd place
B	2.4896	least liked
C	0.8477	3rd place

Table 11.1 Preferences for Cheese Additives in Cheese.jmp *(Continued)*

D	–2.4750	most liked

Comparison to Nominal Logistic Model

The Lack of Fit report shows a test of whether the model fits the data well.

As an ordinal problem, each of the first eight response levels has an intercept, but there are only three parameters for the four levels of Cheese. As a result, there are 3 degrees of freedom in the ordinal model. The ordinal model is the Fitted model in the Lack of Fit test.

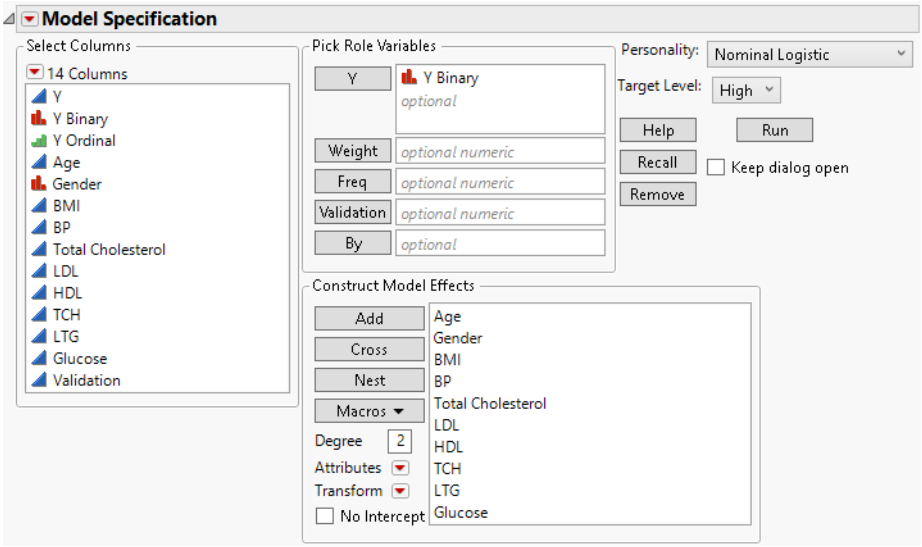
As a nominal problem, each of the first eight response levels has an intercept as well as three parameters for the four levels of Cheese. As a result, there are 8 x 3 = 24 degrees of freedom in the nominal model. Therefore, the nominal model is the Saturated model in the Lack of Fit test.

In this example, the Lack of Fit test for the ordinal model happens to be testing the ordinal response model against the nominal model. The nonsignificance of Lack of Fit leads one to believe that the ordinal model is reasonable.

Launch the Nominal and Ordinal Logistic Personalities

Launch the Nominal Logistic and Ordinal Logistic personalities by selecting **Analyze > Fit Model** and entering one or more non-continuous columns for **Y**. If multiple columns are entered for **Y**, a model for each response is fit.

Figure 11.5 Fit Model Launch Window with Nominal Logistic Selected



For more information about aspects of the Fit Model window that are common to all personalities, see "Model Specification". For more information about the options in the Select Columns red triangle menu, see *Using JMP*. Information specific to the Nominal Logistic and Ordinal Logistic personalities is presented here.

If your model effects have missing values, you can treat these missing values as informative categories. Select the Informative Missing option from the Model Specification red triangle menu.

To specify a model without an intercept term, select the No Intercept option in the Construct Model Effects panel of the Fit Model window. The No Intercept option is not available for the Ordinal Logistic personality.

The event of interest in the logistic regression model is defined by the Target Level option in the Fit Model window. This option is available only when you specify a binary response column in the Nominal Logistic personality.

Note: The Logistic personalities in the Fit Model platform require that your data be in a stacked format such that all of the responses are in one column. Sometimes, your data are formatted in multiple columns. See "Example of Stacking Counts in Multiple Columns" for an example of converting responses in multiple columns into a single column response.

JMP PRO Validation in Logistic Regression Models

Validation is the process of using part of a data set to estimate model parameters, and using the other part to assess the predictive ability of the model.

- The *training* set is used to estimate model parameters.
- The *validation* set is used in the model fitting to assess or validate the predictive ability of the model.
- The *test* set is a final, independent assessment of the model's predictive ability. The test set is available only when using a validation column.

The training, validation, and test sets are created as subsets of the original data. This is done through the use of a validation column in the Fit Model launch window.

The validation column's values determine how the data is split, and what method is used for validation:

- If the column has two distinct values, then training and validation sets are created.
- If the column has three distinct values, then training, validation, and test sets are created.
- If the column has more than three distinct values, or only one, then no validation is performed.

When a validation column is used, model fit statistics are given for the training, validation, and test sets in the Fit Details report. There is also a separate ROC curve, lift curve, and confusion matrix for each of the Training, Validation, and Test sets.

For more information about how a Validation column is used in JMP modeling platforms, see *Predictive and Specialized Modeling*.

The Logistic Fit Report

When you fit a model using the Nominal Logistic or Ordinal Logistic personality, you obtain a Nominal Logistic Fit report or an Ordinal Logistic Fit report, respectively. By default, these reports contain the following reports:

Effect Summary An interactive report that enables you to add or remove effects from the model. See "Effect Summary Report".

Logistic Plot (Available only if the model consists of a single continuous effect.) The logistic probability plot illustrates what the logistic model is fitting. At each value on the horizontal axis, the probability scale in the vertical direction is partitioned into probabilities for each response category. The probabilities are measured as the vertical distance between the curves, with the total across all response category probabilities summing to 1.

The points in the logistic plot represent the observations from the data table. The horizontal position of each point is determined by its value of continuous factor. The vertical position of each point is randomly chosen to be between curves that correspond to the value of its response category. Because a fixed random seed is used, the vertical positions do not differ across multiple fits of the same model.

Iterations (Available only in the Nominal Logistic personality.) After launching Fit Model, an iterative estimation process begins and is reported iteration by iteration. After the fitting process completes, you can open the Iteration History report and see the iteration steps.

Whole Model Test Shows tests that compare the whole-model fit to the model that omits all the regressor effects except the intercept parameters. The test is analogous to the Analysis of Variance table for continuous responses. For more information about the Whole Model Test report, see "Whole Model Test".

Fit Details Shows various measures of fit for the model. See "Fit Details".

Lack of Fit (Available only when there are replicated points with respect to the *X* effects and the model is not saturated.) Shows a lack of fit test, also called a goodness of fit test, that addresses whether more terms are needed in the model. See "Lack of Fit Test".

Parameter Estimates Shows the parameter estimates, standard errors, and associated hypothesis tests. The Covariance of Estimates report gives the variances and covariances of the parameter estimates.

Note: The Covariance of Estimates report appears only for nominal response variables, not for ordinal response variables.

Singularity Details (Appears only when there are linear dependencies among the model terms.) Shows a report that contains the linear functions that the model terms satisfy.

Effect Likelihood Ratio Tests The likelihood ratio chi-square tests are calculated as twice the difference of the log-likelihoods between the full model and the model constrained by the hypothesis to be tested. The constrained model is the model that does not contain the effect. These tests can take time to do because each test requires a separate set of iterations.

Note: Likelihood ratio tests are the platform default if they are projected to take less than 20 seconds to complete. Otherwise, the default effect tests are Wald tests.

Whole Model Test

The Whole Model Test table shows tests that compare the whole-model fit to the model that omits all the regression parameters except the intercept parameters. The test is analogous to the Analysis of Variance table for continuous responses. The negative log-likelihood corresponds to the sums of squares, and the chi-square test corresponds to the *F* test.

The Whole Model Test table shows these quantities:

Model Lists the model labels.

Difference The difference between the Full model and the Reduced model. This model is used to measure the significance of the regressors as a whole to the fit.

Full The complete model that includes the intercepts and all effects.

Reduced The model that includes only the intercept parameters.

–LogLikelihood The negative log-likelihood for the respective models. See "Likelihood, AICc, and BIC".

DF The degrees of freedom (DF) for the Difference between the Full and Reduced model.

Chi-Square The likelihood ratio chi-square test statistic for the hypothesis that all regression parameters are zero. The test statistic is computed by taking twice the difference in negative log-likelihoods between the fitted model and the reduced model that has only intercepts.

Prob>ChiSq The probability of obtaining a greater chi-square value if the specified model fits no better than the model that includes only intercepts.

RSquare (U) The proportion of the total uncertainty that is attributed to the model fit, defined as the ratio of the Difference to the Reduced negative log-likelihood values. RSquare ranges from zero for no improvement in fit to 1 for a perfect fit. An RSquare (U) value of 1 indicates that the predicted probabilities for events that occur are equal to one: There is no uncertainty in predicted probabilities. Because certainty in the predicted probabilities is rare for logistic models, RSquare (U) tends to be small.

RSquare (U) is sometimes referred to as *U*, the uncertainty coefficient, or as *McFadden's pseudo* R^2.

AICc The corrected Akaike Information Criterion. See "Likelihood, AICc, and BIC".

BIC Bayesian Information Criterion. See "Likelihood, AICc, and BIC".

Observations (or Sum Weights) Total number of observations in the sample. If a Freq or Weight column is specified in the Fit Model window, this value is the sum of the values of a column assigned to the Freq or Weight role.

Fit Details

The Fit Details report contains the following statistics:

Entropy RSquare Equivalent to RSquare (U). See "Whole Model Test".

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991).

Mean -Log p The average of -log(p), where p is the fitted probability associated with the event that occurred.

RASE The root average square error, where the differences are between the response and p (the fitted probability for the event that actually occurred).

Mean Abs Dev The average of the absolute values of the differences between the response and p (the fitted probability for the event that actually occurred).

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category.

N The number of observations.

For Entropy RSquare and Generalized RSquare, values closer to 1 indicate a better fit. For Mean -Log p, RASE, Mean Abs Dev, and Misclassification Rate, smaller values indicate a better fit.

To test that the effects as a whole are significant (the Whole Model test), a chi-square statistic is computed by taking twice the difference in negative log-likelihoods between the fitted model and the reduced model that has only intercepts.

If you specified a validation column, the Fit Details report contains columns for each of the Training, Validation, and Test sets.

Lack of Fit Test

The Lack of Fit test addresses whether there is enough information in the current model or whether more complex terms are needed. This test is sometimes called a goodness-of-fit test. The lack of fit test calculates a pure-error negative log-likelihood by constructing categories for every combination of the model effect values in the data. The Saturated row in the Lack Of Fit table contains this log-likelihood. The Lack of Fit report also contains a test of whether the Saturated log-likelihood is significantly better than the Fitted model.

The Saturated degrees of freedom is m–1, where m is the number of unique populations. The Fitted degrees of freedom is the number of parameters not including the intercept.

The Lack of Fit table contains the negative log-likelihood for error due to Lack of Fit, error in a Saturated model (pure error), and the total error in the Fitted model. The chi-square statistic tests for lack of fit.

Logistic Fit Platform Options

- "Options for Nominal and Ordinal Fits"
- "Options for Nominal Fits"
- "Options for Ordinal Fits"

Options for Nominal and Ordinal Fits

The following options are available in both the Nominal Logistic Fit and Ordinal Logistic Fit red triangle menus:

Logistic Plot (Available only if the model consists of a single continuous effect.) Shows or hides the Logistic Plot report. See "The Logistic Fit Report".

Likelihood Ratio Tests Shows or hides the Effect Likelihood Ratio Tests report. The likelihood ratio chi-square tests are calculated as twice the difference of the log-likelihoods between the full model and the model constrained by the hypothesis to be tested. The constrained model is the model that does not contain the effect). These tests can take time to do because each test requires a separate set of iterations. Therefore, they could take a long time for large problems.

Note: Likelihood ratio tests are the platform default if they are projected to take less than 20 seconds to complete. This default option is highly recommended.

Wald Tests Shows or hides the Effect Wald Tests report. The Wald chi-square is a quadratic approximation to the likelihood ratio test, and it is a by-product of the calculations.

Though Wald tests are considered less trustworthy, they do provide an adequate significance indicator for screening effects. Each parameter estimate and effect is shown with a Wald test. This is the default test if the likelihood ratio tests are projected to take more than 20 seconds to complete.

Confidence Intervals Shows or hides profile-likelihood confidence intervals for the model parameters. You can change the confidence level by selecting Set Alpha Level in the Model Specification red triangle menu in the Fit Model window. Each confidence limit requires a set of iterations in the model fit and can take a long time to compute. Furthermore, the effort does not always succeed in finding limits.

ROC Curve Shows or hides an ROC curve for the model. Receiver Operating Characteristic (ROC) curves measure the sorting efficiency of the model's fitted probabilities to sort the response levels. ROC curves can also aid in setting criterion points in diagnostic tests. The higher the curve from the diagonal, the better the fit. An introduction to ROC curves is found in *Basic Analysis*.

If the logistic fit has more than two response levels, it produces a generalized ROC curve (identical to the one in the Partition platform). In such a plot, there is a curve for each response level, which is the ROC curve of that level versus all other levels. See *Predictive and Specialized Modeling*.

If you specified a validation column, an ROC curve is shown for each of the Training, Validation, and Test sets.

Lift Curve Shows or hides a lift curve for the model. A lift curve shows the same information as an ROC curve, but in a way to dramatize the richness of the ordering at the beginning. The vertical axis shows the ratio of how rich that portion of the population is in the chosen response level compared to the rate of that response level as a whole. If you specified a validation column, a lift curve is shown for each of the Training, Validation, and Test sets. See *Predictive and Specialized Modeling* for more information about lift curves.

Confusion Matrix Shows or hides a report of confusion statistics, which contains a Confusion Matrix report and a Confusion Rates report. Both reports are two-way classifications of the actual response levels and the predicted response levels. The predicted response level is the Target Level specified in the launch window. The Confusion Rates report is equal to the Confusion Matrix report, with the numbers divided by the row totals.

For a good model, predicted response levels should be the same as the actual response levels. The Confusion Matrix report provides a way to assess how the predicted responses

align with the actual responses. If you specified a validation column, a confusion matrix is shown for each of the Training, Validation, and Test sets.

If the response is nominal and has a Profit Matrix column property, a Decision Matrix report also appears when this option is selected. For more information about the Decision Matrix report, see *Predictive and Specialized Modeling*.

Profiler Shows or hides the prediction profiler, showing the fitted values for a specified response probability as the values of the factors in the model are changed. This feature is available for both nominal and ordinal responses. For more information about the options in the red triangle menu, see *Profilers*.

Contour Profiler (Available only when the model contains more than one continuous factor.) Shows or hides the Contour Profiler. For more information about the options in the red triangle menu, see *Profilers*.

Model Dialog Shows the completed launch window for the current analysis.

Effect Summary Shows or hides the Effect Summary report, which enables you to interactively update the effects in the model. See "The Logistic Fit Report".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Options for Nominal Fits

The following options are available in the Nominal Logistic Fit red triangle menu:

Plot Options (Available only if the model consists of a single continuous effect.) The Plot Options menu contains the following options:

Show Points Toggles the points in the Logistic Plot on or off.

Show Rate Curve Enables you to compare the rate at various values of the effect variable with the fitted logistic curve. This option is useful only if you have several points for each value of the effect. In these cases, you get reasonable estimates of the rate at each value, and compare this rate with the fitted logistic curve. To prevent too many degenerate points, usually at zero or one, JMP shows only the rate value if there are at least three points at the *x*-value.

Line Color Specifies the color of the plot curves.

Odds Ratios Shows or hides an Odds Ratios report that contains Unit Odds Ratios and Range Odds Ratios.

Figure 11.6 Odds Ratios

Odds Ratios

For ready odds of Ready versus Not Ready

Unit Odds Ratios

Per unit change in regressor

Term	Odds Ratio	Lower 95%	Upper 95%	Reciprocal
heat	0.921244	0.87937	0.965111	1.0854892
soak	0.94481	0.493646	1.808313	1.0584137

Range Odds Ratios

Per change in regressor over entire range

Term	Odds Ratio	Lower 95%	Upper 95%	Reciprocal
heat	0.027069	0.003496	0.209605	36.942229
soak	0.8434	0.120295	5.913179	1.185677

Tests and confidence intervals on odds ratios are Wald based.

Inverse Prediction (Available only for two-level nominal responses.) Finds the *x* value that results in a specified probability. See the appendix "Statistical Details" for more information about inverse prediction.

Save Probability Formula Creates columns in the current data table that contain formulas for linear combinations of the response levels, prediction formulas for the response levels, and a prediction formula giving the most likely response.

For a nominal response model with *r* levels, JMP creates the following columns:

- columns called Lin[j] that contain a linear combination of the regressors for response levels $j = 1, 2, \ldots r - 1$
- a column called Prob[r], with a formula for the fit to the last level, *r*
- columns called Prob[j] for $j < r$ with a formula for the fit to level *j*
- a column called Most Likely response name that selects the most likely level of each row based on the computed probabilities.

JMP PRO **Publish Probability Formulas** Creates probability formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Indicator Parameterization Estimates (Available only when there are nominal columns among the model effects.) Shows or hides the Indicator Function Parameterization report, which gives parameter estimates for the model where nominal columns are coded using indicator (SAS GLM) parameterization and are treated as continuous. Ordinal columns remain coded using the usual JMP coding scheme. The SAS GLM and JMP coding schemes are described in "The Factor Models".

Caution: Standard errors and chi-square values given in the Indicator Function Parameterization report differ from those in the Parameter Estimates report. This is because the estimates are estimating different model parameters.

Options for Ordinal Fits

The following options are available in the Ordinal Logistic Fit red triangle menu:

Save Contains the following Save options:

Save Probability Formula Creates columns in the current data table that contain formulas for linear combinations of the response levels, prediction formulas for the response levels, and a prediction formula giving the most likely response.

For an ordinal response model with *r* levels, JMP creates the following columns:

- a column called Linear that contains the formula for a linear combination of the regressors without an intercept term
- columns called Cum[j] that each contain a formula for the cumulative probability that the response is less than or equal to level *j*, for levels *j* = 1, 2, ... *r* - 1. There is no Cum[j = 1, 2, ... r - 1] that is 1 for all rows
- columns called Prob[j = 1, 2, ... r - 1], for 1 < *j* < *r* that each contain the formula for the probability that the response is level *j*. Prob[j] is the difference between Cum[j] and Cum[j – 1]. Prob[1] is Cum[1], and Prob[r] is 1 – Cum[r – 1].
- a column called Most Likely response name that selects the most likely level of each row based on the computed probabilities.

JMP PRO **Publish Probability Formulas** Creates probability formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Save Quantiles (Available only when the response is numeric and has the ordinal modeling type.) Creates columns in the current data table named OrdQ.05, OrdQ.50, and OrdQ.95 that fit the quantiles for these three probabilities.

Save Expected Value (Available only when the response is numeric and has the ordinal modeling type.) Creates a column in the current data table called Ord Expected. This column contains the linear combination of the response values with the fitted response probabilities for each row and gives the expected value.

Additional Examples of Logistic Regression

This section contains additional examples using logistic regression in the Fit Model platform.

- "Example of Inverse Prediction"
- "Example of Using Effect Summary for a Nominal Logistic Model"
- "Example of a Quadratic Ordinal Logistic Model"
- "Example of Stacking Counts in Multiple Columns"

Example of Inverse Prediction

Inverse prediction enables you to predict the value for one independent variable at a specified probability of the response. If there is more than one independent variable, you must specify values for all of them except the one you are predicting.

An experiment was performed on metal ingots. The ingots were prepared with different heating and soaking times and then tested for readiness to roll. See Cox and Snell (1989). In JMP, the data are contained in the Ingots.jmp sample data table.

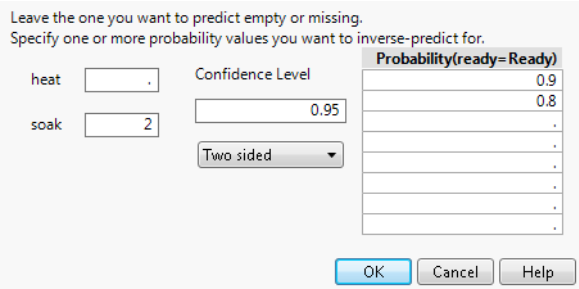
You are interested in predicting the heating time at a soaking time of 2 for probabilities of readiness to roll of 0.8 and 0.9.

1. For the analysis, follow the steps in "Example of Nominal Logistic Regression".
2. Click the red triangle next to Nominal Logistic Fit for ready and select **Inverse Prediction**.
3. Delete the value for heat.

   You want to find the predicted value of heat, so you leave the heat value empty.
4. Enter 2 for soak.

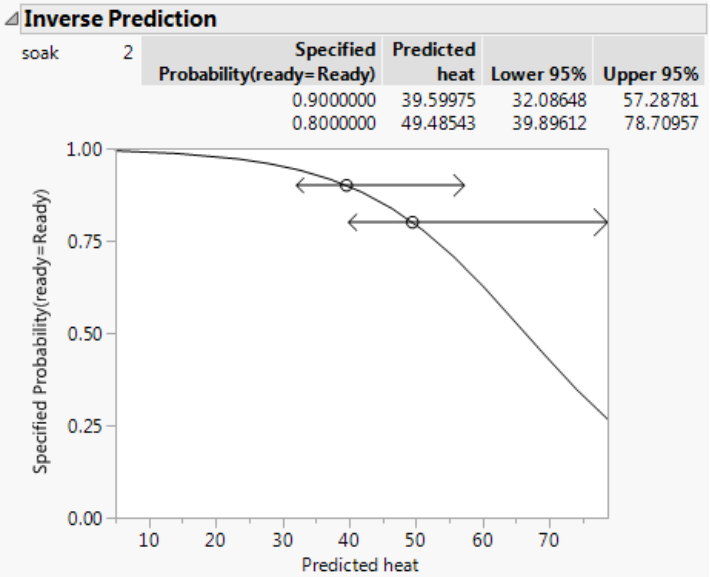
   You want to predict heating time when soak is 2.
5. Under Probability, enter 0.9 and 0.8 in the first two rows.

Figure 11.7 The Inverse Prediction Specification Window



6. Click **OK**.

Figure 11.8 Inverse Prediction Report



When soak is 2, the predicted value of heat for which there is a 90% chance of an ingot being ready to roll is 39.60 with a 95% confidence interval from 32.09 to 57.29. When soak is 2, the predicted value for heat for which there is an 80% chance of an ingot being ready to roll is 49.49 with a 95% confidence interval from 39.90 to 78.71.

Example of Using Effect Summary for a Nominal Logistic Model

A market research study was undertaken to evaluate preference for a brand of detergent. See Ries and Smith (1963). The results of the study are in the Detergent.jmp sample data table. The model is defined by the following:

- the response variable, brand, with values m and x
- an effect called softness (water softness) with values soft, medium, and hard
- an effect called previous use with values yes and no
- an effect called temperature with values high and low
- a count variable, count, which gives the frequency counts for each combination of effect categories

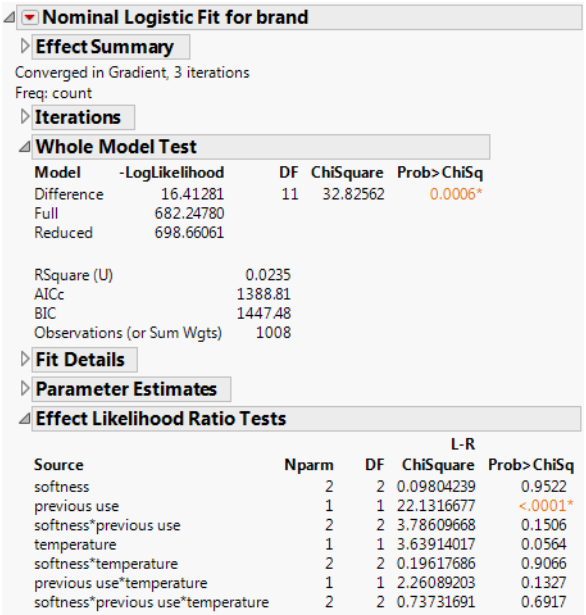
The study begins by specifying the full three-factor factorial model.

1. Select **Help > Sample Data Library** and open Detergent.jmp.
2. Select **Analyze > Fit Model**.
3. Select brand from the Select Columns list and click **Y**.

   Because you specified a nominal response variable, the Personality changes to Nominal Logistic.

   Because brand is a Nominal column with only two levels, the Target Level option appears. This option enables you to specify the response level whose probability you want to model.
4. From the Target Level list, select m.
5. Select count and click **Freq**.
6. Select softness through temperature and click **Macros > Full Factorial**.
7. Click **Run**.

Figure 11.9 Nominal Logistic Fit for Three-Factor Factorial Model



Nominal Logistic Fit for brand

Effect Summary

Converged in Gradient, 3 iterations
Freq: count

Iterations

Whole Model Test

Model	-LogLikelihood	DF	ChiSquare	Prob>ChiSq
Difference	16.41281	11	32.82562	0.0006*
Full	682.24780			
Reduced	698.66061			

RSquare (U)	0.0235
AICc	1388.81
BIC	1447.48
Observations (or Sum Wgts)	1008

Fit Details

Parameter Estimates

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
softness	2	2	0.09804239	0.9522
previous use	1	1	22.1316677	<.0001*
softness*previous use	2	2	3.78609668	0.1506
temperature	1	1	3.63914017	0.0564
softness*temperature	2	2	0.19617686	0.9066
previous use*temperature	1	1	2.26089203	0.1327
softness*previous use*temperature	2	2	0.73731691	0.6917

The Whole Model Test report shows that the three-factor full factorial model as a whole is significant (Prob>ChiSq = 0.0006).

The Effect Likelihood Ratio Tests report shows that the effects that include softness do not contribute significantly to the model fit. This leads you to consider removing softness from the model. You can do this from the Effect Summary report.

8. In the Effect Summary report, select softness*previous use through softness under Source and click **Remove**.

   The report updates to show the two-factor factorial model (Figure 11.10). The Whole Model Test report shows that the two-factor model is also significant as a whole.

Figure 11.10 Nominal Logistic Fit for Two-Factor Factorial Model

Nominal Logistic Fit for brand

Effect Summary

Converged in Gradient, 3 iterations
Freq: count

Iterations

Whole Model Test

Model	-LogLikelihood	DF	ChiSquare	Prob>ChiSq
Difference	13.58479	3	27.16958	<.0001*
Full	685.07582			
Reduced	698.66061			

RSquare (U)	0.0194
AICc	1378.19
BIC	1397.81
Observations (or Sum Wgts)	1008

Fit Details

Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept	0.03043265	0.0666827	0.21	0.6481
previous use[no]	-0.3148498	0.0666827	22.29	<.0001*
temperature[low]	-0.1331196	0.0666827	3.99	0.0459*
previous use[no]*temperature[low]	0.11106673	0.0666827	2.77	0.0958

For log odds of m/x

Covariance of Estimates

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
previous use	1	1	22.6511053	<.0001*
temperature	1	1	3.99625478	0.0456*
previous use*temperature	1	1	2.78794934	0.0950

You conclude that previous use of a detergent brand and water temperature have an effect on detergent preference. You also note that the interaction between temperature and previous use is not statistically significant, so there is no evidence that temperature depends on previous use.

Example of a Quadratic Ordinal Logistic Model

The Ordinal Logistic Personality can fit a quadratic surface to optimize the probabilities of higher or lower levels of an ordinal response.

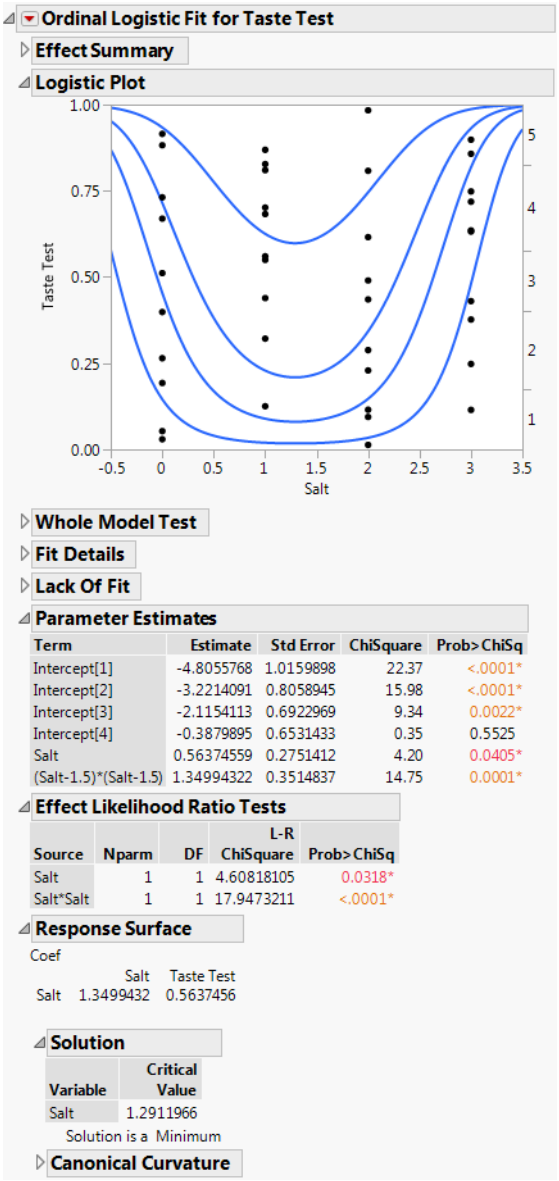
In this example, a microwave popcorn manufacturer wants to find out how much salt consumers like in their popcorn. To do this, the manufacturer looks for the maximum probability of a favorable response as a function of how much salt is added to the popcorn package. In this experiment, the salt amounts are controlled at 0, 1, 2, and 3 teaspoons. Respondents rate the taste on a scale of 1 (low) to 5 (high). The optimal amount of salt is the amount that maximizes the probability of more favorable responses. The ten observations for each of the salt levels are shown in Table 11.2.

Table 11.2 Salt in Popcorn

Salt Amount	Salt Rating Responses
no salt	1, 3, 2, 4, 2, 2, 1, 4, 3, 4
1 tsp.	4, 5, 3, 4, 5, 4, 5, 5, 4, 5
2 tsp.	4, 3, 5, 1, 4, 2, 5, 4, 3, 2
3 tsp.	3, 1, 2, 3, 1, 2, 1, 2, 1, 2

1. Select **Help > Sample Data Library** and open Salt in Popcorn.jmp.
2. Select **Analyze > Fit Model**.
3. Select Taste Test from the Select Columns list and click **Y**.
4. Select Salt from the Select Columns list and click **Macros > Response Surface**.
5. Click **Run**.
6. Click the disclosure icon next to Response Surface to open the report.

Figure 11.11 Ordinal Logistic Fit for Salt in Popcorn.jmp



The report shows how the quadratic model fits the response probabilities. The curves, instead of being shifted logistic curves, become stacked U-shaped curves where each curve achieves its minimum at the same point. The critical value is at Mean(X) – 0.5 * (b1/b2) where b1 is the linear coefficient and b2 is the quadratic coefficient. This formula is for centered X. From the Parameter Estimates table, you can compute that the optimal amount of salt is 1.5 - 0.5 * (0.5637/1.3499) = 1.29 teaspoons.

The vertical distance at a specific amount of salt between each curve measures the probability of each of the five response levels for the specific amount of salt. The probability for the highest response level is the distance from the top curve to the top of the plot rectangle. This distance reaches a maximum when the amount of salt is about 1.3 teaspoons. All curves share the same critical point.

The parameter estimates for Salt and Salt*Salt are the coefficients used to find the critical value. Although the critical value appears on the logistic plot as a minimum, it is a maximum in the sense of maximizing the probability of the highest response. The Solution portion of the report is shown under Response Surface in Figure 11.11, where 1.29 is shown under Critical Value.

Example of Stacking Counts in Multiple Columns

When data that are frequencies (counts) are listed in several columns of your data table, you must transform the data into the form that you need for logistic regression. For example, the Ingots2.jmp data table (Figure 11.12) has columns Nready and Nnotready. These columns give the number of ingots that are ready to roll and ingots that are not ready to roll for each combination of Heat and Soak values.

Figure 11.12 Ingots2.jmp Sample Data Table

	Heat	Soak	Nready	Nnotready	Ntotal	P	Loss
1	7	1	10	0	10	0.5000	6.9315
2	7	1.7	17	0	17	0.5000	11.7835
3	7	2.2	7	0	7	0.5000	4.8520
4	7	2.8	12	0	12	0.5000	8.3178
5	7	4	9	0	9	0.5000	6.2383
6	14	1	31	0	31	0.5000	21.4876
7	14	1.7	43	0	43	0.5000	29.8053
8	14	2.2	31	2	33	0.5000	22.8739
9	14	2.8	31	0	31	0.5000	21.4876
10	14	4	19	0	19	0.5000	13.1698
11	27	1	55	1	56	0.5000	38.8162
12	27	1.7	40	4	44	0.5000	30.4985
13	27	2.2	21	0	21	0.5000	14.5561
14	27	2.8	21	1	22	0.5000	15.2492
15	27	4	15	1	16	0.5000	11.0904

Before fitting a logistic regression model, use the following steps to stack the Nready and Nnotready columns into a single column:

1. Select **Help > Sample Data Library** and open Ingots2.jmp.
2. Select **Tables > Stack**.
3. Select Nready and NNotReady from the Select Columns list and click **Stack Columns**.

4. Click **OK**.

   This creates the new table in Figure 11.13. Label is the response (*Y*) column and Data is the frequency column.

This stacked data table is equivalent to the Ingots.jmp sample data table used in "Example of Nominal Logistic Regression".

Figure 11.13 Stacked Data Table

	Heat	Soak	Ntotal	P	Loss	Label	Data
1	7	1	10	0.5000	6.9315	Nready	10
2	7	1	10	0.5000	6.9315	Nnotready	0
3	7	1.7	17	0.5000	11.7835	Nready	17
4	7	1.7	17	0.5000	11.7835	Nnotready	0
5	7	2.2	7	0.5000	4.8520	Nready	7
6	7	2.2	7	0.5000	4.8520	Nnotready	0
7	7	2.8	12	0.5000	8.3178	Nready	12
8	7	2.8	12	0.5000	8.3178	Nnotready	0
9	7	4	9	0.5000	6.2383	Nready	9
10	7	4	9	0.5000	6.2383	Nnotready	0
11	14	1	31	0.5000	21.4876	Nready	31
12	14	1	31	0.5000	21.4876	Nnotready	0
13	14	1.7	43	0.5000	29.8053	Nready	43
14	14	1.7	43	0.5000	29.8053	Nnotready	0
15	14	2.2	33	0.5000	22.8739	Nready	31
16	14	2.2	33	0.5000	22.8739	Nnotready	2
17	14	2.8	31	0.5000	21.4876	Nready	31
18	14	2.8	31	0.5000	21.4876	Nnotready	0

Statistical Details for the Nominal and Ordinal Logistic Personalities

- "Logistic Regression Model"
- "Odds Ratios"
- "Relationship of Statistical Tests"

Logistic Regression Model

Logistic regression fits nominal Y responses to a linear model of X terms. To be more precise, it fits probabilities for the response levels using a logistic function. For two response levels, the function is:

$$P(Y = r_1) = (1 + e^{-Xb})^{-1} \quad \text{where } r_1 \text{ is the first response level}$$

or equivalently:

$$\log\left(\frac{P(Y = r_1)}{P(Y = r_2)}\right) = Xb \quad \text{where } r_1 \text{ and } r_2 \text{ are the two responses levels, respectively}$$

Note: When Y is binary and has a nominal modeling type, you can set the Target Level in the Fit Model window to specify the level whose probability you want to model. In this section, the target level is designated as r_1.

For r nominal response levels, where $r > 2$, the model is defined by $r - 1$ linear model parameters of the following form:

$$\log\left(\frac{P(Y = j)}{P(Y = r)}\right) = Xb_j$$

The fitting principal of maximum likelihood means that the βs are chosen to maximize the joint probability attributed by the model to the responses that did occur. This fitting principal is equivalent to minimizing the negative log-likelihood (–LogLikelihood):

$$\text{Loss} = -\text{logLikelihood} = \sum_{i=1}^{n} -\log(\text{Prob}(i\text{th row has the } y_j\text{th response}))$$

Odds Ratios

For two response levels, the logistic regression model is specified as follows:

$$\log\left(\frac{\text{Prob}(Y = r_1)}{\text{Prob}(Y = r_2)}\right) = Xb \quad \text{where } r_1 \text{ and } r_1 \text{ are the two response levels}$$

Therefore, the *odds* are defined as follows:

$$\frac{\text{Prob}(Y = r_1)}{\text{Prob}(Y = r_2)} = \exp(X\beta) = \exp(\beta_0) \cdot \exp(\beta_1 X_1) \cdots \exp(\beta_i X_i)$$

Note that $\exp(\beta_i(X_i + 1)) = \exp(\beta_i X_i) \exp(\beta_i)$. This shows that if X_i changes by a unit amount, the odds is multiplied by $\exp(\beta_i)$, which we label the *unit odds ratio*. As X_i changes over its whole range, the odds are multiplied by $\exp((X_{high} - X_{low})\beta_i)$, which we label the *range odds ratio*. For binary responses, the log odds ratio for flipped response levels involves only changing the sign of the parameter. Therefore, you might want the reciprocal of the reported value to focus on the last response level instead of the first.

Two-level nominal effects are coded 1 and -1 for the first and second levels, so range odds ratios or their reciprocals would be of interest.

If no confidence intervals for the parameter estimates have been calculated when you select the Odds Ratio option, selecting the Odds Ratio option produces Wald-based confidence intervals for the odds ratio.

Selecting the Odds Ratio option produces profile likelihood-based confidence intervals for the odds ratios when all of the following conditions are met:

- the Likelihood Ratio Tests option has been selected
- the Confidence Intervals option has been selected
- the number of parameters for each response is less than 8
- the number of rows is less than 1000.

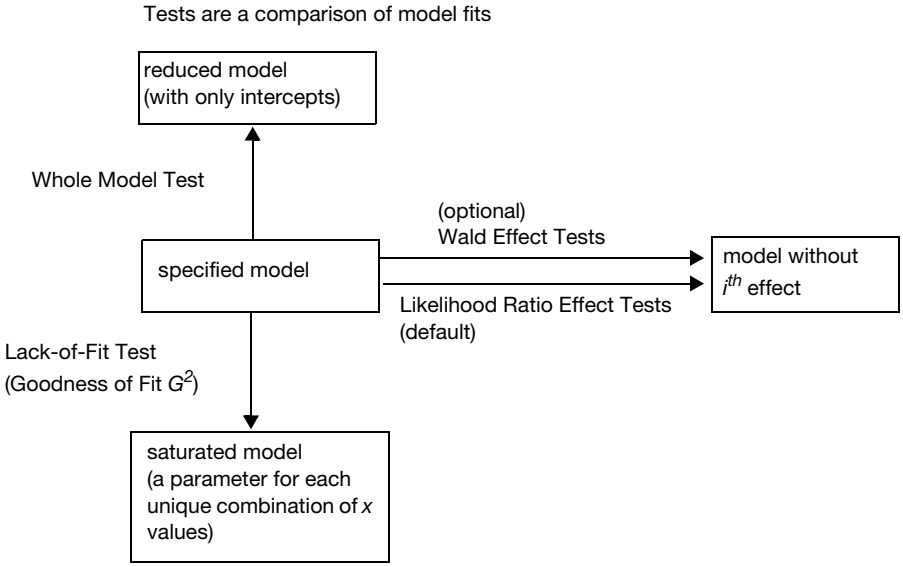
The method used for computing confidence intervals for the odds ratios is noted at the bottom of the Odds Ratios report.

Relationship of Statistical Tests

All of the statistical tests in the Logistic Regression reports compare the fit of the specified model with subset or superset models, as illustrated in Figure 11.14. If a test shows significance, then the higher order model is justified.

- Whole model tests: if the specified model is significantly better than a reduced model without any effects except the intercepts.
- Lack of Fit tests: if a saturated model is significantly better than the specified model.
- Effect tests: if the specified model is significantly better than a model without a given effect.

Figure 11.14 Relationship of Statistical Tests



Chapter 12

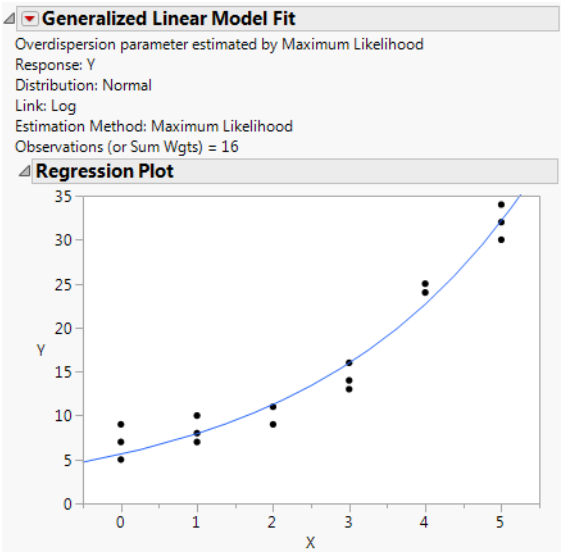
Generalized Linear Models

Fit Models for Nonnormal Response Distributions

Generalized linear models provide a unified way to fit responses that do not fit the usual requirements of traditional linear models. For example, frequency counts are often characterized as having a Poisson distribution and fit using a generalized linear model.

The Generalized Linear Model personality of the Fit Model platform enables you to fit generalized linear models for responses with binomial, normal, Poisson, or exponential distributions. The platform provides reports similar to those that are provided for traditional linear models. The platform also accommodates separation in logistic regression models using the Firth correction.

Figure 12.1 Example of a Generalized Linear Model Fit



Contents

Overview of the Generalized Linear Model Personality 509
Example of a Generalized Linear Model 510
Launch the Generalized Linear Model Personality 513
Generalized Linear Model Fit Report 515
Whole Model Test 516
Generalized Linear Model Fit Report Options 517
Additional Examples of the Generalized Linear Models Personality 520
Using Contrasts to Compare Differences in the Levels of a Variable 520
Poisson Regression with Offset 521
Normal Regression with a Log Link 522
Statistical Details for the Generalized Linear Model Personality 525
Model Selection and Deviance 527

Overview of the Generalized Linear Model Personality

Traditional linear models are used extensively in statistical data analysis. However, there are situations that violate the assumptions of traditional linear models. In these situations, traditional linear models are not appropriate. Traditional linear models assume that the responses are continuous and normally distributed with constant variance across all observations. These assumptions might not be reasonable. For example, these assumptions are not reasonable if you want to model counts, or if the variance of the observed responses increases as the response increases. Another example of violating the assumptions of traditional linear models is when the mean of the response is restricted to a specific range of values, such as proportions that fall between 0 and 1.

For situations such as these that fall into a wider range of data analysis problems, generalized linear models can be applied. Generalized linear models are an extension of traditional linear models. A generalized linear model consists of a linear component, a link function, and a variance function. The link function, $g(\mu_i) = x'_i\beta$, is a monotonic and differentiable function that describes how the expected value of Y_i is related to the linear predictors. An example of generalized linear regression is Poisson regression, where $\log(\mu_i)$ is the link function. For a complete list of the generalized linear regression models available using the Generalized Linear Models personality of the Fit Model platform, see "Statistical Details for the Generalized Linear Model Personality".

Fitted generalized linear models can be summarized and evaluated using the same statistics as traditional linear models. The Fit Model platform provides parameter estimates, standard errors, goodness-of-fit statistics, confidence intervals, and hypothesis tests for generalized linear models. It should be noted that exact distribution theory is not always available or practical for generalized linear models. Therefore, some inference procedures are based on asymptotic results.

An important aspect of fitting generalized linear models is the selection of explanatory variables in the model. Changes in goodness-of-fit statistics are often used to evaluate the contribution of subsets of explanatory variables to a particular model. The *deviance* is defined as twice the difference between the maximum attainable value of the log-likelihood function and the value of the log-likelihood function at the maximum likelihood estimates of the regression parameters. The deviance is often used as a measure of goodness of fit. The maximum attainable log-likelihood is achieved with a model that has a parameter for every observation.

JMP PRO For variable selection and penalized methods in generalized linear modeling, you can use the Generalized Regression personality of the Fit Model platform in JMP Pro. See "Generalized Regression Models".

Example of a Generalized Linear Model

This example uses Poisson regression to model count data from a study of nesting horseshoe crabs. Each female crab had a male crab resident in her nest. The study investigated whether there were other males, called satellites, residing nearby. The data table CrabSatellites.jmp contains a response variable listing the number of male satellites, as well as variables that describe the color, spine condition, weight, and carapace width of the female crab. You are interested in the relationship between the number of satellites and the variables that describe the female crabs.

To fit the Poisson regression model, follow these steps:

1. Select **Help > Sample Data Library** and open CrabSatellites.jmp.
2. Select **Analyze > Fit Model**.
3. Select satell and click **Y**.
4. Select color, spine, width, and weight and click **Add**.
5. From the Personality list, select **Generalized Linear Model**.
6. From the Distribution list, select **Poisson**.

   In the Link Function list, **Log** should be selected for you automatically.
7. Click **Run**.

Figure 12.2 Crab Satellite Results

Generalized Linear Model Fit

Response: satell
Distribution: Poisson
Link: Log
Estimation Method: Maximum Likelihood
Observations (or Sum Wgts) = 173

Whole Model Test

Model	-LogLikelihood	L-R ChiSquare	DF	Prob>ChiSq
Difference	41.6030357	83.2061	7	<.0001*
Full	452.44163			
Reduced	494.044666			

Goodness Of Fit Statistic	ChiSquare	DF	Prob>ChiSq
Pearson	533.8165	165	<.0001*
Deviance	549.5856	165	<.0001*

AICc
921.7613

Effect Tests

Source	DF	L-R ChiSquare	Prob>ChiSq
color	3	9.2405455	0.0263*
spine	2	1.794722	0.4076
width	1	0.1168819	0.7324
weight	1	9.0438942	0.0026*

Parameter Estimates

Term	Estimate	Std Error	L-R ChiSquare	Prob>ChiSq	Lower CL	Upper CL
Intercept	-0.710184	0.9339406	0.5761557	0.4478	-2.532002	1.1287428
color[Light Med]	0.3273541	0.132167	5.7773064	0.0162*	0.0619978	0.5810457
color[Medium]	0.0625029	0.0737267	0.7238945	0.3949	-0.080871	0.2083795
color[Dark Med]	-0.186351	0.0963747	3.8096887	0.0510	-0.377505	0.0007691
spine[Both Good]	0.0210298	0.0947416	0.049346	0.8242	-0.163737	0.2084429
spine[One Worn/Broken]	-0.129342	0.1313469	1.0163566	0.3134	-0.399791	0.1168764
width	0.0167487	0.0489197	0.1168819	0.7324	-0.07991	0.1118218
weight	0.0004965	0.0001663	9.0438942	0.0026*	0.000172	0.0008234

The Whole Model Test shows that the difference between log-likelihoods for the full and reduced models is 41.6. The Prob>ChiSq value is equivalent to the *p*-value for a whole model *F* test. A *p*-value less than 0.0001 indicates that the model as a whole is significant. The report also contains the corrected Akaike Information Criterion (AICc), 921.7613. This value can be compared with other models to determine the best-fitting model for the data. Smaller AICc values indicate better fitting models.

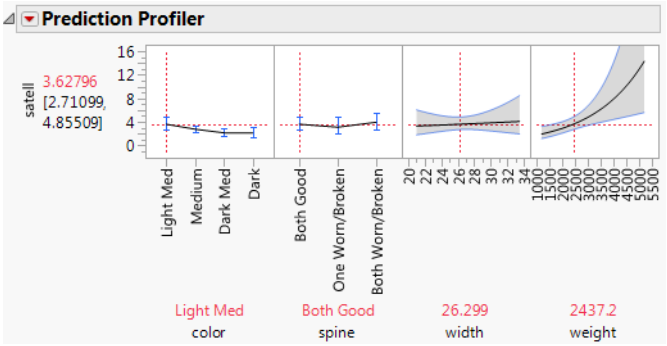
The Effects Tests report shows that weight is a significant factor in the model. Note that the *p*-value for weight, 0.0026, is the same in the Parameter Estimates table as well, since weight is a continuous variable.

The Effect Tests report also shows that the categorical variable color is significant. Color has four levels, Light Med, Medium, Dark Med, and Dark. The parameter estimates for the first three levels are shown in the Parameter Estimates table. The parameter estimate for Dark is the negative of the sum of the parameter estimates for the first three levels.

You can use the Prediction Profiler to further explore the model.

8. Click the red triangle next to Generalized Linear Model Fit and select **Profilers > Profiler**.

Figure 12.3 Prediction Profiler for Satell

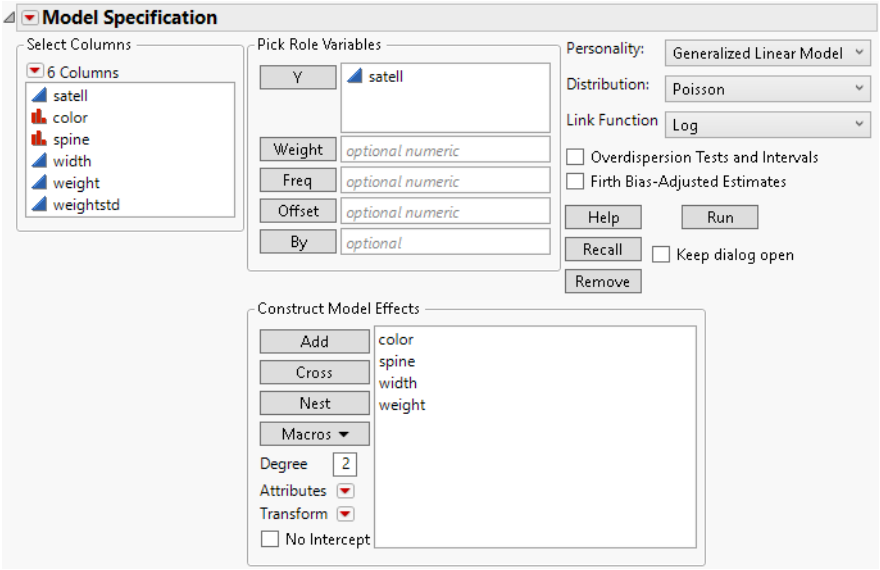


The confidence band on weight indicates that there is less variability in the model for smaller weight values than there is for larger weight values. The profiler enables you to easily explore the levels of the categorical variables. From the profiler, you can see that the predicted number of satellite crabs decreases as the color of the crab changes from light to dark.

Launch the Generalized Linear Model Personality

Launch the Generalized Linear Model personality by selecting **Analyze > Fit Model**, entering one or more columns for **Y**, and selecting **Generalized Linear Model** from the **Personality** menu.

Figure 12.4 Fit Model Launch Window with Generalized Linear Model Selected



For more information about aspects of the Fit Model window that are common to all personalities, see "Model Specification". For more information about the options in the Select Columns red triangle menu, see *Using JMP*. Information specific to the Generalized Linear Model personality is presented here.

If your model effects have missing values, you can treat these missing values as informative categories. Select the Informative Missing option from the Model Specification red triangle menu.

Tip: The No Intercept option is not available in the Generalized Linear Model personality of the Fit Model platform.

When you select Generalized Linear Model for Personality, the Fit Model launch window changes to include additional options. The following additional options are available in the Generalized Linear Model personality:

Distribution Specifies a probability distribution for the response variable.

Link Function Specifies a link function that relates the linear model to the response variable.

Overdispersion Tests and Intervals Specifies that an overdispersion parameter should be included in the model. Overdispersion occurs when the variance of the response is greater than would be expected by the theoretical variance of the response distribution. This can arise in Poisson and binomial response models. McCullagh and Nelder (1989) state that overdispersion is not uncommon in practice.

Note: This option adds a column for Overdispersion to the Goodness-of-Fit Statistics table in the Whole Model Test report.

Firth Bias-Adjusted Estimates Specifies that the Firth bias-adjusted method is used to fit the model. This maximum likelihood-based method has been shown to produce better estimates and tests than maximum likelihood-based models that do not use bias correction. In addition, bias-corrected MLEs ameliorate separation problems that tend to occur in logistic-type models. For more information about the separation problem in logistic regression, see Firth (1993) and Heinze and Schemper (2002).

Offset (Appears as a Pick Role Variables button.) Specifies a variable for the offset. An offset variable is one that is treated like a regression covariate whose parameter is fixed to be 1.0. Offset variables are most often used to scale the modeling of the mean in Poisson regression situations with a log link.

Response Specification for the Binomial Response Distribution

When you select Binomial as the Distribution, the response variable must be specified in one of the following ways:

- If your data are not summarized as frequencies of events, specify a single binary column as the response. The response column must be nominal.
- If your data are summarized as frequencies of events, specify a single binary column as the response and a frequency variable in the Freq role. The response column must be nominal, and the frequency variable contains the count of each response level.
- If your data are summarized as frequencies of events and number of trials, specify two continuous columns in this order: a count of the number of successes, and a count of the number of trials. Alternatively, you can specify the number of failures instead of the number of successes.

Generalized Linear Model Fit Report

By default, the Generalized Linear Model Fit report contains details about the model specification as well as the following reports:

Singularity Details (Appears only when there are linear dependencies among the model terms.) Shows a report that contains the linear functions that the model terms satisfy.

Regression Plot (Appears only when there is one continuous predictor and no more than one categorical predictor.) Shows a plot of the response on the vertical axis and the continuous predictor on the horizontal axis. A regression line is shown over the points. If there is a categorical predictor in the model, each level of the categorical predictor has a separate regression line and a legend appears next to the plot.

Whole Model Test Shows tests that compare the whole-model fit to the model that omits all the effects except the intercept parameters. This report also contains goodness-of-fit statistics and the corrected Akaike's Information Criterion (AICc) value. See "Whole Model Test".

Effect Summary An interactive report that enables you to add or remove effects from the model. See "Effect Summary Report".

Effect Tests The Effect Tests are joint tests that all the parameters for an individual effect are zero. If an effect has only one parameter, as with continuous effects, then the tests are the same as the tests in the Parameter Estimates table.

Note: Even if the Firth adjustment is used, the Effect Tests are based on the non-penalized likelihood function.

Parameter Estimates Shows the parameter estimates, standard errors, and associated hypothesis tests and confidence limits. Simple continuous effects have only one parameter. Models with complex classification effects have a parameter for each anticipated degree of freedom.

Note: If there are more than 1,000 observations, Wald-based confidence intervals are shown. Otherwise, profile-likelihood confidence intervals are shown.

Studentized Deviance Residual by Predicted Shows a plot of studentized deviance residuals on the vertical axis and the predicted response values on the horizontal axis.

Whole Model Test

The Whole Model Test table shows tests that compare the whole-model fit to the model that omits all the regression parameters except the intercept parameter. It also contains two goodness-of-fit statistics and the AICc value to assess model adequacy.

The Whole Model Test table shows these quantities:

Model Lists the model labels.

Difference The difference between the Full model and the Reduced model. This model is used to measure the significance of the regressors as a whole to the fit.

Full The complete model that includes the intercepts and all effects.

Reduced The model that includes only the intercept parameters.

–LogLikelihood The negative log-likelihood for the respective models. See "Likelihood, AICc, and BIC".

L-R ChiSquare The likelihood ratio chi-square test statistic for the hypothesis that all regression parameters are zero. The test statistic is computed by taking twice the difference in negative log-likelihoods between the fitted model and the reduced model that has only an intercept.

DF The degrees of freedom (DF) for the Difference between the Full and Reduced model.

Prob>ChiSq The probability of obtaining a greater chi-square value if the specified model fits no better than the model that includes only an intercept.

Goodness of Fit Statistic Lists the two goodness-of-fit statistics: Pearson and Deviance.

ChiSquare The chi-square test statistic for the respective goodness-of-fit statistics.

DF The degrees of freedom for the respective goodness-of-fit statistics.

Prob>ChiSq The *p*-value for the respective goodness-of-fit statistics.

Overdispersion (Appears only when the Overdispersion Tests and Intervals option is selected in the launch window.) An estimate of the overdispersion parameter. See "Statistical Details for the Generalized Linear Model Personality".

AICc The corrected Akaike Information Criterion. See "Likelihood, AICc, and BIC".

Generalized Linear Model Fit Report Options

The Generalized Linear Model Fit red triangle menu contains the following options:

Custom Test Enables you to test a custom hypothesis. For more information about custom tests, see "Custom Test".

Contrast Enables you to test for differences in levels within a variable. If a contrast involves a covariate, you can specify the value of the covariate at which to test the contrast. For an example of the Contrast option, see "Using Contrasts to Compare Differences in the Levels of a Variable".

Inverse Prediction (Available only for continuous *X* variables.) Enables you to predict an *X* value, given specific values for *Y* and the other *X* variables. For more information about the Inverse Prediction option, see "Inverse Prediction".

Covariance of Estimates Shows or hides a covariance matrix for all the effects in a model. The estimated covariance matrix of the parameter estimator is defined as follows:

$$\Sigma = -\mathbf{H}^{-1}$$

where **H** is the Hessian (or second derivative) matrix evaluated using the parameter estimates on the last iteration. Note that the dispersion parameter, whether estimated or specified, is incorporated into **H**. Rows and columns corresponding to aliased parameters are not included in Σ.

Correlation of Estimates Shows or hides a correlation matrix for all the effects in a model. The correlation matrix is the normalized covariance matrix. For each σ_{ij} element of Σ, the corresponding element of the correlation matrix is $\sigma_{ij}/\sigma_i\sigma_j$, where $\sigma_i = \sqrt{\sigma_{ii}}$.

Profilers Shows a submenu of the following profilers:

Profiler Shows or hides a prediction profiler for examining prediction traces for each X variable. For more information about the prediction profiler, see "Profiler".

Contour Profiler Shows or hides an interactive contour profiler. For more information about the contour profiler, see *Profilers*.

Surface Profiler Shows or hides a 3-D surface profiler. For more information about the surface profiler, see *Profilers*.

Diagnostic Plots Shows a submenu of plots of residuals, predicted values, and actual values. These plots enable you to search for outliers and determine the adequacy of your model. For more information about deviance, see "Model Selection and Deviance". The following plots are available:

Studentized Deviance Residuals by Predicted Shows or hides a plot of studentized deviance residuals on the vertical axis and the predicted response values on the horizontal axis.

Studentized Pearson Residuals by Predicted Shows or hides a plot of the studentized Pearson residuals on the vertical axis and the predicted response values on the horizontal axis.

Deviance Residuals by Predicted Shows or hides a plot of the deviance residuals on the vertical axis and the predicted response values on the horizontal axis.

Pearson Residuals by Predicted Shows or hides a plot of the Pearson residuals on the vertical axis and the predicted response values on the horizontal axis.

Regression Plot (Available only when there is one continuous predictor and no more than one categorical predictor.) Shows or hides a plot of the response on the vertical axis and the continuous predictor on the horizontal axis. A regression line is shown over the points. If there is a categorical predictor in the model, each level of the categorical predictor has a separate regression line and a legend appears next to the plot.

Linear Predictor Plot (Available only when there is one continuous predictor and no more than one categorical predictor.) Shows or hides a plot of responses transformed by the inverse link function on the vertical axis and the continuous predictor on the horizontal axis. A transformed regression line is shown over the points. If there is a categorical predictor in the model, each level of the categorical predictor has a separate transformed regression line and a legend appears next to the plot.

Save Columns Shows a submenu of options that enable you to save certain quantities as new columns in the current data table. For more information about the residual formulas, see "Residual Formulas".

Prediction Formula Creates a formula column in the current data table that predicts the model.

Predicted Values Creates a column in the current data table that contains the values predicted by the model.

Mean Confidence Interval Creates columns in the current data table that contain the 95% confidence limits for the prediction equation for the model. These confidence limits reflect the variation in the parameter estimates.

Note: You can change the α level for the confidence limits in the Fit Model window by selecting Set Alpha Level from the Model Specification red triangle menu.

Save Indiv Confid Limits Creates columns in the current data table that contain the 95% confidence limits for a given individual value for the model. These confidence limits reflect variation in the error and variation in the parameter estimates.

Note: You can change the α level for the confidence limits in the Fit Model window by selecting Set Alpha Level from the Model Specification red triangle menu.

Deviance Residuals Creates a column in the current data table that contains the deviance residuals.

Pearson Residuals Creates a column in the current data table that contains the Pearson residuals.

Studentized Deviance Residuals Creates a column in the current data table that contains the studentized deviance residuals.

Studentized Pearson Residuals Creates a column in the current data table that contains the studentized Pearson residuals.

Model Dialog Shows the completed launch window for the current analysis.

Effect Summary Shows or hides the Effect Summary report, which enables you to interactively update the effects in the model. See "Effect Summary Report".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Examples of the Generalized Linear Models Personality

- "Using Contrasts to Compare Differences in the Levels of a Variable"
- "Poisson Regression with Offset"
- "Normal Regression with a Log Link"

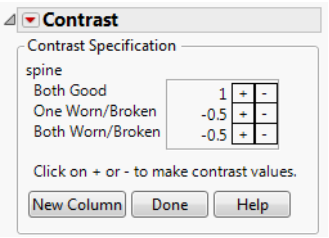
Using Contrasts to Compare Differences in the Levels of a Variable

This example continues the crab satellite example in "Example of a Generalized Linear Model". Suppose that you want to test whether female crabs with good spines attracted a different number of male crabs (satellites) than female crabs with worn or broken spines.

1. Complete step 1 through step 7 of "Example of a Generalized Linear Model".
2. Click the red triangle next to Generalized Linear Model Fit and select **Contrast**. The Choose effects for contrast window appears.
3. Select spine, the variable of interest, and click **Go**.
4. To compare the crabs with good spines to crabs with worn or broken spines, click the **+** button beside Both Good and the **-** button beside both One Worn/Broken and Both Worn/Broken.

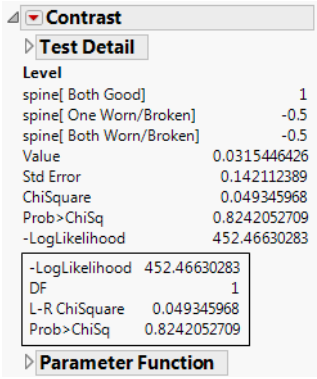
   This creates a contrast specification that compares the female crabs with good spines against the female crabs with worn or broken spines.

Figure 12.5 Contrast Specification Window



5. Click **Done**.

Figure 12.6 Contrast Report



The Prob>Chisq, 0.8242, is much greater than 0.05, so we cannot conclude that there is a difference in satellite attraction based on spine condition.

Poisson Regression with Offset

The sample data table Ship Damage.JMP is adapted from data found in McCullagh and Nelder (1989). The data table contains information about a certain type of damage caused by waves to the forward section of the hull. Hull construction engineers are interested in the risk of damage associated with three variables: ship type (Type), the year in which the ship was constructed (Yr Made), and the block of years the ship was in service (Yr Used).

In this analysis we use the variable Service, the log of the aggregate months of service, as an *offset variable*. An offset variable is one that is treated like a regression covariate whose parameter is fixed to be 1.0. Offset variables are most often used to scale the modeling of the mean in Poisson regression situations with a log link. In this example, we use log(months of service) since one would expect that the number of repairs be proportional to the number of months in service.

To see how an offset variable is used, assume the linear component of the GLM is called η. Then, with a log link function, the model for the mean with the offset included is specified as follows:

$$\exp[\text{Log}(\text{months of service}) + \eta] = [(\text{months of service}) * \exp(\eta)].$$

To run this example, follow these steps:

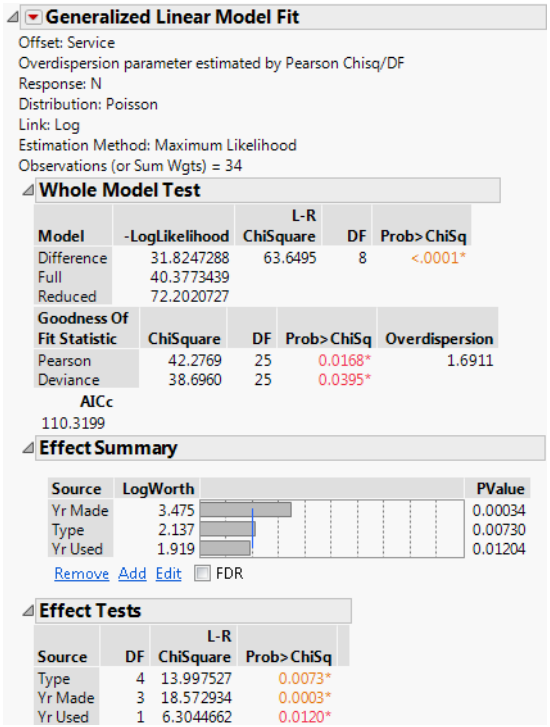
1. Select **Help > Sample Data Library** and open Ship Damage.jmp.
2. Select **Analyze > Fit Model**.
3. From the Personality list, select **Generalized Linear Model**.
4. From the Distribution list, select **Poisson**.

   In the Link Function list, **Log** should be selected for you automatically.

5. Select N and click **Y**.
6. Select Service and click **Offset**.
7. Select Type, Yr Made, Yr Used and click **Add**.
8. Click the check mark box for **Overdispersion Tests and Intervals**.
9. Click **Run**.

From the report, notice that all three effects (Type, Yr Made, Yr Used) are significant.

Figure 12.7 Partial Report for a Poisson with Offset Model



Generalized Linear Model Fit

Offset: Service
Overdispersion parameter estimated by Pearson Chisq/DF
Response: N
Distribution: Poisson
Link: Log
Estimation Method: Maximum Likelihood
Observations (or Sum Wgts) = 34

Whole Model Test

Model	-LogLikelihood	L-R ChiSquare	DF	Prob>ChiSq
Difference	31.8247288	63.6495	8	<.0001*
Full	40.3773439			
Reduced	72.2020727			

Goodness Of Fit Statistic	ChiSquare	DF	Prob>ChiSq	Overdispersion
Pearson	42.2769	25	0.0168*	1.6911
Deviance	38.6960	25	0.0395*	

AICc
110.3199

Effect Summary

Source	LogWorth	PValue
Yr Made	3.475	0.00034
Type	2.137	0.00730
Yr Used	1.919	0.01204

Remove Add Edit FDR

Effect Tests

Source	DF	L-R ChiSquare	Prob>ChiSq
Type	4	13.997527	0.0073*
Yr Made	3	18.572934	0.0003*
Yr Used	1	6.3044662	0.0120*

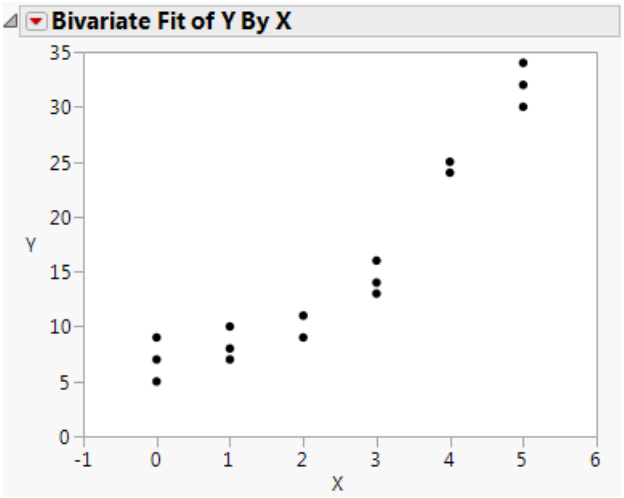
Normal Regression with a Log Link

This example uses the Nor.jmp data table, where X is an explanatory variable and Y is the response variable. First, you explore the relationship between X and Y to determine the appropriate link function to use in the Generalized Linear Model personality of the Fit Model platform.

1. Select **Help > Sample Data Library** and open Nor.jmp.
2. Select **Analyze > Fit Y By X**.
3. Select Y and click **Y, Response**.

4. Select X, click **X, Factor**, and then click **OK**.

Figure 12.8 Y by X Results for Nor.jmp



You can see that Y varies nonlinearly with X and that the variance is approximately constant. Therefore, a normal distribution with a log link function is appropriate to model these data; that is, $\log(\mu_i) = \mathbf{x}_i'\beta$ so that $\mu_i = \exp(\mathbf{x}_i'\beta)$.

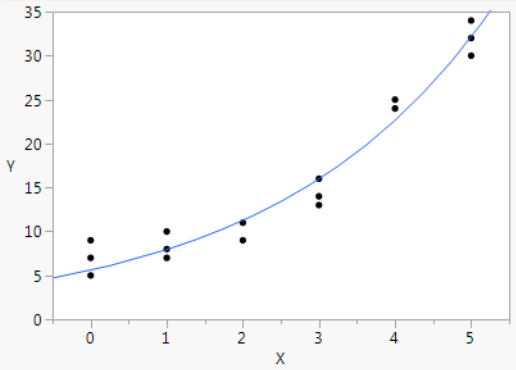
5. Select **Analyze > Fit Model**.
6. In the Personality list, select the **Generalized Linear Model**.
7. In the Distribution list, select **Normal**.
8. In the Link Function list, select **Log**.
9. Select Y and click **Y**.
10. Select X and click **Add**.
11. Click **Run**.

Figure 12.9 Nor Results

Generalized Linear Model Fit

Overdispersion parameter estimated by Maximum Likelihood
Response: Y
Distribution: Normal
Link: Log
Estimation Method: Maximum Likelihood
Observations (or Sum Wgts) = 16

Regression Plot



Whole Model Test

Model	-LogLikelihood	L-R ChiSquare	DF	Prob>ChiSq
Difference	26.5218281	53.0437	1	<.0001*
Full	32.1782768			
Reduced	58.7001049			

Goodness Of Fit Statistic	ChiSquare	DF	Prob>ChiSq	Overdispersion
Pearson	52.3000	14	<.0001*	3.2687
Deviance	52.3000	14	<.0001*	

AICc
72.3566

Effect Tests

Source	DF	L-R ChiSquare	Prob>ChiSq
X	1	53.043656	<.0001*

Parameter Estimates

Term	Estimate	Std Error	L-R ChiSquare	Prob>ChiSq	Lower CL	Upper CL
Intercept	1.7213577	0.0893972	37.973777	<.0001*	1.5239809	1.8978651
X	0.3496015	0.0206493	53.043656	<.0001*	0.3081539	0.3944836

Statistical Details for the Generalized Linear Model Personality

To construct a generalized linear model, you must select response and explanatory variables for your data. You then must choose an appropriate link function and probability distribution for your response. Explanatory variables can be any combination of continuous variables, classification variables, and interactions. Some common examples of generalized linear models are listed in Table 12.1.

Table 12.1 Examples of Generalized Linear Models

Model	Response Variable	Distribution	Default Link Function
Traditional Linear Model	continuous	Normal	identity, $g(\mu) = \mu$
Logistic Regression	a count or a binary random variable	Binomial	logit, $g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$
Poisson Regression in Log Linear Model	a count	Poisson	log, $g(\mu) = \log(\mu)$
Exponential Regression	positive continuous	Exponential	$\frac{1}{\mu}$

The platform fits a generalized linear model to the data by maximum likelihood estimation of the parameter vector. In general, there is no closed-form solution for the maximum likelihood estimates of the parameters. Therefore, the platform estimates the parameters of the model numerically through an iterative fitting process using a technique pioneered by Nelder and Wedderburn (1972). The overdispersion parameter ϕ is estimated by dividing the Pearson goodness-of-fit statistic by its degrees of freedom. Covariances, standard errors, and confidence limits are computed for the estimated parameters based on the asymptotic normality of maximum likelihood estimators.

A number of link functions and probability distributions are available in the Generalized Linear Model personality of the Fit Model platform. Table 12.2 lists the built-in link functions.

Table 12.2 Built-in Link Functions

Link Function Name	Link Function Formula
Identity	$g(\mu) = \mu$
Logit	$g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$
Probit	$g(\mu) = \Phi^{-1}(\mu)$, where Φ is the standard normal cumulative distribution function
Log	$g(\mu) = \log(\mu)$
Reciprocal	$g(\mu) = \frac{1}{\mu}$
Power	$g(\mu) = \begin{cases} \mu^{\lambda} & \text{if } (\lambda \neq 0) \\ \log(\mu) & \text{if } \lambda = 0 \end{cases}$
Comp LogLog	$g(m) = \log(-\log(1-\mu))$

When you select the Power link function, a number box appears that enables you to enter the desired power.

Table 12.3 lists the variance functions associated with the available distributions for the response variable.

Table 12.3 Variance Functions for Response Distributions

Distribution	Variance Function
Normal	$V(\mu) = 1$
Binomial	$V(\mu) = \mu(1-\mu)$
Poisson	$V(\mu) = \mu$
Exponential	$V(\mu) = \mu^2$

Model Selection and Deviance

An important aspect of generalized linear modeling is the selection of explanatory variables in the model. Changes in goodness-of-fit statistics are often used to evaluate the contribution of subsets of explanatory variables to a particular model. The *deviance* is defined to be twice the difference between the maximum attainable log-likelihood and the log-likelihood at the maximum likelihood estimates of the regression parameters. The deviance is often used as a measure of goodness of fit. The maximum attainable log-likelihood is achieved with a model that has a parameter for every observation. Table 12.4 lists the deviance formula for each of the available distributions for the response variable.

Table 12.4 Deviance Formulas for Response Distributions

Distribution	Deviance Formula
Normal	$\sum_i w_i(y_i - \mu_i)^2$
Binomial	$2\sum_i w_i m_i\left[y_i \log\left(\frac{y_i}{\mu_i}\right) + (1-y_i)\log\left(\frac{1-y_i}{1-\mu_i}\right)\right]$
Poisson	$2\sum_i w_i\left[y_i \log\left(\frac{y_i}{\mu_i}\right) - (y_i - \mu_i)\right]$
Exponential	$2\sum_i w_i\left[-\log\left(\frac{y_i}{\mu_i}\right) + \left(\frac{y_i - \mu_i}{\mu_i}\right)\right]$

The Pearson chi-square statistic is defined as follows:

$$X^2 = \sum_i \frac{w_i(y_i - \mu_i)^2}{V(\mu_i)}$$

where

y_i is the i^{th} response

μ_i is the corresponding predicted mean

$V(\mu_i)$ is the variance function

w_i is a known weight for the i^{th} observation

Note: If no weight is specified, $w_i = 1$ for all observations.

One strategy for variable selection is to fit a sequence of models. You start with a simple model that contains only an intercept term, and then include one additional explanatory variable in each successive model. You can measure the importance of the additional explanatory variable by the difference in deviance or fitted log-likelihood values between successive models. Asymptotic tests enable you to assess the statistical significance of the additional term.

When the distribution is non-normal, a normal critical value is used instead of a *t*-distribution critical value in inverse prediction.

Residual Formulas

Deviance

$$r_{Di} = \sqrt{d_i}(sign(y_i - \mu_i))$$

Studentized Deviance

$$r_{Di} = \frac{sign(y_i - \mu_i)\sqrt{d_i}}{\sqrt{\phi(1 - h_i)}}$$

Pearson

$$r_{Pi} = \frac{(y_i - \mu_i)}{\sqrt{V(\mu_i)}}$$

Studentized Pearson

$$r_{Pi} = \frac{y_i - \mu_i}{\sqrt{V(\mu_i)(1 - h_i)}}$$

where

$(y_i - \mu_i)$ is the raw residual

$\text{sign}(y_i - \mu_i)$ is 1 if $(y_i - \mu_i)$ is positive and -1 if $(y_i - \mu_i)$ is negative

d_i is the contribution to the total deviance from observation i

ϕ is the dispersion parameter

$V(\mu_i)$ is the variance function

h_i is the i^{th} diagonal element of the matrix $W_e^{(1/2)}X(X'W_eX)^{-1}X'W_e^{(1/2)}$, where W_e is the

weight matrix used in computing the expected information matrix.

For more information about residuals and generalized linear models, see the GENMOD Procedure chapter in SAS Institute Inc. (2020a).

Appendix A

Statistical Details

Fitting Linear Models

This appendix discusses the different types of response models, their factors, their design coding, and parameterization. It also includes many other details of methods described in the main text.

The JMP system fits linear models to three different types of response models that are labeled continuous, ordinal, and nominal. Many details about the factor side are the same for the different response models, but JMP supports graphics and marginal profiles only for continuous responses—not for ordinal and nominal.

Different computer programs use different design-matrix codings, and thus parameterizations, to fit effects and construct hypothesis tests. JMP uses a different coding than the GLM procedure in the SAS system, although in most cases JMP and SAS GLM procedure produce the same results. The following sections describe the details of JMP coding and highlight those cases when it differs from that of the SAS GLM procedure.

Contents

The Response Models . . . 533
Continuous Responses . . . 533
Nominal Responses . . . 534
Ordinal Responses . . . 535
The Factor Models . . . 537
Continuous Factors . . . 537
Nominal Factors . . . 537
Ordinal Factors . . . 549
Frequencies . . . 554
The Usual Assumptions . . . 555
Assumed Model . . . 555
Relative Significance . . . 555
Multiple Inferences . . . 555
Validity Assessment . . . 556
Alternative Methods . . . 556
Key Statistical Concepts . . . 557
Uncertainty, a Unifying Concept . . . 557
The Two Basic Fitting Machines . . . 558
Likelihood, AICc, and BIC . . . 561
Power Calculations . . . 562
Computations for the LSN . . . 563
Computations for the LSV . . . 563
Computations for the Power . . . 564
Computations for the Adjusted Power . . . 565
Inverse Prediction with Confidence Limits . . . 566

The Response Models

JMP fits linear models to three different types of responses: continuous, nominal, and ordinal. The models and methods available in JMP are practical, are widely used, and suit the need for a general approach in a statistical software tool. As with all statistical software, you are responsible for learning the assumptions of the models that you choose to use, and the consequences if the assumptions are not met. See "The Usual Assumptions" in this chapter.

- "Continuous Responses"
- "Nominal Responses"
- "Ordinal Responses"

Continuous Responses

When the response column (the column assigned the Y role) is continuous, JMP fits the value of the response directly. The basic model is that for each observation,

Y = (some function of the X's and parameters) + error

Statistical tests are based on the assumption that the error term in the model is normally distributed.

Fitting Principle for Continuous Response

The Fitting principle is called *least squares*. The least squares method estimates the parameters in the model to minimize the sum of squared errors. The errors in the fitted model, called *residuals*, are the difference between the actual value of each observation and the value predicted by the fitted model.

The least squares method is equivalent to the maximum likelihood method of estimation if the errors have a normal distribution. This means that the analysis estimates the model that gives the most likely residuals. The log-likelihood is a scale multiple of the sum of squared errors for the normal distribution.

Base Model for Continuous Responses

The simplest model for continuous measurement fits just one value to predict all the response values. This value is the estimate of the *mean*. The mean is just the arithmetic average of the response values. All other models are compared to this base model.

Nominal Responses

Nominal responses are analyzed with a straightforward extension of the logit model. For a binary (two-level) response, a logit response model is specified as follows:

$$\log\left(\frac{P(y=1)}{P(y=2)}\right) = X\beta$$

The above can also be written as follows:

$$P(y=1) = F(X\beta)$$

where $F(x)$ is the cumulative distribution function of the standard logistic distribution:

$$F(x) = \frac{1}{1+e^{-x}} = \frac{e^{x}}{1+e^{x}}$$

For r response levels, JMP fits the probabilities that the response is one of r different response levels given by the data values. The probability estimates must all be positive. For a given configuration of Xs, the probability estimates must sum to 1 over the response levels. The function that JMP uses to predict probabilities is a composition of a linear model and a multi-response logistic function. This is sometimes called a *log-linear* model because the logs of ratios of probabilities are linear models. JMP relates each response probability to the r^{th} probability and fit a separate set of design parameters to these r - 1 models.

$$\log\left(\frac{P(y=j)}{P(y=r)}\right) = X\beta_{(j)} \quad \text{for} \quad j = 1, \ldots, r-1$$

Fitting Principle for Nominal Response

The fitting principle is called *maximum likelihood*. It estimates the parameters such that the joint probability for all the responses given by the data is the greatest obtainable by the model. Rather than reporting the joint probability (likelihood) directly, it is more manageable to report the total of the negative logs of the likelihood.

The uncertainty (negative log-likelihood) is the sum of the negative logs of the probabilities attributed by the model to the responses that actually occurred in the sample data. For a sample of size n, it is often denoted as H and written

$$H = \sum_{i=1}^{n} -\log(P(y = y_i))$$

If you attribute a probability of 1 to each event that did occur, then the sum of the negative logs is zero for a perfect fit.

The nominal model can take a lot of time and memory to fit, especially if there are many response levels. JMP tracks the progress of its calculations with an *iteration history*, which shows the negative log-likelihood values becoming smaller as they converge to the estimates.

Base Model for Nominal Responses

The simplest model for a nominal response is a set of constant response probabilities fitted as the occurrence rates for each response level across the whole data table. In other words, the probability that y is response level j is estimated by dividing the total sample count n into the total of each response level n_j. This probability is specified as follows:

$$p_j = \frac{n_j}{n}$$

All other models are compared to this base model. The base model serves the same role for a nominal response as the sample mean does for continuous models.

The R^2 statistic measures the portion of the uncertainty accounted for by the model, which is

$$1 - \frac{H(\text{full model})}{H(\text{base model})}$$

However, it is rare in practice to get an R^2 near 1 for categorical models.

Ordinal Responses

With an ordinal response (Y), as with nominal responses, JMP fits probabilities that the response is one of r different response levels given by the data.

Ordinal data have an order like continuous data. The order is used in the analysis but the spacing or distance between the ordered levels is not used. If you have a numeric response but want your model to ignore the spacing of the values, you can assign the ordinal level to that response column. If you have a classification variable and the levels are in some natural order such as low, medium, and high, you can use the ordinal modeling type.

Ordinal responses are modeled by fitting a series of parallel logistic curves to the cumulative probabilities. Each curve has the same design parameters but a different intercept and is specified as follows:

$$\mathrm{P}(y \le j) = F(\alpha_j + \mathrm{X}\beta) \quad \text{for} \quad j = 1, \ldots, r-1$$

where r response levels are present and $F(x)$ is the standard logistic cumulative distribution function

$$F(x) = \frac{1}{1+e^{-x}} = \frac{e^{x}}{1+e^{x}}$$

Another way to write this is in terms of an unobserved continuous variable, z, that causes the ordinal response to change as it crosses various thresholds

$$y = \begin{cases} r & \alpha_{r-1} \le z \\ j & \alpha_{j-1} \le z < \alpha_j \\ 1 & z \le \alpha_1 \end{cases}$$

where z is an unobservable function of the linear model and error

$$z = X\beta + \varepsilon$$

and ε has the logistic distribution.

These models are attractive because they recognize the ordinal character of the response, they need far fewer parameters than nominal models, and the computations are fast.

A different but mathematically equivalent way to envision an ordinal model is to think of a nominal model where, instead of modeling the odds, you model the cumulative probability. Instead of fitting functions for all but the last level, you fit only one function and slide it to fit each cumulative response probability.

Fitting Principle for Ordinal Response

The maximum likelihood fitting principle for an ordinal response model is the same as for a nominal response model. It estimates the parameters such that the joint probability for all the responses that occur is the greatest obtainable by the model. It uses an iterative method that is faster and uses less memory than nominal fitting.

Base Model

The simplest model for an ordinal response, like a nominal response, is a set of response probabilities fitted as the occurrence rates of the response in the whole data table.

The Factor Models

The way the x-variables (factors) are modeled to predict an expected value or probability is the subject of the factor side of the model.

The factors enter the prediction equation as a linear combination of x values and the parameters to be estimated. For a continuous response model, where i indexes the observations and j indexes the parameters, the assumed model for a typical observation, y_i, is written

$$y_i = \beta_0 + \beta_1 x_{1i} + \ldots + \beta_k x_{ki} + \varepsilon_i$$

where

y_i is the response

x_{ij} are functions of the data

ε_i is an unobservable realization of the random error

b_j are unknown parameters to be estimated.

The way the x's in the linear model are formed from the factor terms is different for each modeling type. The linear model x's can also be complex effects such as interactions or nested effects. Complex effects are discussed in detail later.

Continuous Factors

Continuous factors are placed directly into the design matrix as regressors. If a column is a linear function of other columns, then the parameter for this column is marked *zeroed* or *nonestimable*. Continuous factors are centered by their mean when they are crossed with other factors (interactions and polynomial terms). Centering is suppressed if the factor has a Column Property of **Mixture** or **Coding**, or if the centered polynomials option is turned off when specifying the model. If there is a coding column property, the factor is coded before fitting.

Nominal Factors

Nominal factors are transformed into indicator variables for the design matrix. SAS GLM constructs an indicator column for each nominal level. JMP constructs the same indicator columns for each nominal level except the last level. When the last nominal level occurs, a one is subtracted from all the other columns of the factor. For example, consider a nominal factor A with three levels coded for GLM and for JMP as shown below.

Table A.1 Nominal Factor A

	GLM			JMP	
A	**A1**	**A2**	**A3**	**A13**	**A23**
A1	1	0	0	1	0
A2	0	1	0	0	1
A3	0	0	1	–1	–1

In GLM, the linear model design matrix has linear dependencies among the columns, and the least squares solution uses a generalized inverse. The solution chosen happens to be such that the A3 parameter is set to zero.

In JMP, the linear model design matrix is coded so that it achieves full rank unless there are missing cells or other incidental collinearity. The parameter for the A effect for the last level is the negative sum of the other levels, which makes the parameters sum to zero over all the effect levels.

Interpretation of Parameters

Note: The parameter for a nominal level is interpreted as the differences in the predicted response for that level from the average predicted response over all levels.

The design column for a factor level is constructed as the zero-one indicator of that factor level minus the indicator of the last level. This is the coding that leads to the parameter interpretation above.

Table A.2 Interpreting Parameters

JMP Parameter Report	How to Interpret	Design Column Coding
Intercept	mean over all levels	$\mathbf{1}'$
A[1]	$\alpha_1 - 1/3(\alpha_1 + \alpha_2 + \alpha_3)$	(A==1) – (A==3)
A[2]	$\alpha_2 - 1/3(\alpha_1 + \alpha_2 + \alpha_3)$	(A==2) – (A==3)

Interactions and Crossed Effects

Interaction effects with both GLM and JMP are constructed by taking a direct product over the rows of the design columns of the factors being crossed. For example, the GLM code

```
PROC GLM;
  CLASS A B;
  MODEL A B A*B;
```

yields this design matrix:

Table A.3 Design Matrix

		A			B			AB								
A	**B**	1	2	3	1	2	3	11	12	13	21	22	23	31	32	33
A1	**B1**	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0
A1	**B2**	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0
A1	**B3**	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0
A2	**B1**	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0
A2	**B2**	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0
A2	**B3**	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0
A3	**B1**	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0
A3	**B2**	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0
A3	**B3**	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1

Using the JMP **Fit Model** command and requesting a factorial model for columns A and B produces the following design matrix. Note that A13 in this matrix is A1–A3 in the previous matrix. However, A13B13 is A13*B13 in the current matrix.

Table A.4 Current Matrix

		A		B					
A	**B**	**13**	**23**	**13**	**23**	**A13 B13**	**A13 B23**	**A23 B13**	**A23 B23**
A1	**B1**	1	0	1	0	1	0	0	0
A1	**B2**	1	0	0	1	0	1	0	0

Table A.4 Current Matrix *(Continued)*

A1	**B3**	1	0	–1	–1	–1	–1	0	0
A2	**B1**	0	1	1	0	0	0	1	0
A2	**B2**	0	1	0	1	0	0	0	1
A2	**B3**	0	1	–1	–1	0	0	–1	–1
A3	**B1**	–1	–1	1	0	–1	0	–1	0
A3	**B2**	–1	–1	0	1	0	–1	0	–1
A3	**B3**	–1	–1	–1	–1	1	1	1	1

The JMP coding saves memory and some computing time for problems with interactions of factors with few levels.

The expected values of the cells in terms of the parameters for a three-by-three crossed model are:

Table A.5 Three-by-Three Crossed Model

	B1	**B2**	**B3**
A1	$\mu + \alpha_1 + \beta_1 + \alpha\beta_{11}$	$\mu + \alpha_1 + \beta_2 + \alpha\beta_{12}$	$\mu + \alpha_1 - \beta_1 - \beta_2 - \alpha\beta_{11} - \alpha\beta_{12}$
A2	$\mu + \alpha_2 + \beta_1 + \alpha\beta_{21}$	$\mu + \alpha_2 + \beta_2 + \alpha\beta_{22}$	$\mu + \alpha_2 - \beta_1 - \beta_2 - \alpha\beta_{21} - \alpha\beta_{22}$
A3	$\mu - \alpha_1 - \alpha_2 + \beta_1 - \alpha\beta_{11} - \alpha\beta_{21}$	$\mu - \alpha_1 - \alpha_2 + \beta_2 - \alpha\beta_{12} - \alpha\beta_{22}$	$\mu - \alpha_1 - \alpha_2 - \beta_1 - \beta_2 + \alpha\beta_{11} + \alpha\beta_{12} + \alpha\beta_{21} + \alpha\beta_{22}$

Nested Effects

Nested effects in GLM are coded the same as interaction effects because GLM determines the right test by what is not in the model. Any effect not included in the model can be soaked up by a containing interaction (or, equivalently, nested) effect.

Nested effects in JMP are coded differently. JMP uses the terms inside the parentheses as grouping terms for each group. For each combination of levels of the nesting terms, JMP constructs the effect on the outside of the parentheses. The levels of the outside term do not need to line up across the levels of the nesting terms. Each level of nest is considered separately with regard to the construction of design columns and parameters.

Table A.6 Nested Effects

				B(A)					
				A1	A1	A2	A2	A3	A3
A	B	A13	A23	B13	B23	B13	B23	B13	B23
A1	B1	1	0	1	0	0	0	0	0
A1	B2	1	0	0	1	0	0	0	0
A1	B3	1	0	–1	–1	0	0	0	0
A2	B1	0	1	0	0	1	0	0	0
A2	B2	0	1	0	0	0	1	0	0
A2	B3	0	1	0	0	–1	–1	0	0
A3	B1	–1	–1	0	0	0	0	1	0
A3	B2	–1	–1	0	0	0	0	0	1
A3	B3	–1	–1	0	0	0	0	–1	–1

Least Squares Means across Nominal Factors

Least squares means are the predicted values corresponding to some combination of levels, after setting all the other factors to some neutral value. The neutral value for direct continuous regressors is defined as the sample mean. The neutral value for an effect with uninvolved nominal factors is defined as the average effect taken over the levels (which happens to result in all zeros in our coding). Ordinal factors use a different neutral value in "Ordinal Least Squares Means". The least squares means might not be estimable, and if not, they are marked nonestimable. The least squares means in JMP agree with those in SAS PROC GLM (Goodnight and Harvey 1978) in all cases except when a weight is used. When a weight variable is used, JMP uses a weighted mean and SAS PROC GLM uses an unweighted mean for its neutral values.

Effective Hypothesis Tests

Generally, the hypothesis tests produced by JMP agree with the hypothesis tests of most other trusted programs, such as SAS PROC GLM (Hypothesis types III and IV). The following two sections describe where there are differences.

In SAS PROC GLM, the hypothesis tests for Types III and IV are constructed using the general form of estimable functions and finding functions that involve only the effects of interest and effects contained by the effects of interest (Goodnight 1978).

The same tests are constructed in JMP. However, because there is a different parameterization, an effect can be tested (assuming full rank for now) by doing a joint test on all the parameters for that effect. The tests do not involve containing interaction parameters because the coding has made them uninvolved with the tests on their contained effects.

If there are missing cells or other singularities, the JMP tests are different from GLM tests. There are several ways to describe them:

- JMP tests are equivalent to testing that the least squares means are different, at least for main effects. If the least squares means are nonestimable, then the test cannot include some comparisons and therefore loses degrees of freedom. For interactions, JMP is testing that the least squares means differ by more than just the marginal pattern described by the containing effects in the model.
- JMP tests an effect by comparing the SSE for the model with that effect to the SSE for the model without that effect. JMP parameterizes the model so that this method makes sense.
- JMP implements the *effective hypothesis tests* described by Hocking (1985, pp. 80–89, 163–166), although JMP uses structural rather than cell-means parameterization. Effective hypothesis tests start with the hypothesis desired for the effect and include "as much as possible" of that test. Of course, if there are containing effects with missing cells, then this test has to drop part of the hypothesis because the complete hypothesis would not be estimable. The effective hypothesis drops as little of the complete hypothesis as possible.
- The differences among hypothesis tests in JMP and GLM (and other programs) that relate to the presence of missing cells are not considered interesting tests anyway. If an interaction is significant, the test for the contained main effects is not interesting. If the interaction is not significant, then it can always be dropped from the model. Some tests are not even unique. If you relabel the levels in a missing cell design, then the GLM Type IV tests can change.

The following section continues this topic in finer detail.

Singularities and Missing Cells in Nominal Effects

Consider the case of linear dependencies among the design columns. With JMP coding, this does not occur unless there is insufficient data to fill out the combinations that need estimating, or unless there is some type of confounding or collinearity of the effects.

With linear dependencies, a least squares solution for the parameters might not be unique and some tests of hypotheses cannot be tested. The strategy chosen for JMP is to set parameter estimates to zero in sequence as their design columns are found to be linearly dependent on previous effects in the model. A special column in the report shows what parameter estimates are zeroed and which parameter estimates are estimable. A separate *singularities* report shows what the linear dependencies are.

In cases of singularities the hypotheses tested by JMP can differ from those selected by GLM. Generally, JMP finds fewer degrees of freedom to test than GLM because it holds its tests to a higher standard of marginality. In other words, JMP tests always correspond to tests across least squares means for that effect, but GLM tests do not always have this property.

For example, consider a two-way model with interaction and one missing cell where A has three levels, B has two levels, and the A3B2 cell is missing.

Table A.7 Two-Way Model with Interaction

A B	A1	A2	B1	A1B1	A2B1	
A1 B1	1	0	1	1	0	
A2 B1	0	1	1	0	1	
A3 B1	–1	–1	1	–1	–1	
A1 B2	1	0	–1	–1	0	
A2 B2	0	1	–1	0	–1	
A3 B2	–1	–1	–1	1	1	Suppose this interaction is missing.

The expected values for each cell are:

Table A.8 Expected Values

	B1	B2
A1	$\mu + \alpha_1 + \beta_1 + \alpha\beta_{11}$	$\mu + \alpha_1 - \beta_1 - \alpha\beta_{11}$
A2	$\mu + \alpha_2 + \beta_1 + \alpha\beta_{21}$	$\mu + \alpha_2 - \beta_1 - \alpha\beta_{21}$
A3	$\mu - \alpha_1 - \alpha_2 + \beta_1 - \alpha\beta_{11} - \alpha\beta_{21}$	$\mu - \alpha_1 - \alpha_2 - \beta_1 + \alpha\beta_{11} + \alpha\beta_{21}$

Obviously, any cell with data has an expectation that is estimable. The cell that is missing has an expectation that is nonestimable. In fact, its expectation is precisely that linear combination of the design columns that is in the singularity report

$$\mu - \alpha_1 - \alpha_2 - \beta_1 + \alpha\beta_{11} + \alpha\beta_{21}$$

Suppose that you want to construct a test that compares the least squares means of B1 and B2. In this example, the average of the rows in the above table give these least squares means.

$$\begin{aligned} \text{LSM(B1)} = \; & (1/3)(\mu + \alpha_1 + \beta_1 + \alpha\beta_{11} + \\ & \mu + \alpha_2 + \beta_1 + \alpha\beta_{21} + \\ & \mu - \alpha_1 - \alpha_2 + \beta_1 - \alpha\beta_{11} - \alpha\beta_{21}) \\ = \; & \mu + \beta_1 \end{aligned}$$

$$\begin{aligned} \text{LSM(B2)} = \; & (1/3)(\mu + \alpha_1 + -\beta_1 - \alpha\beta_{11} + \\ & \mu + \alpha_2 + -\beta_1 - \alpha\beta_{21} + \\ & \mu - \alpha_1 - \alpha_2 - \beta_1 + \alpha\beta_{11} + \alpha\beta_{21}) \\ = \; & \mu - \beta_1 \end{aligned}$$

$$\text{LSM(B1)} - \text{LSM(B2)} = \; 2\beta_1$$

Note that this shows that a test on the β_1 parameter is equivalent to testing that the least squares means are the same. But because β_1 is not estimable, the test is not testable, meaning there are no degrees of freedom for it.

Now, construct the test for the least squares means across the A levels.

$$\begin{aligned} \text{LSM(A1)} = \; & (1/2)(\mu + \alpha_1 + \beta_1 + \alpha\beta_{11} + \mu + \alpha_1 - \beta_1 - \alpha\beta_{11}) \\ = \; & \mu + \alpha_1 \end{aligned}$$

$$\begin{aligned} \text{LSM(A2)} = \; & (1/2)(\mu + \alpha_2 + \beta_1 + \alpha\beta_{21} + \mu + \alpha_2 - \beta_1 - \alpha\beta_{21}) \\ = \; & \mu + \alpha_2 \end{aligned}$$

$$\begin{aligned} \text{LSM(A3)} = \; & (1/2)(\mu - \alpha_1 - \alpha_2 + \beta_1 - \alpha\beta_{11} - \alpha\beta_{21} + \\ & \mu - \alpha_1 - \alpha_2 - \beta_1 + \alpha\beta_{11} + \alpha\beta_{21}) \\ = \; & \mu - \alpha_1 - \alpha_2 \end{aligned}$$

$$\text{LSM(A1)} - \text{LSM(A3)} = \; 2\alpha_1 + \alpha_2$$

$$\text{LSM(A2)} - \text{LSM(A3)} = \; 2\alpha_2 + \alpha_1$$

Neither of these turn out to be estimable, but there is another comparison that is estimable; namely comparing the two A columns that have no missing cells.

$$\text{LSM(A1)} - \text{LSM(A2)} = \; \alpha_1 - \alpha_2$$

This combination is indeed tested by JMP using a test with 1 degree of freedom, although there are two parameters in the effect.

The estimability can be verified by taking its inner product with the singularity combination, and checking that it is zero:

Table A.9 Verification

	singularity	combination
parameters	**combination**	**to be tested**
m	1	0
a_1	–1	1
a_2	–1	–1
b_1	–1	0
ab_{11}	1	0
ab_{21}	1	0

It turns out that the design columns for missing cells for any interaction always knocks out degrees of freedom for the main effect (for nominal factors). Thus, there is a direct relation between the non-estimability of least squares means and the loss of degrees of freedom for testing the effect corresponding to these least squares means.

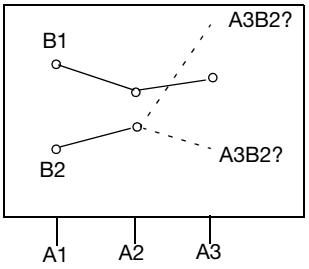
How does this compare with what GLM does? GLM and JMP do the same test when there are no missing cells. That is, they effectively test that the least squares means are equal. But when GLM encounters singularities, it focuses out these cells in different ways, depending on whether they are Type III or Type IV. For Type IV, it looks for estimable combinations that it can find. These might not be unique, and if you reorder the levels, you might get a different result. For Type III, it does some orthogonalization of the estimable functions to obtain a unique test. But the test might not be very interpretable in terms of the cell means.

The JMP approach has several points in its favor, although at first it might seem distressing that you might lose more degrees of freedom than with GLM:

1. The tests are philosophically linked to LSMs.
2. The tests are easy computationally, using reduction sum of squares for reparameterized models.
3. The tests agree with Hocking's "Effective Hypothesis Tests".
4. The tests are *whole marginal tests*, meaning they always go completely across other effects in interactions.

The last point needs some elaboration: Consider a graph of the expected values of the cell means in the previous example with a missing cell for A3B2.

Figure A.1 Expected Values of the Cell Means



The graph shows expected cell means with a missing cell. The means of the A1 and A2 cells are profiled across the B levels. The JMP approach says you cannot test the B main effect with a missing A3B2 cell, because the mean of the missing cell could be anything, as allowed by the interaction term. If the mean of the missing cell was the higher value shown, the B effect would likely test significant. If it were the lower, it would likely test nonsignificant. The point is that you do not know. That is what the least squares means are saying when they are declared nonestimable. That is what the hypotheses for the effects should be saying too—that you do not know.

If you want to test hypotheses involving margins for subsets of cells, then that is what GLM Type IV does. In JMP, you would have to construct these tests yourself by partitioning the effects with a lot of calculations or by using contrasts.

JMP and GLM Hypotheses

GLM works differently than JMP and produces different hypothesis tests in situations where there are missing cells. In particular, GLM does not recognize any difference between a nesting and a crossing in an effect, but JMP does. Suppose that you have a three-layer nesting of A, B(A), and C(A B) with different numbers of levels as you go down the nested design.

Figure A.10 shows the test of the main effect A in terms of the GLM parameters. The first set of columns is the test done by JMP. The second set of columns is the test done by GLM Type IV. The third set of columns is the test equivalent to that by JMP; it is the first two columns that have been multiplied by a matrix:

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

to be comparable to the GLM test. The last set of columns is the GLM Type III test. The difference is in how the test distributes across the containing effects. In JMP, it seems more top-down hierarchical. In GLM Type IV, the test seems more bottom-up. In practice, the test statistics are often similar.

Table A.10 Comparison of GLM and JMP Hypotheses

Parameter	JMP Test for A		GLM-IV Test for A		JMP Rotated Test		GLM-III Test for A	
u	0	0	0	0	0	0	0	0
a1	0.6667	-0.3333	1	0	1	0	1	0
a2	–0.3333	0.6667	0	1	0	1	0	1
a3	–0.3333	-0.3333	-1	-1	-1	-1	-1	-1
a1b1	0.1667	-0.0833	0.2222	0	0.25	0	0.2424	0
a1b2	0.1667	-0.0833	0.3333	0	0.25	0	0.2727	0
a1b3	0.1667	-0.0833	0.2222	0	0.25	0	0.2424	0
a1b4	0.1667	-0.0833	0.2222	0	0.25	0	0.2424	0
a2b1	-0.1667	0.3333	0	0.5	0	0.5	0	.5
a2b2	-0.1667	0.3333	0	0.5	0	0.5	0	.5
a3b1	-0.1111	-0.1111	-0.3333	-0.3333	-0.3333	-0.3333	-0.3333	-0.3333
a3b2	-0.1111	-0.1111	-0.3333	-0.3333	-0.3333	-0.3333	-0.3333	-0.3333
a3b3	-0.1111	-0.1111	-0.3333	-0.3333	-0.3333	-0.3333	-0.3333	-0.3333
a1b1c1	0.0833	-0.0417	0.1111	0	0.125	0	0.1212	0
a1b1c2	0.0833	-0.0417	0.1111	0	0.125	0	0.1212	0
a1b2c1	0.0556	-0.0278	0.1111	0	0.0833	0	0.0909	0
a1b2c2	0.0556	-0.0278	0.1111	0	0.0833	0	0.0909	0
a1b2c3	0.0556	-0.0278	0.1111	0	0.0833	0	0.0909	0

Table A.10 Comparison of GLM and JMP Hypotheses *(Continued)*

a1b3c1	0.0833	-0.0417	0.1111	0	0.125	0	0.1212	0
a1b3c2	0.0833	-0.0417	0.1111	0	0.125	0	0.1212	0
a1b4c1	0.0833	-0.0417	0.1111	0	0.125	0	0.1212	0
a1b4c2	0.0833	-0.0417	0.1111	0	0.125	0	0.1212	0
a2b1c1	-0.0833	0.1667	0	0.25	0	0.25	0	0.25
a2b1c2	-0.0833	0.1667	0	0.25	0	0.25	0	0.25
a2b2c1	-0.0833	0.1667	0	0.25	0	0.25	0	0.25
a2b2c2	-0.0833	0.1667	0	0.25	0	0.25	0	0.25
a3b1c1	-0.0556	-0.0556	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667
a3b1c2	-0.0556	-0.0556	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667
a3b2c1	-0.0556	-0.0556	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667
a3b2c2	-0.0556	-0.0556	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667
a3b3c1	-0.0556	-0.0556	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667
a3b3c2	-0.0556	-0.0556	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667	-0.1667

From the perspective of the JMP parameterization, the tests for A are:

Table A.11 Tests for A

parameter	**GLM–IV test**		**JMP test**	
m	0	0	0	0
a13	2	1	1	0
a23	1	2	0	1
a1:b14	0	0	0	0
a1:b24	0.11111	0	0	0
a1:b34	0	0	0	0

Table A.11 Tests for A *(Continued)*

a2:b12	0	0	0	0
a3:b13	0	0	0	0
a3:b23	0	0	0	0
a1b1:c12	0	0	0	0
a1b2:c13	0	0	0	0
a1b2:c23	0	0	0	0
a1b3:c12	0	0	0	0
a1b4:c12	0	0	0	0
a2b1:c13	0	0	0	0
a2b2:c12	0	0	0	0
a3b1:c12	0	0	0	0
a3b2:c12	0	0	0	0
a3b3:c12	0	0	0	0

So from the JMP perspective, the GLM test looks a little strange, putting a coefficient on the a1b24 parameter.

Ordinal Factors

Factors marked with the ordinal modeling type are coded differently than nominal factors. The parameter estimates are interpreted differently, the tests are different, and the least squares means are different.

For ordinal factors, the first level of the factor is a control or baseline level, and the parameters measure the effect on the response as the ordinal factor is set to each succeeding level. The ordinal factor coding is appropriate for factors that contain levels that represent various doses, where the first dose is zero. The following table shows an example of a three-level ordinal factor:

Table A.12 Ordinal Factors

Term	Coded Column	
A	**a2**	**a3**

Table A.12 Ordinal Factors *(Continued)*

A1	0	0	control level, zero dose
A2	1	0	low dose
A3	1	1	higher dose

The pattern for the design is such that the lower triangle is ones with zeros elsewhere.

For a simple main-effects model, this can be specified as follows:

$$y = \mu + \alpha_2 X_{(a \le 2)} + \alpha_3 X_{(a \le 3)} + \varepsilon$$

noting that μ is the expected response at $A = 1$, $\mu + \alpha_2$ is the expected response at $A = 2$, and $\mu + \alpha_2 + \alpha_3$ is the expected response at $A = 3$. Thus, α_2 estimates the effect moving from $A = 1$ to $A = 2$ and α_3 estimates the effect moving from $A = 2$ to $A = 3$.

If all the parameters for an ordinal main effect have the same sign, then the response effect is monotonic across the ordinal levels.

Ordinal Interactions

The ordinal interactions, as with nominal effects, are produced with a horizontal direct product of the columns of the factors. Consider an example with two ordinal factors A and B, where each factor has three levels. The ordinal coding in JMP produces the design matrix shown next. The pattern for the interaction is a block lower-triangular matrix of lower-triangular matrices of ones.

Table A.13 Ordinal Interactions

						A*B			
						A2		**A3**	
A	**B**	A2	A3	B2	B3	B2	B3	B2	B3
A1	**B1**	0	0	0	0	0	0	0	0
A1	**B2**	0	0	1	0	0	0	0	0
A1	**B3**	0	0	1	1	0	0	0	0
A2	**B1**	1	0	0	0	0	0	0	0
A2	**B2**	1	0	1	0	1	0	0	0
A2	**B3**	1	0	1	1	1	1	0	0

Table A.13 Ordinal Interactions *(Continued)*

						A*B			
						A2		A3	
A	**B**	A2	A3	B2	B3	B2	B3	B2	B3
A3	**B1**	1	1	0	0	0	0	0	0
A3	**B2**	1	1	1	0	1	0	1	0
A3	**B3**	1	1	1	1	1	1	1	1

Note: When you test to see whether there is no effect, there is not much difference between nominal and ordinal factors for simple models. However, there are major differences when interactions are specified. We recommend that you use nominal rather than ordinal factors for most models.

Hypothesis Tests for Ordinal Crossed Models

To see what the parameters mean, examine this table of the expected cell means in terms of the parameters, where μ is the intercept, α_2 is the parameter for level A2, and so on.

Table A.14 Expected Cell Means

	B1	**B2**	**B3**
A1	μ	$\mu + \beta_2$	$\mu + \beta_2 + \beta_3$
A2	$\mu + \alpha_2$	$\mu + \alpha_2 + \beta_2 + \alpha\beta_{22}$	$\mu + \alpha_2 + \beta_2 + \beta_3 + \alpha\beta_{22} + \alpha\beta_{23}$
A3	$\mu + \alpha_2 + \alpha_3$	$\mu + \alpha_2 + \alpha_3 + \beta_2 + \alpha\beta_{22} + \alpha\beta_{32}$	$\mu + \alpha_2 + \alpha_3 + \beta_2 + \beta_3 + \alpha\beta_{22} + \alpha\beta_{23} + \alpha\beta_{32} + \alpha\beta_{33}$

Note that the main effect test for A is really testing the A levels holding B at the first level. Similarly, the main effect test for B is testing across the top row for the various levels of B holding A at the first level. This is the appropriate test for an experiment where the two factors are both doses of different treatments. The main question is the efficacy of each treatment by itself, and fewer points are devoted to looking for *drug interactions* when doses of both drugs are applied. In some cases, it might even be dangerous to apply large doses of each drug.

Note that each cell's expectation can be obtained by adding all the parameters associated with each cell that is to the left and above it, inclusive of the current row and column. The expected value for the last cell is the sum of all the parameters.

Though the hypothesis tests for effects contained by other effects differs with ordinal and nominal codings, the test of effects not contained by other effects is the same. In the crossed design above, the test for the interaction would be the same no matter whether A and B were fit nominally or ordinally.

Ordinal Least Squares Means

As stated previously, least squares means are the predicted values corresponding to some combination of levels, after setting all the other factors to some neutral value. JMP defines the neutral value for an effect with uninvolved ordinal factors as the effect at the first level, meaning the control, or *baseline* level.

This definition of least squares means for ordinal factors maintains the idea that the hypothesis tests for contained effects are equivalent to tests that the least squares means are equal.

Singularities and Missing Cells in Ordinal Effects

With the ordinal coding, you are saying that the first level of the ordinal effect is the baseline. It is thus possible to get good tests on the main effects even when there are missing cells in the interactions—even if you have no data for the interaction.

Example with Missing Cell

The example is the same as above, with two observations per cell except that the A3B2 cell has no data. You can now compare the results when the factors are coded nominally with results when they are coded ordinally. The model fit is the same, as seen in Figure A.2.

Table A.15 Observations

Y	A	B
12	1	1
14	1	1
15	1	2
16	1	2
17	2	1

Table A.15 Observations *(Continued)*

17	2	1
18	2	2
19	2	2
20	3	1
24	3	1

Figure A.2 Summary Information for Nominal Factors (Left) and Ordinal Factors (Right)

Summary of Fit

RSquare	0.891732
RSquare Adj	0.805118
Root Mean Square Error	1.48324
Mean of Response	17.2
Observations (or Sum Wgts)	10

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	4	90.60000	22.6500	10.2955
Error	5	11.00000	2.2000	Prob > F
C. Total	9	101.60000		0.0125*

Summary of Fit

RSquare	0.891732
RSquare Adj	0.805118
Root Mean Square Error	1.48324
Mean of Response	17.2
Observations (or Sum Wgts)	10

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	4	90.60000	22.6500	10.2955
Error	5	11.00000	2.2000	Prob > F
C. Total	9	101.60000		0.0125*

The parameter estimates are very different because of the different coding. Note that the missing cell affects estimability for the nominal parameters but not for the ordinal parameters.

Figure A.3 Parameter Estimates for Nominal Factors (Left) and Ordinal Factors (Right)

Parameter Estimates

Term		Estimate	Std Error	t Ratio	Prob>|t|
Intercept	Biased	18.083333	0.74162	24.38	<.0001*
A[1]	Biased	-3.833333	0.856349	-4.48	0.0065*
A[2]	Biased	-0.333333	0.856349	-0.39	0.7131
B[1]	Biased	-0.75	0.74162	-1.01	0.3583
A[1]*B[1]	Biased	-0.5	1.048809	-0.48	0.6537
A[2]*B[1]	Zeroed	0	0	.	.

Parameter Estimates

Term		Estimate	Std Error	t Ratio	Prob>|t|
Intercept		13	1.048809	12.40	<.0001*
A[2-1]		4	1.48324	2.70	0.0429*
A[3-2]		5	1.48324	3.37	0.0199*
B[2-1]		2.5	1.48324	1.69	0.1527
A[2-1]*B[2-1]		-1	2.097618	-0.48	0.6537
A[3-2]*B[2-1]	Zeroed	0	0	.	.

The singularity details show the linear dependencies (and also identify the missing cell by examining the values).

Figure A.4 Singularity Details for Nominal Factors (Left) and Ordinal Factors (Right)

Singularity Details

Term	Details
Intercept	=A[1] + A[2] + B[1] - A[1]*B[1] - A[2]*B[1]

Singularity Details

Term	Details
A[3-2]*B[2-1]	= 0

The effect tests lose degrees of freedom for nominal. In the case of B, there is no test. For ordinal, there is no loss because there is no missing cell for the *base* first level.

Figure A.5 Effects Tests for Nominal Factors (Left) and Ordinal Factors (Right)

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F	
A	2	1	24.500000	11.1364	0.0206*	LostDFs
B	1	0	0.000000	.	.	LostDFs
A*B	2	1	0.500000	0.2273	0.6537	LostDFs

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F	
A	2	2	81.333333	18.4848	0.0049*	
B	1	1	6.250000	2.8409	0.1527	
A*B	2	1	0.500000	0.2273	0.6537	LostDFs

The least squares means are also different. The nominal LSMs are not all estimable, but the ordinal LSMs are. You can verify the values by looking at the cell means. Note that the A*B LSMs are the same for the two. Figure A.6 shows least squares means for nominal and ordinal factors.

Figure A.6 Least Squares Means for Nominal Factors (Left) and Ordinal Factors (Right)

Effect Details

A

Least Squares Means Table

Level	Least Sq Mean		Std Error	Mean
1	14.250000		0.74161985	14.2500
2	17.750000		0.74161985	17.7500
3	0.000000	NonEstimable	.	22.0000

B

Least Squares Means Table

Level	Least Sq Mean		Std Error	Mean
1	17.333333		0.60553007	17.3333
2	0.000000	NonEstimable	.	17.0000

A*B

Least Squares Means Table

Level	Least Sq Mean		Std Error
1,1	13.000000		1.0488088
1,2	15.500000		1.0488088
2,1	17.000000		1.0488088
2,2	18.500000		1.0488088
3,1	22.000000		1.0488088
3,2	0.000000	NonEstimable	.

Effect Details

A

Least Squares Means Table

Level	Least Sq Mean	Std Error	Mean
1	13.000000	1.0488088	14.2500
2	17.000000	1.0488088	17.7500
3	22.000000	1.0488088	22.0000

B

Least Squares Means Table

Level	Least Sq Mean	Std Error	Mean
1	13.000000	1.0488088	17.3333
2	15.500000	1.0488088	17.0000

A*B

Least Squares Means Table

Level	Least Sq Mean		Std Error
1,1	13.000000		1.0488088
1,2	15.500000		1.0488088
2,1	17.000000		1.0488088
2,2	18.500000		1.0488088
3,1	22.000000		1.0488088
3,2	0.000000	NonEstimable	.

Frequencies

The impact of frequencies, including those with non-integer values, on an analysis is explained by their effect on the loss function. Suppose that you want to estimate the parameter θ using response values y_i and predictors x_{i1}, x_{i2},..., x_{in}. Suppose that the loss function, assuming no frequency variable, is given by the following:

$$L(\theta|\underset{\sim}{y}) = \sum_{i=1}^{n} L(\theta|y_i, x_{i1}, x_{i2},\ldots,x_{in})$$

If frequencies f_i are defined, then the loss function is:

$$L(\theta|\underset{\sim}{y},\underset{\sim}{f}) = \sum_{i=1}^{n} f_i L(\theta|y_i, x_{i1}, x_{i2},\ldots,x_{in})$$

Calculations for all inference-base quantities, such as parameter estimates, standard errors, hypothesis tests, and confidence intervals, are based on this form of the loss function.

The Usual Assumptions

Before you put your faith in statistics, reassure yourself that you know both the value and the limitations of the techniques that you use. Statistical methods are just tools—they cannot guard you from incorrect science (invalid statistical assumptions) or bad data.

Assumed Model

Most statistics are based on the assumption that the model is correct. To the extent that your model might not be correct, you must attenuate your credibility in the statistical reports that result from the model.

Relative Significance

Many statistical tests do not evaluate the model in an absolute sense. Significant test statistics might be saying only that the model fits better than some reduced model, such as the mean. The model can appear to fit the data but might not describe the underlying physical model well at all.

Multiple Inferences

Often the value of the statistical results is not that you believe in them directly, but rather that they provide a key to some discovery. To confirm the discovery, you might need to conduct further studies. Otherwise, you might just be sifting through the data.

For example, if you conduct enough analyses, you can find 5% significant effects in 5% of your studies, even if the factors have no predictive value. Similarly, to the extent that you use your data to shape your model (instead of testing the correct model for the data), you are corrupting the significance levels in your report. The random error then influences your model selection and leads you to believe that your model is better than it really is.

Validity Assessment

There are a variety of techniques and patterns to assess the validity of the model:

- Model validity can be checked against a saturated version of the factors with Lack of Fit tests. The Fit Model platform presents these tests automatically if your contain replicated x values in a non-saturated model.
- You can check the distribution assumptions for a continuous response by looking at plots of residuals and studentized residuals from the Fit Model platform. Or, use the **Save** commands in the platform pop-up menu to save the residuals in data table columns. Then use the **Analyze > Distribution** on these columns to look at a histogram with its normal curve and the normal quantile plot. The residuals are not quite independent, but you can informally identify severely non-normal distributions.
- The best all-around diagnostic tool for continuous responses is the leverage plot because it shows the influence of each point on each hypothesis test. If you suspect that there is a mistaken value in your data, this plot helps determine whether a statistical test is heavily influenced by a single point.
- It is a good idea to scan your data for outlying values and examine them to see whether they are valid observations. You can spot univariate outliers in the Distribution platform reports and plots. Bivariate outliers appear in Fit Y by X scatterplots and in the Multivariate scatterplot matrix. You can see trivariate outliers in a three-dimensional plot produced by the **Graph > Scatterplot 3D**. Higher dimensional outliers can be found with Principal Components or Scatterplot 3D, and with Mahalanobis and jack-knifed distances computed and plotted in the Multivariate platform.

Alternative Methods

The statistical literature describes special nonparametric and robust methods, but JMP implements only a few of them at this time. These methods require fewer distributional assumptions (nonparametric), and then are more resistant to contamination (robust). However, they are less conducive to a general methodological approach, and the small sample probabilities on the test statistics can be time consuming to compute.

If you are interested in linear rank tests and need only normal large sample significance approximations, you can analyze the ranks of your data to perform the equivalent of a Wilcoxon rank-sum or Kruskal-Wallis one-way test.

If you are uncertain that a continuous response adequately meets normal assumptions, you can change the modeling type from continuous to ordinal and then analyze safely, even though this approach sacrifices some richness in the presentations and some statistical power as well.

Key Statistical Concepts

There are two key concepts that unify classical statistics and encapsulate statistical properties and fitting principles into forms that you can visualize:

- a unifying concept of uncertainty
- two basic fitting machines

These two ideas help unlock the understanding of statistics with intuitive concepts that are based on the foundation laid by mathematical statistics.

Statistics is to science what accounting is to business. It is the craft of weighing and balancing observational evidence. Statistical tests are like credibility audits. But statistical tools can do more than that. They are instruments of discovery that can show unexpected things about data and lead to interesting new ideas. Before using these powerful tools, you need to understand a bit about how they work.

Uncertainty, a Unifying Concept

When you do accounting, you total money amounts to get summaries. When you look at scientific observations in the presence of uncertainty or noise, you need some statistical measurement to summarize the data. Just as money is additive, uncertainty is additive if you choose the right measure for it.

The best measure is not the direct probability because to get a joint probability, you have to assume that the observations are independent and then multiply probabilities rather than add them. It is easier to take the log of each probability because then you can sum them and the total is the log of the joint probability.

However, the log of a probability is negative because it is the log of a number between 0 and 1. In order to keep the numbers positive, JMP uses the negative log of the probability. As the probability becomes smaller, its negative log becomes larger. This measure is called uncertainty, and it is measured in reverse fashion from probability.

In business, you want to maximize revenues and minimize costs. In science, you want to minimize uncertainty. Uncertainty in science plays the same role as cost plays in business. All statistical methods fit models such that uncertainty is minimized.

It is not difficult to visualize uncertainty. Just think of flipping a series of coins where each toss is independent. The probability of tossing a head is 0.5, and -log(0.5) is 1 for base 2 logarithms. The probability of tossing h heads in a row is defined as follows:

$$p = \left(\frac{1}{2}\right)^h$$

Solving for h produces the following:

$$h = -\log_2 p$$

You can think of the uncertainty of some event as the number of consecutive "head" tosses you have to flip to get an equally rare event.

Almost everything we do statistically has uncertainty, -log$p$, at the core. Statistical literature refers to uncertainty as *negative log-likelihood*.

The Two Basic Fitting Machines

An amazing fact about statistical fitting is that most of the classical methods reduce to using two simple machines, the spring and the pressure cylinder.

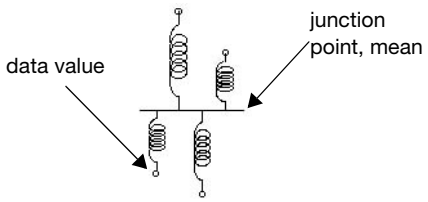
Springs

First, springs are the machine of fit for a continuous response model (Farebrother 1987). Suppose that you have n points and that you want to know the expected value (mean) of the points. Envision what happens when you lay the points out on a scale and connect them to a common junction with springs (Figure A.7). When you let go, the springs wiggle the junction point up and down and then bring it to rest at the mean. This is what must happen according to physics.

If the data are normally distributed with a mean at the junction point where springs are attached, then the physical energy in each point's spring is proportional to the uncertainty of the data point. All you have to do to calculate the energy in the springs (the uncertainty) is to compute the sum of squared distances of each point to the mean.

To choose an estimate that attributes the least uncertainty to the observed data, the spring settling point is chosen as the estimate of the mean. That is the point that requires the least energy to stretch the springs and is equivalent to the least squares fit.

Figure A.7 Connect Springs to Data Points

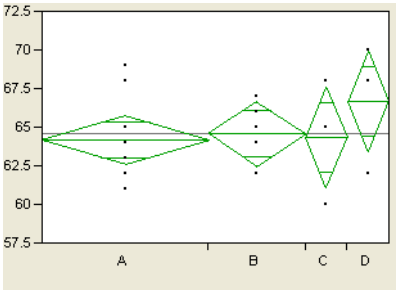


That is how you fit one mean or fit several means. That is how you fit a line, or a plane, or a hyperplane. That is how you fit almost any model to continuous data. You measure the energy or uncertainty by the sum of squares of the distances that you must stretch the springs.

Statisticians put faith in the normal distribution because it is the one that requires the least faith. It is, in a sense, the most random. It has the most non-informative shape for a distribution. It is the one distribution that has the most expected uncertainty for a given variance. It is the distribution whose uncertainty is measured in squared distance. In many cases it is the limiting distribution when you have a mixture of distributions or a sum of independent quantities. It is the distribution that leads to test statistics that can be measured fairly easily.

When the fit is constrained by hypotheses, you test the hypotheses by measuring this same spring energy. Suppose you have responses from four different treatments in an experiment, and you want to test if the means are significantly different. First, envision your data plotted in groups as shown in Figure A.8, but with springs connected to a separate mean for each treatment. Then exert pressure against the spring force to move the individual means to the common mean. Presto! The amount of energy that constrains the means to be the same is the test statistic that you need. That energy is the main ingredient in the *F* test for the hypothesis that tests whether the means are the same.

Figure A.8 A Oneway Plot for a Continuous Response Variable



Pressure Cylinders

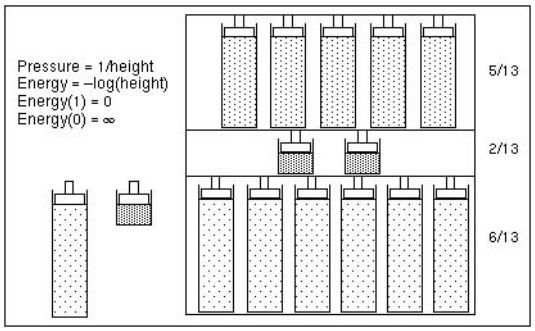
What if your response is categorical instead of continuous? For example, suppose that the response is the country of origin for a sample of cars. For your sample, there are probabilities for the three response levels, American, European, and Japanese. You can set these probabilities for country of origin to some estimate and then evaluate the uncertainty in your data. This uncertainty is found by summing the negative logs of the probabilities of the responses given by the data. It is defined as follows:

$$H = \sum h_{y(i)} = -\sum \log p_{y(i)}$$

The idea of springs illustrates how a mean is fit to continuous data. When the response is categorical, statistical methods estimate the response probabilities directly and choose the estimates that minimize the total uncertainty of the data. The probability estimates must be nonnegative and sum to 1. You can picture the response probabilities as the composition along a scale whose total length is 1. For each response observation, load into its response area a gas pressure cylinder, such as a tire pump. Let the partitions between the response levels vary until an equilibrium of lowest potential energy is reached. The sizes of the partitions that result then estimate the response probabilities.

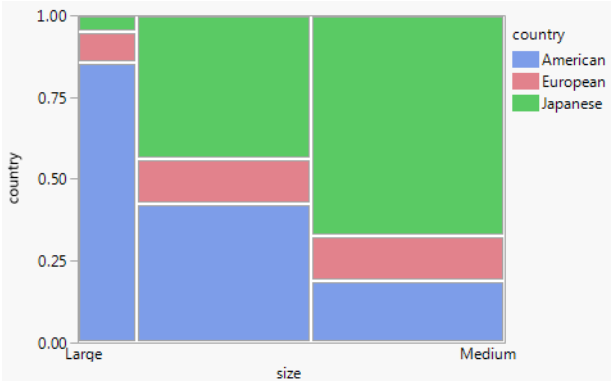
Figure A.9 shows what the situation looks like for a single category such as the medium size cars (see the mosaic column from Carpoll.jmp labeled medium in Figure A.10). Suppose there are thirteen responses (cars). The first level (American) has six responses, the next has two, and the last has five responses. The response probabilities become 6/13, 2/13, and 5/13, respectively, as the pressure against the response partitions balances out to minimize the total energy.

Figure A.9 Effect of Pressure Cylinders in Partitions



As with springs for continuous data, you can divide your sample by some factor and fit separate sets of partitions. Then test that the response rates are the same across the groups by measuring how much additional energy you need to push the partitions to be equal. Imagine the pressure cylinders for car origin probabilities grouped by the size of the car. The energy required to force the partitions in each group to align horizontally tests whether the variables have the same probabilities. Figure A.10 shows these partitions.

Figure A.10 A Mosaic Plot for Categorical Data



Likelihood, AICc, and BIC

Many statistical models in JMP are fit using a technique called *maximum likelihood*. This technique seeks to estimate the parameters of a model, which we denote generically by β, by maximizing the likelihood function. The likelihood function, denoted L(β), is the product of the probability density functions (or probability mass functions for discrete distributions) evaluated at the observed data values. Given the observed data, maximum likelihood estimation seeks to find values for the parameters, β, that maximize L(β).

Rather than maximize the likelihood function L(β), it is more convenient to work with the negative of the natural logarithm of the likelihood function, *-Log* L(β). The problem of maximizing L(β) is reformulated as a minimization problem where you seek to minimize the negative log-likelihood (*-LogLikelihood* = *-Log* L(β)). Therefore, smaller values of the negative log-likelihood or twice the negative log-likelihood (*-2LogLikelihood*) indicate better model fits.

You can use the value of negative log-likelihood to choose between models and to conduct custom hypothesis tests that compare models fit using different platforms in JMP. This is done through the use of likelihood ratio tests. One reason that *-2LogLikelihood* is reported in many JMP platforms is that the distribution of the difference between the full and reduced model *-2LogLikelihood* values is asymptotically Chi-square. The degrees of freedom associated with this likelihood ratio test are equal to the difference between the numbers of parameters in the two models (Wilks 1938).

The corrected Akaike's Information Criterion (AICc) and the Bayesian Information Criterion (BIC) are information-based criteria that assess model fit. Both are based on *-2LogLikelihood*.

AICc is defined as follows:

$$\text{AICc} = -2LogLikelihood + 2k + 2k(k+1)/(n-k-1)$$

where k is the number of estimated parameters in the model and n is the number of observations used in the model. This value can be used to compare various models for the same data set to determine the best-fitting model. The model having the smallest value, as discussed in Akaike (1974), is usually the preferred model.

BIC is defined as follows:

$$\text{BIC} = -2LogLikelihood + k\ln(n)$$

where k is the number of estimated parameters in the model and n is the number of observations used in the model. When comparing the BIC values for two models, the model with the smaller BIC value is considered better.

In general, BIC penalizes models with more parameters more than AICc does. For this reason, it leads to choosing more parsimonious models, that is, models with fewer parameters, than does AICc. For a detailed comparison of AICc and BIC, see Burnham and Anderson (2004).

Simplified Formulas for AICc and BIC in Least Squares Regression

In the case of least squares regression, the AICc and BIC can also be calculated based on the sum of squared errors (SSE). In terms of SSE, AICc and BIC are defined as follows:

$$\text{AICc} = n\ln\left(\frac{SSE}{n}\right) + 2k + 2k(k+1)/(n-k-1) + n\ln(2\pi) + n$$

$$\text{BIC} = n\ln\left(\frac{SSE}{n}\right) + k\ln(n) + n\ln(2\pi) + n$$

where k is the number of estimated parameters in the model, n is the number of observations used in the model, and SSE is the error sum of squares in the model.

Power Calculations

The next sections give formulas for computing the least significant number (LSN), least significant value (LSV), power, and adjusted power. With the exception of LSV, these computations are provided for each effect, and for a collection of user-specified contrasts (under Custom Test and LS Means Contrast). LSV is computed only for a single linear contrast. In the details below, the *hypothesis* refers to the collection of contrasts of interest.

- "Computations for the LSN"
- "Computations for the LSV"
- "Computations for the Power"
- "Computations for the Adjusted Power"

Computations for the LSN

The least significant number (LSN) solves for N in the equation:

$$\alpha = 1 - FDist\left[\frac{N\delta^2}{df_{Hyp}\sigma^2}, df_{Hyp}, N - df_{Hyp} - 1\right]$$

where

$FDist$ is the cumulative distribution function of the central F distribution

df_{Hyp} represents the degrees of freedom for the hypothesis

σ^2 is the error variance

δ^2 is the squared effect size

For retrospective analyses, δ^2 is estimated by the sum of squares for the hypothesis divided by n, the size of the current sample. If the test is for an effect, then $\delta^2$is estimated by the sum of squares for that effect divided by the number of observations in the current study. For retrospective studies, the error variance σ^2 is estimated by the mean square error. These estimates, along with an α value of 0.05, are entered into the Power Details window as default values.

When you are conducting a prospective analysis to plan a future study, consider determining the sample size that will achieve a specified power (see "Computations for the Power").

Computations for the LSV

The least significant value (LSV) is computed only for a single linear contrast.

Test of a Single Linear Contrast

Consider the one-degree-freedom test $L\beta = 0$, where L is a row vector of constants. The test statistic for a t test for this hypothesis is:

$$\frac{Lb}{s\sqrt{L(X'X)^{-1}L'}}$$

where s is the root mean square error. We reject the hypothesis at significance level α if the absolute value of the test statistic exceeds the 1 - $\alpha/2$ quantile of the t distribution, $t_{1-\alpha/2}$, with degrees of freedom equal to those for error.

To find the least significant value, denoted $(Lb)^{LSV}$, we solve for Lb:

$$(Lb)^{LSV} = t_{1-\alpha/2}s\sqrt{L(X'X)^{-1}L'}$$

Test of a Single Parameter

In the special case where the linear contrast tests a hypothesis setting a single β_i equal to 0, this reduces to the following:

$$b_i^{LSV} = t_{1-\alpha/2} s \sqrt{(X'X)_{ii}^{-1}} = t_{1-\alpha/2} StdError(b_i)$$

Test of a Difference in Means

In a situation where the test of interest is a comparison of two group means, the literature talks about the *least significant difference (LSD)*. In the special case where the model contains only one nominal variable, the formula for testing a single linear contrast reduces to the formula for the LSD:

$$\text{LSD} = t_{1-\alpha/2} s \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

However, in JMP, the parameter associated with a level for a nominal effect measures the difference between the mean of that level and the mean for all levels. So, the LSV for such a comparison is half the LSD for the differences of the means.

Note: If you are testing a contrast across the levels of a nominal effect, keep in mind how JMP codes nominal effects. Namely, the parameter associated with a given level measures the difference to the average for all levels.

Computations for the Power

Suppose that you are interested in computing the power of a test of a linear hypothesis, based on significance level α and a sample size of N. You want to detect an effect of size δ.

To calculate the power, begin by finding the critical value for an α-level F test of the linear hypothesis. This is given by solving for F_C in the equation

$$\alpha = 1 - FDist[F_C, df_{Hyp}, N - df_{Model} - 1]$$

Here, df_{Hyp} represents the degrees of freedom for the hypothesis, df_{Model} represents the degrees of freedom for the model, and N is the proposed (or actual) sample size.

Then calculate the noncentrality parameter associated with the desired effect size. The noncentrality parameter is defined as follows:

$$\lambda = (N\delta^2)/\sigma^2$$

where σ^2 is a proposed (or estimated) value of the error variance.

Given an effect of size δ, the test statistic has a non-central *F* distribution, where the distribution function is denoted *FDist* below, with noncentrality parameter λ. To obtain the power of your test, calculate the probability that the test statistic exceeds the critical value:

$$\text{Power} = 1 - FDist\left[F_C, df_{Hyp}, N - df_{Model} - 1, \frac{N\delta^2}{\sigma^2}\right]$$

In obtaining retrospective power for a study with *n* observations, JMP estimates the noncentrality parameter $\lambda = (n\delta^2)/\sigma^2$ by $\hat{\lambda} = SS_{Hyp}/\hat{\sigma}^2$, where SS_{Hyp} represents the sum of squares due to the hypothesis.

Computations for the Adjusted Power

The adjusted power calculation (Wright and O'Brien 1988) is relevant only for retrospective power analysis. Adjusted power calculates power using a noncentrality parameter estimate that has been adjusted to remove the positive bias that occurs when parameters are simply replaced by their sample estimates.

The estimate of the noncentrality parameter, λ, obtained by estimating δ and σ by their sample estimates, is calculated as follows:

$$\hat{\lambda} = SS_{Hyp}/MSE$$

Wright and O'Brien (1988) explain that an unbiased estimate of the noncentrality parameter is given by the following:

$$[\hat{\lambda}(df_{Error} - 2)/df_{Error}] - df_{Hyp} = \frac{\hat{\lambda}(N - df_{Model} - 1 - 2)}{N - df_{Model} - 1} - df_{Hyp}$$

The expression on the right illustrates the calculation of the unbiased noncentrality parameter when a sample size *N*, different from the study size *n*, is proposed for a retrospective power analysis. Here, df_{Hyp} represents the degrees of freedom for the hypothesis and df_{Model} represents the degrees of freedom for the whole model.

Unfortunately, this adjustment to the noncentrality estimate can lead to negative values. Negative values are set to zero, reintroducing some slight bias. The adjusted noncentrality estimate is

$$\hat{\lambda}_{adj} = \text{Max}\left[0, \frac{\hat{\lambda}(N - df_{Model} - 1 - 2)}{N - df_{Model} - 1} - df_{Hyp}\right]$$

The adjusted power is

$$\text{Power}_{\text{adj}} = 1 - FDist[F_C, df_{Hyp}, N - df_{Model} - 1, \hat{\lambda}_{\text{adj}}]$$

Confidence limits for the noncentrality parameter are constructed as described in Dwass (1955):

$$\text{Lower CL for } \lambda = \text{Max}\left[0, \left[\left(\sqrt{SS_{\text{Hyp}}/MSE}\right) - \sqrt{df_{Hyp} F_C}\right]^2\right]$$

$$\text{Upper CL for } \lambda = \left[\left(\sqrt{SS_{\text{Hyp}}/MSE}\right) - \sqrt{df_{Hyp} F_C}\right]^2$$

Confidence limits for the power are obtained by substituting these confidence limits for λ into the following equation:

$$\text{Power} = 1 - FDist[F_C, df_{Hyp}, N - df_{Model} - 1, \lambda]$$

Inverse Prediction with Confidence Limits

Inverse prediction estimates a value of an independent variable from a response value. In bioassay problems, inverse prediction with confidence limits is especially useful. In JMP, you can request inverse prediction estimates for continuous and binary response models. If the response is continuous, you can request confidence limits for an individual response or an expected response.

The confidence limits are computed using Fieller's theorem (1954), which is based on the following logic. The goal is predicting the value of a single regressor and its confidence limits given the values of the other regressors and the response.

- Let **b** estimate the parameters β so that we have **b** distributed as N(β,V).
- Let **x** be the regressor values of interest, with the i^{th} value to be estimated.
- Let **y** be the response value.

We desire a confidence region on the value of **x**[i] such that $\beta' x = y$ with all other values of **x** given.

The inverse prediction is

$$x[i] = \frac{y - \beta'_{(i)} x_{(i)}}{\beta[i]}$$

where the subscript (i) in parentheses indicates that the i^{th} component is omitted. A confidence interval can be formed from the relation

$$(y - b'x)^2 < t^2 x'Vx$$

where t is the t value for the specified confidence level.

The equation

$$(y - b'x)^2 - t^2 x'Vx = 0$$

can be written as a quadratic in terms of $z = x[i]$:

$$gz^2 + hz + f = 0$$

where

$$g = b[i]^2 - t^2 V[i, i]$$

$$h = -2yb[i] + 2b[i]b'_{(i)}x_{(i)} - 2t^2 V[i, (i)]'x_{(i)}$$

$$f = y^2 - 2yb'_{(i)}x_{(i)} + (b'_{(i)}x_{(i)})^2 - t^2 x_{(i)}'V_{(i)}x_{(i)}$$

Depending on the values of g, h, and f, the set of values satisfying the inequality, and hence the confidence interval for the inverse prediction, can have a number of forms:

- an interval of the form (ϕ_1, ϕ_2), where $\phi_1 < \phi_2$
- two disjoint intervals of the form $(-\infty, \phi_1) \cup (\phi_2, \infty)$, where $\phi_1 < \phi_2$
- the entire real line, $(-\infty, \infty)$
- only one of $-(\infty, \phi)$ or (ϕ, ∞)

In the case where the Fieller confidence interval is the entire real line, Wald intervals are presented.

Note: The Fit Y by X logistic platform and the Fit Model Nominal Logistic personalities use t values when computing confidence intervals for inverse prediction. The Fit Model Generalized Linear Model personality, as well as PROC PROBIT in SAS/STAT, use z values, which give different results.

Appendix B

References

Aitken, M. (1987). "Modelling Variance Heterogeneity in Normal Regression Using GLIM." *Journal of the Royal Statistical Society, Series C* 36:332–339.

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Anderson, T. W. (1958). *An Introduction to Multivariate Statistical Analysis*. New York: John Wiley & Sons.

Belsley, D. A., Kuh, E., and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: John Wiley & Sons.

Benjamini, Y., and Hochberg, Y. (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society, Series B* 57:289–300.

Box, G. E. P. (1954). "Some Theorems on Quadratic Forms Applied in the Study of Analysis of Variance Problems, Part 2: Effects of Inequality of Variance and of Correlation between Errors in the Two-Way Classification." *Annals of Mathematical Statistics* 25:484–498.

Box, G. E. P., and Cox, D. R. (1964). "An Analysis of Transformations." *Journal of the Royal Statistical Society, Series B* 26:211–243.

Box, G. E. P., and Meyer, R. D. (1986). "An Analysis for Unreplicated Fractional Factorials." *Technometrics* 28:11–18.

Box, G. E. P., and Meyer, R. D. (1993). "Finding the Active Factors in Fractionated Screening Experiments." *Journal of Quality Technology* 25:94–105.

Burnham, K. P., and Anderson, D. R. (2004). "Multimodel Inference: Understanding AIC and BIC in Model Selection." *Sociological Methods and Research* 33:261–304.

Burnham, K. P., Andersen, D. R., and Huyvaert, K. P. (2011). "AIC Model Selection and Multimodel Inference in Behavioral Ecology: Some Background, Observations, and Comparisons." *Behavioral Ecology and Sociobiology* 65:23–35.

Candes, E., and Tao, T. (2007). "The Dantzig Selector: Statistical Estimation when p is Much Larger than n." *The Annals of Statistics* 35:2313–2351.

Carroll, R. J., and Ruppert, D. (1988). *Transformation and Weighting in Regression*. London: Chapman & Hall.

Chilès, J.-P., and Delfiner, P. (2012). *Geostatistics: Modeling Spatial Uncertainty*. 2nd ed. New York: John Wiley & Sons.

Cobb, G. W. (1998). *Introduction to Design and Analysis of Experiments*. New York: Springer-Verlag.

Cohen, J. (1977). *Statistical Power Analysis for the Behavioral Sciences*. New York: Academic Press.

Cole, J. W. L., and Grizzle, J. E. (1966). "Applications of Multivariate Analysis of Variance to Repeated Measures Experiments." *Biometrics* 22:810–828.

Conover, W. J. (1999). *Practical Nonparametric Statistics*. 3rd ed. New York: John Wiley & Sons.

Cook, R. D., and Weisberg, S. (1982). *Residuals and Influence in Regression*. New York: Chapman & Hall.

Cook, R. D., and Weisberg, S. (1983). "Diagnostics for Heteroscedasticity in Regression." *Biometrika* 70:1–10.

Cornell, J. A. (1990). *Experiments with Mixtures*. 2nd ed. New York: John Wiley & Sons.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Cressie, N. A. C. (1993). *Statistics for Spatial Data*. Rev. ed. New York: John Wiley & Sons.

Daniel, C. (1959). "Use of Half-Normal Plots in Interpreting Factorial Two-Level Experiments." *Technometrics* 1:311–341.

Dunnett, C. W. (1955). "A Multiple Comparisons Procedure for Comparing Several Treatments with a Control." *Journal of the American Statistical Association* 50:1096–1121.

Dwass, M. (1955). "A Note on Simultaneous Confidence Intervals." *Annals of Mathematical Statistics* 26:146–147.

Efron, B. (1977). "The Efficiency of Cox's Likelihood Function for Censored Data." *Journal of the American Statistical Association* 72:557–565.

Farebrother, R. W. (1987). "Mechanical Representations of the L1 and L2 Estimation Problems." In *Statistical Data Analysis Based on L_1 Norm and Related Methods*, edited by Y. Dodge, 455–464. Amsterdam: North-Holland.

Fieller, E. C. (1954). "Some Problems in Interval Estimation." *Journal of the Royal Statistical Society, Series B* 16:175–185.

Firth, D. (1993). "Bias Reduction of Maximum Likelihood Estimates." *Biometrika* 80:27–38.

Fleming, T. R., and Harrington, D. P. (1991). *Counting Processes and Survival Analysis*. New York: John & Sons.

Goodnight, J. H. (1978). *Tests of Hypotheses in Fixed Effects Linear Models*. Technical Report R–101, SAS Institute Inc., Cary, NC.

Goodnight, J. H., and Harvey, W. R. (1978). *Least-Squares Means in the Fixed-Effects General Linear Models*. Technical Report R-103, SAS Institute Inc., Cary, NC.

Goos, P., and Jones, B. (2011). *Optimal Design of Experiments: A Case Study Approach*. Chichester, UK: John Wiley & Sons.

Greenhouse, S. W., and Geisser, S. (1959). "On Methods in the Analysis of Profile Data." *Psychometrika* 32:95–112.

Harrell, F. E. (1986). "The LOGIST Procedure." In *SUGI Supplemental Library Guide, Version 5 Edition*. Cary, NC: SAS Institute Inc.

Harvey, A. C. (1976). "Estimating Regression Models with Multiplicative Heteroscedasticity." *Econometrica* 44:461–465.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer-Verlag.

Hayter, A. J. (1984). "A Proof of the Conjecture That the Tukey-Kramer Method Is Conservative." *Annals of Statistics* 12:61–75.

Heinze, G., and Schemper, M. (2002). "A Solution to the Problem of Separation in Logistic Regression." *Statistics in Medicine* 21:2409–2419.

Hocking, R. R. (1985). *The Analysis of Linear Models*. Monterey, CA: Brooks/Cole.

Hoenig, J. M., and Heisey, D. M. (2001). "The Abuse of Power: The Pervasive Fallacy of Power Calculations for Data Analysis." *American Statistician* 55:19–24.

Hoerl, A. (1962). "Application of Ridge Analysis to Regression Problems." *Chemical Engineering Progress* 58:54–59.

Hoerl, A., and Kennard, R. (1970). "Ridge Regression: Biased Estimation for Non-orthogonal Problems." *Technometrics* 12:55–67.

Hsu, J. C. (1992). "The Factor Analytic Approach to Simultaneous Inference in the General Linear Model." *Journal of Computational and Graphical Statistics* 1:151–168.

Hsu, J. C. (1996). Multiple Comparisons: Theory and Methods. London: Chapman & Hall.

Hu, W., Laber, E. B., Barker, C., and Stefanski, L. A. (2019. "Assessing Tuning Parameter Selection Variability in Penalized Regression." *Technometrics* 61:154–164.

Huber, P. J., and Ronchetti, E. M. (2009). *Robust Statistics*. 2nd ed. John Wiley & Sons.

Hui, F., Warton, D., and Foster, S. (2015). "Tuning Parameter Selection for the Adaptive Lasso Using ERIC." *Journal of the American Statistical Association* 110:262–269.

Huynh, H., and Feldt, L. S. (1970). "Conditions Under Which Mean Square Ratios in Repeated Measurements Designs Have Exact F-Distributions." *Journal of the American Statistical Association* 65:1582–1589.

Huynh, H., and Feldt, L. S. (1976). "Estimation of the Box Correction for Degrees of Freedom from Sample Data in the Randomized Block and Split Plot Designs." *Journal of Educational Statistics* 1:69–82.

Kackar, R. N., and Harville, D. A. (1984). "Approximations for Standard Errors of Estimators of Fixed and Random Effects in Mixed Linear Models." *Journal of the American Statistical Association* 79:853–862.

Kalbfleisch, J. D., and Prentice, R. L. (2002). *The Statistical Analysis of Failure Time Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Kenward, M. G., and Roger, J. H. (1997). "Small Sample Inference for Fixed Effects from Restricted Maximum Likelihood." *Biometrics* 53:983–997.

Koenker, R., and Hallock, K. (2001). "Quantile Regression: An Introduction." *Journal of Economic Perspectives* 15:143–156.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Lenth, R. V. (1989). "Quick and Easy Analysis of Unreplicated Factorials." *Technometrics* 31:469–473.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

Mallows, C. L. (1973). "Some Comments on C_p." *Technometrics* 15:661–675.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

McClave, J. T., and Dietrich, F. H. (1988). *Statistics*. San Francisco: Dellen.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

McCulloch, C. E., Searle, S. R., and Neuhaus, J. M. (2008). *Generalized, Linear, and Mixed Models*. New York: John Wiley & Sons.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Miller, A. J. (1990). *Subset Selection in Regression*. New York: Chapman & Hall.

Montgomery, D. C. (1991). "Using Fractional Factorial Designs for Robust Process Development." *Quality Engineering* 3:193–205.

Muller, K. E., and Barton, C. N. (1989). "Approximate Power for Repeated-Measures ANOVA Lacking Sphericity." *Journal of the American Statistical Association* 84:549–555. Also see "Correction to 'Approximate Power for Repeated-Measures ANOVA Lacking Sphericity'," *Journal of the American Statistical Association* 86 (1991): 255–256.

Nagelkerke, N. J. D. (1991). "A Note on a General Definition of the Coefficient of Determination." *Biometrika* 78:691–692.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Nelson, F. D. (1976). "On a General Computer Algorithm for the Analysis of Models with Limited Dependent Variables." *Annals of Economic and Social Measurement* 5:493–509.

Nelson, P. R., Wludyka, P. S., and Copeland, K. A. F. (2005). *The Analysis of Means: A Graphical Method for Comparing Means, Rates, and Proportions*. Philadelphia: SIAM.

Patterson, H. D., and Thompson, R. (1974). "Maximum Likelihood Estimation of Components of Variance." In *Proceedings of the Eighth International Biometric Conference,* 197–207. Washington, DC: International Biometric Society.

Portnoy, S., and Koenker, R. (1997). "The Gaussian Hare and the Laplacian Tortoise: Computation of Squared-Error vs. Absolute-Error Estimators." *Statistical Science* 12:279–300.

Rawlings, J. O. (1988). *Applied Regression Analysis: A Research Tool*. Pacific Grove, CA: Wadsworth & Brooks/Cole Advanced Books & Software.

Ries, P. N., and Smith, H. (1963). "The Use of Chi-Square for Preference Testing in Multidimensional Problems." *Chemical Engineering Progress* 59:39–43.

Sall, J. P. (1990). "Leverage Plots for General Linear Hypotheses." *American Statistician* 44:308–315.

SAS Institute Inc. (2020a). "The GENMOD Procedure." In *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/genmod.pdf.

SAS Institute Inc. (2020b). "The GLM Procedure." In *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/glm.pdf.

SAS Institute Inc. (2020c). "Introduction to Statistical Modeling with SAS/STAT Software." In *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/intromod.pdf.

SAS Institute Inc. (2020d). "The MIXED Procedure." In *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/mixed.pdf.

Satterthwaite, F. E. (1946). "An Approximate Distribution of Estimates of Variance Components." *Biometrics Bulletin* 2:110–114.

Scheffé, H. (1958). "Experiments with Mixtures." *Journal of the Royal Statistical Society, Series B* 20:344–360.

Schuirmann, D. J. (1987). "A Comparison of the Two One-Sided Tests Procedure and the Power Approach for Assessing the Equivalence of Average Bioavailability." *Journal of Pharmacokinetics and Biopharmaceutics* 15:657–680.

Searle, S. R., Casella, G., and McCulloch, C. E. (1992). *Variance Components*. New York: John Wiley & Sons.

Seber, G. A. F. (1984). *Multivariate Observations*. New York: John Wiley & Sons.

Singer, J. D. (1998). "Using SAS PROC MIXED to Fit Multilevel Models, Hierarchical Models, and Individual Growth Models." *Journal of Educational and Behavioral Statistics* 23:323–355.

Snedecor, G. W., and Cochran, W. G. (1967). Statistical Methods. 6th ed. Ames: Iowa State University Press.

Spiller, S. A., Fitzsimons, G. J., Lynch, J. G., Jr., and McClelland, G. (2013). "Spotlights, Floodlights, and the Magic Number Zero: Simple Effects Tests in Moderated Regression." *Journal of Marketing Research* 50:277–288.

Stone, C., and Koo, C. Y. (1985). "Additive Splines in Statistics." In *Proceedings of the Statistical Computing Section*, 45–48. Alexandria, VA: American Statistical Association.

Sullivan, L. M., Dukes, K. A., and Losina, E. (1999). "An Introduction to Hierarchical Linear Modelling." *Statistics in Medicine* 18:855–888.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58:267–288.

Tukey, J. W. (1953). "The Problem of Multiple Comparisons." In *Multiple Comparisons*, 1948–1983, edited by H. I. Braun, vol. 8 of *The Collected Works of John W. Tukey* (published 1994), 1–300. London: Chapman & Hall. Unpublished manuscript.

Walker, S. H., and Duncan, D. B. (1967). "Estimation of the Probability of an Event as a Function of Several Independent Variables." *Biometrika* 54:167–179.

Westfall, P. H., Tobias, R. D., and Wolfinger, R. D. (2011). *Multiple Comparisons and Multiple Tests Using SAS.* 2nd ed. Cary, NC: SAS Institute Inc.

Wilks, S. S. (1938). "The Large-Sample Distribution of the Likelihood Ratio for Testing Composite Hypotheses." *Annals of Mathematical Statistics*. 9:60–62.

Wolfinger, R. D., Tobias, R. D., and Sall, J. (1994). "Computing Gaussian Likelihoods and Their Derivatives for General Linear Mixed Models." *SIAM Journal on Scientific Computing* 15:1294–1310.

Wright, S. P., and O'Brien, R. G. (1988). "Power Analysis in an Enhanced GLM Procedure: What it Might Look Like." In *Proceedings of the Thirteenth Annual SAS Users Group International Conference*, 1097–1102. Cary, NC: SAS Institute Inc. http://www.sascommunity.org/sugi/SUGI88/Sugi-13-220%20Wright%20OBrien.pdf.

Zou, H. (2006). "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association* 101:1418–1429.

Zou, H., and Hastie, T. (2005). "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society, Series B* 67:301–320.

Appendix C

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

  - You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - You must cause any modified files to carry prominent notices stating that You changed the files; and
  - You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
  - You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Predictive and Specialized Modeling

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Predictive and Specialized Modeling*. Cary, NC: SAS Institute Inc.

JMP® 16 Predictive and Specialized Modeling

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Predictive and Specialized Modeling

1 Learn about JMP 17
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 19
JMP Help 20
JMP Documentation Library 20
Additional Resources for Learning JMP 26
JMP Tutorials 26
Sample Data Tables 26
Learn about Statistical and JSL Terms 27
Learn JMP Tips and Tricks 27
JMP Tooltips 27
JMP User Community 28
Free Online Statistical Thinking Course 28
JMP New User Welcome Kit 28
Statistics Knowledge Portal 28
JMP Training 28
JMP Books by Users 29
The JMP Starter Window 29
JMP Technical Support 29

2 Introduction to Predictive and Specialized Modeling 31
Overview of Modeling Techniques

3 Neural Networks 33
Fit Nonlinear Models Using Nodes and Layers
Overview of the Neural Platform 35
Launch the Neural Platform 35
The Neural Launch Window 36
The Model Launch Control Panel 37
Validation Methods for Neural 38
Hidden Layer Structure 39
Boosting 40
Fitting Options 41

Model Reports 43
Training and Validation Measures of Fit 43
Confusion Statistics 44
Neural Platform Options 44
Neural Model Options 45
Example of a Neural Network 46

4 Partition Models 49
Use Decision Trees to Explore and Model Your Data

Overview of the Partition Platform 51
Example of the Partition Platform 51
Launch the Partition Platform 54
The Partition Report 55
Control Buttons 55
Report for Categorical Responses 56
Report for Continuous Responses 58
Partition Platform Options 61
Show Fit Details 64
Specify Profit Matrix 65
Decision Matrix Report 67
Informative Missing 68
Actual by Predicted Plot 69
ROC Curve 70
Lift Curve 71
Node Options 72
Validation in Partition 73
Additional Examples of Partitioning 74
Example of a Continuous Response 74
Example of Informative Missing 77
Example of Profit Matrix and Decision Matrix Report 80
Statistical Details for the Partition Platform 83
Responses and Factors 83
Splitting Criterion 84
Predicted Probabilities in Decision Tree and Bootstrap Forest 84

5 Bootstrap Forest 87
Fit a Model By Averaging Many Trees

Overview of the Bootstrap Forest Platform 89
Example of Bootstrap Forest with a Categorical Response 90
Bootstrap Forest Model 90
Missing Values 92

Example of Bootstrap Forest with a Continuous Response ... 93
Launch the Bootstrap Forest Platform ... 95
Launch Window ... 95
Specification Window ... 97
The Bootstrap Forest Report ... 99
Model Validation-Set Summaries ... 100
Specifications ... 100
Overall Statistics ... 100
Cumulative Validation ... 102
Per-Tree Summaries ... 103
Bootstrap Forest Platform Options ... 104

6 Boosted Tree ... 107
Fit Many Layers of Trees, Each Based on the Previous Layer

Overview of the Boosted Tree Platform ... 109
Example of Boosted Tree with a Categorical Response ... 109
Example of Boosted Tree with a Continuous Response ... 111
Launch the Boosted Tree Platform ... 113
Specification Window ... 115
The Boosted Tree Report ... 117
Model Validation - Set Summaries ... 118
Specifications ... 119
Overall Statistics ... 119
Cumulative Validation ... 120
Boosted Tree Platform Options ... 121
Statistical Details for the Boosted Tree Platform ... 123
Overfit Penalty ... 123

7 K Nearest Neighbors ... 125
Predict Response Values Using Nearby Observations

Overview of the K Nearest Neighbors Platform ... 127
Example of K Nearest Neighbors with Categorical Response ... 128
Example of K Nearest Neighbors with Continuous Response ... 130
Launch the K Nearest Neighbors Platform ... 132
The K Nearest Neighbors Report ... 134
Continuous Responses ... 134
Categorical Responses ... 135
K Nearest Neighbors Platform Options ... 136
Model Fit Options ... 136

8 Naive Bayes 139
Classify Observations Using Probabilistic Assumptions
Overview of the Naive Bayes Platform 141
Example of Naive Bayes 142
Launch the Naive Bayes Platform 144
The Naive Bayes Report 145
Fit Details Report 146
Response Column Report 146
Confusion Matrix Report 146
ROC Curves 146
Naive Bayes Platform Options 147
Additional Example of Naive Bayes 148
Statistical Details for the Naive Bayes Platform 149
Algorithm 149
Saved Probability Formulas 151

9 Support Vector Machines 153
Predict Response Categories Using Support Vectors
Overview of Support Vector Machines 155
Example of Support Vector Machines 155
Launch the Support Vector Machines Platform 157
The Support Vector Machines Launch Window 158
Model Launch Control Panel 159
The Support Vector Machine Report 162
Model Comparison Report 162
Support Vector Machine Model Report 163
Support Vector Machines Platform Options 165
Support Vector Machine Model Report Options 166
Additional Example of the SVM Platform 168
Example of Support Vector Regression for Continuous Response 168

10 Model Screening 171
Fit and Compare Multiple Prediction Models
Example of Model Screening for Continuous Response 173
Launch the Model Screening Platform 175
The Model Screening Report 178
Summary Across the Folds Report 179
Training, Validation, and Test Measures of Fit 180
Model Screening Platform Options 181
Decision Thresholds Report 184

11 Model Comparison 187
Compare the Predictive Ability of Fitted Models

Example of Model Comparison 189
Launch the Model Comparison Platform 192
The Model Comparison Report 193
Model Comparison Platform Options 195
Continuous and Categorical Responses 195
Continuous Responses 195
Categorical Responses 195
Additional Example of Model Comparison 196

12 Make Validation Column 199
Divide Data into Training, Validation, and Test Sets

Overview of the Make Validation Column Platform 201
Example of Make Validation Column 201
Launch the Make Validation Column Platform 203
Make Validation Column Report 206
Specify Rates or Relative Rates 206
Set Cutpoints 206
Options 207
Go 207
Make Validation Column Platform Options 207
Additional Example of the Make Validation Column Platform 208

13 Formula Depot 211
Manage Models and Generate Scoring Code

Overview of the Formula Depot Platform 213
Example of Formula Depot 213
Launch the Formula Depot Platform 214
Platforms That Publish Prediction Formulas to the Formula Depot 215
Formula Depot Platform Options 215
Formula Depot Model Options 217
Generating Scoring Code from the Formula Depot Platform 217

14 Fit Curve 221
Fit Built-In Nonlinear Models to Your Data

Overview of the Fit Curve Platform 223
Example Using the Fit Curve Platform 224
Launch the Fit Curve Platform 227
The Fit Curve Report 228
Fit Curve Options 228

Model Comparison Report ... 229
Model Fit Report ... 231
Model Fit Options ... 231
Test Parallelism ... 233
Compare Parameter Estimates ... 236
Equivalence Test ... 237
Statistical Details for the Fit Curve Platform ... 238
Model Formulas ... 238

15 Nonlinear Regression ... 245
Fit Custom Nonlinear Models to Your Data

Example of Fitting a Custom Model ... 247
Launch the Nonlinear Platform ... 250
The Nonlinear Fit Report ... 251
Nonlinear Platform Options ... 255
Create a Formula Using the Model Library ... 259
Customize the Nonlinear Model Library ... 261
Additional Examples ... 262
Example of Maximum Likelihood: Logistic Regression ... 263
Example of a Probit Model with Binomial Errors: Numerical Derivatives ... 264
Example of a Poisson Loss Function ... 265
Example of Setting Parameter Limits ... 267
Statistical Details for the Nonlinear Platform ... 270
Profile Likelihood Confidence Limits ... 271
How Custom Loss Functions Work ... 272
Notes Concerning Derivatives ... 273
Notes on Effective Nonlinear Modeling ... 274

16 Functional Data Explorer ... 277
Perform Functional Data Analysis

Overview of the Functional Data Explorer Platform ... 279
Example of Functional Data Explorer ... 280
Launch the Functional Data Explorer Platform ... 283
The Functional Data Explorer Report ... 286
Model Reports ... 287
Functional Data Explorer Platform Options ... 292
Functional Data Explorer Group Options ... 293
Data Processing Report Options ... 294
Model Report Options ... 297
Additional Examples of the Functional Data Explorer Platform ... 298
Example for Multiple Functional Processes ... 299

Example of Functional DOE ... 303
Statistical Details for the Functional Data Explorer Platform ... 305
Functional Model Fits ... 306
Function Summaries Details ... 307

17 Gaussian Process ... 309
Fit Data Using Smoothing Models
Example of Gaussian Process ... 311
Launch the Gaussian Process Platform ... 313
The Gaussian Process Report ... 314
Actual by Predicted Plot ... 314
Model Report ... 314
Marginal Model Plots ... 315
Gaussian Process Platform Options ... 315
Additional Examples of the Gaussian Process Platform ... 316
Example of a Gaussian Process Model ... 317
Example of Gaussian Process Model with Categorical Predictors ... 318
Statistical Details for the Gaussian Process Platform ... 320
Models with Continuous Predictors ... 320
Models with Categorical Predictors ... 321
Variance Formula Parameterization ... 322
Model Fit Details ... 322

18 Time Series Analysis ... 323
Fit Time Series Models and Transfer Functions
Overview of the Time Series Platform ... 325
Example of the Time Series Platform ... 326
Launch the Time Series Platform ... 328
The Time Series Analysis Report ... 330
Time Series Graph ... 330
Time Series Basic Diagnostics Chart ... 331
Time Series Platform Options ... 333
Time Series Diagnostics ... 333
Differencing and Decomposition ... 334
ARIMA and Seasonal ARIMA Models ... 337
Smoothing Models ... 338
State Space Smoothing Models ... 339
Transfer Function Models ... 340
Smoothing Model Specification Windows ... 343
Reports ... 345
Difference Report ... 345

Decomposition Reports ... 346
Model Comparison Report ... 348
ARIMA and Seasonal ARIMA Model Report ... 351
State Space Smoothing Report ... 355
Transfer Function Report ... 357
Spectral Density Report ... 359
Additional Examples of the Time Series Platform ... 360
Example of Creating Time ID Column ... 360
Example Using Box-Cox Transformation ... 365
Example Using a Holdback Set ... 367
Statistical Details for the Time Series Platform ... 369
Statistical Details for Spectral Density ... 369
Statistical Details for X-11 Decomposition ... 370
Statistical Details for Exponential Smoothing Models ... 370
Statistical Details for ARIMA Models ... 374
Statistical Details for Transfer Functions ... 376

19 Time Series Forecast ... 377
Forecast Multiple Time Series

Example of Time Series Forecast ... 379
Launch the Time Series Forecast Platform ... 381
Data Format ... 381
The Time Series Forecast Report ... 383
Analysis of Time Pattern ... 383
Group Summary of Time ... 383
Group Summary of Y ... 384
Modeling Specifications ... 384
Time Series Forecast Platform Options ... 387
Model Reports ... 389

20 Matched Pairs Analysis ... 391
Compare Measurements on the Same Subject

Overview of the Matched Pairs Platform ... 393
Example of Comparing Matched Pairs ... 393
Launch the Matched Pairs Platform ... 395
Multiple Y Columns ... 395
The Matched Pairs Report ... 396
Difference Plot and Report ... 397
Across Groups ... 397
Matched Pairs Platform Options ... 398
Example Comparing Matched Pairs across Groups ... 399

Statistical Details for the Matched Pairs Platform ... 400
Graphics for Matched Pairs ... 401
Correlation of Responses ... 402

21 Modeling Utilities ... 403
Exploring Data for Outliers, Missing Values, and Patterns

Explore Outliers Utility ... 405
Example of the Explore Outliers Utility ... 405
Launch the Explore Outliers Utility ... 409
Quantile Range Outliers ... 410
Robust Fit Outliers ... 413
Robust PCA Outliers ... 415
K Nearest Neighbor Outliers ... 418
Explore Outliers Utility Options ... 420
Additional Examples of the Explore Outliers Utility ... 421
Explore Missing Values Utility ... 423
Example of the Explore Missing Values Utility ... 424
The Missing Value Report ... 426
Explore Missing Values Utility Options ... 432
Explore Patterns Utility ... 433
Example of the Explore Patterns Utility ... 433
Launch the Explore Patterns Utility ... 436
Explore Patterns Report ... 437
Explore Patterns Utility Options ... 438
Additional Example of the Explore Patterns Utility ... 442
Statistical Details for the Explore Patterns Utility ... 444

22 Response Screening ... 447
Test Many Responses in Large-Scale Data

Overview of the Response Screening Platform ... 449
Example of Response Screening ... 450
Launch the Response Screening Platform ... 453
The Response Screening Report ... 455
Result Table ... 456
FDR PValue Plot ... 458
FDR LogWorth by Effect Size ... 459
FDR LogWorth by RSquare ... 460
The PValues Data Table ... 460
PValues Data Table Scripts ... 461
Response Screening Platform Options ... 461
Means Data Table ... 463

Compare Means Data Table ... 464
The Response Screening Personality in Fit Model ... 466
Launch Response Screening in Fit Model ... 467
The Fit Response Screening Report ... 468
Fit Response Screening Options ... 469
PValues Data Table ... 470
Y Fits Data Table ... 471
Additional Examples of Response Screening ... 472
Example of Tests of Practical Significance and Equivalence ... 472
Example of the MaxLogWorth Option ... 474
Example of Robust Fit ... 475
Response Screening Personality ... 478
Statistical Details for the Response Screening Platform ... 479
The False Discovery Rate ... 479

23 Process Screening ... 481
Screen Many Processes for Stability and Capability

Overview of the Process Screening Platform ... 483
Example of Process Screening ... 483
Launch the Process Screening Platform ... 486
Launch Window Roles ... 486
Launch Window Options ... 487
Limits Table ... 489
The Process Screening Report ... 491
Process Screening Platform Options ... 497
Process Performance Graph ... 502
Additional Example of Process Screening ... 504
Statistical Details for the Process Screening Platform ... 506
Scaling Factors for Using Medians to Estimate Sigma ... 506

24 Predictor Screening ... 509
Screen Many Predictors for Significant Effects

Overview of the Predictor Screening Platform ... 511
Example of Predictor Screening ... 511
Launch the Predictor Screening Platform ... 513
The Predictor Screening Report ... 514
Predictor Screening Platform Options ... 514

25 Association Analysis ... 515
Perform Market Basket Analysis

Overview of the Association Analysis Platform ... 517

Example of the Association Analysis Platform ... 518
Launch the Association Analysis Platform ... 520
Data Format ... 521
The Association Analysis Report ... 522
Frequent Item Sets ... 522
Rules ... 522
Association Analysis Platform Options ... 523
Singular Value Decomposition ... 524
SVD Report ... 525
SVD Report Options ... 526
Topic Analysis ... 527
Topic Analysis Report ... 527
Topic Analysis Report Options ... 528
Additional Example: SVD Analysis ... 529
Statistical Details for the Association Analysis Platform ... 531
Frequent Item Set Generation ... 531
Association Analysis Performance Measures ... 532

26 Process History Explorer ... 535
Identify Problem Components in Complex Process Histories

Example of Process History Explorer ... 537
Launch the Process History Explorer Platform ... 539
Data Format ... 540
The Process History Explorer Report ... 540
Process History Explorer Platform Options ... 541

A Statistical Details ... 545
Predictive and Specialized Modeling

Measures of Fit ... 547
Entropy RSquare ... 547
Validation in JMP Modeling ... 547
Validation Column Role ... 548
K-Fold and Holdback Validation ... 550

B References ... 553

C Technology License Notices ... 557

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 19
JMP Help . . . 20
JMP Documentation Library . . . 20
Additional Resources for Learning JMP . . . 26
JMP Tutorials . . . 26
Sample Data Tables . . . 26
Learn about Statistical and JSL Terms . . . 27
Learn JMP Tips and Tricks . . . 27
JMP Tooltips . . . 27
JMP User Community . . . 28
Free Online Statistical Thinking Course . . . 28
JMP New User Welcome Kit . . . 28
Statistics Knowledge Portal . . . 28
JMP Training . . . 28
JMP Books by Users . . . 29
The JMP Starter Window . . . 29
JMP Technical Support . . . 29

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Introduction to Predictive and Specialized Modeling

Overview of Modeling Techniques

Predictive and Specialized Modeling provides details about more technical modeling techniques, such as Response Screening, Partitioning, and Neural Networks.

- The Neural platform implements a fully connected multi-layer perceptron with one or two layers. Use neural networks to predict one or more response variables using a flexible function of the input variables. See "Neural Networks".
- The Partition platform recursively partitions data according to a relationship between the *X* and *Y* values, creating a decision tree of partitions. See "Partition Models".
- JMP PRO The Bootstrap Forest platform enables you to fit an ensemble model by averaging many decision trees each of which is fit to a random subset of the training data. See "Bootstrap Forest".
- JMP PRO The Boosted Tree platform produces an additive decision tree model that consists of many smaller decision trees that are constructed in layers. The tree in each layer consists of a small number of splits, typically five or fewer. Each layer is fit using the recursive fitting methodology. See "Boosted Tree".
- JMP PRO The K Nearest Neighbors platform predicts a response value for a given observation using the responses of the observations in that observation's local neighborhood. It can be used with a categorical response for classification and with a continuous response for prediction. See "K Nearest Neighbors".
- JMP PRO The Naive Bayes platform classifies observations into groups that are defined by the levels of a categorical response variable. The variables (or factors) that are used for classification are often called features in the data mining literature. See "Naive Bayes".
- JMP PRO The Support Vector Machines platform classifies observations into groups that are defined by levels of a categorical response variable. The model classifies data by optimizing a hyperplane that separates the classes. See "Support Vector Machines".
- JMP PRO The Model Screening platform enables you to quickly run multiple predictive models and compare the results. Measures of fit are provided for each model along with overlaid diagnostic plots. See "Model Screening".
- JMP PRO The Model Comparison platform enables you to compare the predictive ability of different models. Measures of fit are provided for each model along with overlaid diagnostic plots. See "Model Comparison".
- JMP PRO The Make Validation Column platform lets you partition the data into two or three sets, using one of five different methods to create these partitions. See "Make Validation Column".

- JMP PRO The Formula Depot platform enables you to organize, compare, profile, and score models for deployment. For model exploration work, you can use the Formula Depot to store candidate models outside of your JMP data table. See "Formula Depot".
- The Fit Curve platform provides predefined models, such as polynomial, logistic, Gompertz, exponential, peak, and pharmacokinetic models. Compare different groups or subjects using a variety of analytical and graphical techniques. See "Fit Curve".
- The Nonlinear platform lets you fit custom nonlinear models, which include a model formula and parameters to be estimated. See "Nonlinear Regression".
- The Gaussian Process platform models the relationship between a continuous response and one or more continuous predictors. These models are common in areas like computer simulation experiments, such as the output of finite element codes, and they often perfectly interpolate the data. See "Gaussian Process".
- JMP PRO The Functional Data Explorer platform enables you to convert functional data into a form that can be analyzed in another JMP platform. See "Functional Data Explorer".
- The Time Series platform lets you explore, analyze, and forecast univariate time series. See "Time Series Analysis".
- The Time Series Forecast platform lets you model and forecast multiple time series. The best fitting model is automatically selected from a set of up to 30 exponential smoothing models. See "Time Series Forecast".
- The Matched Pairs platform compares the means between two or more correlated variables and assesses the differences. See "Matched Pairs Analysis".
- The Modeling Utilities assist in the data cleaning and pre-processing stages of data analysis. Each utility has exploratory tools to give you a more thorough understanding of your data. See "Modeling Utilities".
- The Response Screening platform automates the process of conducting tests across a large number of responses. Your test results and summary statistics are presented in data tables, rather than reports, to enable data exploration. See "Response Screening".
- The Process Screening platform enables you to explore a large number of processes across time. The platform calculates control chart, process stability, and process capability metrics, and detects large process shifts. See "Process Screening".
- The Predictor Screening platform enables you to screen a data set for significant predictors. See "Predictor Screening".
- JMP PRO The Association Analysis platform enables you to identify items that have an affinity for each other. It is frequently used to analyze transactional data (also called market baskets) to identify items that often appear together in transactions. See "Association Analysis".
- The Process History Explorer platform enables you to identify problem components in complex process histories. See "Process History Explorer".

Chapter 3

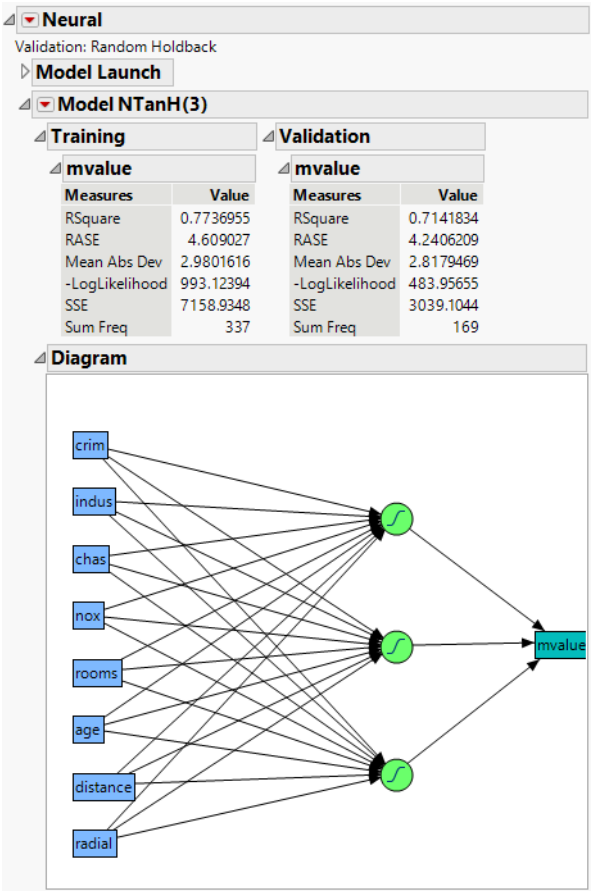
Neural Networks

Fit Nonlinear Models Using Nodes and Layers

JMP PRO *Most features in this platform are available only in JMP Pro and noted with this icon.*

The Neural platform implements a fully connected multi-layer perceptron with one or two layers. Use neural networks to predict one or more response variables using a flexible function of the input variables. Neural networks can be very good predictors when it is not necessary to describe the functional form of the response surface, or to describe the relationship between the inputs and the response.

Figure 3.1 Example of a Neural Network



Contents

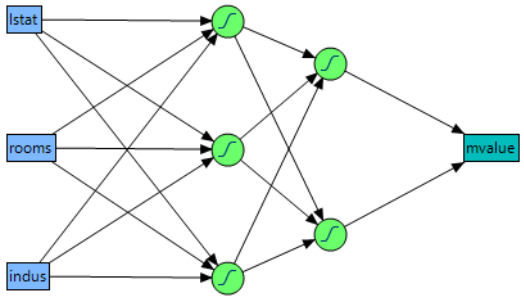
Overview of the Neural Platform 35
Launch the Neural Platform 35
The Neural Launch Window 36
The Model Launch Control Panel 37
Validation Methods for Neural 38
Hidden Layer Structure 39
Boosting 40
Fitting Options 41
Model Reports 43
Training and Validation Measures of Fit 43
Confusion Statistics 44
Neural Platform Options 44
Neural Model Options 45
Example of a Neural Network 46

Overview of the Neural Platform

Think of a neural network as a function of a set of derived inputs, called hidden nodes. The hidden nodes are nonlinear functions of the original inputs. You can specify up to two layers of hidden nodes, where each layer can contain as many hidden nodes as you want.

Figure 3.2 shows a two-layer neural network with three X variables and one Y variable. In this example, the first layer has two nodes, and each node is a function of all three nodes in the second layer. The second layer has three nodes, and all nodes are a function of the three X variables. The predicted Y variable is a function of both nodes in the first layer.

Figure 3.2 Neural Network Diagram



The functions applied at the nodes of the hidden layers are called activation functions. The activation function is a transformation of a linear combination of the X variables. For more information about the activation functions, see "Hidden Layer Structure". The function applied at the response is a linear combination (for continuous responses), or a logistic transformation (for nominal or ordinal responses).

The main advantage of a neural network model is that it can efficiently model different response surfaces. Given enough hidden nodes and layers, any surface can be approximated to any accuracy. The main disadvantage of a neural network model is that the results are not easily interpretable. This is because there are intermediate layers rather than a direct path from the X variables to the Y variables, as in the case of regular regression.

Launch the Neural Platform

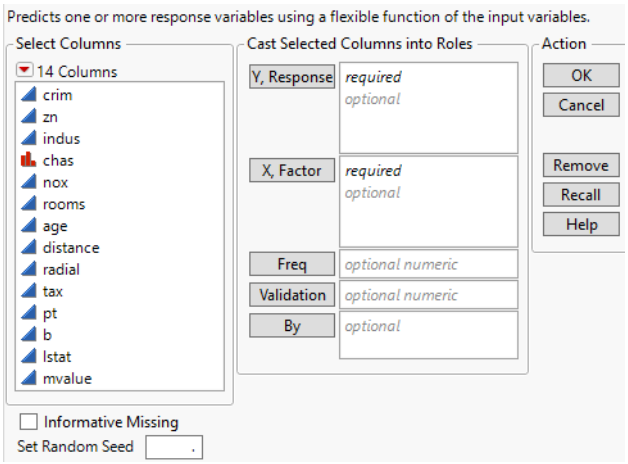
To launch the Neural platform, select **Analyze** > **Predictive Modeling** > **Neural**.

Launching the Neural platform is a two-step process. First, enter your variables on the Neural launch window. Second, specify your options in the Model Launch control panel.

JMP PRO The Neural Launch Window

Use the Neural launch window to specify X and Y variables, a validation column, and to enable Informative Missing value coding.

Figure 3.3 The Neural Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Response The response variable or variables that you want to analyze. When multiple responses are specified, the models for the responses share all parameters in the hidden layers (those parameters not connected to the responses). The responses are conditionally independent given the predictor variables, but are marginally correlated through those variables to create one overall neural model.

X, Factor The predictor variables.

Freq A column whose numeric values assign a frequency to each row in the analysis.

JMP PRO **Validation** A numeric column that defines the validation sets. See "Validation Methods for Neural". If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

JMP PRO **Informative Missing** Check this box to enable informative coding of missing values. This coding allows estimation of a predictive model despite the presence of missing values. It is useful in situations where missing data are informative. If this option is not checked, rows with missing values are ignored.

For a continuous variable, missing values are replaced by the mean of the variable. Also, a missing value indicator, named <colname> Is Missing, is created and included in the model. If a variable is transformed using the Transform Covariates fitting option on the Model Launch control panel, missing values are replaced by the mean of the transformed variable.

For a categorical variable, missing values are treated as a separate level of that variable.

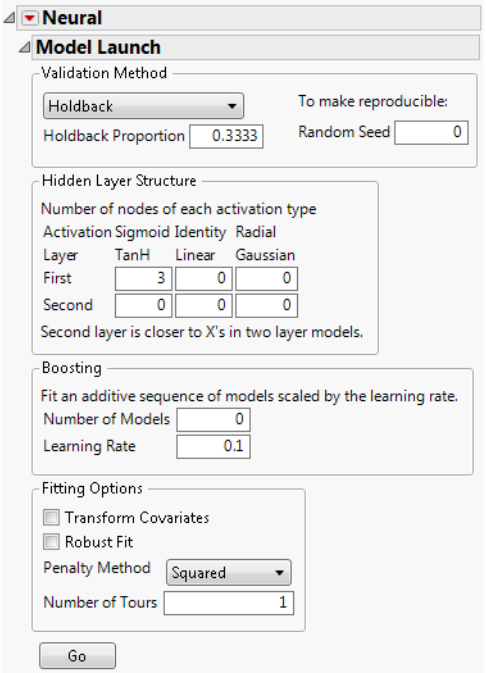
Set Random Seed Sets the seed for the starting values used in the fitting procedure. Set Random Seed is useful if you want to reproduce an analysis. If you set a random seed and save the script, the seed is automatically saved in the script.

The Model Launch Control Panel

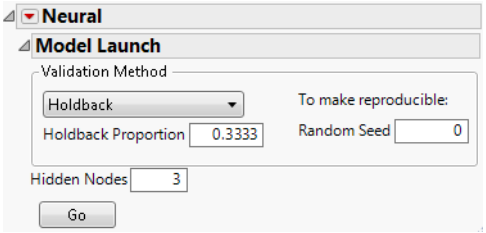
Use the Model Launch control panel to specify the validation method, the structure of the hidden layer, whether to use gradient boosting, and other fitting options.

Figure 3.4 The Model Launch Control Panel

JMP Pro



JMP



Validation Method Select the method that you want to use for model validation. See "Validation Methods for Neural".

Random Seed Specify a nonzero numeric random seed if you want to reproduce the validation assignment for future launches of the Neural platform. By default, the Random Seed is set to zero, which does not produce reproducible results. When you save the analysis to a script, the random seed that you enter is saved to the script.

Hidden Layer Structure or Hidden Nodes Specify the number of hidden nodes of each type in each layer. See "Hidden Layer Structure".

Note: The standard edition of JMP uses only the TanH activation function, and can fit only neural networks with one hidden layer.

JMP PRO **Boosting** Specify options for gradient boosting. See "Boosting".

JMP PRO **Fitting Options** Specify options for variable transformation and model fitting. See "Fitting Options".

Go Fits the neural network model and shows the model reports.

After you click **Go** to fit a model, you can reopen the Model Launch Control Panel and change the settings to fit another model.

Validation Methods for Neural

Neural utilizes a validation method in its fitting routine. Choices of validation method include holdback, K-fold, or (in JMP Pro) the use of a validation column. To fit a model, the Neural platform does the following:

- applies a penalty on the model parameters
- uses the validation set to tune the penalties on the parameters.

Select one of the following validation methods:

Excluded Rows Holdback Uses row states to subset the data. Rows that are unexcluded are used as the training set, and excluded rows are used as the validation set.

For more information about using row states and how to exclude rows, see *Using JMP*.

Holdback Randomly divides the original data into the training and validation sets. You specify the proportion of the original data to use as the validation set (holdback). The random selection is based on stratified sampling across the model factors to attempt to create training and validation sets that are more balanced than ones based on simple random sampling.

KFold Divides the data into K subsets, or folds. In turn, each of the K folds is used to validate the model fit on the rest of the data, fitting a total of K models. The final model is for the fold that resulted in the model with the best validation statistic.

This method is useful for small data sets, because it makes efficient use of limited amounts of data.

JMP PRO **Validation Column** Uses a numeric column that defines the validation sets. The column's values determine how the data is split:

- If the validation column has two levels, the smaller value defines the training set and the larger value defines the validation set.
- If the validation column has three levels, the values, in order of increasing size, define the training, validation, and test sets.
- If the validation column has more than three unique values, then K-Fold validation is performed. The number of folds is determined by the number of values in the validation column.

The Neural platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

Hidden Layer Structure

Note: The standard edition of JMP uses only the TanH activation function, and can fit only neural networks with one hidden layer.

The Neural platform can fit one or two-layer neural networks. Increasing the number of nodes in the first layer, or adding a second layer, makes the neural network more flexible. You can add an unlimited number of nodes to either layer. The second layer nodes are functions of the X variables. The first layer nodes are functions of the second layer nodes. The Y variables are functions of the first layer nodes.

Caution: You cannot use boosting with a two-layer neural network. If you specify any non-zero values in the second layer and also specify boosting, the second layer is ignored.

The functions applied at the nodes of the hidden layers are called activation functions. An activation function is a transformation of a linear combination of the X variables. The following activation functions are available:

TanH The hyperbolic tangent function is a sigmoid function. TanH transforms values to be between -1 and 1, and is the centered and scaled version of the logistic function. The hyperbolic tangent function is:

$$\frac{e^{2x}-1}{e^{2x}+1}$$

where x is a linear combination of the X variables.

Linear The identity function. The linear combination of the X variables is not transformed.

The Linear activation function is most often used in conjunction with one of the non-linear activation functions. In this case, the Linear activation function is placed in the second layer, and the non-linear activation functions are placed in the first layer. This is useful if you want to first reduce the dimensionality of the X variables, and then have a nonlinear model for the Y variables.

For a continuous Y variable, if only Linear activation functions are used, the model for the Y variable reduces to a linear combination of the X variables. For a nominal or ordinal Y variable, the model reduces to a logistic regression.

Gaussian The Gaussian function. Use this option for radial basis function behavior, or when the response surface is Gaussian (normal) in shape. The Gaussian function is:

$$e^{-x^2}$$

where x is a linear combination of the X variables.

Boosting

JMP PRO Boosting is the process of building a large additive neural network model by fitting a sequence of smaller models. Each of the smaller models is fit on the scaled residuals of the previous model. The models are combined to form the larger final model. The process uses validation to assess how many component models to fit, not exceeding the specified number of models.

Boosting is often faster than fitting a single large model. However, the base model should be a 1 to 2 node single-layer model. The benefit of faster fitting can be lost if a large number of models is specified.

Use the Boosting panel in the Model Launch control panel to specify the number of component models and the learning rate. Use the Hidden Layer Structure panel in the Model Launch control panel to specify the structure of the base model.

The learning rate must be $0 < r \leq 1$. Learning rates close to 1 result in faster convergence on a final model, but also have a higher tendency to overfit data. Use learning rates close to 1 when a small Number of Models is specified.

As an example of how boosting works, suppose you specify a base model consisting of one layer and two nodes, with the number of models equal to eight. The first step is to fit a one-layer, two-node model. The predicted values from that model are scaled by the learning rate, then subtracted from the actual values to form a scaled residual. The next step is to fit a different one-layer, two-node model, where the response values are the scaled residuals of the previous model. This process continues until eight models are fit, or until the addition of another model fails to improve the validation statistic. The component models are combined to form the final, large model. In this example, if six models are fit before stopping, the final model consists of one layer and 2 x 6 = 12 nodes.

Caution: You cannot use boosting with a two-layer neural network. If you specify any non-zero values in the second layer and also specify boosting, the second layer is ignored.

Fitting Options

JMP PRO The following model fitting options are available:

Transform Covariates Transforms all continuous variables to near normality using either the Johnson Su or Johnson Sb distribution. Transforming the continuous variables helps mitigate the negative effects of outliers or heavily skewed distributions. See the Save Transformed Covariates option in "Neural Model Options".

Robust Fit Trains the model using least absolute deviations instead of least squares. This option is useful if you want to minimize the impact of response outliers. This option is available only for continuous responses.

Penalty Method Choose the penalty method. To mitigate the tendency neural networks have to overfit data, the fitting process incorporates a penalty on the likelihood. See "Penalty Method".

Number of Tours Specify the number of times to restart the fitting process, with each iteration using different random starting points for the parameter estimates. The iteration with the best validation statistic is chosen as the final model.

Penalty Method

The penalty is $\lambda p(\beta_i)$, where λ is the penalty parameter, and $p(\ )$ is a function of the parameter estimates, called the penalty function. Validation is used to find the optimal value of the penalty parameter.

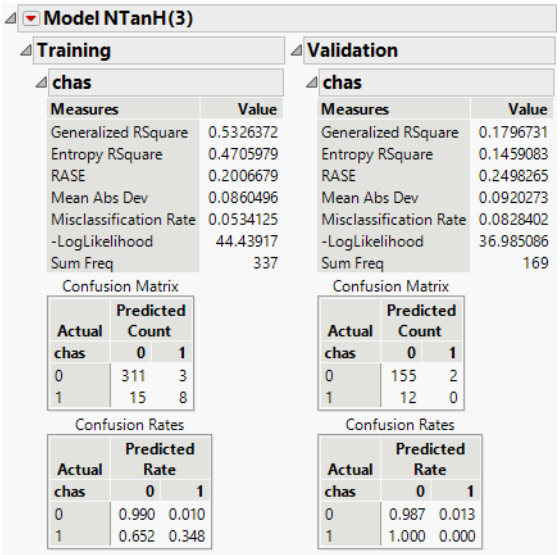
Table 3.1 Descriptions of Penalty Methods

Method	Penalty Function	Description
Squared	$\sum \beta_i^2$	Use this method if you think that most of your X variables are contributing to the predictive ability of the model.
Absolute	$\sum \lvert\beta_i\rvert$	Use either of these methods if you have a large number of X variables, and you think that a few of them contribute more than others to the predictive ability of the model.
Weight Decay	$\sum \frac{\beta_i^2}{1+\beta_i^2}$	
NoPenalty	none	Does not use a penalty. You can use this option if you have a large amount of data and you want the fitting process to go quickly. However, this option can lead to models with lower predictive performance than models that use a penalty.

Model Reports

A model report is created for every neural network model. Measures of fit appear for the training and validation sets. In addition, confusion statistics appear when the response is nominal or ordinal.

Figure 3.5 Example of a Neural Model Report



Model NTanH(3)

Training

chas

Measures	Value
Generalized RSquare	0.5326372
Entropy RSquare	0.4705979
RASE	0.2006679
Mean Abs Dev	0.0860496
Misclassification Rate	0.0534125
-LogLikelihood	44.43917
Sum Freq	337

Confusion Matrix

Actual chas	Predicted Count 0	1
0	311	3
1	15	8

Confusion Rates

Actual chas	Predicted Rate 0	1
0	0.990	0.010
1	0.652	0.348

Validation

chas

Measures	Value
Generalized RSquare	0.1796731
Entropy RSquare	0.1459083
RASE	0.2498265
Mean Abs Dev	0.0920273
Misclassification Rate	0.0828402
-LogLikelihood	36.985086
Sum Freq	169

Confusion Matrix

Actual chas	Predicted Count 0	1
0	155	2
1	12	0

Confusion Rates

Actual chas	Predicted Rate 0	1
0	0.987	0.013
1	1.000	0.000

Training and Validation Measures of Fit

Measures of fit appear for the training and validation sets (Figure 3.5).

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The value is 1 for a perfect model, and 0 for a model no better than a constant model. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991).

Entropy RSquare (Appears only when the response is nominal or ordinal.) A measure of fit that compares the log-likelihoods from the fitted model and the constant probability model. Entropy RSquare ranges from 0 to 1, where values closer to 1 indicate a better fit. See "Entropy RSquare".

RSquare Gives the RSquare for the model.

RASE Gives the root average squared error. When the response is nominal or ordinal, the differences are between 1 and p (the fitted probability for the response level that actually occurred).

Mean Abs Dev The average of the absolute values of the differences between the response and the predicted response. When the response is nominal or ordinal, the differences are between 1 and p (the fitted probability for the response level that actually occurred).

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category. Appears only when the response is nominal or ordinal.

-LogLikelihood Gives the negative of the log-likelihood. See *Fitting Linear Models*.

SSE Gives the error sums of squares. Available only when the response is continuous.

Sum Freq Gives the number of observations that are used. If you specified a Freq variable in the Neural launch window, Sum Freq gives the sum of the frequency column.

If there are multiple responses, fit statistics are given for each response, and an overall Generalized RSquare and negative Log-Likelihood is given.

Confusion Statistics

For nominal or ordinal responses, a Confusion Matrix report and Confusion Rates report is given (Figure 3.5). The Confusion Matrix report shows a two-way classification of the actual response levels and the predicted response levels. For a categorical response, the predicted level is the one with the highest predicted probability. The Confusion Rates report is equal to the Confusion Matrix report, with the numbers divided by the row totals.

Neural Platform Options

The Neural red triangle menu contains the following options:

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Neural Model Options

Each model report has a red triangle menu that contains options for producing additional output or saving results. The model report red triangle menu contains the following options:

Diagram Shows a diagram representing the hidden layer structure.

Show Estimates Shows the parameter estimates in a report.

Profiler Launches the Prediction Profiler. For nominal or ordinal responses, each response level is represented by a separate row in the Prediction Profiler. For more information about the options in the red triangle menu, see *Profilers*.

Categorical Profiler Launches the Prediction Profiler. Similar to the Profiler option, except that all categorical probabilities are combined into a single profiler row. Available only for nominal or ordinal responses. For more information about the options in the red triangle menu, see *Profilers*.

Contour Profiler Launches the Contour Profiler. This is available only when the model contains more than one continuous factor. For more information about the options in the red triangle menu, see *Profilers*.

Surface Profiler Launches the Surface Profiler. This is available only when the model contains more than one continuous factor. For more information about the options in the red triangle menu, see *Profilers*.

ROC Curve Creates an ROC curve. Available only for nominal or ordinal responses. For more information about ROC Curves, see "ROC Curve".

Lift Curve Creates a lift curve. Available only for nominal or ordinal responses. For more information about Lift Curves, see "Lift Curve".

Plot Actual by Predicted Plots the actual versus the predicted response. Available only for continuous responses.

Plot Residual by Predicted Plots the residuals versus the predicted responses. Available only for continuous responses.

Save Formulas Creates new columns in the data table containing formulas for the predicted response and the hidden layer nodes.

Save Profile Formulas Creates new columns in the data table containing formulas for the predicted response. Formulas for the hidden layer nodes are embedded in this formula.

This option produces formulas that can be used by the Interactive HTML version of the Profiler.

Save Fast Formulas Creates new columns in the data table containing formulas for the predicted response. Formulas for the hidden layer nodes are embedded in this formula. This option produces formulas that evaluate faster than the other options, but cannot be used in the Interactive version of the Profiler.

JMP PRO **Publish Prediction Formula** Creates prediction formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot".

Make SAS Data Step Creates SAS code that you can use to score a new data set.

Save Validation Creates a new column in the data table that identifies which rows were used in the training and validation sets. This option is not available when a Validation column is specified on the Neural launch window. See "The Neural Launch Window".

JMP PRO **Save Transformed Covariates** Creates new columns in the data table showing the transformed covariates. The columns contain formulas that show the transformations. This option is available only when the Transform Covariates option is checked on the Model Launch control panel. See "Fitting Options".

Remove Fit Removes the entire model report.

Example of a Neural Network

This example uses the Boston Housing.jmp data table. Suppose you want to create a model to predict the median home value as a function of several demographic characteristics. Follow the steps below to build the neural network model:

1. Select **Help > Sample Data Library** and Boston Housing.jmp.
2. Launch the Neural platform by selecting **Analyze** > **Predictive Modeling** > **Neural**.
3. Assign mvalue to the **Y, Response** role.
4. Assign the other columns (crim through lstat) to the **X, Factor** role.
5. Click **OK**.
6. Enter 0.2 for the **Holdback Proportion**.
7. Enter 1234 for the **Random Seed**.

Note: In general, results vary due to the random nature of choosing a validation set. Entering the seed above enable you to reproduce the results shown in this example.

8. Enter 3 for the number of TanH nodes in the first layer.
9. Check the **Transform Covariates** option.
10. Click **Go**.

Figure 3.6 Neural Report

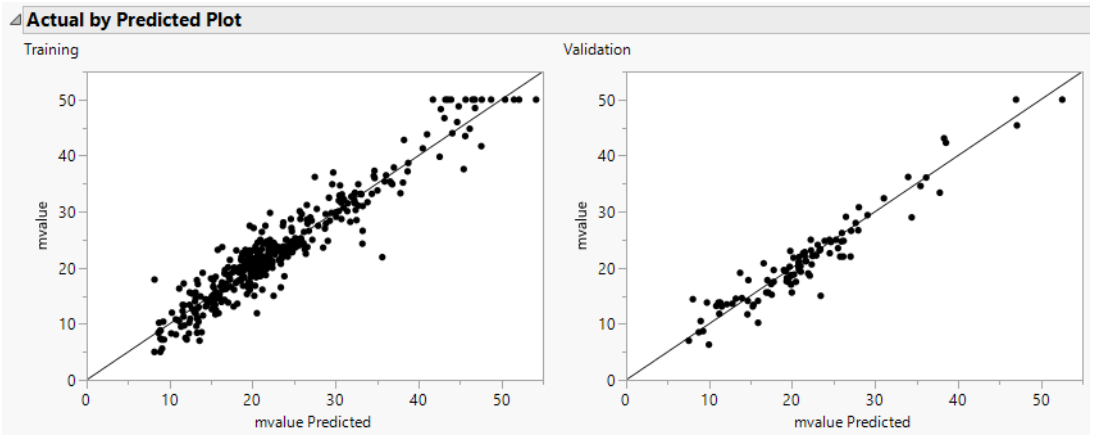
Model NTanH(3)

Training		Validation	
mvalue		mvalue	
Measures	Value	Measures	Value
RSquare	0.9074859	RSquare	0.9125961
RASE	2.8561453	RASE	2.4304519
Mean Abs Dev	2.1278654	Mean Abs Dev	1.8284984
-LogLikelihood	997.23823	-LogLikelihood	235.31561
SSE	3295.6567	SSE	602.52384
Sum Freq	404	Sum Freq	102

Results are provided for both the training and validation sets. Use the results of the validation set as a representation of the model's predictive power on future observations.

The R-Square statistic for the Validation set is 0.913, signifying that the model is predicting well on data not used to train the model. As an additional assessment of model fit, click the red triangle next to Model NTanH(3) and select **Plot Actual by Predicted**.

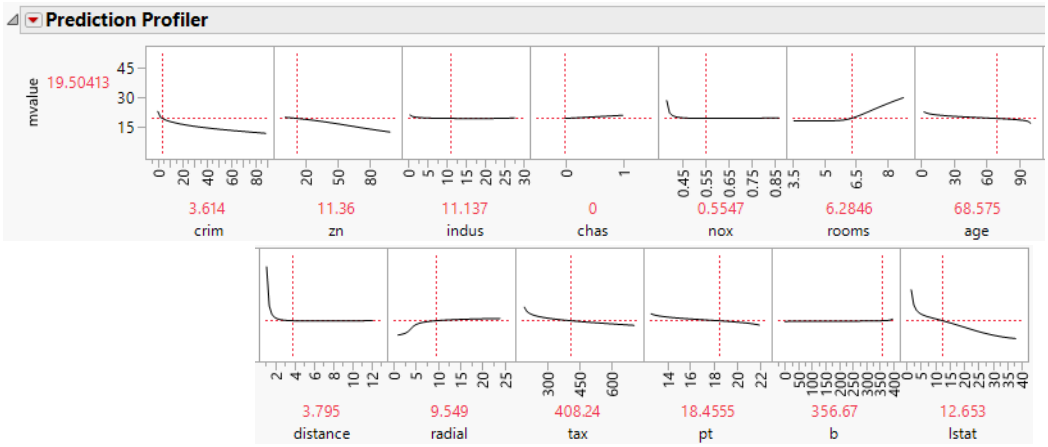
Figure 3.7 Actual by Predicted Plot



The points fall along the line, signifying that the predicted values are similar to the actual values.

To get a general understanding of how the *X* variables are impacting the predicted values, click the red triangle next to Model NTanH(3) and select **Profiler**.

Figure 3.8 Profiler



Some of the variables have profiles with positive slopes, and some negative. For example, the variable rooms has a positive slope. This indicates that the more rooms a home has, the higher the predicted median value. The variable pt is the pupil teacher ratio by town. This variable has a negative slope, indicating that the higher the pupil to teacher ratio, the lower the median value.

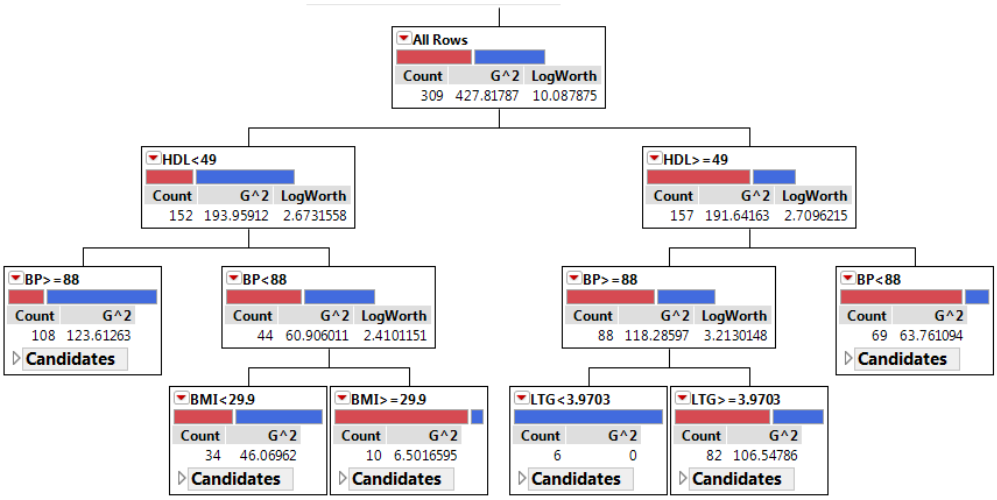
Chapter 4

Partition Models

Use Decision Trees to Explore and Model Your Data

The Partition platform recursively partitions data according to a relationship between the predictors and response values, creating a decision tree. The partition algorithm searches all possible splits of predictors to best predict the response. These splits (or *partitions*) of the data are done recursively to form a tree of decision rules. The splits continue until the desired fit is reached. The partition algorithm chooses optimum splits from a large number of possible splits, making it a powerful modeling, and data discovery tool.

Figure 4.1 Example of a Decision Tree



Contents

Overview of the Partition Platform 51
Example of the Partition Platform 51
Launch the Partition Platform 54
The Partition Report 55
Control Buttons 55
Report for Categorical Responses 56
Report for Continuous Responses 58
Partition Platform Options 61
Show Fit Details 64
Specify Profit Matrix 65
Decision Matrix Report 67
Informative Missing 68
Actual by Predicted Plot 69
ROC Curve 70
Lift Curve 71
Node Options 72
Validation in Partition 73
Additional Examples of Partitioning 74
Example of a Continuous Response 74
Example of Informative Missing 77
Example of Profit Matrix and Decision Matrix Report 80
Statistical Details for the Partition Platform 83
Responses and Factors 83
Splitting Criterion 84
Predicted Probabilities in Decision Tree and Bootstrap Forest 84

Overview of the Partition Platform

The Partition platform recursively partitions data according to a relationship between the predictors and response values, creating a decision tree. Variations of partitioning go by many names and brand names: decision trees, CART™, CHAID™, C4.5, C5, and others. The technique is often considered as a data mining technique for the following reasons:

- it is useful for exploring relationships without having a good prior model
- it handles large problems easily
- the results are interpretable

A classic application of partitioning is to create a diagnostic heuristic for a disease. Given symptoms and outcomes for a number of subjects, partitioning can be used to generate a hierarchy of questions to help diagnose new patients.

Predictors can be either continuous or categorical (nominal or ordinal). If a predictor is continuous, then the splits are created by a *cutting value*. The sample is divided into values below and above this cutting value. If a predictor is categorical, then the sample is divided into two groups of levels.

The response can also be either continuous or categorical (nominal or ordinal). If the response is continuous, then the platform fits the means of the response values and the split is chosen to minimize the sum of squared errors. If the response is categorical, then the fitted value is a probability for the levels of the response and the split is chosen to minimize the residual log-likelihood chi-square.

For more information about split criteria, see "Statistical Details for the Partition Platform".

For more information about recursive partitioning, see Hawkins and Kass (1982) and Kass (1980).

Example of the Partition Platform

In this example, you use the Partition platform to construct a decision tree that predicts the one-year disease progression (low or high) of patients with diabetes.

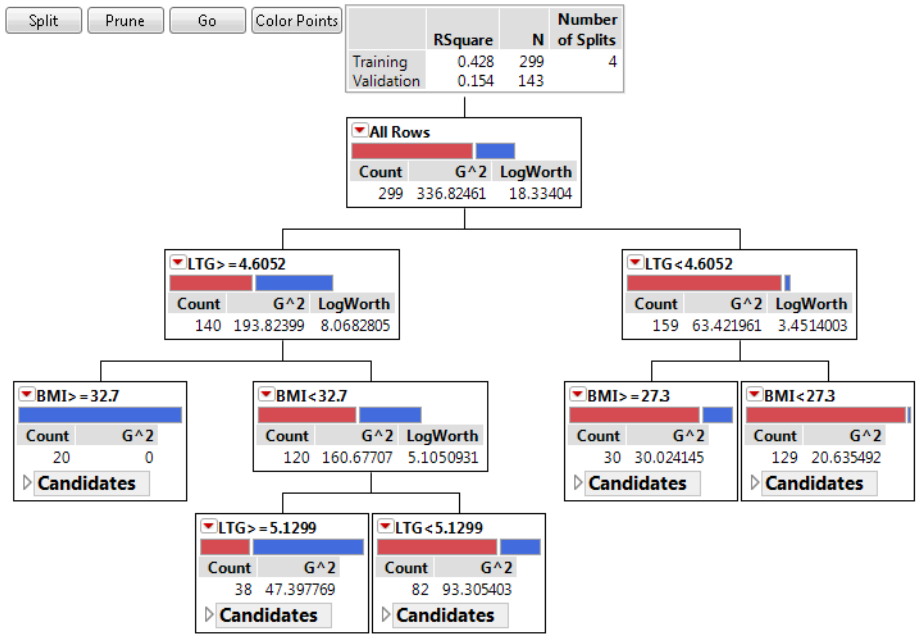
1. Select **Help > Sample Data Library** and Diabetes.jmp.
2. Select **Analyze** > **Predictive Modeling** > **Partition**.
3. Select Y Binary and click **Y, Response**.
4. Select Age through Glucose and click **X, Factor**.
5. Enter 0.33 for the **Validation Portion**.

Note: In JMP Pro, a validation column can be used for validation. Select Validation and click **Validation**. Set the **Validation Portion** to 0.

6. Click **OK**.
7. On the platform report window, click **Go** to perform automatic splitting.

Note: Because you are using a random Validation Portion, your results differ from those in Figure 4.2.

Figure 4.2 Partition Report for Diabetes



Automatic splitting resulted in four splits. The final RSquare for the Validation set is 0.154. The decision tree shows the four splits and the counts of observations in each split.

8. Click the red triangle next to Partition for Y Binary and select **Column Contributions**.

Figure 4.3 Column Contributions Report

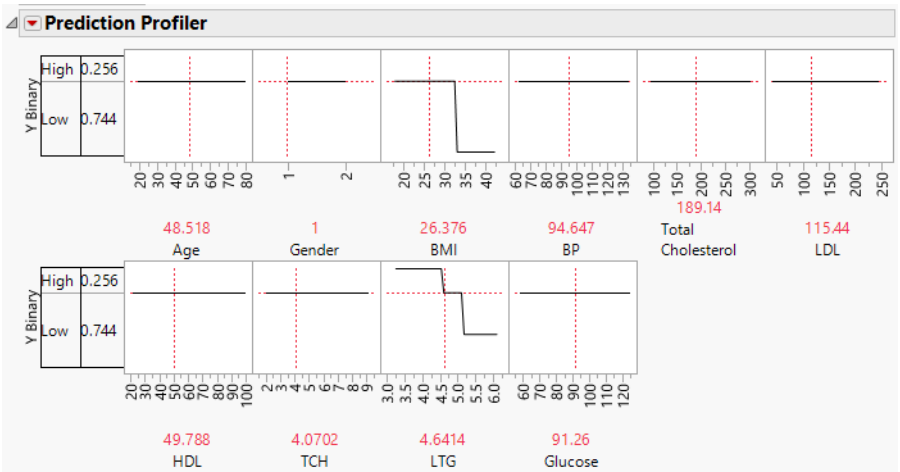
Column Contributions

Term	Number of Splits	G^2	Portion
LTG	2	99.5525528	0.6844
BMI	2	45.9092475	0.3156
Age	0	0	0.0000
Gender	0	0	0.0000
BP	0	0	0.0000
Total Cholesterol	0	0	0.0000
LDL	0	0	0.0000
HDL	0	0	0.0000
TCH	0	0	0.0000
Glucose	0	0	0.0000

The Column Contributions report shows that LTG and BMI are the only predictors in the decision tree model. Each column is used in two splits. Your results can differ. When the Validation Portion is used, the validation set is selected at random from the data table. If you redo your analysis, a new random validation set is selected and your results can differ from your first run.

9. Click the red triangle next to Partition for Y Binary and select **Save Columns > Profiler.**

Figure 4.4 Profiler for Partition Model



The profiler enables you to change the values of BMI and LTG to obtain the predicted Y Binary outcome. No other factors had a split in the partition model. Their profiles are flat lines.

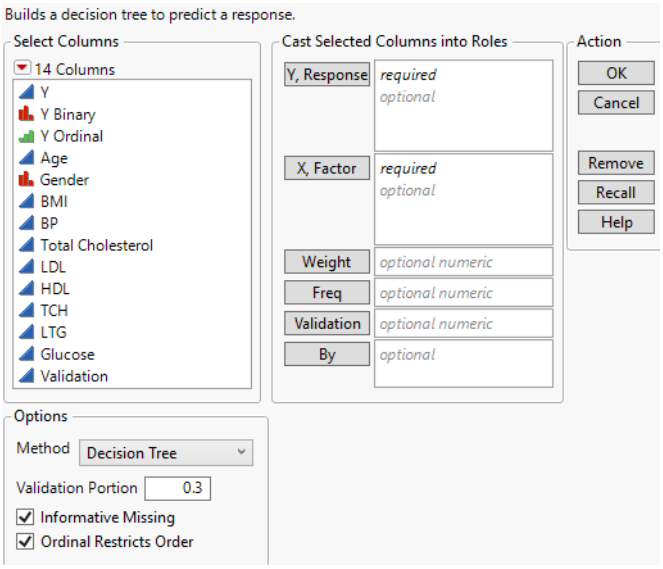
10. Click the red triangle next to Partition for Y Binary and select **Save Columns > Save Prediction Formula**.

In the Diabetes.jmp data table, columns called Prob(Y Binary==Low), Prob(Y Binary==High), and Most Likely Y Binary are added. To see how these response probabilities are calculated, in the Columns panel, next to each column, double-click the Formula icon .

Launch the Partition Platform

Launch the Partition platform by selecting **Analyze** > **Predictive Modeling** > **Partition**.

Figure 4.5 Partition Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Response The response variable or variables that you want to analyze.

X, Factor The predictor variables.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

JMP PRO **Validation** A numeric column that defines the validation sets. See "Validation in Partition". If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

By A column or columns whose levels define separate analyses. For each level of the column, the corresponding rows are analyzed using the other variables that you specify. The results appear in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

JMP PRO **Method** Enables you to select the partition method (Bootstrap Forest, Boosted Tree, K Nearest Neighbors, or Naive Bayes).

For more information about these methods, see "Bootstrap Forest", "Boosted Tree", "K Nearest Neighbors", and "Naive Bayes".

Validation Portion The portion of the data to be used as the validation set. See "Validation in Partition".

Informative Missing If selected, enables missing value categorization for categorical predictors and informative treatment of missing values for continuous predictors. See "Informative Missing".

Ordinal Restricts Order If selected, restricts consideration of splits to those that preserve the ordering.

The Partition Report

The initial Partition report shows a partition plot, control buttons, a summary panel, and a decision tree. The partition plot and decision tree are initialized without any splits. The reports details are different for categorical and continuous responses.

- "Control Buttons"
- "Report for Categorical Responses"
- "Report for Continuous Responses"

Control Buttons

Use the control buttons to interact with the decision tree.

Split Creates a partition of the data using the optimal split. To specify multiple splits, hold the Shift key as you click **Split**.

Prune Removes the worst split.

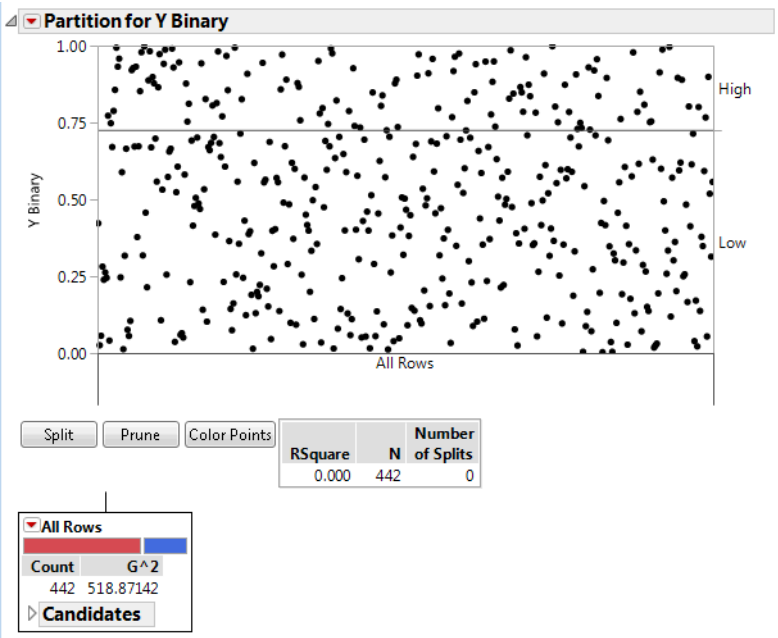
Go (Available when you are using validation.) Automatically adds splits to the decision tree until the validation statistic is optimized. See "Validation in Partition". Without validation, you simply decide the number of splits to use in the partition model.

Color Points For categorical responses, colors observations according to response level. These colors are added to the data table.

Report for Categorical Responses

The sample data table Diabetes.jmp was used to create a report for the categorical response Y Binary.

Figure 4.6 Partition Report for a Categorical Response



Partition Plot

Each point in the Partition Plot represents an observation in the data table. If validation is used, the plot is only for the training data. The initial partition plot does not show splits.

Notice the following:

- The left vertical axis is the proportion of each response outcome.
- The right vertical axis shows the order in which the response levels are plotted.
- Horizontal lines divide each split by the response variable. The initial horizontal line shows the overall proportion of the first plotted response in the data set.
- Splits are shown below the X axis with a text description and a vertical line that splits the observations in the plot. The vertical lines extend into the plot and indicate the boundaries for each node. The most recent split appears directly below the horizontal axis and on top of existing splits. The plot is updated with each split or prune of the decision tree.

Summary Report

Figure 4.7 Summary Report for a Categorical Response

	RSquare	N	Number of Splits
Training	0.428	299	4
Validation	0.154	143	

The Summary Report provides fit statistics for the training data and validation and test data (if used). The fit statistics in the Summary Panel update as you add splits or prune the decision tree.

RSquare The current value of R^2.

N The number of observations.

Number of Splits The current number of splits in the decision tree.

Node Reports

Each node in the tree has a report and a red triangle menu with additional options. Terminal nodes also have a Candidates report.

Figure 4.8 Terminal Node Report for a Categorical Response

All Rows

Count	G^2
442	518.87142

Candidates

Term	Candidate G^2		LogWorth	Cut Point
Age	10.5000264		1.71376465	51
Gender	1.8302510		0.75424581	2
BMI	92.8760803		31.25572705	27.3
BP	64.8300680		18.98689929	100
Total Cholesterol	20.3048623		4.01316712	194
LDL	12.5858490		0.82750128	122.2
HDL	44.2535587		11.48909721	46
TCH	64.3516993		17.86426783	4
LTG	102.8078418	*	35.97159929	4.8203
Glucose	43.2683018		11.05809993	99

Count Number of training observations that are characterized by the node.

G^2 A fit statistic used for categorical responses (instead of sum of squares that is used for continuous responses). Lower values indicate a better fit. See "Statistical Details for the Partition Platform".

Candidates For each column, the Candidates report provides details about the optimal split for that column. The optimal split over all terms is marked with an asterisk.

Term Shows the candidate columns.

Candidate G^2 Likelihood ratio chi-square for the best split. Splitting on the predictor with the largest G^2 maximizes the reduction in the model G^2.

LogWorth The LogWorth statistic, defined as $-\log_{10}(p\text{-value})$. The optimal split is the one that maximizes the LogWorth. See "Statistical Details for the Partition Platform".

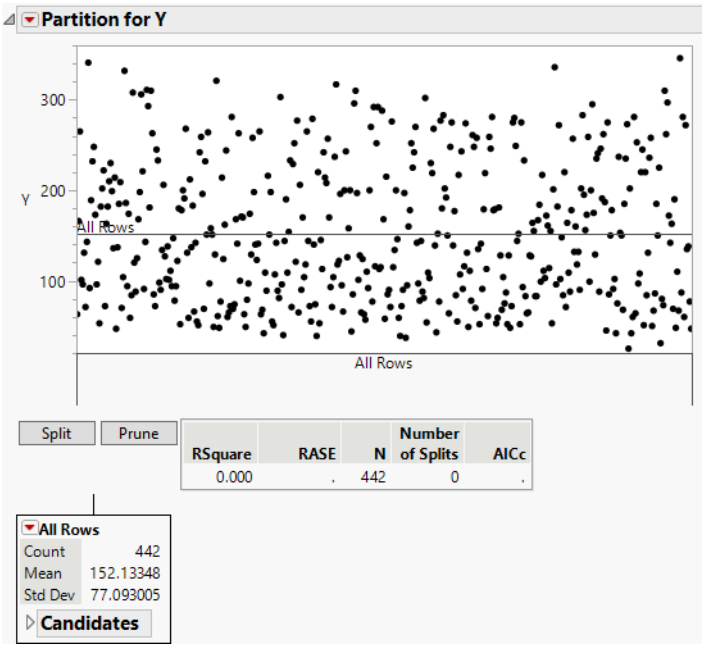
Cut Point The value of the predictor that determines the split. For a categorical term, the levels in the left-most split are listed.

The optimal split is noted by an asterisk. However, there are cases where the Candidate G^2 is higher for one variable, but the Logworth is higher for a different variable. In this case > and < are used to point in the best direction for each variable. The asterisk corresponds to the condition where they agree. See "Statistical Details for the Partition Platform".

Report for Continuous Responses

The sample data table Diabetes.jmp was used to create a report for the continuous response Y.

Figure 4.9 Partition Report for a Continuous Response



Partition Plot

The partition plot is initialized without any splits. Each point represents an observation in the data table. If validation is used, the plot is only for the training data.

Notice the following:

- The vertical axis represents the response value of the observations.
- Horizontal lines show the mean response value for each node of the decision tree. The initial horizontal line is at the overall mean of the response.
- Vertical axis divisions represent splits in the decision tree. A text description of the most recent split appears below the horizontal axis. Observations are reorganized into their respective nodes as splits are created or removed.

Tip: To see tooltips for narrow partitions, hover over the labels on the horizontal axis of the partition plot.

Summary Report

Figure 4.10 Summary Report for a Continuous Response

	RSquare	RASE	N	Number of Splits	AICc
Training	0.490	54.690663	299	4	3253.83
Validation	0.366	61.16064	143		

The Summary Report provides fit statistics for the training data and validation and test data (if used). The fit statistics in the Summary Panel update as you add splits or prune the decision tree.

RSquare The current value of R^2.

RASE The root average square error. See *Fitting Linear Models*.

N The number of observations.

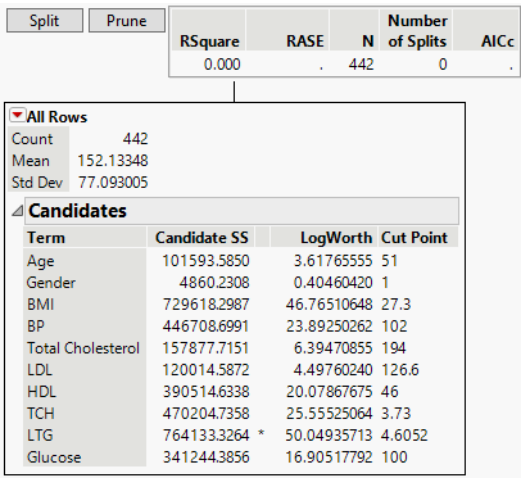
Number of Splits The current number of splits in the decision tree.

AICc The corrected Akaike's Information Criterion. See *Fitting Linear Models*.

Node Reports

Each node in the tree has a report and a red triangle menu with additional options. Terminal nodes also have a Candidates report.

Figure 4.11 Terminal Node Report for a Continuous Response



Count The number of observations (rows) in the branch.

Mean The average response for all observations in that branch.

Std Dev The standard deviation of the response for all observations in that branch.

Candidates For each column, the Candidates report provides details about the optimal split for that column. The optimal split over all columns is marked with an asterisk.

Term Shows the candidate columns.

Candidate SS Sum of squares for the best split.

LogWorth The LogWorth statistic, defined as $-\log_{10}(p\text{-value})$. The optimal split is the one that maximizes the LogWorth. See "Statistical Details for the Partition Platform".

Cut Point The value of the predictor that determines the split. For a categorical term, the levels in the left-most split are listed.

The optimum split is noted by an asterisk. However, there are cases where the Candidate SS is higher for one variable, but the Logworth is higher for a different variable. In this case > and < are used to point in the best direction for each variable. The asterisk corresponds to the condition where they agree. See "Statistical Details for the Partition Platform".

Partition Platform Options

The Partition red triangle menu options give you the ability to customize reports according to your needs. The available options are determined by the type of data that you use for your analysis.

Display Options Contains options that show or hide report elements.

Show Points Shows the points. For categorical responses, this option shows the points or colored panels.

Show Tree Shows the large tree of partitions.

Show Graph Shows the partition graph.

Show Split Bar (Available only for categorical responses.) Shows the colored bars that indicate the split proportions in each leaf.

Show Split Stats Shows the split statistics. For more information about the categorical split statistic G^2, see "Statistical Details for the Partition Platform".

Show Split Prob (Available only for categorical responses.) Shows the Rate and Prob statistics in the node reports.

JMP automatically shows the Rate and Prob statistics when you select **Show Split Count**. For more information about Rate and Prob, see "Statistical Details for the Partition Platform".

Show Split Count (Available only for categorical responses.) Shows frequency counts in the node reports. When you select this option, JMP automatically selects **Show Split Prob**. And when you deselect **Show Split Prob**, the counts do not appear.

Show Split Candidates Shows the Candidates report.

Sort Split Candidates Sorts the Candidates reports by the statistic or the log(worth), whichever is appropriate.

Split Best Splits the tree at the optimal split point. This is equivalent to clicking the **Split** button.

Prune Worst Removes the terminal split that has the least discrimination ability. This is equivalent to clicking the **Prune** button.

Minimum Size Split Define the minimum size split allowed by entering a number or a fractional portion of the total sample size. To specify a number, enter a value greater than or equal to 1. To specify a fraction of the sample size, enter a value less than 1. The default value is set to the maximum of 5, or the floor of the number of rows divided by 10,000.

Lock Columns Interactively lock columns so that they are not considered for splitting. You can turn the display off or back on without affecting the individual locks.

Plot Actual by Predicted (Available only for continuous responses.) Shows a plot of actual values by predicted values. See "Actual by Predicted Plot".

Small Tree View Shows a small version of the partition tree to the right of the partition plot.

Tree 3D Shows a 3-D plot of the tree structure. To access this option, press Shift and click the red triangle menu.

Leaf Report Shows the mean and count or rates for the bottom-level leaves of the report.

Column Contributions Shows a report indicating each input column's contribution to the fit. The report also shows how many times it defined a split and the total G^2 or Sum of Squares attributed to that column.

Split History Shows a plot of RSquare versus the number of splits. If you use excluded row validation, holdback validation, or a validation column, separate curves are drawn for training and validation RSquare values. The RSquare curve is blue for the training set and red for the validation set. If you select K Fold Crossvalidation, the RSquare curve for all of the data is blue, and the curve for the cross validation RSquare is green.

ROC Curve (Available only for categorical responses.) Receiver Operating Characteristic (ROC) curves display the efficiency of a model's fitted probabilities to sort the response levels. See "ROC Curve".

Lift Curve (Available only for categorical responses.) Lift curves display the predictive ability of a partition model. See "Lift Curve".

Show Fit Details (Appears only for categorical responses.) The Fit Details report shows several measures of fit and provides a Confusion Matrix report. See "Show Fit Details".

Save Columns Contains options for saving model and tree results, and creating SAS code.

Save Residuals Saves the residual values from the model to the data table.

Save Predicteds Saves the predicted values from the model to the data table.

Save Leaf Numbers Saves the leaf numbers of the tree to a column in the data table.

Save Leaf Labels Saves leaf labels of the tree to the data table. The labels document each branch that the row would trace along the tree. Each branch is separated by "&". An example label might be: "size(Small,Medium)&size(Small)". However, JMP does not include redundant information in the form of category labels that are repeated. A category label for a leaf might refer to an inclusive list of categories in a higher tree node. A caret ('^") appears where the tree node with redundant labels occurs. Therefore, "size(Small,Medium)&size(Small)" is presented as ^&size(Small).

Save Prediction Formula Saves prediction formulas to a column or multiple columns in the data table. The formulas consist of nested conditional clauses that describe the tree structure. If the response is continuous, one column that contains a Predicting property is added. If the response is categorical, columns that contain a Response Probability property are added for each level of the response. In addition, a Most Likely column that contains the response level with the highest probability of occurrence for each observation is added.

Save Tolerant Prediction Formula Saves a formula that predicts even when there are missing values and when Informative Missing has not been checked. The prediction formula tolerates missing values by randomly allocating response values for missing predictors to a split. If the response is continuous, the column contains a Predicting property. If the response is categorical, the column contains a Response Probability property. If you have checked Informative Missing, you can save the Tolerant Prediction Formula by holding the Shift key as you click the report's red triangle.

Save Leaf Number Formula Saves a column containing a formula in the data table that computes the leaf number.

Save Leaf Label Formula Saves a column containing a formula in the data table that computes the leaf label.

Make SAS DATA Step Creates SAS code for scoring a new data set.

JMP PRO **Publish Prediction Formula** Creates a prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot".

JMP PRO **Publish Tolerant Prediction Formula** Creates a tolerant prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot". If you have checked Informative Missing, you can use this option by holding the Shift key as you click the report's red triangle.

Specify Profit Matrix (Available only for categorical responses.) Enables you to specify profits or costs associated with correct or incorrect classification decisions. For a nominal response, you can specify the profit matrix entries using a probability threshold. See "Show Fit Details".

Profiler Shows an interactive profiler report. Changes in the factor values are reflected in the estimated classification probabilities. See *Profilers*.

Color Points (Available only for categorical responses.) Colors points based on their response level. This is equivalent to clicking the Color Points button. See "Report for Categorical Responses".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Show Fit Details

Figure 4.12 Fit Details for Categorical Response (Y Binary from Diabetes.jmp)

Fit Details

Measure	Training	Validation	Definition
Entropy RSquare	0.4276	0.1543	1-Loglike(model)/Loglike(0)
Generalized RSquare	0.5656	0.2464	(1-(L(0)/L(model))^(2/n))/(1-L(0)^(2/n))
Mean -Log p	0.3224	0.5312	∑ -Log(ρ[j])/n
RASE	0.3201	0.4080	√ ∑(y[j]-ρ[j])²/n
Mean Abs Dev	0.2083	0.2917	∑ |y[j]-ρ[j]|/n
Misclassification Rate	0.1371	0.2238	∑ (ρ[j]≠ρMax)/n
N	299	143	n

Confusion Matrix

Training

Actual	Predicted Count	
Y Binary	Low	High
Low	212	12
High	29	46

Actual	Predicted Rate	
Y Binary	Low	High
Low	0.946	0.054
High	0.387	0.613

Validation

Actual	Predicted Count	
Y Binary	Low	High
Low	84	13
High	19	27

Actual	Predicted Rate	
Y Binary	Low	High
Low	0.866	0.134
High	0.413	0.587

Entropy RSquare A measure of fit that compares the log-likelihoods from the fitted model and the constant probability model. Entropy RSquare ranges from 0 to 1, where values closer to 1 indicate a better fit. See "Entropy RSquare".

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991). Values closer to 1 indicate a better fit.

Mean -Log p The average of -log(p), where p is the fitted probability associated with the event that occurred. Smaller values indicate a better fit.

RASE The root average square error, where the differences are between the response and p (the fitted probability for the event that actually occurred). Smaller values indicate a better fit.

Mean Abs Dev The average of the absolute values of the differences between the response and p (the fitted probability for the event that actually occurred). Smaller values indicate a better fit.

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category. Smaller values indicate a better fit.

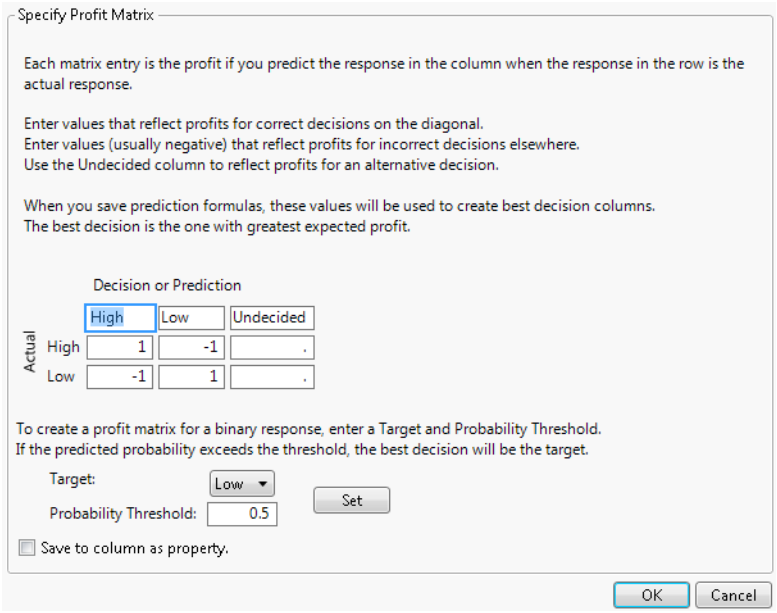
The Confusion Matrix report shows matrices for the training set and for the validation and test sets (if defined). The Confusion Matrix Report contains confusion matrices and confusion rates matrices. A confusion matrix is a two-way classification of actual and predicted responses. A confusion rates matrix is equal to the confusion matrix, with the numbers divided by the row totals.

If the response has a Profit Matrix column property, or if you specify costs using the Specify Profit Matrix option, then a Decision Matrix report appears. See "Decision Matrix Report".

Specify Profit Matrix

A profit matrix can be used with categorical responses. A profit matrix is used to assign costs to undesirable outcomes and profits to desirable outcomes.

Figure 4.13 Specify Profit Matrix Window



You can assign profit and cost values to each combination of actual and predicted response categories. To specify the costs of classifying into an alternative category, enter values in the Undecided column. To save your assignments to the response column as a property, check **Save to column as property.** Leaving this option unchecked applies the Profit Matrix only to the current Partition report.

Probability Threshold Specification for Profit Matrix

When the response is binary, instead of entering weights into the profit matrix, you can specify a probability threshold in the Profit Matrix window. For more information about how values are calculated for the profit matrix, see *Using JMP*.

Target The level whose probability is modeled.

Probability Threshold A threshold for the probability of the target level. If the probability that an observation falls into the target level exceeds the probability threshold, the observation is classified into that level.

When you define costs using the Specify Profit Matrix option and then select Show Fit Details, a Decision Matrix report appears. See "Decision Matrix Report".

When you specify a profit matrix and save the model prediction formula, the formula columns saved to the data table include the following:

– Profit for <level>: For each level of the response, a column gives the expected profit for classifying each observation into that level.

- Most Profitable Prediction for <column name>: For each observation, gives the level of the response with the highest expected profit.
- Expected Profit for <column name>: For each observation, gives the expected profit for the classification defined by the Most Profitable Prediction column.
- Actual Profit for <column name>: For each observation, gives the actual profit for classifying that observation into the level specified by the Most Profitable Prediction column.

Decision Matrix Report

Figure 4.14 Fit Details Report with Decision Matrix Report

Fit Details

Measure	Training	Validation	Definition
Entropy RSquare	0.3788	0.0367	1-Loglike(model)/Loglike(0)
Generalized RSquare	0.5382	0.0647	(1-(L(0)/L(model))^(2/n))/(1-L(0)^(2/n))
Mean -Log p	0.4134	0.6402	∑ -Log(ρ[j])/n
RASE	0.3649	0.4694	√ ∑(y[j]-ρ[j])²/n
Mean Abs Dev	0.2854	0.3861	∑ |y[j]-ρ[j]|/n
Misclassification Rate	0.1915	0.3810	∑ (ρ[j]≠pMax)/n
N	94	42	n

Confusion Matrix

Training

Actual Severity	Predicted Count High	Predicted Count Low
High	46	12
Low	6	30

Validation

Actual Severity	Predicted Count High	Predicted Count Low
High	15	11
Low	5	11

Decision Matrix

Training

Actual Severity	Decision Count High	Decision Count Low
High	51	7
Low	12	24

Actual Severity	Decision Rate High	Decision Rate Low
High	0.879	0.121
Low	0.333	0.667

Misclassification Rate
0.2021

Validation

Actual Severity	Decision Count High	Decision Count Low
High	20	6
Low	6	10

Actual Severity	Decision Rate High	Decision Rate Low
High	0.769	0.231
Low	0.375	0.625

Misclassification Rate
0.2857

Specified Profit Matrix

Actual	Decision High	Decision Low
High	1	-5
Low	-3	1

Note: This report is available only if the response has a Profit Matrix column property or if you specify costs using the Specify Profit Matrix option. The report is part of the Fit Details report.

When a profit matrix is defined, the partition algorithm uses the values in the matrix to calculate the profit for each decision. When you select Show Fit Details, a Decision Matrix report appears.

In the Decision Matrix report, the decision counts reflect the most profitable prediction decisions based on the weighting in the profit matrix. The report gives Decision Count and Decision Rate matrices for the training set and for validation and test sets (if defined). For reference, the profit matrix is also shown.

Note: If you change the weights in your Profit Matrix using the Specify Profit Matrix option, the Decision Matrix report automatically updates to reflect your changes.

Decision Count Matrix Shows a two-way classification with actual responses in rows and classification counts in columns.

Specified Profit Matrix Gives the weights that define the Profit Matrix.

Decision Rate Matrix Shows rate values corresponding to the proportion of a given row's observations that are classified into each category. If all observations are correctly classified, the rates on the diagonal are all equal to one.

Tip: You can obtain a decision rate matrices for a response using the default profit matrix with costs of 1 and -1. Select **Specify Profit Matrix** from the red triangle menu, make no changes to the default values, and click **OK**.

The matrices are arranged in two rows:

- The Decision Count matrices are in the first row.
- The Specified Profit Matrix is to the right in the first row.
- The Decision Rate matrices are in the second row.

Informative Missing

The Informative Missing option enables informative treatment of missing values on the predictors. The model that is fit is deterministic. The Informative Missing option is found on the launch window and is selected by default. When the Informative Missing option is selected, categorical and continuous predictors are handled differently:

- Rows containing missing values for a categorical predictor are entered into the analysis as a separate level of the variable.
- Rows containing missing values for a continuous predictor are assigned to a split as follows: The values of the continuous predictor are sorted. Missing rows are first considered to be on the low end of the sorted values. All splits are constructed. The missing rows are then considered to be on the high end of the sorted values. Again, all splits are constructed. The optimal split is determined using the LogWorth criterion. For further splits on the given predictor, the algorithm commits the missing rows to high or low values, as determined by the first split induced by that predictor.

If the Informative Missing option is not selected, the missing values are handled as follows:

- When a predictor with missing values is used as a splitting variable, each row with a missing value on that predictor is randomly assigned to one of the two sides of the split.
- The first time a predictor with missing values is used as a splitting variable an *Imputes* column is added to the Summary Report showing the number of imputations. As additional imputations are made, the Imputes column updates (Figure 4.15), where five imputations were performed.

Note: The number of Imputes can be greater than the number of rows that contain missing values. The imputation occurs at each split. A row with missing values can be randomly assigned multiple times. Each time a row is randomly assigned it increments the imputation count.

Figure 4.15 Impute Message in Summary Report

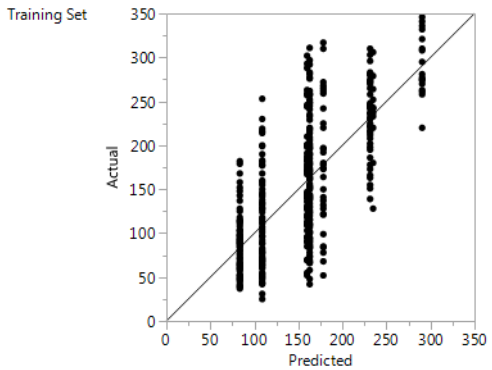
RSquare	RASE	N	Number of Splits	AICc
0.448	6.8246278	501	1	3352.2

Actual by Predicted Plot

For continuous responses, the Actual by Predicted plot is the typical plot of the actual response versus the predicted response. When you fit a Decision Tree, all observations in a leaf have the same predicted value. If there are *n* leaves, then the Actual by Predicted plot shows at most *n* distinct predicted values. The actual values form a scatter of points around each leaf mean on *n* vertical lines.

The diagonal line is the Y = X line. For a perfect fit, all the points would be on this diagonal. When validation is used, plots are shown for both the training and the validation sets (Figure 4.16).

Figure 4.16 Actual by Predicted Plots for a Continuous Response



ROC Curve

The ROC Curve option is available only for categorical responses. Receiver Operating Characteristic (ROC) curves display the efficiency of a model's fitted probabilities in sorting the response levels. An introduction to ROC curves is found in *Basic Analysis*.

The predicted response for each observation in a partition model is a value between 0 and 1. To use the predicted response to classify observations as positive or negative, a *cut point* is used. For example, if the cut point is 0.5, an observation with a predicted response at or above 0.5 would be classified as positive, and an observation below 0.5 as negative. There are trade offs in classification as the cut point is varied.

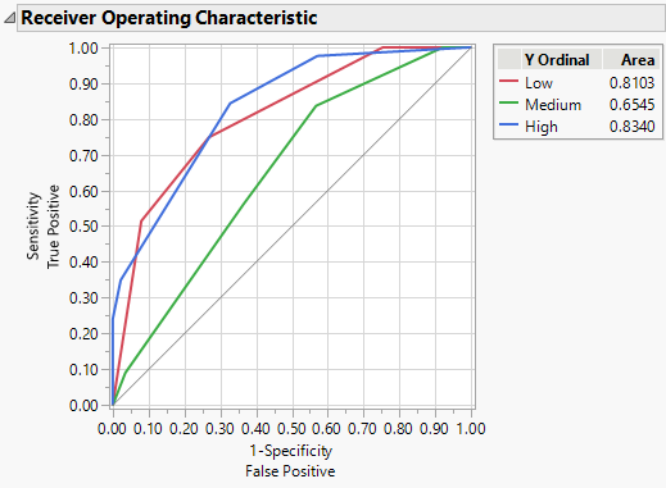
To generate a ROC curve, each predicted response level is considered as a possible cut point and the following values are computed for each possible cut point:

- The *sensitivity* is the proportion of true positives or the percent of positive observations with a predicted response greater than the cut point.
- The *specificity* is the proportion of true negatives or the proportion of negative observations with a predicted response less than the cut point.

The ROC curve plots sensitivity against 1 - specificity. A partition model with n splits has n+1 predicted values. The ROC curve for the partition model has n+1 line segments.

If your response has more than two levels, the Partition report contains a separate ROC curve for each response level versus the other levels. Each curve is the representation of a level as the positive response level. If there are only two levels, one curve is the reflection of the other.

Figure 4.17 ROC Curves for a Three Level Response



If the model perfectly rank-orders the response values, then the sorted data contains all of the positive values first, followed by all of the other values. In this situation, the curve moves all the way to the top before it moves at all to the right. If the model does not predict well, the curve follows the diagonal line from the bottom left to top right of the plot.

In practice, the ROC curve lies above the diagonal. The area under the curve is the indicator of the goodness of fit for the model. A value of 1 indicates a perfect fit and a value near 0.5 indicates that the model cannot discriminate among groups.

When your response has more than two levels, the ROC curve plot enables you to see which response categories have the largest area under the curve.

Lift Curve

The Lift Curve option provides another plot to display the predictive ability of a partition model. The lift curve plots the lift versus the portion of the observations. There is a point for each unique predicted probability value. Each predicted probability of a response level defines a portion of the observations that have a predicted probability greater than or equal to the unique predicted probability value. For a particular level of the response, the *lift* value is the ratio of the proportion of observed responses in that portion to the overall proportion of observed responses.

Note: For smaller models, it is possible that a large portion of the points have the same predicted probability value. If this probability is the highest predicted probability for the response level, the lift curve does not start at Portion = 0. For example, this is shown in the Low lift curve in Figure 4.18.

Figure 4.18 Lift Curve

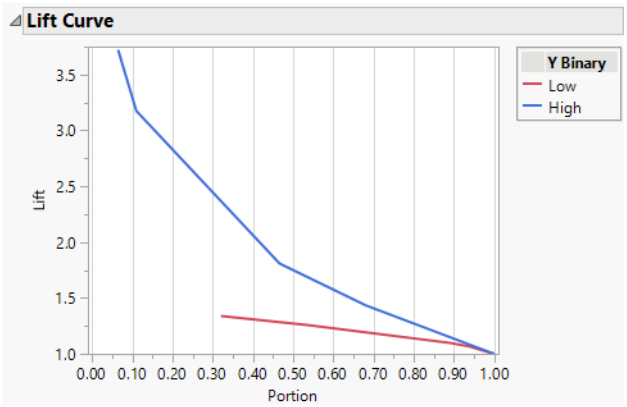


Figure 4.19 Lift Table for Lift Curve

Prob High	N > Prob High	Portion	N High in Portion	Portion High	Lift = portion high/ overall high of .27
0.97	20	0.06	20	1.00	3.72
0.77	44	0.14	39	0.89	3.30
0.33	68	0.22	47	0.69	2.57
0.31	168	0.54	78	0.46	1.73
0.04	309	1.00	83	0.27	1.00

Figure 4.19 provides a table of values to demonstrate the calculation of Lift and Portion used for the High lift curve shown in Figure 4.18. A partition model with five splits was built to predict the response, Y Binary. Y Binary has two levels: Low and High. The lift curve is based on 309 observations. There are 83 observed High responses for an overall rate of 0.27.

- Prob High: The five unique predicted probability values from the partition model for the High response level.
- N > Prob High: The number of observations that have a predicted probability value equal to or greater than the value in Prob High.
- Portion: N > Prob High divided by 309, the total number of observations.
- N High in Portion: The number of observations in each portion that have an observed High response.
- Portion High: N High in Portion divided by N > Prob High.
- Lift: Portion High divided by 0.27, the overall rate of the observed High response.

Lift measures how many High responses fall in each portion as compared to the expected number of High responses for that portion. For the first 6% of the data set the lift is 3.72. Using the model to select the 6% of the observations with the highest predicted values results in 3.72 more High responses than if that 6% were selected at random.

Node Options

This section describes the options on the red triangle menu for each node.

Split Best Finds and executes the best split at or below this node.

Split Here Splits at the selected node on the best column to split by.

Split Specific Lets you specify where a split takes place. This is useful in showing what the criterion is as a function of the cut point, as well as in determining custom cut points. When specifying a splitting column, you can choose the following options for how the split is performed:

Optimal Value Splits at the optimal value of the selected variable.

Specified Value Enables you to specify the level where the split takes place.

Output Split Table Produces a data table showing all possible splits and their associated split value.

Prune Below Eliminates the splits below the selected node.

Prune Worst Finds and removes the worst split below the selected node.

Select Rows Selects the data table rows corresponding to this leaf. You can extend the selection by holding down the Shift key and choosing this command from another node.

Show Details Produces a data table that shows the split criterion for a selected variable. The data table, composed of split intervals and their associated criterion values, has an attached script that produces a graph for the criterion.

Lock Prevents a node or its subnodes from being chosen for a split. When checked, a lock icon appears in the node title.

Validation in Partition

The use of validation with partition models is important given that partition models are easily overfit. When this happens, the model predicts the data used to build the model very well, but predicts future observations poorly. Validation is the process of using part of a data set to estimate model parameters, and using the other part to assess the predictive ability of the model. For more information about validation, see "Validation in JMP Modeling".

In Partition, when a validation method is used, the **Go** button appears. The **Go** button provides for repeated splitting without having to repeatedly click the **Split** button. When you click the **Go** button, splitting occurs until the validation RSquare is better than what the next 10 splits would obtain. This rule can result in complex trees that are not very interpretable, but have good predictive power.

Using the **Go** button turns on the **Split History** command. If using the **Go** button results in a tree with more than 40 nodes, the **Show Tree** command is turned off.

Select one of the following validation methods:

Excluded Rows Uses row states to subset the data. Rows that are unexcluded are used as the training set, and excluded rows are used as the validation set.

For more information about using row states and how to exclude rows, see *Using JMP*.

Holdback Randomly divides the original data into the training and validation data sets. The Validation Portion on the platform launch window is used to specify the proportion of the original data to use as the validation data set (holdback). See "Launch the Partition Platform" for more information about the Validation Portion.

JMP PRO **Validation Column** Uses a numeric column that defines the validation sets. This column should contain at most three distinct values:

- If the validation column has two levels, the smaller value defines the training set and the larger value defines the validation set.
- If the validation column has three levels, the values, in order of increasing size, define the training, validation, and test sets.
- If the validation column has more than three levels, the rows that contain the smallest three values define the validation sets. All other rows are excluded from the analysis.

If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

Tip: To use K Fold or Nested K Fold crossvalidation, fit a partition model through the Model Screening platform. See "Model Screening".

Additional Examples of Partitioning

- "Example of a Continuous Response"
- "Example of Informative Missing"
- "Example of Profit Matrix and Decision Matrix Report"

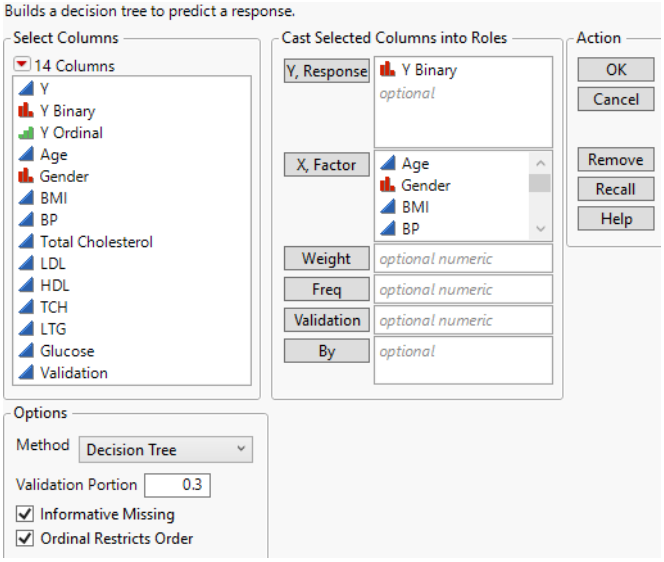
Example of a Continuous Response

In this example, you use the Partition platform to construct a decision tree that predicts the one-year disease progression measured on a quantitative scale for patients with diabetes.

1. Select **Help > Sample Data Library** and Diabetes.jmp.
2. Select **Analyze** > **Predictive Modeling** > **Partition**.
3. Select Y and click **Y, Response**.
4. Select Age through Glucose and click **X, Factor**.
5. Select a validation procedure based on your JMP installation:
   - For JMP Pro, select Validation and click **Validation**.
   - For JMP, enter 0.3 as the **Validation Portion**.

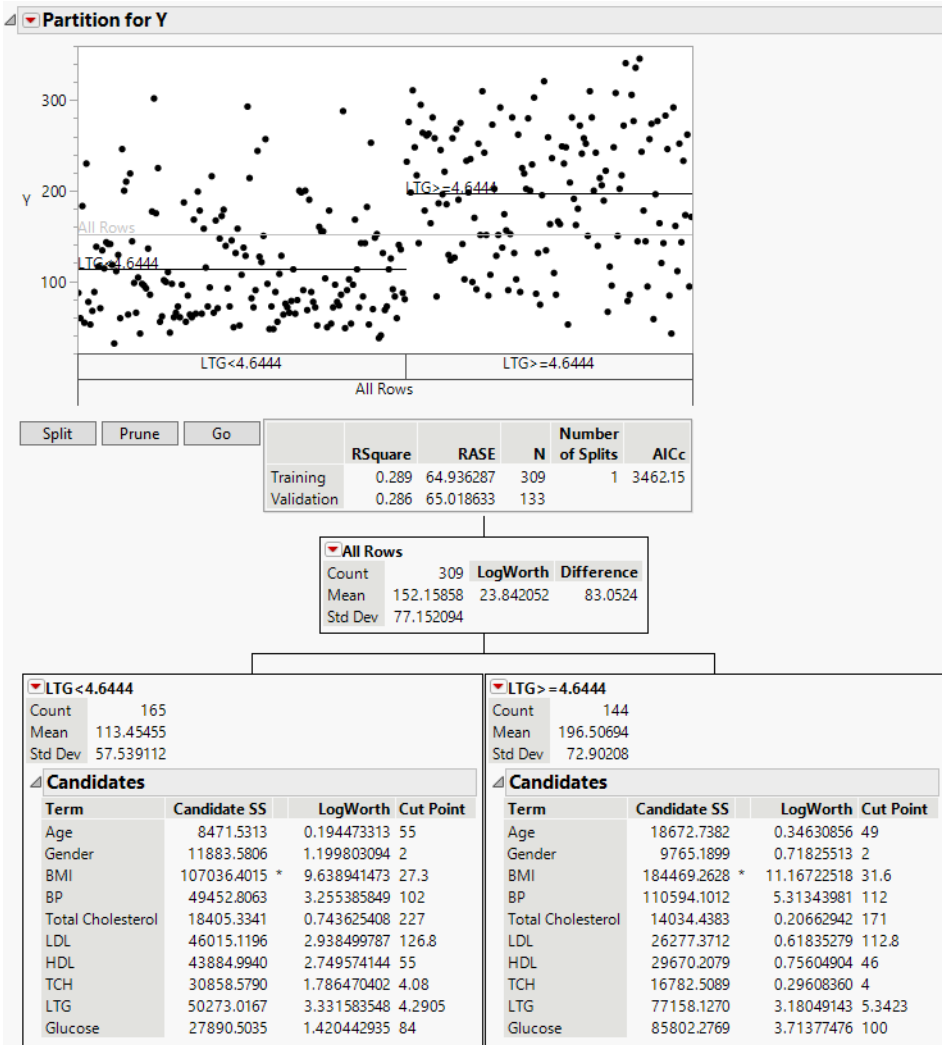
Note: Results using the validation proportion can differ from those shown here due to the random selection of validation rows.

Figure 4.20 Completed Launch Window with Validation Portion = 0.3



6. Click **OK**.
7. On the platform report window, click **Split** once to perform a split.

Figure 4.21 Report after First Split with Decision Tree Hidden



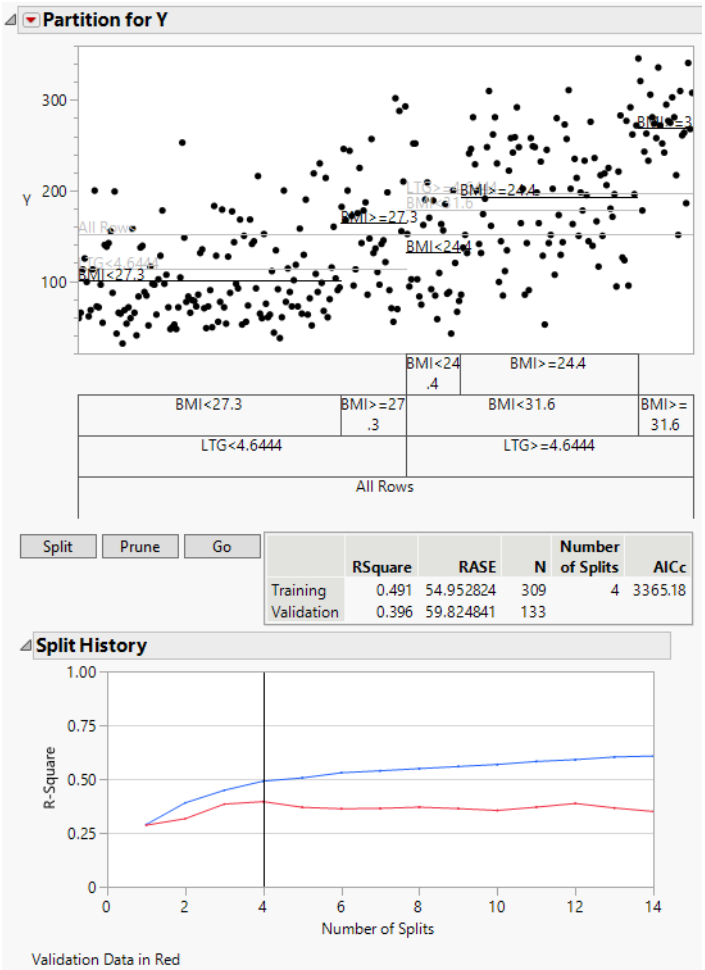
The original 309 values in the training data set are now split into two parts:

- The left leaf, corresponding to LTG < 4.6444, has 165 observations.
- The right leaf, corresponding to LTG >= 4.6444 has 144 observations.

For both the right and left leaf the next split would be on BMI. The Candidate SS for BMI on the right leaf is higher than the Candidate SS for BMI on the left leaf. Thus, the next split is on the right leaf.

8. Click **Go** to use automatic splitting.

Figure 4.22 Report after Automatic Splitting with Validation



The solution found has four splits. The Split History plot shows that there is no further improvement in the validation data set after four splits. The RSquare value of 0.39 on the validation data does not support this model as a strong predictor of disease progression. The scatter across partitions in the partition plot further indicate that this model does not separate the Y levels well.

Example of Informative Missing

In this example, you construct a decision tree model to predict if a customer is a credit risk. Since your data set contains missing values, you also explore the effectiveness of the Informative Missing option.

Launch the Partition Platform

1. Select **Help > Sample Data Library** and open Equity.jmp.
2. Select **Analyze** > **Predictive Modeling** > **Partition**.
3. Select BAD and click **Y, Response**.
4. Select LOAN through DEBTINC and click **X, Factor**.
5. Click **OK**.

Create the Decision Tree and ROC Curve with Informative Missing

1. Press Shift and click **Split**.
2. Enter 5 for the number of splits and click **OK.**
3. Click the red triangle next to Partition for BAD and select **ROC Curve**.
4. Click the red triangle next to Partition for BAD and select **Save Columns > Save Prediction Formula**.

   The columns Prob(BAD==Good Risk) and Prob(BAD==Bad Risk) contain the formulas that Informative Missing utility uses to classify the credit risk of future loan applicants. You are interested in how this model performs in comparison to a model that does not use informative missing.

Create the Decision Tree and ROC Curve without Informative Missing

1. Click the red triangle next to Partition for BAD and select **Redo > Relaunch Analysis**
2. De-select **Informative Missing**.
3. Click OK and repeat the steps in "Create the Decision Tree and ROC Curve with Informative Missing".

   The columns Prob(BAD==Good Risk) 2 and Prob(BAD==Bad Risk) 2 contain the formulas that do not use the informative missing utility.

Compare the ROC Curves

Visually compare the ROC curves from the two models. The model at left is with Informative Missing, and the model at right is without Informative Missing.

Figure 4.23 ROC Curves for Models with (Left) and without (Right) Informative Missing

The area under the curve (AUC) for the model with informative missing (0.8695) is higher than the AUC for the model without informative missing (0.7283). Because there are only two levels for the response, the ROC curves for each model are reflections of one another and the AUCs are equal.

Note: Your AUC can differ from that shown for the model without informative missing. When informative missing is not used, the assignment of missing rows to sides of a split is random. Rerunning the analysis can result in slight differences in results.

Use the Model Comparison Platform

Next, compare the models using the Model Comparison platform to compare the two sets of formulas that you created in step 4 and step 3.

1. Select **Analyze > Predictive Modeling > Model Comparison**.
2. Select Prob(BAD==Good Risk), Prob(BAD==Bad Risk), Prob(BAD==Good Risk) 2, and Prob(BAD==Bad Risk) 2 and click **Y, Predictors**.

   The first pair of formula columns contains the formulas from the model with informative missing. The second pair of formula columns contains the formulas from the model without informative missing.
3. Click **OK**.

Figure 4.24 Measures of Fit from Model Comparison

Measures of Fit for BAD

Creator	.2 .4 .6 .8	Entropy RSquare	Generalized RSquare	Mean -Log p	RMSE	Mean Abs Dev	Misclassification Rate	N
Partition		0.3813	0.5015	0.3092	0.3013	0.1817	0.1158	5960
Partition		0.1232	0.1834	0.4381	0.3687	0.2737	0.1768	5960

The Measures of Fit report shows that the first model, which was fit with informative missing, performs better than the second model, which was not fit with informative missing. The first model has higher RSquare values as well as a lower RMSE value and a lower Misclassification Rate. These findings align the ROC curves comparison.

Note: Again, your results can differ due to the random differences when Informative Missing is not used.

Example of Profit Matrix and Decision Matrix Report

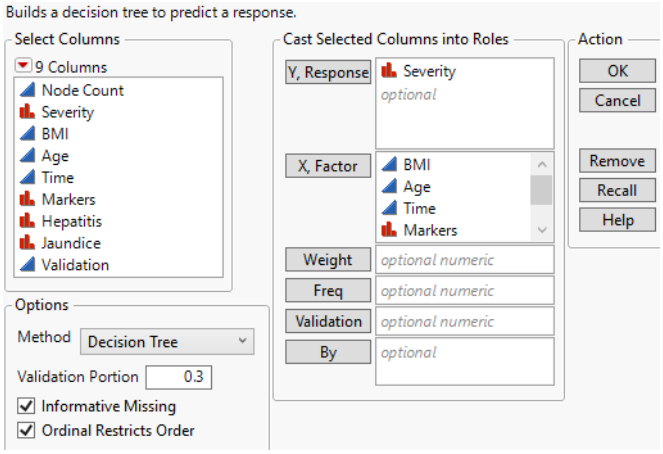
For this example, consider a study of patients who have liver cancer. Based on various measurements and markers, you want to classify patients according to their disease severity (high or low). There are two errors that one can make in classification of patients: classifying a subject who has high severity into the low group, or classifying a patient with low severity into the high group. Clinically, the misclassification of a high patient as low is a costly error, as that patient might not receive the aggressive treatment needed. Classifying a patient with low severity into the high severity group is a less costly error. That patient might receive the more aggressive treatment than needed, but this is not a major concern.

In this example, you define a profit matrix in the context of a liver cancer study and obtain a Decision Matrix report. The Decision Matrix report helps you assess your classification rates relative to the costs in your profit matrix.

1. Select **Help > Sample Data Library** and open Liver Cancer.jmp.
2. Select **Analyze > Predictive Modeling > Partition**.
3. Select Severity and click **Y, Response**.
4. Select BMI through Jaundice and click **X, Factor**.
5. Select a validation procedure based on your JMP installation:
   - For JMP Pro, select Validation and click **Validation**.
   - For JMP, enter 0.3 as the **Validation Proportion**.

Note: Results using the validation proportion can differ from those shown here, due to the random selection of validation rows.

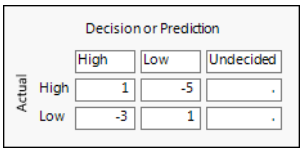
Figure 4.25 Completed Launch Window with Validation Portion = 0.3



6. Click **OK**.
7. Press Shift and click **Split.**
8. Enter 10 for the number of splits and click **OK**.

   Check that the Number of Splits is 10 in the panel beneath the plot.
9. Click the red triangle next to Partition for Severity and select **Specify Profit Matrix**.
10. Change the entries to the following values:
    - Enter 1 in the High, High box.
    - Enter -5 in the High, Low box.
    - Enter -3 in the Low, High box.
    - Enter 1 in the Low, Low box.

Figure 4.26 Completed Profit Matrix



Tip: You can save this profit matrix as a column property for use in later analyses. Select the check box "Save to column as property" at the bottom of the profit matrix window.

Note the following:

- Each value of 1 reflects your profit when you make a correct decision.
- The -3 value indicates that if you classify a Low severity patient as High severity, your loss is 3 times as much as the profit of a correct decision.

– The -5 value indicates that if you classify a High severity patient as Low severity, your loss is 5 times as much as the profit of a correct decision.

11. Click **OK**.
12. Click the red triangle next to Partition for Severity and select **Show Fit Details**.

Figure 4.27 Confusion Matrix and Decision Matrix Reports

Fit Details

Measure	Training	Validation	Definition
Entropy RSquare	0.3788	0.0367	1-Loglike(model)/Loglike(0)
Generalized RSquare	0.5382	0.0647	(1-(L(0)/L(model))^(2/n))/(1-L(0)^(2/n))
Mean -Log p	0.4134	0.6402	∑ -Log(ρ[j])/n
RASE	0.3649	0.4694	√ ∑(y[j]-ρ[j])²/n
Mean Abs Dev	0.2854	0.3861	∑ |y[j]-ρ[j]|/n
Misclassification Rate	0.1915	0.3810	∑ (ρ[j]≠ρMax)/n
N	94	42	n

Confusion Matrix

Training

Actual Severity	Predicted Count High	Predicted Count Low
High	46	12
Low	6	30

Validation

Actual Severity	Predicted Count High	Predicted Count Low
High	15	11
Low	5	11

Decision Matrix

Training

Actual Severity	Decision Count High	Decision Count Low
High	51	7
Low	12	24

Actual Severity	Decision Rate High	Decision Rate Low
High	0.879	0.121
Low	0.333	0.667

Misclassification Rate
0.2021

Validation

Actual Severity	Decision Count High	Decision Count Low
High	20	6
Low	6	10

Actual Severity	Decision Rate High	Decision Rate Low
High	0.769	0.231
Low	0.375	0.625

Misclassification Rate
0.2857

Specified Profit Matrix

Actual	Decision High	Decision Low
High	1	-5
Low	-3	1

The Confusion Matrix and Decision Matrix reports follow the list of Measures in the Fit Details report. Notice that the Confusion Matrix report and the confusion matrices in the Decision Matrix report show different counts. This is because the weighting in the profit matrix results in different decisions than do the predicted probabilities without weighting.

The Confusion Matrix for the validation set shows classifications based on predicted probabilities alone. Based on these, 11 High severity patients would be classified as Low severity and 5 Low severity patients would be classified as High severity.

The Decision Matrix report incorporates the profit matrix weights. Using those weights, only 6 High severity patients are classified as Low severity. However, this comes at the expense of misclassifying 6 Low severity patients into the High severity group (1 additional patient).

13. Click the red triangle next to Partition for Severity and select **Save Columns > Save Prediction Formula**.

    Eight columns are added to the data table.

    Tip: To quickly return to the data table, click the View Associated Data icon in the bottom right corner of the report window (Windows) or the Show Data Table icon on the tool bar menu (macOS).

    – The first three columns involve only the predicted probabilities. The confusion matrix counts are based on the Most Likely Severity column, which classifies a patient into the level with the highest predicted probability. These probabilities are given in the Prob(Severity == High) and Prob(Severity == Low) columns.

    – The last five columns involve the profit matrix weighting. The column called Most Profitable Prediction for Severity contains the decision based on the profit matrix. The decision for a patient is the level that results in the largest profit. The profits are given in the Profit for High and Profit for Low columns.

Statistical Details for the Partition Platform

- "Responses and Factors"
- "Splitting Criterion"
- "Predicted Probabilities in Decision Tree and Bootstrap Forest"

Responses and Factors

The response can be either continuous or categorical (nominal or ordinal):

- If the response is categorical, then it is fitting the probabilities estimated for the response levels, minimizing the residual log-likelihood chi-square [2*entropy].
- If the response is continuous, then the platform fits means, minimizing the sum of squared errors.

The factors can be either continuous or categorical (nominal or ordinal):

- If the factor is continuous, then the partition is done according to a splitting "cut" value for the factor.
- If the factor is categorical, then it divides the X categories into two groups of levels and considers all possible groupings into two levels.

Splitting Criterion

Node splitting is based on the LogWorth statistic, which is reported in Candidate reports for nodes. LogWorth is calculated as follows:

$-\log_{10}(p\text{-value})$

where the adjusted p-value is calculated in a complex manner that takes into account the number of different ways splits can occur. This calculation is very fair compared to the unadjusted p-value, which favors Xs with many levels, and the Bonferroni p-value, which favors Xs with small numbers of levels. Details about the method are discussed in Sall (2002).

For continuous responses, the Sum of Squares (SS) is reported in node reports. This is the change in the error sum-of-squares due to the split.

A candidate SS that has been chosen is:

$SS_{test} = SS_{parent} - (SS_{right} + SS_{left})$ where SS in a node is just $s^2(n - 1)$.

Also reported for continuous responses is the Difference statistic. This is the difference between the predicted values for the two child nodes of a parent node.

For categorical responses, the $\mathbf{G}^2$ (likelihood ratio chi-square) appears in the report. This is actually twice the [natural log] entropy or twice the change in the entropy. Entropy is Σ $-\log(p)$ for each observation, where p is the probability attributed to the response that occurred.

A candidate G^2 that has been chosen is:

$G^2_{\text{test}} = G^2_{\text{parent}} - (G^2_{\text{left}} + G^2_{\text{right}})$.

Partition actually has two rates; one used for training that is the usual ratio of count to total, and another that is slightly biased away from zero. By never having attributed probabilities of zero, this allows logs of probabilities to be calculated on validation or excluded sets of data, used in Entropy R-Square.

Predicted Probabilities in Decision Tree and Bootstrap Forest

The predicted probabilities for the Decision Tree and Bootstrap Forest methods are calculated as described below by the Prob statistic.

For categorical responses in Decision Tree, the Show Split Prob command shows the following statistics:

Rate The proportion of observations at the node for each response level.

Prob The predicted probability for that node of the tree. The method for calculating Prob for the i^{th} response level at a given node is defined as follows:

$$Prob_i = \frac{n_i + prior_i}{\sum(n_i + prior_i)}$$

where the summation is across all response levels; n_i is the number of observations at the node for the i^{th} response level; and $prior_i$ is the prior probability for the i^{th} response level, calculated as follows:

$$prior_i = \lambda p_i + (1-\lambda)P_i$$

where p_i is the $prior_i$ from the parent node, P_i is the $Prob_i$ from the parent node, and λ is a weighting factor currently set at 0.9.

The method for calculating Prob assures that the predicted probabilities are always nonzero.

Chapter 5

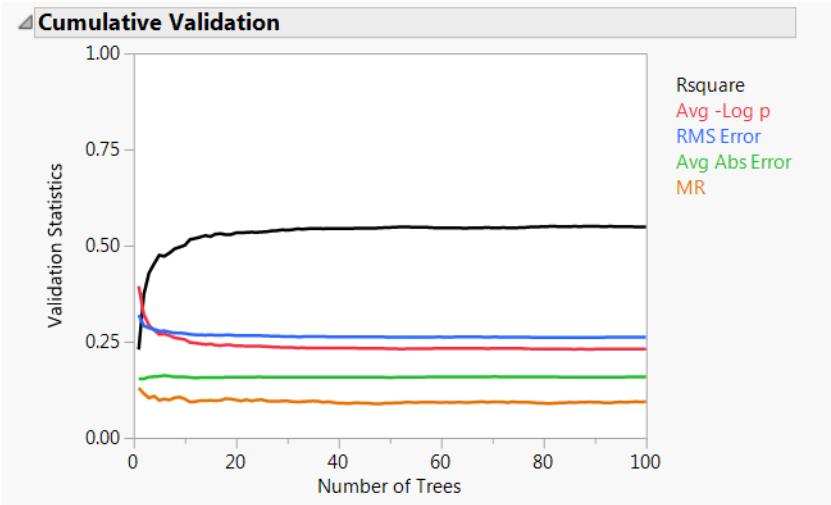
JMP PRO Bootstrap Forest

Fit a Model By Averaging Many Trees

The Bootstrap Forest platform is available only in JMP Pro.

The Bootstrap Forest platform fits an ensemble model by averaging many decision trees each of which is fit to a bootstrap sample of the training data. Each split in each tree considers a random subset of the predictors. In this way, many weak models are combined to produce a more powerful model. The final prediction for an observation is the average of the predicted values for that observation over all the decision trees.

Figure 5.1 Example of a Cumulative Validation Report



Contents

Overview of the Bootstrap Forest Platform 89
Example of Bootstrap Forest with a Categorical Response 90
Bootstrap Forest Model 90
Missing Values 92
Example of Bootstrap Forest with a Continuous Response 93
Launch the Bootstrap Forest Platform 95
Launch Window 95
Specification Window 97
The Bootstrap Forest Report 99
Model Validation-Set Summaries 100
Specifications 100
Overall Statistics 100
Cumulative Validation 102
Per-Tree Summaries 103
Bootstrap Forest Platform Options 104

JMP PRO Overview of the Bootstrap Forest Platform

The Bootstrap Forest platform predicts a response value by averaging the predicted response values across many decision trees. Each tree is grown on a *bootstrap sample* of the training data. A bootstrap sample is a random sample of observations, drawn with replacement. In addition, the predictors are sampled at each split in the decision tree. The decision tree is fit using the recursive partitioning methodology described in "Partition Models".

This is the fitting process for the training set:

1. For each tree, select a bootstrap sample of observations.
2. Fit the individual decision tree, using recursive partitioning.
   - Select a random set of predictors for each split.
   - Continue splitting until a stopping rule that is specified in the Bootstrap Forest Specification window is met.
3. Repeat step 1 and step 2 until the number of trees specified in the Bootstrap Forest Specification window is reached or until Early Stopping occurs.

For an individual tree, the bootstrap sample of observations that is used to fit the tree is drawn with replacement. You can specify the proportion of observations to be sampled. If you specify that 100% of the observations are to be sampled, because they are drawn with replacement, the expected proportion of unused observations is 1/e, or approximately 36.8%. For each individual tree, these unused observations are called the *out-of-bag* observations. The observations used in fitting the tree are called *in-bag* observations. For a continuous response, the Bootstrap Forest platform provides measures for the error rate for out-of-bag observations, called *out-of-bag error*.

For a continuous response, the predicted value for an observation is the average of its predicted values over the collection of individual trees. For a categorical response, the predicted probability for an observation is the average of its predicted probabilities over the collection of individual trees. The observation is classified into the level for which its predicted probability is the highest.

For more information about bootstrap forests, see Hastie et al. (2009).

JMP PRO Example of Bootstrap Forest with a Categorical Response

In this example, you construct a bootstrap forest model to predict whether a customer is a bad credit risk. But you are aware that your data set contains missing values, so you also explore the degree to which values are missing.

JMP PRO Bootstrap Forest Model

1. Select **Help > Sample Data Library** and open Equity.jmp.
2. Select **Analyze** > **Predictive Modeling** > **Bootstrap Forest**.
3. Select BAD and click **Y, Response**.
4. Select LOAN through DEBTINC and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.
7. Enter 4 next to Number of Terms per Split.
8. Enter 30 next to Maximum Splits per Tree.
9. Select **Multiple Fits over Number of Terms** and enter 10 next to Max Number of Terms.
10. (Optional) Select **Suppress Multithreading** and enter 123 next to Random Seed.

    Because the bootstrap forest method involves random sampling, these actions ensure that your results are exactly the same as the results shown below.
11. Click **OK**.

Figure 5.2 Overall Statistics Report

Bootstrap Forest for BAD

Specifications

Overall Statistics

Measure	Training	Validation	Test	Definition
Entropy RSquare	0.5124	0.4828	0.4824	1-Loglike(model)/Loglike(0)
Generalized RSquare	0.6302	0.6153	0.6068	(1-(L(0)/L(model))^(2/n))/(1-L(0)^(2/n))
Mean -Log p	0.2354	0.2793	0.2618	∑ -Log(ρ[j])/n
RASE	0.2596	0.2886	0.2771	√ ∑(y[j]-ρ[j])²/n
Mean Abs Dev	0.1580	0.1779	0.1688	∑ |y[j]-ρ[j]|/n
Misclassification Rate	0.0898	0.1133	0.0990	∑ (ρ[j]≠ρMax)/n
N	3576	1192	1192	n

Confusion Matrix

Training

Actual BAD	Predicted Count Good Risk	Bad Risk
Good Risk	2819	86
Bad Risk	235	436

Actual BAD	Predicted Rate Good Risk	Bad Risk
Good Risk	0.970	0.030
Bad Risk	0.350	0.650

Validation

Actual BAD	Predicted Count Good Risk	Bad Risk
Good Risk	884	33
Bad Risk	102	173

Actual BAD	Predicted Rate Good Risk	Bad Risk
Good Risk	0.964	0.036
Bad Risk	0.371	0.629

Test

Actual BAD	Predicted Count Good Risk	Bad Risk
Good Risk	921	28
Bad Risk	90	153

Actual BAD	Predicted Rate Good Risk	Bad Risk
Good Risk	0.970	0.030
Bad Risk	0.370	0.630

Because the Multiple Fits over Number of Terms option was specified, models were created using 4, 5, 6, 8, and 10 as the number of predictors in each split. The Model Validation-Set Summaries report shows that the model whose Validation set has the highest Entropy RSquare is the five-term model. This is also the model with the smallest misclassification rate. This model is determined to be the best model, and the results in the Overall report are for this model.

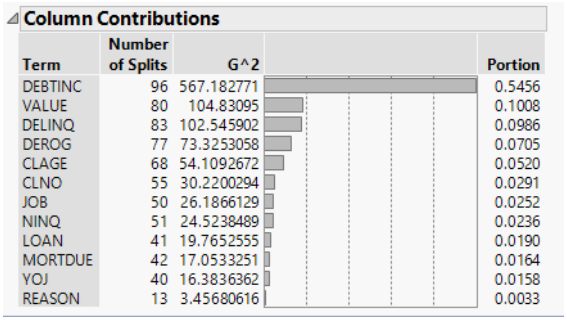
The Overall report shows that the misclassification rates for the Validation and Test sets are about 11.3% and 9.9%, respectively. The confusion matrices suggest that the largest source of misclassification is the classification of bad risk customers as good risks.

The results for the Test set give you an indication of how well your model extends to independent observations. The Validation set was used in selecting the Bootstrap Forest model. For this reason, the results for the Validation set give a biased indication of how the model generalizes to independent data.

You are interested in determining which predictors contributed the most to your model.

12. Click the red triangle next to Bootstrap Forest for BAD and select **Column Contributions**.

Figure 5.3 Column Contributions Report



Column Contributions

Term	Number of Splits	G^2	Portion
DEBTINC	96	567.182771	0.5456
VALUE	80	104.83095	0.1008
DELINQ	83	102.545902	0.0986
DEROG	77	73.3253058	0.0705
CLAGE	68	54.1092672	0.0520
CLNO	55	30.2200294	0.0291
JOB	50	26.1866129	0.0252
NINQ	51	24.5238489	0.0236
LOAN	41	19.7652555	0.0190
MORTDUE	42	17.0533251	0.0164
YOJ	40	16.3836362	0.0158
REASON	13	3.45680616	0.0033

The Column Contributions report suggests that the strongest predictor of a customer's credit risk is DEBTINC, which is the debt to income ratio. The next highest contributors to the model are DELINQ, the number of delinquent credit lines, and VALUE, the assessed value of the customer.

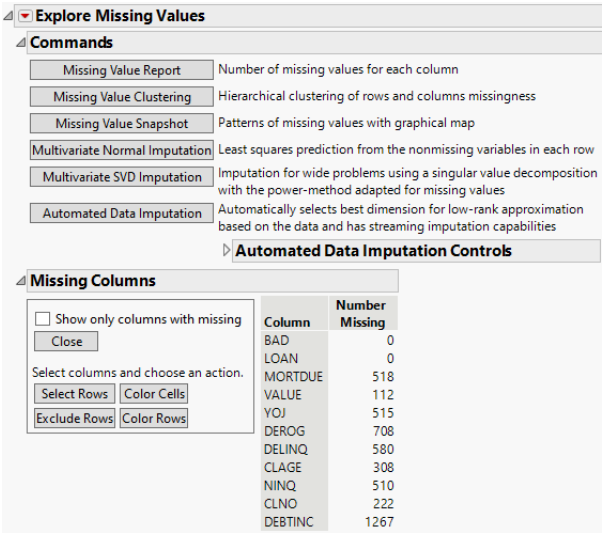
JMP PRO Missing Values

Next, you explore the extent to which predictor values are missing.

1. Select **Analyze > Screening > Explore Missing Values**.
2. Select Bad through DEBTINC and click **Y, Columns**.
3. Click **OK** in the Alert that appears.

   The columns REASON and JOB are not added to the Y, Columns list because they have a Character data type. You can see how many values are missing for these two columns using Distribution (not illustrated in this example).
4. Click **OK**.

Figure 5.4 Missing Values Report



The DEBTINC column contains 1267 missing values, which amounts to about 21% of the observations. Most other columns involved in the Bootstrap Forest analysis also contain missing values. The Informative Missing option in the launch window ensures that the missing values are treated in a way that acknowledges any information that they carry. See "Informative Missing".

JMP PRO Example of Bootstrap Forest with a Continuous Response

In this example, you construct a bootstrap forest model to predict the percent body fat for male subjects.

1. Select **Help > Sample Data Library** and open Body Fat.jmp.
2. Select **Analyze > Predictive Modeling > Bootstrap Forest**.
3. Select Percent body fat and click **Y, Response**.
4. Select Age (years) through Wrist circumference (cm) and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.
7. (Optional) Select **Suppress Multithreading** and enter 123 next to Random Seed.

   Because the bootstrap forest method involves random sampling, these actions ensure that your results are exactly the same as the results shown below.

8. Click **OK**.

Figure 5.5 Overall Statistics

Bootstrap Forest for Percent body fat

Specifications

Target	Percent body fat	Training Rows:	180
Validation Column:	Validation	Validation Rows:	72
		Test Rows:	0
Number of Trees in the Forest:	100	Number of Terms:	13
Number of Terms Sampled per Split:	10	Bootstrap Samples:	180
		Minimum Splits per Tree:	10
		Minimum Size Split:	5

Overall Statistics

Individual Trees	RASE
In Bag	2.025251
Out of Bag	5.995064

	RSquare	RASE	N
Training	0.892	2.6941446	180
Validation	0.675	4.9577438	72

The Overall Statistics report shows that the Validation RSquare is 0.675.

You are interested in obtaining a model-independent indication of the most important predictors.

9. Click the red triangle next to Bootstrap Forest for Percent body fat and select **Column Contributions**.

Figure 5.6 Column Contributions

Column Contributions

Term	Number of Splits	SS	Portion
Abdomen circumference (cm)	106	4349.56265	0.6453
Chest circumference (cm)	35	700.64626	0.1039
Age (years)	46	287.666673	0.0427
Knee circumference (cm)	36	269.801386	0.0400
Height (inches)	44	189.408593	0.0281
Wrist circumference (cm)	48	171.226928	0.0254
Thigh circumference (cm)	33	164.742015	0.0244
Weight (lbs)	24	147.372437	0.0219
Hip circumference (cm)	34	114.631003	0.0170
Ankle circumference (cm)	29	101.095245	0.0150
Neck circumference (cm)	32	94.1033567	0.0140
Biceps (extended) circumference (cm)	27	83.2687719	0.0124
Forearm circumference (cm)	28	66.9446242	0.0099

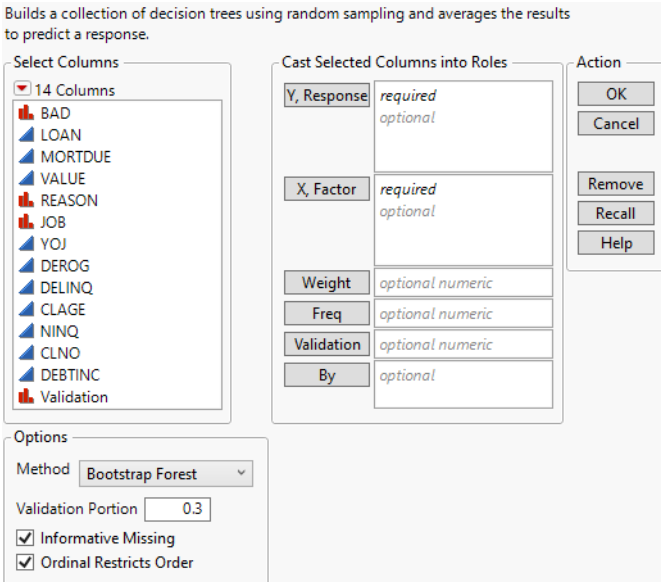
The Column Contributions report suggests that Abdomen circumference (cm), Chest circumference (cm), and Age (years) are the strongest predictors for Percent body fat.

JMP PRO Launch the Bootstrap Forest Platform

Launch the Bootstrap Forest platform by selecting **Analyze > Predictive Modeling > Bootstrap Forest**.

JMP PRO Launch Window

Figure 5.7 Bootstrap Forest Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Bootstrap Forest platform launch provides the following options:

Y, Response The response variable or variables that you want to analyze.

X, Factor The predictor variables.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

Validation A numeric column that defines the validation sets. This column should contain at most three distinct values:.

- If the validation column has two levels, the smaller value defines the training set and the larger value defines the validation set.

- If the validation column has three levels, the values, in order of increasing size, define the training, validation, and test sets.
- If the validation column has more than three levels, the rows that contain the smallest three values define the validation sets. All other rows are excluded from the analysis.

The Bootstrap Forest platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Method Enables you to select the partition method (Decision Tree, Bootstrap Forest, Boosted Tree, K Nearest Neighbors, or Naive Bayes). These alternative methods, except for Decision Tree, are available in JMP Pro.

For more information about these methods, see "Partition Models", "Boosted Tree", "K Nearest Neighbors", and "Naive Bayes".

Validation Portion The portion of the data to be used as the validation set.

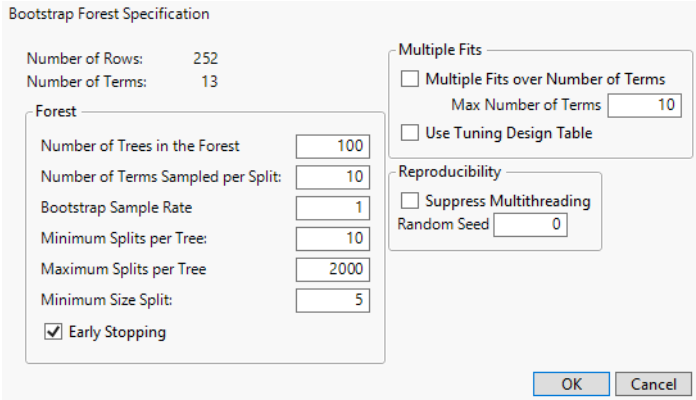
Informative Missing If selected, enables missing value categorization for categorical predictors and informative treatment of missing values for continuous predictors. See "Informative Missing".

Ordinal Restricts Order If selected, restricts consideration of splits to those that preserve the ordering.

JMP PRO Specification Window

After you select **OK** in the launch window, the Bootstrap Forest Specification window appears.

Figure 5.8 Bootstrap Forest Specification Window



JMP PRO Specification Panel

Number of Rows The number of rows in the data table.

Number of Terms The number of columns that are specified as predictors.

JMP PRO Forest Panel

Number of Trees in the Forest The number of trees to grow and then average.

Number of Terms Sampled per Split The number of predictors to consider as splitting candidates at each split. For each split, a new random sample of predictors is taken as the candidate set.

Bootstrap Sample Rate The proportion of observations to sample (with replacement) for growing each tree. A new random sample is generated for each tree.

Minimum Splits Per Tree The minimum number of splits for each tree.

Maximum Splits Per Tree The maximum number of splits for each tree.

Minimum Size Split The minimum number of observations needed on a candidate split.

Early Stopping (Available only if validation is used.) If selected, the process stops growing additional trees if the additional trees do not improve the validation statistic. The validation statistic is the validation set's Entropy RSquare value for a categorical response

and its RSquare value for a continuous response. If not selected, the process continues until the specified number of trees is reached.

JMP PRO Multiple Fits Panel

Multiple Fits over Number of Terms If selected, creates a bootstrap forest for several values of number of terms sampled per split. The model for which results are displayed is the model whose Validation Set's Entropy RSquare value (for a categorical response) or RSquare (for a continuous response) is the largest.

The lower bound is the Number of Terms Sampled per Split specification. The upper bound is specified by the following option:

Max Number of Terms The maximum number of terms to consider for a split.

Use Tuning Table Design Opens a window where you can select a data table containing values for the Forest panel tuning parameters, called a *tuning design table*. A tuning design table has a column for each option that you want to specify and has one or multiple rows that each represent a single Bootstrap Forest model design. If an option is not specified in the tuning design table, the default value is used.

For each row in the table, JMP creates a Bootstrap Forest model using the tuning parameters specified. If more than one model is specified in the tuning design table, the Model Validation-Set Summaries report lists the RSquare value for each model. The Bootstrap Forest report shows the fit statistics for the model with the largest RSquare value.

You can create a tuning design table using the Design of Experiments facilities. A bootstrap forest tuning design table can contain the following case-insensitive columns in any order:

- Number Trees
- Number Terms
- Portion Bootstrap
- Minimum Splits per Tree
- Maximum Splits per Tree
- Minimum Size Split

JMP PRO Reproducibility Panel

Suppress Multithreading If selected, all calculations are performed on a single thread.

Random Seed Specify a nonzero numeric random seed to reproduce the results for future launches of the platform. By default, the Random Seed is set to zero, which does not

produce reproducible results. When you save the analysis to a script, the random seed that you enter is saved to the script.

JMP PRO The Bootstrap Forest Report

After you click **OK** in the Bootstrap Forest Specification window, a window appears that informs you of approximately how long it will take JMP to fit the model. If the approximated time is too long, you have the option to view the current model by selecting **Accept Current Model**. The Bootstrap Forest report appears after the complete model is fit or after the current model is accepted.

Figure 5.9 Accept Current Model Dialog

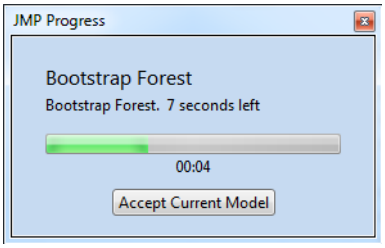


Figure 5.10 Bootstrap Forest Report for a Categorical Response

Bootstrap Forest for BAD

Specifications

Overall Statistics

Measure	Training	Validation	Test	Definition
Entropy RSquare	0.5124	0.4828	0.4824	1-Loglike(model)/Loglike(0)
Generalized RSquare	0.6302	0.6153	0.6068	(1-(L(0)/L(model))^(2/n))/(1-L(0)^(2/n))
Mean -Log p	0.2354	0.2793	0.2618	∑ -Log(p[j])/n
RASE	0.2596	0.2886	0.2771	√ ∑(y[j]-p[j])²/n
Mean Abs Dev	0.1580	0.1779	0.1688	∑ |y[j]-p[j]|/n
Misclassification Rate	0.0898	0.1133	0.0990	∑ (p[j]≠pMax)/n
N	3576	1192	1192	n

Confusion Matrix

Training

Actual	Predicted Count	
BAD	Good Risk	Bad Risk
Good Risk	2819	86
Bad Risk	235	436

Actual	Predicted Rate	
BAD	Good Risk	Bad Risk
Good Risk	0.970	0.030
Bad Risk	0.350	0.650

Validation

Actual	Predicted Count	
BAD	Good Risk	Bad Risk
Good Risk	884	33
Bad Risk	102	173

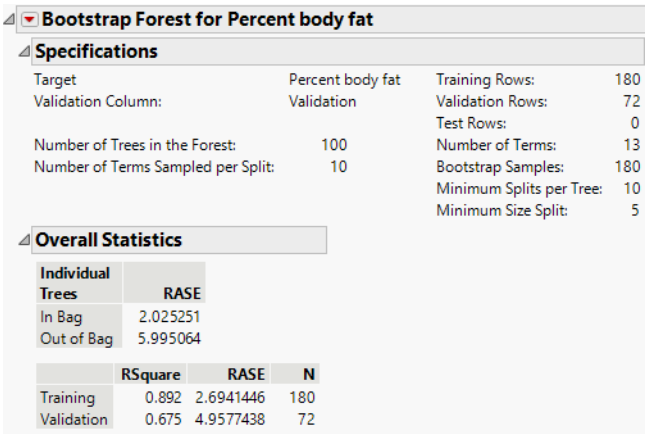
Actual	Predicted Rate	
BAD	Good Risk	Bad Risk
Good Risk	0.964	0.036
Bad Risk	0.371	0.629

Test

Actual	Predicted Count	
BAD	Good Risk	Bad Risk
Good Risk	921	28
Bad Risk	90	153

Actual	Predicted Rate	
BAD	Good Risk	Bad Risk
Good Risk	0.970	0.030
Bad Risk	0.370	0.630

Figure 5.11 Bootstrap Forest Report for a Continuous Response



Bootstrap Forest for Percent body fat

Specifications

Target	Percent body fat	Training Rows:	180
Validation Column:	Validation	Validation Rows:	72
		Test Rows:	0
Number of Trees in the Forest:	100	Number of Terms:	13
Number of Terms Sampled per Split:	10	Bootstrap Samples:	180
		Minimum Splits per Tree:	10
		Minimum Size Split:	5

Overall Statistics

Individual Trees	RASE
In Bag	2.025251
Out of Bag	5.995064

	RSquare	RASE	N
Training	0.892	2.6941446	180
Validation	0.675	4.9577438	72

The following reports are provided, depending on whether the response is categorical or continuous:

- "Model Validation-Set Summaries"
- "Specifications"
- "Overall Statistics"
- "Cumulative Validation"
- "Per-Tree Summaries"

JMP PRO Model Validation-Set Summaries

(Available when you select the Multiple Fits over Number of Terms option in Bootstrap Forest Specification window.) Provides fit statistics for all the models fit. See Figure 5.10 and "Multiple Fits Panel".

JMP PRO Specifications

Shows the settings used in fitting the model.

JMP PRO Overall Statistics

Provides fit statistics for the training set, and for the validation and test sets if they are specified. The specific form of the report depends on the modeling type of the response.

Suppose that multiple models are fit using the Multiple Fits over Multiple Terms option in the Bootstrap Forest Specification window. Then the model for which results are displayed in the Overall Statistics and Cumulative Validation reports is the model for which the validation set's Entropy RSquare value (for a categorical response) or RSquare (for a continuous response) is the largest.

JMP PRO Categorical Response

Measures Report

Gives the following statistics for the training set, and for the validation and test sets if they are specified.

Note: For Entropy RSquare and Generalized RSquare, values closer to 1 indicate a better fit. For Mean -Log p, RASE, Mean Abs Dev, and Misclassification Rate, smaller values indicate a better fit.

Entropy RSquare A measure of fit that compares the log-likelihoods from the fitted model and the constant probability model. It ranges from 0 to 1. See "Entropy RSquare".

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The value is 1 for a perfect model, and 0 for a model no better than a constant model. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R2, which is a normalized version of Cox and Snell's pseudo R2.

Mean -Log P The average of negative log(p), where p is the fitted probability associated with the event that occurred.

RASE The root average squared prediction error. The differences are between 1 and p, the fitted probability for the response level that actually occurred.

Mean Abs Dev The average of the absolute values of the differences between the response and the predicted response. The differences are between 1 and p, the fitted probability for the response level that actually occurred.

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category.

N The number of observations.

Confusion Matrix Report

(Available only for categorical responses.) Shows classification statistics for the training set, and for the validation and test sets if they are specified. The Confusion Matrix Report contains confusion matrices and confusion rates matrices. A confusion matrix is a two-way classification of actual and predicted responses. A confusion rates matrix is equal to the confusion matrix, with the numbers divided by the row totals.

Decision Matrix

(Available only for categorical responses and if the response has a Profit Matrix column property or if you specify costs using the Specify Profit Matrix option.) Gives Decision Count and Decision Rate matrices for the training set, and for the validation and test sets if they are specified. See "Additional Examples of Partitioning".

JMP PRO Continuous Response

Individual Trees Report

Gives RASE values, which are averaged over all trees, for In Bag and Out of Bag observations. Training set observations that are used to construct a tree are called *in-bag* observations. Training observations that are not used to construct a tree are called *out-of-bag* (OOB) observations.

For each tree, the Out of Bag RASE is computed as the square root of the sum of squared errors divided by the number of OOB observations. The squared Out of Bag RASE for each tree is given in the Per-Tree Summaries report as OOB SSE/N.

RSquare and RASE Report

Gives Rsquare, root average squared prediction error, and the number of observations for the training set, and for the validation and test sets, if they are defined.

JMP PRO Cumulative Validation

(Available only if validation is used.) Shows a plot of the fit statistics for the Validation set versus the number of trees.

For a continuous response, the single fit statistic is R-Square. For a categorical response, the fit statistics are listed below and are described in "Measures Report".

- RSquare (Entropy RSquare)
- Avg - Log p (Mean - Log p)
- Root Average Squared Prediction Error (RASE)

- Avg Abs Error (Mean Abs Dev)
- MR (Misclassification Rate)

The Cumulative Details report below the Cumulative Validation plot gives the values used in the plot.

JMP PRO Per-Tree Summaries

The Per-Tree Summaries report involves the concepts of in-bag and out-of-bag observations. For an individual tree, the bootstrap sample of observations used in fitting the tree is drawn with replacement. Even if you specify that 100% of the observations are to be sampled, because they are drawn with replacement, the expected proportion of unused observations is $1/e$. For each individual tree, the unused observations are called the *out-of-bag* observations. The observations used in fitting the tree are called *in-bag* observations.

The Per-Tree Summaries report shows the following summary statistics for each tree:

Splits The number of splits in the decision tree.

Rank The rank of the tree's OOB Loss in ascending order. The tree with the smallest OOB loss has Rank 1.

OOB Loss A measure of the total predictive inaccuracy of the tree when applied to the Out Of Bag rows. Lower values indicate a higher predictive accuracy.

OOB Loss/N The OOB Loss divided by the number of OOB rows, OOB N.

RSquare (Available only for continuous responses.) The RSquare value for the tree.

IB SSE (Available only for continuous responses.) Sum of squared errors for the In Bag rows.

IB SSE/N (Available only for continuous responses.) Sum of squared errors for the In Bag rows divided by the number of In Bag observations. The number of In Bag observations is equal to the number of observations in the training set multiplied by the bootstrap sampling rate that you specify in the Bootstrap Forest Specification window.

OOB N (Available only for continuous responses.) The number of Out Of Bag rows.

OOB SSE (Available only for continuous responses.) Sum of squared errors when the tree is applied to the Out Of Bag rows.

OOB SSE/N (Available only for continuous responses.) The OOB SSE divided by the number of OOB rows, OOB N.

JMP PRO Bootstrap Forest Platform Options

The Bootstrap Forest red triangle menu contains the following options:

Plot Actual by Predicted (Available only for continuous responses.) Provides a plot of actual versus predicted values.

Column Contributions Displays a report that shows each input column's contribution to the fit. The report also shows:

- The total number of instances over all of the trees when the specified column is used to split the data.
- The total G^2 (for a categorical response) or SS, sum of squares (for a continuous response), attributed to the column.
- A bar chart of G^2 or SS.
- The proportion of G^2 or SS attributed to the column.

Show Trees Provides various options for displaying trees in the Tree Views report. The report gives a picture of the tree that is fit at each layer of the boosting process. For a description of the Prob column shown by the Show names categories estimates option, see "Predicted Probabilities in Decision Tree and Bootstrap Forest".

ROC Curve (Available only for categorical responses.) See "ROC Curve".

Lift Curve (Available only for categorical responses.) See "Lift Curve".

Save Columns Contains options for saving model and tree results, and creating SAS code.

Save Predicteds Saves the predicted values from the model to the data table.

Save Prediction Formula Saves the prediction formula to a column in the data table. The formula consists of nested conditional clauses that describe the tree structure. If the response is continuous, the column contains a Predicting property. If the response is categorical, the column contains a Response Probability property.

Save Tolerant Prediction Formula (The Save Prediction Formula option should be used instead of this option. Use this option only when Save Prediction Formula is not available.) Saves a formula that predicts even when there are missing values and when Informative Missing has not been selected. The prediction formula tolerates missing values by randomly allocating response values for missing predictors to a split. If the response is continuous, the column contains a Predicting property. If the response is categorical, the column contains a Response Probability property. If you have selected Informative Missing, you can save the Tolerant Prediction Formula by holding the Shift key as you click the report's red triangle.

Save Residuals (Available only for continuous responses.) Saves the residuals to the data table.

Save Cumulative Details (Available only if validation is used.) Creates a data table containing the fit statistics for each tree.

Publish Prediction Formula Creates a prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot".

Publish Tolerant Prediction Formula (The Publish Prediction Formula option should be used instead of this option. Use this option only when Publish Prediction Formula is not available.) Creates a tolerant prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot". If you have selected Informative Missing, you can use this option by holding the Shift key as you click the report's red triangle.

Make SAS DATA Step Creates SAS code for scoring a new data set.

Specify Profit Matrix (Available only for categorical responses.) Enables you to specify profit or costs associated with correct or incorrect classification decisions. See "Show Fit Details".

Profiler Shows a Prediction Profiler. See *Profilers*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Chapter 6

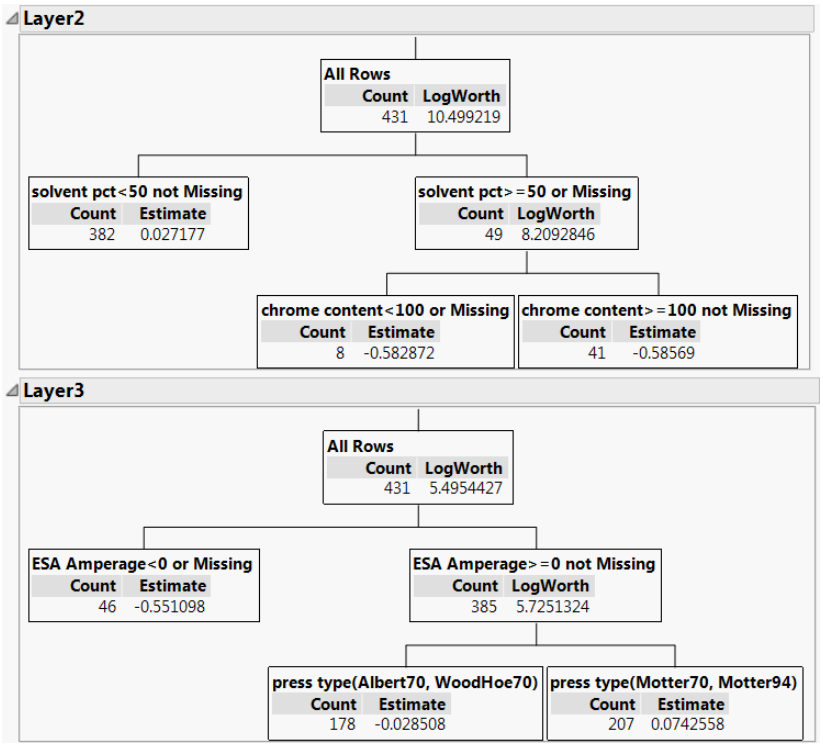
JMP PRO Boosted Tree

Fit Many Layers of Trees, Each Based on the Previous Layer

The Boosted Tree platform is available only in JMP Pro.

Boosting is the process of building a large, additive decision tree by fitting a sequence of smaller decision trees, called *layers*. The tree at each layer consists of a small number of splits. The tree is fit based on the residuals of the previous layers, which allows each layer to correct the fit for bad fitting data from the previous layers. The final prediction for an observation is the sum of the predictions for that observation over all of the layers.

Figure 6.1 Example of Boosted Tree Layers



Contents

Overview of the Boosted Tree Platform . . . 109
Example of Boosted Tree with a Categorical Response . . . 109
Example of Boosted Tree with a Continuous Response . . . 111
Launch the Boosted Tree Platform . . . 113
Specification Window . . . 115
The Boosted Tree Report . . . 117
Model Validation - Set Summaries . . . 118
Specifications . . . 119
Overall Statistics . . . 119
Cumulative Validation . . . 120
Boosted Tree Platform Options . . . 121
Statistical Details for the Boosted Tree Platform . . . 123
Overfit Penalty . . . 123

JMP PRO Overview of the Boosted Tree Platform

The Boosted Tree platform produces an additive decision tree model that is based on many smaller decision trees that are constructed in *layers*. The tree in each layer consists of a small number of splits, typically five or fewer. Each layer is fit using the recursive fitting methodology described in "Partition Models". The only difference is that fitting stops at a specified number of splits. For a given tree, the predicted value for an observation in a leaf is the mean of all observations in that leaf.

This is the fitting process:

1. Fit an initial layer.
2. Compute residuals. These are obtained by subtracting the predicted mean for observations within a leaf from their actual value.
3. Fit a layer to the residuals.
4. Construct the additive tree. For a given observation, sum its predicted values over the layers.
5. Repeat step 2 to step 4 until the specified number of layers is reached, or, if validation is used, until fitting an additional layer no longer improves the validation statistic.

The final prediction is the sum of the predictions for an observation over all the layers.

By fitting successive layers on residuals from previous layers, each layer can improve the fit.

For categorical responses, only those with two response levels are supported. For a categorical response, the residuals fit at each layer are offsets of linear logits. The final prediction is a logistic transformation of the sum of the linear logits over all the layers.

For more information about boosted trees, see Hastie et al. (2009).

JMP PRO Example of Boosted Tree with a Categorical Response

In this example, you construct a boosted tree model to predict which printing jobs are affected by a defect called *banding*.

1. Select **Help > Sample Data** and open Bands Data.jmp.
2. Select **Analyze > Predictive Modeling > Boosted Tree**.
3. Select Banding? and click **Y, Response**.
4. Select the Predictors column group and click **X, Factor**.
5. Enter 0.2 for **Validation Portion**.

6. Click **OK**.

   The Boosted Tree Specification window appears.

7. (Optional) In the Reproducibility panel, select **Suppress Multithreading** and enter 123 for Random Seed.

   Because the boosted tree fit involves a random component, these actions ensure that you obtain the exact results shown below.

8. Click **OK**.

Figure 6.2 Overall Statistics for Nominal Response

Overall Statistics

Measure	Training	Validation	Definition
Entropy RSquare	0.7103	0.3299	1-Loglike(model)/Loglike(0)
Generalized RSquare	0.8340	0.4803	(1-(L(0)/L(model))^(2/n))/(1-L(0)^(2/n))
Mean -Log p	0.1985	0.4383	∑ -Log(ρ[j])/n
RASE	0.2196	0.3801	√ ∑(y[j]-ρ[j])²/n
Mean Abs Dev	0.1650	0.2828	∑ |y[j]-ρ[j]|/n
Misclassification Rate	0.0232	0.1852	∑ (ρ[j]≠ρMax)/n
N	431	108	n

Confusion Matrix

Training

Actual Banding?	Predicted Count band	noband
band	179	9
noband	1	242

Actual Banding?	Predicted Rate band	noband
band	0.952	0.048
noband	0.004	0.996

Validation

Actual Banding?	Predicted Count band	noband
band	22	17
noband	3	66

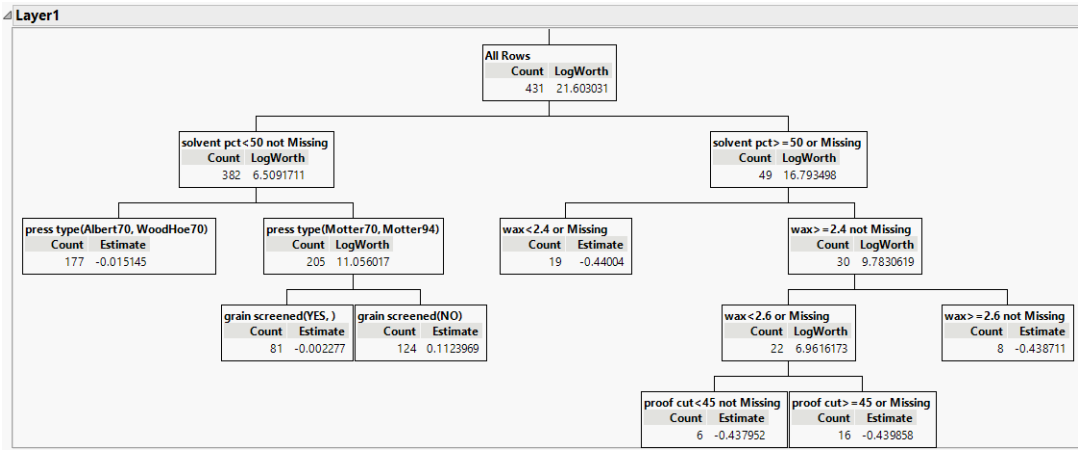
Actual Banding?	Predicted Rate band	noband
band	0.564	0.436
noband	0.043	0.957

Because the response, Banding?, is categorical, the Boosted Tree analysis provides a Misclassification Rate under Measure and a Confusion Matrix report. The Misclassification Rate for the validation set is 0.1852, or about 19%.

9. Click the red triangle next to Boosted Tree for Banding? and select **Show Trees > Show names categories estimates**.

   A Tree Views report appears, with outlines for the layers. You can examine the layers to see the trees that are fit and the predicted values.

Figure 6.3 Layer 1 of the Boosted Tree



10. Click the red triangle next to Boosted Tree for Banding? and select **Save Columns > Save Prediction Formula**.

    Columns called Prob(Banding?==noband), Prob(Banding?==band), and Most Likely Banding? are added to the data table. Examine the Prob(Banding?==noband) column to see how model predictions are calculated from the layers.

JMP PRO Example of Boosted Tree with a Continuous Response

In this example, you construct a boosted tree model to predict the percent body fat given a combination of nominal and continuous factors.

1. Select **Help > Sample Data** and open the Body Fat.jmp sample data table.
2. Select **Analyze > Predictive Modeling > Boosted Tree**.
3. Select Percent body fat and click **Y, Response**.
4. Select Age (years) through Wrist circumference (cm) and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.
7. Click **OK**.

Figure 6.4 Overall Statistics for Continuous Response

Overall Statistics

	RSquare	RASE	N
Training	0.848	3.2012478	180
Validation	0.611	5.4301412	72

The Overall Statistics report provides the R-square and RASE for the boosted tree model. The R-square for the validation set is 0.611. The RASE for the validation set is about 5.43.

You are interested in obtaining a model-independent indication of the important predictors for Percent body fat.

8. Click the red triangle next to Boosted Tree for Percent body fat and select **Profiler**.
9. Click the red triangle next to Prediction Profiler and select **Assess Variable Importance > Independent Uniform Inputs**.

Note: Because Assess Variable Importance uses randomization, your results might not exactly match those in Figure 6.5.

Figure 6.5 Summary Report for Variable Importance

Summary Report

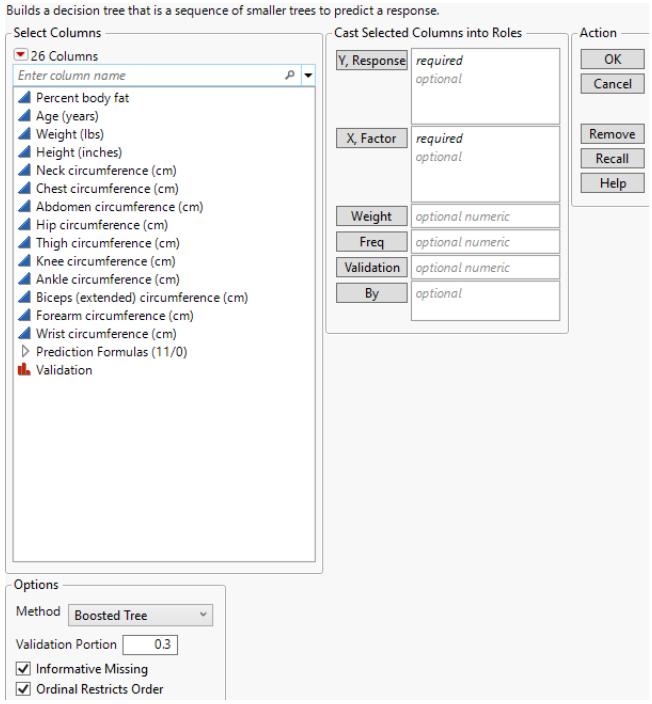
Column	Main Effect	Total Effect	.2 .4 .6 .8
Abdomen circumference (cm)	0.865	0.884	
Age (years)	0.065	0.084	
Wrist circumference (cm)	0.005	0.01	
Hip circumference (cm)	0.004	0.007	
Biceps (extended) circumference (cm)	0.003	0.005	
Chest circumference (cm)	0.002	0.004	
Ankle circumference (cm)	0.002	0.003	
Height (inches)	0.001	0.002	
Weight (lbs)	0.001	0.002	
Thigh circumference (cm)	0.001	0.002	
Knee circumference (cm)	0.001	0.001	
Neck circumference (cm)	9e-18	2e-17	
Forearm circumference (cm)	9e-18	2e-17	

The Summary Report shows that Abdomen circumference (cm) is the most important predictor of Percent body fat.

JMP PRO Launch the Boosted Tree Platform

Launch the Boosted Tree platform by selecting **Analyze > Predictive Modeling > Boosted Tree**.

Figure 6.6 Boosted Tree Launch Window Using Body Fat.jmp



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Boosted Tree platform launch window has the following options:

Y, Response The response variable or variables that you want to analyze.

X, Factor The predictor variables.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

Validation A numeric column that defines the validation sets. This column should contain at most three distinct values:

- If the validation column has two levels, the smaller value defines the training set and the larger value defines the validation set.

- If the validation column has three levels, the values, in order of increasing size, define the training, validation, and test sets.
- If the validation column has more than three levels, the rows that contain the smallest three values define the validation sets. All other rows are excluded from the analysis.

The Boosted Tree platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Method Enables you to select the partition method (Decision Tree, Bootstrap Forest, Boosted Tree, K Nearest Neighbors, or Naive Bayes). These alternative methods, except for Decision Tree, are available in JMP Pro.

For more information about these methods, see "Partition Models", "Bootstrap Forest", "K Nearest Neighbors", and "Naive Bayes".

Validation Portion The portion of the data to be used as the validation set.

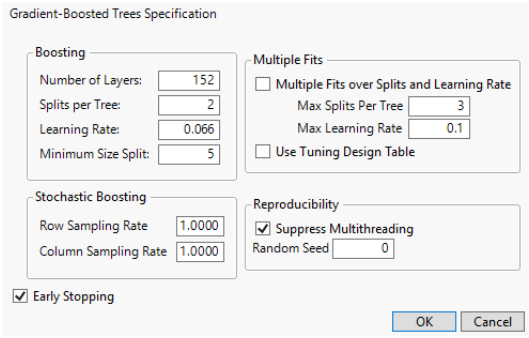
Informative Missing If selected, enables missing value categorization for categorical predictors and informative treatment of missing values for continuous predictors. See "Informative Missing".

Ordinal Restricts Order If selected, restricts consideration of splits to those that preserve the ordering.

JMP PRO Specification Window

After you select **OK** in the launch window, the Gradient-Boosted Trees Specification window appears.

Figure 6.7 Boosted Tree Specification Window



JMP PRO Boosting Panel

Number of Layers The maximum number of layers to include in the final tree.

Splits per Tree The number of splits for each layer.

Learning Rate A number such that $0 < r \leq 1$. Learning rates close to 1 result in faster convergence on a final tree, but also have a higher tendency to overfit data. Use learning rates closer to 1 when a small Number of Layers is specified. The learning rate is a small fraction typically between 0.01 and 0.1 that slows the convergence of the model. This preserves opportunities for later layers to use different splits than the earlier layers.

Overfit Penalty (Available only for categorical responses.) A biasing parameter that helps protect against fitting probabilities equal to zero.See "Overfit Penalty".

Minimum Size Split The minimum number of observations needed on a candidate split.

JMP PRO Multiple Fits Panel

Multiple Fits over Splits and Learning Rate If selected, creates a boosted tree for every combination of Splits per Tree (in integer increments) and Learning Rate (in 0.1 increments).

The lower bounds for the combinations are specified by the Splits per Tree and Learning Rate options. The upper bounds for the combinations are specified by the following options:

Max Splits per Tree Upper bound for Splits per Tree.

Max Learning Rate Lower bound for Learning Rate.

Use Tuning Design Table Opens a window where you can select a data table containing values for some tuning parameters, called a *tuning design table*. A tuning design table has a column for each option that you want to specify and has one or multiple rows that each represent a single Boosted Tree model design. If an option is not specified in the tuning design table, the default value is used.

For each row in the table, JMP creates a Boosted Tree model using the tuning parameters specified. If more than one model is specified in the tuning design table, the Model Validation-Set Summaries report lists the R-Square value for each model. The Boosted Tree report shows the fit statistics for the model with the largest R-Square value.

You can create a tuning design table using the Design of Experiments facilities. A boosted tree tuning design table can contain the following case-insensitive columns in any order:

- Number of Layers
- Splits per Tree
- Learning Rate
- Minimum Size Split
- Row Sampling Rate
- Column Sampling Rate

JMP PRO Stochastic Boosting Panel

Row Sampling Rate Proportion of training rows to sample for each layer.

Note: When the response is categorical, the training rows are sampled using stratified random sampling.

Column Sampling Rate Proportion of predictor columns to sample for each layer.

JMP PRO Reproducibility Panel

Suppress Multithreading If selected, all calculations are performed on a single thread.

Random Seed Specify a nonzero numeric random seed to reproduce the results for future launches of the platform. By default, the Random Seed is set to zero, which does not produce reproducible results. When you save the analysis to a script, the random seed that you enter is saved to the script.

JMP PRO Early Stopping

Early Stopping If selected, the boosting process stops fitting additional layers when the additional layers do not improve the validation statistic. If not selected, the boosting process continues until the specified number of layers is reached. This option appears only if validation is used.

JMP PRO The Boosted Tree Report

After you click OK in the Gradient-Boosted Trees Specification window, the Boosted Tree report opens.

Figure 6.8 Boosted Tree Report for a Continuous Response

Boosted Tree for Percent body fat

Model Validation-Set Summaries

The fit below was the best of these models fit.

N Splits	Learning Rate	Row Sampling Rate	Column Sampling Rate	N Layers	Minimum Size Split	RSquare	RMSE
3	0.1	1	1	30	5	0.6034	5.4804
4	0.1	1	1	31	5	0.6160	5.3920
5	0.1	1	1	26	5	0.6215	5.3533
3	0.2	1	1	35	5	0.6074	5.4522
4	0.2	1	1	20	5	0.6146	5.4023
5	0.2	1	1	20	5	0.5931	5.5508

Specifications

Target Column:	Percent body fat	Number of training rows:	180
Validation Column:	Validation	Number of validation rows:	72
Number of Layers:	26		
Splits per Tree:	5		
Learning Rate:	0.1		

Overall Statistics

	RSquare	RMSE	N
Training	0.864	3.0203083	180
Validation	0.622	5.3533415	72

Cumulative Validation

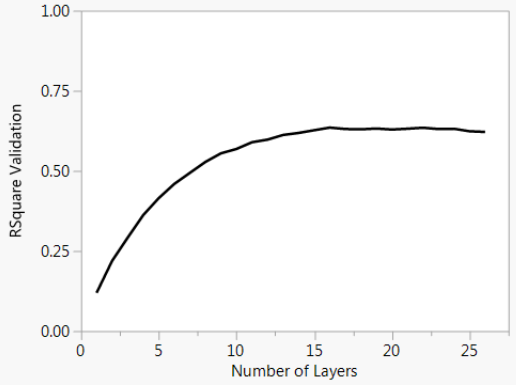
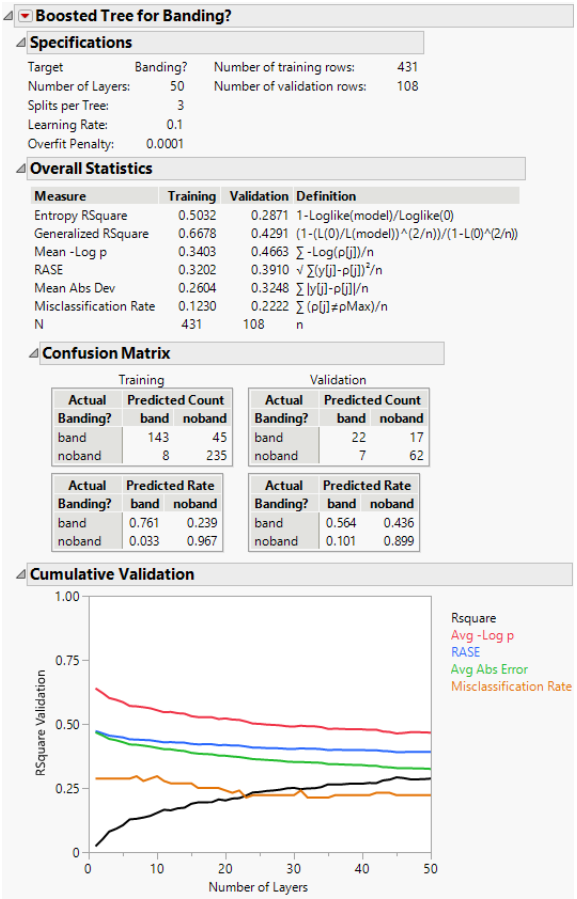


Figure 6.9 Boosted Tree Report for a Categorical Response



The following reports are provided, depending on whether the response is categorical or continuous:

- "Model Validation - Set Summaries"
- "Specifications"
- "Overall Statistics"
- "Cumulative Validation"

JMP PRO Model Validation - Set Summaries

Shows fit statistics for all the models fit if you selected the Multiple Fits over Splits and Learning Rate option in the Specification window. See Figure 6.8 and "Multiple Fits Panel".

JMP PRO Specifications

Shows the settings used in fitting the model.

JMP PRO Overall Statistics

Shows fit statistics for the training set, and for the validation and test sets if they are specified.

Suppose that you fit multiple models using the Multiple Fits over Splits and Learning Rate option in the Boosted Tree Specification window. Then the model for which results are displayed in the Overall Statistics and Cumulative Validation reports is the model for which the validation set's Entropy R-square value (for a categorical response) or R-square (for a continuous response) is the largest.

JMP PRO Measures Report

(Available only for categorical responses.) Gives the following statistics for the training set, and for the validation and test sets if they are specified.

Note: For Entropy R-Square and Generalized R-Square, values closer to 1 indicate a better fit. For Mean -Log p, RASE, Mean Abs Dev, and Misclassification Rate, smaller values indicate a better fit.

Entropy RSquare A measure of fit that compares the log-likelihoods from the fitted model and the constant probability model. It ranges from 0 to 1. See "Entropy RSquare".

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The value is 1 for a perfect model, and 0 for a model no better than a constant model. The Generalized R-Square measure simplifies to the traditional R-Square for continuous normal responses in the standard least squares setting. Generalized R-Square is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2.

Mean -Log P The average of negative log(p), where p is the fitted probability associated with the event that occurred.

RASE The root average squared prediction error. The differences are between 1 and p, the fitted probability for the response level that actually occurred.

Mean Abs Dev The average of the absolute values of the differences between the response and the predicted response. The differences are between 1 and p, the fitted probability for the response level that actually occurred.

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category.

N The number of observations.

JMP PRO Confusion Matrix Report

(Available only for categorical responses.) Shows classification statistics for the training set, and for the validation and test sets if they are specified. The Confusion Matrix Report contains confusion matrices and confusion rates matrices. A confusion matrix is a two-way classification of actual and predicted responses. A confusion rates matrix is equal to the confusion matrix, with the numbers divided by the row totals.

JMP PRO Decision Matrix

(Available only for categorical responses and if the response has a Profit Matrix column property or if you specify costs using the Specify Profit Matrix option.) Gives Decision Count and Decision Rate matrices for the training set, and for the validation and test sets if they are specified. See "Additional Examples of Partitioning".

JMP PRO Cumulative Validation

(Available only if validation is used.) Shows a plot of the fit statistics for the Validation set versus the number of layers.

For a continuous response, the single fit statistic is R-Square. For a categorical response, the fit statistics are listed below and are described in "Measures Report".

- R-Square (Entropy R-Square)
- Avg - Log p (Mean - Log p)
- Root Average Squared Prediction Error (RASE)
- Avg Abs Error (Mean Abs Dev)
- MR (Misclassification Rate)

The Cumulative Details report below the Cumulative Validation plot gives the values used in the plot.

JMP PRO Boosted Tree Platform Options

The Boosted Tree red triangle menu contains the following options:

Show Trees Provides options for displaying trees in the Tree Views report. The report gives a picture of the tree that is fit at each layer of the boosting process.

Plot Actual by Predicted (Available only for continuous responses.) Provides a plot of actual versus predicted values.

Column Contributions Displays a report showing each input column's contribution to the fit. The report also shows:

- The total number of instances over all of the trees when the specified column is used to split the data.
- The total G^2 (for a categorical response) or SS, sum of squares (for a continuous response) attributed to the column.
- A bar chart of G^2 or SS.
- The proportion of G^2 or SS attributed to the column.

ROC Curve (Available only for categorical responses.) See "ROC Curve".

Lift Curve (Available only for categorical responses.) See "Lift Curve".

Save Columns Contains options for saving model and tree results, and creating SAS code.

Save Predicteds Saves the predicted values from the model to the data table.

Save Prediction Formula Saves the prediction formula to a column in the data table. The formula consists of nested conditional clauses that describe the tree structure. If the response is continuous, the column contains a Predicting column property. If the response is categorical, the column contains a Response Probability column property.

Save Tolerant Prediction Formula (The Save Prediction Formula option should be used instead of this option. Use this option only when Save Prediction Formula is not available.) Saves a formula that predicts even when there are missing values and when Informative Missing has not been selected. The prediction formula tolerates missing values by randomly allocating response values for missing predictors to a split. If the response is continuous, the column contains a Predicting column property. If the response is categorical, the column contains a Response Probability column property. If you have selected Informative Missing, you can save the Tolerant Prediction Formula by holding the Shift key as you click the report's red triangle.

Save Residuals (Available only for continuous responses.) Saves the residuals to the data table.

Save Offset Estimates (Available only for categorical responses.) Saves the sums of the linear components. These are the logits of the fitted probabilities.

Save Tree Details Creates a data table containing split details and estimates for each layer.

Save Cumulative Details (Available only if validation is used.) Creates a data table containing the fit statistics for each layer.

Publish Prediction Formula Creates a prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot".

Publish Tolerant Prediction Formula (The Publish Prediction Formula option should be used instead of this option. Use this option only when Publish Prediction Formula is not available.) Creates a tolerant prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot". If you have selected Informative Missing, you can use this option by holding the Shift key as you click the report's red triangle.

Make SAS DATA Step Creates SAS code for scoring a new data set.

Specify Profit Matrix (Available only for categorical responses.) Enables you to specify profit or costs associated with correct or incorrect classification decisions. See "Show Fit Details".

Profiler Shows a Prediction Profiler. See *Profilers*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Statistical Details for the Boosted Tree Platform

This section describes details specific to the Boosted Tree Platform. For statistical details about recursive decision trees, see "Statistical Details for the Partition Platform".

JMP PRO Overfit Penalty

When the response is categorical, a parametric penalty is imposed. For each layer, the estimates minimize the negative log-likelihood plus the penalty value multiplied by the sum of squares of the estimates for each observation. This penalty encourages each new layer not to overfit the training data.

Chapter 7

JMP PRO K Nearest Neighbors

Predict Response Values Using Nearby Observations

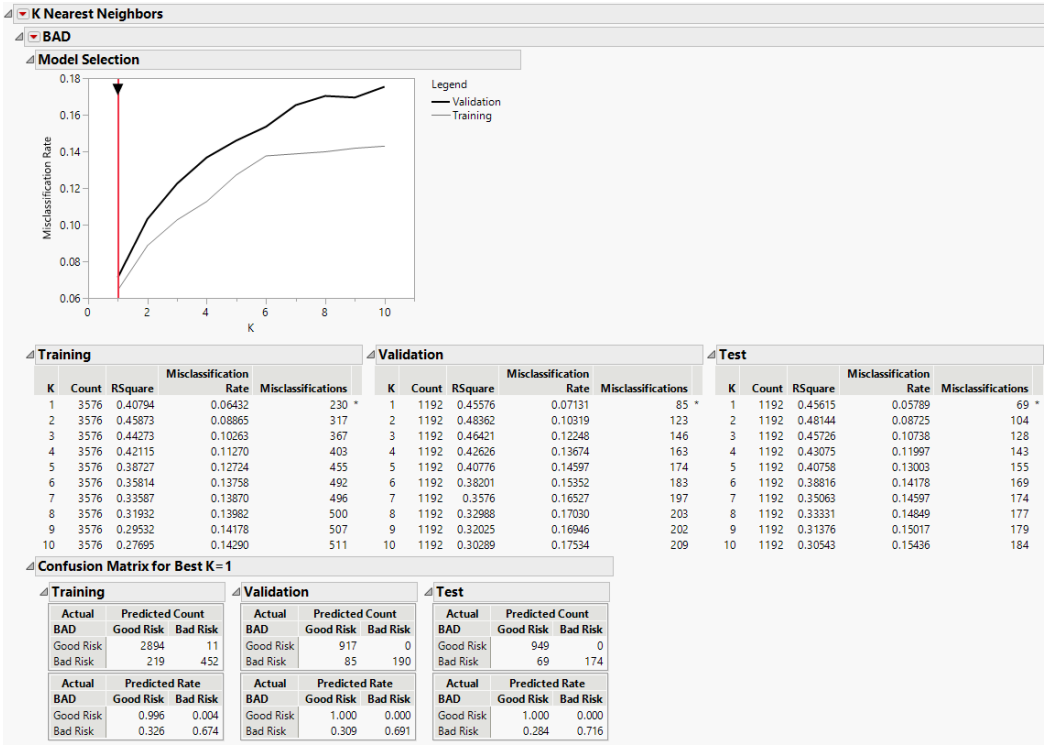
The K Nearest Neighbors platform is available only in JMP Pro.

The K Nearest Neighbors platform predicts a response value for a given observation using the responses of the observations in that observation's local neighborhood. It can be used for classification of a categorical response as well as for prediction of a continuous response.

K Nearest Neighbors is a nonparametric method that is based on the distance to neighboring observations. Because of this fact, K Nearest Neighbors is able to classify observations using irregular predictor value boundaries. However, because the algorithm is sensitive to irrelevant predictors, the selection of predictors can impact your results.

K Nearest Neighbors has been used successfully in many applications, such as classifying satellite imagery and electrocardiogram (EKG) patterns.

Figure 7.1 Example of the K Nearest Neighbors Platform



Contents

Overview of the K Nearest Neighbors Platform 127
Example of K Nearest Neighbors with Categorical Response 128
Example of K Nearest Neighbors with Continuous Response 130
Launch the K Nearest Neighbors Platform 132
The K Nearest Neighbors Report 134
Continuous Responses 134
Categorical Responses 135
K Nearest Neighbors Platform Options 136
Model Fit Options 136

JMP PRO Overview of the K Nearest Neighbors Platform

The K Nearest Neighbors platform predicts a response value based on the responses of the *k* nearest neighbors. The *k* nearest neighbors to a given observation are determined by identifying the *k* smallest Euclidean distances between the predictor values for that observation and the predictor values for each of the other observations. The K Nearest Neighbors platform models both continuous and categorical responses.

A potential drawback of the *k* nearest neighbors method is that for large scale problems, the prediction formula is often complex and hard to interpret, limiting its usefulness. In addition, K Nearest Neighbors does not calculate probabilities for categorical responses. For more information about the *k* nearest neighbors method, see Hastie et al. (2009), Hand et al. (2001), and Shmueli et al. (2017).

Continuous Responses

For a continuous response, the predicted value is the average of the responses for the *k* nearest neighbors. Each continuous predictor is scaled by its standard deviation. With this scaling, a single predictor with a large range does not excessively influence the distance calculation. Missing values for a continuous predictor are replaced by the mean of that predictor. See "Example of K Nearest Neighbors with Continuous Response".

Categorical Responses

For a categorical response, the predicted value is the most frequent response level for the *k* nearest neighbors. If two or more levels are tied as the most frequent levels, the predicted response is assigned by selecting one of these levels at random.

Note: Because ties for most frequent levels in the case of a categorical response are broken at random, results from independent runs of the platform might differ. To obtain reproducible results, use the Set Random Seed option in the launch window or include the `Set Random Seed()` function in a JSL script.

In the categorical prediction models, each categorical predictor is expressed in terms of indicator variables, with one indicator variable representing each level. A row with a missing value for a categorical predictor is represented by values of zero on all indicator variables for that predictor.

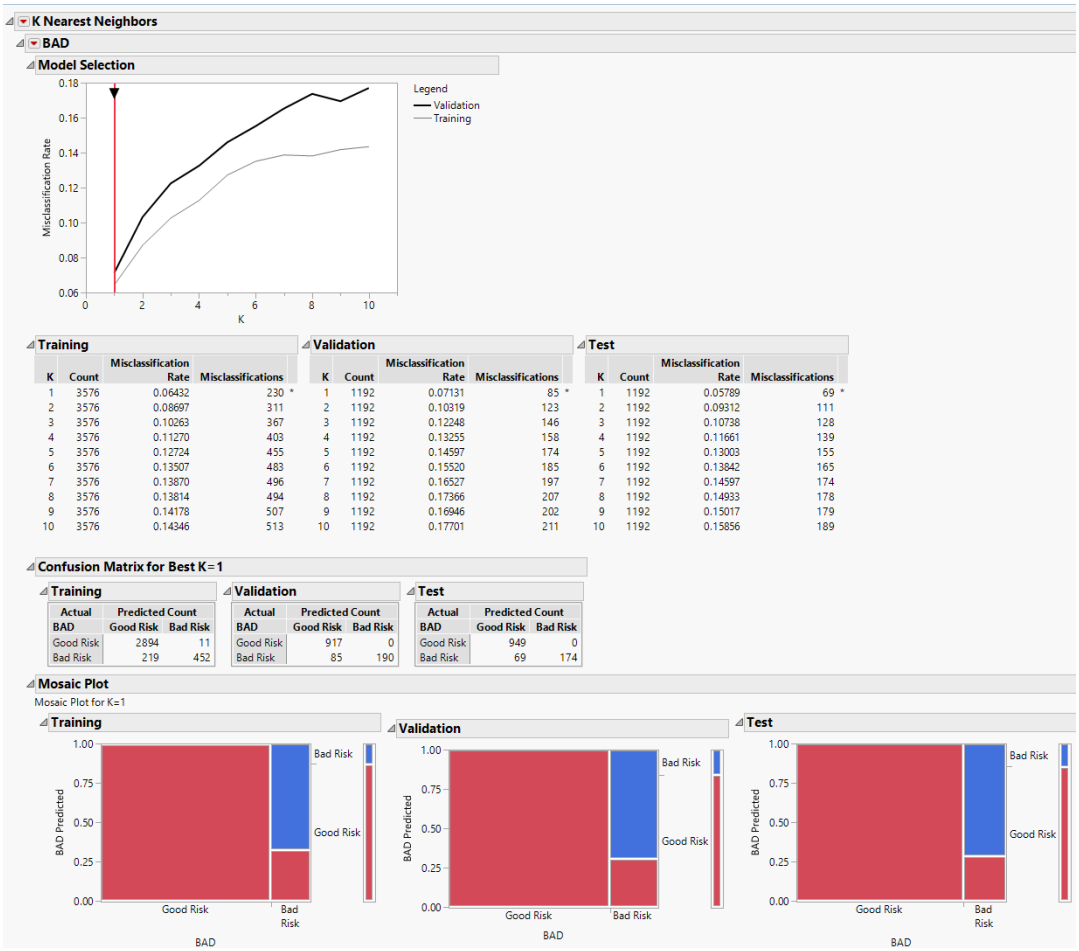
JMP PRO Example of K Nearest Neighbors with Categorical Response

You have historical financial data for 5,960 customers who applied for home equity loans. Each customer was classified as being a Good Risk or Bad Risk. There are missing values for many of the predictors. You want to construct a model to use in classifying the credit risk of future customers.

1. Select **Help > Sample Data Library** and open Equity.jmp.
2. Select **Analyze > Predictive Modeling > K Nearest Neighbors**.
3. Select BAD and click **Y, Response**.
4. Select LOAN through CLNO and click **X, Factor**.

   Because one of the potential predictors, DEBTINC, has many missing values, you do not include it in your model. Missing values for continuous predictors are replaced by the average of the predictor. This procedure sometimes works well for values that are missing at random. Although the high missing rate of the DEBINC indicates that the missing might be informative, we do not investigate that in this example.
5. Select Validation and click **Validation**.
6. Click **OK**.

Figure 7.2 K Nearest Neighbors Report



Training				Validation				Test			
K	Count	Misclassification Rate	Misclassifications	K	Count	Misclassification Rate	Misclassifications	K	Count	Misclassification Rate	Misclassifications
1	3576	0.06432	230 *	1	1192	0.07131	85 *	1	1192	0.05789	69 *
2	3576	0.08697	311	2	1192	0.10319	123	2	1192	0.09312	111
3	3576	0.10263	367	3	1192	0.12248	146	3	1192	0.10738	128
4	3576	0.11270	403	4	1192	0.13255	158	4	1192	0.11661	139
5	3576	0.12724	455	5	1192	0.14597	174	5	1192	0.13003	155
6	3576	0.13507	483	6	1192	0.15520	185	6	1192	0.13842	165
7	3576	0.13870	496	7	1192	0.16527	197	7	1192	0.14597	174
8	3576	0.13814	494	8	1192	0.17366	207	8	1192	0.14933	178
9	3576	0.14178	507	9	1192	0.16946	202	9	1192	0.15017	179
10	3576	0.14346	513	10	1192	0.17701	211	10	1192	0.15856	189

Training			Validation			Test		
Actual BAD	Predicted Count Good Risk	Bad Risk	Actual BAD	Predicted Count Good Risk	Bad Risk	Actual BAD	Predicted Count Good Risk	Bad Risk
Good Risk	2894	11	Good Risk	917	0	Good Risk	949	0
Bad Risk	219	452	Bad Risk	85	190	Bad Risk	69	174

For each value of *K*, JMP constructs a model using only the training set observations. Each of these models is used to classify the validation set observations. The validation set results are used to select a best model. In this example, the model based on the single nearest neighbor ($K = 1$) has the smallest misclassification rate. The test set verifies that the single nearest neighbor model is the best performer for independent data.

7. Click the BAD red triangle and select **Publish Prediction Formula**.
8. Next to **Number of Neighbors, K**, leave the default value of 1.
9. Click **OK**.

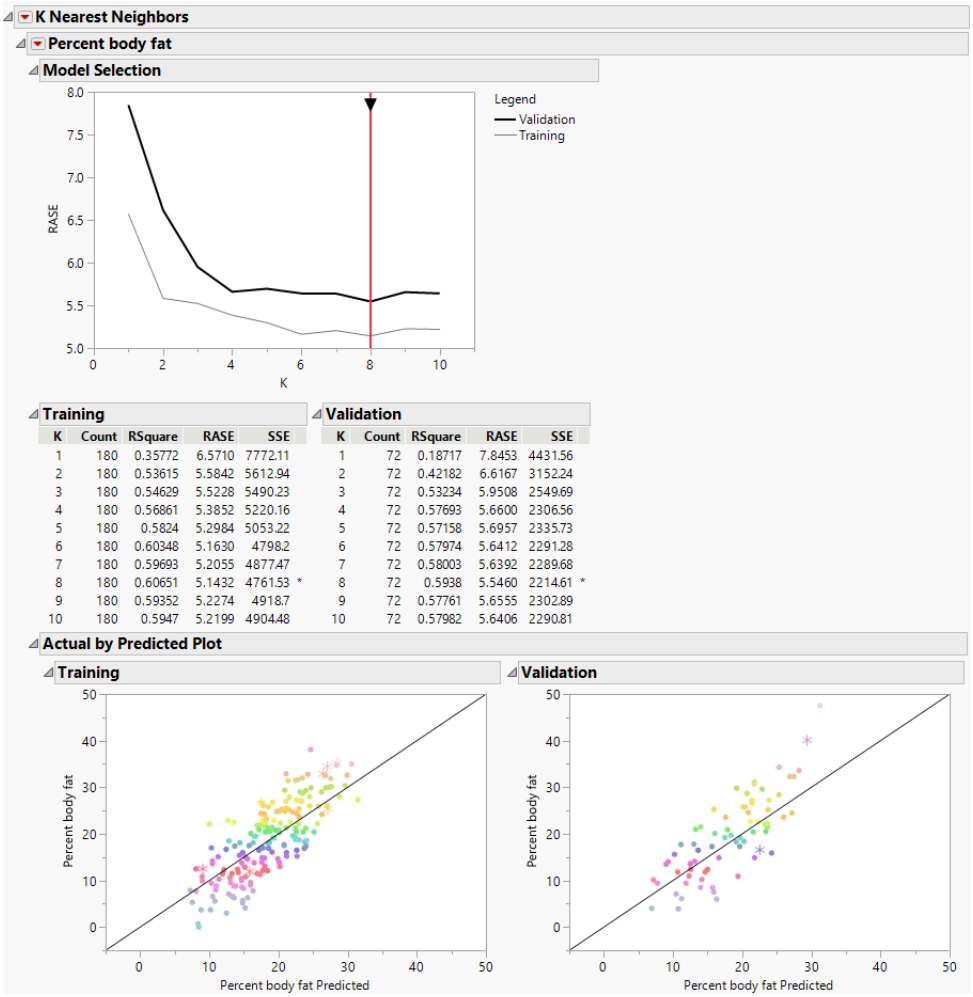
   The prediction equation is saved in the Formula Depot. You can compare the performance of alternative models published to the Formula Depot with that of the $K = 1$ nearest neighbor model using the Model Comparison option in the Formula Depot. See "Formula Depot".

JMP PRO Example of K Nearest Neighbors with Continuous Response

In this example, you want to predict the percent body fat for males using 13 predictors. The Body Fat.jmp sample data table contains percent body fat estimates that are based on underwater weighing and on various body circumference measurements.

1. Select **Help > Sample Data Library** and open Body Fat.jmp.
2. Select **Analyze > Predictive Modeling > K Nearest Neighbors**.
3. Select Percent body fat and click **Y, Response**.
4. Select Age (years) through Wrist circumference (cm) and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.
7. Click the Percent body fat red triangle and select **Plot Actual by Predicted**.

Figure 7.3 K Nearest Neighbors Report

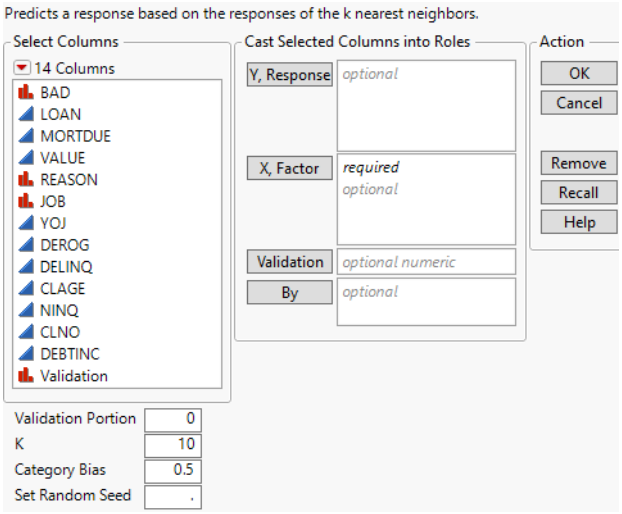


The $K = 8$ model had the lowest RASE for the validation set. Among k nearest neighbor models, the model based on 8 nearest neighbors seems to perform the best. The Actual by Predicted plot for the training set shows that the points fall along the line, signifying that the predicted values are similar to the actual values. Most of the points on the plot for the validation set fall along the line, with a few observations on the upper end that are a bit farther away.

JMP PRO Launch the K Nearest Neighbors Platform

Launch the K Nearest Neighbors platform by selecting **Analyze > Predictive Modeling > K Nearest Neighbors**.

Figure 7.4 K Nearest Neighbors Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The K Nearest Neighbors launch window provides the following options:

Y, Response The response variable or variables that you want to analyze.

> **Note:** The K Nearest Neighbors platform can be used as a utility to determine the distances between neighboring observations, even without the presence of a response variable. If you do not specify a response variable, a blank report appears. However, the red triangle menu options Save Near Neighbor Rows and Save Near Neighbor Distances are available.

X, Factor The predictor variables.

Validation A numeric column that defines the validation sets. This column should contain at most three distinct values:

– If the validation column has two levels, the smaller value defines the training set and the larger value defines the validation set.

- If the validation column has three levels, the values, in order of increasing size, define the training, validation, and test sets.
- If the validation column has more than three levels, the rows that contain the smallest three values define the validation sets. All other rows are excluded from the analysis.

The K Nearest Neighbors platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Validation Portion The portion of the data to be used as the validation set.

Number of Neighbors, K Maximum number of nearest neighbors to analyze. Models are fit for one nearest neighbor up to the value that you specify for *K*.

Note: The maximum number of neighbors, *K*, must be no larger than one less than the number of rows in the training data table. If you specify a *K* that is larger than the maximum allowable *K*, a warning appears.

Category Bias Specifies a tuning parameter that ensures that the fitted probabilities for categorical responses are always positive. By default, the Category Bias is 0.5.

Set Random Seed Sets the seed for the randomization process used in tie-breaking for nominal and ordinal responses. If you specify a Validation Portion, this option also sets the seed for the rows used for validation. Set Random Seed is useful if you want to reproduce an analysis. If you set a random seed and save the script, the seed is automatically saved in the script.

JMP PRO The K Nearest Neighbors Report

The K Nearest Neighbors report contains a separate report for each response variable. Each response variable report contains information about the fitted model for that response. This information includes a Model Selection report and summary information for each of the *k* models that were fit. The report shows tables for the training set and for the validation and test sets if you defined these using validation.

The Model Selection report displays a solution path plot across *K* based on the Misclassification Rate for categorical responses or the RASE for continuous responses. By default, the slider is placed on the value of *K* that corresponds to the best performing model. You can drag the slider to change the value of *K* in the report.

The statistics reported depend on the modeling type of the response. Each row in the summary tables corresponds to a model defined by *k* nearest neighbors, where *K* ranges from one to the value that you specified as Number of Neighbors, K in the launch window.

JMP PRO Continuous Responses

By default, in addition to the Model Selection graph, the report for a continuous response contains the Summary Table report.

Summary Table

An asterisk marks the model for the value of *K* that has the smallest RASE. The report for a continuous response contains the following columns:

K Number of nearest neighbors used in the model. *K* ranges from 1 to the Number of Neighbors, K that you specified in the launch window.

Count Number of observations.

RSquare The RSquare value for the model.

RASE Root mean average squared prediction error for the model. The model with the smallest RASE is marked with an asterisk. If there are tied RASE values, the model with the smallest *K* is marked with the asterisk.

SSE Sum of squared errors for the model.

JMP PRO Categorical Responses

By default, in addition to the Model Selection graph, the report for a categorical response contains the Summary Table, Confusion Matrix, and Mosaic Plot reports.

Summary Table

An asterisk marks the model for the value of *K* that has the smallest misclassification rate. The report for a categorical response contains the following columns:

K Number of nearest neighbors used in the model. *K* ranges from 1 to the Number of Neighbors, K that you specified in the launch window.

Count Number of observations.

Misclassification Rate Proportion of observations misclassified by the model. This is calculated as Misclassifications divided by Count. The model with the smallest misclassification rate is marked with an asterisk. If there are tied misclassification rates, the model with the smallest *K* is marked with the asterisk.

Misclassifications Number of observations that are incorrectly predicted by the model.

Confusion Matrix

By default, a Confusion Matrix Report is shown for the model with the smallest Misclassification Rate. If there are ties for the smallest misclassification rate, a report is shown for the model with the smallest *K*. The Confusion Matrix Report contains confusion matrices and confusion rates matrices. A confusion matrix is a two-way classification of actual and predicted responses. A confusion rates matrix is equal to the confusion matrix, with the numbers divided by the row totals. If you use validation, confusion matrices and confusion rates matrices for the validation and test sets appear. Use the Confusion Matrix Report and the misclassification rates to evaluate your model.

Tip: If you change the position of the slider in the solution path plot, an additional Confusion Matrix Report is displayed for the chosen value of *K*. Use the additional report to compare an alternative model to the default best model.

Mosaic Plot

By default, a mosaic plot is shown for the model with the smallest Misclassification Rate. If there are ties for the smallest misclassification rate, a mosaic plot is shown for the model with the smallest *K*. A mosaic plot is a stacked bar chart where each segment is proportional to its group's frequency count. For more information about mosaic plots, see *Basic Analysis*. If you use validation, mosaic plots for the validation and test sets are shown.

Tip: If you change the position of the slider in the solution path plot, the mosaic plot updates to display the results for the chosen value of *K*.

JMP PRO K Nearest Neighbors Platform Options

The K Nearest Neighbors red triangle menu contains the following options:

Save Near Neighbor Rows Saves *K* columns to the data table. The columns are named RowNear <k>. For a given row, the k^{th} column contains the row number of its k^{th} nearest neighbor.

Caution: The row numbers in the columns RowNear <k> do not update when you reorder the rows in your data table. If you reorder the rows, the values in those columns are incorrect.

Save Near Neighbor Distances Saves *K* columns to the data table. The columns are named Distance <k>. For a given row, the k^{th} column contains the distance to that row's k^{th} nearest neighbor.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Model Fit Options

Mosaic Plot (Available only for nominal or ordinal responses.) Shows or hides the mosaic plot. See "Mosaic Plot".

Plot Actual by Predicted (Available only for continuous responses.) Plots the actual versus the predicted responses for the model with the smallest RASE. If there are ties for the smallest RASE, the plot is based on the model with the smallest *K*.

Tip: If you change the position of the slider on the solution path plot to a different *K*, the Actual by Predicted plot is updated to reflect the model for the chosen value of *K*.

Plot Residual by Predicted (Available only for continuous responses.) Plots the residuals versus the predicted responses for the model with the smallest RASE. If there are ties for the smallest RASE, the plot is based on the model with the smallest *K*.

Tip: If you change the position of the slider on the solution path plot to a different *K*, the Residual by Predicted plot is updated to reflect the model for the chosen value of *K*.

Save Predicteds Saves *K* predicted value columns to the data table. The columns are named Predicted <Response> <k>. The k^{th} column contains predictions for the model based on the *k* nearest neighbors, where Response is the name of the response column.

Save Prediction Formula Saves a column that contains a prediction formula for a specific *k* nearest neighbor model. Enter a value for *K* when prompted. The prediction formula contains all the training data, so this option might not be practical for large data tables.

Caution: The values obtained from Save Prediction Formula and Save Predicteds do not necessarily match. The values obtained from the Save Prediction Formula option use all of the rows in the training set, including the row for the predicted value. The values from the Save Predicteds option do not use the row for the predicted value in the training set, only all other rows.

Publish Prediction Formula Creates a prediction formula for the specified *k* nearest neighbor model and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot".

Chapter 8

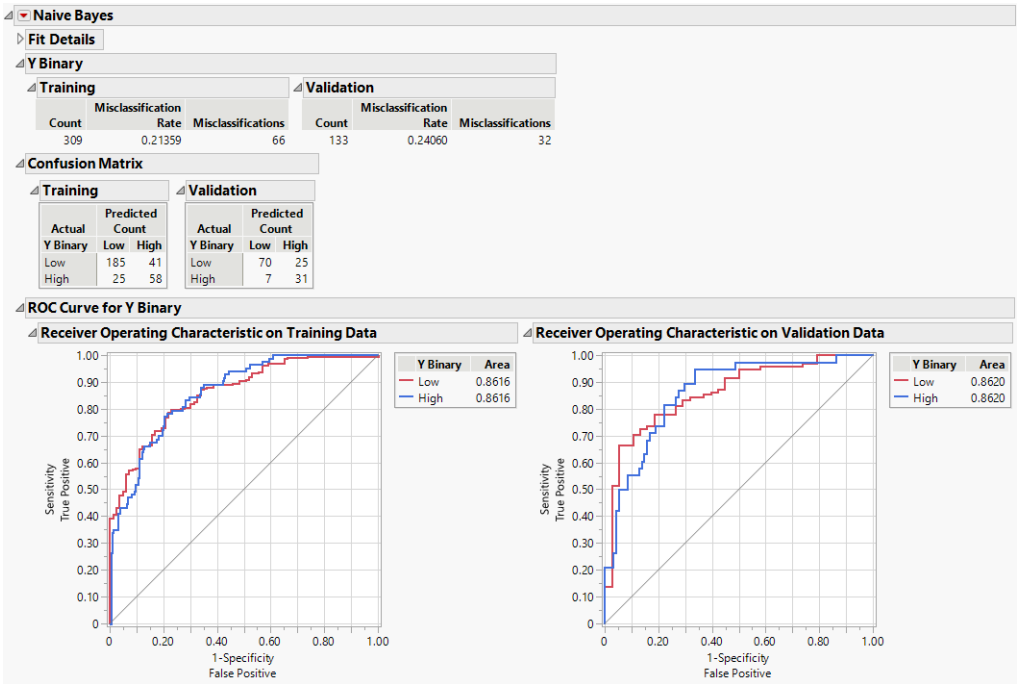
JMP PRO Naive Bayes

Classify Observations Using Probabilistic Assumptions

The Naive Bayes platform is available only in JMP Pro.

The Naive Bayes platform fits a model to predict the value of a categorical variable. Naive Bayes is a fast and computationally simple method. It is especially suitable for situations where there are a large number of predictors.

Figure 8.1 Example of Naive Bayes Analysis



Contents

Overview of the Naive Bayes Platform 141
Example of Naive Bayes 142
Launch the Naive Bayes Platform 144
The Naive Bayes Report 145
Fit Details Report 146
Response Column Report 146
Confusion Matrix Report 146
ROC Curves 146
Naive Bayes Platform Options 147
Additional Example of Naive Bayes 148
Statistical Details for the Naive Bayes Platform 149
Algorithm 149
Saved Probability Formulas 151

JMP PRO Overview of the Naive Bayes Platform

The Naive Bayes platform classifies observations into *classes* that are defined by the levels of a categorical response variable. The variables (or factors) that are used for classification are often called *features* in the data mining literature.

For each class, the naive Bayes algorithm computes the conditional probability of each feature value occurring. If a feature is continuous, its conditional marginal density is estimated. The naive Bayes technique assumes that, within a class, the features are independent. (This is the reason that the technique is referred to as "naive".) Classification is based on the idea that an observation whose feature values have high conditional probabilities within a certain class has a high probability of belonging to that class. See Hastie et al. (2009).

Because the algorithm estimates only one-dimensional densities or distributions, the algorithm is extremely fast. This makes it suitable for large data sets, and in particular, data sets with large numbers of features. All nonmissing feature values for an observation are used in calculating the conditional probabilities.

Each observation is assigned a *naive score* for each class. An observation's naive score for a given class is the proportion of training observations that belong to that class multiplied by the product of the observation's conditional probabilities. The *naive probability* that an observation belongs to a class is its naive score for that class divided by the sum of its naive scores across all classes. The observation is assigned to the class for which it has the highest naive probability.

Caution: Because the conditional probabilities of class membership are assumed independent, the naive Bayes estimated probabilities are inefficient.

Naive Bayes requires a large number of training observations to ensure representation for all predictor values and classes. If an observation in the Validation set is being classified and it has a categorical predictor value that was missing in the Training set, then the platform uses the non-missing features to predict. If an observation is missing all predictor values, the predicted response is the most frequent response. The prediction formulas handle missing values by having them contribute nothing to the observation's score.

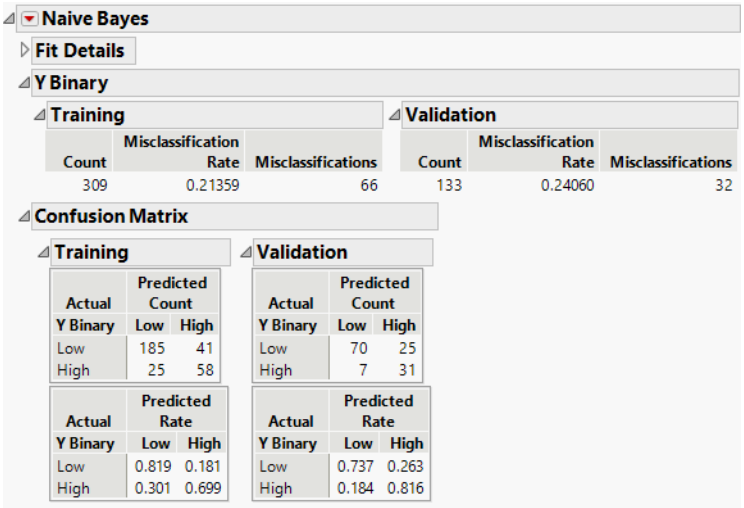
For more information about the naive Bayes technique, see Hand et al. (2001), and Shmueli et al. (2010).

JMP PRO Example of Naive Bayes

You want to construct a classification model to be used in predicting the disease progression for future patients as High or Low. You have baseline medical data for 442 diabetic patients. You also have a binary measure of diabetes disease progression obtained one year after each patient's initial visit. This measure quantifies disease progression as being either Low or High.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Analyze > Predictive Modeling > Naive Bayes**.
3. Select Y Binary and click **Y, Response**.
4. Select Age through Glucose and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.

Figure 8.2 Naive Bayes Report

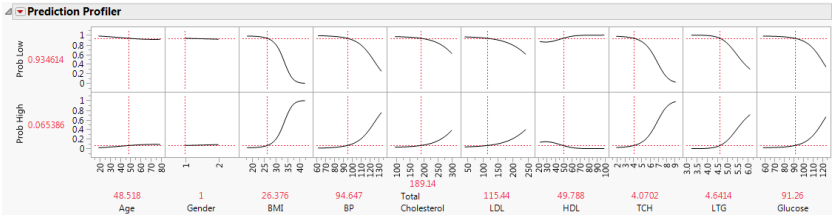


The Training Set has about a 21% misclassification rate and the Validation Set has about a 24% misclassification rate. The Confusion matrix suggests that, for both the Training and Validation sets, the larger source of misclassification comes from classifying patients with Low disease progression as having High disease progression. The Validation set results indicate how your model extends to independent observations.

You are interested in which individual predictors have the greatest impact on the naive Bayes classification.

7. Click the Naive Bayes red triangle and select **Profiler**.

Figure 8.3 Prediction Profiler for Disease Progression



8. Click the Prediction Profiler red triangle and select **Assess Variable Importance > Independent Uniform Inputs**.

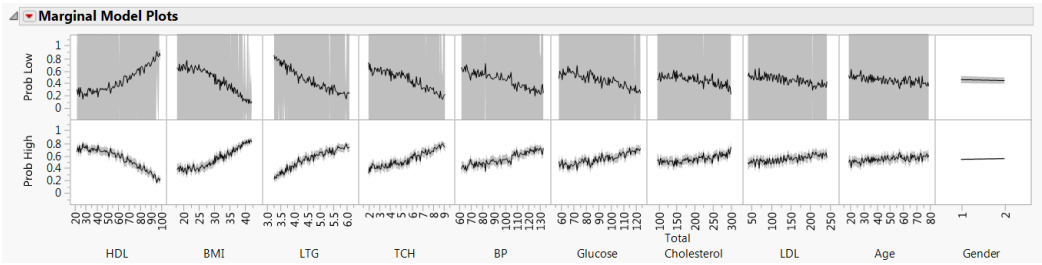
Figure 8.4 Variable Importance

Overall

Column	Main Effect	Total Effect
HDL	0.208	0.362
BMI	0.195	0.341
LTG	0.156	0.304
TCH	0.095	0.217
BP	0.05	0.136
Glucose	0.038	0.094
Total Cholesterol	0.019	0.042
LDL	0.016	0.037
Age	0.007	0.014
Gender	0.001	0.002

The Summary Report indicates that HDL, BMI, and LTG have the greatest impact on the estimated probabilities.

Figure 8.5 Marginal Model Plots Report

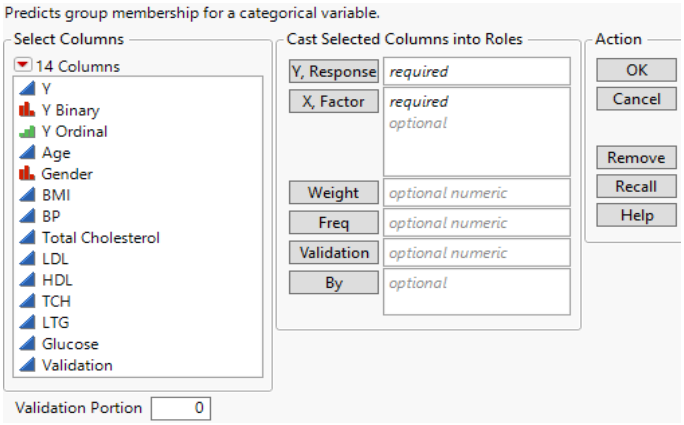


The second row of plots in the Marginal Model Plots report shows that higher values of HDL are associated with a lower probability of classifying a patient as High. Also, higher BMI and LTG values are associated with a higher probability of classifying a patient as High.

JMP PRO Launch the Naive Bayes Platform

Launch the Naive Bayes platform by selecting **Analyze > Predictive Modeling > Naive Bayes**.

Figure 8.6 Naive Bayes Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Naive Bayes launch window provides the following options:

Y, Response The categorical response column whose values are the classes of interest.

X, Factor Categorical or continuous predictor columns.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

Validation A numeric column that defines the validation sets. This column should contain at most three distinct values:

- If the column has three unique values, then:

  rows with the smallest value are used for the Training set.

  rows with the middle value are used for the Validation set.

  rows with the largest value are used for the Test set.
- If the column has two unique values, then only Training and Validation sets are used.
- If the validation column has more than three levels, the rows that contain the smallest three values define the validation sets. All other rows are excluded from the analysis.

The Naive Bayes platform uses the validation column to train and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Validation Portion The portion of the data to be used as the Validation set.

Note: If neither a Validation column or a Validation Portion is specified in the launch window and if there are excluded rows, these rows are treated as a Validation set. For more information about validation see "Validation in JMP Modeling".

JMP PRO The Naive Bayes Report

After you click **OK** in the launch window, the Naive Bayes report appears. By default, the Naive Bayes report contains reports for fit details, the response column, the Confusion Matrix Report, and the ROC and Lift curves.

Figure 8.7 Naive Bayes Report

Naive Bayes

Fit Details

Y Binary

Training

Count	Misclassification Rate	Misclassifications
309	0.21359	66

Validation

Count	Misclassification Rate	Misclassifications
133	0.24060	32

Confusion Matrix

Training

Actual Y Binary	Predicted Count Low	Predicted Count High
Low	185	41
High	25	58

Actual Y Binary	Predicted Rate Low	Predicted Rate High
Low	0.819	0.181
High	0.301	0.699

Validation

Actual Y Binary	Predicted Count Low	Predicted Count High
Low	70	25
High	7	31

Actual Y Binary	Predicted Rate Low	Predicted Rate High
Low	0.737	0.263
High	0.184	0.816

JMP PRO Fit Details Report

The Fit Details report shows various measures of fit for the model for the Training set, and for the Validation and Test sets if they are specified. The Measure column lists the different fit statistics and the Definition column shows the formulas for the corresponding fit statistics. See "Measures of Fit for Categorical Responses". By default, the Fit Details report in the Naive Bayes report window is closed.

JMP PRO Response Column Report

The response column report shows performance statistics for the naive Bayes classification in a summary table for the Training set, and the Validation and Test sets if they are specified. The summary tables contain the following columns:

Count The number of observations in the set corresponding to the table (Training, Validation, or Test set).

Misclassification Rate Proportion of observations in the corresponding set that are misclassified by the model. This is calculated as Misclassifications divided by Count.

Misclassifications The number of observations in the corresponding set that are classified incorrectly.

JMP PRO Confusion Matrix Report

The Confusion Matrix Report shows a confusion matrix and a confusion rates matrix for the Training set, and for the Validation and Test sets if they are specified. A confusion matrix is a two-way classification of actual and predicted responses. A confusion rates matrix is equal to the confusion matrix, with the numbers divided by the row totals.

JMP PRO ROC Curves

The report displays the Receiver Operating Characteristic (ROC) curve for the Training set, and for the Validation and Test sets if they are specified. The ROC curve measures the ability of the fitted probabilities to classify response levels correctly. The further the curve from the diagonal, the better the fit. An introduction to ROC curves is found in *Basic Analysis*.

See "ROC Curve" for more information about ROC curves.

JMP PRO Naive Bayes Platform Options

The Naive Bayes red triangle menu contains the following options:

ROC Curve Shows or hides the ROC plot. The ROC plot is shown by default. See "ROC Curves".

Lift Curve Shows or hides the lift curve for the model. If you used validation, Lift curve is shown for each of the Training, Validation, and Test sets, if specified.

A lift curve shows how effectively response levels are classified as their fitted probabilities decrease. The fitted probabilities are plotted along the horizontal axis in descending order. The vertical coordinate for a fitted probability is the proportion of correct classifications for that probability or higher, divided by the overall correct classification rate. Use the lift curve to see whether you can correctly classify a large proportion of observations if you select only those with a fitted probability that exceeds a threshold value.

If the response has two levels, the Lift curve plot displays a lift curve for the first level of the response only. If the response has more than two levels, the Lift curve plot displays a sub-outline of the curves for each response level. See "Lift Curve" for more information about lift curves.

Save Predicteds Saves the predicted classifications to the data table in a column called Naive Predicted <Y, Response>.

Save Prediction Formula Saves a column called Naive Predicted Formula <Y, Response> to the data table. This column contains the prediction formula for the classifications.

Save Probability Formula Saves columns to the data table that contain formulas used for classifying each observation. Three groups of columns are saved:

Naive Score <Class>, Sum For each column that represents a class, this column gives a score formula that measures strength of membership in the given class. In the Naive Score Sum column., these scores are summed across classes. See "Saved Probability Formulas".

Naive Prob <Class> For each class, this column gives a formula for the conditional probability that an observation is in that class. See "Saved Probability Formulas".

Naive Predicted Formula <Y, Response> Gives the formula for the predicted class.

Publish Probability Formula Creates probability formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option opens the Formula Depot window. See "Formula Depot".

Profiler Shows or hides an interactive profiler report. Changes in the factor values are reflected in the estimated classification probabilities. See *Profilers*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

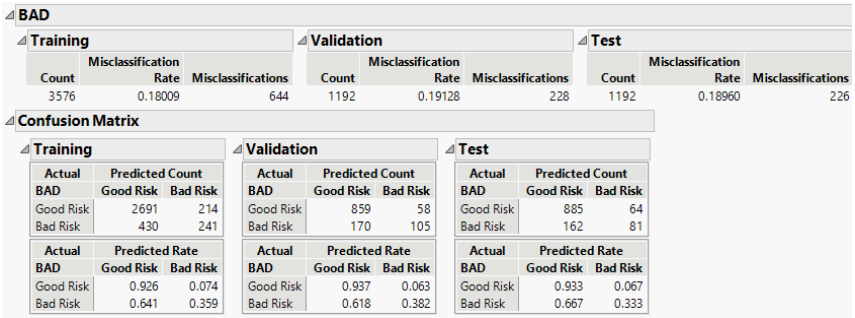
JMP PRO Additional Example of Naive Bayes

You have historical financial data for 5,960 customers who applied for home equity loans. Each customer was classified as being a Good Risk or Bad Risk. There is missing data on most of the predictors. You want to construct a model to use in classifying the credit risk of future customers.

1. Select **Help > Sample Data Library** and open Equity.jmp.
2. Select **Analyze > Predictive Modeling > Naive Bayes**.
3. Select BAD and click **Y, Response**.

   One of the potential predictors, DEBTINC, has many missing values that might be informative. However, naive Bayes is not prepared to handle large number of missing values well, so you do not include DEBTINC in your model.
4. Select LOAN through CLNO and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.

Figure 8.8 Naive Bayes Report for BAD



BAD

Training

Count	Misclassification Rate	Misclassifications
3576	0.18009	644

Validation

Count	Misclassification Rate	Misclassifications
1192	0.19128	228

Test

Count	Misclassification Rate	Misclassifications
1192	0.18960	226

Confusion Matrix

Training

Actual BAD	Predicted Count Good Risk	Bad Risk
Good Risk	2691	214
Bad Risk	430	241

Actual BAD	Predicted Rate Good Risk	Bad Risk
Good Risk	0.926	0.074
Bad Risk	0.641	0.359

Validation

Actual BAD	Predicted Count Good Risk	Bad Risk
Good Risk	859	58
Bad Risk	170	105

Actual BAD	Predicted Rate Good Risk	Bad Risk
Good Risk	0.937	0.063
Bad Risk	0.618	0.382

Test

Actual BAD	Predicted Count Good Risk	Bad Risk
Good Risk	885	64
Bad Risk	162	81

Actual BAD	Predicted Rate Good Risk	Bad Risk
Good Risk	0.933	0.067
Bad Risk	0.667	0.333

The Training, Validation, and Test sets show misclassification rates between 18% and 19%. The confusion matrices for all of the sets suggest that the largest source of misclassification is the classification of Bad Risk customers as Good Risk customers.

You are interested in the probabilities that customers with certain financial background values are classified as High Risk.

7. Click the Naive Bayes red triangle and select **Save Probability Formulas**.

   Three sets of columns are added to the data table.

   – The three Naive Score columns contain naive score formulas for Good Risk, Bad Risk, and the sum of both.

   – The two Naive Prob columns contain probability formulas for Good Risk and Bad Risk.

   – The Naive Predicted Formula Bad column contains a formula that assigns an observation to the class for which the observation has the highest naive probability.

   Use these formulas to score new customers. For more information about the formula columns, see "Saved Probability Formulas".

JMP PRO Statistical Details for the Naive Bayes Platform

- "Algorithm"
- "Saved Probability Formulas"

JMP PRO Algorithm

The naive Bayes method classifies an observation into the class for which its probability of membership, given the values of its features, is highest. The method assumes that the features are conditionally independent within each class.

Denote the possible classifications by $C_1, \ldots, C_K$. Denote the features, or predictors, by $X_1, X_2, \ldots, X_p$.

The conditional probability that an observation with predictor values $x_1, x_2, \ldots, x_p$ belongs in the class C_k is computed as follows:

$$P(C_k|(x_1, \ldots, x_p)) = \frac{\left(P(C_k)\prod_{j=1}^{p}[P(x_j|C_k)]\right)/(\mathbf{R})}{\sum_{k=1}^{K}\left(P(C_k)\left(\prod_{j=1}^{p}[P(x_j|C_k)]\right)/\mathbf{R}\right)},$$

where **R** is a regularization constant. In the formula above, the conditional probability that an observation with $X_j = x_j$ belongs to the class C_k, $P(x_j|C_k)$, is determined as follows:

- If X_j is categorical:

$$P(x_j|C_k) = \frac{\#\text{ of observations in } x_j|C_k}{\#\text{ of observations in } C_k}$$

- If X_j is continuous:

$$P(x_j|C_k) = \frac{1}{s_{jk}}\phi((x_j - m_{jk})/s_{jk})$$

  Here, ϕ is the standard normal density function, and m and s are the mean and standard deviation, respectively, of the predictor values within the class C_k.

The unconditional probability that an observation belongs in class C_k, $P(C_k)$, is computed as follows:

$$P(C_k) = \frac{\#\text{ of observations in } C_k + (0.5/K)}{\text{Total \# of observations} + 0.5}$$

An observation is classified into the class for which its conditional probability is the largest.

Note: In the formula for $P(C_k)$, 0.5 is the prior bias factor. This value is the default value. To change the prior default factor, go to File > Preferences > Platforms > Naive Bayes, select the Prior Bias check box and change the value.

JMP PRO Saved Probability Formulas

This section describes the formulas saved using the Save Probability Formula option. The conditional probability that an observation with predictor values x_1, x_2, …, x_p belongs in the class C_k differs slightly from that given by $P(C_k|(x_1,..., x_p))$, shown in the section "Algorithm". This is done for computational efficiency.

Naive Score Formulas

The Naive Score formula for a given class C_k, $S(C_k)$, is a variation of the numerator in the expression for $P(C_k|(x_1,..., x_p))$ and is computed as follows:

$$S(C_k) = \exp[\ln\{P(C_k)\} + Continuous + Categorical + \ln(\mathbf{R})]$$

The Naive Score formula is a combination of scores from continuous and categorical predictors. Recall that **R** is a regularization constant. The continuous portion of the formula is computed as follows:

$$Continuous = \sum_{j=1}^{p_1} \left\{ \text{Normal Log Density}\left(\frac{x_j - m_{jk}}{s_{jk}} \right) + \ln(1/s_{jk}) \right\}$$

where

$j = 1,..., p_1$ continuous predictors.

The categorical portion of the formula is computed as follows:

$$Categorical = \sum_{r=1}^{p_2} \left\{ \sum_{l=1}^{L_r} 1_{rl} \ln(P(x_{rl}|C_k)) \right\}$$

where

$r = 1,...,p_2$ categorical variables

$l = 1,..., L_r$ levels of the r^{th} categorical variable

1_{rl} is an indicator variable that equals 1 when x_{rl} is the l^{th} level of the r^{th} categorical predictor and 0 otherwise

Naive Score Sum Formulas

The Naive Score Sum formula, S, sums the Naive Score formulas over all classes. This is a variation of the denominator in the expression for $P(C_k|(x_1,..., x_p))$.

$$S = \sum_{k=1}^{K} S(C_k)$$

Naive Prob Formulas

The Naive Prob formula for a given class C_k equals $P(C_k|(x_1,..., x_p))$. In the JMP formulas,

$$P(C_k|(x_1, \ldots, x_p)) = \frac{S(C_k)}{S}$$

Naive Predicted Formula

The Naive Predicted Formula for an observation classifies that observation into the class for which $P(C_k|(x_1,..., x_p))$ is the largest. This is equivalent to classifying an observation into the class for which its Naive Score formula is the largest.

Chapter 9

JMP PRO Support Vector Machines

Predict Response Categories Using Support Vectors

The Support Vector Machines platform is available only in JMP Pro.

In JMP Pro, the Support Vector Machines (SVM) platform provides a machine learning tool for both nonlinear regression and classification. The SVM algorithm is a supervised learning algorithm that uses the training data to build a model to predict or classify new observations.

If you have a categorical response, you can visualize the model classification in the Response Profile Plot. The SVM platform also provides misclassification rates, confusion matrices, and ROC and Lift curves to help you determine how well your model fits the data. If you have a continuous response, fit statistics and actual by predicted plots are used to determine how well the model fits the data.

Figure 9.1 Response Profile Plot

Contents

Overview of Support Vector Machines 155
Example of Support Vector Machines 155
Launch the Support Vector Machines Platform 157
The Support Vector Machines Launch Window 158
Model Launch Control Panel 159
The Support Vector Machine Report 162
Model Comparison Report 162
Support Vector Machine Model Report 163
Support Vector Machines Platform Options 165
Support Vector Machine Model Report Options 166
Additional Example of the SVM Platform 168
Example of Support Vector Regression for Continuous Response 168

JMP PRO Overview of Support Vector Machines

A support vector machine (SVM) model is a supervised learning algorithm that is used to predict or classify new observations. A model is fit on a set of training data where the responses are known. Then, the model is used to predict the responses of new observations.

When the response is categorical, SVM models classify data by optimizing a hyperplane that separates the classes. This can also be viewed as finding the hyperplane that maximizes the margin between the classes. In simple problems, this hyperplane is linear. However, more complicated data often cannot be separated linearly. For these scenarios, the SVM platform provides the option to use a radial basis function kernel to map the points to a nonlinear dimension that can make the classes easier to separate.

When the response is continuous, the models that are fit are known as support vector regression (SVR) models. In a typical regression problem, the goal is to fit a model that minimizes the error between a predicted response and the actual response. In an SVR problem, the goal is to fit a model such that the error between a predicted response and the actual response falls within a range of $-\varepsilon$ to ε. This provides a more flexible fit. In JMP Pro, ε is equal to 0.1. The SVR algorithm doubles the data by creating two classes, $Y + \varepsilon$ and $Y - \varepsilon$. Then the same algorithm that is used for the classification problem is also used for the prediction (SVR) problem.

The maximization in SVM algorithms is performed by solving a quadratic programming problem. In JMP Pro, the algorithm used by the SVM platform is based on the Sequential Minimal Optimization (SMO) algorithm introduced by John Platt in 1998. Typically, the SVM quadratic programming problem is very large. The SMO algorithm splits the overall quadratic programming problem into a series of smaller quadratic programming problems. Smaller quadratic programming problems are solved analytically instead of numerically, meaning they produce closed-form solutions. Therefore, the SMO algorithm is more efficient than solving the overall quadratic programming problem (Platt 1998).

JMP PRO Example of Support Vector Machines

You have baseline medical data for 442 diabetic patients. You also have a binary measure of diabetes disease progression obtained one year after each patient's initial visit. This measure quantifies disease progression as either Low or High. You want to construct a classification model to predict the disease progression for future patients as High or Low. You explore both kernel function options.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Analyze > Predictive Modeling > Support Vector Machines**.

3. Select Y Binary and click **Y, Response**.
4. Select Age through Glucose and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.
7. In the Model Launch control panel, check that the kernel function is a Radial Basis Function and select Tuning Design.
8. Enter 10 next to Number of Runs.
9. Click **Go**.
10. Click the gray triangle next to Model Launch to open the Model Launch control panel.
11. Change the kernel function to a Linear function and select Tuning Design.
12. Enter 10 next to Number of Runs.
13. Click **Go**.

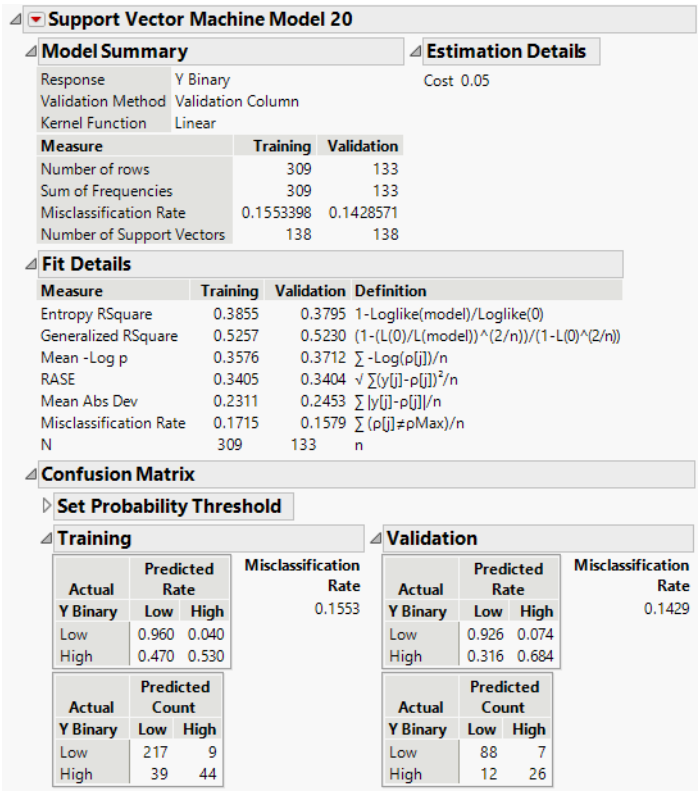
Figure 9.2 Model Comparison Report

Model Comparison

Show	Method	Kernel Function	Cost	Gamma	# SV	Training Misclassification Rate	Validation Misclassification Rate	Validation Generalized RSquare	Probability Threshold	Threshold Validation Misclassification Rate	Best
☐	Model 1	Radial Basis Function	2.57061	0.43877	251	0.00324	0.25564	0.00838	0.39079	0.25564	
☐	Model 2	Radial Basis Function	3.04271	0.31899	218	0.01294	0.28571	-0.4206	0.23223	0.28571	
☐	Model 3	Radial Basis Function	3.93906	0.46899	255	0.00324	0.28571	-0.0081	0.53372	0.28571	
☐	Model 4	Radial Basis Function	1.073	0.27968	208	0.0712	0.21805	0.22158	0.17372	0.21805	
☐	Model 5	Radial Basis Function	0.41677	0.0281	156	0.15534	0.18797	0.50483	0.36015	0.18797	
☐	Model 6	Radial Basis Function	0.23369	0.39898	246	0.26861	0.28571	0.22323	6e-8	0.28571	
☐	Model 7	Radial Basis Function	4.8013	0.15144	167	0.05178	0.2782	-0.4245	0.26219	0.2782	
☐	Model 8	Radial Basis Function	3.51202	0.20295	181	0.04207	0.28571	-0.454	0.22753	0.28571	
☐	Model 9	Radial Basis Function	4.2693	0.06093	140	0.11974	0.19549	0.42841	0.32812	0.19549	
☐	Model 10	Radial Basis Function	1.87641	0.0919	150	0.13269	0.21053	0.43499	0.30598	0.21053	
☐	Model 11	Linear	4.42735	.	122	0.17476	0.17293	0.51943	0.47722	0.17293	
☐	Model 12	Linear	4.99046	.	123	0.16828	0.17293	0.52037	0.47551	0.17293	
☐	Model 13	Linear	3.8272	.	123	0.17476	0.17293	0.51923	0.47763	0.17293	
☐	Model 14	Linear	2.74788	.	124	0.17152	0.17293	0.51724	0.48019	0.17293	
☐	Model 15	Linear	3.22207	.	123	0.17476	0.17293	0.51851	0.47842	0.17293	
☐	Model 16	Linear	1.29304	.	124	0.17152	0.17293	0.51518	0.4819	0.17293	
☐	Model 17	Linear	1.74671	.	124	0.16828	0.17293	0.51557	0.48143	0.17293	
☐	Model 18	Linear	0.83575	.	123	0.16505	0.15789	0.51498	0.48207	0.15789	
☐	Model 19	Linear	2.22	.	125	0.17152	0.17293	0.51579	0.48255	0.17293	
☑	Model 20	Linear	0.04975	.	138	0.15534	0.14286	0.52302	0.41946	0.14286	Smallest ...

The Model Comparison report shows that the best model in terms of misclassification rate and RSquare is Model 20. This model has a linear kernel function with cost parameter 0.04975. This is the model to further analyze.

Figure 9.3 Model Report for Best Fitting Model



Support Vector Machine Model 20

Model Summary

Response	Y Binary
Validation Method	Validation Column
Kernel Function	Linear

Measure	Training	Validation
Number of rows	309	133
Sum of Frequencies	309	133
Misclassification Rate	0.1553398	0.1428571
Number of Support Vectors	138	138

Estimation Details

Cost 0.05

Fit Details

Measure	Training	Validation	Definition
Entropy RSquare	0.3855	0.3795	1-Loglike(model)/Loglike(0)
Generalized RSquare	0.5257	0.5230	(1-(L(0)/L(model))^(2/n))/(1-L(0)^(2/n))
Mean -Log p	0.3576	0.3712	∑ -Log(ρ[j])/n
RASE	0.3405	0.3404	√ ∑(y[j]-ρ[j])²/n
Mean Abs Dev	0.2311	0.2453	∑ |y[j]-ρ[j]|/n
Misclassification Rate	0.1715	0.1579	∑ (ρ[j]≠pMax)/n
N	309	133	n

Confusion Matrix

Set Probability Threshold

Training

Actual Y Binary	Predicted Rate Low	Predicted Rate High	Misclassification Rate
Low	0.960	0.040	0.1553
High	0.470	0.530	

Actual Y Binary	Predicted Count Low	Predicted Count High
Low	217	9
High	39	44

Validation

Actual Y Binary	Predicted Rate Low	Predicted Rate High	Misclassification Rate
Low	0.926	0.074	0.1429
High	0.316	0.684	

Actual Y Binary	Predicted Count Low	Predicted Count High
Low	88	7
High	12	26

The Model Summary report shows that the misclassification rates for the training and validation sets are very similar. This is a good indication that the model did not overfit the data. The confusion matrices provide more information about the types of observations that are misclassified by the model. In the confusion matrices, the upper left corner shows that the model correctly categorizes the Low responses most of the time (96% in training and 92.6% in validation). However, fewer of the High responses are correctly categorized (53% in training and 68.4% in validation). Therefore, most of the misclassifications are the High responses being misclassified as Low.

JMP PRO Launch the Support Vector Machines Platform

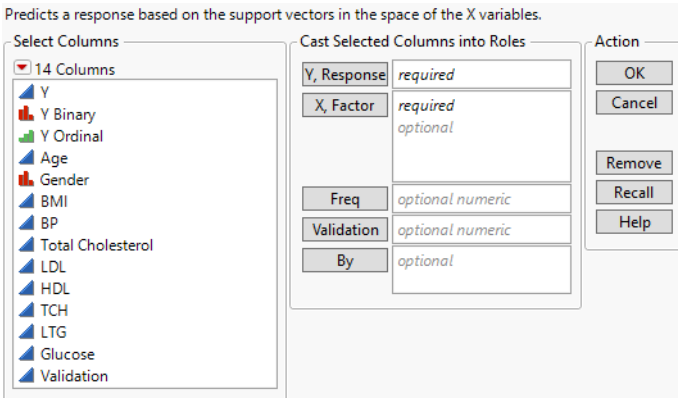
Launch the Support Vector Machines (SVM) platform by selecting **Analyze > Predictive Modeling > Support Vector Machines**.

Launching the SVM platform is a two-step process. First, enter your variables on the SVM launch window. Second, specify your options in the Model Launch control panel.

JMP PRO The Support Vector Machines Launch Window

Use the SVM launch window to specify your response and model factors. If used, validation and by columns are also specified in the launch windows.

Figure 9.4 Support Vector Machines Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Support Vector Machines launch window provides the following options:

Y, Response The response variable that you want to analyze.

X, Factor The predictor variables.

Freq A column whose numeric values assign a frequency to each row in the analysis.

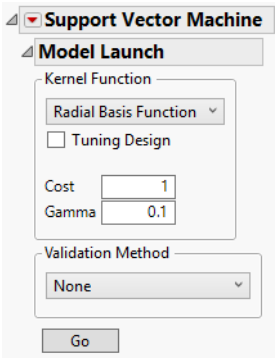
Validation A numeric column that defines the validation sets. See "Validation Method". If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

JMP PRO Model Launch Control Panel

When you click OK in the launch window, the Support Vector Machine report window appears, showing a Model Launch control panel for fitting models. Use the Model Launch control panel to specify the kernel function and associated parameter values, as well as the validation method.

Figure 9.5 The Model Launch Control Panel



The Model Launch control panel contains the following options:

Kernel Function Specifies the kernel function used in the model. Choose from the following kernel functions:

Radial Basis Function Selects the radial basis function kernel to create a nonlinear hyperplane to separate the classes.

– The Cost parameter is the penalty associated with misclassifying an observation in the training set. A higher cost parameter implements an algorithm that is less likely to misclassify a point in the training set, whereas a lower cost parameter produces a wider margin. The Cost parameter must be greater than 0, and the default value is 1.

– The Gamma parameter is the parameter in the kernel function. This parameter determines the amount of curvature there is to the decision line; a higher Gamma value indicates more curvature. A nonlinear decision line provides a more flexible fit, but too much curvature can lead to overfitting. The Gamma parameter must be greater than 0, and the default value is 1/(# of predictors).

Linear Selects the linear kernel function to create a linear hyperplane to separate the classes.

– The Cost parameter is the penalty associated with misclassifying an observation in the training set. A higher cost parameter implements an algorithm that is less likely to misclassify a point in the training set, whereas a lower cost parameter produces a wider margin. The Cost parameter must be greater than 0, and the default value is 1.

Note: If you specify parameter values that are out of range, the default values are used.

Tip: To find the best fitting model, fit a range of kernel functions and parameter values and use the Model Comparison report.

Tuning Design Enables you to fit a range of parameter values for the specified kernel. The models with the largest RSquare and the smallest Misclassification Rate or RASE are identified in the Model Comparison report. After you select Tuning Design, you must specify minimum and maximum values for the parameters. Default values are provided based on the data and the minimum must be greater than zero. You must also specify a value for the Number of Runs. The SVM platform fits that many models over a grid of parameter values determined by the minimum and maximum values.

Validation Method Specifies the model validation method. When you click the Go button for the first time, the first SVM model is fit using the specified validation method. This Validation Method is then used for all SVM models fit from within the SVM window. This ensures that all models in the report window are fit using the same validation method and validation set.

Holdback Randomly divides the original data into training and validation sets. You can specify the proportion of the original data to use as the validation set (holdback).

KFold (Available only when *Y* is continuous or nominal.) Randomly divides the original data into *K* subsets. In turn, each of the *K* sets is used to validate the model fit on the rest of the data, fitting a total of *K* models. If *Y* is continuous, the model that has the best validation RASE statistic is chosen as the final model. If *Y* is nominal, the model that has the best validation misclassification rate is chosen as the final model.

Validation Column (Available only if you specified a Validation column in the launch window.) Uses the values in the specified Validation column to divide the data into parts. The column's values determine how the data are split, and what method is used for validation:

- If there are two values, the smaller value defines the training set and the larger value defines the validation set.
- If there are three values, these values define the training, validation, and test sets in order of increasing size.
- If the validation column has more than three levels, then Validation Column K Fold is used.

The SVM platforms uses the validation column to train and evaluate the model, unless a Tuning Design is used. If the Tuning Design option is selected, the SVM platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

Note: If the validation column does not lead to a valid partition of the data, the Holdback validation method is used instead.

Validation Column K Fold (Available only when the Y, Response column has exactly two levels and a Validation column is specified in the launch window.) Uses the values in the specified Validation column to divide the data into *K* sets, where *K* is the number of unique values in the column. Then, K-Fold validation is performed.

None No validation used.

Go Fits the specified SVM model and shows the model report.

Note: If you have a large data table, a progress bar is shown for each model that is fit to the data. The total number of models fit is $k!/2(k-2)!$, where k is the number of levels of the response variable. Each progress bar has an **Accept Current Estimates** button. Click this button if you want to stop the fitting algorithm early and accept the current estimates. Because prediction calculations are performed after you click this button, it may take some time for the report to appear.

Missing Values

Rows that contain any missing predictor values are not included in the SVM modeling procedure. Therefore, any columns saved to the data table will contain missing values in those rows. If you want to include data with missing values in an SVM model, some form of preprocessing is required. See "Explore Missing Values Utility".

JMP PRO The Support Vector Machine Report

When you click **Go** in the Model Launch control panel, a Model Comparison report appears above the control panel and a model report appears below the control panel. The model reports are numbered sequentially. Each time a new model is fit, it is added to the Model Comparison report and a new model report is added to the window. See "Model Comparison Report" and "Support Vector Machine Model Report". Click on the column headings to sort the models by any of the columns in the Model Comparison report. The first click sorts in ascending order; click the column heading a second time to sort in descending order.

JMP PRO Model Comparison Report

The Model Comparison Report contains the following information:

Show A check box that indicates whether the Support Vector Machine Model report for the corresponding fit should appear in the report window.

Method The model number.

Kernel Function The kernel function for the corresponding model.

Cost The value of the cost parameter for the corresponding model.

Gamma The value of the gamma parameter for the corresponding model. If the model uses a linear kernel, this value is missing.

 # SV The number of support vectors used in the corresponding model.

Training Misclassification Rate (Appears only if the response is categorical.) The misclassification rate for the observations in the training set. This rate is based on the classification decision rule that is calculated by the platform.

Validation Misclassification Rate (Appears only if the response is categorical and validation is used.) The misclassification rate for the observations in the validation set. This rate is based on the classification decision rule that is calculated by the platform.

Test Misclassification Rate (Appears only if the response is categorical and a test set is specified using a validation column.) The misclassification rate for the observations in the test set. This rate is based on the classification decision rule that is calculated by the platform.

Training RASE (Appears only if the response is continuous.) The square root of the mean squared prediction error (Root Average Square Error) for the observations in the training set. See "RASE".

Validation RASE (Appears only if the response is continuous and validation is used.) The square root of the mean squared prediction error (Root Average Square Error) for the observations in the validation set. See "RASE".

Test RASE (Appears only if the response is continuous and a test set is specified using a validation column.) The square root of the mean squared prediction error (Root Average Square Error) for the observations in the test set. See "RASE".

Validation Generalized RSquare (Appears only if validation is used.) The Generalized RSquare value for observations in the validation set. See "Fit Details".

Probability Threshold (Appears only if the response is binary and validation is used.) The cutoff probability determined by the classification decision rule that is calculated by the platform. An observation is classified into the target level when its predicted probability exceeds this value.

Tip: You can change this value by using the Set Probability Threshold option in the Confusion Matrix report. See "Confusion Matrix".

Conditional Validation Misclassification Rate (Appears only if the response is binary and validation is used.) The misclassification rate for the observations in the validation set, conditioned on the probability threshold value.

Best Indicates which model fit has the smallest misclassification rate. If validation is used without a test set, this is the model with the smallest validation misclassification rate. If validation is used with a test set, this is the model with the smallest test misclassification rate.

Model Comparison Plot

Shows a contour plot of the model performance of the validation set over the grid of parameter values. This report is available only if Tuning Design is specified in the Model Launch Control Panel.

JMP PRO Support Vector Machine Model Report

Response Profile Plot (Available only when the response is categorical.) Gives a visual representation of the classification model. The points on the plot are the actual data observations and are on by default only when there are exactly two continuous variables. For the two variables plotted, the shaded contours represent a plane of the prediction space determined by fixed values of the remaining model factors. The predictions are based on the classification decision rule that is calculated by the platform. Controls for the fixed values are located above the plot. When you change the fixed values of the factors, using either the slider or number box, the prediction space for the plotted variables is

automatically updated. You can also change the axes of the plot to display any continuous factor using the red triangles on each axis.

The Response Profile Plot red triangle menu contains the following options:

Grid Density Determines the fineness of the prediction grid underlying the shaded contours. A higher grid density provides a smoother decision line.

Show Points Shows or hides the points on the plot. On by default when there are only two variables.

Model Summary Gives the name of the response column, the validation method, and the type of kernel function used in the model fit. The Model Summary table also contains information about the model fit for the training, validation, and test sets. The number of observations and the number of support vectors are reported for each set. If the response is categorical, the misclassification rate is reported for each set. The misclassification rate is the proportion of observations misclassified by the model. This is calculated as the number of misclassifications divided by the total number of observations. If the response is continuous, the RASE and R Square values are reported for each set.

Estimation Details Gives the values of the parameters used in the model.

Fit Details (Available only if the response is categorical.) Gives the following statistics for the training set, and for the validation and test sets if they are specified:

Entropy RSquare A measure of fit that compares the log-likelihoods from the fitted model and the constant probability model. Entropy RSquare ranges from 0 to 1, where values closer to 1 indicate a better fit. See "Entropy RSquare".

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991). Values closer to 1 indicate a better fit.

Mean -Log p The average of -log(p), where p is the fitted probability associated with the event that occurred. Smaller values indicate a better fit.

RASE The square root of the mean squared prediction error (Root Average Square Error). RASE is computed as follows, where *Source* indicates the Training, Validation, or Test set.

$$\text{RASE}_{Source} = \sqrt{\frac{SSE_{Source}}{n}}$$

Mean Abs Dev The average of the absolute values of the differences between the response and the predicted response. The differences are between 1 and p, the fitted probability for the response level that actually occurred. Smaller values indicate a better fit.

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category.

Note: The misclassification rates in the Fit Details report might not match the misclassification rates in the Confusion Matrix report. When the response is binary, the rates in the Fit Details report use a probability cutoff of 0.5, but the rates in the Confusion Matrix report use the probability threshold value as the cutoff.

N The number of observations.

Confusion Matrix (Available only if the response is categorical.) A confusion matrix is shown for each set (training, validation, and test) specified in the model. A confusion matrix is a two-way classification of actual and predicted responses. Use the confusion matrices and the misclassification rates to evaluate your model.

The confusion matrices and the misclassification rates use the value in the probability threshold box as the cutoff value. By default, this value is based on the classification decision rule that is calculated by the platform. You can change the cutoff value by dragging the slider or entering a new value in the box next to Probability Threshold. If you change the probability threshold, the confusion matrices and misclassification rates automatically update. The probability threshold and conditional validation misclassification rate columns in the Model Comparison report are also updated.

Actual by Predicted Plot (Available only if the response is continuous.) Plots the actual response versus the predicted response. The diagonal line is the Y = X line. The closer the points are to this line, the better the model fits the data. When validation is used, plots are shown for the training, validation, and, if specified, test sets.

JMP PRO Support Vector Machines Platform Options

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Support Vector Machine Model Report Options

Each Support Vector Machine Model report contains the following options:

Response Profile Plot (Available only if the response is categorical.) Shows or hides the Response Profile Plot.

Support Vector Coefficients Shows or hides the table of support vector coefficients. Select rows in the table to highlight the support vectors in the original data table and the Response Profile Plot.

Confusion Matrix (Available only if the response is categorical.) Shows or hides the confusion matrix. See "Confusion Matrix".

ROC Curve (Available only if the response is categorical.) Shows or hides the ROC Curve. If you use validation, ROC Curves for the validation and, if specified, test sets appear in addition to the curve for the training set. For more information, see "ROC Curve".

Lift Curve (Available only if the response is categorical.) Shows or hides the Lift Curve. If you use validation, Lift Curves for the validation and, if specified, test sets appear in addition to the curve for the training set. For more information, see "Lift Curve".

Plot Actual by Predicted (Available only if the response is continuous.) Shows or hides a plot of the actual values versus the predicted values.

Plot Residual by Predicted (Available only if the response is continuous.) Show or hides a plot of the residuals versus the predicted values.

Profiler Shows or hides the Prediction Profiler. Each response level is represented by a separate row in the Prediction Profiler. For more information about the options in the red triangle menu, see *Profilers*.

Contour Profiler (Available only when the model contains more than one continuous factor.) Shows or hides the Contour Profiler. For more information about the options in the red triangle menu, see *Profilers*.

Surface Profiler (Available only when the model contains at least one continuous factor.) Shows or hides a surface profiler. For more information about the options in the red triangle menu, see *Profilers*.

Save Predicteds Saves the predicted classifications based on the threshold probability cutoff value to the data table in a column called Predicted <Y, Response>.

Save Prediction Formula Saves the prediction formulas based on the threshold probability cutoff value to the data table. If the response is binary, a single prediction formula column is added to the data table. If the response has more than two levels, a prediction formula column is added to the data table for each pairwise model, in addition to a formula column for the final prediction.

Publish Prediction Formula Creates probability formulas based on the threshold probability cutoff value and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option opens the Formula Depot window. See "Formula Depot".

Save Probabilities Saves the probability of each response level as a separate column in the data table, as well as a column called Most Likely <Y, Response>.

Save Probability Formula (Available only if the response is binary.) Saves the following formula columns to the data table:

Decision Value A hidden column that contains the formula for the decision value.

Probability Two probability formula columns, one for each level of the binary response.

Most Likely The predicted classification based on which probability value is higher.

Threshold Predicted The predicted classification based on the threshold probability cutoff value obtained from the confusion matrix. By default, this value is based on the classification decision rule that is calculated by the platform. If the threshold value is changed to 0.5, the Threshold Predicted column is not saved to the data table because it would be equivalent to the Most Likely column.

Publish Probability Formula (Available only if the response is binary.) Creates probability formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option opens the Formula Depot window. See "Formula Depot".

Save Validation (Available only when the Holdback or KFold validation methods are used.) Creates a new column in the data table that identifies which rows were used in the training and validation sets.

Remove Fit Removes the entire model report.

Additional Example of the SVM Platform

Example of Support Vector Regression for Continuous Response

Support vector regression (SVR) models use the SVM algorithm to predict continuous response data. This example uses the same medical data on diabetic patients that was used in "Example of Support Vector Machines". Instead of a binary measure of diabetes disease progression, you now have a continuous measure. Higher values correspond to more disease progression. Vary the cost parameter of the radial basis function to find the best fitting model.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Analyze > Predictive Modeling > Support Vector Machines**.
3. Select Y and click **Y, Response**.
4. Select Age through Glucose and click **X, Factor**.
5. Select Validation and click **Validation**.
6. Click **OK**.
7. In the Model Launch control panel, check that the kernel function is a Radial Basis Function with Cost parameter 1 and Gamma parameter 0.1.
8. Click **Go**.
9. Click the gray triangle next to Model Launch to open the Model Launch control panel.
10. Change the cost parameter to 0.1.
11. Click **Go**.
12. Click the gray triangle next to Model Launch to open the Model Launch control panel.
13. Change the cost parameter to 2.
14. Click **Go**.

Figure 9.6 Model Comparison Report

Model Comparison

Show	Method	Kernel Function	Cost	Gamma	# SV	Training RASE	Validation RASE	Validation Generalized RSquare	Best
☑	Model 1	Radial Basis Function	1	0.1	266	43.8147	58.3285	0.42552	
☑	Model 2	Radial Basis Function	0.1	0.1	280	52.5804	57.5025	0.44167	Smallest RASE
☑	Model 3	Radial Basis Function	2	0.1	265	40.553	60.7801	0.37621	

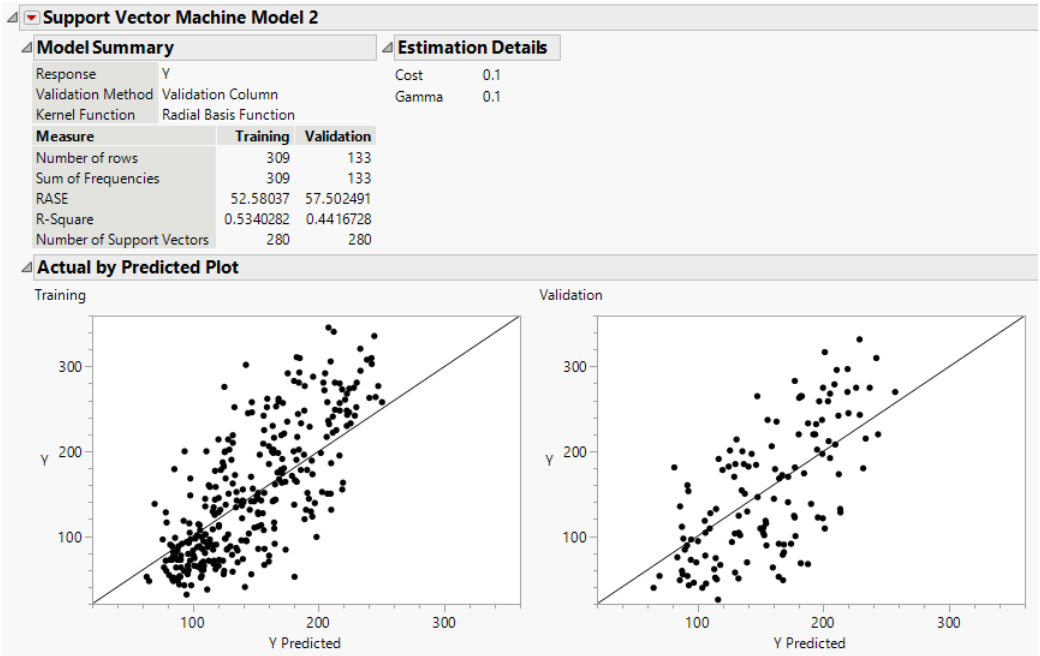
Validation Column: Validation

The Model Comparison report contains the results of the models using different cost parameters. Recall that a higher cost parameter implements an algorithm that is less likely to misclassify a point, whereas a lower cost parameter implements a more flexible

algorithm. In this scenario, the more flexible model (cost parameter equal to 0.1) produces the best model fit.

15. In the Show column of the Model Comparison report, deselect Model 1 and Model 3.

Figure 9.7 Model Report for Continuous Response



The model report for a continuous response contains the Model Summary, Estimation Details, and Actual by Predicted Plot. In the Model Summary report, the RASE is 52.58 for the training set and 57.50 for the validation set.

Chapter 10

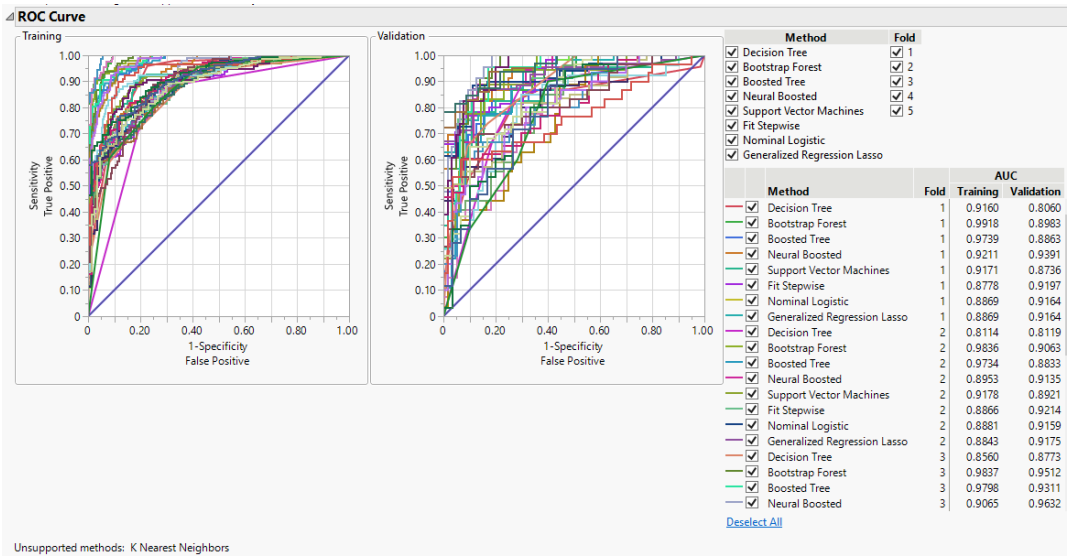
JMP PRO Model Screening

Fit and Compare Multiple Prediction Models

The Model Screening platform is available only in JMP Pro.

The Model Screening platform in JMP Pro enables you to run multiple predictive modeling platforms from one launch window and assemble summaries from the different methods. This helps you pick the best performing model without launching each individual model platform. You can also easily launch an individual platform report for any of the models from the Model Screening report window for further refinement.

Figure 10.1 Model Screening Report



Contents

Example of Model Screening for Continuous Response 173
Launch the Model Screening Platform 175
The Model Screening Report 178
Summary Across the Folds Report 179
Training, Validation, and Test Measures of Fit 180
Model Screening Platform Options 181
Decision Thresholds Report 184

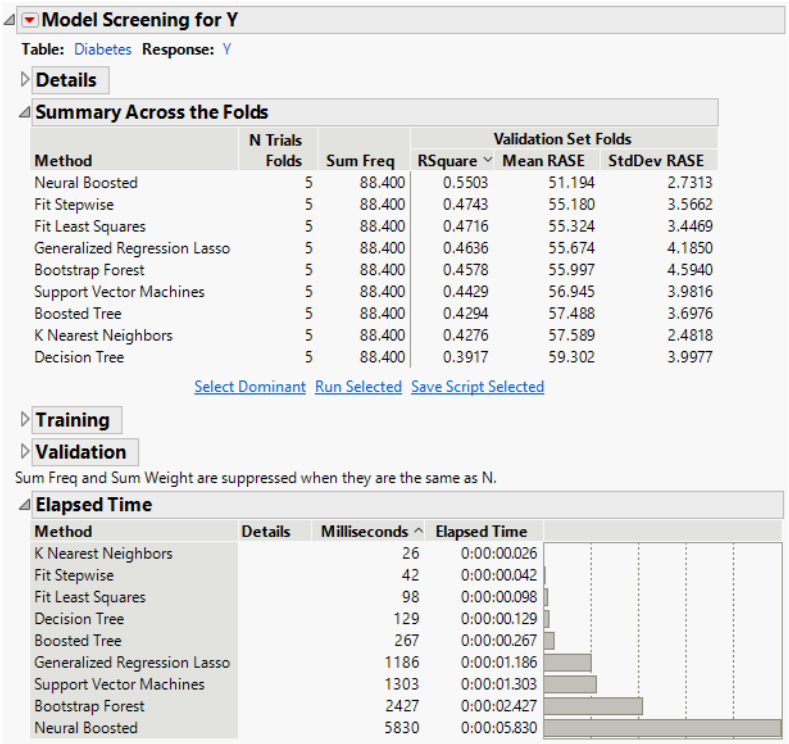
JMP PRO Example of Model Screening for Continuous Response

You have baseline medical data for 442 diabetic patients. You also have a continuous measure of diabetes disease progression obtained one year after each patient's initial visit. Higher values correspond to more disease progression. You want to assess various models for their ability to predict diabetes disease progression.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Analyze > Predictive Modeling > Model Screening**.
3. Select Y and click **Y, Response**.
4. Select Age through Glucose and click **X, Factor**.
5. In the Folded Crossvalidation section, select the box next to **K Fold Crossvalidation**.
6. (Optional) In the Options section, type 42920 next to **Set Random Seed**.

   Use the random seed in order to match the example output.
7. Click **OK**.
8. Click the red triangle next to Model Screening for Y and select **Optional Reports > Elapsed Time**.

Figure 10.2 Model Report for Continuous Response



Model Screening for Y

Table: Diabetes **Response:** Y

Details

Summary Across the Folds

Method	N Trials Folds	Sum Freq	Validation Set Folds RSquare	Mean RASE	StdDev RASE
Neural Boosted	5	88.400	0.5503	51.194	2.7313
Fit Stepwise	5	88.400	0.4743	55.180	3.5662
Fit Least Squares	5	88.400	0.4716	55.324	3.4469
Generalized Regression Lasso	5	88.400	0.4636	55.674	4.1850
Bootstrap Forest	5	88.400	0.4578	55.997	4.5940
Support Vector Machines	5	88.400	0.4429	56.945	3.9816
Boosted Tree	5	88.400	0.4294	57.488	3.6976
K Nearest Neighbors	5	88.400	0.4276	57.589	2.4818
Decision Tree	5	88.400	0.3917	59.302	3.9977

Select Dominant Run Selected Save Script Selected

Training

Validation

Sum Freq and Sum Weight are suppressed when they are the same as N.

Elapsed Time

Method	Details	Milliseconds	Elapsed Time
K Nearest Neighbors		26	0:00:00.026
Fit Stepwise		42	0:00:00.042
Fit Least Squares		98	0:00:00.098
Decision Tree		129	0:00:00.129
Boosted Tree		267	0:00:00.267
Generalized Regression Lasso		1186	0:00:01.186
Support Vector Machines		1303	0:00:01.303
Bootstrap Forest		2427	0:00:02.427
Neural Boosted		5830	0:00:05.830

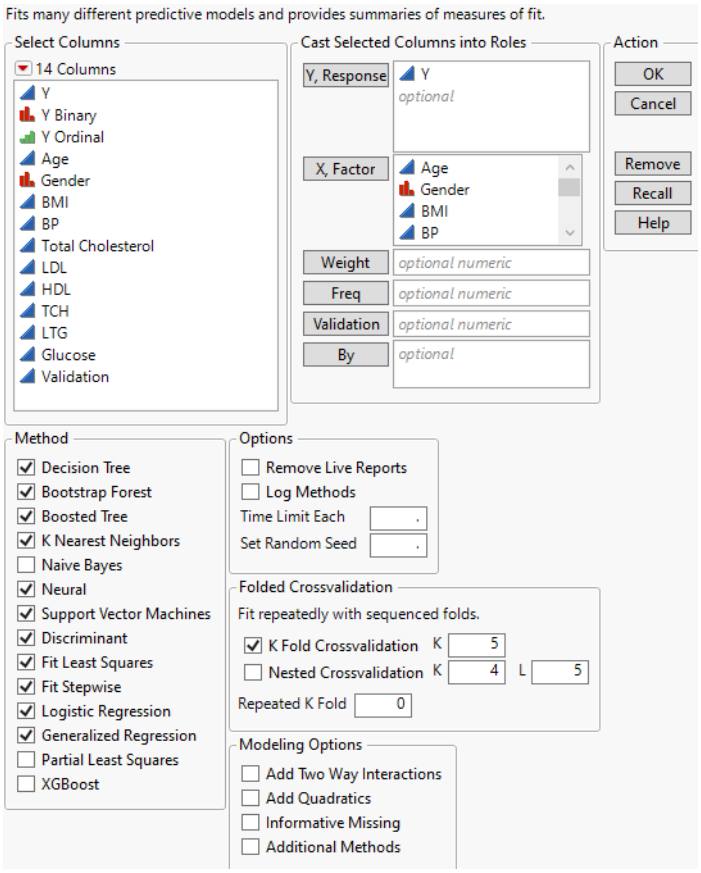
The best performing model based on the Validation RSquare averaged across the folds, is Neural Boosted. The average Validation RSquare for Neural Boosted is 0.5503. The Elapsed Time report shows that the Neural Boosted models also took the longest to fit. In this case, the elapsed time was only 5 seconds, but this information could be important for larger data sets or more complex models.

Tip: To see performance metrics for the individual models, click the gray disclosure icon next to Validation.

JMP PRO Launch the Model Screening Platform

To launch the Model Screening platform, select **Analyze > Predictive Modeling > Model Screening**.

Figure 10.3 The Model Screening Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Response The response variable or variables that you want to analyze.

X, Factor The predictor variables.

Weight (Not applicable to the K Nearest Neighbors, Support Vector Machines, or Neural modeling platforms.) A column whose numeric values assign a weight to each row in the analysis.

Freq (Not applicable to the K Nearest Neighbors modeling platform.) A column whose numeric values assign a frequency to each row in the analysis.

Validation (Not applicable if any of the Crossvalidation options are selected in the launch window.) A numeric column that defines the validation sets. If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see "Make Validation Column".

Note: If you specify a validation column with more than three levels, this column is used to perform K Fold crossvalidation.

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Methods Enables you to select the desired modeling platforms. By default, the modeling platforms that are fit are Decision Tree (Partition), Bootstrap Forest, Boosted Tree, K Nearest Neighbors, Neural, Support Vector Machines, Discriminant, Fit Least Squares, Fit Stepwise, Logistic Regression, and Generalized Regression. Naive Bayes, Partial Least Squares, and XGBoost are also available.

Notes:

- XGBoost is not supported by JMP and is available only if the XGBoost add-in is installed. For more information about XGBoost, see https://community.jmp.com.
- Decision Tree (Partition), Discriminant, and Partial Least Squares all require some type of validation set in order to fit a model.
- If there are fewer than 20 observations in a validation set, a Decision Tree (Partition) model cannot be fit.
- The modeling platforms use default options and tuning parameters in model fitting. You can try to improve the fit past what the default yields by calling platforms directly and choosing different options.

Options

Provides additional options.

Remove Live Reports Does not include the individual model platform reports in the Model Screening report window.

Tip: Select this option to free up memory when you have a large problem with many methods and fits.

Log Methods Writes out a progress message to the log each time a fitting platform is called.

Time Limit Each Specifies a time limit, in seconds, for each fit. For platforms that support early stopping, the best estimates up to that point are provided. For platforms that do not support early stopping, no result is provided.

Set Random Seed Sets a random seed that is used for any random components of the model fit routines. This enables you to rerun the platform and obtain the same model fits.

Folded Crossvalidation

Provides options for various types of crossvalidation.

K Fold Crossvalidation Divides the data randomly into *K* parts or folds. Each model is fit to the data *K* times, each time with a different fold held out as a crossvalidation set. A total of *K* models are fit. The default value of *K* is 5.

- *K* specifies the number of folds for K Fold Crossvalidation. The default is 5 and *K* must be greater than 1.

Nested Crossvalidation Divides the data into nested folds for crossvalidation. First, the data are divided into $k = 1, ..., K$ equals parts, or *folds*. For each fold, the k^{th} fold is used as a test set and the remaining data are divided further into *L* equal parts. These *L* subdivisions are called *inner folds*. Then, a model is fit to the data *L* times with a different inner fold held out each time as a crossvalidation set. The *L* models then use the k^{th} fold as a common test data set. In all, a total of *K***L* models are fit. The default value of *K* is 4 and the default value of *L* is 5.

For example, set K = 2 and L = 3. The data are initially divided into two folds. The first fold is held out as a test set and the second fold is divided into 3 inner folds. Three models are fit to the data, each time with a different inner fold held out as a crossvalidation set. Then, all three models are tested on the first fold.

The second fold is then held out as a test set and the first fold is divided into 3 inner folds. Three models are fit to the data, each time with a different inner fold held out as a crossvalidation set. Then, all three models are tested on the second fold.

- *K* specifies the number of folds for Nested Crossvalidation. The default is 4 and *K* must be greater than 1.
- *L* specifies the number of inner folds for Nested Crossvalidation. The default is 5 and *L* must be greater than 1.

Note: If both K Fold Crossvalidation and Nested Crossvalidation are selected, Nested Crossvalidation is performed.

Repeated K Fold Specifies the number of times the K Fold Crossvalidation or Nested Crossvalidation process is repeated.

Modeling Options

Provides additional options for the modeling platforms.

Add Two Way Interactions Adds all two way interaction effects to linear models.

Add Quadratics Adds effects for the squares of continuous variables to linear models.

Informative Missing Enables informative missing for all platforms.

Additional Methods Calls several additional methods, such as Ridge, Elastic Net and Lasso, in the Generalized Regression platform. See *Fitting Linear Models*.

Caution: This results in additional model fits.

When you click OK, the specified models are fit and a set of progress bars are shown. The upper progress bar reports the progress across all fits. The lower progress bar reports the progress for the current individual model fit. You can stop the lower progress bar to employ early stopping and the upper progress bar will continue to run.

JMP PRO The Model Screening Report

The content of the Model Screening report depends on both the type of crossvalidation implemented and the modeling type of the response variable. At the top of the Model Screening report, the name of the data table and column name of the response variable are shown. If a validation column is specified, the name of that column is also shown.

By default, the Details report appears next. This report contains the individual model reports for all of the models for each fold and trial. These reports are the actual output from each platform and contain all reports and command features. For example, if you decide on a model that is best, you can go to the corresponding platform report in the Details report and save the desired prediction formula column. You can hide the Details report by selecting the Remove Live Reports option in the launch window.

If either the K Fold Crossvalidation option or the Nested Crossvalidation option is specified in the launch window, the Summary Across the Folds report appears. This report is a summary of the measures of fit averaged across the test folds.

Measures of fit are shown for each individual model fit, for the training, validation, and test sets if applicable. If crossvalidation is used by specifying a validation column in the launch window, only the individual model fit measures of fit are shown in the report.

Summary Across the Folds Report

The Summary Across the Folds report contains a summary of the measures of fit across the folds, as well as across the trials if Repeated K Fold is specified. If the K Fold Crossvalidation option is specified, the measures of fit are summarized across the validation sets. If the Nested K Fold Crossvalidation option is specified, the measures of fit are summarized across the test sets. The report contains a table with the following columns:

Method The name of the method used to fit the model.

N Trials Folds The total number of models that were fit across all folds and trials, if applicable.

Sum Freq The average number of observations in the validation or test sets. The test sets are used to estimate model performance.

RSquare The mean RSquare across all validation or test set folds. This column contains the Entropy RSquare when the response is categorical.

Mean RASE The mean RASE (Root Average Square Error) across all validation or test set folds.

StdDev RASE The standard deviation of the RASE across all validation or test sets.

Mean AUC (Available only for categorical responses.) The mean area under the ROC curve (AUC) across all validation or test sets.

Mean MR (Available only for categorical responses.) The mean misclassification rate (MR) across all validation or test sets.

The following options are available below the table:

Select Dominant Selects each model that is better than or equal to all of the other models in terms of a combination of model fitting criteria. This is also referred to as selecting the Pareto Frontier. For continuous responses, RSquare and Sum Freq are considered when determining the dominant model. For categorical responses, Entropy RSquare, Misclassification Rate, AUC, and Sum Freq are considered when determining the dominant model.

Run Selected Runs the individual model for each selected row. If any type of folded crossvalidation is specified in the launch window, the model run uses the fold, inner fold, and trial combination that corresponds to the final model. The final model is the model that produces the highest weighted average RSquare. The weight average RSquare is the

average, weighted by number of observations, of the training RSquare, validation RSquare, and when a test set is used, the test RSquare. A validation column is created in the data table for any fold, inner fold, and trial combination that is required by clicking Run Selected.

Save Script Selected Saves a model script to the script window for each selected row. If any type of folded crossvalidation is specified in the launch window, the validation set specified in the script uses the fold, inner fold, and trial combination that corresponds to the final model. The final model is the model that produces the highest weighted average RSquare. The weight average RSquare is the average, weighted by number of observations, of the training RSquare, validation RSquare, and when a test set is used, the test RSquare. A validation column is created in the data table for any fold, inner fold, and trial combination that is requested by clicking Save Script Selected.

Training, Validation, and Test Measures of Fit

There is a measures of fit report for each model data set that is specified. These could be Training, Training and Validation, or Training, Validation, and Test sets. Each report contains a table with the following columns:

Method The name of the method used to fit the model.

N The number of observations in the set.

Sum Wgt The sum of the weights.

RSquare (Available only for continuous responses.) The RSquare value of the fitted model.

Entropy RSquare (Available only for categorical responses.) A measure of fit that compares the log-likelihoods from the fitted model and the constant probability model. Entropy RSquare ranges from 0 to 1, where values closer to 1 indicate a better fit. See "Entropy RSquare".

Misclassification Rate (Available only for categorical responses.) The proportion of observations misclassified by the model. Smaller values indicate a better fit.

Note: In these tables, the misclassification rate is always calculated using a probability threshold of 0.5.

AUC (Available only for categorical responses.) The area under the ROC curve. Values closer to 1 indicate a better fit.

RASE The square root of the mean squared prediction error (Root Average Square Error). RASE is computed as follows, where *Source* indicates the Training, Validation, or Test set.

$$\text{RASE}_{Source} = \sqrt{\frac{SSE_{Source}}{n}}$$

Generalized RSquare (Available only for categorical responses.) A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The value is 1 for a perfect model, and 0 for a model no better than a constant model. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991).

Fold (Available only if the K Fold Crossvalidation option or the Nested Crossvalidation option is specified in the launch window.) Identifies the fold that is held out for the model fit in that row.

Inner Fold (Available only if the Nested Crossvalidation option is specified in the launch window.) Identifies the inner fold that is held out for the model fit in that row.

Trial (Available only if the Repeated K Fold option is specified in the launch window.) Identifies the trial number for the model fit in that row.

The following options are available below each table:

Select Dominant Selects each model that is better than or equal to all of the other models in terms of a combination of model fitting criteria. For continuous responses, RSquare and Sum Freq are considered when determining the dominant model. For categorical responses, Entropy RSquare, Misclassification Rate, AUC, and Sum Freq are considered when determining the dominant model.

Run Selected Runs the individual models specified in each selected row.

Save Script Selected Saves a model script to the script window for each selected row.

JMP PRO Model Screening Platform Options

Plot Actual by Predicted (Available only for continuous responses.) Shows or hides a report of plots that overlay the actual by predicted points from all of the model fits. There are separate plots for the Training, Validation, and Test sets. The report contains a Methods legend with check boxes that enable you to specify which fits appear on the plots based on the method, fold, inner fold, and trial combination. Click the Deselect All button at the bottom of the legend to remove all model fits from the plots.

Notes:

- If you select a subset of models from the Training, Validation, or Test reports and then select Plot Actual by Predicted, only the plots for the selected models are shown.
- The actual by predicted plots are not supported by all of the modeling platforms. A list of unsupported platforms is shown below the plots.

Profiler (Available only for continuous responses.) Shows or hides prediction profilers for each type of model fit. See *Profilers*. If any type of folded crossvalidation is specified in the launch window, the selected model in the prediction profiler uses the fold, inner fold, and trial combination that corresponds to the final model. The final model is the model that produces the highest weighted average RSquare. The weight average RSquare is the average, weighted by number of observations, of the training RSquare, validation RSquare, and when a test set is used, the test RSquare.

Notes:

- Confidence intervals are shown in prediction profilers for Standard Least Squares models and Generalized Regression models.
- Prediction profilers are not available for K Nearest Neighbors models.

ROC Curve (Available only for categorical responses.) Shows or hides a report of plots that overlay the ROC curves for all of the model fits. There are separate plots for the Training, Validation, and Test sets. The report contains two legends with check boxes that enable you to specify which fits appear on the plots. Click the Deselect All button at the bottom of the legend to remove all model fits from the plots. For more information about ROC Curves, see "ROC Curve".

Notes:

- If you select a subset of models from the Training, Validation, or Test reports and then select ROC Curves, only the plots for the selected models are shown.
- The ROC plots are not supported by all of the modeling platforms. A list of unsupported platforms is shown below the plots.

Show Profit (Available only for binary categorical responses that have a Profit Matrix column property.) Shows or hides the expected profit for each model.

Decision Threshold (Available only for binary categorical responses.) Shows or hides Decision Thresholds reports for the training, validation, and test sets, if specified. Each report contains a graph of the distribution of fitted probabilities for each model, confusion matrices for each model, and classification graphs to adjust thresholds and compare the model fits. See "Decision Thresholds Report".

Note: If you select a subset of models from the Training, Validation, or Test reports and then select Decision Threshold, only the information for the selected models is shown.

Save Results Table Saves the information in the Validation report to a new data table titled Model Screening Statistics Validation Set. If there is a test set, such as when the Nested Crossvalidation option is specified, the information in the Test report is also saved to a new data table titled Model Screening Statistics Test Set.

Save KFold Results Table (Available only if a Crossvalidation option is specified in the launch window.) Saves the information in the Summary Across the Folds report to a new data table titled KFold Results.

Save Prediction Formulas (Available only if the live reports are included in the model report.) Saves the prediction formulas for the selected models to the data table. If you select a model in the Summary Across the Folds report and any type of fold crossvalidation is specified in the launch window, the prediction formula is for the fold, inner fold, and trial combination that corresponds to the final model. The final model is the model that produces the highest weighted average RSquare. The weight average RSquare is the average, weighted by number of observations, of the training RSquare, validation RSquare, and when a test set is used, the test RSquare.

Notes:

- A prediction formula is not saved for Stepwise models.
- A prediction formula is not saved for SVM models when the response is ordinal.

Optional Reports Shows a submenu of additional report options.

Elapsed Time Shows or hides a report that contains the total elapsed time that was spent fitting each type of model.

Cardinality of Predictors Shows or hides a report of the number of levels and how many parameters are used in the linear model fit for each categorical predictor.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Decision Thresholds Report

The Decision Thresholds report enables you to explore thresholds for binary classification models. There is a Decision Thresholds report for each model data set that is specified by the validation method. These could be Training, Training and Validation, or Training, Validation, and Test sets. Each Decision Thresholds report contains a graph of the distribution of fitted probabilities, a bar chart of classifications, and confusion matrices. All are organized by model fit, fold, trial, and class level. The report also contains a tabbed section on classification accuracy measures and an option to set the profit matrix. The report updates as you adjust the probability threshold.

Distribution of Fitted Probabilities

The distribution of fitted probabilities, or model scores, enables you to see how each individual model fit differentiates between the two classes. A vertical line on the graph represents the probability threshold, which determines the classification of each observation. By default, the probability threshold is 0.5. You can change the probability threshold by dragging the vertical line or by clicking the Probability Threshold value and entering a new value. This changes the probability threshold value across the whole Decision Thresholds report. The value of the probability threshold must be between 0 and 1.

Classification Counts

The bar chart of classifications shows the classification counts for each level of the response variable at the current threshold. Green bars represent correctly classified observations; red bars represent incorrectly classified observations.

Confusion Matrices

The confusion matrices, also known as contingency tables, show the two-way classification of actual and predicted responses for each individual model fit. Confusion rates matrices are shown as well. The rates are equal to the values in the confusion matrices divided by the row totals.

Classification Accuracy Measures

False Classification by Threshold Shows a plot of the misclassification count by probability threshold and a plot of the misclassification rate by probability threshold. Each plot contains two curves for each individual model fit. The curves for the low response category are solid and the curves for the high response category are dashed. The curves intersect at the threshold that yields equal misclassification, counts or rates, of each response level. There is also a vertical line on each graph that represents the current

probability threshold value. You can change the probability threshold value by dragging the vertical line. This changes the probability threshold value across the whole report.

False Classification by Portion Shows a plot of the misclassification count or rate by the portion of the rank ordered scores. Each plot contains two curves for each individual model fit. The curves for the low response category are solid and the curves for the high response category are dashed.

True Classification by Threshold Shows a plot of the true count by the probability threshold and a plot of the true rate by the probability threshold. Each plot contains two curves for each individual model fit. The curves for the low response category are dashed and those for the high response level are solid. The curves intersect at the threshold that yields equal correct classifications, counts or rates, for each response. There is also a vertical line on each graph that represents the current probability threshold value. You can change the probability threshold value by dragging the vertical line. This changes the probability threshold value across the whole report.

True Classification by Portion Shows a plot of the true count or rate by the portion of the rank ordered scores. Each plot contains two curves for each individual model fit. The curves for the low response category are dashed and the curves for the high response category are solid.

Profit by Threshold (Available only if a profit matrix is specified.)Shows a plot of the average profit by the probability threshold. There is a curve for each individual model fit and a vertical line that represents the current probability threshold value. The specified profit matrix is also shown next to the plot.

Metrics Shows a table of classification accuracy metrics for each model. A legend is provided to describe how the metric in each column is calculated.

Set Profit Matrix

Enables you to assign costs to undesirable outcomes and profits to desirable outcomes. See "Specify Profit Matrix". If you change the probability threshold in the profit matrix window and click OK, the Decision Thresholds report is updated using that value as the probability threshold.

Chapter 11

JMP PRO Model Comparison

Compare the Predictive Ability of Fitted Models

The Model Comparison platform is available only in JMP Pro.

The Model Comparison platform in JMP Pro lets you compare the predictive ability of different models. Measures of fit are provided for each model along with overlaid diagnostic plots.

Figure 11.1 Example of Comparing Models

Contents

Example of Model Comparison 189
Launch the Model Comparison Platform 192
The Model Comparison Report. 193
Model Comparison Platform Options 195
Continuous and Categorical Responses 195
Continuous Responses 195
Categorical Responses 195
Additional Example of Model Comparison 196

JMP PRO Example of Model Comparison

This section provides an example of using the Model Comparison platform. The example uses demographic data to build a model for median home price. A regression model and a bootstrap forest model are compared.

Begin by selecting **Help > Sample Data Library** and opening Boston Housing.jmp.

Create a Validation Column

1. Select **Analyze** > **Predictive Modeling** > **Make Validation Column**.
2. Do not select any columns in the launch window.

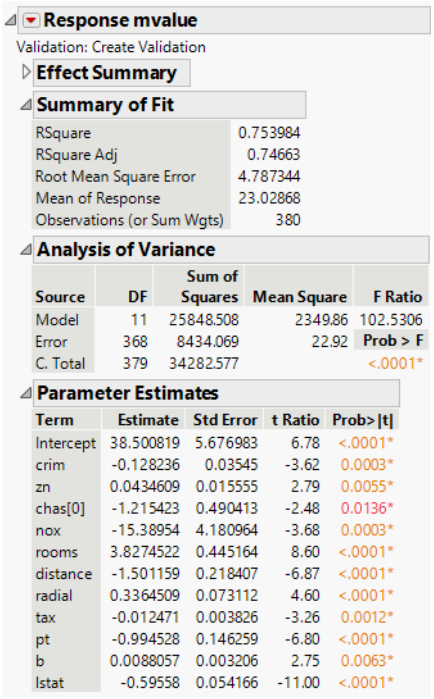
   This indicates that the platform will create a simple random validation column
3. Click **OK**.
4. In the box next to New Column Name, type Create Validation.
5. In the box next to Random Seed, enter 1234.
6. Click **Go**.

   A new Validation column is created. The rows assigned a 0 are the training set. The rows assigned a 1 are the validation set.

Create the Regression Model and Save the Prediction Formula to a Column

1. Select **Analyze > Fit Model**.
2. Select mvalue and click **Y**.
3. Select crim through lstat and click **Add**.
4. Select Create Validation and click **Validation**.
5. Select **Stepwise** in the Personality list.
6. Click the **Run** button.
7. Select **P-value Threshold** from the Stopping Rule list.
8. Click the **Go** button.
9. Click the **Run Model** button.

Figure 11.2 Fit Model Report



Response mvalue

Validation: Create Validation

Effect Summary

Summary of Fit

RSquare	0.753984
RSquare Adj	0.74663
Root Mean Square Error	4.787344
Mean of Response	23.02868
Observations (or Sum Wgts)	380

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	11	25848.508	2349.86	102.5306
Error	368	8434.069	22.92	Prob > F
C. Total	379	34282.577		<.0001*

Parameter Estimates

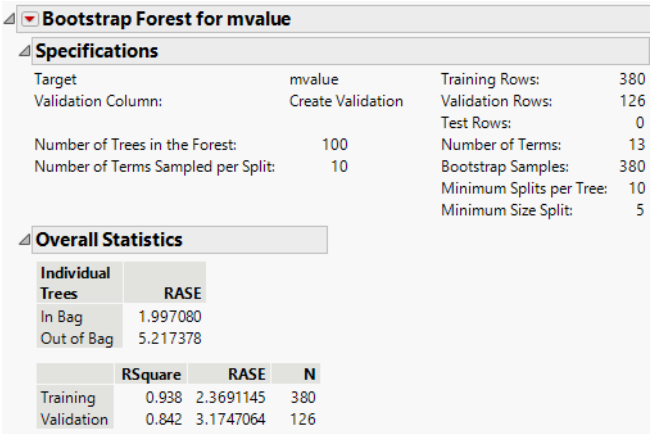
Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	38.500819	5.676983	6.78	<.0001*
crim	-0.128236	0.03545	-3.62	0.0003*
zn	0.0434609	0.015555	2.79	0.0055*
chas[0]	-1.215423	0.490413	-2.48	0.0136*
nox	-15.38954	4.180964	-3.68	0.0003*
rooms	3.8274522	0.445164	8.60	<.0001*
distance	-1.501159	0.218407	-6.87	<.0001*
radial	0.3364509	0.073112	4.60	<.0001*
tax	-0.012471	0.003826	-3.26	0.0012*
pt	-0.994528	0.146259	-6.80	<.0001*
b	0.0088057	0.003206	2.75	0.0063*
lstat	-0.59558	0.054166	-11.00	<.0001*

10. To save the prediction formula to a column, click the Response red triangle and select **Save Columns > Prediction Formula**.

Create the Bootstrap Forest Model and Save the Prediction Formula to a Column

1. Select **Analyze > Predictive Modeling > Bootstrap Forest**.
2. Select mvalue and click **Y, Response**.
3. Select crim through lstat and click **X, Factor**.
4. Select Create Validation and click **Validation**.
5. Click **OK**.
6. Select the **Early Stopping** check box.
7. Select the **Multiple Fits over number of terms** check box.
8. Enter 617 in the box next to **Random Seed**.
9. Click **OK**.

Figure 11.3 Bootstrap Forest Model



Bootstrap Forest for mvalue

Specifications

Target	mvalue	Training Rows:	380
Validation Column:	Create Validation	Validation Rows:	126
		Test Rows:	0
Number of Trees in the Forest:	100	Number of Terms:	13
Number of Terms Sampled per Split:	10	Bootstrap Samples:	380
		Minimum Splits per Tree:	10
		Minimum Size Split:	5

Overall Statistics

Individual Trees	RASE
In Bag	1.997080
Out of Bag	5.217378

	RSquare	RASE	N
Training	0.938	2.3691145	380
Validation	0.842	3.1747064	126

10. To save the prediction formula to a column, click the Bootstrap Forest red triangle and select **Save Columns > Save Prediction Formula**.

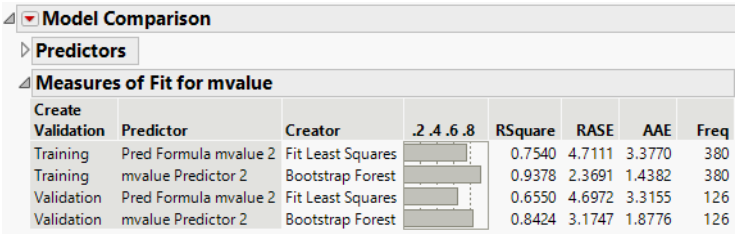
Compare the Models

1. Select **Analyze > Predictive Modeling > Model Comparison**.
2. Select the two prediction formula columns and click **Y, Predictors**.
3. Select Create Validation and click **Group**.

> **Tip:** If a **Group** column is not specified, JMP automatically recognizes when the same validation column has been used for all predictors and prompts you to add it as a grouping variable.

4. Click **OK**.

Figure 11.4 Model Comparison Report



Model Comparison

Predictors

Measures of Fit for mvalue

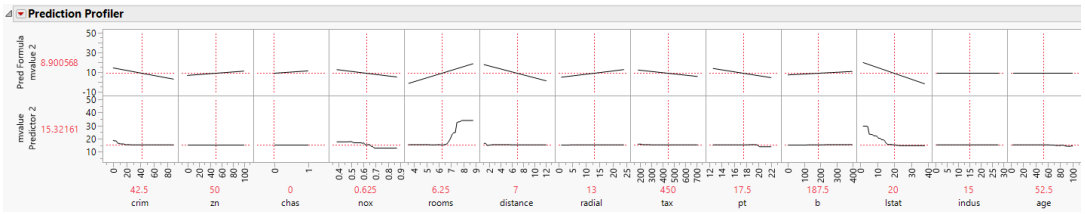
Create Validation	Predictor	Creator	.2 .4 .6 .8	RSquare	RASE	AAE	Freq
Training	Pred Formula mvalue 2	Fit Least Squares		0.7540	4.7111	3.3770	380
Training	mvalue Predictor 2	Bootstrap Forest		0.9378	2.3691	1.4382	380
Validation	Pred Formula mvalue 2	Fit Least Squares		0.6550	4.6972	3.3155	126
Validation	mvalue Predictor 2	Bootstrap Forest		0.8424	3.1747	1.8776	126

The rows in the training set were used to build the models, so the RSquare statistics for Create Validation = Training might be artificially inflated. In this case, the statistics are not representative of the models' future predictive ability. This is especially true for the bootstrap forest model.

Compare the models using the statistics for Create Validation = Validation. In this case, the bootstrap forest model predicts better than the regression model.

5. Click the Model Comparison red triangle and select Profiler.

Figure 11.5 Prediction Profiler for All Models



The prediction profiler enables you to compare the impact of each factor in the different models. The profiler is especially interesting when comparing different types of models such as here where you have a regression model and a partition model.

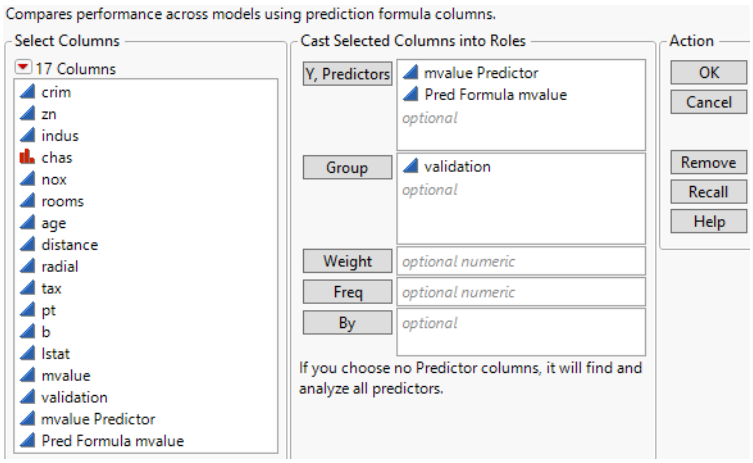
Related Information

- *Fitting Linear Models*
- "Partition Models"

JMP PRO Launch the Model Comparison Platform

To launch the Model Comparison platform, select **Analyze > Predictive Modeling > Model Comparison**.

Figure 11.6 The Model Comparison Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Predictors The columns that contain the predictions for the models that you want to compare. They can be either formula columns or just data columns. Prediction formula columns created by JMP platforms have either the Predicting or Response Probability column property. If you specify a column that does not contain one of these properties, the platform prompts you to specify which column is being predicted by the specified Y column.

For a categorical response with *k* levels, most model fitting platforms save *k* columns to the data table, each predicting the probability for a level. All *k* columns need to be specified as **Y, Predictors**. For platforms that do not save *k* columns of probabilities, the column containing the predicted response level can be specified as a **Y, Predictors** column.

If you do not specify any **Y, Predictors** columns, JMP uses the prediction formula columns in the data table that have either the Predicting or Response Probability column property.

Group The column that separates the data into groups, which are evaluated separately. If a **Group** column is not specified, JMP automatically recognizes when the same validation column has been used for all predictors and prompts you to add it as a grouping variable.

The other role buttons are common among JMP platforms. See *Using JMP*.

JMP PRO The Model Comparison Report

Figure 11.7 shows an example of the initial Model Comparison report for a continuous response.

Figure 11.7 Initial Model Comparison Report

Model Comparison

Predictors

Measures of Fit for mvalue

Create Validation	Predictor	Creator	.2 .4 .6 .8	RSquare	RASE	AAE	Freq
Training	Pred Formula mvalue 2	Fit Least Squares		0.7540	4.7111	3.3770	380
Training	mvalue Predictor 2	Bootstrap Forest		0.9378	2.3691	1.4382	380
Validation	Pred Formula mvalue 2	Fit Least Squares		0.6550	4.6972	3.3155	126
Validation	mvalue Predictor 2	Bootstrap Forest		0.8424	3.1747	1.8776	126

The Predictors report shows all responses and all models being compared for each response. The fitting platform that created the predictor column is also listed.

The Measures of Fit report shows measures of fit for each model. The columns are different for continuous and categorical responses.

Measures of Fit for Continuous Responses

RSquare The *r*-squared statistic. In data tables that contain no missing values, the *r*-squared statistics in the Model Comparison report and original models match. However, if there are any missing values, the *r*-squared statistics differ.

RASE The square root of the mean squared prediction error. This is computed as follows:

- Square and sum the prediction errors (differences between the actual responses and the predicted responses) to obtain the *SSE*.
- Denote the number of observations by n.
- RASE is:

$$\text{RASE} = \sqrt{\frac{SSE}{n}}$$

AAE The average absolute error.

Freq The column that contains frequency counts for each row.

Measures of Fit for Categorical Responses

Entropy RSquare One minus the ratio of the negative log-likelihoods from the fitted model and the constant probability model. It ranges from 0 to 1.

Generalized RSquare A measure that can be applied to general regression models. It is based on the likelihood function L and is scaled to have a maximum value of 1. The value is 1 for a perfect model, and 0 for a model no better than a constant model. The Generalized RSquare measure simplifies to the traditional RSquare for continuous normal responses in the standard least squares setting. Generalized RSquare is also known as the Nagelkerke or Craig and Uhler R^2, which is a normalized version of Cox and Snell's pseudo R^2. See Nagelkerke (1991).

Mean -Log p The average of -log(p), where p is the fitted probability associated with the event that occurred.

RASE The root average squared prediction error. For categorical responses, the differences are between 1 and p, the fitted probability for the response level that actually occurred.

Mean Abs Dev The average of the absolute values of the differences between the response and the predicted response. For categorical responses, the differences are between 1 and p (the fitted probability for the response level that actually occurred).

Misclassification Rate The rate for which the response category with the highest fitted probability is not the observed category.

N The number of observations.

Related Information

"Training and Validation Measures of Fit" provides more information about measures of fit for categorical responses.

JMP PRO Model Comparison Platform Options

JMP PRO Continuous and Categorical Responses

Model Averaging Makes a new column of the arithmetic mean of the predicted values (for continuous responses) or the predicted.probabilities (for categorical responses).

JMP PRO Continuous Responses

Plot Actual by Predicted Shows a scatterplot of the actual versus the predicted values. The plots for the different models are overlaid.

Plot Residual by Row Shows a plot of the residuals by row number. The plots for the different models are overlaid.

Profiler Shows a profiler for each response based on prediction formula columns in your data. The profilers have a row for each model being compared.

JMP PRO Categorical Responses

ROC Curve Shows ROC curves for each level of the response variable. The curves for the different models are overlaid.

AUC Comparison Provides a comparison of the area under the ROC curve (AUC) from each model. The area under the curve is the indicator of the goodness of fit, where 1 is a perfect fit.

The report includes the following information:

- standard errors and confidence intervals for each AUC
- standard errors, confidence intervals, and hypothesis tests for the difference between each pair of AUCs
- an overall hypothesis test for testing whether all AUCs are equal

Lift Curve Shows lift curves for each level of the response variable. The curves for the different models are overlaid.

Cum Gains Curve Shows cumulative gains curves for each level of the response variable. A cumulative gains curve is a plot of the proportion of a response level that is identified by the model against the proportion of all responses. A cumulative gains curve for a perfect model would reach 1.0 at the overall proportion of the response level. The curves for the different models are overlaid.

Confusion Matrix Shows confusion matrices for each model. A confusion matrix is a two-way classification of actual and predicted responses. Count and rate confusion matrices are shown. Separate confusion matrices are produced for each level of the Group variable.

If the response has a Profit Matrix column property, then Actual by Decision Count and Actual by Decision Rate matrices are shown to the right of the confusion matrices. For more information about these matrices, see "Additional Examples of Partitioning".

Profiler Shows a profiler for each response based on prediction formula columns in your data. The profilers have a row for each model being compared.

Decision Threshold (Available only for binary categorical responses.) Shows or hides Decision Thresholds reports for the training, validation, and test sets, if specified. Each report contains a graph of the distribution of fitted probabilities for each model, confusion matrices for each model, and classification graphs to compare the model fits. See "Decision Thresholds Report".

Related Information

- "ROC Curve"
- "Lift Curve"

JMP PRO Additional Example of Model Comparison

This example uses automobile data to build a model to predict the size of the purchased car. A logistic regression model and a decision tree model are compared.

Begin by selecting **Help > Sample Data Library** and opening Car Physical Data.jmp.

Create the Logistic Regression Model

1. Select **Analyze > Fit Model**.
2. Select Type and click **Y**.
3. Select the following columns and click **Add**: Country, Weight, Turning Cycle, Displacement, and Horsepower.
4. Click **Run**.

The Nominal Logistic Fit report appears.

5. To save the prediction formulas to columns, click the Nominal Logistic red triangle and select **Save Probability Formula**.

Create the Decision Tree Model and Save the Prediction Formula to a Column

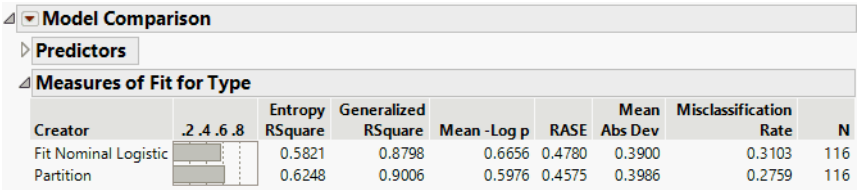
1. Select **Analyze > Predictive Modeling > Partition**.
2. Select Type and click **Y, Response**.
3. Select the Country, Weight, Turning Cycle, Displacement, and Horsepower columns and click **X, Factor**.
4. Make sure that **Decision Tree** is selected in the Method list.
5. Click **OK**.

   The Partition report appears.
6. Click **Split** 10 times.
7. To save the prediction formulas to columns, click the Partition red triangle and select **Save Columns > Save Prediction Formula**.

Compare the Models

1. Select **Analyze > Predictive Modeling > Model Comparison**.
2. Select all columns that begin with Prob and click **Y, Predictors**.
3. Click **OK**.

Figure 11.8 Initial Model Comparison Report


Model Comparison

Predictors

Measures of Fit for Type

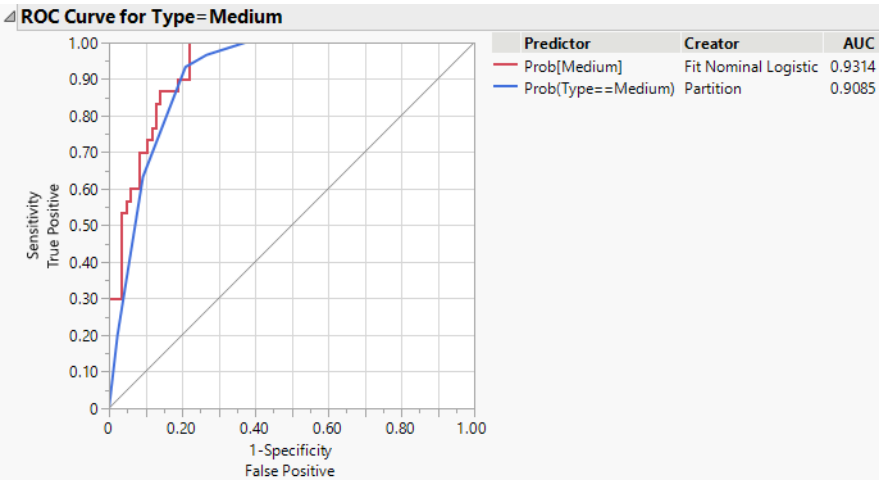
Creator	.2 .4 .6 .8	Entropy RSquare	Generalized RSquare	Mean -Log p	RASE	Mean Abs Dev	Misclassification Rate	N
Fit Nominal Logistic		0.5821	0.8798	0.6656	0.4780	0.3900	0.3103	116
Partition		0.6248	0.9006	0.5976	0.4575	0.3986	0.2759	116

The report shows that the Partition model has slightly higher values for Entropy RSquare and Generalized RSquare and a slightly lower value for Misclassification Rate.

4. Click the Model Comparison red triangle and select **ROC Curve**.

   ROC curves appear for each Type, one of which is shown in Figure 11.9.

Figure 11.9 ROC Curve for Medium



Examining all the ROC curves, you see that the two models are similar in their predictive ability.

5. Click the Model Comparison red triangle and select **AUC Comparison**.

   AUC Comparison reports appear for each Type, one of which is shown in Figure 11.10.

Figure 11.10 AUC Comparison for Medium

AUC Comparison for Type=Medium

Predictor	AUC	Std Error	Lower 95%	Upper 95%
Prob[Medium]	0.9314	0.0218	0.8742	0.9637
Prob(Type==Medium)	0.9085	0.0255	0.8448	0.9477

Predictor	vs. Predictor	AUC Difference	Std Error	Lower 95%	Upper 95%	ChiSquare	Prob>ChiSq
Prob[Medium]	Prob(Type==Medium)	0.0229	0.0234	-0.023	0.0687	0.9553	0.3284

Test	ChiSquare	DF	Prob>ChiSq
All AUCs equal	0.95534	1	0.3284

The report shows results for a hypothesis test for the difference between the AUC values (area under the ROC curve). Examining the results, you see there is no statistical difference between the values for any level of Type.

You conclude that there is no large difference between the predictive abilities of the two models for the following reasons:

- The R Square values and the ROC curves are similar.
- There is no statistically significant difference between AUC values.

Chapter 12

JMP PRO Make Validation Column

Divide Data into Training, Validation, and Test Sets

The Make Validation Column platform is available only in JMP Pro.

Validation is the process of using part of a data set to estimate model parameters and using another part to assess the predictive ability of a model. You can partition the data into two or three sets. The Make Validation Column platform provides five different methods to create these partitions.

Contents

Overview of the Make Validation Column Platform 201
Example of Make Validation Column 201
Launch the Make Validation Column Platform 203
Make Validation Column Report 206
Specify Rates or Relative Rates 206
Set Cutpoints 206
Options 207
Go 207
Make Validation Column Platform Options 207
Additional Example of the Make Validation Column Platform 208

JMP PRO Overview of the Make Validation Column Platform

Validation is the process of using part of a data set to estimate model parameters and using another part to assess the predictive ability of a model. With complex data, this can reduce the risk of model overfitting.

A validation column partitions the data into two or three parts.

- The training set is used to estimate the model parameters.
- The validation set is used to help choose a model with good predictive ability.
- The testing set checks the model's predictive ability after a model has been chosen.

A validation column can be used as a validation method in many JMP platforms. See "Validation in JMP Modeling".

The Make Validation Column platform enables you to create training, validation, and test sets using a variety of methods. You can specify stratification, grouping, or cutpoint columns to determine the method used to create the validation column.

JMP PRO Example of Make Validation Column

The Lipid Data.jmp data table contains blood measurements, physical measurements, and questionnaire data from 95 subjects at a California hospital. You are interested in creating a validation column to use during future analyses.

1. Select **Help > Sample Data Library** and open Lipid Data.jmp.
2. Select **Analyze > Distribution**.
3. Select Gender and click **Y, Columns**.
4. Click **OK**.

Figure 12.1 Distribution of Gender in Lipid Data.jmp

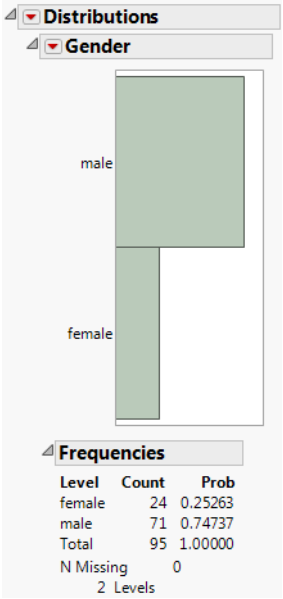


Figure 12.1 illustrates the distribution of Gender in the data set. Notice that there is not an equal proportion of males and females represented. Because there are fewer females within the data, you want to be sure to balance the genders across the validation and training sets.

5. Select **Analyze > Predictive Modeling > Make Validation Column**.
6. Select Gender and click **Stratification Columns**.
7. Click **OK**.

   The Make Validation Column report appears with a description of the validation method you selected. There are also options to change the rates, column types, or set a seed.
8. (Optional) Type 1234 next to **Random Seed** in the Options section of the report.
9. Click **Go**.

   A Validation column is added to the data table. You can explore the distribution of the validation and training sets by creating a Mosaic Plot.
10. Select **Analyze > Fit Y by X**.
11. Assign Validation to **Y, Response**, and Gender to **X, Factor**.
12. Click **OK**.

Figure 12.2 Distribution of Gender across Validation and Training Sets

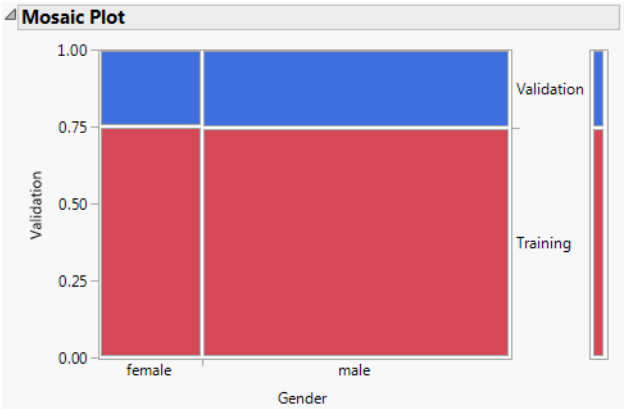
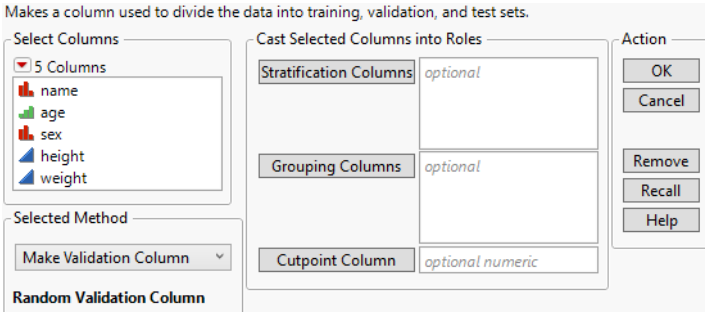


Figure 12.2 illustrates the distribution of Gender across each of the validation and training sets. Note that about 75% of both females and males are in the training set and about 25% of both females and males are in the validation set.

JMP PRO Launch the Make Validation Column Platform

Launch the Make Validation Column platform by selecting **Analyze > Predictive Modeling > Make Validation Column**.

Figure 12.3 Make Validation Column Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Make Validation Column launch window provides the following options:

Stratification Columns Assigns one or more stratification columns.

Grouping Columns Assigns one or more grouping columns.

Cutpoint Column Assigns a numeric cutpoint column.

Cutpoint Batch ID When a cutpoint column is assigned, you can also assign a column for cutpoint batch IDs. This enables you to determine cutpoint values within each level of the Cutpoint Batch ID column.

Selected Method

Provides two methods for validation.

Make Validation Column creates a validation column based on the specified stratification, grouping, and cutpoint columns. The validation column method determined by the specified stratification, grouping, and cutpoint columns is described below the box. After a method is selected and you click OK, you specify the allocations for each set in the Make Validation Column report. See "Specify Rates or Relative Rates" and "Set Cutpoints". There are five methods for constructing the holdback sets.

Random Validation Column The default method if there are no column assignments in the launch window. This method partitions the data into sets based on the allocations entered in the Make Validation Column report.

Stratified Validation Column The selected method if one or more stratification columns are assigned. This method partitions the data into balanced sets based on the levels of the specified stratification columns. As in the Random Validation Column method, rows are randomly assigned to the holdback sets based on the allocations entered in the Make Validation Column report. However, this is done at each level or combination of levels of the stratifying columns. Use this method when you want a balanced representation of the levels of a column in each of the training, validation, and test sets.

Grouped Validation Column The selected method if one or more grouping columns are specified. This method partitions the data into sets in such a way that entire levels of a specified column or combinations of levels of two or more columns are placed in the same set. Because of this, the sizes of the resulting sets vary slightly from the sizes that you specified. Use this option when splitting levels across holdback sets is not desirable.

Stratify by Group Validation Column The selected method if both stratification and grouping columns are specified. This method partitions the data to balance the levels across the stratification column while requiring that the specified groups stay together in the same holdback sets. As in Grouped Validation Column, groups can be created as levels of a specified column or combination of levels of two or more columns. The sizes of the resulting sets vary slightly from the sizes you specified.

Cutpoint Validation Column The selected method if a cutpoint column is specified. This method partitions the data into sets based on the time series cutpoints. Use this option when you want to assign your data to holdback sets based on time periods. The training set consists of rows between the first cutpoint and the second cutpoint. The validation set consists of rows between the second and third cutpoints. The test set consists of the remaining rows. These sets are chosen based on options in the Set Cutpoints report.

Make Autovalidation Table Creates a new data table that contains a duplication of the rows in the original data table concatenated to the rows in the original data table. The new data table, which can be used for crossvalidation, has four additional columns:

Valid Set Assigns a value of 0 to the original data and a value of 1 to the duplicated data. The values in this column designate the training and validation sets. Use this column in the Validation role in the launch window of an analysis.

Valid ID Assigns the row number of the original observation. This allows matching of training and validation set rows for each original observation.

Valid Weight Assigns the autovalidation weight, to be used in the Freq role in the launch window of an analysis. For each value of Valid ID, the same uniform random number is generated for the training observation and the validation observation. For the training set, Valid Weight is calculated as follows:

Valid Weight = -log(1 - Valid Uniform)

For the validation set, Valid Weight is calculated as follows:

Valid Weight = -log(Valid Uniform)

The Valid Weight column is constructed so that the training data weights are negatively correlated with the validation data weights. This ensures that the difference in the fit of the validation data yields an effective crossvalidation of the fitting method.

Null Factor Assigns the same normal random number for each value of Valid ID.

Tip: Use Make Autovalidation Table for small data tables, where using a subset as the training data could cause estimation problems.

JMP PRO Make Validation Column Report

When you click OK in the launch window, the Make Validation Column report window appears. The report contains a description of the validation method you selected and relevant options for rates, cutpoints, column type, and setting a random seed.

JMP PRO Specify Rates or Relative Rates

This report enables you to specify the allocations for the training, validation, and test sets. In the boxes next to Training Set, Validation Set, and Test Set, enter values that represent the proportions or numbers of rows that you would like to include in each of these sets. The default values construct a training set that contains about 75% of the rows and a validation set that contains about 25% of the rows.

Depending on the number of rows in the data table and the selected validation method, the actual sizes of the resulting sets might vary slightly from the sizes that you specified. The Adjusted Rates, Row Counts, and Group Count columns provide the actual sizes of the sets.

JMP PRO Set Cutpoints

This report is shown only when a cutpoint column is specified in the launch window. The cutpoints are determined by one of the following four methods:

Proportions Determines the cutpoints based on the proportions of rows specified for each set. In the boxes next to Training Set, Validation Set, and Test Set, enter values that represent the proportions that you would like to include in each of these sets. Depending on the number of rows and the proportions, the actual sizes of the resulting sets might vary slightly from the sizes you specified.

Number of Rows Determines the cutpoints based on the number of rows specified for each set. In the boxes next to Training Set, Validation Set, and Test Set, enter values that represent the number of rows that you would like to include in each of these sets. This option enables you to specify the sets exactly.

Fixed Time or Date Determines the cutpoints based on fixed data points in the specified cutpoint column. If you select this option, the minimum and maximum values for the cutpoint column are shown for reference. In the boxes next to Training Set, Validation Set, and Test Set, enter the value that represents the minimum value that you would like to include in each of these sets.

Elapsed Time Determines the cutpoints based on the amount of time that has elapsed since the first timestamp in the cutpoint column. If you select this option, the total elapsed time from the first value to the last value is shown for reference. In the boxes next to Training

Set, Validation Set, and Test Set, enter the values that represent the amounts of elapsed time that you would like to include in each of these sets.

JMP PRO Options

You can also specify the following additional options:

New Column Name Enables you to specify the name of the validation column.

Validation Column Type (Not available for the Cutpoint Validation Column method.) Enables you to specify the validation column type as fixed or formula.

- The fixed column type produces a column with fixed values.
- The formula column type produces a formula column as the validation column.

Random Seed (Not available for the Cutpoint Validation Column method or if a formula column type is specified.) Enables you to set a random seed to reproduce the same validation column in the future.

JMP PRO Go

Click **Go** in the Make Validation Column report to add the validation column to the data table. The validation column has a Notes property that gives the stratifying, grouping, or cutpoint variables.

JMP PRO Make Validation Column Platform Options

The Make Validation Column Utility has the following red triangle menu options:

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

JMP PRO Additional Example of the Make Validation Column Platform

This example uses weekly weather data collected over one year from 16 weather stations across the United States. Run the Weather Station Locations script in the data table to view a map of the locations. Not every weather station has a weekly temperature measurement for every week of the year. You are interested in creating a validation column for this data table based on the dates of data collection. For each weather station, you want to use the first 60% of observations for training, the next 25% of observations for validation, and the final 15% of observations for testing. This example shows the importance of using a Batch ID column in this type of scenario.

Create Validation Column with Cutpoints

1. Select **Help > Sample Data Library** and open Functional Data/Weekly Weather Data.jmp.
2. Select **Analyze > Predictive Modeling > Make Validation Column**.
3. Select Week of Year and click **Cutpoint Column**.
4. Click **OK**.
5. In the list next to Determine cutpoints using, select **Proportions**.
6. In the boxes next to Training Set, Validation Set, and Test Set, enter 0.60, 0.25, and 0.15, respectively.
7. In the box next to New Column Name, type Cutpoint Validation.
8. Click **Go**.

   A validation column called Cutpoint Validation is added to the data table.
9. Select **Analyze > Tabulate**.
10. Click ID and drag it to the Drop zone for rows.
11. Click Cutpoint Validation and drag it on top of N.
12. Click Row% and drag it on top of the cells.

Figure 12.4 Cutpoint Validation Column Proportions

	Cutpoint Validation		
	Training	Validation	Test
ID	Row %	Row %	Row %
Bristol, TN	62.79%	20.93%	16.28%
Corpus Christi, TX	70.27%	18.92%	10.81%
Denver, CO	58.14%	23.26%	18.60%
Goodwell, OK	64.71%	17.65%	17.65%
Grand Canyon National Park, AZ	53.66%	26.83%	19.51%
Greenville, ME	60.47%	20.93%	18.60%
Harrisburg, PA	60.53%	26.32%	13.16%
Helena, MT	60.00%	25.00%	15.00%
Lynchburg, VA	71.79%	17.95%	10.26%
Madera, CA	67.50%	22.50%	10.00%
Miami Beach, FL	63.16%	26.32%	10.53%
Minneapolis/St. Paul, MN	57.50%	27.50%	15.00%
N. Myrtle Beach, SC	69.05%	16.67%	14.29%
Peason Ridge Range, LA	57.89%	28.95%	13.16%
Philip, SD	59.52%	23.81%	16.67%
Terra Haute, IN	47.22%	33.33%	19.44%

Figure 12.4 shows that not all of the weather stations have the correct proportions for the training, validation, and test sets. Use a Batch ID column to obtain the correct percentages.

Create Validation Column with Cutpoints and Batch ID

1. Select **Analyze > Predictive Modeling > Make Validation Column**.
2. Select Week of Year and click **Cutpoint Column**.
3. Select ID and click **Cutpoint Batch ID**.
4. Click **OK**.
5. In the list next to Determine cutpoints using, select **Proportions**.
6. In the boxes next to Training Set, Validation Set, and Test Set, enter 0.60, 0.25, and 0.15, respectively.
7. In the box next to New Column Name, type Cutpoint Batch Validation.
8. Click **Go**.

   A validation column called Cutpoint Batch Validation is added to the data table.
9. Select **Analyze > Tabulate**.
10. Click ID and drag it to the Drop zone for rows.
11. Click Cutpoint Batch Validation and drag it on top of N.
12. Click Row% and drag it on top of the cells.

Figure 12.5 Cutpoint Validation Column with Batch ID Proportions

	Cutpoint Batch Validation		
	Training	Validation	Test
ID	Row %	Row %	Row %
Bristol, TN	60.47%	25.58%	13.95%
Corpus Christi, TX	59.46%	24.32%	16.22%
Denver, CO	60.47%	25.58%	13.95%
Goodwell, OK	58.82%	26.47%	14.71%
Grand Canyon National Park, AZ	60.98%	24.39%	14.63%
Greenville, ME	60.47%	25.58%	13.95%
Harrisburg, PA	60.53%	26.32%	13.16%
Helena, MT	60.00%	25.00%	15.00%
Lynchburg, VA	58.97%	25.64%	15.38%
Madera, CA	60.00%	25.00%	15.00%
Miami Beach, FL	60.53%	26.32%	13.16%
Minneapolis/St. Paul, MN	60.00%	25.00%	15.00%
N. Myrtle Beach, SC	59.52%	26.19%	14.29%
Peason Ridge Range, LA	60.53%	26.32%	13.16%
Philip, SD	59.52%	26.19%	14.29%
Terra Haute, IN	61.11%	25.00%	13.89%

Figure 12.5 shows that using a Cutpoint Batch ID column ensures that each weather station has proportions for the training, validation, and test sets that are much closer to the specified values.

Chapter 13

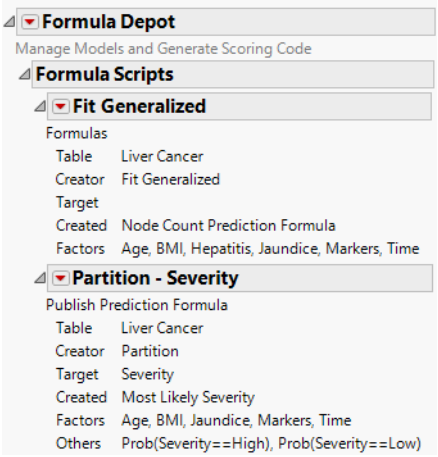
JMP PRO Formula Depot

Manage Models and Generate Scoring Code

The Formula Depot Platform is available only in JMP Pro.

The Formula Depot is a repository to organize, compare, and profile models. Scoring code for deployment within or outside of JMP can be generated for models published to the Formula Depot. For model exploration work, you can use the Formula Depot to store candidate models outside of your JMP data table. The model profiler and model compare platforms are accessible from the Formula Depot. A model that is selected for further use can be saved to your JMP table or saved to a JMP table with new data for scoring. For use in an environment outside of JMP, you can generate scoring code in C, Python, JavaScript, SAS, or SQL.

Figure 13.1 Example of the Formula Depot



Contents

Overview of the Formula Depot Platform 213
Example of Formula Depot 213
Launch the Formula Depot Platform 214
Platforms That Publish Prediction Formulas to the Formula Depot 215
Formula Depot Platform Options 215
Formula Depot Model Options 217
Generating Scoring Code from the Formula Depot Platform 217

JMP PRO Overview of the Formula Depot Platform

The Formula Depot is a repository to organize, compare, profile, and score models for deployment. Models are prediction formulas. Prediction formulas are saved to the Formula Depot as column scripts. You can add prediction formulas to JMP tables to score data. You can also use the Formula Depot to generate scoring code for prediction formulas to facilitate the deployment of models in environments outside of JMP.

The Formula Depot enables you to perform the following tasks:

- save prediction formulas outside of data tables
- save prediction formulas from multiple data tables in a common location
- save intermediate cleaning formulas within a prediction formula
- compare models
- profile models
- add prediction formulas to a data table for scoring of new data within JMP
- generate scoring code (C, Python, JavaScript, SAS, or SQL) for deploying models outside of JMP
- The Formula Depot includes intermediate formulas in the generated scoring code. A model constructed from inputs with column formulas includes the corresponding formulas in the scoring code. For example, if your model includes a recoded variable the recode formula is included in the scoring code. This enables you to apply your prediction formula to raw data that has not been processed.

Note: Informative missing is not a supported intermediate formula.

JMP PRO Example of Formula Depot

The Liver Cancer.jmp sample data table contains data on the severity of liver cancer in patients when they entered a study. The file also contains a number of table scripts for models. This example uses these scripts to generate models to demonstrate the Formula Depot.

1. Select **Help > Sample Data Library** and open Liver Cancer.jmp.
2. Click the green triangle next to the Lasso Poisson, Validation Column script.
3. Click the red triangle next to Poisson Adaptive Lasso with Validation Column and select **Save Columns > Publish Prediction Formula**.

   This option opens a Formula Depot that contains the prediction formula for the Fit Generalized model.

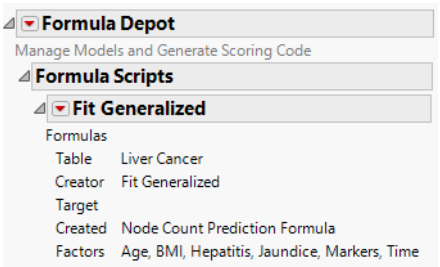
4. To add the prediction formula to the data table, select **Run Script** from the Fit Generalized - Node Count red triangle in the Formula Depot.

Note: The result of step 3 and step 4 can be obtained using the **Save Columns > Save Prediction Formula** at step 3. However, then the prediction formula is not part of the Formula Depot

5. To generate scoring code for use outside of JMP select a code type from the Fit Generalized - Node Count red triangle. A script window appears containing the code. See "Generating Scoring Code from the Formula Depot Platform".
6. To save the Formula Depot, select **File > Save**.

Note: The Formula Depot is saved as a JMP Report File (*.jrp).

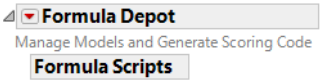
Figure 13.2 Formula Depot with a Generalized Model



JMP PRO Launch the Formula Depot Platform

Launch the Formula Depot by selecting **Analyze > Predictive Modeling > Formula Depot**.

Figure 13.3 Empty Formula Depot from Launch



Alternatively, if there is not an open Formula Depot then a Formula Depot opens when you select a Publish command.

JMP PRO Platforms That Publish Prediction Formulas to the Formula Depot

The platforms that publish prediction formulas and generate scoring code include:

- Discriminant
- Least Squares Regression
- Logistic Regression
- Partition
- Uplift
- K Nearest Neighbors
- Naive Bayes
- Neural
- Latent Class Analysis
- Principal Components
- Generalized Regression
- PLS
- Gaussian Process

In platforms that do not publish prediction formulas to the Formula Depot, you can save the prediction formula to the data table. From the data table, add it to the Formula Depot by selecting **Add Formula from Column**. However, the scoring code might not be fully functional for such models. For more information about scoring code see "Generating Scoring Code from the Formula Depot Platform".

In addition, you can publish a formula directly from the formula editor. Select the formula, right-click, and select **Publish expression to formula depot**.

JMP PRO Formula Depot Platform Options

Add Formula From Column Enables you to add existing prediction formulas from data table columns to the current formula depot. If your table has only one formula, that formula is added to the depot. If you have multiple column formulas, a dialog box enables you to select the formulas of interest. If your formula columns are selected in your data table, they will be pre-selected in the dialog box.

Show Scripts Opens a new Formula Window (or appends to an open Formula Window) that contains scripts for all of the formulas that are in the current formula depot.

Copy Scripts Copies all scripts from the current formula depot to the clipboard.

Copy Formulas as Functions Enables you to select models from the current formula depot to be copied as functions for use in JSL. Selected models are copied onto the clipboard. The functions act on scalar variables rather than columns. One Namespace is generated for each selected model. Each Namespace has one function, called predict, that takes the model inputs as parameters and returns the model predictor.

Copy Formulas as Transforms Enables you to select models from the current formula depot to be copied. Selected models are copied onto the clipboard within a `Transform Columns()` statement.

Run Scripts Enables you to save models from the current formula depot to new columns in your JMP data table. Formulas with intermediate column formulas will add the intermediate columns to your JMP data table.

Note: If you have more than one data table open, a Choose Table window appears that enables you to specify which data table to use for Run Scripts.

Generate C Code, Generate Python Code, Generate JavaScript Code, Generate SAS Code, and Generate SQL Code Enables you to select models from the current formula depot for code generation. A new script window appears that contains scoring code for the selected models in C, Python, JavaScript, SAS DS2, or SQL, respectively. You can use this code to facilitate the deployment of the model in the environment or framework of your choice. See "Generating Scoring Code from the Formula Depot Platform".

Model Comparison Enables you to select models from the current formula depot to be compared using the model comparison utility. If models from multiple tables are stored in the depot, first select the table of interest and then the models of interest. Hold the shift key when selecting the Model Comparison option to launch the Model Comparison in a new window. See "Model Comparison".

Note: If you have more than one data table open, a Choose Table window appears that enables you to specify which data table to use for the Model Comparison.

Remove Model Comparison Removes all Model Comparison reports from the current formula depot.

Profiler Enables you to select models from the current formula depot to be profiled using the profiler. If models from multiple tables are stored in the depot you first select the table of interest and then the models of interest. Hold the shift key when selecting the profiler option to launch the profiler in a new window. See *Profilers*.

Note: If you have more than one data table open, a Choose Table window appears that enables you to specify which data table to use for the Profiler.

Remove Profiler Removes all Profilers from the current formula depot.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

JMP PRO Formula Depot Model Options

Each prediction formula that is saved in the Formula Depot has an individual options menu. In addition to Show, Copy, Run, and Code Generation options that correspond to the main menu options, the individual menus include the following options:

Rename New Column Enables you to rename the model. This new name is applied as the column name if you run the script to add the prediction formula to a JMP data table. In addition, this name is included in generated code.

Remove Removes the model from the Formula Depot. This command cannot be undone.

JMP PRO Generating Scoring Code from the Formula Depot Platform

Scoring code generation is intended to facilitate using models built in JMP in a production environment or other framework of your choice. Many platforms publish prediction formulas to the Formula Depot, however; not all prediction formulas generate complete code. Code can be complete, it can be a code fragment, or the code can include unsupported functions that require additional programming for implementation. Error messages often indicate an unsupported function call that requires additional programming in the targeted language. Target language specific libraries might be required for the implementation of more complex functions. These libraries are discussed in the language specific sections below. For example, you can perform the following tasks:

- Deploy your model to SAS Model Manager using the generated SAS code.
- Augment your ETL process with in-database scoring using the generated SQL code.

- Create a node for a data transformation pipeline with an application built with the generated C code.
- Create a Jupyter notebook to show live scoring results using the generated Python code.
- Enable customers to score their own data with a web application that includes the generated JavaScript.

For the C, Python, and JavaScript languages, you must include supporting code such as .h files and utility libraries when deploying or compiling the generated code. These files are available in your JMP installation folder inside the Scoring folder.

Tip: To find the location of installation folder on your machine, use the JSL command `Get Path Variable("$ALL_HOME/Scoring");`.

C Code The C scoring code that is generated must be compiled into a library and then linked into an application. You might use either a static or a dynamic link approach. The files jmp_lib.h, jmp_parms.h, and jmp_score.h needed for compiling and linking can be found in the Scoring/C folder in your JMP installation folder.

Python Code The file jmp_score.py that is needed to run your Python scoring application can be found in the Scoring/Python folder in your JMP installation folder. When deploying your models, copy jmp_score.py to the same location as your executable scoring code.

When calling the generated code to score your data, it is important that the input data is passed in a data structure compatible with the code generated by JMP. The Python scoring code expects the input and output arguments to be of a dictionary-like type. Values can be accessed and written by key, such as DataFrame row objects from the Spark, Dask and Pandas frameworks, or the standard Python "dict" class.

Use of linear algebra operators or advanced operators such as Vec Quadratic and Design Norm require the NumPy Python library.

JavaScript Code The file jmp_score.js that is needed to run your JavaScript scoring application can be found in Scoring/JavaScript folder.

SAS Code The generated code fragment that once wrapped by PROC DS2 statements can be used in SAS applications including the SAS In-Database Code Accelerator. The code is added to a SAS Window and includes a variable name mapping section in the comment block that precedes the code. Any temporary variable names associated with intermediate columns are deleted at the end of the DATA Step by the drop command.

Tip: For models with an ifmax call, such as logistic or neural models for a categorical response, move temporary variable declarations before the **method run()** statement.

SQL Code The SQL code fragment that once wrapped in a select statement can be used in SQL queries against most major database servers.

Note: Code that contains “placeholder” or “ERROR” indicates an unsupported function call.

Chapter 14

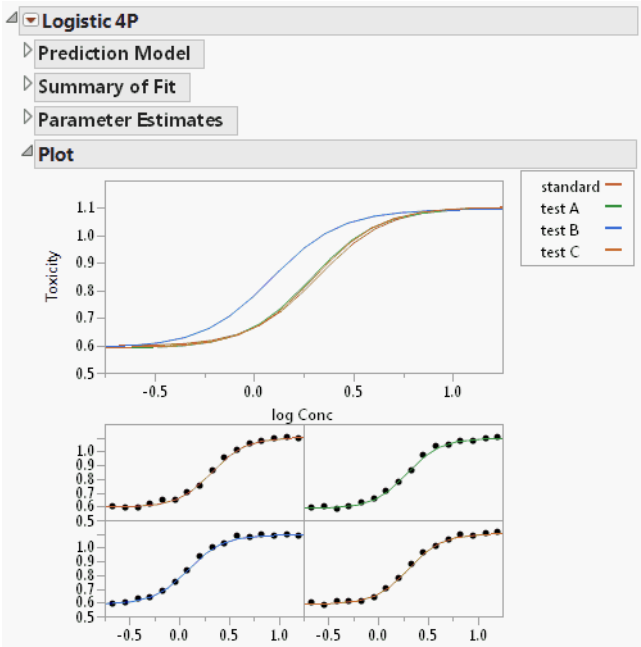
Fit Curve

Fit Built-In Nonlinear Models to Your Data

In many situations, especially in the physical and biological sciences, well-known nonlinear equations describe the relationship between variables. For example, pharmacological bioassay experiments can demonstrate how the strength of the response to a drug changes as a function of drug concentration. Sigmoid curves often accurately model response strength as a function of drug concentration. Another example is exponential growth curves, which can model the size of a population over time.

The Fit Curve platform does not require you to specify starting values for parameter estimates or create model formulas. To specify your own starting values and create model formulas, use the more powerful custom Nonlinear platform, which can also fit any nonlinear model. See "Nonlinear Regression".

Figure 14.1 Example of Nonlinear Fit in the Fit Curve Platform



Contents

Overview of the Fit Curve Platform 223
Example Using the Fit Curve Platform 224
Launch the Fit Curve Platform 227
The Fit Curve Report 228
Fit Curve Options 228
    Model Comparison Report 229
Model Fit Report 231
Model Fit Options 231
    Test Parallelism 233
    Compare Parameter Estimates 236
    Equivalence Test 236
Statistical Details for the Fit Curve Platform 238
    Model Formulas 238

Overview of the Fit Curve Platform

Some models are *linear* in the parameters (for example, a quadratic or other polynomial); others can be transformed to be such (for example, when you use a log transformation of x). The Fit Model or Fit Y by X platforms are more appropriate in these situations. An example in *Fitting Linear Models* shows a significant linear relationship between oxygen uptake and time spent running. For more information about Fit Model, see *Fitting Linear Models*. For more information about Fit Y by X, see *Basic Analysis*.

The Fit Curve platform enables you to fit models that are *nonlinear* in the parameters. The initial example in this chapter shows the analysis of a nonlinear relationship: drug toxicity as a function of concentration. The effect of concentration on toxicity changes from low to high doses, so this relationship is nonlinear.

The following are examples of equations for linear and nonlinear functions.

Linear function: $Y = \beta_0 + \beta_1 e^{x}$

Nonlinear function: $Y = \beta_0 + \beta_1 e^{\beta_2 x}$

The Fit Curve platform provides predefined models, such as polynomial, logistic, probit, Gompertz, exponential, peak, and pharmacokinetic models. Specifying a grouping variable lets you estimate separate model parameters for each level of the grouping variable. The fitted models and estimated parameters can be compared across the levels of the grouping variable.

Fit Curve also enables you to build a model to create the prediction formula. Then you set upper and lower parameter limits in Nonlinear. See "Example of Setting Parameter Limits". In JMP Pro, you can also specify a set of supplementary variables and fit a generalized regression model within the Fit Curve platform to determine how these variables affect the response.

Example Using the Fit Curve Platform

This example shows how to build a model for toxicity as a function of the concentration of a drug. You have a standard formulation of the drug and want to compare it to three new formulations.

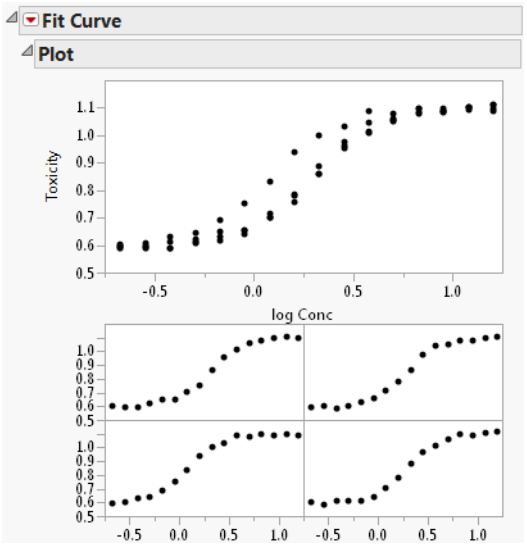
You are interested in a toxicity ratio of non-surviving to surviving cells at a specific concentration of each drug. From prior research, you know the toxicity ratios for 16 different concentrations of each drug formulation. A higher ratio indicates more toxicity, which could be detrimental to development of the drug. Log concentration was calculated to decrease the range of concentration values and make it easier to detect differences in the curves.

Follow these steps to build the model:

1. Select **Help > Sample Data Library** and open Nonlinear Examples/Bioassay.jmp.
2. Select **Analyze > Specialized Modeling > Fit Curve**.
3. Assign Toxicity to the **Y, Response** role.
4. Assign log Conc to the **X, Regressor** role.
5. Assign Formulation to the **Group** role.
6. Click **OK**.

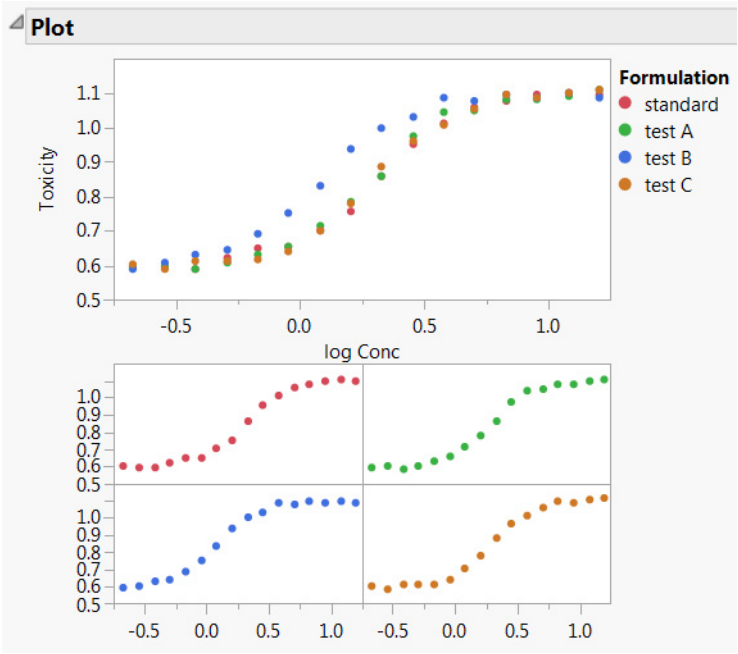
   The Fit Curve Report appears. The Plot report contains an overlaid plot of the fitted model of each formulation.

Figure 14.2 Initial Fit Curve Report



7. To see a legend identifying each drug formulation, right-click one of the graphs and select **Row Legend**. Select **Formulation** for the column and click **OK**.

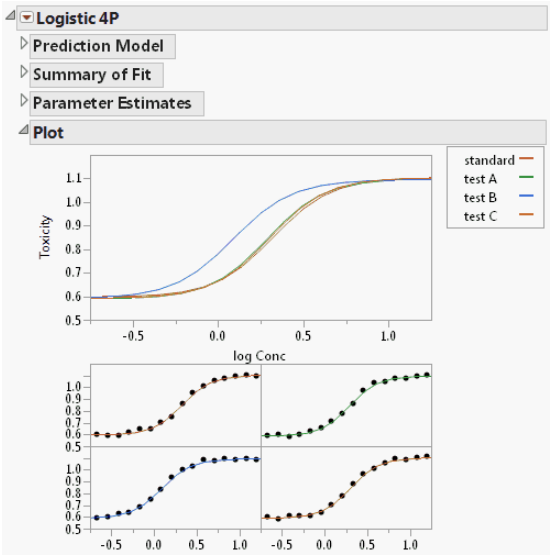
Figure 14.3 Fit Curve Report with Plot Legend



The curves appear S-shaped, so a sigmoid curve would be an appropriate fit. Table 14.1 shows formulas and graphical depictions of the different types of models that the Fit Curve platform offers.

8. Click the Fit Curve red triangle and select **Sigmoid Curves > Logistic Curves > Fit Logistic 4P**.

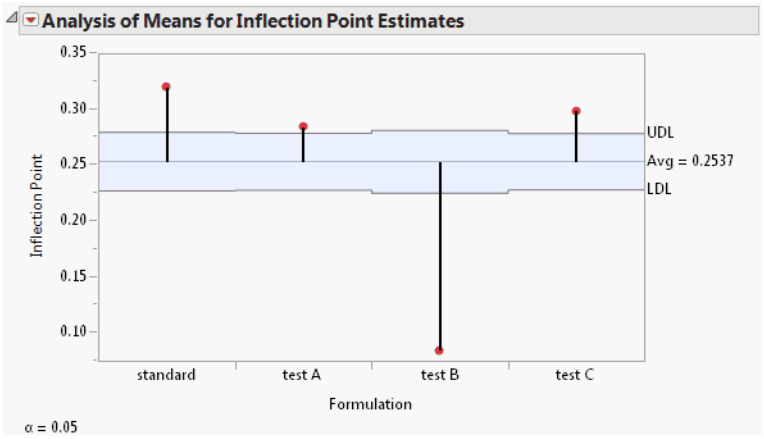
Figure 14.4 Logistic 4P Report



The Logistic 4P report appears. There is also a separate plot for each drug formulation. The plot of the fitted curves suggests that formulation B might be different, because the test B curve starts to rise sooner than the others. Inflection point parameters cause this rise.

9. Click the Logistic 4P red triangle and select **Compare Parameter Estimates**.

Figure 14.5 Parameter Comparison Report

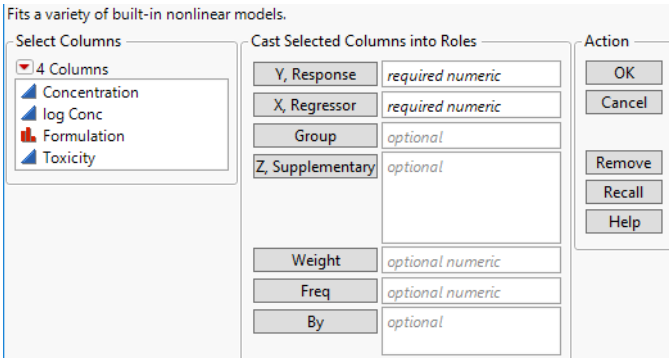


Notice that the Inflection Point parameter for the test B formulation is significantly lower than the average inflection point. This agrees with the plots shown in Figure 14.4. Drug formulation B has a higher toxicity ratio than the other formulations.

Launch the Fit Curve Platform

To launch the Fit Curve platform, select **Analyze > Specialized Modeling > Fit Curve**. The launch window is shown in Figure 14.6.

Figure 14.6 Fit Curve Platform Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Fit Curve platform launch window has the following features:

Y, Response Specify the *Y* variable.

X, Regressor Specify the *X* variable.

Group Specify a grouping variable. The fitted model has separate parameters for each level of the grouping variable. This enables you to compare fitted models and estimated parameters across the levels of the grouping variable.

Z, Supplementary Specify one or more supplementary variables. Supplementary variables are not used in any of the calculations in the Fit Curve platform and including them does not affect the results. Supplementary variables are variables you might want to use in future analyses of the results from Fit Curve. When you specify supplementary variables, they are included in the table that is created by the Make Parameter Table option. In JMP Pro, if you specify one or more supplementary variables and a grouping variable, a Curve DOE Analysis option is made available for the fitted models. See "Curve DOE Analysis".

Weight Specify a variable that contains the weights of the observations.

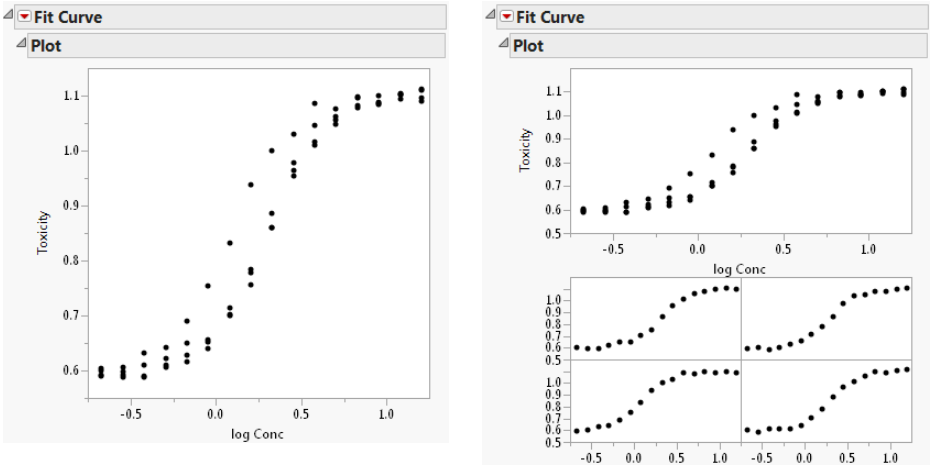
Freq Specify a variable that contains the frequencies of the observations.

By Specify a variable to perform a separate analysis for every level of the variable.

The Fit Curve Report

The Fit Curve report initially contains only a plot of Y versus X. If you specify a Group variable, the report includes overlaid and individual plots for each group of the fitted model.

Figure 14.7 Fit Curve Reports: No Grouping Variable (left) and with Group Variable (right)



After fitting a model, the fitted model appears on the plot (when no grouping variable is specified on the platform launch window). A Model Comparison report appears above the plots and a Model Report appears below the plot. Each time a new model is fit, it appears in the Model Comparison report and a new model report appears in the report window. See "Model Comparison Report" and "Model Fit Report".

Note: Any rows that are excluded in the data table are also hidden in the Fit Curve plot.

Fit Curve Options

Select any of the following built-in models from the Fit Curve red triangle menu. See "Model Formulas".

Polynomials Fits first degree to fifth degree polynomials.

Sigmoid Curves Fits Logistic, Probit, Gompertz, and Weibull models. These models are S-shaped and have both upper and lower asymptotes. The Logistic 2P, 3P, and 4P and Probit 2P and 4P models are symmetric. The Logistic 5P and both Gompertz models are not symmetric. The Logistic 2P is available only when the response is between 0 and 1. The Weibull Growth is available only when both the response values and regressor values are

non-negative. Examples of Sigmoid curves include learning curves and modeling tumor growth, both of which increase initially and then taper off.

Exponential Growth and Decay Fits Exponential, Biexponential, Mechanistic Growth, and Cell Growth models. The Exponential 2P and 3P are similar, but the 3P model has an asymptote. The Biexponential models assume there are two separate growth or decay processes. The Mechanistic Growth and Exponential 3P models always increase (or decrease), but the rate of growth (or decay) slows so that the model has an asymptote. Examples of exponential growth and decay functions are virus spread and drug half-life, respectively.

Peak Models Fits Gaussian Peak and Lorentzian Peak models. These models increase up to a peak and then decrease. The Gaussian Peak model is a scaled version of the Gaussian probability density function (PDF). The Lorentzian Peak model is a scaled version of the Cauchy distribution, a continuous probability distribution. These models can be used for some chemical concentration assays and artificial neural networks.

Pharmacokinetic Models Fits the One Compartment Oral Dose model, the Two Compartment IV Bolus Dose model, and the Biexponential 4P model. This option is used to model the concentration of drugs in the body.

Fit Michaelis-Menten Fits the Michaelis-Menten biochemical kinetics model, which relates the rate of enzymatic reactions to substrate concentration.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

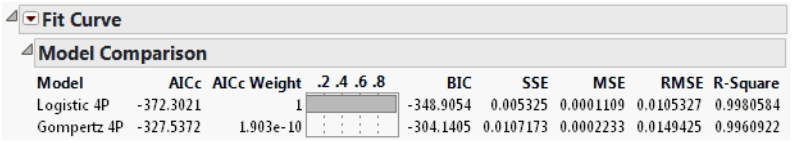
Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Model Comparison Report

To create the report shown in Figure 14.8, select **Sigmoid Curves > Logistic Curves > Fit Logistic 4P** and **Sigmoid Curves > Fit Gompertz 4P** from the Fit Curve red triangle menu.

Figure 14.8 Model Comparison Report


Fit Curve

Model Comparison

Model	AICc	AICc Weight	.2 .4 .6 .8	BIC	SSE	MSE	RMSE	R-Square
Logistic 4P	-372.3021	1		-348.9054	0.005325	0.0001109	0.0105327	0.9980584
Gompertz 4P	-327.5372	1.903e-10		-304.1405	0.0107173	0.0002233	0.0149425	0.9960922

The Model Comparison report shows fit statistics used for comparing multiple models. The statistics are AICc, AICc Weight, BIC, SSE, MSE, RMSE, and R-Square, and are defined below.

AICc Gives a measure of the goodness of fit of an estimated statistical model that can be used to compare two or more models. AICc is a modification of the AIC adjusted for small samples. AICc can be computed only when the number of data points is at least two greater than the number of parameters. The model with the lowest AICc value is the best, which is the Logistic 4P in our example. See *Fitting Linear Models*.

AICc Weight Gives normalized AICc values that sum to one. The AICc weight can be interpreted as the probability that a particular model is the true model given that one of the fitted models is the truth. Therefore, the model with the AICc weight closest to one is the best fit. In our example, the Logistic 4P model is clearly the better fit. The AICc weights are calculated using only nonmissing AICc values:

AICcWeight = exp[-0.5(AICc-min(AICc))] / sum(exp[-0.5(AICc-min(AICc))])

where min(AICc) is the smallest AICc value among the fitted models. The AICc Weight column is then sorted in decreasing order.

BIC Gives a measure based on the likelihood function of model fit that is helpful when comparing different models. The model with the lower BIC value is the better fit. See *Fitting Linear Models*.

SSE The sum of the squared differences between each observation and its predicted value.

MSE Gives the average of the squares of the errors of each value.

RMSE The square root of the MSE that estimates the standard deviation of the random error.

R-Square Estimates the proportion of variation in the response that can be attributed to the model rather than to random error. The model with the R-Square value closest to one is the better fit.

The Model Comparison platform provides additional options, such as plotting residual and actual values. See "Model Comparison".

Model Fit Report

A report is created for each fitted model. The red triangle menu for each model report contains the following options.

Prediction Model Gives the algebraic form of the prediction formula and the parameters.

Summary of Fit Gives the same fit statistics as the Model Comparison report.

Parameter Estimates Gives the estimates of the parameters, standard errors, Chi-squared statistics, *p*-values, and confidence intervals. The correlations and covariances of the estimates are given also.

Tip: You can sort the Parameter Estimates report table by clicking on the column that you want to sort by.

Plot Gives plots of the data with the fitted model (Figure 14.7). The plots are shown only when you select a Grouping variable on the platform launch window.

Model Fit Options

Each model report contains a red triangle menu with some or all of the following options:

Test Parallelism Helps determine whether the curves are similar in shape when they are shifted along the X axis. In certain situations, it is important to establish parallelism before making further comparisons between groups. This option is available only when a Group variable is specified on the platform launch window. This option is available for the Sigmoid models (Logistic and Gompertz), as well as the Linear Regression model, with the exception of higher-order polynomials. See "Test Parallelism".

Area Under Curve Gives the area under the fitted curve. This option is available for the following models: One Compartment, Two Compartment, Gaussian Peak, and Lorentzian Peak. This option is also available for Bi-Exponential 4P Models, but only when all parameters are positive. The range of integration depends on the type of model and is specified in the report.

If a Grouping variable is specified on the platform launch window, an Analysis of Means is performed for comparing the estimates across groups. If the result for a group exceeds a decision limit, the result is considered different from the overall mean of AUC.

Time to Peak Response Displays the estimate of the regressor, *X*, at the peak of the fitted curve. The standard error of the estimate is also shown. This option is available for Cell Growth 4P and One Compartment models.

Peak Response Displays the estimate of the response, *Y*, at the peak of the fitted curve. The standard error of the estimate is also shown. This option is available for Cell Growth 4P and One Compartment models.

Compare Parameter Estimates Gives an analysis for testing the equality of parameters across levels of the grouping variable. This option is available only when a Group variable is specified on the platform launch window. See "Compare Parameter Estimates".

Equivalence Test Gives an analysis for testing the equivalence of models across levels of the grouping variable. This option is available only when a Group variable is specified on the platform launch window. See "Equivalence Test".

JMP PRO **Curve DOE Analysis** (Available only if a grouping variable and at least one supplementary variable are specified in the launch window.) Launches a Generalized Regression report within the Fit Curve platform. A generalized regression model is fit to each parameter of the nonlinear model using the supplementary variables as model effects. By default, a two degree factorial model is fit and the Estimation Method is Forward Selection. These models are then combined to create a profiler of the response as a function of the regressor variable and the supplementary variables. You can then use the CDOE Profiler to explore how the supplementary variables affect the response.

The Curve DOE Analysis report contains the following red triangle menu options:

Generalized Regression for Model Parameters Shows or hides the Generalized Regression reports for each model parameter. For more information on Generalized Regression model reports, see *Fitting Linear Models*.

Diagnostic Plots Shows or hides actual by predicted and residual plots for the response variable.

CDOE Profiler Shows or hides the CDOE Profiler, which enables you to explore how the response changes based on the supplementary variables. For more information about the CDOE Profiler red triangle menu options, see *Profilers*.

Save Prediction Formula Saves the prediction formula for the response to a new column in the data table.

Make Parameter Table Saves the parameter estimates, standard errors, and t-ratios in a data table. This option is available only when a Group variable is specified on the platform launch window.

Plot Actual by Predicted Plots actual Y values on the vertical axis and predicted Y values on the horizontal axis.

Plot Residual by Predicted Plots the residuals on the vertical axis and the predicted Y values on the horizontal axis.

Profiler Shows or hides a profiler of the fitted prediction function. The derivatives are derivatives of the prediction function with respect to the *X* variable. For more information about profilers, see *Profilers*.

Save Formulas Contains options for saving a variety of formula columns in the data table.

Save Prediction Formula Saves the prediction equation.

Save Std Error of Predicted Saves the standard error of the predicted values.

Save Parametric Prediction Formula Saves the prediction equation in parametric form. This is helpful if you want to use the fitted model in the custom Nonlinear platform.

Save Residual Formula Saves the residuals.

Save Studentized Residual Formula Saves the studentized residual formula, a standard residual that is divided by its estimated standard deviation.

Save First Derivative Saves the derivative of the prediction function with respect to the *X* variable.

Save Std Error of First Derivative Saves the equation of the standard error of the first derivative.

Save Inverse Prediction Formula Saves the equation for predicting *X* from *Y*.

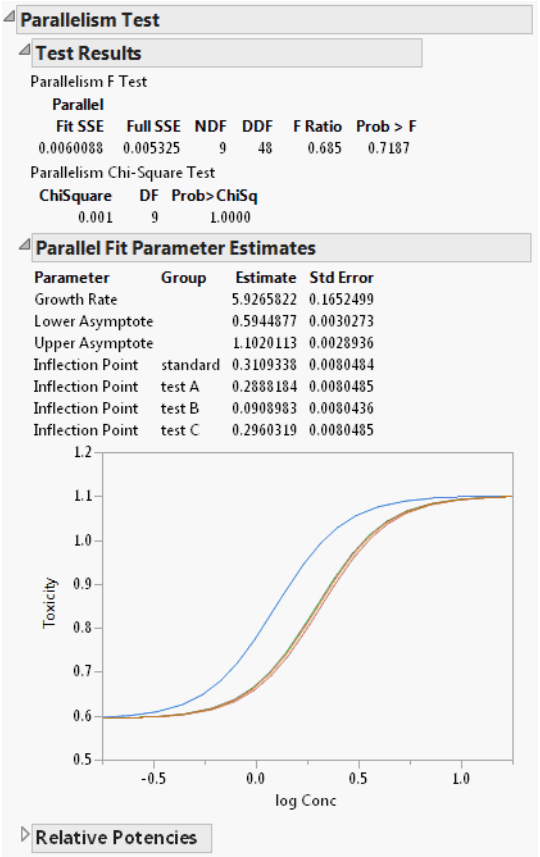
Custom Inverse Prediction Predicts an *X* value for a specific *Y* value. For more information about inverse prediction, see *Fitting Linear Models*.

Remove Fit Removes the model report, the entry from the Model Comparison report, and the fitted line from the plot.

Test Parallelism

The Test Parallelism option provides an analysis for testing if the fitted models between groups have the same shape, but are shifted along the X axis (Figure 14.9). In the Bioassay example, the curve for drug formulation B is shifted to the left of the other three curves. However, you do not know whether the curves still have the same shape (are parallel), or if formulation B is different. The Parallelism Test tells us if the shapes for the different drug formulations have similar shapes and are shifted along the horizontal axis. Select **Test Parallelism** from the fitted model's red triangle menu to add the report.

Figure 14.9 Parallelism Test



The report gives the following results:

Test Results Gives the results of an *F* Test and a Chi-Square Test for parallelism. The tests compare the error sums-of-squares for a full model (Full SSE) and a reduced model (Fit SSE). The full model gives each group different parameters. The reduced model forces the groups to share every parameter except for the inflection point. A small *p*-value indicates that the group models are significantly different from one another. Therefore, the reduced model with shared parameters is insufficient and the full model should be used. In Figure 14.9, the *p*-value is greater than 0.05, indicating that there is not enough evidence to conclude that differences exist between the curves. In this case, the reduced model with shared parameters is appropriate.

The fit statistics are calculated as follows:

F Ratio [(Fit SSE - Full SSE) / NDF] / [Full SSE / DDF]

ChiSquare Fit SSE - Full SSE

Parallel Fit Parameter Estimates Gives the parameter estimates under the reduced model (same parameters, except for inflection point). A plot of the fitted curves under the reduced model is provided. The inflection point for drug formulation B is much lower than that of the other three drug formulations.

Relative Potencies Gives the potency and relative potency for each level of the grouping variable. The potency is 10^(EC_{50}), where EC_{50} is the concentration at which the response half way between baseline and maximum is obtained. For the Logistic 2P, 3P, and 4P, the potency is 10^(inflection point parameter). The relative potency is the potency of one level of the grouping variable divided by the potency of another level. For example, in Figure 14.10, the relative potency of test A to standard is $\text{Potency}_{\text{test A}}/\text{Potency}_{\text{standard}}$.

Figure 14.10 Relative Potencies by Group

Relative Potencies

Relative Potency versus standard

Group	Potency	Relative Potency	Std Error
standard	2.0461329	1	0
test A	1.9445466	1.0522416	0.0246535
test B	1.2328161	1.6597227	0.0388878
test C	1.977115	1.0349084	0.0242474

Relative Potency versus test A

Group	Potency	Relative Potency	Std Error
standard	2.0461329	0.950352	0.0222663
test A	1.9445466	1	0
test B	1.2328161	1.5773209	0.036957
test C	1.977115	0.9835273	0.0230435

Relative Potency versus test B

Group	Potency	Relative Potency	Std Error
standard	2.0461329	0.6025103	0.014117
test A	1.9445466	0.6339864	0.0148545
test B	1.2328161	1	0
test C	1.977115	0.6235429	0.0146098

Relative Potency versus test C

Group	Potency	Relative Potency	Std Error
standard	2.0461329	0.9662691	0.0226392
test A	1.9445466	1.0167486	0.0238219
test B	1.2328161	1.6037388	0.037576
test C	1.977115	1	0

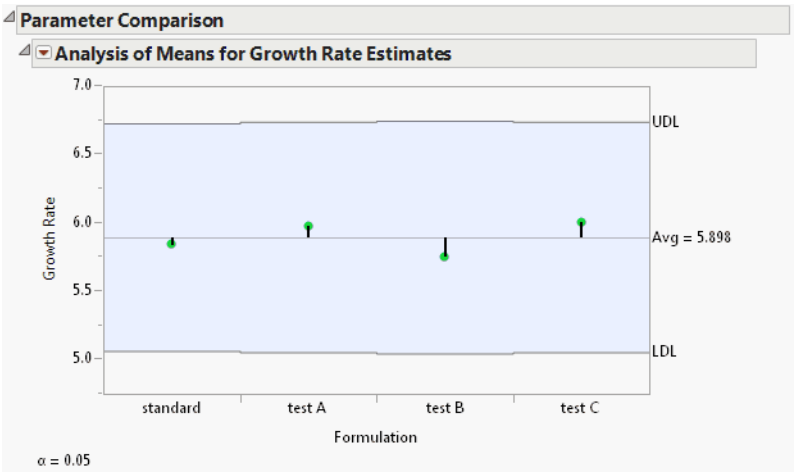
In the **Relative Potency versus standard** panel from Figure 14.10, note that the relative potencies for drug formulations A and C are nearly one. This indicates that their potencies are similar to that of the standard formulation. The potency for drug formulation B is lower than that of the standard. This means that drug formulation B increases in toxicity as a function of concentration faster than the standard.

In the parallelism test, the curves are parallel, which enables you to calculate relative potencies. Based on the relative potencies, you conclude that formulation B is more potent than the other drug formulations. Taken with the prior findings, drug formulation B appears to be more toxic.

Compare Parameter Estimates

The Compare Parameter Estimates report gives results for testing the equality of parameters across the levels of the grouping variable. There is an Analysis of Means (ANOM) report for each parameter, which tests whether the parameters are equal to an overall mean. If the result for a parameter exceeds the decision limits, then the parameter is different from the overall mean. Figure 14.11 shows the ANOM report for growth rate estimates. Select **Compare Parameter Estimates** from the fitted model's red triangle menu to add the report.

Figure 14.11 Parameter Comparison for Growth Rate Estimates



The Analysis of Means red triangle menu contains the following options:

Set Alpha Level Sets the alpha level for the test.

Show Summary Report Shows or hides a report containing the parameter estimates, the decision limits, and whether the parameter exceeded the limits.

Display Options Contains options for showing or hiding decision limits, shading, and the center line. Also contains options for changing the appearance of the points.

For more information about the Analysis of Means report, see *Basic Analysis*.

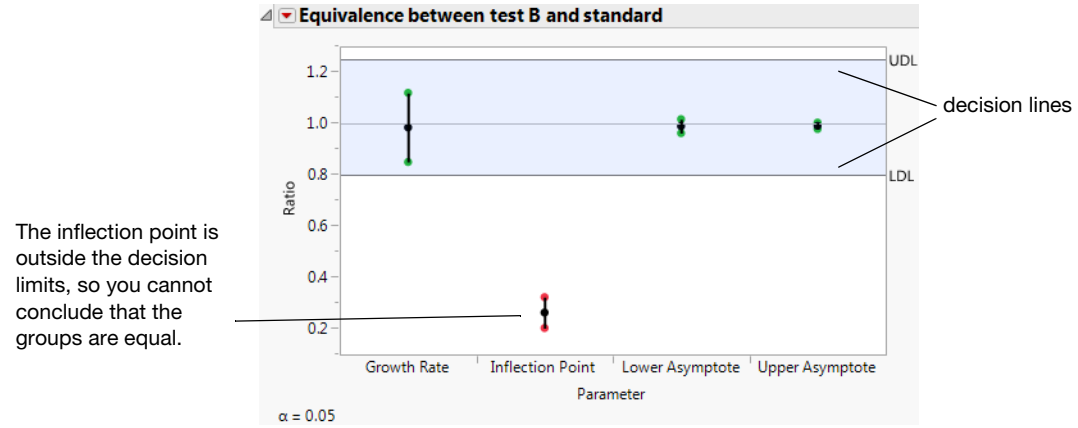
Equivalence Test

The Equivalence Test report gives an analysis for testing the equivalence of models across levels of the grouping variable (Figure 14.12). After selecting the option, you specify the level of the grouping variable that you want to test against every other level. There is a report for every level versus the chosen level. Select **Equivalence Test** from the fitted model's red triangle menu to add the report.

The equality of the parameters is tested by analyzing the ratio of the parameters. The default decision lines are placed at ratio values of 0.8 and 1.25, representing a 25% difference.

If all of the confidence intervals are inside the decision lines, then the two groups are practically equal. If a single interval falls outside the lines (Figure 14.12), then you cannot conclude that the groups are equal. The inflection point for drug formulation B is lower than the standard, which agrees with the previous findings.

Figure 14.12 Equivalence Test



The Equivalence red triangle menu contains the following options:

Set Alpha Level Sets the alpha level for the test. The default value is 0.05.

Set Decision Lines Changes the decision limits for the ratio. The default values are set at 0.8 and 1.25, representing a 25% difference.

Show Summary Report Shows or hides a report containing the parameter estimates, the decision limits, and whether the parameter exceeded the limits.

Display Options Contains options for showing or hiding decision limits, shading, and the center line. Also contains options for changing the appearance of the points. For additional formatting options, right-click the graph and select **Customize**.

Statistical Details for the Fit Curve Platform

Model Formulas

Table 14.1 provides the formulas for the models on the Fit Curve red triangle menu.

Table 14.1 Fit Curve Model Formulas

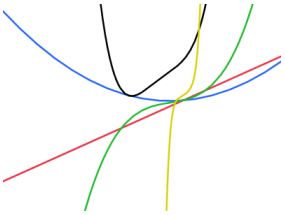
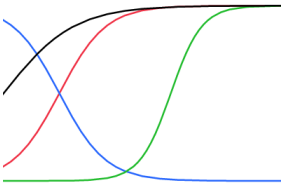
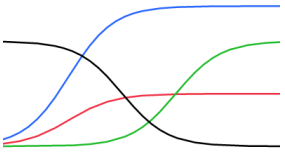
Model	Formula
Polynomials 	$\beta_0 + \sum_{i=1}^{k} \beta_i x^i$ where k is the order of the polynomial. These models can also be fit using the Fit Model and Fit Y by X platforms.
Logistic 2P 	$\frac{1}{1 + \text{Exp}(-a(x-b))}$ a = Growth Rate b = Inflection Point Available only when all response values are between zero and one.
Logistic 3P 	$\frac{c}{1 + \text{Exp}(-a(x-b))}$ a = Growth Rate b = Inflection Point c = Asymptote

Table 14.1 Fit Curve Model Formulas *(Continued)*

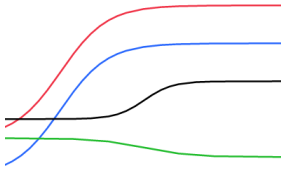
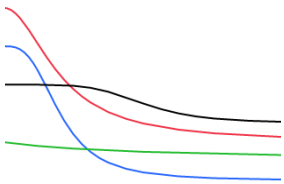
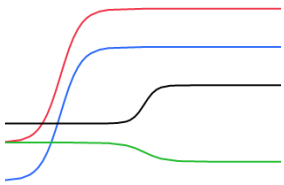
Model	Formula
Logistic 4P 	$c + \frac{d-c}{1+\text{Exp}(-a(x-b))}$ a = Growth Rate b = Inflection Point c = Lower Asymptote d = Upper Asymptote
Logistic 4P Rodbard 	$c + \frac{d-c}{1+(x/b)^a}$ a = Growth Rate b = Inflection Point c = Lower Asymptote d = Upper Asymptote Available only when the regressor values are positive.
Logistic 4P Hill 	$c + \frac{d-c}{1+10^{(-a(x-b))}}$ a = Growth Rate b = Inflection Point c = Lower Asymptote d = Upper Asymptote

Table 14.1 Fit Curve Model Formulas *(Continued)*

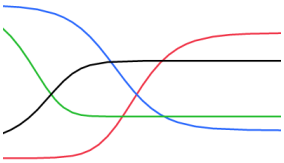
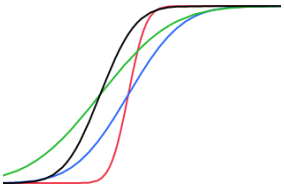
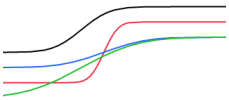
Model	Formula
Logistic 5P 	$c + \frac{d - c}{(1 + \text{Exp}(-a(x - b)))^f}$ a = Growth Rate b = Inflection Point c = Asymptote 1 d = Asymptote 2 f = Power
Probit 2P 	$\Phi\left(\frac{x - b}{a}\right)$ a = Growth Rate b = Inflection Point Φ = Normal Distribution CDF Available only when all response values are between zero and one.
Probit 4P 	$c + (d - c) \cdot \Phi\left(\frac{x - b}{a}\right)$ a = Growth Rate b = Inflection Point c = Asymptote 1 d = Asymptote 2 Φ = Normal Distribution CDF

Table 14.1 Fit Curve Model Formulas *(Continued)*

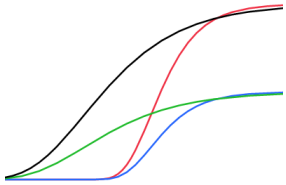
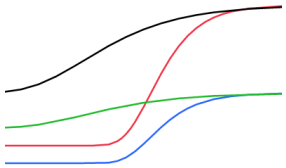
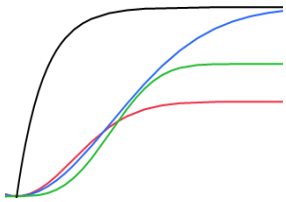
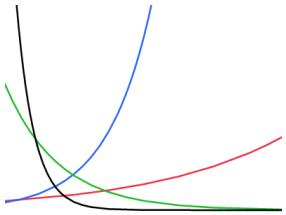
Model	Formula
Gompertz 3P 	$a\text{Exp}(-\text{Exp}(-b(x-c)))$ a = Asymptote b = Growth Rate c = Inflection Point
Gompertz 4P 	$a+(b-a)\text{Exp}(-\text{Exp}(-c(x-d)))$ a = Lower Asymptote b = Upper Asymptote c = Growth Rate d = Inflection Point
Weibull Growth 	$a(1-\text{Exp}(-(x/c)^{b}))$ a = Upper Asymptote b = Growth Rate c = Inflection Point Available only when both the response values and regressor values are non-negative.
Exponential 2P 	$a\text{Exp}(bx)$ a = Scale b = Growth Rate

Table 14.1 Fit Curve Model Formulas *(Continued)*

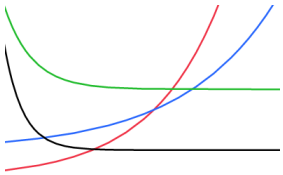
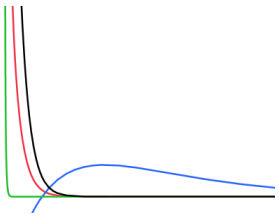
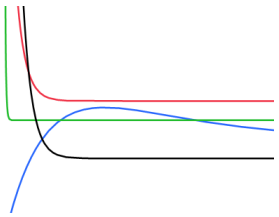
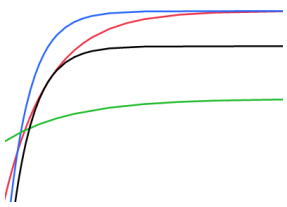
Model	Formula
Exponential 3P 	$a + b\text{Exp}(cx)$ a = Asymptote b = Scale c = Growth Rate
Biexponential 4P 	$a\text{Exp}(-bx) + c\text{Exp}(-dx)$ a = Scale 1 b = Decay Rate 1 c = Scale 2 d = Decay Rate 2 Available only when the response values are positive.
Biexponential 5P 	$a + b\text{Exp}(-cx) + d\text{Exp}(-fx)$ a = Asymptote b = Scale 1 c = Decay Rate 1 d = Scale 2 f = Decay Rate 2
Mechanistic Growth 	$a(1 - b\text{Exp}(-cx))$ a = Asymptote b = Scale c = Growth Rate

Table 14.1 Fit Curve Model Formulas *(Continued)*

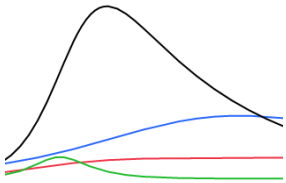
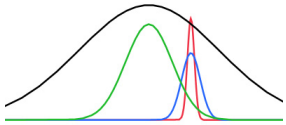
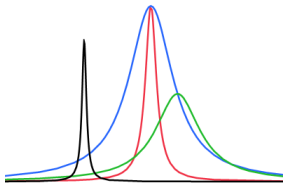
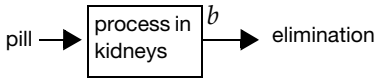
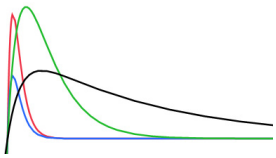
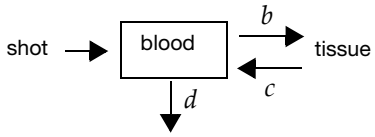
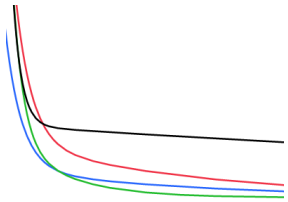
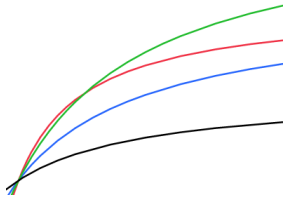
Model	Formula
Cell Growth 4P 	$\frac{ab}{(a-b)\text{Exp}(-cx)+b\text{Exp}(dx)}$ a = Peak value if mortality rate, d, is zero b = Response at time zero c = Cell Division Rate d = Cell Mortality Rate Available only when the response values are positive and the regressor values are non-negative.
Gaussian Peak 	$a\text{Exp}\left(-\frac{1}{2}\left(\frac{x-b}{c}\right)^2\right)$ a = Peak Value b = Critical Point c = Growth Rate
Lorentzian Peak 	$\frac{ab^2}{(x-c)^2+b^2}$ a = Peak Value b = Growth Rate c = Critical Point
One Compartment Oral Dose  	$\frac{abc}{c-b}(\text{Exp}(-bx)-\text{Exp}(-cx))$ a = Area Under Curve b = Elimination Rate c = Absorption Rate Available only when the response values and the regressor values are all positive.

Table 14.1 Fit Curve Model Formulas *(Continued)*

Model	Formula
Two Compartment IV Bolus Dose  	$\frac{a}{\alpha-\beta}((\alpha-b)\mathrm{Exp}(-\alpha x)-(\beta-b)\mathrm{Exp}(-\beta x))$ $\alpha = \frac{1}{2}(b+c+d+\sqrt{(b+c+d)^2-4bd})$ $\beta = \frac{1}{2}(b+c+d-\sqrt{(b+c+d)^2-4bd})$ a = Initial Concentration b = Transfer Rate In c = Transfer Rate Out d = Elimination Rate Available only when the response values and the regressor values are all positive.
Michaelis-Menten 	$\frac{ax}{b+x}$ a = Max Reaction Rate b = Inverse Affinity Available only when the response values and the regressor values are all positive.

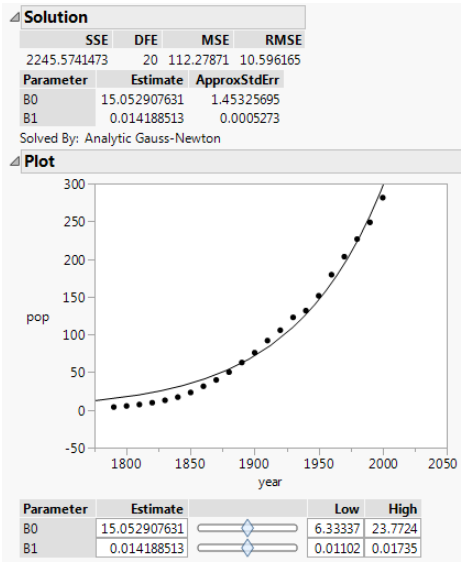
Chapter 15

Nonlinear Regression

Fit Custom Nonlinear Models to Your Data

The Nonlinear platform is a good choice for models that are *nonlinear* in the parameters. This chapter focuses on custom nonlinear models, which include a model formula and parameters to be estimated. Use the default least squares loss function or a custom loss function to fit models. The platform minimizes the sum of the loss function across the observations.

Figure 15.1 Example of a Custom Nonlinear Fit



The Nonlinear platform also provides predefined models, such as polynomial, logistic, Gompertz, exponential, peak, and pharmacokinetic models. See "Fit Curve".

Note: Some models are *linear* in the parameters (for example, a quadratic or other polynomial) or can be transformed to be such (for example, when you use a log transformation of x). The Fit Model or Fit Y by X platforms are more appropriate in these situations. For more information about these platforms, see *Fitting Linear Models* and *Basic Analysis*.

Contents

Example of Fitting a Custom Model 247
Launch the Nonlinear Platform 250
The Nonlinear Fit Report 251
Nonlinear Platform Options 255
Create a Formula Using the Model Library 259
    Customize the Nonlinear Model Library 261
Additional Examples 262
    Example of Maximum Likelihood: Logistic Regression 263
    Example of a Probit Model with Binomial Errors: Numerical Derivatives 264
    Example of a Poisson Loss Function 265
    Example of Setting Parameter Limits 267
Statistical Details for the Nonlinear Platform 270
    Profile Likelihood Confidence Limits 271
    How Custom Loss Functions Work 272
    Notes Concerning Derivatives 273
    Notes on Effective Nonlinear Modeling 274

Example of Fitting a Custom Model

This example fits a custom model by creating a formula column for the model and then fitting the model using the Nonlinear platform. This method requires a few more steps than fitting a built-in model, but it allows any nonlinear model to be fit. You can also provide a custom loss function, and specify several other options for the fitting process.

Tip: If the custom model is one that you use often for several data tables, you can add the model to the Model Library instead. See "Customize the Nonlinear Model Library".

The data are in the US Population.jmp data table. The response variable is the population (in millions) of the Unites States and the predictor is the year.

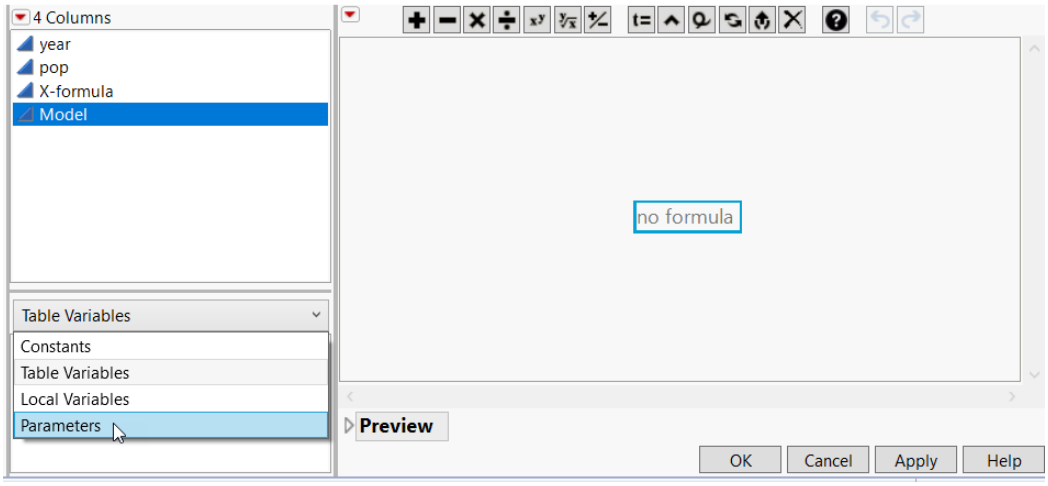
Create a Formula Column

To fit a custom model, you must first create a model column with initial parameter estimates.

1. Select **Help > Sample Data Library** and open Nonlinear Examples/US Population.jmp.
2. Create a new column called Model.
3. Right-click the Model column and select **Column Properties > Formula**.

   The Formula Editor appears.
4. Select **Parameters** from the list below the list of columns.

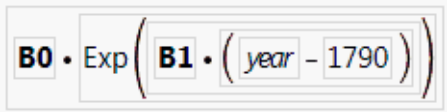
Figure 15.2 Select Parameters



5. Select **New Parameter**.
6. Type B0 for Name.

7. Type 3.9 for Value. This is the initial estimate of the parameter.
8. Click **OK**.
9. Select **New Parameter**.
10. Type B1 for Name and enter 0.022 for Value.
11. Click **OK**.
12. Enter the model formula using the Formula Editor functions, the column year, and the parameters.

Figure 15.3 Completed Model Formula



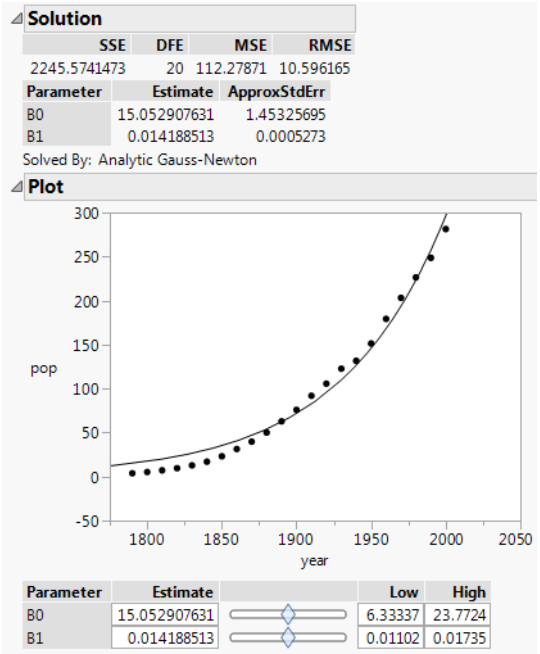
Tip: Click the gray triangle next to Transcendental to find the Exp command.

13. Click **OK**.

Fit a Nonlinear Model

1. Select **Analyze > Specialized Modeling > Nonlinear**.
2. Assign Model to the **X, Predictor Formula** role.
3. Assign pop to the **Y, Response** role.
4. Click **OK**.
5. Click **Go** on the Control Panel to fit the model.

Figure 15.4 Plot and Solution Report



The final parameter estimates are shown in the **Solution** report, along with other fit statistics. The fitted model is shown on the plot.

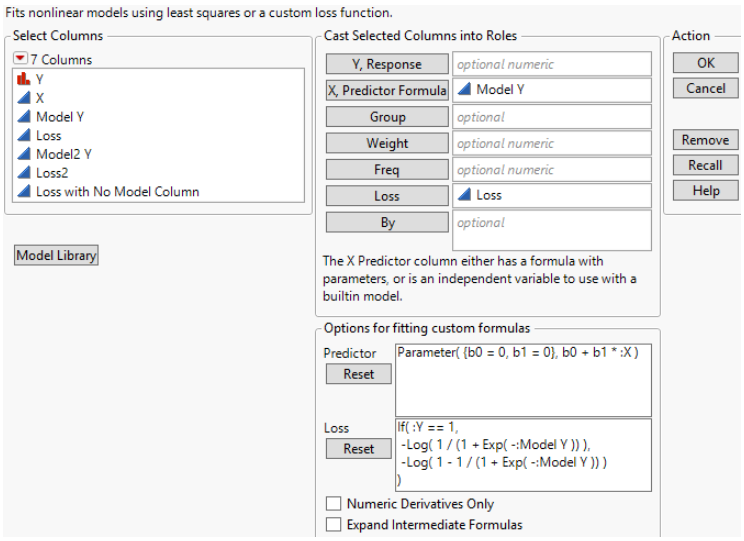
Parameters for Models with a Grouping Variable

In the formula editor, when you add a parameter, note the check box for **Expand Into Categories, selecting column**. This option is used to add several parameters (one for each level of a categorical variable for example) at once. When you select this option, a dialog appears that enables you to select a column. After selection, a new parameter appears in the Parameters list with the name D_*column*, where D is the name that you gave the parameter. When you use this parameter in the formula, a Match expression is inserted, containing a separate parameter for each level of the grouping variable.

Launch the Nonlinear Platform

To launch the Nonlinear platform, select **Analyze > Specialized Modeling > Nonlinear**.

Figure 15.5 Nonlinear Platform Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Nonlinear platform launch window has the following features:

Y, Response Select the *Y* variable.

X, Predictor Formula Select either the *X* variable or a column containing the model formula with parameters.

Note: If you select a column that does not contain a model formula with parameters and you also do not specify a custom formula, the Fit Curve platform is launched instead.

Note: If you select a formula column that contains a parameter list item that does not have an assignment, JMP automatically sets that parameter to zero and displays a message in the log.

Group Specify a grouping variable. The fitted model has separate parameters for each level of the grouping variable. This enables you to compare fitted models and estimated parameters across the levels of the grouping variable.

Note: If the formula does not have separate parameters for each group level, the Group variable is ignored by JMP.

Weight Specify a variable containing the weights of observations.

Freq Specify a variable representing the frequency of an observation.

Loss Specify a formula column giving a loss function.

By Specify a variable to perform a separate analysis for every level of the variable.

Model Library Launches the Model Library tool, which helps you choose initial values to create a formula column. See "Create a Formula Using the Model Library".

Options for fitting custom formulas If you specify a column in the X, Predictor Formula role that does not contain a model formula with parameters, you can use this option to create a custom formula. In the Predictor field, specify parameters, parameter values, and a custom formula that uses the selected *X* variable. Then, click the **Reset** button and launch the platform.

Numeric Derivatives Only Uses numeric derivatives only. This option is useful when you have a model for which it is too messy to take analytic derivatives. It can also be valuable in obtaining convergence in tough cases. This option is used only when a formula column is provided in the X, Predictor Formula role.

Expand Intermediate Formulas Tells JMP that if an ingredient column to the model is a column that itself has a formula, to substitute the inner formula, as long as it refers to other columns. To prevent an ingredient column from expanding, use the **Other** column property with a name of "Expand Formula" and a value of 0. This option is used only when a formula column is provided in the X, Predictor Formula role.

The Nonlinear Fit Report

The initial Nonlinear Fit report includes the following items, shown in Figure 15.6.

Control Panel Provides options for controlling the fitting process.

Go Starts the fitting process.

Stop Stops the fitting process.

Step Proceeds through the fitting process one iteration at a time.

Reset Resets the editable values into the formula, resets the iteration values, and calculates the SSE at these new values.

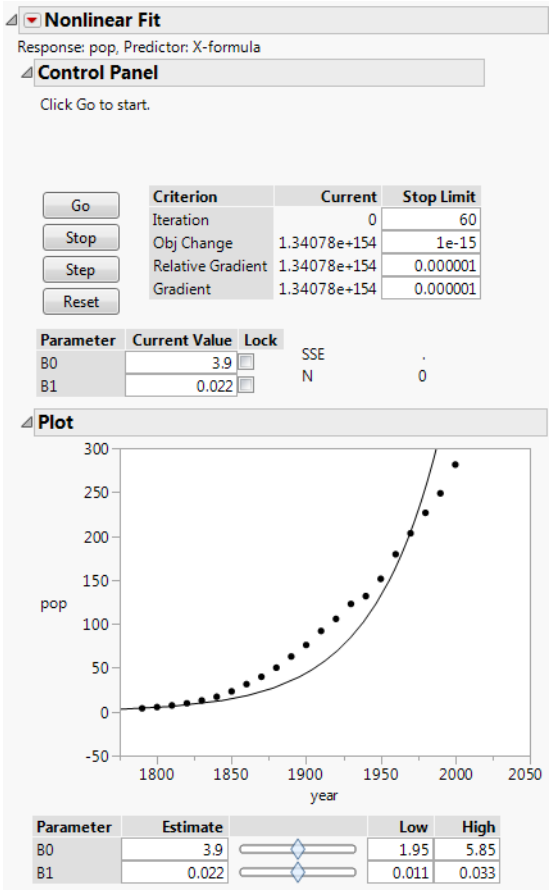
Criterion Shows iteration measures from the fitting process.

Current Shows the current value of each Criterion.

Stop Limit Sets limits on the measures listed under Criterion.

Plot Shows a plot of the *X* and *Y* variables for models with only one *X* variable. The model based on the current values is shown on the plot. To change the current values of the parameters, use the sliders or edit boxes beneath the plot.

Figure 15.6 Initial Nonlinear Fit Report



After you click **Go** to fit a model, the report includes the following additional items, shown in Figure 15.7.

Save Estimates Saves the current parameter values to the parameters in the formula column.

Confidence Limits Computes confidence intervals for all parameters. The intervals are profile likelihood confidence intervals, and are shown in the Solution report. The confidence limit computations involve a new set of iterations for each limit of each parameter, and the iterations often do not find the limits successfully. The **Edit Alpha** and **Convergence Criterion** options are for the confidence interval computations. For more information about the **Goal SSE for CL**, see "Profile Likelihood Confidence Limits".

Solution Shows the parameters estimates and other statistics.

SSE Shows the residual sum of squared errors. SSE is the objective that is to be minimized. If a custom loss function is specified, this is the sum of the loss function.

DFE Shows the degrees of freedom for error, which is the number of observations used minus the number of parameters fitted.

MSE Shows the mean squared error. It is the estimate of the variance of the residual error, which is the SSE divided by the DFE.

RMSE Estimates the standard deviation of the residual error, which is square root of the MSE.

RSquare (Available only if the model has an intercept and the sum of the residuals is close to zero.) Shows the RSquare value.

Parameter Lists the names that you gave the parameters in the fitting formula.

Estimate Lists the parameter estimates produced. Keep in mind that with nonlinear regression, there might be problems with this estimate even if everything seems to work.

ApproxStdErr Lists the approximate standard error, which is computed analogously to linear regression. It is formed by the product of the RMSE and the square root of the diagonals of the derivative cross-products matrix inverse.

Note: If the number of observations is less than or equal to the number of parameters in the fitted model, ApproxStdErr is not reported.

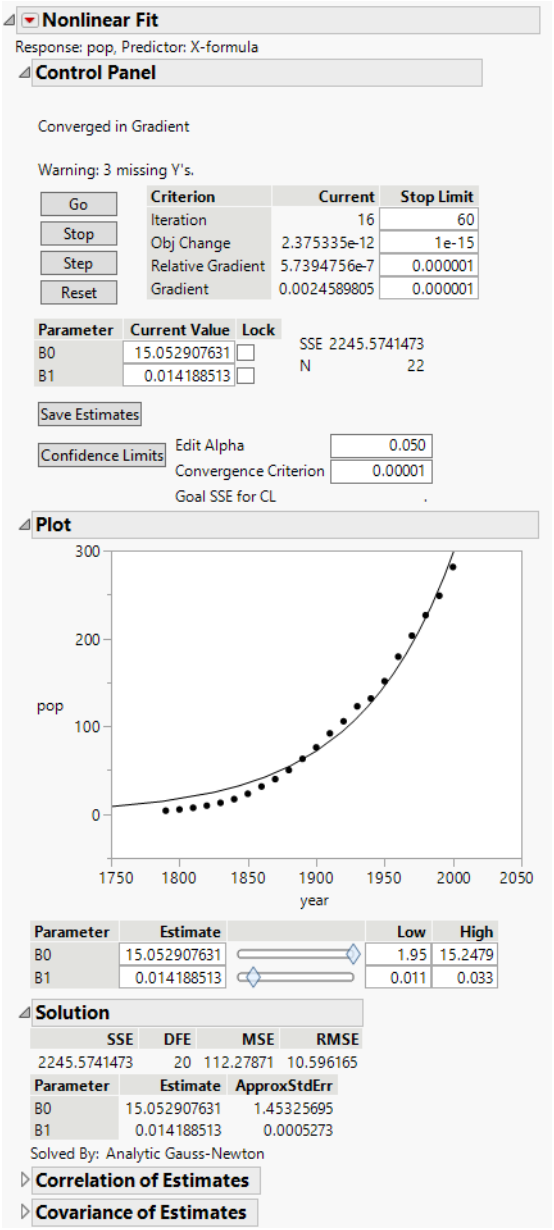
Lower CL and Upper CL Shows the confidence limits for the parameters. They are missing until you click the **Confidence Limits** on the Control Panel. For more information about the confidence intervals, see "Profile Likelihood Confidence Limits".

Excluded Data Shows a report showing fit statistics for excluded rows. This is useful for validating the model on observations not used to fit the model. You can use this feature in conjunction with the Remember Solution option to change the exclusions, and get a new report reflecting the different exclusions

Correlation of Estimates Displays the correlations between the parameter estimates.

Covariance of Estimates Displays the covariances between the parameter estimates.

Figure 15.7 Fitted Model Report



Nonlinear Platform Options

The Nonlinear Fit red triangle menu contains the following options:

Parameter Bounds Sets bounds on the parameters. When the option is selected, editable boxes appear in the Control Panel. Unbounded parameters are signified by leaving the field blank.

Plot Shows or hides a plot of the *X* and *Y* variables for models with only one *X* variable. The model shown on the plot is based on the current values of the parameters. To change the current values of the parameters, use the sliders or edit boxes beneath the plot. If you specify a Group variable at launch, then a curve shows for each group.

Iteration Options Specifies options for the fitting algorithm.

Iteration Log Records each step of the fitting process in a new window.

Numeric Derivatives Only Useful when you have a model that is too messy to take analytic derivatives for. It can also be valuable in obtaining convergence in tough cases.

Expand Intermediate Formulas Tells JMP that if an ingredient column to the formula is a column that itself has a formula, to substitute the inner formula, as long as it refers to other columns. To prevent an ingredient column from expanding, use the **Other** column property with a name of "Expand Formula" and a value of 0.

Newton Specifies whether Gauss-Newton (for regular least squares) or Newton-Raphson (for models with loss functions) is the optimization method.

QuasiNewton SR1 Specifies QuasiNewton SR1 as the optimization method.

QuasiNewton BFGS Specifies QuasiNewton BFGS as the optimization method.

Accept Current Estimates Tells JMP to produce the solution report with the current estimates, even if the estimates did not converge.

Show Derivatives Shows the derivatives of the nonlinear formula in the JMP log. See "Notes Concerning Derivatives", for technical information about derivatives.

Unthreaded Runs the iterations in the main computational thread. In most cases, JMP does the computations in a separate computational thread. This improves the responsiveness of JMP while doing other things during the nonlinear calculations. However, there are some isolated cases (models that have side effects that call display routines, for example) that should be run in the main thread, so this option should be turned on.

Profilers Provides various profilers for viewing response surfaces.

Profiler Shows the Prediction Profiler. The Profiler lets you view vertical slices of the surface across each *x*-variable in turn, as well as find optimal values of the factors.

Contour Profiler Shows the Contour Profiler. The Contour profiler lets you see two-dimensional contours as well as three dimensional mesh plots.

Surface Profiler Creates a three-dimensional surface plot. This option is available only for models with two or more X variables.

Parameter Profiler Shows the Prediction Profiler and profiles the SSE or loss as a function of the parameters.

Parameter Contour Profiler Shows the Contour Profiler and contours the SSE or loss as a function of the parameters.

Parameter Surface Profiler Creates a three-dimensional surface plot and profiles the SSE or loss as a function of the parameters. This option is available only for models with two or more parameters.

SSE Grid Create a grid of values around the solution estimates and compute the error sum of squares for each value. The solution estimates should have the minimum SSE. When the option is selected, the **Specify Grid for Output** report is shown with these features:

Parameter Lists the parameters in the model.

Min Displays the minimum parameter values used in the grid calculations. By default, Min is the solution estimate minus 2.5 times the ApproxStdErr.

Max Displays the maximum parameter value used in the grid calculations. By default, Max is the solution estimate plus 2.5 times the ApproxStdErr.

Number of Points Gives the number of points to create for each parameter. To calculate the total number of points in the new grid table, multiply all the Number of Points values. Initially Number of Points is 11 for the first two parameters and 3 for the rest. If you specify new values, use odd values to ensure that the grid table includes the solution estimates. Setting Number of Points to 0 for any parameter records only the solution estimate in the grid table.

When you click **Go**, JMP creates the grid of points in a new table. A highlighted row marks the solution estimate row if the solution is in the table.

Revert to Original Parameters Resets the platform to the original parameter values (the values given in the formula column parameters).

Remember Solution Creates a report called Remembered Models, which contains the current parameter estimates and summary statistics. Results of multiple models can be remembered and compared. This is useful if you want to compare models based on different parameter restrictions, or models fit using different options. Click the radio

button for a particular model to display that model in the Plot and the parameter estimates in the Control Panel.

Custom Estimate Gives an estimate of a function of the parameters. You provide an expression involving only parameters. JMP calculates the expression using the current parameter estimates, and also calculates a standard error of the expression using a first-order Taylor series approximation.

Custom Inverse Prediction Estimates the *X* value for a given *Y* value. It also calculates a standard error for the estimated *X*. JMP must be able to invert the model. The standard error is based on the first-order Taylor series approximation using the inverted expression. The confidence interval uses a *t*-quantile with the standard error, and is a Wald interval.

Show Prediction Expression Shows the prediction model or the loss function at the top of the report.

Save Pred Confid Limits Saves asymptotic confidence limits for the model prediction. This is the confidence interval for the average *Y* at a given *X* value.

Save Indiv Confid Limits Saves asymptotic confidence limits for an individual prediction. This is the confidence interval for an individual *Y* value at a given *X* value.

Save Formulas Gives options for saving model results to data table columns:

Save Prediction Formula Saves the prediction formula with the current parameter estimates.

Save Std Error of Predicted Saves the standard error for a model prediction. This is the standard error for predicting the average *Y* for a given *X*. The formula is of the form `Sqrt(VecQuadratic(matrix1,vector1))`. `matrix1` is the covariance matrix associated with the parameter estimates, and `vector1` is a composition of the partial derivatives of the model with respect to each parameter.

Save Std Error of Individual Saves the standard error for an individual prediction. This is the standard error for predicting an individual *Y* value for a given *X* value. The formula is of the form `Sqrt(VecQuadratic(matrix1,vector1)+mse)`. `matrix1` is the covariance matrix associated with the parameter estimates, `vector1` is a composition of the partial derivatives of the model with respect to each parameter, and `mse` is the estimate of error variance.

Save Residual Formula Saves the formula for computing the residuals.

Save Pred Confid Limit Formula Saves the formula to calculate the confidence interval for a model prediction. This is a confidence interval for the average *Y* for a given *X*.

Save Indiv Confid Limit Formula Saves the formula to calculate the confidence interval for an individual prediction. This is a confidence interval for an individual *Y* for a given *X*.

Save Inverse Prediction Formula Saves formulas for the inverse of the model, the standard error of an inverse prediction, and the standard error of an individual inverse prediction.

Save Specific Solving Formula Equivalent to Save Inverse Prediction Formula in simple cases. However, this command allows the formula to be a function of several variables and allows expressions to be substituted. This feature works only for solving easily invertible operators and functions that occur just once in the formula.

After selecting this command, a dialog appears that enables you to select the variable to solve for. You can also edit the names of the columns in the resulting table. You can also substitute values for the names in the dialog. In these cases, the formula is solved for those values.

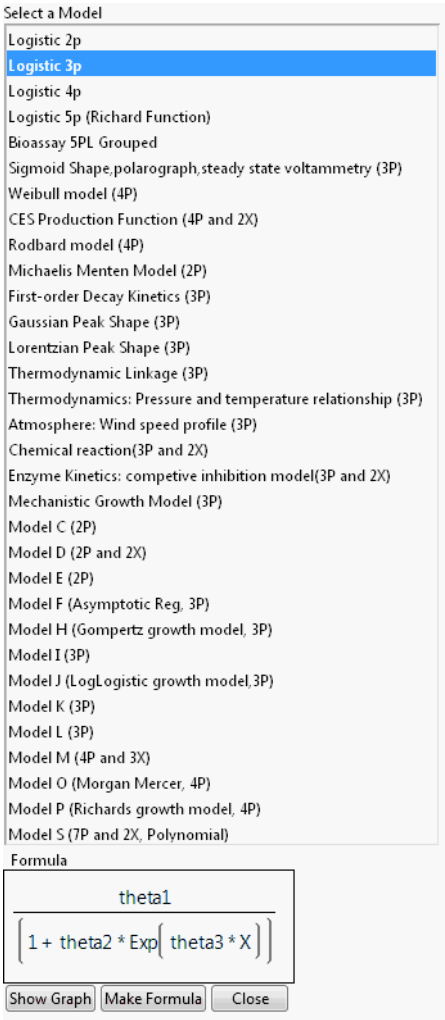
Note: The standard errors, confidence intervals, and hypothesis tests are correct only if least squares estimation is done, or if maximum likelihood estimation is used with a proper negative log-likelihood.

Save Estimates to Table Saves the parameter estimates to a new data table.

Create a Formula Using the Model Library

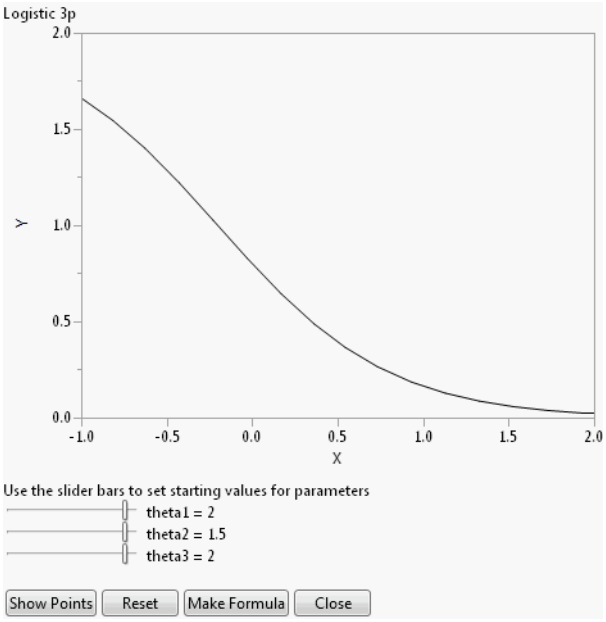
The Model Library can assist you in creating the formula column with parameters and initial values. Click the **Model Library** button on the Nonlinear launch window to open the library. Select a model in the list to see its formula in the **Formula** box.

Figure 15.8 Nonlinear Model Library Dialog



Click **Show Graph** to show a 2-D theoretical curve for one-parameter models and a 3-D surface plot for two-parameter models. No graph is available for models with more than two explanatory (*X*) variables. On the graph window, change the default initial values of parameters using the slider, or clicking and entering values in directly.

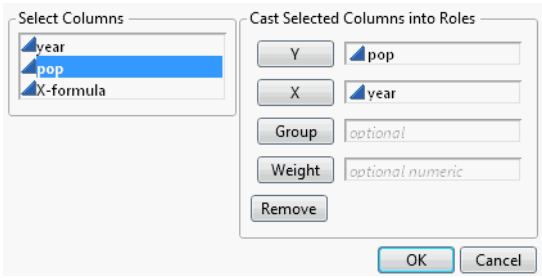
Figure 15.9 Example Graph in Model Library



The **Reset** button sets the initial values of parameters back to their default values.

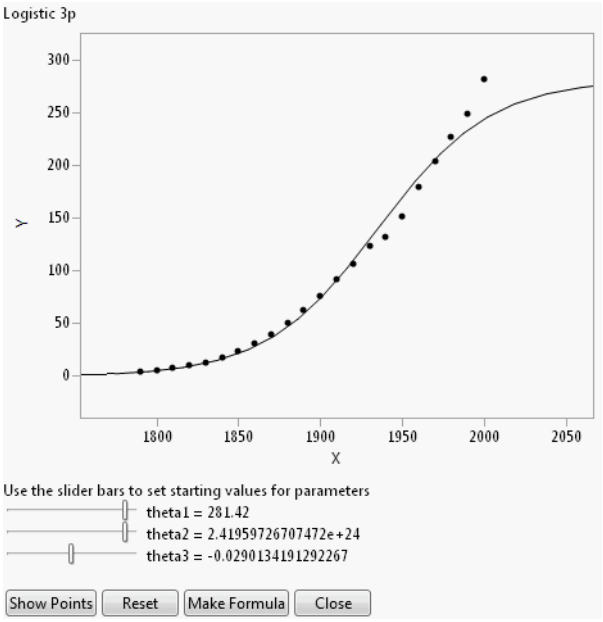
Click **Show Points** to overlay the actual data points to the plot. The dialog in Figure 15.10 opens, asking you to assign columns into *X* and *Y* roles, and an optional Group role. The Group role allows for fitting the model to every level of a categorical variable. If you specify a Group role here, also specify the Group column on the platform launch window.

Figure 15.10 Select Roles



For most models, the starting values are constants. Showing points enables you to adjust the parameter values to see how well the model fits for different values of the parameters. For the US population example, the points are shown in Figure 15.11.

Figure 15.11 Show Points



Clicking **Make Formula** at this point (after using **Show Points**) creates a new data table column named after the model that you chose from the Model Library. This column has the formula as a function of the latest parameter starting values.

Note: If you click **Make Formula** before using the **Show Graph** or **Show Points** buttons, you are asked to provide the *X* and *Y* roles, and an optional Group role (Figure 15.10). After that, you are brought back to the plot so that you have the opportunity to adjust the parameters starting values if desired. At that point click **Make Formula** again to create the new column.

Once the formula is created in the data table, continue the analysis by assigning the new column as the **X, Predictor Formula** in the Nonlinear launch dialog.

Customize the Nonlinear Model Library

The Model Library is created by a built-in script named NonlinLib.jsl, located in the Resources/Builtins folder in the folder that contains JMP (Windows) or in the Application Package (macOS). You can customize the nonlinear library script by modifying this script.

Note: You might not be able to directly edit the file. If this is the case, open the script, make the desired edits, and save to a different location on your computer. Use the same filename. Then, delete the old NonlinLib.jsl script in the Resources/Builtins folder and move the new NonlinLib.jsl file into the Resources/Builtins folder.

To add a model, you must add three lines to the list named `Listofmodellist#`. These three lines are actually a list themselves, which consists of the following three parts.

- Model name, a quoted string
- Model formula, an expression
- Model scale

For example, suppose you want to add a model called “Simple Exponential Growth” that has the form

$$y = b_1 e^{kx}$$

Add the following lines to the NonlinLib.jsl script

```
{//Simple Exponential Growth
     "Simple Exponential Growth",
     Expr(Parameter({b1=2, k=0.5}, b1*exp(k * :X))),
     lowx = -1; highx = 2; lowy = 0;  highy = 2},
```

Some things to note:

- The first line is simply an open bracket (starting the list) and an optional comment. The second line is the string that is displayed in the model library window.
- The values of `lowx`, `highx`, `logy`, and `highy` specify the initial window for the theoretical graph.
- There is a comma as the last character in the example above. If this is the final entry in the `Listofmodellist#` list, the comma can be omitted.
- If the model uses more than two parameters, replace the last line (containing the graph limits) with the quoted string “String Not Available”.

To delete a model, delete the corresponding three-lined list from the `Listofmodellist#` list.

Additional Examples

- “Example of Maximum Likelihood: Logistic Regression”
- “Example of a Probit Model with Binomial Errors: Numerical Derivatives”
- “Example of a Poisson Loss Function”
- “Example of Setting Parameter Limits”

Example of Maximum Likelihood: Logistic Regression

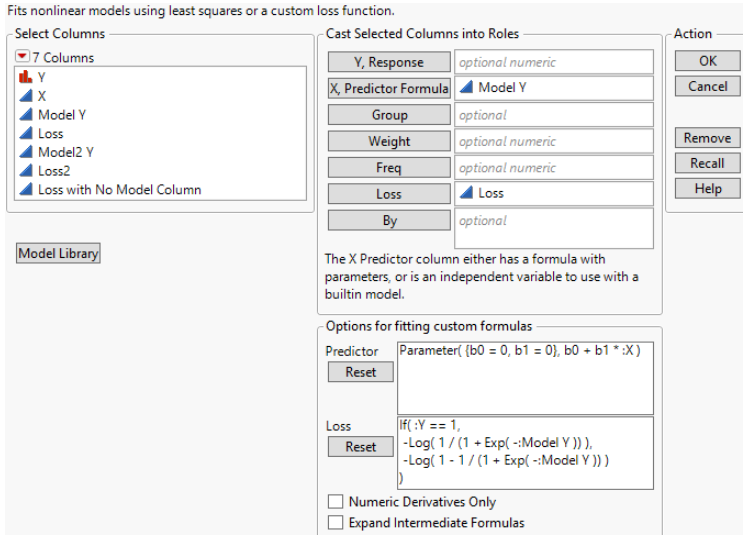
This example shows how to use the Nonlinear platform to minimize a loss function. The loss function is the negative of a log-likelihood function, thus producing maximum likelihood estimates.

The Logistic w Loss.jmp data table in the Nonlinear Examples sample data folder has an example for fitting a logistic regression using a loss function. The Y column contains ones for events and zeros for non-events. The Model Y column has the linear model, and the Loss column has the loss function. In this example, the loss function is the negative log-likelihood for each observation, or the negative log of the probability of getting the observed response.

Run the model by following the steps below:

1. Select **Help > Sample Data Library** and open Nonlinear Examples/Logistic w Loss.jmp.
2. Select **Analyze > Specialized Modeling > Nonlinear**.
3. Assign Model Y to the **X, Predictor Formula** role.
4. Assign Loss to the **Loss** role.

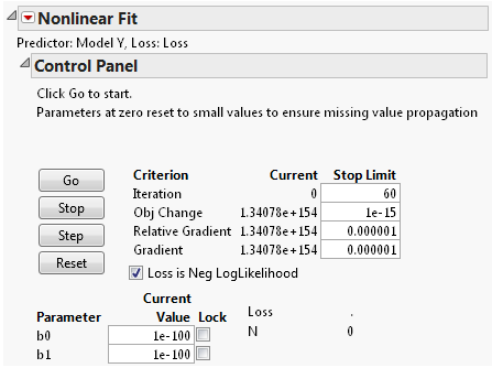
Figure 15.12 Nonlinear Launch Window



5. Click **OK**.

   The Nonlinear Fit Control Panel appears.

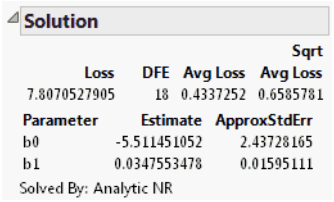
Figure 15.13 Nonlinear Fit Control Panel



6. Click **Go**.

   The parameter estimates are shown in the Solution report.

Figure 15.14 Solution Report



The Loss value in the Solution report is the negative log-likelihood evaluated at the parameter estimates.

Example of a Probit Model with Binomial Errors: Numerical Derivatives

The Ingots2.jmp sample data table includes the numbers of ingots tested for readiness after different treatments of heating and soaking times. The response variable, NReady, is binomial, depending on the number of ingots tested (Ntotal) and the heating and soaking times. Maximum likelihood estimates for parameters from a probit model with binomial errors are obtained using:

- numerical derivatives
- the negative log-likelihood as a loss function
- the Newton-Raphson method.

The average number of ingots ready is the product of the number tested and the probability that an ingot is ready for use given the amount of time it was heated and soaked. Using a probit model, the P column contains the model formula:

```
Normal Distribution(b0+b1*Heat+b2*Soak)
```

The argument to the `Normal Distribution` function is a linear model of the treatments.

To specify binomial errors, the loss function, Loss, has the formula

```
-(Nready*Log(p) + (Ntotal - Nready)*Log(1 - p))
```

Follow these steps to fit the model:

1. Select **Help > Sample Data Library** and open Ingots2.jmp.
2. Select **Analyze > Specialized Modeling > Nonlinear**.
3. Assign P to the **X, Predictor Formula** role,
4. Assign Loss to the **Loss** role.
5. Select the **Numeric Derivatives Only** option.
6. Click **OK**.
7. Click **Go**.

The platform used the Numerical SR1 method to obtain the parameter estimates shown in Figure 15.15.

Figure 15.15 Solution for the Ingots2 Data

Solution

Loss	DFE	Avg Loss	Sqrt Avg Loss
47.479945327	16	2.9674966	1.7226423

Parameter	Estimate	ApproxStdErr
b0	-2.8934153	0.51256572
b1	0.0399554554	0.01202329
b2	0.0362537934	0.15017139

Solved By: Numerical SR1

Example of a Poisson Loss Function

A Poisson distribution is often used to model count data.

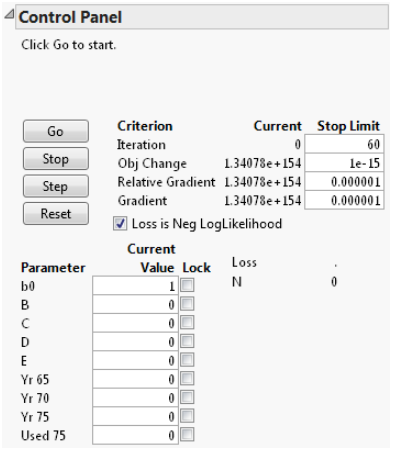
$$P(Y = n) = \frac{e^{-\mu}\mu^{n}}{n!}, n = 0, 1, 2, \ldots$$

where μ can be a single parameter, or a linear model with many parameters. Many texts and papers show how the model can be transformed and fit with iteratively reweighted least squares (Nelder and Wedderburn 1972). However, in JMP it is more straightforward to fit the model directly. For example, McCullagh and Nelder (1989) show how to analyze the number of reported damage incidents caused by waves to cargo-carrying vessels.

The data are in the Ship Damage.jmp sample data table. The model formula is in the model column, and the loss function (or negative log-likelihood) is in the Poisson column. To fit the model, follow the steps below:

1. Select **Help > Sample Data Library** and open Ship Damage.jmp.
2. Select **Analyze > Specialized Modeling > Nonlinear**.
3. Assign model to the **X, Predictor Formula** role.
4. Assign Poisson to the **Loss** role.
5. Click **OK**.
6. Set the **Current Value** (initial value) for b0 to 1, and the other parameters to 0.

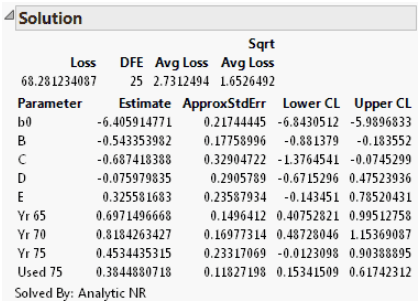
Figure 15.16 Enter New Parameters



7. Click **Go**.
8. Click the **Confidence Limits** button.

The Solution report appears. The results include the parameter estimates and confidence intervals, and other summary statistics.

Figure 15.17 Solution Table for the Poisson Loss Example



Solution

Loss	DFE	Avg Loss	Sqrt Avg Loss
68.281234087	25	2.7312494	1.6526492

Parameter	Estimate	ApproxStdErr	Lower CL	Upper CL
b0	-6.405914771	0.21744445	-6.8430512	-5.9896833
B	-0.543353982	0.17758996	-0.881379	-0.183552
C	-0.687418388	0.32904722	-1.3764541	-0.0745299
D	-0.075979835	0.2905789	-0.6715296	0.47523936
E	0.325581683	0.23587934	-0.143451	0.78520431
Yr 65	0.6971496668	0.1496412	0.40752821	0.99512758
Yr 70	0.8184263427	0.16977314	0.48728046	1.15369087
Yr 75	0.4534435315	0.23317069	-0.0123098	0.90388895
Used 75	0.3844880718	0.11827198	0.15341509	0.61742312

Solved By: Analytic NR

Note: The standard errors, confidence intervals, and hypothesis tests are correct only if least squares estimation is done, or if maximum likelihood estimation is used with a proper negative log-likelihood.

Example of Setting Parameter Limits

The Fit Curve personality enables you to fit a model and then use the prediction equation in the full personality of the Nonlinear platform. This method requires more steps and user input but allows any nonlinear model to be fit.

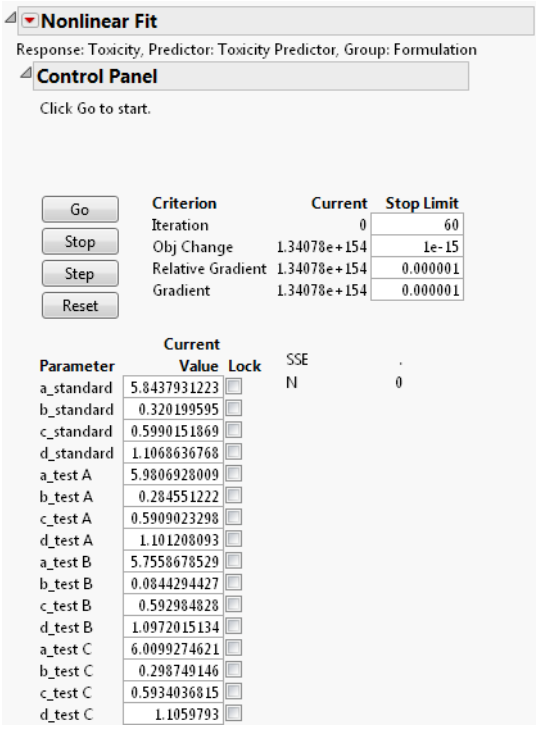
Complete "Example Using the Fit Curve Platform" to fit the model. This example shows how to save the prediction formula from Fit Curve and then set parameter limits in Nonlinear.

1. Click the Logistic 4P red triangle and select **Save Formulas > Save Parametric Prediction Formula**.

   A new column named Toxicity Predictor appears in the data table.
2. Select **Analyze > Specialized Modeling > Nonlinear.**
3. Assign Toxicity to the **Y, Response** role.
4. Assign Toxicity Predictor to the **X, Predictor Formula** role.
5. Assign Formulation to the **Group** role.
6. Click **OK**.

   The Nonlinear Fit window appears. In the Control Panel, parameter values and locking options are shown. The letters listed before each parameter correspond to variables from the Prediction Model in the Fit Curve function.

Figure 15.18 Nonlinear Fit Control Panel



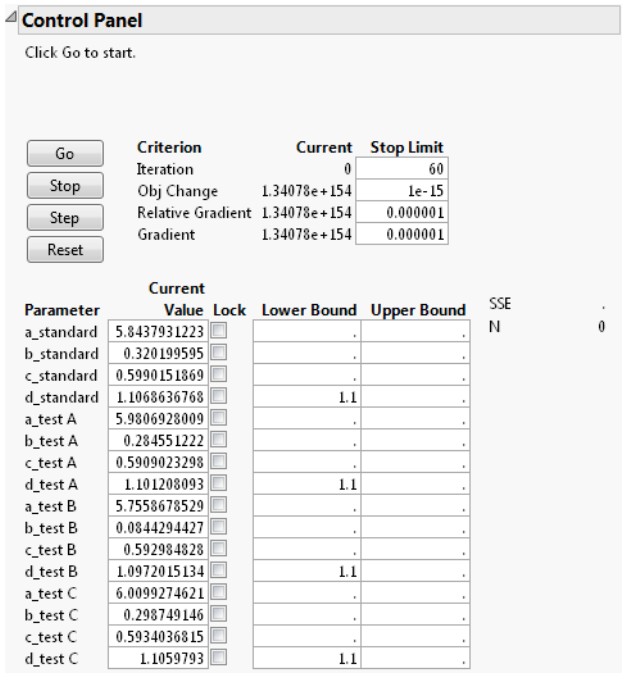
Tip: You can lock parameters if you know the values from prior information.

7. Click the Nonlinear Fit red triangle and select **Parameter Bounds**.

   Options for setting the lower and upper parameters appear next to the parameters.

8. Set the lower bounds for the parameters as shown in Figure 15.19. You know from prior experience that the maximum toxicity of the drug is at least 1.1.

Figure 15.19 Setting Parameter Bounds



Control Panel

Click Go to start.

Go
Stop
Step
Reset

Criterion	Current	Stop Limit
Iteration	0	60
Obj Change	1.34078e+154	1e-15
Relative Gradient	1.34078e+154	0.000001
Gradient	1.34078e+154	0.000001

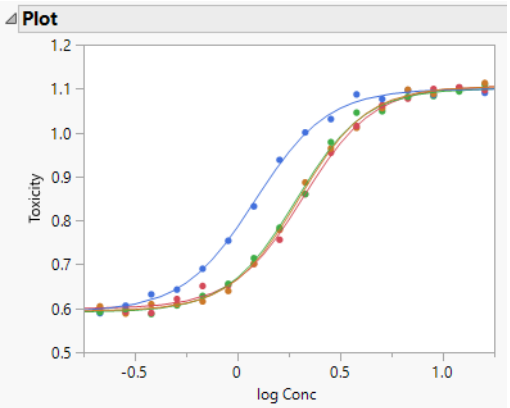
Parameter	Current Value	Lock	Lower Bound	Upper Bound
a_standard	5.8437931223		.	.
b_standard	0.320199595		.	.
c_standard	0.5990151869		.	.
d_standard	1.1068636768		1.1	.
a_test A	5.9806928009		.	.
b_test A	0.284551222		.	.
c_test A	0.5909023298		.	.
d_test A	1.101208093		1.1	.
a_test B	5.7558678529		.	.
b_test B	0.0844294427		.	.
c_test B	0.592984828		.	.
d_test B	1.0972015134		1.1	.
a_test C	6.0099274621		.	.
b_test C	0.298749146		.	.
c_test C	0.5934036815		.	.
d_test C	1.1059793		1.1	.

SSE	.
N	0

9. Click **Go**.

   The final parameter estimates are shown in the **Solution** report, along with other fit statistics. The fitted model is shown on the plot.

Figure 15.20 Nonlinear Fit Plot and Parameter Estimates



Parameter	Estimate		Low	High
a_standard	5.843793154		2.9219	8.76569
b_standard	0.3201995954		0.1601	0.4803
c_standard	0.5990151873		0.29951	0.89852
d_standard	1.1068636765		0.55343	1.6603
a_test A	5.9806928191		2.99035	8.97104
b_test A	0.2845512223		0.14228	0.42683
c_test A	0.5909023301		0.29545	0.88635
d_test A	1.1012080929		0.5506	1.65181
a_test B	5.6527018259		2.87793	8.6338
b_test B	0.0860027315		0.04221	0.12664
c_test B	0.5917244683		0.29649	0.88948
d_test B	1.1		0.5486	1.6458
a_test C	6.0099274821		3.00496	9.01489
b_test C	0.2987491449		0.14937	0.44812
c_test C	0.5934036814		0.2967	0.89011
d_test C	1.1059792995		0.55299	1.65897

Options below the plot allow for adjusting parameter limits and estimates.

Statistical Details for the Nonlinear Platform

- “Profile Likelihood Confidence Limits”
- “How Custom Loss Functions Work”
- “Notes Concerning Derivatives”
- “Notes on Effective Nonlinear Modeling”

Profile Likelihood Confidence Limits

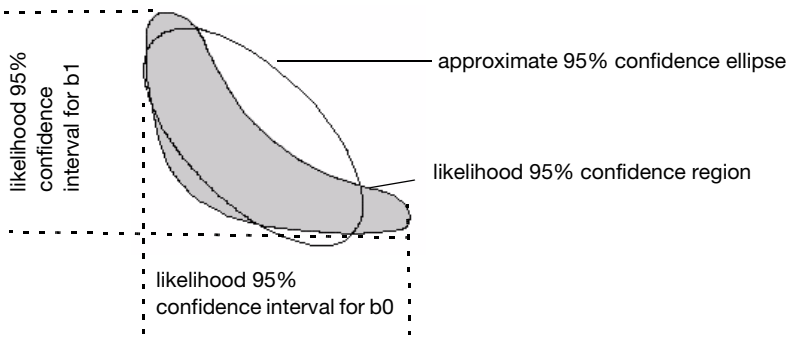
The upper and lower confidence limits for the parameters are based on a search for the value of each parameter after minimizing with respect to the other parameters. The search looks for values that produce an SSE greater by a certain amount than the solution's minimum SSE. The goal of this difference is based on the *F*-distribution. The intervals are sometimes called *likelihood confidence intervals* or *profile likelihood confidence intervals* (Bates and Watts 1988; Ratkowsky 1990).

Profile confidence limits all start with a *goal SSE*. This is a sum of squared errors (or sum of loss function) that an *F* test considers significantly different from the solution SSE at the given alpha level. If the loss function is specified to be a negative log-likelihood, then a Chi-square quantile is used instead of an *F* quantile. For each parameter's upper confidence limit, the parameter value is increased until the SSE reaches the goal SSE. As the parameter value is moved up, all the other parameters are adjusted to be least squares estimates subject to the change in the profiled parameter. Conceptually, this is a compounded set of nested iterations. Internally there is a way to do this with one set of iterations developed by Johnston and DeLong. See the *SAS/ETS User's Guide* (SAS Institute Inc. 2020).

Figure 15.21 shows the contour of the goal SSE or negative likelihood, with the least squares (or least loss) solution inside the shaded region:

- The asymptotic standard errors produce confidence intervals that approximate the region with an ellipsoid and take the parameter values at the extremes (at the horizontal and vertical tangents).
- Profile confidence limits find the parameter values at the extremes of the true region, rather than the approximating ellipsoid.

Figure 15.21 Diagram of Confidence Limits for Parameters



Likelihood confidence intervals are more trustworthy than confidence intervals calculated from approximate standard errors. If a particular limit cannot be found, computations begin for the next limit. When you have difficulty obtaining convergence, try the following:

- use a larger alpha, resulting in a shorter interval, more likely to be better behaved

- relax the confidence limit criteria.

How Custom Loss Functions Work

The nonlinear facility can minimize or maximize functions other than the default sum of squares residual. This section shows the mathematics of how it is done.

Suppose that $f(\beta)$ is the model. Then the Nonlinear platform attempts to minimize the sum of the loss functions defined as follows:

$$L = \sum_{i=1}^{n} \rho(f(\beta))$$

The loss function $\rho(\bullet)$ for each row can be a function of other variables in the data table. It must have nonzero first- and second-order derivatives. The default $\rho(\bullet)$ function, squared-residuals, is

$$\rho(f(\beta)) = (y - f(\beta))^2$$

To specify a model with a custom loss function, construct a variable in the data table and build the loss function. After launching the Nonlinear platform, select the column containing the loss function as the loss variable.

The nonlinear minimization formula works by taking the first two derivatives of $\rho(\bullet)$ with respect to the model, and forming the gradient and an approximate Hessian as follows:

$$L = \sum_{i=1}^{n} \rho(f(\beta))$$

$$\frac{\partial L}{\partial \beta_j} = \sum_{i=1}^{n} \frac{\partial \rho(f(\beta))}{\partial f} \frac{\partial f}{\partial \beta_j}$$

$$\frac{\partial^2 L}{\partial \beta_j \partial \beta_k} = \sum_{i=1}^{n} \left[\frac{\partial^2 \rho(f(\beta))}{(\partial f)^2} \frac{\partial f}{\partial \beta_j} \frac{\partial f}{\partial \beta_k} + \frac{\partial \rho(f(\beta))}{\partial f} \frac{\partial^2 f}{\partial \beta_k \partial \beta_j} \right]$$

If $f(\bullet)$ is linear in the parameters, the second term in the last equation is zero. If not, you can still hope that its sum is small relative to the first term, and use

$$\frac{\partial^2 L}{\partial \beta_j \partial \beta_k} \cong \sum_{i=1}^{n} \frac{\partial^2 \rho(f(\beta))}{(\partial f)^2} \frac{\partial f}{\partial \beta_j} \frac{\partial f}{\partial \beta_k}$$

The second term is probably small if ρ is the squared residual because the sum of residuals is small. The term is zero if there is an intercept term. For least squares, this is the term that distinguishes Gauss-Newton from Newton-Raphson.

Note: The standard errors, confidence intervals, and hypothesis tests are correct only if least squares estimation is done, or if maximum likelihood estimation is used with a proper negative log-likelihood.

Notes Concerning Derivatives

The nonlinear platform takes symbolic derivatives for formulas with most common operations. This section shows what type of derivative expressions result.

If you open the Negative Exponential.jmp nonlinear sample data example, the actual formula for the Nonlinear column looks something like this:

```
Parameter({b0=0.5, b1=0.5}, b0*(1-Exp(-b1*X)))
```

The Parameter block in the formula is hidden if you use the formula editor. That is how it is stored in the column and how it appears in the Nonlinear Launch dialog. Two parameters named **b0** and **b1** are given initial values and used in the formula to be fit.

The Nonlinear platform makes a separate copy of the formula, and edits it to extract the parameters from the expression. Then it maps the references to them to the place where they are estimated. Nonlinear takes the analytic derivatives of the prediction formula with respect to the parameters. If you use the **Show Derivatives** command, you get the resulting formulas listed in the log, like this:

Prediction Model:

```
b0 * First(T#1=1-(T#2=Exp(-b1*X)), T#3=-(-1*T#2*X))
```

The Derivative of Model with respect to the parameters is:

```
{T#1, T#3*b0}
```

The derivative facility works like this:

- In order to avoid calculating subexpressions repeatedly, the prediction model is threaded with assignments to store the values of subexpressions that it needs for derivative calculations. The assignments are made to names like `T#1`, `T#2`, and so on.
- When the prediction model needs additional subexpressions evaluated, it uses the `First` function, which returns the value of the first argument expression, and also evaluates the other arguments. In this case additional assignments are needed for derivatives.
- The derivative table itself is a list of expressions, one expression for each parameter to be fit. For example, the derivative of the model with respect to **b0** is `T#1`; its thread in the prediction model is `1-(Exp(-b1*X))`. The derivative with respect to **b1** is `T#3*b0`, which

is `-(-1*Exp(-b1*X)*X)*b0` if you substitute in the assignments above. Although many optimizations are made, it does not always combine the operations optimally. You can see this by the expression for `T#3`, which does not remove a double negation.

If you specify a loss function, then the formula editor takes derivatives with respect to parameters, if it has any. And it takes first and second derivatives with respect to the model, if there is one.

If the derivative mechanism does not know how to take the analytic derivative of a function, then it takes numerical derivatives, using the `NumDeriv` function. If this occurs, the platform shows the delta that it used to evaluate the change in the function with respect to a delta change in the arguments. You might need to experiment with different delta settings to obtain good numerical derivatives.

Tips

There are always many ways to represent a given model, and some ways behave much better than other forms. Ratkowsky (1990) covers alternative forms in his text.

If you have repeated subexpressions that occur several places in a formula, then it is better to make an assignment to a temporary variable. Then refer to it later in the formula. For example, one of the model formulas above was this:

```
If(Y==0, Log(1/(1+Exp(model))), Log(1 - 1/(1 + Exp(model))));
```

This could be simplified by factoring out an expression and assigning it to a local variable:

```
temp=1/(1+Exp(model));
If(Y==0, Log(temp), Log(1-temp));
```

The derivative facility can track derivatives across assignments and conditionals.

Notes on Effective Nonlinear Modeling

We strongly encourage you to *center polynomials*.

Anywhere you have a complete polynomial term that is linear in the parameters, it is always good to center the polynomials. This improves the condition of the numerical surface for optimization. For example, if you have an expression like the following:

$$a_1 + b_1 x + c_1 x^2$$

you should transform it to

$$a_2 + b_2(x - \bar{x}) + c_2(x - \bar{x})^2$$

The two models are equivalent, apart from a transformation of the parameters, but the second model is far easier to fit if the model is nonlinear.

The transformation of the parameters is easy to solve.

$$a_1 = a_2 - b_2\bar{x} + c_2\bar{x}^2$$
$$b_1 = b_2 - 2c_2\bar{x}$$
$$c_1 = c_2$$

If the number of iterations still goes to the maximum, increase the maximum number of iterations or relax one of the convergence criteria.

There is really no one omnibus optimization method that works well on all problems. JMP has options like **Newton**, **QuasiNewton BFGS**, **QuasiNewton SR1**, and **Numeric Derivatives Only** to expand the range of problems that are solvable by the Nonlinear Platform.

If the default settings are unable to converge to the solution for a particular problem, using various combinations of these settings to increase the odds of obtaining convergence.

Some models are very sensitive to starting values of the parameters. Working on new starting values is often effective. Edit the starting values and click **Reset** to see the effect. The plot often helps. Use the sliders to visually modify the curve to fit better. The parameter profilers can help, but might be too slow for anything but small data sets.

Chapter 16

JMP PRO Functional Data Explorer

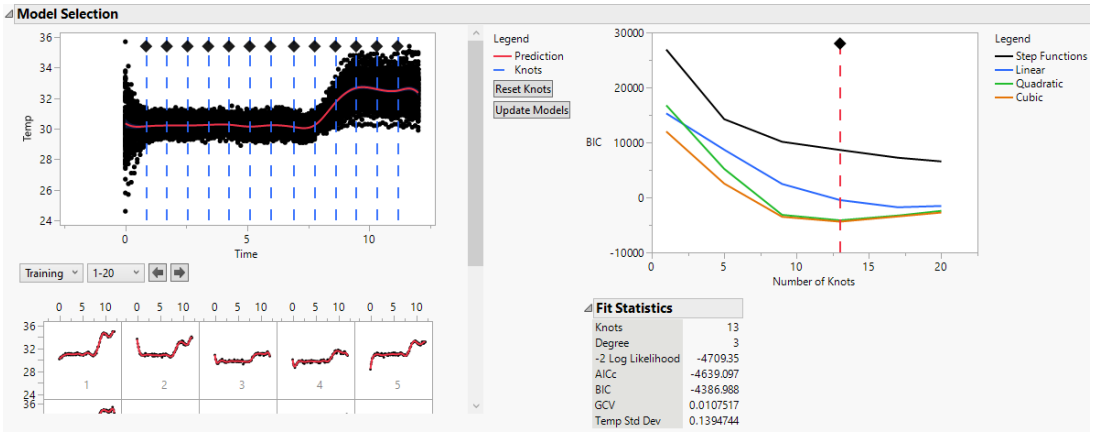
Perform Functional Data Analysis

The Functional Data Explorer platform is available only in JMP Pro.

The Functional Data Explorer (FDE) platform is designed for data that are functions, signals, or series. It can be used as an exploratory data analysis tool or as a dimension-reduction technique. In the case of dimension reduction, the FDE platform converts functional data into a form that can be analyzed in another JMP platform.

Data preprocessing tools are available in the FDE platform, including several types of transformations for output data and alignments for input data. A functional model is created by fitting a B-spline, P-spline, or Fourier basis model to the data. When a model is fit, functional principal components analysis (functional PCA) is automatically performed on the functional model. Results from the functional PCA, such as the functional principal component scores, are saved to a separate data table for feature extraction or dimension reduction.

Figure 16.1 B-Spline Model Selection in FDE



Contents

Overview of the Functional Data Explorer Platform 279
Example of Functional Data Explorer 280
Launch the Functional Data Explorer Platform 283
The Functional Data Explorer Report 286
Model Reports 287
Functional Data Explorer Platform Options 292
Functional Data Explorer Group Options 293
Data Processing Report Options 294
Model Report Options 297
Additional Examples of the Functional Data Explorer Platform 298
Example for Multiple Functional Processes 298
Example of Functional DOE 302
Statistical Details for the Functional Data Explorer Platform 304
Functional Model Fits 305
Function Summaries Details 306

JMP PRO Overview of the Functional Data Explorer Platform

Functional data can be defined as data that are recorded over a continuous domain, where a set of measurements form a curve or image. Often, the domain is time and the sets of measurements are defined by an ID variable. A functional data observation for ID level i at a specific point t on the domain is written as $f_i(t)$. The Functional Data Explorer platform enables you to explore and analyze functional data.

The form of functional data can be dense or sparse. Dense functional data occur when observations are on the same equally spaced grid of points for all levels of the ID variable. Sparse functional data occur when ID levels have different numbers of observations that are unequally spaced across the domain. The Functional Data Explorer platform can handle both forms of functional data.

Although functional data can be expressed in many ways, it can generally be classified into the following two cases:

- The response of interest, $f(t)$, has a functional form.
- There are one or more covariates, $f(t)$'s, that have functional forms. These are sometimes referred to as functional or signal processes.

The Functional Data Explorer platform is useful as an exploratory tool for any type of functional data. However, the strength of the platform is taking many functional processes (that might be associated with a scalar response) and extracting key features to use in further modeling. This can be done by first fitting a functional model to the data using a B-spline, P-spline, or Fourier basis model. Then, a functional principal components analysis (functional PCA) is performed on the functional model. Results from the functional PCA, such as the functional principal component (FPC) scores, are saved and used for feature extraction and analysis in another modeling platform, such as the Generalized Regression personality of the Fit Model platform. Alternatively, you can specify a set of supplementary variables and fit a generalized regression model within the FDE platform to determine how these variables affect the response.

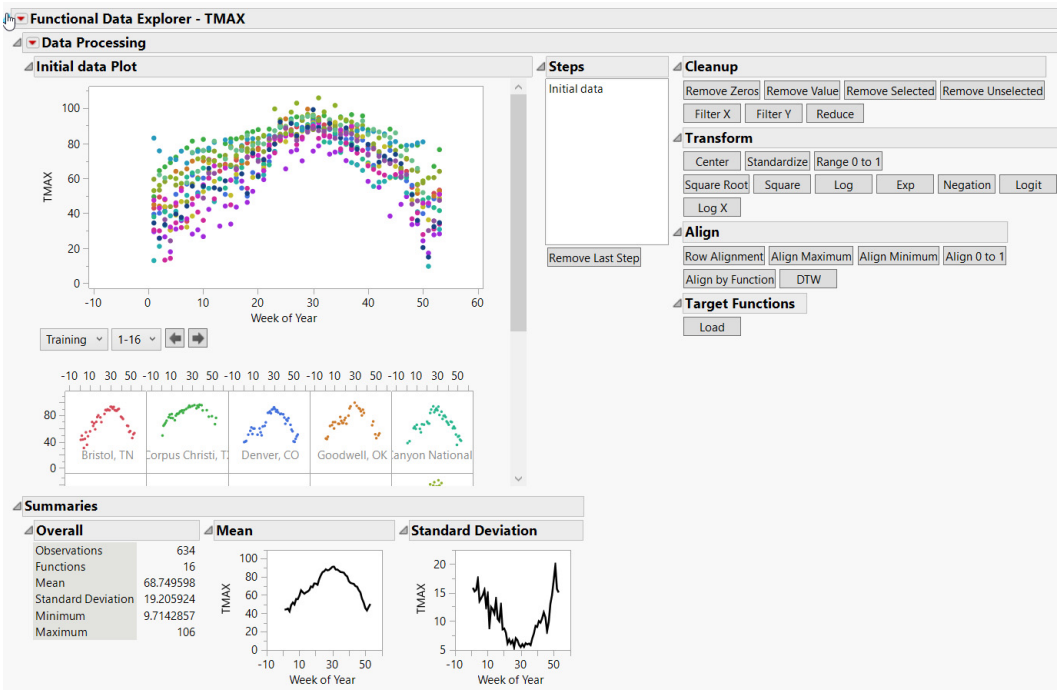
For more information about functional data analysis, see Ramsay and Silverman (2005).

JMP PRO Example of Functional Data Explorer

This example analyzes weekly weather data collected from 16 weather stations across the United States. Run the Weather Station Locations script in the data table to view a map of the locations. Daily temperatures are summarized as weekly averages. Not every weather station has a weekly temperature measurement for every week of the year. This is an example of sparse functional data.

1. Select **Help > Sample Data Library** and open Functional Data/Weekly Weather Data.jmp.
2. Select **Analyze > Specialized Modeling > Functional Data Explorer**.
3. Select TMAX and click **Y, Output**.
4. Select Week of Year and click **X, Input**.
5. Select ID and click **ID, Function**.
6. Click **OK**.

Figure 16.2 Initial Functional Data Explorer Report



The initial Functional Data Explorer report contains plots of the raw data, summary statistics, and summary plots for the functional mean and functional standard deviation of the data. There are also buttons for data processing options. Data processing options are

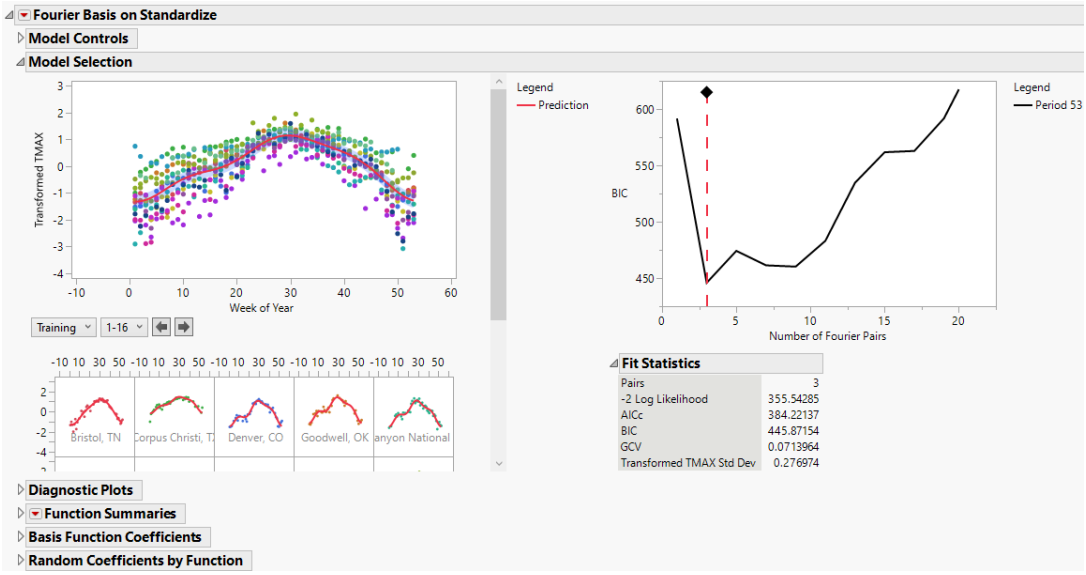
also accessible from the Data Processing red triangle menu. Prior to modeling, it is often a good idea to standardize your output data.

7. Click the **Standardize** button under the Transform menu.

   The data plots and summary statistics are updated based on the specified transformation. Standardized is added to the Steps list.

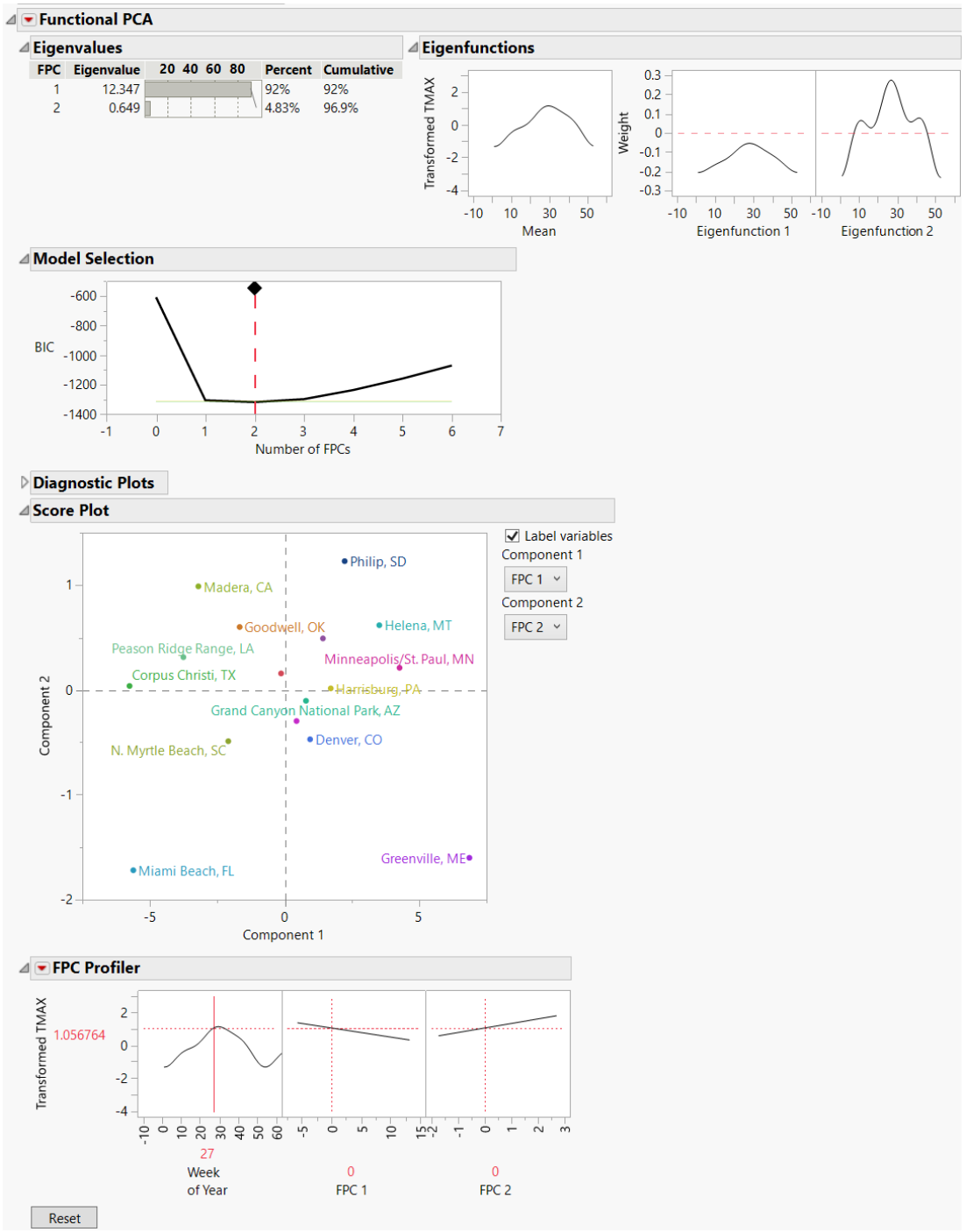
8. Click the Functional Data Explorer red triangle and select **Models > Fourier Basis**.

Figure 16.3 Fourier Basis Model Report



The Fourier Basis report includes several reports that contain information about the selected model. In the Model Selection report, the displayed model is the best fitting model according to the BIC fit criterion. For the weather data, the Fourier Basis model that is chosen has a period of 53 and three basis function pairs. Fit statistics and coefficients are also available for the model. Scroll down to view the Functional PCA report.

Figure 16.4 Functional PCA Report



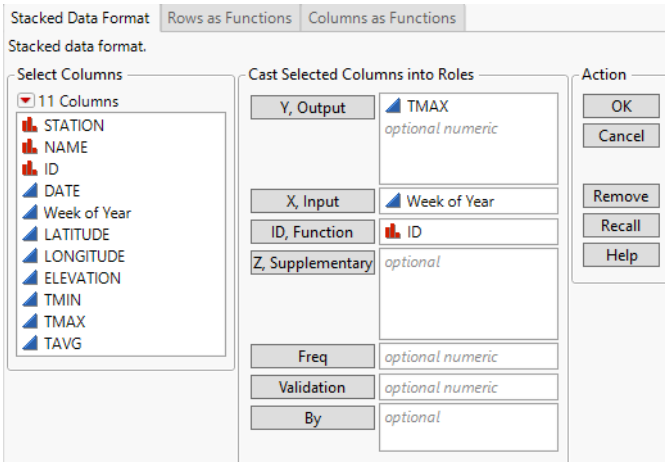
The Functional PCA report shows that the first two eigenvalues explain nearly 97% of the variation in the data. In the Model Selection graph, click and drag the red dashed line to 3 FPCs to see that the first three eigenvalues explain 99% of the variation in the data. However, the first eigenvalue alone explains 92%. You can use the Score Plot to detect individual functions that are outliers from the other functions. In the Score Plot, most of the locations are clustered together except for the Miami Beach, FL and Greenville, ME locations. Scroll up to the individual function plots. The function for the Miami Beach location is flatter, indicating less temperature variability than the rest of the locations. The function for the Greenville location has a lower maximum, indicating consistently colder temperatures than the rest of the locations.

Tip: Deselect the Label variables option in the Score Plot report to better identify outliers.

JMP PRO Launch the Functional Data Explorer Platform

Launch the Functional Data Explorer platform by selecting **Analyze > Specialized Modeling > Functional Data Explorer**.

Figure 16.5 Functional Data Explorer Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Functional Data Explorer launch window includes tabs for different types of data formats. Select the tab based on your data format.

Stacked Data Format Select for data tables where each row corresponds to a single observation. There are separate columns for the output, input, and ID variables.

Note: The Stacked Data Format is the only data format that enables you to specify multiple functional processes. If you assign more than one column to **Y, Output** in the Stacked Data Format tab, each Y variable is analyzed separately. A Fit Group report contains the individual reports for the Y variables.

Rows as Functions Select for data tables where each row corresponds to the full output function for one level of the ID variable. Each column is a level of the input variable.

Caution: The Rows as Functions format assumes that your observations are equally spaced in the input domain unless the FDE X column property is used. The FDE X column property enables this data format to use input variables specified in the column names.

Columns as Functions Select for data tables where each column corresponds to the entire output function for one level of the ID variable. Each row corresponds to a level of the input variable.

JMP PRO Launch Window Options

Y, Output Assigns the functional process, $f(t)$. There must be at least two observed output values for each level of the ID variable.

Note: Functions with fewer than two observed output values are removed from the analysis.

X, Input (Available for Stacked Data Format and Columns as Functions.) Assigns the input variable t. If no variable is specified for X, Input, the row number is used. Using the row number assumes that the observations are equally spaced in the input domain.

ID, Function (Available for Stacked Data Format and Rows as Functions.) Assigns the ID variable to each function. For Stacked Data, if no ID variable is assigned all observations are assumed to come from one function.

Z, Supplementary (Available for Stacked Data Format and Rows as Functions.) Assigns one or more supplementary variables. Supplementary variables are not used in any of the calculations in the Functional Data Explorer platform and including them does not affect the results. Supplementary variables are variables you might want to use in future analyses of the results from Functional Data Explorer. When you specify supplementary variables, they are included in a Supplementary column group in the tables that are created by the Save Data and Save Summaries options. These columns retain any column properties that were specified in the original data table. A Functional DOE Analysis option is also made available for the fitted models. See "Functional DOE Analysis".

Freq (Available only for Stacked Data Format.) Assigns a column whose numeric values represent a frequency for each row in the analysis. The effect of a frequency column is to expand the data table, so that any row with integer frequency *k* is expanded to *k* identical rows.

Validation Assigns an optional numeric column that defines the validation sets. This column should contain only two distinct values. The smaller value defines the training set and the larger value defines the validation set. If there are more than two values, the smallest value defines the training set and all other values define the validation set. The FDE platform uses the validation column to train and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

Note: The Validation option enables you to hold out complete functions, not a sample of observations from each function. Therefore, all observations that have the same ID value must be classified as either test or validation. You cannot have observations with the same ID value in both sets. For more information about this type of validation column, see "Grouped Validation Column".

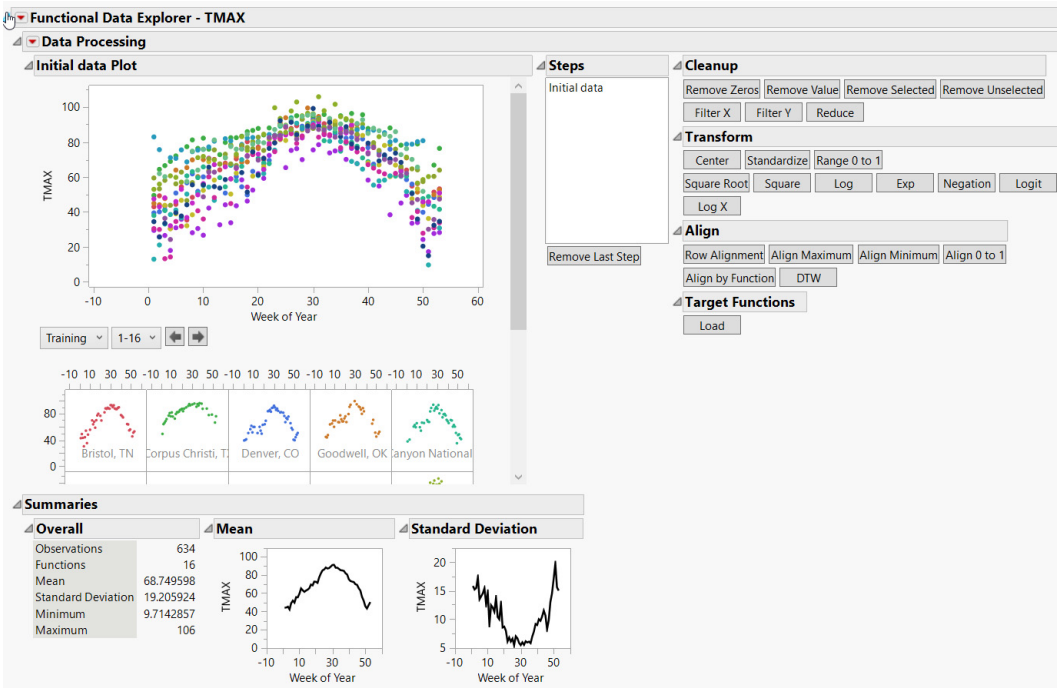
Caution: If you click the Validation button with no columns selected in the Select Columns list, and add a validation column to your data table it will not be a grouped validation column. Instead, use the Make Validation Column Platform, see "Grouped Validation Column".

By Assigns a column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

JMP PRO The Functional Data Explorer Report

The initial Functional Data Explorer report contains a Data Processing report and a Summaries report.

Figure 16.6 Initial Functional Data Explorer Report



The Data Processing Report includes an initial data plot of all observations, as well as a grid of individual data plots that correspond to levels of the ID variable. The maximum number of individual plots shown in the grid is twenty. There are drop-down menus and arrows above the grid to interactively select which individual plots are shown. If you specified a validation set, the first drop-down menu enables you to select if the individual plots shown are from the training set or the validation set. If there are more than twenty functions, the second drop-down menu and the arrows enable you to view different groups of plots. All graphs plot the functional response data over the values of the input variable. The Data Processing Report also contains data processing buttons.

The Summaries report contains a table of overall summary statistics, including the number of observations, number of functions, and the overall mean, standard deviation, minimum, and maximum values. There are also plots of the functional mean and functional standard deviation. The functional summary statistics displayed in the plots are computed at each unique value of the input variable.

The Data Processing and Summaries reports are useful for preprocessing your data prior to fitting a model or performing direct functional principal components analysis. There are data cleaning, transformation, and alignment buttons available in the Data Processing Report that correspond to the options in the Data Processing red triangle menu. See "Data Processing Report Options". Each time you perform a preprocessing step, the graphs and summary results are automatically updated. You can perform multiple preprocessing steps in sequence. Each step is added to the Steps list of the Data Processing report. To remove the most recent step, click the **Remove Last Step** button below the Steps outline.

JMP PRO Model Reports

Use the options in the Models submenu in the Functional Data Explorer red triangle menu to fit models to your data. See "Models" for the available models. Each time you fit a different type of model to the data, a model report appears. Each model report contains the following reports:

- "Model Controls"
- "Model Selection"
- "Diagnostic Plots"
- "Function Summaries"
- "Basis Function Coefficients"
- "Random Coefficients by Function"
- "Functional PCA"

JMP PRO Model Controls

The Model Controls report enables you to define parameters of models to compare in the Model Selection report. The appearance of the Model Controls report depends on the type of model that is fit.

B-Spline and P-Spline Model Controls

When a B-Spline or P-Spline model is fit, you can specify the following parameters:

Number of Knots Add, remove, or specify a range for the number of knots in each spline. The knots must be non-zero integers.

Note: The maximum number of knots allowed for B-Spline models is one less than the maximum number of observations per function or the number of unique inputs. The maximum number of knots allowed for P-Spline models is two less than the number of unique inputs. If you specify a number larger than the maximum, a warning message appears.

Spline Degree Add or remove spline degree fits from the Model Selection report.

Fourier Basis Model Controls

When a Fourier Basis model is fit, you can specify the following parameters:

Number of Fourier Pairs Add, remove, or specify a range for the number of Fourier pairs to compare.

Period Change the period of the function.

After you specify the model controls, click **Go** to view the updated models in the Model Selection report.

Tip: To specify the Model Controls prior to fitting a model, press Shift, click the Functional Data Explorer red triangle, and select the desired model. See "Models".

JMP PRO Model Selection

The Model Selection report contains an overall prediction plot, a grid of individual prediction plots, a solution path plot, and a table of fit statistics. The grid of individual prediction plots has the same layout and controls as the grid of individual plots in the Data Processing report. At most, there are twenty plots shown at a time. There are drop-down menus and arrows that enable you to view different groups of individual prediction plots. The solution path plot shows a model selection criterion plotted over values of a model parameter. The Bayesian Information Criterion (BIC) is the default fitting criterion. See "Model Report Options". For B-Spline and P-Spline models, there is a separate solution path for each spline degree plotted across the defined number of knots. For Fourier Basis models, the solution path is plotted across the number of Fourier pairs for a defined period. Use the Model Selection option to change the solution path plot parameters.

The current solution is designated by the dotted vertical line in the solution path plot. By default, the slider is placed at the number of knots or Fourier pairs that corresponds to the model that has the smallest model selection criterion value. You can drag the slider at the top of the dotted vertical line to change the number of knots or Fourier pairs in the current model. Dragging the slider automatically updates the prediction plots in the Model Selection report, as well as the information in all other reports.

The Fit Statistics table contains a description of the current solution model. It also displays the -2 Log Likelihood, the values for the AICc, BIC, and GCV model fitting criterion, and a value for the response standard deviation, denoted as <Y, Output> Std Dev. The response standard deviation is defined as the residual sigma from the fitted model. When a P-Spline model is selected, the penalty parameter λ (Lambda) is also displayed.

The prediction plots show the raw data and prediction curves that correspond to the current model. If there is a validation set, the predicted curves are not shown for functions that are in the validation set. For spline models, the default model selected is the degree of spline with the best fit. Click a specific spline in the solution path plot or the legend to change the current model selection. The curve in the overall prediction plot is a prediction of the mean curve. The curves in the individual prediction plots are prediction curves for each specific function. For B-Spline models, the overall prediction plot also displays the location of the knots. You can change the location of the knots by dragging the blue slider bars to different locations. To update the model reports according to the new knot locations, click the **Update Models** button. To reset the knots to their default locations, click the **Reset Knots** button.

JMP PRO Diagnostic Plots

The Diagnostic Plots report contains the Actual by Predicted plot and the Residual by Predicted plot. These plots help assess how well the current model fits the data. The Diagnostic Plots report is closed by default.

JMP PRO Function Summaries

Displays summaries from the Functional PCA for each level of the ID variable. The functional principal components associated with eigenvalues that explain more than 1% variation in the data are displayed by default. The mean, standard deviation, median, minimum, maximum, integrated difference, root integrated square error (RISE), and root integrated function square (RIFS) are also shown. The integrated difference and RISE summary values are used to determine how much the ID specific function differs from the overall mean function. The RIFS summary value is used for optimal curve fitting. See "Function Summaries Details". The Function Summaries red triangle menu contains the following options:

Customize Function Summaries Displays a window that enables you to select the number of FPCs and the summary statistics that are shown in the Function Summaries report. If the number of FPCs to be shown is specified, the Functional PCA report is also updated. There is also a checkbox, Save Graphs, that determines whether a graph for each function is included in the data table produced by the Save Summaries option.

Tip: If you have multiple functional processes, you can customize all Function Summaries reports to show the same summary values by clicking Ctrl and selecting Customize Function Summaries.

Save Summaries Saves the summary statistics specified in the Function Summaries report to a new data table. The name of the new data table describes the model fit. This data table contains formula columns for the eigenfunctions, mean function, prediction function, and conditional prediction function. There is also a column that contains the image of a graph of the raw data and the specified model fit for each function. In the data table, there is a profiler script that launches the prediction profilers for the prediction and conditional prediction formulas. These formulas are functions of the input variable, the ID variable, and the eigenfunctions.

JMP PRO Basis Function Coefficients

Displays the estimated basis function coefficients and their standard deviations. These are common across all levels of the ID variable and are fixed estimates in the mixed model framework. To view standard errors and confidence intervals for the coefficients, right-click in the table and select **Columns**.

JMP PRO Random Coefficients by Function

Displays the estimated random coefficients for each basis function and functional process combination. These are unique to each level of the ID variable and are random effects estimates in the mixed model framework.

JMP PRO Functional PCA

Functional principal components analysis (functional PCA) is performed on the fitted functional model. The Functional PCA report lists the eigenvalues that correspond to each functional principal component (FPC) in order from largest to smallest. The percent of variation accounted for by each FPC and the cumulative percent is listed and shown in a bar chart. There is a graph of the mean function as well as a graph for each component. The component graphs show the values of the eigenfunction.

You can perform model selection in the Functional PCA report to refine the selected number of functional principal components. There is a solution path plot that shows the Bayesian Information Criterion (BIC) plotted versus the number of FPCs. The current number of FPCs is designated by the dotted vertical line in the solution path plot. It is possible that models with different numbers of FPCs might have similar fits. Therefore, the solution path plot provides zones, which are intervals of values of the BIC statistic. There is a green zone and a yellow zone. The green zone contains values in the interval of the minimum BIC to the minimum BIC plus four and the yellow zone contains values in the interval of the minimum BIC plus four to the minimum BIC plus 10. By default, the model with the smallest number of FPCs within the green zone is selected. You can drag the slider at the top of the vertical line to change the number of FPCs. Dragging the slider automatically updates the other information in the Functional PCA report.

Note: The zones may appear small on your plot. Zoom in on the y-axis to better visualize the zones.

When Direct Functional PCA is performed, there is also an overall prediction plot and a grid of individual prediction plots. The grid of individual prediction plots has the same layout and controls as the grid of individual plots in the Data Processing report. At most, there are twenty plots shown at a time and there are drop-down menus and arrows that enable you to view different groups of individual prediction plots. Updating the number of FPCs automatically updates the prediction plots as well.

The prediction plots show the raw data and prediction curves that correspond to the current model. If there is a validation set, the predicted curves are not shown for functions that are in the validation set. The curve in the overall prediction plot is a prediction of the mean curve, given the specified number of FPCs. The curves in the individual prediction plots are prediction curves for each specific function, given the specified number of FPCs.

Note: The Functional PCA report is not shown if only a single function is modeled. Otherwise, if JMP is unable to perform Functional PCA, an error message appears in the Functional PCA report.

The following options are available in the Functional PCA red triangle menu:

Diagnostic Plots Shows or hides the Actual by Predicted and the Residual by Predicted plots. Use these plots help assess how well the model fits the data, given the selected number of functional principal components.

Score Plot Shows or hides a score plot of the FPC scores. Use the lists under Select Component to specify which FPCs are plotted on each axis of the Score Plot. If there is only one FPC, the FPC scores are plotted on the line $y = x$ and the lists to change the components are not shown. Score plots are useful for detecting outliers. In the case of FPC scores, the Score Plot is useful for detecting levels of the ID variable that have outlier functions. If you select a point in the score plot, the FPC Profiler is set to the scores for that function.

Tip: Hover over a point in the score plot to view a prediction plot of the fitted curve for that level of the ID variable.

FPC Profiler Shows or hides a profiler of the FPC scores. The FPC Profiler includes a column for the input variable and a column for each FPC score. For each target function that is specified, there are two additional profilers. One measures the difference from the target function, and the other measures the integrated error from the target function. For more information about FPC Profiler red triangle menu options, see *Profilers*.

Tip: Use the Reset button to reset all of the FPC scores to 0 in the profiler.

Customize Number of FPC's Specifies the number of FPC scores to show in the Functional PCA. Specifying the number of FPC scores in this option also updates the Function Summaries report. To view the mean model, set the number of FPC's to 0.

JMP PRO Functional Data Explorer Platform Options

The Functional Data Explorer red triangle menu contains the following options:

Summaries A submenu of the following options for functional summary statistics:

Plot Mean Shows or hides a plot of the functional mean in the Summaries report. On by default.

Plot Standard Deviation Shows or hides a plot of the functional standard deviation in the Summaries report. On by default.

Plot Median Shows or hides a plot of the functional median in the Summaries report.

Models A submenu of the following model options:

B-Splines Fits a basis spline (B-Spline) model to the data. Use the B-Spline model for non-periodic data.

P-Splines Fits a penalized basis spline (P-Spline) model to the data.

Fourier Basis Fits a Fourier Basis model to the data. Use the Fourier Basis model for periodic data. A periodic model assumes that the function finishes where it starts. See "Fourier Basis Model".

Note: If there are fewer than three unique input values, neither a Fourier basis model or a P-Spline model can be fit to the data, and a warning message appears.

Model Controls Shows a submenu that enables you to open the Model Controls panel prior to fitting a model. See "Model Controls".

Direct Functional PCA (Not available if there is only a single function.) Performs functional principal components analysis directly on the data, without fitting a basis function model first. This reduces computation time, particularly for large data sets. The implementation of Direct Functional PCA is as follows:

1. Align the input data to be between 0 and 1 and interpolate the observations to a common grid of input values.

2. Perform functional principal components analysis on the data.
3. Smooth the first eigenfunction using a P-Spline model with a knot at each grid point.
4. Remove the first smoothed eigenfunction from the data and repeat step 2 to step 4 until a large amount of the variation in the data is explained.

Once you perform a Direct Functional PCA, a Functional PCA report is shown. See "Functional PCA".

Save Data Saves the processed data to a new data table. The processed data are saved in the stacked data format.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Functional Data Explorer Group Options

Functional Data Explorer Group options are available only when there are multiple functional processes specified in the Stacked Data Format tab in the launch window.

Summaries A submenu of options to show or hide plots of functional summary statistics for all functional processes. See "Summaries".

Data Processing A submenu of data processing options that can be applied to all functional processes. See "Data Processing Report Options".

Models A submenu of model options that can be fit to all functional processes. See "Models".

Save Data Saves the processed data for all functional processes to a new data table. The processed data are saved in the stacked data format.

Save Summaries Saves the summary statistics specified in each Function Summaries report to a new data table. The name of the new data table describes the model fit. This data table contains formula columns for the eigenfunctions, mean function, prediction function, and

conditional prediction function for each of the Y variables. In the data table, there is a profiler script for each Y variable that launches the prediction profilers for the prediction and conditional prediction formulas. These formulas are functions of the input variable, the ID variable, and the eigenfunctions.

Arrange in Rows Specifies how many reports are displayed across the window.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Data Processing Report Options

The Data Processing red triangle menu contains the following options:

Cleanup A submenu of the following data cleanup options:

Remove Zeros Removes observations with zero values. If there are no zeros in the data, an alert appears, indicating that no zero values were found.

Remove Value Displays a specifications window that enables you to specify a value to remove from the data.

Remove Selected Removes observations that correspond to rows that are selected in the data table.

Remove Unselected Removes observations that correspond to rows that are not selected in the data table.

Caution: Remove Selected and Remove Unselected remove the row numbers. When Auto Recalc is enabled, you must add or delete rows before using these options.

Filter X Removes X values that fall outside of a specified interval. When you select the Filter X option, you must specify Below and Above values. The X values that fall outside of the specified interval are not used for the analysis.

Filter Y Removes Y values that fall outside of a specified interval. When you select the Filter Y option, you must specify Below and Above values. The Y values that fall outside of the specified interval are not used for the analysis.

Reduce Reduces the data over the X values using one of the following techniques:

- Use the Grid tab to interpolate observations to a common grid of values. You can specify the grid size. By default, the grid size is half the number of unique input values and therefore reduces the number of total observations. If you are not interested in reducing the number of total observations, but simply want your observations to be on the same grid, specify the grid size to be the number of unique input values.
- Use the Bin tab to create a specified number of bins that are evenly spaced over the unique X values. For each function (or level of the ID, Function variable), the observations within a bin are averaged to produce a Y value for the corresponding bin level.
- Use the Thin tab to remove every N observation over the X values, where N is determined by the specified thinning rate. This is done for each function (or level of the ID, Function variable). By default, the thinning rate is 2, which removes half of the observations in each function.

Note: The Remove options exclude the specified observations from the analysis and modeling reports, but the observations remain unchanged in the data table.

Transform A submenu of the following options to transform the data:

Center Centers the output.

Standardize Standardizes the output by centering and scaling the data to have mean 0 and variance 1.

Range 0 to 1 Scales the output to lie within the range of 0 and 1.

Square Root Transforms the data by computing the square root of the output. The output values must be nonnegative.

Square Transforms the data by computing the square of the output.

Log Transforms the data by computing the natural logarithm of the output.

Exp Transforms the data by computing the exponential function of the output.

Negation Transforms the data by negating the output.

Logit Transforms the data by computing the logit function of the output. The output values must be between 0 and 1.

Log X Transforms the data by computing the natural logarithm of the input.

Align A submenu of the following options to align the input data:

Row Alignment Replaces the input values with the row number.

Align Maximum Aligns the functions using the observed maximum output value for each ID level. The input value associated with the observed maximum output value is set to zero for each ID level and the other input values are shifted up or down based on the difference between the observed maximum and zero.

Align Minimum Aligns the functions using the observed minimum output value for each ID level. The input value associated with the observed minimum output value is set to zero for each ID level and the other input values are shifted up or down based on the difference between the observed minimum and zero.

Align 0 to 1 Aligns the output functions such that the range of the input values is 0 to 1.

Tip: Align 0 to 1 is particularly useful when you fit a P-Spline model.

Align by Function Aligns the output functions such that each function starts at the overall minimum of the input values and ends at the overall maximum of the input values.

Dynamic Time Warping (Available only when there is more than one function.) Aligns the output functions using dynamic time warping (DTW). DTW is a function alignment technique that finds an optimal warping to align two or more functions together. When you select the DTW option, a Select Reference Function window appears. Use this to select the reference function. The reference function is the function that the remaining functions are aligned to.

Once you select a reference function and click OK, a warping function plot is shown along with a list for the remaining query functions. On the warping function plot, the reference function is on the y-axis and the selected query function is on the x-axis. Deviations from the red diagonal line ($y = x$) indicate that the inputs of the query function have been warped for better alignment.

Target Functions (Available only when there is more than one function.) A submenu that enables you to load target functions.

Load Targets Shows a window that enables you to specify a target function. A target function is used for curve matching, where it is desirable for all of the functions to look like the target function. You can also specify two target functions to compare the remaining curves to the “best” and “worse” case functions.

If you specify one or more target functions, the data from the functions are not used in model fitting. For each specified target function, two rows are added to the FPC Profiler. See “FPC Profiler”.

Note: Target functions must be loaded before any other preprocessing steps are performed.

Dynamic Time Warping Options

Plot Warping Functions Shows or hides the warping function plot. On by default.

Save Distance Matrix Saves the distance matrix to a separate data table. The distance matrix can be useful for clustering the functions. The distance matrix data table contains a hierarchical clustering script.

Save Warping Functions Saves the warping functions to a separate data table. Each row of the data table contains the DTW adjusted input variable, the original input variable, and the ID variable.

JMP PRO Model Report Options

Model Selection Displays a submenu of choices to use as the model selection criteria. The choices are AICc, BIC, and GCV. See *Fitting Linear Models*.

Plot Basis Shows or hides a plot of all the basis functions on one graph.

Diagnostic Plots Shows or hides the Diagnostics Plots report. See "Diagnostic Plots"

Function Summaries Shows or hides the Function Summaries report. See "Function Summaries".

Basis Function Coefficients Shows or hides the Basis Function Coefficients report. See "Basis Function Coefficients".

Random Coefficients by Function Shows or hides the Random Coefficients by Function report. See "Random Coefficients by Function".

Functional PCA Shows or hides the Functional PCA report. See "Functional PCA".

Functional DOE Analysis (Available only if at least one supplementary variable is specified in the launch window.) Launches a Generalized Regression report within the FDE platform. A generalized regression model is fit to each of the FPC score functions using the supplementary variables as model effects. By default, a two degree factorial model is fit and the Estimation Method is Best Subset. Note that categorical quadratic effects are not included. If the number of terms in the model is greater than 21 or the number of functions is greater than 1000, the Estimation Method automatically switches to Pruned Forward. Alternatively, you can specify a model script in the original data table that defines the desired model fit. Modeling the FPC scores using the supplementary variables enables you to use the model fit to determine how the response changes based on the supplementary

variables. Use the FDOE Profiler to explore how the supplementary variables affect the response.

The Functional DOE Analysis report contains the following red triangle menu options:

Generalized Regression for FPC Scores Shows or hides the Generalized Regression reports for each FPC score. For more information on Generalized Regression model reports, see *Fitting Linear Models*.

Diagnostic Plots Shows or hides actual by predicted and residual plots for the response variable.

FDOE Profiler Shows or hides the FDOE Profiler, which enables you to explore how the response changes based on the supplementary variables. For more information about the FPC Profiler red triangle menu options, see *Profilers*.

Save Columns Shows a list with the options Save Prediction Formula and Save Residual Formula. These options save the corresponding formula to a new column in the data table.

Save Data Saves the modeled data to a new data table. The modeled data are saved in the stacked data format.

Remove Fit Removes the model report for the specified fit.

JMP PRO Additional Examples of the Functional Data Explorer Platform

- "Example for Multiple Functional Processes"
- "Example of Functional DOE"

JMP PRO Example for Multiple Functional Processes

This example uses the Fermentation Process.jmp and Fermentation Process Batch Yield Results.jmp sample data tables to analyze enzyme production. Yield is the amount of an enzyme produced by genetically modified yeast. There are 100 process measurements per batch that were taken at equally spaced times over a 12-hour period.

Use the Functional Data Explorer platform to fit models to the data and save functional principal components to a new data table. The functional principal components are then analyzed in the Generalized Regression personality of the Fit Model platform.

Fit Functional Models

1. Select **Help > Sample Data Library** and open Functional Data/Fermentation Process.jmp.
2. Select **Analyze > Specialized Modeling > Functional Data Explorer**.
3. In the Stacked Data Format tab, select Ethanol through pH and click **Y, Output**.
4. Select Time and click **X, Input**.
5. Select BatchID and click **ID, Function**.
6. Click **OK**.
7. Click the Functional Data Explorer Group red triangle and select **Data Processing > Align > Align 0 to 1**. This aligns the input variable to be between 0 and 1 in each Functional Data Explorer report.
8. Click the Functional Data Explorer Group red triangle and select **Models > B-Splines**. This fits a B-spline model to each of the functional processes.

Figure 16.7 Functional Data Explorer Report for Ethanol

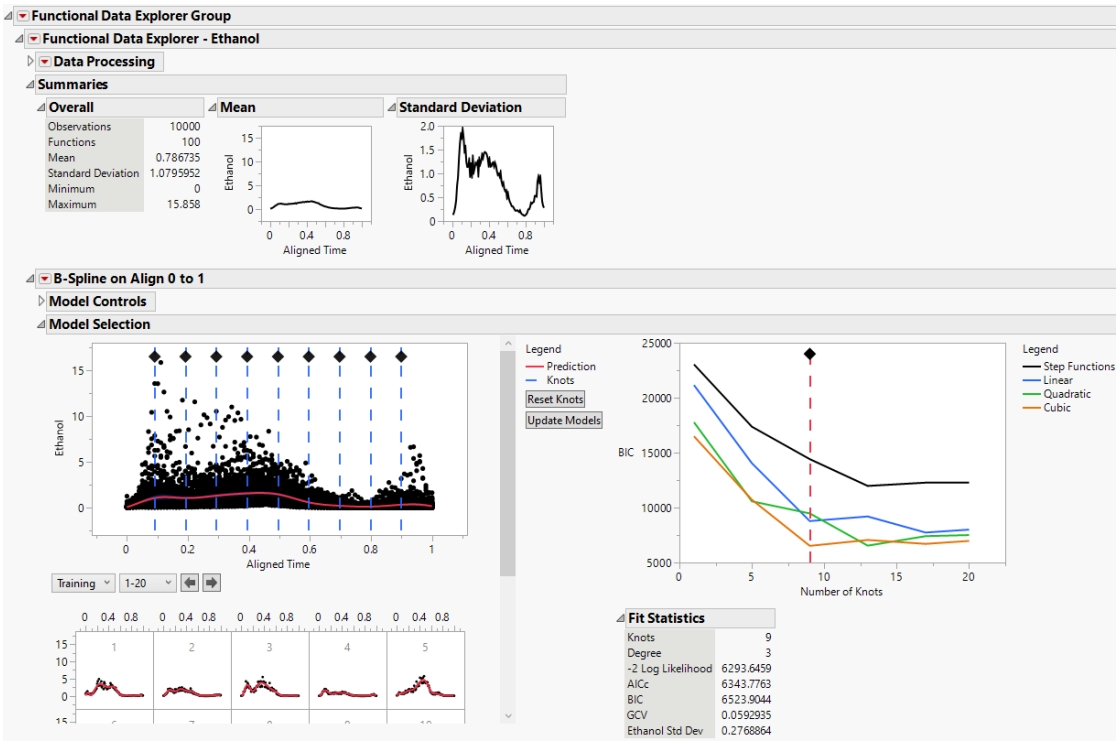


Figure 16.8 Model Summary Report for Ethanol

Function Summaries

BatchID	FPC 1	FPC 2	FPC 3	FPC 4	FPC 5	FPC 6	Mean	Std Dev	Median	Minimum	Maximum	Integrated Difference	Root Integrated Square Error	Root Integrated Function Square
1	0.6845185	0.4272826	0.3090187	-0.174881	-0.261418	0.2534972	1.2774239	1.2615192	0.6691253	0.0276312	3.4526584	-0.297809	0.9637933	1.7832112
2	0.2003363	-0.162552	0.1458985	-0.006107	0.0059254	0.0042297	0.843837	0.715458	0.7468215	0.0334158	2.0258972	-0.736315	0.317081	1.1007059
3	1.0570202	0.1670821	-0.025524	0.1554196	0.1599784	-0.213975	1.5128208	1.351404	1.4364321	-0.018162	3.834653	-0.060525	1.1267728	2.0232175
4	-0.225892	-0.342518	0.0456513	0.0469127	0.095793	-0.009734	0.5218201	0.4534864	0.4347383	0.0261908	1.6606284	-1.061051	0.4292202	0.6897988
5	0.7066471	0.8878144	-0.416058	0.1123721	0.0512395	0.1779	1.4818016	1.4399951	0.8069698	0.068398	4.6822729	-0.091374	1.2326783	2.0610547
6	-0.164355	0.0402334	0.0698129	0.0188387	-0.004275	-0.031643	0.6448696	0.4674211	0.6306233	0.0430701	1.5070566	-0.936894	0.1937517	0.7960802
7	0.4840975	0.0604706	-0.02545	0.138502	-0.044549	-0.079275	1.1816064	0.8311383	1.3331523	0.0450854	2.3372741	-0.395091	0.5320304	1.4384966
8	0.0598675	0.3232751	-0.194786	0.1468215	0.0999753	-0.021161	0.8861075	0.7813035	0.7196766	0.0031109	2.5739592	-0.69338	0.4349227	1.1820858
9	-0.35775	-0.244627	0.0950901	0.0453786	0.0302305	0.027442	0.4309337	0.3339422	0.3986422	0.0494439	1.0654932	-1.153327	0.4507855	0.5436156
10	0.1030905	0.3791323	-0.32985	-0.309402	0.1014623	0.1254908	1.0504657	0.8455706	0.804157	0.068398	3.0154441	-0.528012	0.622775	1.343811

Basis Function Coefficients

Parameter	Mean Estimate	Standard Deviation Estimate
Aligned Time[0]	0.068398	0
Aligned Time[0.04]	0.3628335	0.6072559
Aligned Time[0.1]	1.446524	2.3982214
Aligned Time[0.19]	0.8752812	0.9861143
Aligned Time[0.29]	1.3486495	1.4351681
Aligned Time[0.39]	1.5859873	1.4953555
Aligned Time[0.49]	1.69221	1.2326668
Aligned Time[0.6]	0.396788	0.7335251
Aligned Time[0.7]	0.2061623	0.2751322
Aligned Time[0.8]	0.0235344	0
Aligned Time[0.89]	0.3734323	0.6151888
Aligned Time[0.95]	0.3903473	1.3136877
Aligned Time[1]	0.1561339	0.1818355

Figure 16.7 and Figure 16.8 show the model reports for one of the functional process variables, Ethanol. Scroll through the full report to view the models fit for each of the process variables. Next, use the FPCs in the Function Summaries report in an analysis.

Save FPCs and Link Yield Results

1. Press Ctrl, click any Function Summaries red triangle, and select **Customize Function Summaries**.
2. In the box next to **Enter number of FPCs to show**, type 3.
3. Click the **Deselect All Summaries** box.
4. Click **OK**.
5. Click the Functional Data Explorer Group red triangle and select **Save Summaries**.
6. Select **Help > Sample Data Library** and open Functional Data/Fermentation Process Batch Yield Results.jmp.
7. In the Functional Data Explorer Model Summaries.jmp data table, right-click BatchID and deselect **Link ID**.
8. In the Functional Data Explorer Model Summaries.jmp data table, right-click BatchID and select **Link Reference** > Fermentation Process Batch Yield Results.jmp.

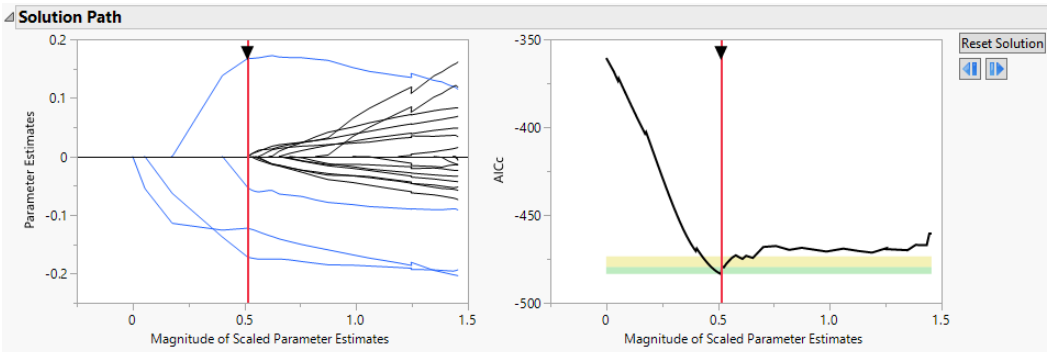
   This virtually joins the yield data table and the summaries data table.

Fit a Generalized Regression Model

Use the Generalized Regression personality of the Fit Model platform to determine how Yield is affected by the functional process variables.

1. In the Functional Data Explorer Model Summaries.jmp data table, select **Analyze > Fit Model**.
2. Click the triangle next to referenced by BatchID to Fermentation Process Batch Yield Results.
3. Select Yield[BatchID] and click **Y**.
4. Select the remaining columns, except Time and BatchID, and click **Add**.
5. Change the Personality to Generalized Regression.
6. Click **Run**.
7. Select the **Adaptive** box.
8. Click **Go**.

Figure 16.9 Generalized Regression Report for Batch Yield



Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	0.53436	0.0020214	69880551	<.0001*	0.5303981	0.5383219
Ethanol FPC 1	0	0	0	1.0000	0	0
Ethanol FPC 2	-0.042765	0.0060948	49.23396	<.0001*	-0.054711	-0.03082
Ethanol FPC 3	0	0	0	1.0000	0	0
Temp FPC 1	0	0	0	1.0000	0	0
Temp FPC 2	0	0	0	1.0000	0	0
Temp FPC 3	0	0	0	1.0000	0	0
Molasses Feed FPC 1	0	0	0	1.0000	0	0
Molasses Feed FPC 2	8.4212e-5	1.4668e-5	32.960213	<.0001*	5.5463e-5	0.000113
Molasses Feed FPC 3	0	0	0	1.0000	0	0
NH3 Feed FPC 1	0	0	0	1.0000	0	0
NH3 Feed FPC 2	-0.000598	8.3927e-5	50.851939	<.0001*	-0.000763	-0.000434
NH3 Feed FPC 3	0	0	0	1.0000	0	0
Air FPC 1	-9.712e-6	3.8557e-6	6.3454451	0.0118*	-1.727e-5	-2.156e-6
Air FPC 2	0	0	0	1.0000	0	0
Air FPC 3	0	0	0	1.0000	0	0
Tank Level FPC 1	0	0	0	1.0000	0	0
Tank Level FPC 2	0	0	0	1.0000	0	0
Tank Level FPC 3	0	0	0	1.0000	0	0
pH FPC 1	0	0	0	1.0000	0	0
pH FPC 2	0	0	0	1.0000	0	0
pH FPC 3	0	0	0	1.0000	0	0

The Generalized Regression report shows that Yield is significantly affected by certain components of Ethanol, Molasses Feed, NH3 Feed, and Air. The RSquare for the model is 0.73225. By using FDE to perform dimension reduction on the functional processes first, you greatly reduce the number of variables, while still retaining the ability to build a reasonable prediction models.

Example of Functional DOE

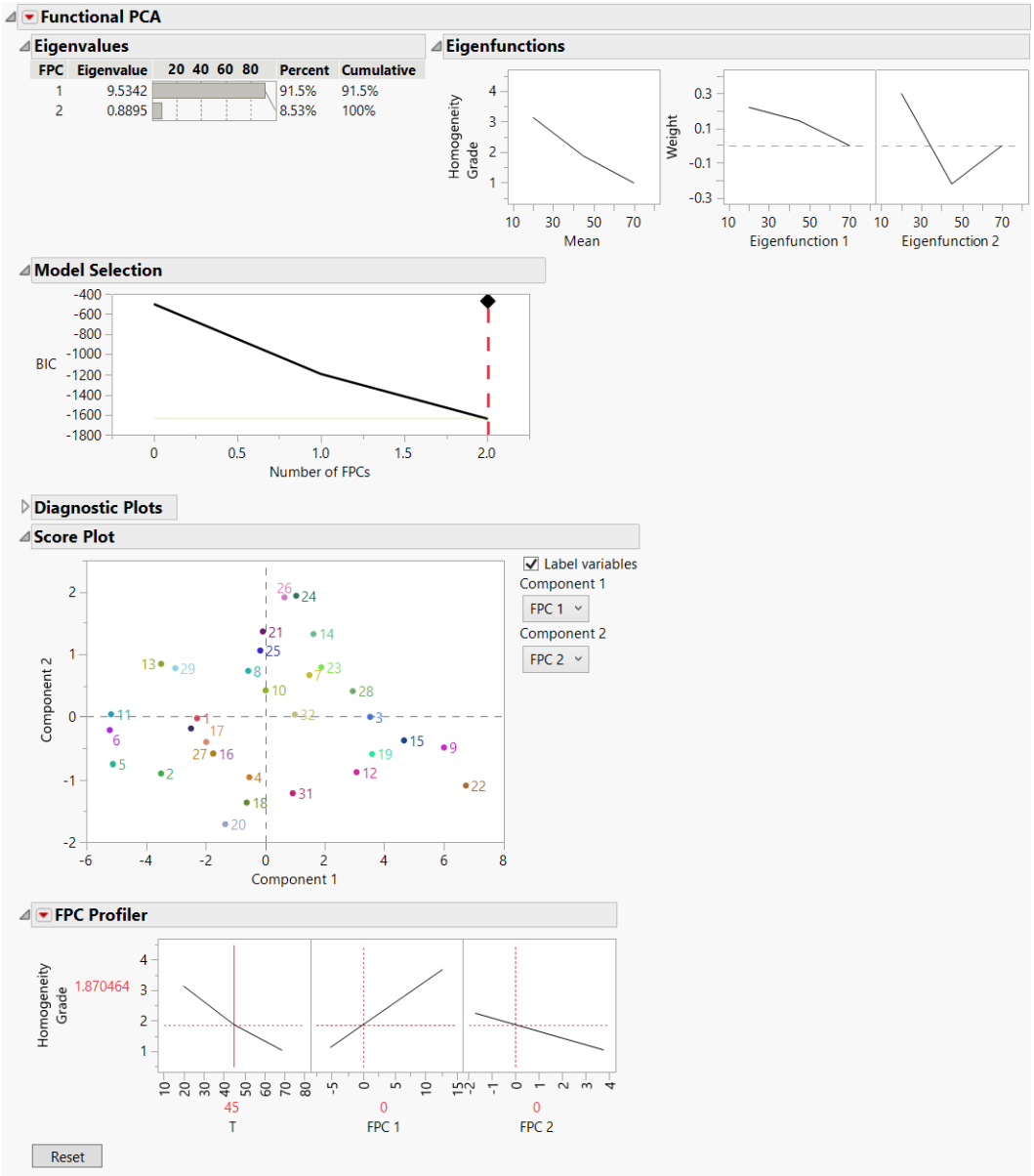
This example uses the Formulation For Homogeneity DOE.jmp data table to explore how combinations of ingredients effect the homogeneity grade of a compound across different temperatures. The three ingredients that form the compound are the primary ingredient labeled Active, Water, and Solvent. There are 32 combinations of ingredients; these combinations are denoted by the values in the Formulation column. For each formulation, the Homogeneity Grade is measured from 20 to 70 degrees Celsius at 5 degree intervals. By plotting these measurements across the different temperatures, you can form a curve for each formulation. Use the Functional Data Explorer platform to determine how the curves change depending on the values of Active, Water, and Solvent.

1. Select **Help > Sample Data Library** and open Functional Data/Formulation For Homogeneity DOE.jmp.
2. Select **Analyze > Specialized Modeling > Functional Data Explorer**.
3. In the Stacked Data Format tab, select Homogeneity Grade and click **Y, Output**.
4. Select T and click **X, Input**.
5. Select Formulation and click **ID, Function**.
6. Select Solvent, Active, and Water and click **Z, Supplementary**.

   These variables are not used in the initial functional data analysis. However, by specifying them here as supplementary variables, you can use them later when fitting models.
7. Click **OK**.
8. Click the Functional Data Explorer red triangle and select **Models > B-Splines**.

   The best fitting model is a linear spline model with one knot.

Figure 16.10 Functional PCA for Homogeneity Grade



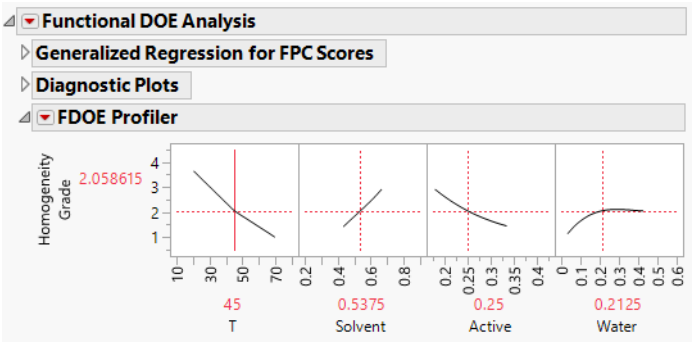
The Functional PCA report shows that two functional principal components are needed to explain the variability in the data. The FPC Profiler shows how Homogeneity Grade is affected by the FPCs across the different temperatures. However, there is no way to tell how the actual ingredients affect the Homogeneity Grade.

9. Click the red triangle next to B-Spline on Initial data and select **Functional DOE Analysis**.

The Functional DOE Analysis option fits a generalized regression model to each principal component score using the three supplementary variables as predictors. Because the data table contains a Model script, the models that are fit are based on the model defined in the script.

Note: If your data table does not contain a Model script, a two degree factorial model is fit.

Figure 16.11 FDOE Profiler for Homogeneity Grade



The FDOE Profiler enables you to explore how Homogeneity Grade changes based on the predictors you are actually interested in. In general, Homogeneity Grade seems to decrease as the amount of Active increases.

Note: In this example, the values of Active, Solvent, and Water are constrained to sum to 1. This constraint is reflected in the profiler.

JMP PRO Statistical Details for the Functional Data Explorer Platform

- "Functional Model Fits"
- "Function Summaries Details"

Functional Model Fits

All of the model fits in the Functional Data Explorer platform rely on basis function expansion. Basis functions are a set of independent functions. Any function, $f(t)$, can be approximated by taking a linear combination of K basis functions, denoted as ϕ_k. In general, a function is approximated as follows:

$$\hat{f}(t) = \sum_{k=1}^{K} c_k \phi_k(t)$$

where the c_k parameters are the basis coefficients. The amount of smoothing is determined by the number of basis functions, K. For more information about basis function expansion, see Ramsay and Silverman (2005).

JMP PRO Fourier Basis Model

The $k = 1,..., K$ functions that define the Fourier basis are defined such that $\phi_0 = 1$, $\phi_{2k-1} = \sin(r\omega t)$, and $\phi_{2k} = \cos(r\omega t)$. Then, the approximated function is defined as follows:

$$\hat{f}_i(t) = c_{i0} + c_{i1}\sin\omega t + c_{i2}\cos\omega t + c_{i3}\sin 2\omega t + c_{i4}\cos 2\omega t + \ldots$$

This approximates a periodic function with period A defined as $A = 2\pi/\omega$. The coefficients are a combination of fixed basis function coefficients and random coefficients by function. Each c is defined as follows:

$$c_{ik} = \beta_k + \alpha_{ik}$$

where β_k is the fixed coefficient for basis function k and α_{ik} is the random coefficient for basis function k for a specific functional process i. For the Fourier Basis model, the estimates for the β_k and α_{ik} parameters are found in the Basis Function Coefficients table and Random Coefficients by Function table, respectively.

Fourier basis models have an intercept term and an equal number of sine and cosine terms, which are referred to as Fourier pairs. Therefore, K is always odd. For example, if $K = 7$, there is an intercept term and three Fourier pairs.

Function Summaries Details

The Function Summaries report includes the following summary statistics: integrated difference, root integrated square error (RISE), and root integrated function squared (RIFS). These summary statistics are defined in this section. For all equations in this section, it is assumed that the input values have been aligned between 0 and 1. The following notation is used for all functions:

$\hat{f}_i(x)$ is the estimated function for curve i at input value x

$\hat{m}(x)$ is the estimated mean function at input value x

Integrated Difference

The integrated difference is the average difference between points on the ID specific curve and the overall mean curve. It can be used to determine if the ID specific curve falls above or below the mean curve, on average. The integrated difference for curve i is defined as follows:

$$\text{ID}_i = \int_0^1 (\hat{f}_i(x) - \hat{m}(x))dx$$

RISE

Root integrated square error (RISE) is the average distance between the ID specific curve and the overall mean curve. It can be used to determine which ID specific curves are most similar or most different from the overall mean curve. The RISE summary value for curve i is defined as follows:

$$\text{RISE}_i = \sqrt{\int_0^1 (\hat{f}_i(x) - \hat{m}(x))^2 dx}$$

RIFS

Root integrated function squared (RIFS) is a summary value for analyzing optimal curve scenarios. If you pre-subtract off an optimal curve, you can use the smallest values of RIFS to determine which functions are closest to the optimal curve. The RIFS summary value for curve i is defined as follows:

$$\text{RIFS}_i = \sqrt{\int_0^1 \hat{f}_i(x)^2 dx}$$

Chapter 17

Gaussian Process

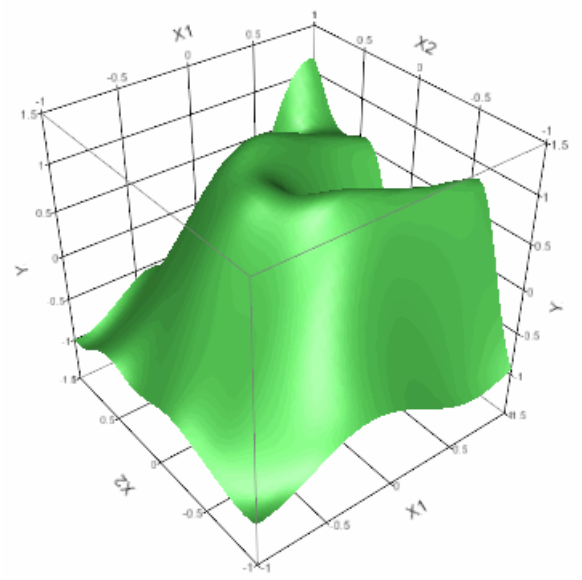
Fit Data Using Smoothing Models

Use the Gaussian Process platform to model the relationship between a continuous response and one or more predictors. These types of models are common in computer simulation experiments, such as the output of finite element codes, and they often perfectly interpolate the data. Gaussian processes can deal with these no-error-term models, in which the same input values always results in the same output value.

The Gaussian Process platform fits a spatial correlation model to the data. The correlation of the response between two observations decreases as the values of the independent variables become more distant.

One purpose for using this platform is to obtain a prediction formula that can be used for further analysis and optimization.

Figure 17.1 Gaussian Process Prediction Surface Example



Contents

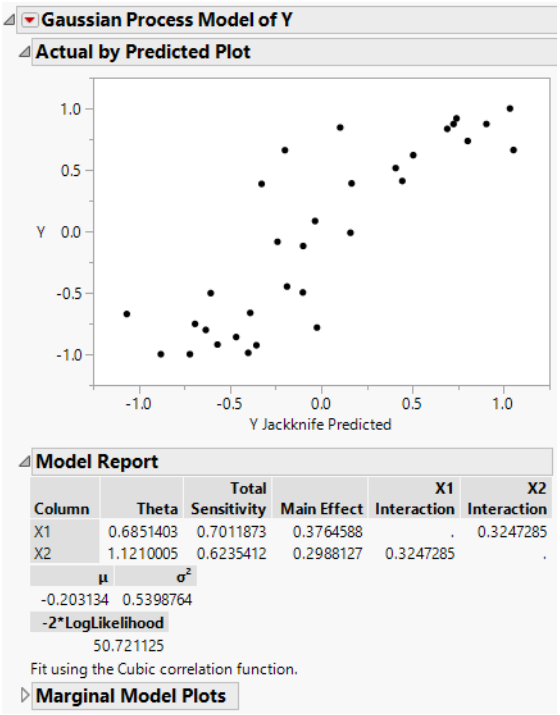
Example of Gaussian Process 311
Launch the Gaussian Process Platform 313
The Gaussian Process Report 314
Actual by Predicted Plot 314
Model Report 314
Marginal Model Plots 315
Gaussian Process Platform Options 315
Additional Examples of the Gaussian Process Platform 316
Example of a Gaussian Process Model 317
Example of Gaussian Process Model with Categorical Predictors 318
Statistical Details for the Gaussian Process Platform 320
Models with Continuous Predictors 320
Models with Categorical Predictors 321
Variance Formula Parameterization 322
Model Fit Details 322

Example of Gaussian Process

This example uses data from a space filling design in two variables with a deterministic equation for Y (the response). You can use the Gaussian Process platform to find the explanatory power of X1 and X2 on Y. You can view the equation for Y in the column formula.

1. Select **Help > Sample Data Library** and open 2D Gaussian Process Example.jmp.
2. Select **Analyze > Specialized Modeling > Gaussian Process.**
3. Select X1 and X2 and click **X**.
4. Select Y and click **Y**
5. Select **Correlation Type > Cubic.**
6. JMP PRO Deselect **Fast GASP**.
7. Click **OK**.

Figure 17.2 Gaussian Process Report

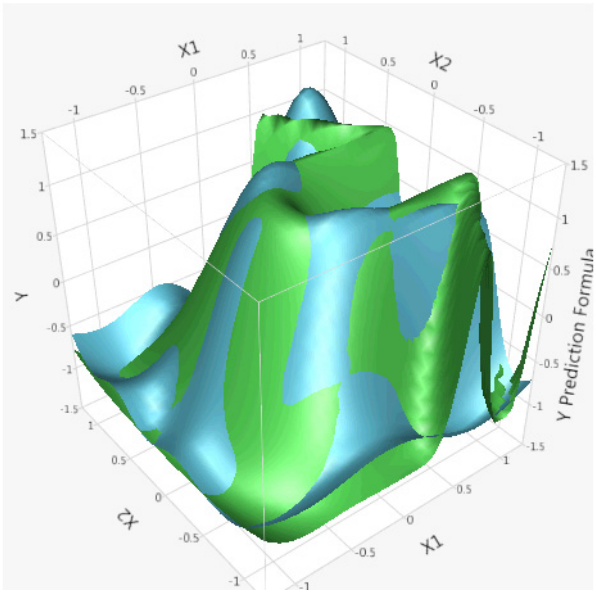


Note: The estimated parameters can be different due to different starting points in the minimization routine, the choice of correlation type, and the inclusion of a nugget parameter.

Now, visualize the fitted surface compared to the original surface.

8. Click the red triangle next to Gaussian Process Model of Y and select **Save Prediction Formula**.
9. Select **Graph > Surface Plot**.
10. Select X1 through Y Prediction Formula and click **Columns**.
11. Click **OK**.
12. In the Surface column, select **Both sides** for the Y Prediction Formula.

Figure 17.3 3D Surface Plot of the Actual and Predicted Ys

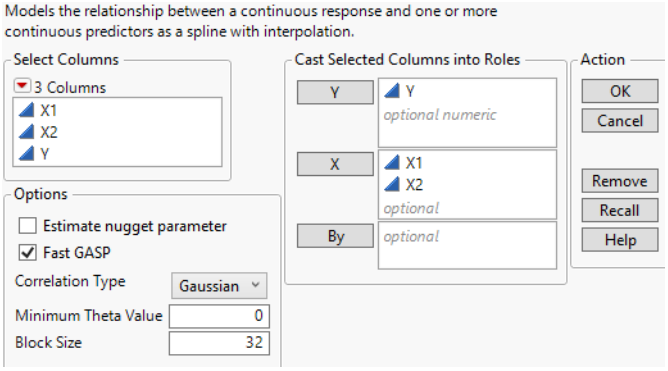


The two surfaces are similar. The impact of X1 and X2 on the response Y can be visualized. You can rotate the plot to view it from different angles. Marginal plots are another tool to use to understand the impact of the factors on the response.

Launch the Gaussian Process Platform

Launch the Gaussian Process platform by selecting **Analyze > Specialized Modeling > Gaussian Process**.

Figure 17.4 Gaussian Process Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y Assigns the continuous columns to analyze.

X Assigns the columns to use as explanatory variables. Categorical variables are allowed in JMP Pro when the Fast GASP option is specified.

Estimate Nugget Parameter Introduces a ridge parameter into the estimation procedure. A ridge parameter is useful if there is noise or randomness in the response, and you want the prediction model to smooth over the noise instead of perfectly interpolating.

JMP PRO **Fast GASP** Option to use the Fast GASP algorithm. Fast GASP breaks the Gaussian process model into small pieces (called blocks) to speed computation time. Blocks allow for the use of multiple CPUs and parallel processing.

Note: When there are more than 2,500 observations, the Fast GASP algorithm is required.

For more information about Fast GASP, see Parker (2015).

Correlation Type Choose the correlation structure for the model. The platform fits a spatial correlation model to the data, where the correlation of the response between two observations decreases as the values of the independent variables become more distant.

Gaussian Restricts the correlation between two points to always be nonzero, no matter the distance between the points.

Cubic Allows the correlation between two points to be zero for points that are far enough apart. This method is a generalization of a cubic spline.

JMP PRO The Fast GASP algorithm does not support the cubic correlation function.

Minimum Theta Value Sets the minimum theta value to use in the fitted model. The default is 0. The theta values are analogous to a slope parameter in regular regression models. Small theta values indicate that a variable has little influence on the predicted values.

JMP PRO **Block Size** The number of observations in each computational block used by the Fast GASP algorithm. There must be at least 25 observations per block and a maximum of the number of rows in the data set up to a maximum of 2,500.

The Gaussian Process Report

- "Actual by Predicted Plot"
- "Model Report"
- "Marginal Model Plots"

Actual by Predicted Plot

The Actual by Predicted plot shows the actual Y values on the Y axis and the jackknife predicted values on the X axis. One measure of goodness-of-fit is how well the points lie along the diagonal (Y = X) of the plot.

The jackknife values are not true jackknife values in that the model is not re-fit with the associated row for each Y excluded. Rather, the row is excluded from the prediction model for each associated Y but the correlation parameters retain the contribution of the row in them. For Gaussian processes that perfectly interpolate the data this jackknife procedure provides predictions that are not equal to the input.

Model Report

The Model Report shows a functional ANOVA table for the model parameter estimates. Specifically, it is an analysis of variance table where the variation is computed using a function-driven method.

Theta Gaussian Process model parameter estimates. See "Statistical Details for the Gaussian Process Platform".

Total Sensitivity Sum of the main effect and all interaction terms for each factor. It is a measure of the amount of influence a factor and all its two-way interactions have on the response variable.

Total variation is the integrated variability over the entire experimental space.

Main Effect The functional main effect of each factor is the integrated total variation due to that factor alone. The main effect is the ratio of the functional effect and the total variation for each factor in the model.

Interactions Functional interaction effects are computed in a similar way to main effects.

JMP PRO **Categorical Input** When the model includes categorical factors, a correlation matrix for each categorical factor is provided. The off-diagonal entries correspond to Gaussian Process model parameter estimates. See "Models with Categorical Predictors".

Mu and Sigma^2 The mean and variance model parameters.

Nugget The estimated nugget value. A nugget value is reported if you selected **estimate nugget parameter** in the Gaussian Process launch window. A nugget value is also reported if JMP has added a nugget parameter in order to avoid a singular covariance matrix.

-2LogLikelihood The estimated value of twice the negative log-likelihood function. See *Fitting Linear Models*.

Marginal Model Plots

A marginal plot appears for each factor in the model. For each plot, all factors except one are integrated out using a distribution that is uniform over the ranges of the other factors in the data. The marginal prediction of the response for the remaining factor of interest is drawn in blue. The data points for the observed response values are included to show how well the marginal model fits the data. For models with a small number of factors, these plots can be used as diagnostic plots.

Gaussian Process Platform Options

The Gaussian Process red triangle menu contains the following options:

Profiler Opens the standard Profiler. See *Profilers*.

Contour Profiler Opens the Contour Profiler. See *Profilers*.

Surface Profiler Opens the Surface Profiler. See *Profilers*.

Save Prediction Formula Creates a new prediction formula column in the active data table.

Save Variance Formula Creates a new variance formula column in the active data table.

JMP PRO **Publish Prediction Formula** Creates a prediction formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot".

JMP PRO **Publish Variance Formula** Creates a variance formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See "Formula Depot".

Save Jackknife Predicted Values Saves the jackknife predicted values to the active data table. These are the X-axis values for the Actual by Predicted Plot.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Examples of the Gaussian Process Platform

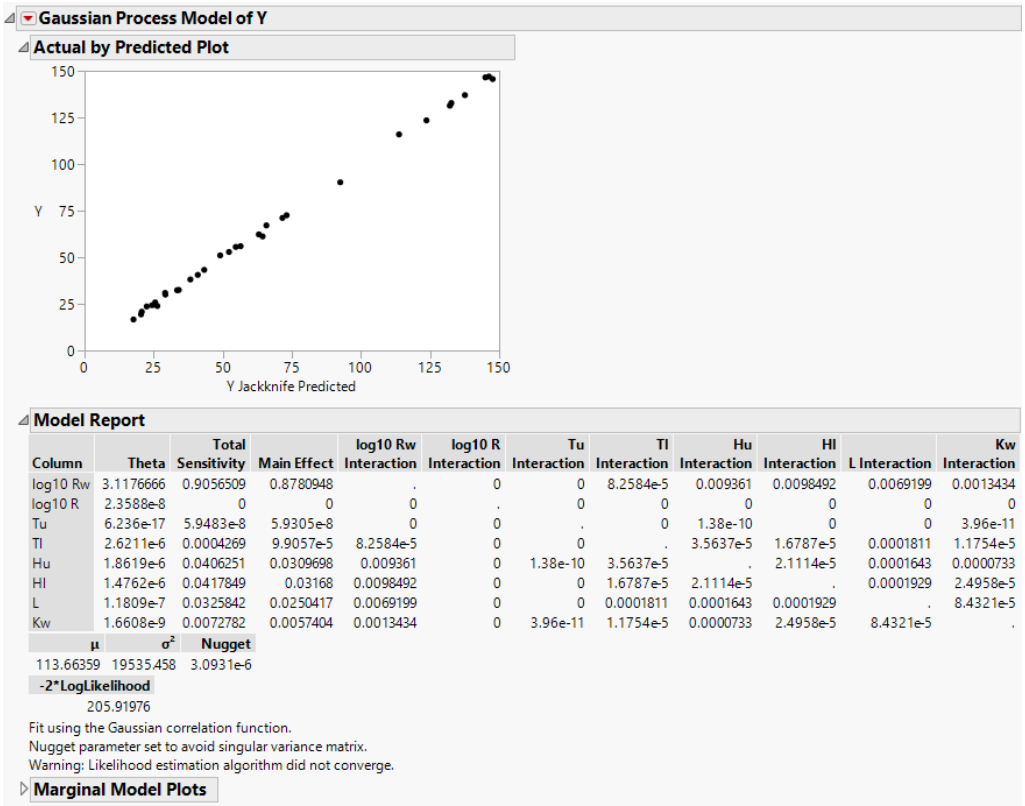
- "Example of a Gaussian Process Model"
- "Example of Gaussian Process Model with Categorical Predictors"

Example of a Gaussian Process Model

This example uses data that demonstrates the flow of water through a Borehole that is drilled from the ground surface through two aquifers. Given a specified engineering model the Gaussian process lets us understand the impact of factors included in the model on the response, Y.

1. Select **Help > Sample Data Library** and open Design Experiment/Borehole Latin Hypercube.jmp.
2. Select **Analyze > Specialized Modeling > Gaussian Process**.
3. Select log10 Rw through Kw and click **X**.
4. Select Y and click **Y**.
5. JMP PRO In JMP Pro, to run the analysis faster, leave the Fast GASP checked.
6. Click **OK**.

Figure 17.5 Borehole Latin Hypercube Report



Model Report

Column	Theta	Total Sensitivity	Main Effect	log10 Rw Interaction	log10 R Interaction	Tu Interaction	Tl Interaction	Hu Interaction	Hl Interaction	L Interaction	Kw Interaction
log10 Rw	3.1176666	0.9056509	0.8780948	.	0	0	8.2584e-5	0.009361	0.0098492	0.0069199	0.0013434
log10 R	2.3588e-8	0	0	0	.	0	0	0	0	0	0
Tu	6.236e-17	5.9483e-8	5.9305e-8	0	0	.	0	1.38e-10	0	0	3.96e-11
Tl	2.6211e-6	0.0004269	9.9057e-5	8.2584e-5	0	0	.	3.5637e-5	1.6787e-5	0.0001811	1.1754e-5
Hu	1.8619e-6	0.0406251	0.0309698	0.009361	0	1.38e-10	3.5637e-5	.	2.1114e-5	0.0001643	0.0000733
Hl	1.4762e-6	0.0417849	0.03168	0.0098492	0	0	1.6787e-5	2.1114e-5	.	0.0001929	2.4958e-5
L	1.1809e-7	0.0325842	0.0250417	0.0069199	0	0	0.0001811	0.0001643	0.0001929	.	8.4321e-5
Kw	1.6608e-9	0.0072782	0.0057404	0.0013434	0	3.96e-11	1.1754e-5	0.0000733	2.4958e-5	8.4321e-5	.

μ	σ^2	Nugget
113.66359	19535.458	3.0931e-6

-2*LogLikelihood
205.91976

Fit using the Gaussian correlation function.
Nugget parameter set to avoid singular variance matrix.
Warning: Likelihood estimation algorithm did not converge.

Marginal Model Plots

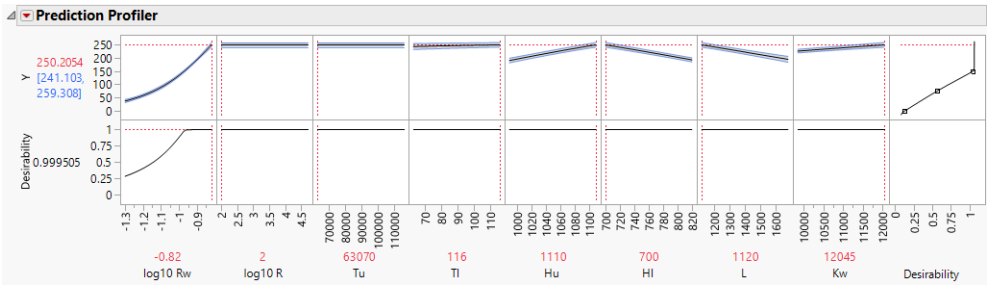
The data on the actual by predicted plot fall along the Y = X line, indicating that the Gaussian process prediction model is a good approximation of the true function. In the Model Report, you see that the first factor, log10 Rw, has the highest total sensitivity. The estimated total sensitivity for log10 Rw explains more than 90% of the variation in the response. Factors with small theta values have little (or no) impact on the prediction formula. Use the profiler to visualize the sensitivities.

7. Click the Gaussian Process Model of Y red triangle and select **Profiler**.

   The desirability functions automatically appear because the response, Y, has a Response Limits column property.

8. Click the Prediction Profiler red triangle and select **Optimize and Desirability > Maximize Desirability**.

Figure 17.6 Gaussian Process Model Profiler



The desirability function is set to maximize the response Y because Y has goal set to Maximize in the Response Limits column property. Maximizing the desirability functions identifies the values of the factors that maximize the response. The shaded bands represent the 95% confidence intervals.

Note: Your estimates can differ from those shown in Figure 17.5, which were found using the Fast GASP algorithm.

JMP PRO Example of Gaussian Process Model with Categorical Predictors

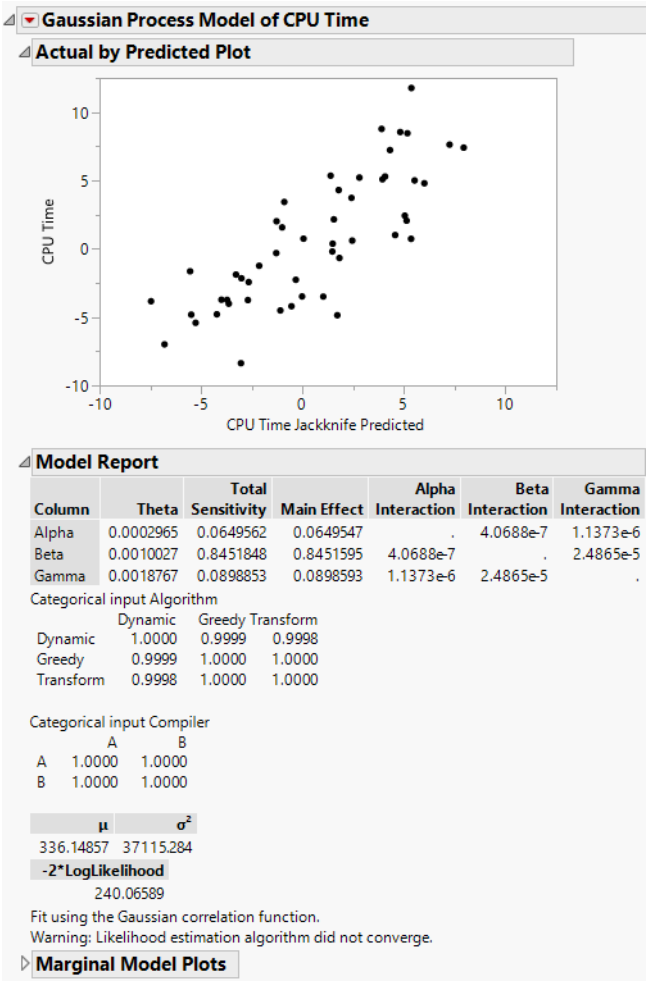
This example uses the Algorithm Data.jmp sample data table. These data are simulated CPU times from a 50 run space filling designed experiment. The Algorithm Factors.jmp sample data table provides the factors and settings for the design. The design has three continuous and two categorical factors. The goal is to predict CPU Time using a Gaussian Process model that contains both continuous and categorical factors.

1. Select **Help > Sample Data Library** and open Design Experiment/Algorithm Data.jmp.
2. Select **Analyze > Specialized Modeling > Gaussian Process**.
3. Select Alpha through Compiler and click **X**.

4. Select CPU Time and click **Y**.
5. JMP PRO To run the analysis, leave the Fast GASP checked. Click **OK**.

Note: The Fast GASP option must be used for models that contain categorical factors. See "Models with Categorical Predictors".

Figure 17.7 Algorithm Data Report



The actual by predicted plot shows a strong correlation between the actual and predicted CPU times. This in an indication that the Gaussian process prediction model is a good approximation of the true function. In the Model Report, the Beta predictor has the highest total sensitivity. This indicates that of the continuous predictors, Beta explains the most variation in the response. There is a separate Categorical Input matrix for each of the categorical predictors, Algorithm and Compiler. These matrices are correlation matrices and

show the correlation between levels for each categorical predictor. The off-diagonals of the matrices are the τ parameters.

Statistical Details for the Gaussian Process Platform

- "Models with Continuous Predictors"
- "Models with Categorical Predictors"
- "Variance Formula Parameterization"
- "Model Fit Details"

Models with Continuous Predictors

If the Gaussian Process model contains only continuous predictors, the Gaussian Process platform implements two possible correlation structures, the Gaussian and the Cubic.

The Gaussian correlation structure uses the product exponential correlation function with a power of 2 as the estimated model. This model assumes that Y is normally distributed with mean μ and covariance matrix $\sigma^2\mathbf{R}$. The elements of the **R** matrix are defined as follows:

$$r_{ij} = \exp\left(-\sum_{k=1}^{K} \theta_k (x_{ik} - x_{jk})^2\right)$$

where

K = # of continuous predictors

θ_k = theta parameter for the k^{th} predictor

x_{ik} = the value of the k^{th} predictor for subject i

x_{jk} = the value of the k^{th} predictor for subject j

The Cubic correlation structure also assumes that Y is normally distributed with mean μ and covariance matrix $\sigma^2\mathbf{R}$. The **R** matrix consists of the following elements:

$$r_{ij} = \prod_k \rho(d;\theta_k)$$

where

$$d = x_{ik} - x_{jk}$$

$$\rho(d;\theta) = \begin{cases} 1 - 6(d\theta)^2 + 6(|d|\theta)^3, & |d| \le \frac{1}{2\theta} \\ 2(1 - |d|\theta)^3, & \frac{1}{2\theta} < |d| \le \frac{1}{\theta} \\ 0, & \frac{1}{\theta} < |d| \end{cases}$$

See Santer (2003). The theta parameter used in the Cubic correlation structure is the reciprocal of the parameter often used in the literature. The reciprocal is used so that when theta has no effect on the model, then rho has a value of zero, rather than infinity.

JMP PRO Models with Categorical Predictors

If the Gaussian Process model includes categorical predictors, the Gaussian correlation structure is used for the correlation structure. The elements of the **R** matrix are defined as follows:

$$r_{ij} = \left(\prod_{p=1}^{P} \tau_{p_{ij}} \right) \exp\left(- \sum_{k=1}^{K} \theta_k (x_{ik} - x_{jk})^2 \right)$$

where

K = # of continuous predictors

P = # of categorical predictors

θ_k = theta parameter for the k^{th} continuous predictor

x_{ik} = the value of the k^{th} continuous predictor for subject i

x_{jk} = the value of the k^{th} continuous predictor for subject j

$\tau_{p_{ij}}$ = the correlation between the observed level of predictor p for subject i and the observed level of predictor p for subject j

There is a τ parameter for each combination of levels of a categorical variable, where τ_{ij} corresponds to the unique combination formed by the observed levels of subject i and subject j. Thus, the covariance element, r_{ij}, depends on the combination of levels of the categorical predictors obtained from the i^{th} and j^{th} observations. See Qian et al. (2012).

Variance Formula Parameterization

The saved variance formula uses the previously defined parameterization of **R**, except when the model includes categorical predictors. When the Gaussian Process model includes categorical predictors, the saved variance formula uses the following parameterization of **R**:

$$r_{ij} = \exp\left(-\sum_{k=1}^{K} \theta_k (x_{ik} - x_{jk})^2 - \sum_{p=1}^{P} \phi_{p_{ij}} \right)$$

where $\phi_{p_{ij}} = -\ln(\tau_{p_{ij}})$ and all other variables are as previously defined.

Model Fit Details

The model parameters are fit via maximum likelihood. The fitted parameters are provided in the platform report. These are the parameters:

- μ is the Gaussian Process mean,
- σ^2 is the Gaussian Process variance,
- Theta corresponds to the values of θ_k in the definition of **R**.
- The off-diagonals of the categorical input correlation matrices correspond to the values of τ_{pij} in the definition of **R**.

Note: If your report contains the note **Nugget parameters set to avoid singular variance matrix**, JMP has added a ridge parameter to the variance matrix so that it is invertible.

Chapter 18

Time Series Analysis

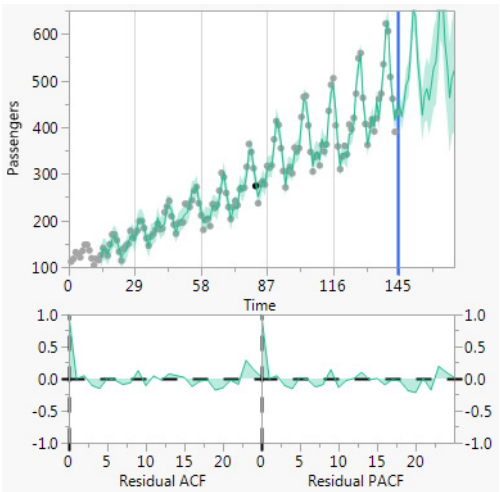
Fit Time Series Models and Transfer Functions

The Time Series platform enables you to explore, analyze, and forecast univariate time series. A time series is a set of observations taken over a series of equally spaced time periods. Observations that are close together in time are typically correlated. Time series methodology takes advantage of this dependence between observations to better predict what the series will look like in the future.

Characteristics that are common in time series data include seasonality, trend, and autocorrelation. The Time Series platform provides options to handle these characteristics. Graphs such as variograms, autocorrelation plots, partial autocorrelation plots, and spectral density plots can be used to identify the type of model appropriate for describing and predicting (forecasting) the time series. There are also several decomposition methods in the platform that enable you to remove seasonal or general trends in the data to simplify the analysis. Alternatively, the platform can fit more sophisticated ARIMA models and State Space Smoothing models that have the ability to incorporate seasonality and long term trends all in one model. You can also perform a Box-Cox transformation and analyze and model the transformed series.

There are several methods to assess the forecasting performance of models. The Forecast on Holdback feature partitions the time series into a training portion to build models and a and holdback portion to assess forecasting performance.

Figure 18.1 Forecast Plot



Contents

Overview of the Time Series Platform . . . 325
Example of the Time Series Platform . . . 326
Launch the Time Series Platform . . . 328
The Time Series Analysis Report . . . 330
Time Series Graph . . . 330
Time Series Basic Diagnostics Chart . . . 331
Time Series Platform Options . . . 333
Time Series Diagnostics . . . 333
Differencing and Decomposition . . . 334
ARIMA and Seasonal ARIMA Models . . . 337
Smoothing Models . . . 338
State Space Smoothing Models . . . 339
Transfer Function Models . . . 340
Smoothing Model Specification Windows . . . 343
Reports . . . 345
Difference Report . . . 345
Decomposition Reports . . . 346
Model Comparison Report . . . 348
ARIMA and Seasonal ARIMA Model Report . . . 351
State Space Smoothing Report . . . 355
Transfer Function Report . . . 358
Spectral Density Report . . . 359
Additional Examples of the Time Series Platform . . . 360
Example of Creating Time ID Column . . . 360
Example Using Box-Cox Transformation . . . 365
Example Using a Holdback Set . . . 367
Statistical Details for the Time Series Platform . . . 369
Statistical Details for Spectral Density . . . 370
Statistical Details for X-11 Decomposition . . . 370
Statistical Details for Exponential Smoothing Models . . . 371
Statistical Details for ARIMA Models . . . 375
Statistical Details for Transfer Functions . . . 376

Overview of the Time Series Platform

A time series is a set $y_1, y_2, \ldots, y_N$ of observations that are observed over a series of equally spaced time periods. Some examples of time series data include quarterly sales reports, monthly average temperatures, and counts of sunspots. The Time Series platform enables you to explore patterns and trends found in these types of data. You can then use these patterns and trends to forecast, or predict, into the future.

Characteristics that are common in time series data include seasonality, trend, and autocorrelation. *Seasonality* refers to patterns that occur over a known period of time. For example, data that are collected monthly might look similar in summer months across all years of data collection. *Trend* refers to long term movements of a series, such as gradual increases or decreases of values across time. *Autocorrelation* is the degree to which each point in a series is correlated with earlier values in the series.

There are many different models and forecasting methods available in the Time Series platform. However, not all methods can handle trend or seasonality. In order to choose an appropriate model, it is essential to determine which characteristics are present in the series. The Time Series platform provides graphs such as variograms, autocorrelation plots, partial autocorrelation plots, and spectral density plots that can be used to identify the type of model appropriate for describing and forecasting the evolution of the time series. There are also several differencing and decomposition methods in the platform that enable you to remove seasonal or general trends in the data to explore and simplify the analysis. You can also view and apply a Box-Cox transformation to your data.

Alternatively, the platform can fit more sophisticated models that can incorporate seasonality and long term trends. One such model in the platform that has this ability is Winter's Additive Method, which is an advanced exponential smoothing model. In addition, the platform can fit AutoRegressive Integrated Moving Average (ARIMA) models and State Space Smoothing models. Both of these types of models are the most statistically complex, but also provide the most flexibility. Advanced exponential smoothing, ARIMA, and State Space Smoothing models are harder to interpret, but they are excellent tools for forecasting.

The Time Series platform can also fit transfer function models when supplied with an input series.

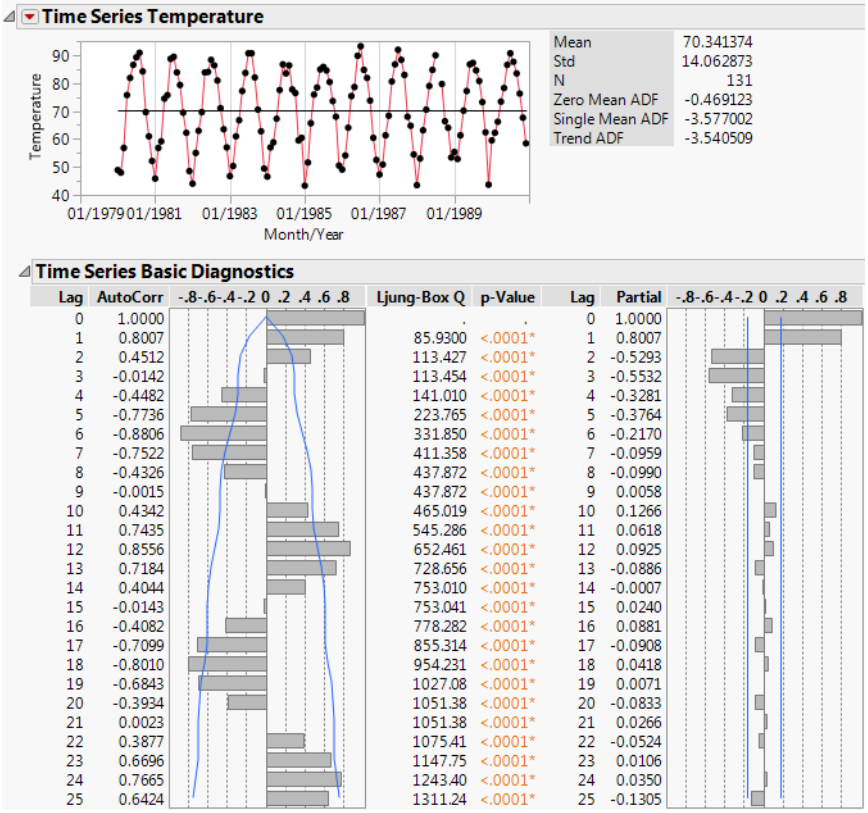
Example of the Time Series Platform

This example uses the Raleigh Temps.jmp sample data table, which contains maximum monthly temperatures measured in degrees Fahrenheit from 1980 to 1990. Use the Time Series platform to examine the series and predict the maximum monthly temperatures for the next two years.

1. Select **Help > Sample Data Library** and open Time Series/Raleigh Temps.jmp.
2. Select **Analyze > Specialized Modeling > Time Series**.
3. Select Temperature and click **Y, Time Series**.
4. Select Month/Year and click **X, Time ID**.
5. In the box next to **Forecast Periods**, type 24.

   This is the number of future periods that are forecast by the models fit to the data. You want to predict the monthly temperature for the next two years, which is 24 months.
6. Click **OK**.

Figure 18.2 Time Series Analysis Report for Raleigh Temps.jmp

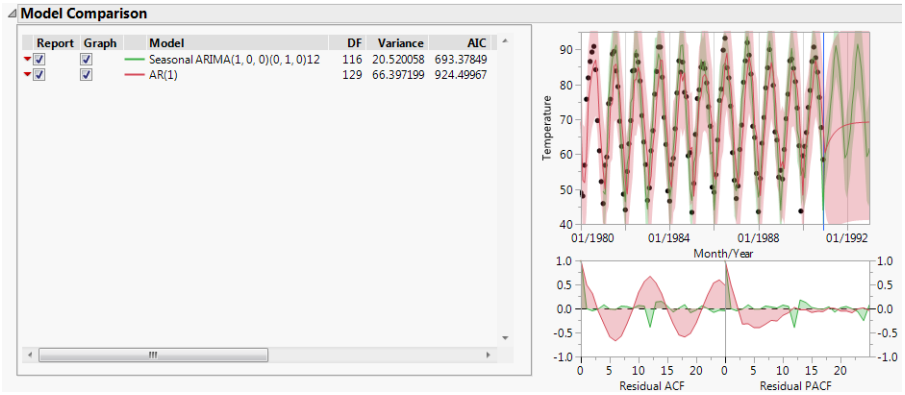


Lag	AutoCorr	Ljung-Box Q	p-Value	Lag	Partial
0	1.0000	.	.	0	1.0000
1	0.8007	85.9300	<.0001*	1	0.8007
2	0.4512	113.427	<.0001*	2	-0.5293
3	-0.0142	113.454	<.0001*	3	-0.5532
4	-0.4482	141.010	<.0001*	4	-0.3281
5	-0.7736	223.765	<.0001*	5	-0.3764
6	-0.8806	331.850	<.0001*	6	-0.2170
7	-0.7522	411.358	<.0001*	7	-0.0959
8	-0.4326	437.872	<.0001*	8	-0.0990
9	-0.0015	437.872	<.0001*	9	0.0058
10	0.4342	465.019	<.0001*	10	0.1266
11	0.7435	545.286	<.0001*	11	0.0618
12	0.8556	652.461	<.0001*	12	0.0925
13	0.7184	728.656	<.0001*	13	-0.0886
14	0.4044	753.010	<.0001*	14	-0.0007
15	-0.0143	753.041	<.0001*	15	0.0240
16	-0.4082	778.282	<.0001*	16	0.0881
17	-0.7099	855.314	<.0001*	17	-0.0908
18	-0.8010	954.231	<.0001*	18	0.0418
19	-0.6843	1027.08	<.0001*	19	0.0071
20	-0.3934	1051.38	<.0001*	20	-0.0833
21	0.0023	1051.38	<.0001*	21	0.0266
22	0.3877	1075.41	<.0001*	22	-0.0524
23	0.6696	1147.75	<.0001*	23	0.0106
24	0.7665	1243.40	<.0001*	24	0.0350
25	0.6424	1311.24	<.0001*	25	-0.1305

The Time Series graph shows that the series is cyclic. This cyclic component is also apparent in the autocorrelation chart. Points that are 1 lag apart are positively correlated, with an AutoCorr value of 0.8007. As points become farther apart, they become negatively correlated, then positively correlated again, and then the pattern repeats. The Time Series graph and the autocorrelation chart provide evidence of seasonality in the time series.

7. Click the Time Series red triangle and select **ARIMA**.
8. Set *p*, the autoregressive order, to 1 because the series showed evidence of autocorrelation.
9. Click **Estimate**.
10. Click the Time Series red triangle and select **Seasonal ARIMA**.
11. In the ARIMA box, set *p*, the autoregressive order, to 1 because the series showed evidence of autocorrelation.
12. In the Seasonal ARIMA box, set *D*, the seasonal differencing order, to 1 because the series showed evidence of seasonality.
13. Click **Estimate**.
14. In the Model Comparison table, check the box under Graph for both models.

Figure 18.3 Model Comparison Table for Raleigh Temps.jmp

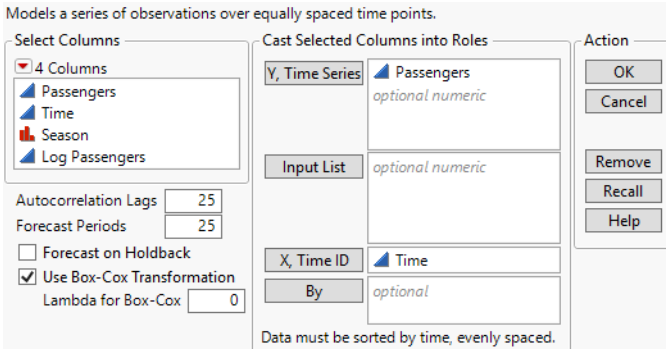


The Model Comparison table is sorted by the AIC statistic, in decreasing order. This means that the best fitting model appears at the top of the report. The AIC value for the seasonal ARIMA model (693.4) is much smaller than the value for the regular ARIMA model (924.5). The graph shows that while the ARIMA model predicts the observed points relatively well, the residuals are larger than those from the Seasonal ARIMA model. Also, the Seasonal ARIMA model has more realistic predictions for future observations with narrower prediction intervals. These results make sense since the series showed evidence of a seasonal component.

Launch the Time Series Platform

Launch the Time Series platform by selecting **Analyze > Specialized Modeling > Time Series**. The Time Series launch window for the Seriesg.jmp sample data table is shown in Figure 18.4.

Figure 18.4 The Time Series Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Time Series platform launch window contains the following options:

Y, Time Series Assigns one or more columns as time series variables. Displayed on the Y axis.

Input List Assigns one or more columns as input series variables. Displayed in the Input Time Series Panel and used in transfer function models. The input series variable must be numeric, either as a time series or an indicator.

X, Time ID Assigns one variable for labeling the time axis (X axis). If no variable is specified for Time ID, the row number is used instead.

Note: If you use an X, Time ID variable, you can specify the time frequency by using the Time Frequency column property. You can choose Annual, Quarterly, Monthly, Weekly, Daily, Hourly, By Minute, and By Second. This helps JMP determine the spacing of the data when plotting the forecast values. If no frequency is specified, the data is treated as equally spaced numeric data.

Caution: It is assumed that the observations of the variable assigned to **X, Time ID** are equally spaced. However, the Time Series platform checks only whether the time stamps are increasing. The platform does not check if the observations are equally spaced.

By Assigns a column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Note: If you use a By variable, you might have to change the number of autocorrelation lags depending on how many observations there are for each level of the By variable. The number of lags must be greater than one but less than the number of observations per level.

Autocorrelation Lags Specifies the number of lags to use in computing the autocorrelations and partial correlations. This is the maximum number of periods between points used in the computation of these correlations. It must be greater than one but less than the number of rows. The default number of lags is 25.

Tip: A commonly used rule for the maximum number of lags is $n/4$, where n is the number of observations.

Forecast Periods Specifies the number of observations that are forecast using each model fitted to the data. The default number of forecasts is 25.

- If you select the Forecast on Holdback option, the Forecast Periods value specifies the number of observations at the end of the series that are not included in the model fitting. These values are then forecast using the fitted model.
- If you do not select the Forecast on Holdback option, the Forecast Periods value specifies the number of future observations that are forecast after the end of the series.

Forecast on Holdback Determines whether forecasts are made on future observations or on the holdback observations. If the Forecast on Holdback option is selected, the forecasts are made on the holdback set that is determined by the number specified in the Forecast Periods option.

Use Box-Cox Transformation Transforms the original data using a Box-Cox transformation with the lambda that is specified in the Lambda for Box-Cox option. When this option is selected, all analyses in the Time Series report are performed on the transformed data. However, all forecasts are transformed back and reported on the original scale.

Lambda for Box-Cox (Available only when Use Box-Cox Transformation is selected.) Specifies the lambda parameter used for the Box-Cox transformation of the original data.

Note: In order to preserve the order and spacing of a time series, the Time Series platform treats excluded rows in the data table as missing values.

The Time Series Analysis Report

The initial Time Series report displays the time series graph, summary statistics and tests for the time series variable, and a basic diagnostics report. If a column is specified for Input List in the launch window, the Transfer Function Analysis and Input Time Series Panel reports are shown. Both the Transfer Function Analysis report and the Input Time Series Panel report contain the same initial information as the Time Series report.

- "Time Series Graph"
- "Time Series Basic Diagnostics Chart"

Time Series Graph

The Time Series graph plots each times series by the time ID. If no time ID is specified, the row number is used instead. If a holdback set is specified, there is a vertical reference line on the graph that separates the training data from the holdback data.

The platform also performs several tests for stationarity using *Augmented Dickey-Fuller* (ADF) tests. The following tests and summary statistics are displayed next to the time series graph:

Lambda for Box-Cox (Appears only if a Box-Cox Transformation is specified.) The value of lambda used in the Box-Cox transformation.

Mean The sample mean.

SD The sample standard deviation.

N The series length.

Zero Mean ADF A test against a random walk with zero mean, which is defined as follows:

$$x_t = \phi x_{t-1} + e_t$$

Single Mean ADF A test against a random walk with a non-zero mean, which is defined as follows:

$$x_t - \mu = \phi(x_{t-1} - \mu) + e_t$$

Trend ADF A test against a random walk with a non-zero mean and a linear trend, which is defined as follows:

$$x_t - \mu - \beta t = \phi[x_{t-1} - \mu - \beta(t-1)] + e_t$$

Time Series Basic Diagnostics Chart

The information that is shown in the Time Series Basic Diagnostics chart depends on the Time Series Report red triangle menu options. The red triangle menu options that show or hide information from the diagnostics chart are Autocorrelation, Partial Autocorrelation, Variogram, and AR Coefficients. By default, Autocorrelation and Partial Autocorrelation are shown.

Autocorrelation Chart

The Autocorrelation option shows or hides the following columns in the Time Series Basic Diagnostics chart:

Lag The number of periods between points.

Note: The number of lags begins with 0 as the default. To compute correlations beginning with lag 1, modify the JMP preferences before generating the graph. Select **File > Preferences > Platforms > Time Series,** and then select **Suppress Lag 0 in ACF and PACF**.

AutoCorr The autocorrelation for the *k*th lag, which is computed as follows:

$$r_k = \frac{c_k}{c_0} \quad \text{where} \quad c_k = \frac{1}{N} \sum_{t=k+1}^{N} (y_t - \bar{y})(y_{t-k} - \bar{y})$$

and $\bar{y}$ is the mean of the N non-missing points in the time series. By definition, the first autocorrelation (lag 0) always has length 1.

The bars graphically depict the autocorrelations. The blue curves represent twice the large-lag standard error (± 2 standard errors), which is computed as follows:

$$\mathrm{SE}_k = \sqrt{\frac{1}{N}\left(1 + 2\sum_{i=1}^{k-1} r_i^2\right)}$$

Ljung-Box Q Used to test whether a group of autocorrelations is significantly different from zero, or to test that the residuals from a model can be distinguished from white noise. Q is the test statistic.

p-Value The p-value from the Ljung-Box test.

Partial Autocorrelation Chart

The Partial Autocorrelation option shows or hides the following columns in the Time Series Basic Diagnostics chart:

Lag The number of periods between points.

Partial The partial autocorrelation for the *k*th lag.

The bars graphically depict the partial autocorrelations. The blue lines represent ± 2 standard errors for approximate 95% prediction limits, where the standard error is computed as follows:

$$SE_k = \frac{1}{\sqrt{n}} \quad \text{for all } k$$

Variogram Chart

The Variogram option shows or hides the following columns in the Time Series Basic Diagnostics chart:

Lag The number of periods between points.

Variogram The variogram measures the variance of the differences of points *k* lags apart and compares it to that for points one lag apart. The variogram is computed from the autocorrelations as follows:

$$V_k = \frac{1 - r_{k+1}}{1 - r_1}$$

where r_k is the autocorrelation at lag *k*.

AR Coefficients Chart

The AR Coefficients option shows or hides the following columns in the Time Series Basic Diagnostics chart:

Lag The number of periods between points.

AR Coef The coefficients approximate those that you would obtain from fitting a high-order, purely autoregressive model.

Time Series Platform Options

- "Time Series Diagnostics"
- "Differencing and Decomposition"
- "ARIMA and Seasonal ARIMA Models"
- "Smoothing Models"
- "State Space Smoothing Models"
- "Transfer Function Models"
- "Smoothing Model Specification Windows"

Time Series Diagnostics

Graph Shows a submenu of options to control the time series plot appearance.

Time Series Graph Shows or hides the time series graph.

Show Points Shows or hides the points in the time series graph.

Connecting Lines Shows or hides the lines connecting the points in the time series graph.

Mean Line Shows or hides a horizontal line in the time series graph that depicts the mean of the time series.

Autocorrelation Shows or hides the Autocorrelation plot in the Time Series Basic Diagnostics Chart. The autocorrelation graph describes the correlation between all pairs of points in the time series for a given separation in time (lag). See "Autocorrelation Chart".

Tip: The autocorrelation graph of the sample is often called the *sample autocorrelation function*.

Partial Autocorrelation Shows or hides the Partial Autocorrelation plot in the Time Series Basic Diagnostics Chart. The partial autocorrelation graph describes the partial correlation between all the pairs of points in the time series for a given separation in time (lag). See "Partial Autocorrelation Chart".

Tip: The Autocorrelation and Partial Autocorrelation graphs can help you determine whether the time series is stationary (meaning it has a fixed mean and standard deviation over time) and what model might be appropriate to fit the time series.

Variogram Shows or hides the graph of the variogram in the Time Series Basic Diagnostics Chart. See "Variogram Chart".

AR Coefficients Shows or hides the graph of the least squares estimates of the autoregressive (AR) coefficients in the Time Series Basic Diagnostics Chart. See "AR Coefficients Chart".

Spectral Density Shows or hides graphs of the spectral density as a function of period and frequency. The spectral density option also displays the White Noise test report, which gives results from two tests on the data. See "Spectral Density Report" and "Statistical Details for Spectral Density".

Differencing and Decomposition

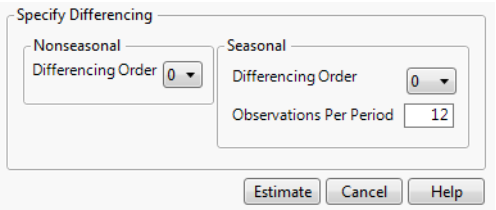
Difference Shows the Differencing Specification window (Figure 18.5). The window enables you to specify the differencing operation that you want to apply to the time series. Differencing the values in a time series can transform a nonstationary series into a stationary series. The differenced series is given by the following equation:

$$w_t = (1 - \mathbf{B})^d (1 - \mathbf{B}^s)^D y_t$$

where t is the time index and $\mathbf{B}$ is the backshift operator defined by $\mathbf{B}y_t = y_{t-1}$.

Note: Many time series do not exhibit a fixed mean, such as time series with trend or seasonality. Such nonstationary series are not suitable for description by time series models that assume a stationary time series such as ARMA models. Removing the trend and/or seasonality creates a differenced series that is stationary and enables you to describe the series using the models that assume stationarity.

Figure 18.5 Differencing Specification Window



The Differencing Specification window enables you to specify the Nonseasonal Differencing Order, d, the Seasonal Differencing Order, D, and the number of Observations per Period, s. Selecting zero for the value of the differencing order is equivalent to no differencing of that kind. Each time you specify a differencing operation and click **Estimate**, a new Difference Report is displayed in the report window. See "Additional Examples of the Time Series Platform".

Decomposition (Not available when either Forecast on Holdback or Use Box-Cox Transformation are selected in the launch window.) Shows a submenu of decomposition methods. Decomposition of time series data isolates and removes linear trends and seasonal cycles from a time series. This can help with better model estimation. Three Decomposition options are provided.

Remove Linear Trend Estimates the linear trend of the time series using a linear regression model and removes the linear trend from the data. A Time Series report for the detrended series is added to the report window, along with the linear trend information. See "The Time Series Analysis Report" and "Linear Trend Report".

Remove Cycle Estimates the cyclic component of a time series using a single cosine wave and then removes the cyclic component from the data. When you select the Remove Cycle option, the Define Cycle dialog appears. This dialog window enables you to specify the number of units per cycle and indicate whether a constant should be subtracted from the data. A Time Series report for the decycled series is added to the report window, along with the cycle information. See "The Time Series Analysis Report" and "Cycle Report".

X11 Removes trend and seasonal effects using the X-11 method developed by the US Bureau of the Census (Shiskin et. al. 1967). For more information about the X-11 method, see "Statistical Details for X-11 Decomposition". When selected, the Select Decomposition Type dialog appears. This dialog window enables you to specify a multiplicative or additive X-11 adjustment. Once you click **OK**, an X11 report is added to the report window. See "X11 Report".

The X11 option is available only for monthly or quarterly data. The X, Time ID column must contain numeric values equally spaced by month or quarter without any gaps or missing values. JMP returns an error if you request X11 for a time column that does not satisfy these requirements. For an example of an appropriate time column for an X11 analysis, see "Create Appropriate Time ID Column".

Note: When you select the **Remove Linear Trend** or the **Remove Cycle** options, JMP adds a column to the data table that contains the detrended or decycled data. If this column is already present in the data table when you select the option, JMP overwrites the existing column.

Tip: Typically, you would begin decomposition by removing any linear trend, and then removing long cycles, such as a 12-month cycle. Then you could start removing short cycles, such as 6-month cycles.

Show Lag Plot Shows or hides a plot with observations at time *t* on the Y axis and observations at time *t* +/- *p* on the X axis. The +/- *p* is known as the lag. This plot is useful in determining how an observation at time *t* is related to another observation at time *t* +/- *p*. If

there is not an identifiable structure to the plot, the observations are not related. However, if there is a structure to the plot, this indicates that there is some relationship between observations across time. Identifying the structure helps when building a time series model.

Show Box-Cox Transformation Plot (Not available if you specified Use Box-Cox Transformation in the launch window.) Shows or hides the Box-Cox Transformation plot. The Box-Cox transformation is a power transformation based on the Lambda parameter (λ) and uses the following formula:

$$Y_\lambda = \begin{cases} \frac{y^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln(y) & \text{if } \lambda = 0 \end{cases}$$

The Box-Cox Transformation Plot shows a graph of the transformed time series and options for setting the Lambda parameter (λ). The Lambda box shows the current value of the parameter. You can also use the Lambda slider to change the value of the parameter, which then automatically updates the plot. The Low and High boxes set the range of the slider.

Tip: Use the Box-Cox Transformation plot to see the transformed time series for different values of Lambda. Once you decide on a value, relaunch the analysis and select **Use Box-Cox Transformation** and enter the desired Lambda next to Lambda for Box-Cox. This runs the Time Series platform on the transformed series.

Cross Correlation (Available only in the Transfer Function Analysis red triangle menu.) Shows or hides a cross-correlation plot to the report. The length of the plot is twice that of an autocorrelation plot, or 2 × ACF length + 1. The plot includes plots of the output series versus all input series, in both numerical and graphical forms. The blue lines indicate two standard errors.

Note: For the cross correlation plot, the standard errors are calculated under the null hypothesis as 1/sqrt(n - k), where n is the number of nonmissing data values and k is the number of autocorrelation lags.

Prewhitening (Available only in the Input Series red triangle menu.) Shows the Prewhitening Specification window that enables you to set the prewhitening order. Prewhitening is a technique used to help identify the transfer function model. This technique involves fitting an ARIMA model to the input series such that the residuals are equivalent to white noise. The same model is then used to estimate the output series. You can use the cross correlation between the filtered input and filtered output series to

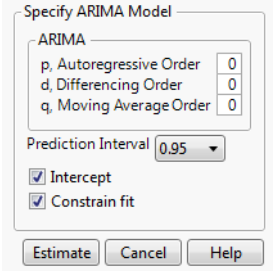
determine an appropriate lag for the transfer function model. For information about prewhitening, see Box et al. (1994).

ARIMA and Seasonal ARIMA Models

ARIMA Shows the ARIMA Specification window, which enables you to specify the ARIMA model that you want to fit. An ARIMA model predicts future values of a time series by a linear combination of its past values and a series of errors (also known as *random shocks* or *innovations*). The ARIMA model performs a maximum likelihood fit of the specified ARIMA model to the time series. See "ARIMA Model".

Note: An ARIMA model is commonly denoted ARIMA(p,d,q). If any of p, d, or q are zero, the corresponding letters are often dropped. For example, if p and d are zero, then the model would simply be a moving average model, denoted as MA(q).

Figure 18.6 ARIMA Specification Window



***p*, Autoregressive Order** The order p of the polynomial $\varphi(\mathbf{B})$ operator.

***d*, Differencing Order** The order d of the differencing operator.

***q*, Moving Average Order** The order q of the differencing operator $\theta(\mathbf{B})$.

Prediction Interval Enables you to set the prediction level between 0 and 1 for the forecast prediction intervals.

Intercept Determines whether the intercept term μ is a part of the model.

Constrain fit If checked, the fitting procedure constrains the autoregressive parameters to always remain within the stable region and the moving average parameters within the invertible region.

Tip: Deselect the Constrain fit option if the fitter is having difficulty finding the true optimum or if you want to speed up the fit. You can use the Model Summary table to see whether the resulting fitted model is stable and invertible.

Once you specify the model and click **Estimate**, a Model Report is added to the report window. See "Reports".

Seasonal ARIMA Shows the Seasonal ARIMA Specification window, which enables you to specify the Seasonal ARIMA model that you want to fit. This window has the same elements as the ARIMA specification window, but it also contains the seasonal element specifications. The additional Observations per Period option enables you to specify the number of observations per period, denoted as s. For more information about the Seasonal ARIMA model, see "Seasonal ARIMA Model".

Note: Seasonal ARIMA models are denoted as Seasonal ARIMA$(p,d,q)(P,D,Q)s$.

Once you specify the model and click **Estimate**, a Model Report is added to the report window. See "Reports".

Smoothing Models

Shows a submenu of smoothing models. Once you select a smoothing model, a specification window appears. See "Smoothing Model Windows". For each model that is specified, a Smoothing Model Report appears in the report window. See "ARIMA and Seasonal ARIMA Model Report". Smoothing models represent the evolution of a time series by the model:

$$y_t = \mu_t + \beta_t t + s(t) + a_t$$

where

μ_t is the time-varying mean term

β_t is the time-varying slope term

$s(t)$ is one of the s time-varying seasonal terms

a_t are the random shocks

For more information about the general smoothing model equation, see "Statistical Details for Exponential Smoothing Models". The following smoothing models are available:

Simple Moving Average (Not available when either Forecast on Holdback or Use Box-Cox Transformation are selected in the launch window.) A model that estimates values by using an average of several adjacent points, defined by the smoothing window. The Simple Smoothing Average Specification window enables you to specify aspects of the smoothing window. Once the smoothing window options are specified, a Simple Moving

Average report is shown. By default, this report produces plotted values that are equal to the average of consecutive observations in a time window. Multiple Simple Moving Average models can be added and shown on the same plot. See "Simple Smoothing Average Specification Window".

Simple Exponential Smoothing A model with a level component. See "Simple Exponential Smoothing".

Double Exponential Smoothing A model with a level component and a trend component. This is a special case of Linear Exponential Smoothing. See "Double (Brown) Exponential Smoothing".

Linear Exponential Smoothing A model with a level component and a trend component. See "Linear (Holt) Exponential Smoothing".

Damped-Trend Linear Exponential Smoothing A model with a level component and a damped trend component. This model is appropriate for a series that exhibits a trend more complicated than a linear trend. See "Damped-Trend Linear Exponential Smoothing".

Seasonal Exponential Smoothing A model with a level component and a seasonal component. See "Seasonal Exponential Smoothing".

Winters Method A model with a level component, a trend component, and a seasonal component. See "Winters Method (Additive)".

Note: Each smoothing model, except for the Simple Moving Average model, is a traditional exponential smoothing model and has an ARIMA model equivalent. You might not be able to specify the equivalent ARIMA model using the ARIMA option because some smoothing models intrinsically constrain the ARIMA model parameters in ways that the ARIMA option does not allow.

State Space Smoothing Models

Shows the Specify State Space Smoothing Models window, which enables you to fit a variety of state space smoothing models as defined by Hyndman et al. (2008). A state space smoothing model is defined based on its error, trend component, and seasonal component.

- The errors can be additive (A) or multiplicative (M).
- The trend component can be none (N), additive (A), additive damped (A_d), multiplicative (M), or multiplicative damped (M_d).
- The seasonal component can be none (N), additive (A), or multiplicative (M).

A specific model can be represented by its ETS (Error, Trend, Seasonal). Use the check boxes in the Specify State Space Smoothing Models window to select the error, trend, and seasonality for the desired models. Click **Select Recommended** to select the check boxes that correspond to the models recommended by the platform. The window opens with the recommended models selected. Click **Select All** to select all check boxes or **Deselect All** to deselect all check boxes. The window also contains the following options:

Period Specifies seasonality values to be considered in the model fitting process.

Constrain Parameters Constrains the parameters in such a way that the further an observation is from the present, the less effect it has on the present state of the model. In State Space Smoothing models, a forecast at time *t*, given all previous observations, is the same as the weighted sum of all observations up to time *t*. The weights are a function of the parameters. Therefore, constraining the parameters ensures that the weights for past observations go to zero and that the further an observation is from the present, the faster the weight goes to zero.

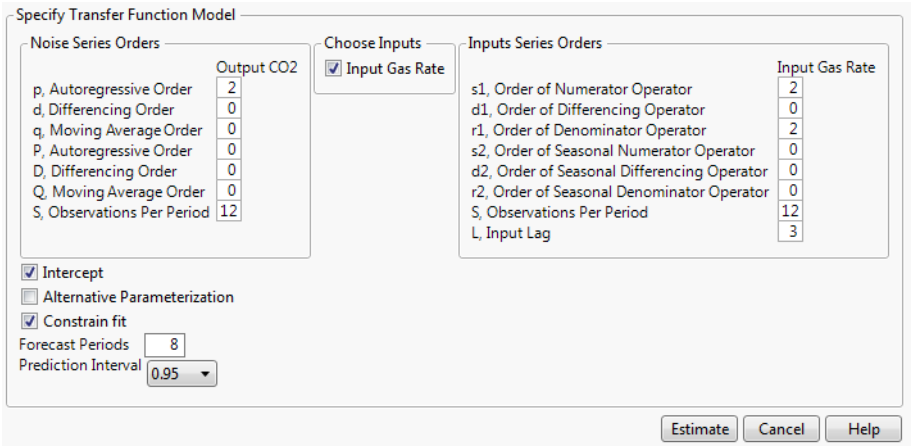
When you click **OK**, the specified set of models is fit. Summary values for each state space smoothing model are added to the Model Comparison table. Individual fit reports are added to the report window.

Transfer Function Models

Transfer Function (Available only in the Transfer Function Analysis red triangle menu.) Shows the Transfer Function Model Specification window. Building a transfer function model is similar to building an ARIMA model; it is an iterative process of exploring, fitting, and comparing models. Before building a model and during the data exploration process, it is sometimes useful to prewhiten the data. See "Prewhitening". For more information about transfer functions, see "Statistical Details for Transfer Functions".

Note: Currently, the Transfer Function option has limited support of missing values. Because the Time Series platform also treats excluded rows as missing values, transfer function models cannot be fit on a data table with excluded rows.

Figure 18.7 Transfer Function Model Specification Window



The Transfer Function Model Specification window contains the following sections:

Noise Series Orders Contains specifications for the noise series. Lowercase letters are coefficients for non-seasonal polynomials, and uppercase letters are coefficients for seasonal polynomials.

Choose Inputs Enables you select the input series for the model.

Input Series Orders Contains specifications for the input series. The first three orders relate to non-seasonal polynomials. The next four orders relate to seasonal polynomials. The final option is for an input lag.

There are three additional options that control model fitting:

Intercept Specifies whether the model has an intercept. If unchecked, it is assumed that μ is zero.

Alternative Parameterization Specifies whether the general regression coefficient is factored out of the numerator polynomials.

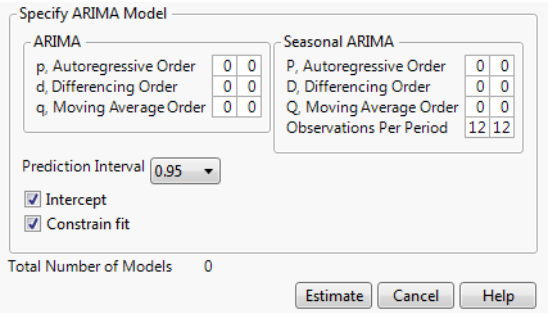
Constrain Fit Toggles the constraining of the AR and MA coefficients.

Forecast Periods Specifies the number of forecasting periods that are used for forecasting. If there are rows at the end of the data table that contain missing values for the Y variable and nonmissing values for the input variables, these rows are used in the initial forecasting settings. The values for the input variables are treated as future values of the input variables.

Prediction Interval Specifies the confidence level for the prediction interval.

ARIMA Model Group Shows the ARIMA Model Group window, which enables you to fit a range of ARIMA or Seasonal ARIMA models by specifying the range of orders. As you enter ranges into the window, the Total Number of Models updates accordingly.

Figure 18.8 ARIMA Model Group Specification Window



Once you specify the models and click **Estimate**, a Model Report for each specified model is added to the report window. See "Reports".

Combine and Save Forecasts from Models Creates a new data table with the combined results from all model fits in the report.

Save Spectral Density Creates a new data table containing the spectral density and periodogram where the *(i+1)*th row corresponds to the frequency $f_i = i / N$ (that is, the ith harmonic of $1 / N$). The new data table has the following columns:

Period The period of the ith harmonic, $1 / f_i$.

Frequency The frequency of the harmonic, f_i.

Angular Frequency The angular frequency of the harmonic, $2\pi f_i$.

Sine The Fourier sine coefficients, a_i.

Cosine The Fourier cosine coefficients, b_i.

Periodogram The periodogram, $I(f_i)$.

Spectral Density The spectral density, a smoothed version of the periodogram.

Number of Forecast Periods Shows a window that enables you to set the number of future periods that are forecast for the fitted models. The initial value is set in the Time Series launch window. All existing and future forecast results will use the new number of periods once it is changed.

Maximum Iterations Shows a window that enables you to reset the maximum number of iterations for future optimizations used in fitting ARIMA models.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

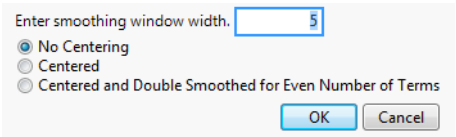
Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Smoothing Model Specification Windows

Simple Smoothing Average Specification Window

The Simple Smoothing Average Specification window appears when you select Simple Moving Average as the smoothing model. Let w be the smoothing window width in a simple moving average (SMA) model. Let $f_t=(y_t+y_{t-1}+y_{t-2}+...+y_{t-(w-2)}+y_{t-(w-1)})/w$ be the average of w consecutive observations for some time point t.

Figure 18.9 Simple Smoothing Average Specification Window



Enter smoothing window width The smoothing window width, w, that defines the number of consecutive points to average. The larger the window width, the more the series is smoothed.

No Centering The smoothing window is constructed from the points leading up to and including the time point, t, the point at which the series is being estimated. In other words, f_t is the plotted value for time t.

Centered The smoothing window is centered around the time point at which the series is being estimated.

- For odd w, f_t is the plotted value for time t-(w-1)/2.
- For even w, f_t is the plotted value for time t-(w-1)/2. When saved to a data table, f_t is at t-(w-2)/2.

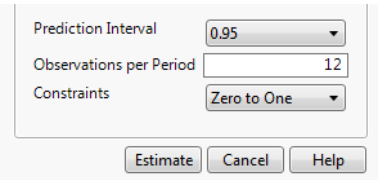
Centered and Double Smoothed for Even Number of Terms For even w, the smoothing window cannot be centered around the time point at which the series is being estimated. This option creates two smoothing windows that are almost centered, and averages them together. The smoothing estimates are calculated as follows:

$$f_{t-\frac{w}{2}} = \frac{y_t + 2\sum_{i=1}^{w-1} y_{t-i} + y_{t-w}}{2w}$$

Smoothing Model Windows

The Smoothing Model specification windows appear when you select one of the smoothing model options other than Simple Moving Average. The title of the window and the available options depend on the smoothing model option that you select.

Figure 18.10 Smoothing Model Specification Window



Prediction Interval Enables you to set the prediction level for the forecast prediction intervals.

Observations per Period (Available only for seasonal smoothing models.) Enables you to set the number of observations per period in a seasonal smoothing model.

Constraints Enables you to specify what type of constraint you want to enforce on the smoothing weights during the fit. The following constraint options are available:

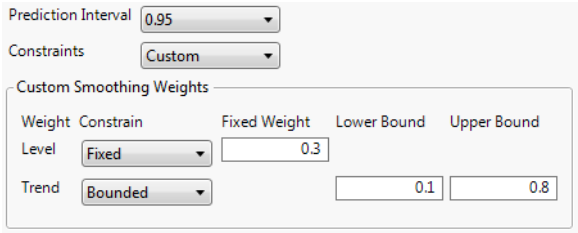
Zero To One Constrains the values of the smoothing weights to the range zero to one.

Unconstrained Allows the parameters to range freely.

Stable Invertible Constrains the parameters such that the equivalent ARIMA model is stable and invertible.

Custom Expands the dialog to enable you to set constraints on individual smoothing weights. Each smoothing weight can be **Bounded**, **Fixed**, or **Unconstrained** as determined by the setting of the popup menu next to the weight's name. When entering values for fixed or bounded weights, the values can be positive or negative real numbers.

Figure 18.11 Custom Smoothing Weights



The example shown in Figure 18.11 has the Level weight (α) fixed at a value of 0.3 and the Trend weight (γ) bounded by 0.1 and 0.8. In this case, the value of the Trend weight is allowed to move within the range 0.1 to 0.8 while the Level weight is held constant at 0.3. Note that you can specify all the smoothing weights in advance by using these custom constraints. In that case, none of the weights would be estimated from the data although forecasts and residuals would still be computed.

Reports

- "Difference Report"
- "Decomposition Reports"
- "Model Comparison Report"
- "ARIMA and Seasonal ARIMA Model Report"
- "State Space Smoothing Report"
- "Transfer Function Report"
- "Spectral Density Report"

Difference Report

The Difference Report contains graphs of the autocorrelations and partial autocorrelations of the differenced series. These graphs can be used to determine whether the differenced series is stationary.

The ARIMA and Seasonal ARIMA models that are available in the Time Series platform accommodate a differencing operation. In a two step process, these models first difference the time series according to the differencing operation, and then fit the differenced series. The Difference option is a useful preprocessing tool for determining the order of differencing to specify for the ARIMA model.

The Difference red triangle menu contains the following options:

Graph Shows a submenu of options to control the appearance of the differenced series plot. See "Time Series Platform Options".

Autocorrelation Shows or hides the autocorrelation of the differenced series.

Partial Autocorrelation Shows or hides the partial autocorrelations of the differenced series.

Variogram Shows or hides the variogram of the differenced series.

Save Saves a new column that contains the values in the differenced series to the original data table. Some of the leading elements are lost in the differencing process. They are represented as missing values in the saved Difference column.

Remove Fit Removes the Difference report from the report window.

Decomposition Reports

This section provides details about the reports obtained from the three decomposition options:

- "Linear Trend Report"
- "Cycle Report"
- "X11 Report"

Linear Trend Report

Contains the values of β_0 and β_1 from the linear regression model that is fit to the data:

$$\text{Trend}_t = \beta_0 + \beta_1 time$$

The detrended series is equal to $D_t = O_t$ - Trend$_t$, where O_t is the original time series.

Cycle Report

Contains the values of the cyclical component that is fit to the data:

$$\text{Cycle}_t = C + A \cdot \cos\left(2 \cdot \pi \cdot \left(\left(\frac{1}{U}\right) \cdot t + P\right)\right)$$

The parameter values are defined as follows:

- C is the (optional) Constant
- A is the Amplitude of the cosine wave

- U is the number of Units per Cycle
- P is the Phase of the cosine wave
- t is one less than the row number of a given observation

The decycled series is equal to $D_t = O_t$ - $Cycle_t$, where O_t is the original time series.

X11 Report

Depending on your selection of Decomposition Type, the X11 option adds an X11 - Multiplicative report or an X11 - Additive report. The reports contain the same four plots:

Original and Adjusted Overlays the X11-adjusted time series on the original time series, O_t. The X11-adjusted values are O_t/S_t for the multiplicative adjustment and O_t - S_t for the additive adjustment.

D10 - Final Seasonal Factors Plots the seasonal factor components, S_t, over time.

D12 - Final Trend Cycle Plots the trend cycle components, C_t, over time.

D13 - Final Irregular Series Plots the irregular components, I_t, over time.

X11 Report Options

The X11 reports have the following red triangle options:

Show Tables Shows or hides the X11 summary tables, as described in Shiskin et. al. (1967). The tables are grouped into five categories (labeled B through F), described in Table 18.1.

Save Columns Saves four columns to the data table: the seasonally adjusted time series, the seasonal components (S_t), the trend cycle components (C_t), and the irregular series components (I_t).

Save All Columns Saves columns to the data table for all of the tables produced in the report by the **Show Tables** option.

Remove Fit Removes the X11 report from the report window.

Table 18.1 Descriptions of the Categories of X11 Output Tables

Letter Prefix	Category Description
B	preliminary estimates of seasonal, trend cycle, and irregular components
C	intermediate estimates of seasonal, trend cycle, and irregular components

Table 18.1 Descriptions of the Categories of X11 Output Tables *(Continued)*

Letter Prefix	Category Description
D	final estimates of seasonal, trend cycle, and irregular components
E	analytical tables
F	summary measures

For more information about the contents of the X11 output tables, see Shisken et. al. (1967) or *SAS/ETS 15.2 User's Guide* (search for "The output from PROC X11").

Model Comparison Report

Once a model is fit, the Model Comparison Report is displayed in the report window. This report contains the Model Comparison table and plots for the models. Each time a new model is fit, a new row is added to the Model Comparison table, with a unique color-coding. The Model Comparison table summarizes the fit statistics for each model and is used to compare several models fitted to the same time series. By default, the models are sorted by the AIC statistic, in increasing order. To sort the model by a different statistic, right-click any column in the Model Comparison Table and select Sort by Column. Alternatively, you can also click on the column header to sort. For definitions of the fit statistics, see "Model Summary Table". The only fit statistic that is unique to the Model Comparison Table is Weights. This fit statistic is the normalized AIC Weight. The AIC Weight for a model is calculated as follows:

$$\text{AICWeight} = \exp[-0.5(\text{AIC} - \text{BestAIC})] / \sum_{k=1}^{K} (\exp[-0.5(\text{AIC}_k - \text{BestAIC})])$$

K is the total number of models, AIC_k is the AIC value for model k, and BestAIC is the AIC value for the model with the minimum AIC value.

If you specify a holdback set, the fit statistics that are computed on the training data are not shown in the Model Comparison table. Instead, the following metrics to determine the forecasting performance of the individual models on the holdback set are computed: Root Mean Square Error (RMSE), Mean Square Error (MSE), Mean Absolute Percentage Error (MAPE), and Mean Absolute Error (MAE). By default, the models are sorted by the RMSE statistic, in increasing order. To sort the model by a different statistic, right-click any column in the Model Comparison Table and select Sort by Column. Alternatively, you can also click on the column header to sort. For definitions of the fit statistics, see "Model Summary Table".

Caution: If you do not specify a holdback set and fit a combination of state space smoothing models and ARIMA models, a caution message is shown at the top of the Model Comparison table. In this case, the state space smoothing models are not comparable to the ARIMA models because likelihood values and likelihood-based information criteria, such as the AIC statistic, are not comparable between the two different model classes. However, if you specify a holdback set, the models can be compared because the RMSE statistic is used.

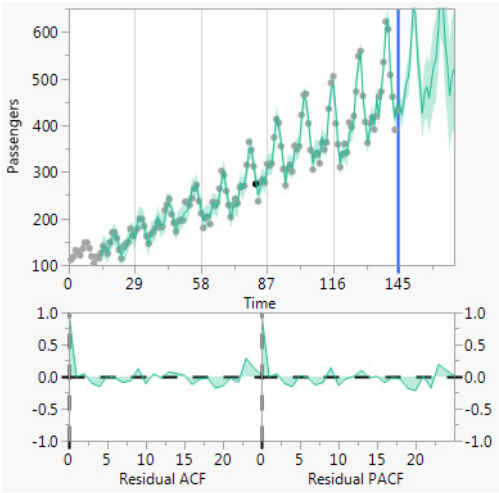
Figure 18.12 Model Comparison Table

Model Comparison

Report	Graph	Model	DF	Variance	AIC	SBC	RSquare	-2LogLH	Weights	.2 .4 .6 .8	MAPE	MAE
☑	☐	Seasonal ARIMA(0, 1, 1)(0, 1, 1)12	128	138.49122	1020.9047	1029.5302	0.990	1014.9047	1.000000		3.197010	8.990978
☑	☐	ARIMA(1, 1, 1)	140	978.9437	1394.1215	1403.0101	0.932	1388.1215	0.000000		8.687058	24.364166
☑	☐	ARMA(1, 1)	141	989.13835	1407.7483	1416.6577	0.919	1401.7483	0.000000		9.704486	25.348507
☑	☐	AR(1)	142	1134.3871	1426.1794	1432.1190	0.909	1422.1794	0.000000		9.873726	26.556219
☑	☐	MA(1)	142	4264.6121	1616.8626	1622.8022	0.693	1612.8626	0.000000		23.865796	53.716974

You can select which full model reports are shown in the report window using the **Report** check box. Two model plots appear to the right of the Model Comparison table. The top plot is a time series plot of the data, forecasts, and prediction limits. Below that are plots of the autocorrelation and partial autocorrelation functions. You can select which models are displayed on the model plots using the **Graph** check box.

Figure 18.13 Model Plots



Model Comparison Report Options

Each model in the Model Comparison report has the following red triangle menu options:

Fit New Opens a specification window for the model. You can change the settings to fit a different model.

Simulate Once (Not available if a holdback set is specified.) Provides one simulation of the model to the upper end of the time axis. The simulation is shown on the Model Comparison time series plot.

Simulate More (Not available if a holdback set is specified.) Provides the specified number of simulations of the model to the upper end of the time axis. The simulations are shown on the Model Comparison time series plot.

Remove Model Simulation (Not available if a holdback set is specified.) Removes the simulations for the model.

Remove All Simulation (Not available if a holdback set is specified.) Removes the simulations for all models.

Generate Simulation (Not available if a holdback set is specified.) Generates simulations for the model, and stores the results in a data table. You can specify the random seed, number of simulations, and the number of forecast periods.

Set Seed (Not available if a holdback set is specified.) Specifies the seed for generating the simulated trajectories.

Remove Fit Removes the corresponding model from the report window.

Remove All Models Removes all models that are listed in the Model Comparison table from the report window.

Remove Unselected Removes all models that do not have the Report check box selected from the report window.

Remove Selected Removes all models that have the Report check box selected from the report window.

Hide All Reports Hides all of the models that are listed in the Model Comparison table from the report window. This is equivalent to deselecting all of the Report check boxes.

Hide All Graphs Hides all of the models that are listed in the Model Comparison table from the Model Plots. This is equivalent to deselecting all of the Graph check boxes.

Note: The Time Series platform treats excluded rows as missing values. If the last several observations in the data table are excluded or missing, the simulation options in the Model Comparison Report will not work properly. The simulation procedure uses the last several observations of the time series. If the last several observations are missing, the simulation produces all missing values into the future.

ARIMA and Seasonal ARIMA Model Report

The time series modeling options are used to fit theoretical models to the series and use the fitted model to predict (forecast) future values of the series. These options also produce statistics and residuals that enable you to determine the adequacy of the model that you have chosen to use. You can select the modeling options multiple times. Each time you select a model, that model is added to the Model Comparison table. When the Report check box next to a model in the Model Comparison table is selected, a report is produced for that model. The report specifies the model in its title.

The following reports are shown by default:

- Model Summary Table
- Parameter Estimates Table
- Forecast Plot
- Residuals
- Iteration History

Model Summary Table Contains fit statistics for the model. In the formulas below, n is the length of the series and k is the number of fitted parameters in the model.

DF The number of degrees of freedom in the fit, $n - k$.

Sum of Squared Innovations The sum of the squared innovations.

Sum of Squared Residuals The sum of the squared residuals.

Variance Estimate The sum of squared innovations divided by the number of degrees of freedom ($n - k$). This is the sample estimate of the variance of the random shocks a_t, described in the section "ARIMA Model".

Standard Deviation The square root of the variance estimate. This is a sample estimate of the standard deviation of the random shocks a_t.

Akaike's 'A' Information Criterion The AIC value, computed as $-2\text{loglikelihood} + 2k$. Smaller AIC values indicate better fit.

Schwarz's Bayesian Criterion The SBC value, computed as $-2\text{loglikelihood} + k^*\ln(n)$. Smaller SBC values indicate better fit. Schwarz's Bayesian Criterion is equivalent to the Bayesian Information Criterion (BIC).

RSquare The R-Square value, computed as follows:

$$R^2 = 1 - \frac{\text{SSE}}{\text{SST}}$$

where

$$\text{SST} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$\text{SSE} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$\hat{y}_i$ are the one-step-ahead forecasts

$\bar{y}$ is the mean

If the model does not fit the series well, the model error sum of squares, SSE, might be larger than the total sum of squares, SST. As a result, R^2 can be negative.

RSquare Adj The adjusted R^2 value, computed as follows:

$$1 - \left[\frac{(n-1)}{(n-k)}(1 - R^2)\right]$$

MAPE The Mean Absolute Percentage Error value, computed as follows:

$$\frac{100}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

MAE The Mean Absolute Error value, computed as follows:

$$\frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

–2LogLikelihood Twice the negative log-likelihood function evaluated at the best-fit parameter estimates. Smaller values are better fits. See *Fitting Linear Models*.

Stable Indicates whether the autoregressive operator is stable. That is, whether all the roots of $\phi(z) = 0$ lie outside the unit circle.

Invertible Indicates whether the moving average operator is invertible. That is, whether all the roots of $\theta(z) = 0$ lie outside the unit circle.

Note: The ϕ and θ operators are defined in the section "ARIMA Model".

Parameter Estimates Table Shows the estimates for the time series model parameters. Each type of model has its own set of parameters, which are described in the sections on specific time series models. Each Parameter Estimates table contains the following columns:

Term The name of the parameter, which are described in the sections for each model type. Some models contain an *intercept* or mean term. In those models, the related *constant estimate* is also shown. The definition of the constant estimate is given under the description of ARIMA models.

Factor (Shown only for multiplicative Seasonal ARIMA models.) The factor of the model that contains the parameter. In the multiplicative seasonal models, Factor 1 is nonseasonal and Factor 2 is seasonal.

Lag (Shown only for ARIMA and Seasonal ARIMA models.) The degree of the lag or backshift operator that is applied to the term to which the parameter is multiplied.

Estimate The parameter estimates of the time series model.

Std Error The estimates of the standard errors of the parameter estimates. These estimates are used to calculate tests and prediction intervals.

t Ratio The test statistics for the hypotheses that each parameter is zero. The test statistic for a parameter is the ratio of the parameter estimate to its standard error. If the hypothesis is true, then this statistic has an approximate Student's *t* distribution. Looking for a *t*-ratio greater than 2 in absolute value is a common rule for judging significance because it approximates the 0.05 significance level.

Prob>|t| The observed *p*-value calculated for each parameter. The *p*-value is the probability of getting a *t*-ratio greater (in absolute value) than the computed value, given a true hypothesis.

Constant Estimate (Shown only for models that contain an intercept or mean term.) The definition of the constant estimate is given under ARIMA model.

Mu (Shown only for ARIMA and Seasonal ARIMA models.) The estimate for the intercept value of an ARIMA or seasonal ARIMA model.

Forecast A plot that shows both the observed and predicted values for the time series. The plot is divided by a vertical line into two regions.

If you specified a holdback set, a vertical line separates the training data from the holdback data. To the left of the vertical line, the one-step-ahead forecasts are overlaid with the

training data points. To the right of the vertical line, the values forecast by the model are overlaid with the holdback data points. The prediction intervals for the forecast are also shown. You can control the number of forecast values using the Forecast Periods option in the platform launch window.

If you did not specify a holdback set, the vertical line separates the observed data from the future forecasts. To the left of the vertical line, the one-step-ahead forecasts are overlaid with the observed data points. To the right of the vertical line, the future values forecast by the model and the prediction intervals for the forecast are shown. You can control the number of future forecast values using the Forecast Periods option in the platform launch window or by selecting Number of Forecast Periods from the Time Series red triangle menu.

Residuals A graph that shows the values of the residuals based on the fitted model. These values are the observed values of the time series minus the one-step-ahead predicted values. The autocorrelation and partial autocorrelation reports for these residuals are also shown. These reports can be used to determine whether the fitted model is adequate to describe the data. If the fitted model is adequate, the points in the residual plot should be normally distributed about zero and the autocorrelation and partial autocorrelation of the residuals should not have any significant components for lags greater than zero.

Forecasting Errors on Holdback Series (Available only if a holdback set is specified.) A plot of the forecasting errors on the observations in the holdback set. The autocorrelation and partial autocorrelation reports for these errors are also shown.

Iteration History Contains the value of the objective function at each iteration. This can be useful for diagnosing problems with the fitting procedure. Attempting to fit a model that is poorly suited to the data can result in a failure to converge on an optimum value for the likelihood. The Iteration History table contains the following quantities:

Iter The iteration number.

Iteration History The objective function value for each step.

Step The type of iteration step.

Obj-Criterion The norm of the gradient of the objective function.

ARIMA and Seasonal ARIMA Model Report Options

Each ARIMA and Seasonal ARIMA model report has a red triangle menu that contains the following options:

Show Points Shows or hides the data points in the forecast graph.

Show Prediction Interval Shows or hides the prediction intervals in the forecast graph.

Save Columns Creates a new data table that contains columns that represent the results of the model.

Save Prediction Formula (Not available if a holdback set is specified.) Saves the data and prediction formula to a new data table.

Create SAS Job Creates SAS code that duplicates the model analysis in SAS.

Submit to SAS Submits SAS code to SAS that duplicates the model analysis. If you are not connected to a SAS server, this option guides you through the connection process.

Residual Statistics Controls the displays of residual statistics are shown for the model. These displays are described in the section "Time Series Platform Options". However, they are applied to the series of residuals.

Remove Fit Removes the individual model fit from the report.

State Space Smoothing Report

Each State Space Smoothing model is added to the Model Comparison table. If the Report check box for a state space smoothing model in the Model Comparison table is selected, a State Space Smoothing report appears. Each State Space Smoothing report has a unique outline name that identifies the model. The following reports are included:

Model Type Describes the best fitting model for the series in terms of error, trend, and seasonality.

Model Summary Contains fit statistics for the model. In the formulas below, n is the length of the series and k is the number of fitted parameters in the model.

–2LogLikelihood Twice the negative log-likelihood function evaluated at the best-fit parameter estimates. Smaller values are better fits. See *Fitting Linear Models*.

AIC The Akaike's Information Criterion (AIC) value, computed as $-2\text{loglikelihood} + 2k$. Smaller AIC values indicate better fit.

BIC The Schwarz's Bayesian Criterion (SBC) value, computed as $-2\text{loglikelihood} + k*\ln(n)$. Smaller SBC values indicate better fit. Schwarz's Bayesian Criterion is equivalent to the Bayesian Information Criterion (BIC).

Nparm The number of parameters in the calculation of the information criteria. This number is one more than the number of free parameters in the likelihood function.

Note: Nparm is not the number of parameters in the parameter estimates table.

Sigma The estimate of the one-step-ahead forecast standard deviation.

RSquare The R-Square value, computed as follows:

$$R^2 = 1 - \frac{\text{SSE}}{\text{SST}}$$

where

$$\text{SST} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$\text{SSE} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$\hat{y}_i$ are the one-step-ahead forecasts

$\bar{y}$ is the mean

If the model does not fit the series well, the model error sum of squares, SSE, might be larger than the total sum of squares, SST. As a result, R^2 can be negative.

RSquare Adj The adjusted R^2 value, computed as follows:

$$1 - \left[\frac{(n-1)}{(n-k)}(1 - R^2)\right]$$

MAPE The Mean Absolute Percentage Error value, computed as follows:

$$\frac{100}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

where

$\hat{y}_i$ are the one-step-ahead forecasts

MAE The Mean Absolute Error value, computed as follows:

$$\frac{1}{n}\sum_{i=1}^{n}\left|y_i-\hat{y}_i\right|$$

where

$\hat{y}_i$ are the one-step-ahead forecasts

Parameter Estimates A table that shows the estimate, standard error, and 95% confidence interval for each parameter in the model.

Forecast A plot that shows both the observed and predicted values for the time series. See "Forecast".

One-Step-Ahead Forecasting Errors A plot of the one-step-ahead forecasting errors over time. For each time point *t*, the error is calculated as the observed value of the time series at *t* minus the one-step-ahead predicted value at *t*. If a holdback set is specified, the errors are computed only for the training data. The autocorrelation and partial autocorrelation reports for these errors are also shown.

Forecasting Errors on Holdback Series (Available only if a holdback set is specified.) A plot of the forecasting errors on the observations in the holdback set. The autocorrelation and partial autocorrelation reports for these errors are also shown.

Component States A report that contains a plot for each state of the model and the corresponding formula for the model. Each plot includes the state estimates of the series and the state forecasts. If a holdback set is specified, the state estimates are for the training portion and the state forecasts are for the holdback portion. If a holdback set is not specified, the state estimates are for the portion up to the last observation of the time series and the state forecasts are for after the last observation of the time series. A model can have up to three states: Level, Trend, and Seasonal.

The Component States red triangle menu has a Save States option. Select this option to save the component states at each time point to a separate data table.

One-Step-Ahead Relative Forecasting Errors for Multiplicative Error Model (Available only for multiplicative error models.) A plot of the one-step-ahead relative forecasting errors over time. For each time point *t*, the error is calculated as the one-step-ahead-forecasting error at *t* divided by the one-step-ahead predicted value at *t*.

State Space Smoothing Report Options

Each state space smoothing report has a red triangle menu that contains the following options:

Show Points Shows or hides the data points in the forecast graph.

Show Prediction Interval Shows or hides the prediction intervals in the forecast graph.

Save Columns Creates a new data table that contains columns that represent the results of the model.

Remove Fit Removes the individual model fit from the report.

Transfer Function Report

Each transfer function model is added to the Model Comparison table. If the Report check box for a transfer function model in the Model Comparison table is selected, a Transfer Function Model report is produced. If a holdback set is specified, a model plot is also available in the Model Comparison report.

Each Transfer Function Model report contains the following reports:

- Model Summary
- Parameter Estimates
- Residuals
- Forecasting Errors on Holdback Series
- Interactive Forecasting
- Iteration History

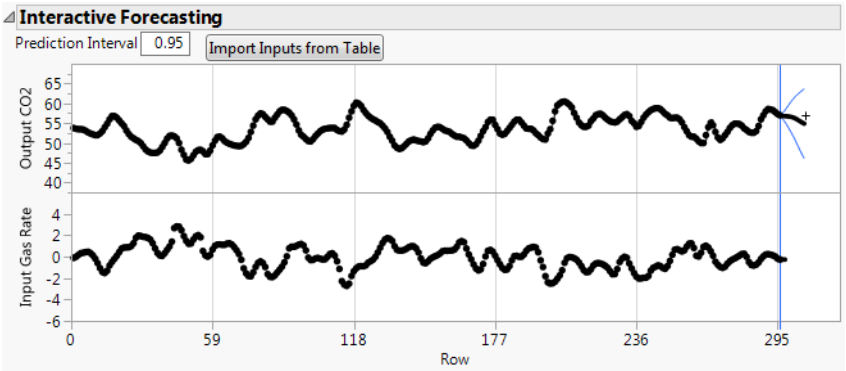
The information in the Model Summary, Parameter Estimates, Residuals, Forecasting Errors on Holdback Series, and Iteration History reports is the same as in the ARIMA and Seasonal ARIMA model report. For more information about these reports, see "ARIMA and Seasonal ARIMA Model Report". The Parameter Estimates table is followed by the formula of the model, where **B** is the backshift operator.

Interactive Forecasting

The Interactive Forecasting report provides a forecasting graph based on a prediction interval specified in the Transfer Function Specification window. The prediction interval around the prediction is shown in blue. Change the confidence level for this prediction interval by entering a number in the Prediction Interval box above the graph.

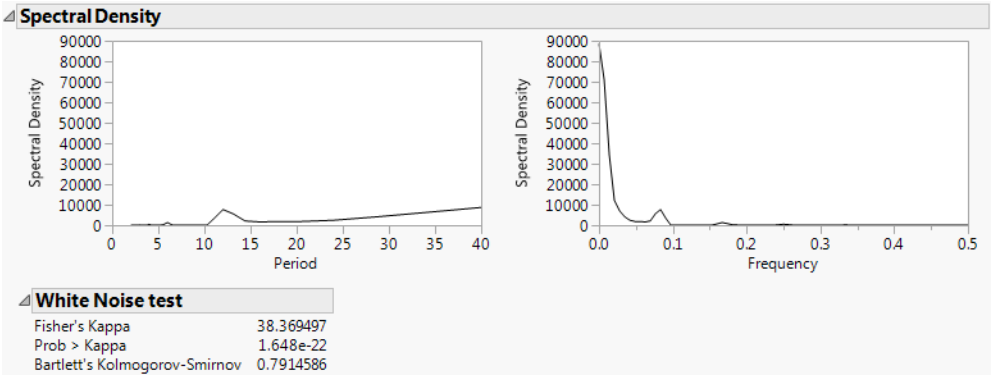
You can drag the plus sign in the graph to change the number of forecast periods in the graph. In the forecast periods, you can change the input values using the Import Inputs from Table button or by dragging points in the input graph to different values. The results of your changes are reflected in the forecast periods of the output graph.

Figure 18.14 Interactive Forecasting Graph



Spectral Density Report

Figure 18.15 Spectral Density Plots and White Noise Test Report



The White Noise Test report contains the following statistics:

Fisher's Kappa Tests the null hypothesis that the values in the series are drawn from a normal distribution with variance 1 against the alternative hypothesis that the series has a periodic component. Kappa is the ratio of the maximum value of the periodogram, $I(f_i)$, and its average value.

Prob > Kappa The probability of observing a value larger than Kappa if the null hypothesis is true, given by the following equation:

$$Pr(k > \kappa) = 1 - \sum_{j=0}^{q} (-1)^j \binom{q}{j} \left[\max\left(1 - \frac{jk}{q}, 0\right)\right]^{q-1}$$

where

$q = N / 2$ if N is even, $q = (N - 1) / 2$ if N is odd

κ is the observed value of Kappa

The null hypothesis is rejected if this probability is less than the significance level α.

Bartlett's Kolmogorov-Smirnov Compares the normalized cumulative periodogram to the cumulative distribution function of the uniform distribution on the interval (0, 1). The test statistic equals the maximum absolute difference of the cumulative periodogram and the uniform CDF. If it exceeds $a/(\sqrt{q})$, then one typically rejects the hypothesis that the series comes from a normal distribution. The values $a = 1.36$ and $a = 1.63$ correspond to significance levels 5% and 1% respectively.

Additional Examples of the Time Series Platform

- "Example of Creating Time ID Column"
- "Example Using Box-Cox Transformation"
- "Example Using a Holdback Set"

Example of Creating Time ID Column

This example uses the SeriesP.jmp sample data table to show how to perform a time series analysis. You first create a new column that is appropriate for the Time ID.

Create Appropriate Time ID Column

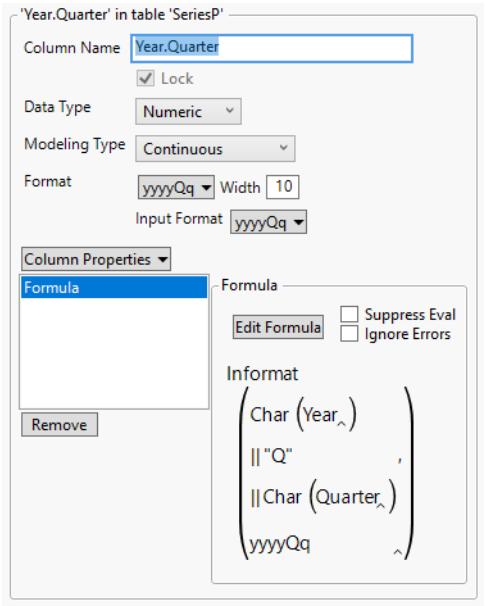
1. Select **Help > Sample Data Library** and open Time Series/SeriesP.jmp.

   The SeriesP.jmp data table contains a Year column and a Quarter column to identify the time period during which the responses were observed. However, the Time Series platform requires one column with unique, equally spaced time points to label the X axis. If no Time ID is specified, then the row number is used to identify the time periods. To avoid this and make the report easier to interpret, you construct a Time ID column from Year and Quarter.
2. Select **Cols > New Columns**. In the Column Name box, type Year.Quarter.
3. Select **Best > Date > yyyyQq** next to Format.

4. Select **Column Properties > Formula**.
5. Click the gray triangle next to Date Time and click **Informat**.
6. Click the gray triangle next to Character.
7. Select Year, click **Char,** and then click **Concat**.
8. Type "Q" in the box (including quotation marks) and press Enter.
9. Click **Concat**.
10. Select Quarter and click **Char**.
11. Click the <formatName> box, type yyyyQq, and press Enter.
12. Click **OK**.

    The completed New Column window should appear as in Figure 18.16.

Figure 18.16 New Column



13. Click **OK**.

Note: This time column also works for X11 analyses.

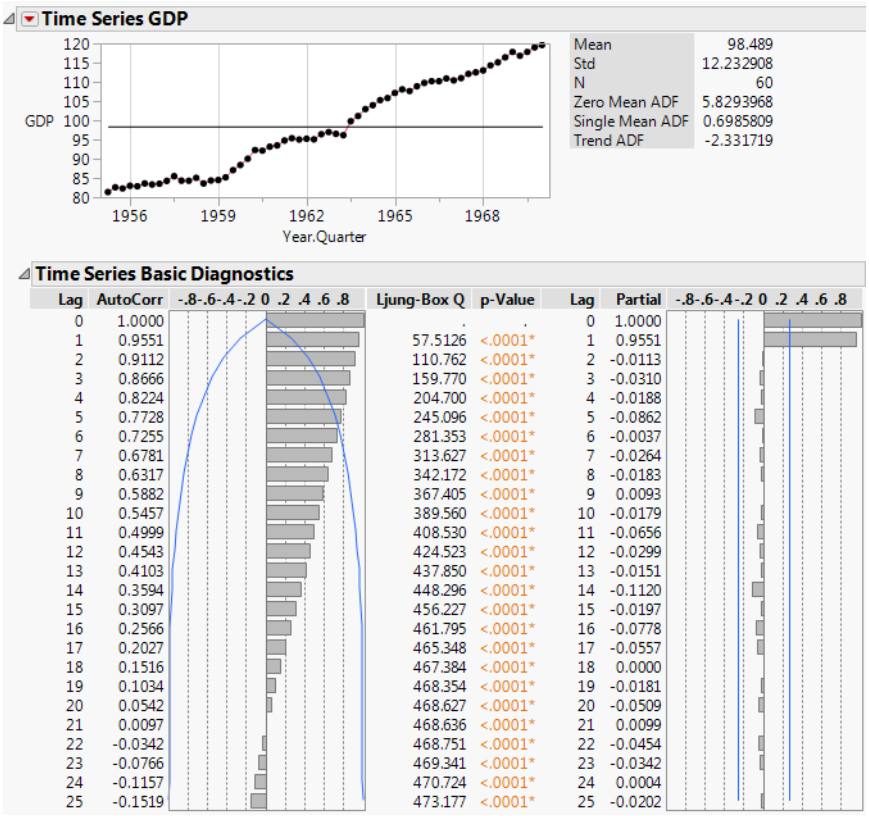
Time Series Analysis

Now that the data table contains an appropriate Time ID column, proceed with the analysis.

1. Select **Analyze > Specialized Modeling > Time Series**.
2. Select GDP and click **Y, Time Series**.

3. Select Year.Quarter and click **X, Time ID**.
4. Click **OK**.

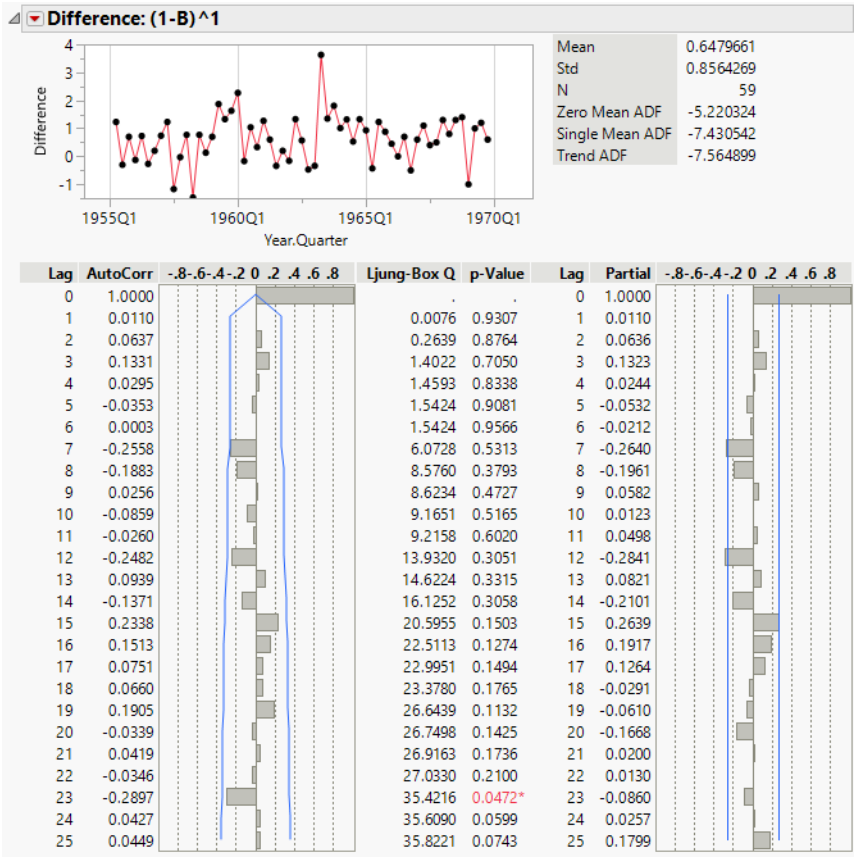
Figure 18.17 Time Series Report for SeriesP.jmp



The series shows an increasing trend over time that is fairly linear. In addition, the autocorrelation chart shows that there is strong correlation between points that are close together. The AutoCorr values for points with lags of 1, 2, and 3 are 0.9551, 0.9112, 0.8666, respectively.

5. Click the Time Series GDP red triangle and select **Difference**.
6. Select 1 for the Nonseasonal Differencing Order and click **Estimate**.

Figure 18.18 Difference Report for SeriesP.jmp



Lag	AutoCorr	Ljung-Box Q	p-Value	Lag	Partial
0	1.0000	.	.	0	1.0000
1	0.0110	0.0076	0.9307	1	0.0110
2	0.0637	0.2639	0.8764	2	0.0636
3	0.1331	1.4022	0.7050	3	0.1323
4	0.0295	1.4593	0.8338	4	0.0244
5	-0.0353	1.5424	0.9081	5	-0.0532
6	0.0003	1.5424	0.9566	6	-0.0212
7	-0.2558	6.0728	0.5313	7	-0.2640
8	-0.1883	8.5760	0.3793	8	-0.1961
9	0.0256	8.6234	0.4727	9	0.0582
10	-0.0859	9.1651	0.5165	10	0.0123
11	-0.0260	9.2158	0.6020	11	0.0498
12	-0.2482	13.9320	0.3051	12	-0.2841
13	0.0939	14.6224	0.3315	13	0.0821
14	-0.1371	16.1252	0.3058	14	-0.2101
15	0.2338	20.5955	0.1503	15	0.2639
16	0.1513	22.5113	0.1274	16	0.1917
17	0.0751	22.9951	0.1494	17	0.1264
18	0.0660	23.3780	0.1765	18	-0.0291
19	0.1905	26.6439	0.1132	19	-0.0610
20	-0.0339	26.7498	0.1425	20	-0.1668
21	0.0419	26.9163	0.1736	21	0.0200
22	-0.0346	27.0330	0.2100	22	0.0130
23	-0.2897	35.4216	0.0472*	23	-0.0860
24	0.0427	35.6090	0.0599	24	0.0257
25	0.0449	35.8221	0.0743	25	0.1799

The Difference report helps determine an appropriate model to be fit to the original time series. The plot of differences shows that the differenced series no longer has the trend that was observed in the original data. This indicates that lag-1 differencing is an appropriate choice. Also, even after removing the trend, the series shows no sign of seasonality. For these reasons, models to fit the original series should be able to handle linear trends, but do not necessarily need to handle seasonality. Linear exponential smoothing and ARIMA models would be appropriate.

7. Click the Time Series GDP red triangle and select **Smoothing Model > Linear Exponential Smoothing**.
8. Click **Estimate**.
9. Click the Time Series GDP red triangle and select **ARIMA Model Group**. This enables you to fit multiple ARIMA models for a range of values of $(p,d,q)(P,D,Q)$.
10. In the ARIMA box, set the following ranges:

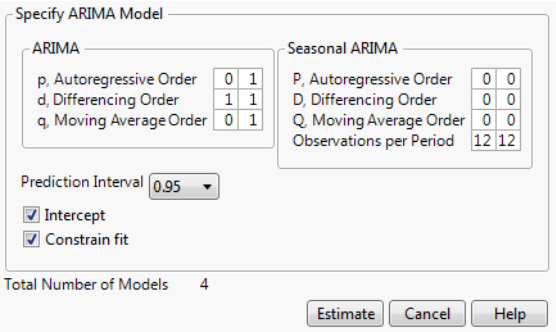
- Fix d, the differencing order, at 1 by setting the range from 1 to 1 because the differencing report showed lag-1 differencing was appropriate.
- Set p, the autoregressive order, to range from 0 to 1 because the original series showed evidence of autocorrelation.
- Set q, the moving average order, to range from 0 to 1.

Note: In most cases, it is sufficient to keep p and q small.

- Leave P, D, and Q set at 0, since the series showed no evidence of seasonality.

These settings lead to the fitting of four total models.

Figure 18.19 ARIMA Model Group Specification



11. Click **Estimate**.

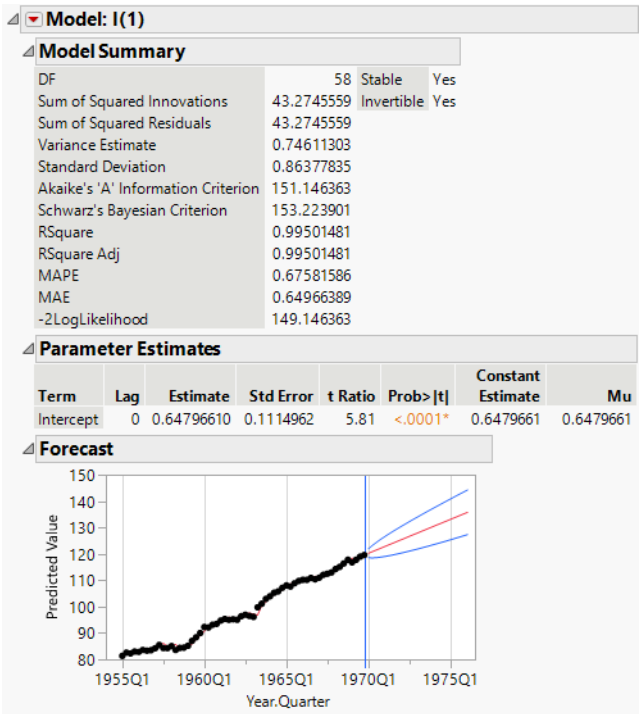
Figure 18.20 Model Comparison Table

Model Comparison

Report	Graph	Model	DF	Variance	AIC	SBC	RSquare	-2LogLH	Weights	.2 .4 .6 .8	MAPE	MAE
✓		I(1)	58	0.746113	151.14636	153.22390	0.995	149.14636	0.477865		0.675816	0.649664
✓		ARI(1, 1)	57	0.7591096	153.13924	157.29432	0.995	149.13924	0.176424		0.676216	0.649960
✓		IMA(1, 1)	57	0.7591199	153.14002	157.29510	0.995	149.14002	0.176355		0.676179	0.649933
✓		ARIMA(1, 1, 1)	56	0.748633	153.91882	160.15143	0.995	147.91882	0.119474		0.668189	0.641981
✓		Linear (Holt) Exponential Smoothing	56	0.7878954	155.66571	159.78660	0.994	151.66571	0.049882		0.716452	0.686737

The Model Comparison Table is sorted such that the best fitting model, according to the AIC criterion, is at the top of the list. In this case, the ARIMA(0,1,0) model (denoted I(1) in the report) best fits the original time series. It should also be noted that although the I(1) model is "best," all of the models have extremely similar values for the fit statistics. They could all be considered appropriate.

Figure 18.21 Model Report for ARIMA(0,1,0)



The model report for I(1) shows the forecast graph. The blue lines indicate the prediction intervals. GDP is predicted to continue increasing at a linear rate.

Example Using Box-Cox Transformation

In this example, you use a Box-Cox transformation to analyze the Passengers column in the Seriesg.jmp data table. The Lambda value is set to zero, which makes the Box-Cox transformation an approximate log transformation. The script in the Seriesg.jmp data table runs a time series analysis on the Log Passengers column. By using the Box-Cox transformation for the analysis, the results are easier to interpret since all forecasts are transformed back and reported on the original scale.

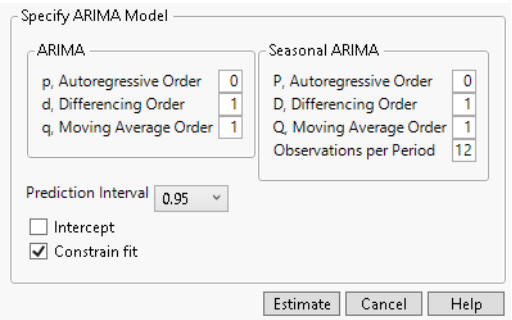
1. Select **Help > Sample Data Library** and open Time Series/Seriesg.jmp.
2. Select **Analyze > Specialized Modeling > Time Series**.
3. Select Passengers and click **Y, Time Series**.
4. Select Time and click **X, Time ID**.
5. Select **Use Box-Cox Transformation**.

   Use the default value of 0 for Lambda.

6. Click **OK**.
7. Click the red triangle next to Time Series Box-Cox Transformed Passengers and select **Seasonal ARIMA**.
8. In the ARIMA section of the Seasonal ARIMA Specification window, set the following parameters.
   - Set *p*, Autoregressive Order to 0.
   - Set *d*, Differencing Order to 1
   - Set *q*, Moving Average Order to 1
9. In the Seasonal ARIMA section of the Seasonal ARIMA Specification window, set the following parameters.
   - Set *P*, Autoregressive Order to 0
   - Set *D*, Differencing Order to 1
   - Set *Q*, Moving Average Order to 1
10. Deselect Intercept to run a model without an intercept.

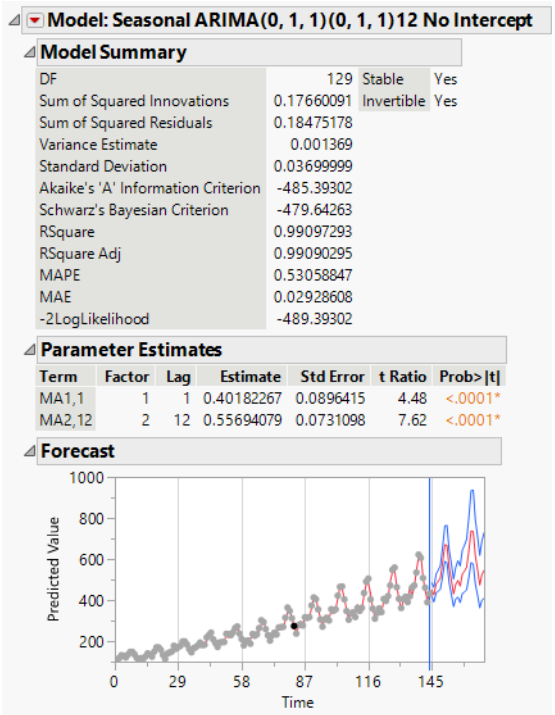
    Your window should now match the window shown in Figure 18.22.

Figure 18.22 Seasonal ARIMA Specifications



11. Click **Estimate**.

Figure 18.23 Seasonal ARIMA Model for Box-Cox Transformed Passengers Data



The parameter estimates are equivalent to those obtained from running the same model on the Log Passengers column. However, the forecasts and prediction intervals in Figure 18.23 are on the original scale.

Example Using a Holdback Set

This example uses a holdback set to evaluate predictions made on monthly data on lead production. Since a holdback set is used, both State Space Smoothing and ARIMA models can be fit and compared to find the best fitting model for the data.

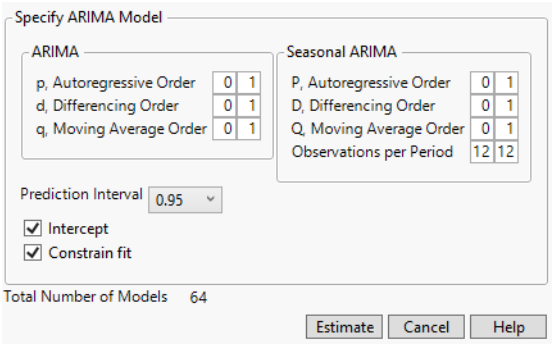
1. Select **Help > Sample Data Library** and open Time Series/Lead Production.jmp.
2. Select **Analyze > Specialized Modeling > Time Series**.
3. Select LEADPROD and click **Y, Time Series**.
4. Select DATE and click **X, Time ID**.
5. Select **Forecast on Holdback**.

   Selecting this options means that forecasts are made on the holdback set instead of on future observations.
6. Enter 12 next to Forecast Periods.

This assigns the last 12 observations in the data to the holdback set. Since this is monthly data, this is equivalent to the last year.

7. Click **OK**.
8. Click the Time Series LEADPROD red triangle and select **State Space Smoothing**.
9. Type 12 next to Period.
10. Select Constrain Parameters.
11. Click **OK**.
12. Click the Time Series LEADPROD red triangle and select **ARIMA Model Group**.
13. Set the range for all parameters from 0 to 1. The Specify ARIMA Model window looks like the one in Figure 18.24.

Figure 18.24 ARIMA Model Group Specifications



14. Click **Estimate**.

Figure 18.25 Model Comparison

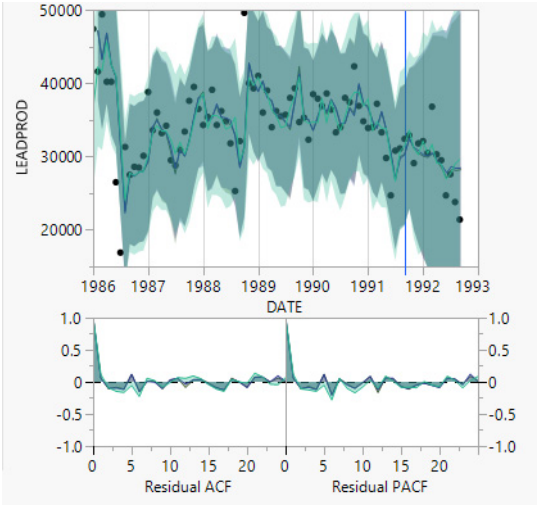
Model Comparison

Report	Graph	Model Class	Model	Holdback Evaluation RMSE	MSE	MAPE	MAE
		ARIMA	Seasonal ARIMA(0, 1, 1)(1, 0, 0)12	3219.4131	10364621	8.900164	2416.3319
		ARIMA	Seasonal ARIMA(0, 1, 1)(0, 0, 1)12	3316.8746	11001657	9.148390	2456.0873
		State Space Smoothing	AAA12 Constrained	3350.9303	11228734	8.623519	2245.5657
		ARIMA	Seasonal ARIMA(1, 1, 0)(1, 0, 0)12	3397.0855	11540190	9.082427	2392.5823
		ARIMA	Seasonal ARIMA(0, 1, 1)(1, 1, 0)12	3448.6372	11893099	9.829459	2649.8142
		ARIMA	Seasonal ARIMA(1, 1, 0)(1, 1, 0)12	3484.3764	12140879	9.708056	2604.6290
		ARIMA	Seasonal ARIMA(1, 1, 0)(0, 0, 1)12	3491.2648	12188930	9.302681	2435.7441
		State Space Smoothing	MAM12 Constrained	3512.1513	12335206	10.552196	2813.4655
		ARIMA	Seasonal ARIMA(0, 1, 0)(1, 0, 0)12	3525.0858	12426230	9.374006	2440.6892
		ARIMA	Seasonal ARIMA(0, 1, 0)(1, 1, 0)12	3551.3973	12612423	9.367947	2480.3210
		ARIMA	Seasonal ARIMA(1, 1, 1)(1, 0, 1)12	3579.0242	12809414	10.520274	2739.9652
		ARIMA	Seasonal ARIMA(0, 1, 0)(0, 1, 0)12	3597.3098	12940638	11.622868	3219.0476
		ARIMA	Seasonal ARIMA(0, 1, 0)(0, 0, 1)12	3631.8176	13190099	9.792052	2544.4269
		ARIMA	Seasonal ARIMA(1, 1, 0)(0, 1, 0)12	3661.8395	13409068	12.030672	3366.0935
		ARIMA	Seasonal ARIMA(0, 1, 1)(0, 1, 0)12	3673.6369	13495608	12.063523	3379.4644
		ARIMA	Seasonal ARIMA(1, 1, 0)(1, 0, 1)12	3700.6621	13694900	11.077791	2890.6426
		ARIMA	Seasonal ARIMA(0, 1, 0)(1, 0, 1)12	3723.7664	13866437	11.222827	2935.8075

The fit statistics are comparable between the State Space Smoothing models and the ARIMA models because they are evaluated on the holdback set.

15. Select the Graph check box next to the first three models in the Model Comparison report.

Figure 18.26 Model Comparison Graph



The evaluation statistics and the forecasting plots show that there is no substantial differences in forecasting performance among the top models. Based on the report, either an ARIMA model or a State Space Smoothing model is appropriate for this data.

Statistical Details for the Time Series Platform

- "Statistical Details for Spectral Density"
- "Statistical Details for X-11 Decomposition"
- "Statistical Details for Exponential Smoothing Models"
- "Statistical Details for ARIMA Models"
- "Statistical Details for Transfer Functions"

Statistical Details for Spectral Density

The least squares estimates of the coefficients of the Fourier series are computed as follows:

$$a_t = \frac{2}{N} \sum_{i=1}^{N} y_t \cos(2\pi f_i t)$$

$$b_t = \frac{2}{N} \sum_{i=1}^{N} y_t \sin(2\pi f_i t)$$

Then the $f_i = i/N$ are combined to form the periodogram $I(f_i) = (N/2)(a_i^2 + b_i^2)$, which represents the intensity at frequency f_i.

The periodogram is then smoothed and scaled by $1/(4\pi)$ to form the spectral density.

Statistical Details for X-11 Decomposition

This method adjusts the original time series using either a multiplicative or an additive decomposition. The model is fit using an iterative process to estimate the three X-11 components: trend cycle, seasonal, and irregular. The trend cycle component contains both the long-term trend and the long-term cyclical effects. The irregular component contains the effects of variation unexplained by the trend and seasonal components. For a historical overview of the development of the X-11 method, see *SAS/ETS 15.2 User's Guide* (search for "Historical Development of X-11").

The multiplicative adjustment fits the following model:

$$O_t = C_t \cdot S_t \cdot I_t$$

where

O_t is the original time series

C_t is the trend cycle component

S_t is the seasonal component

I_t is the irregular component

The adjusted multiplicative trend is O_t/S_t.

The additive adjustment fits the following model:

$$O_t = C_t + S_t + I_t$$

The adjusted additive trend is O_t - S_t.

Statistical Details for Exponential Smoothing Models

Smoothing models are defined as follows:

$$y_t = \mu_t + \beta_t t + s(t) + a_t$$

where

μ_t is the time-varying mean term

β_t is the time-varying slope term

$s(t)$ is one of the s time-varying seasonal terms

a_t are the random shocks

Models without a trend have $\beta_t = 0$ and nonseasonal models have $s(t) = 0$. The estimators for these time-varying terms are defined as follows:

L_t is a smoothed level that estimates μ_t

T_t is a smoothed trend that estimates β_t

S_{t-j} for $j = 0, 1, \ldots, s - 1$ are the estimates of the $s(t)$

Each smoothing model defines a set of recursive smoothing equations that describe the evolution of these estimators. The smoothing equations are written in terms of model parameters called *smoothing weights*:

α is the level smoothing weight

γ is the trend smoothing weight

φ is the trend damping weight

δ is the seasonal smoothing weight

While these parameters enter each model in a different way (or not at all), they have the common property that larger weights give more influence to recent data while smaller weights give less influence to recent data. For more information on smoothing weights, see *SAS/ETS 15.2 User's Guide* (the Forecasting Process Details chapter).

Simple Exponential Smoothing

The model for simple exponential smoothing is $y_t = \mu_t + \alpha_t$.

The smoothing equation, $L_t = \alpha y_t + (1 - \alpha)L_{t-1}$, is defined in terms of a single smoothing weight α. This model is equivalent to an ARIMA(0, 1, 1) model where the following is true:

$$(1 - B)y_t = (1 - \theta B)\alpha_t \quad \text{where} \quad \theta = 1 - \alpha$$

The moving average form of the model is defined as follows:

$$y_t = a_t + \sum_{j-1}^{\infty} \alpha a_{t-j}$$

Double (Brown) Exponential Smoothing

The model for double exponential smoothing is $y_t = \mu_t + \beta_1 t + a_t$.

The smoothing equations, defined in terms of a single smoothing weight α, are defined as follows:

$$L_t = \alpha y_t + (1 - \alpha)L_{t-1}$$

$$T_t = \alpha(L_t - L_{t-1}) + (1 - \alpha)T_{t-1}$$

This model is equivalent to an ARIMA(0, 1, 1)(0, 1, 1)$_1$ model where the following is true:

$$(1 - B)^2 y_t = (1 - \theta B)^2 a_t \quad \text{where} \quad \theta_{1,1} = \theta_{2,1} \quad \text{with} \quad \theta = 1 - \alpha$$

The moving average form of the model is defined as follows:

$$y_t = a_t + \sum_{j=1}^{\infty} (2\alpha + (j-1)\alpha^2)a_{t-j}$$

Linear (Holt) Exponential Smoothing

The model for linear exponential smoothing is $y_t = \mu_t + \beta_t t + a_t$.

The smoothing equations, in terms of smoothing weights α and γ, are defined as follows:

$$L_t = \alpha y_t + (1 - \alpha)(L_{t-1} + T_{t-1})$$

$$T_t = \gamma(L_t - L_{t-1}) + (1 - \gamma)T_{t-1}$$

This model is equivalent to an ARIMA(0, 2, 2) model where the following is true:

$$(1-B)^2 y_t = (1-\theta B-\theta_2 B^2)a_t \quad \text{with} \quad \theta = 2-\alpha-\alpha\gamma \quad \text{and} \quad \theta_2 = \alpha-1$$

The moving average form of the model is defined as follows:

$$y_t = a_t + \sum_{j=1}^{\infty} (\alpha + j\alpha\gamma)a_{t-j}$$

Damped-Trend Linear Exponential Smoothing

The model for damped-trend linear exponential smoothing is $y_t = \mu_t + \beta_t t + a_t$.

The smoothing equations, in terms of smoothing weights α, γ, and φ, are defined as follows:

$$L_t = \alpha y_t + (1-\alpha)(L_{t-1} + \varphi T_{t-1})$$

$$T_t = \gamma(L_t - L_{t-1}) + (1-\gamma)\varphi T_{t-1}$$

This model is equivalent to an ARIMA(1, 1, 2) model where the following is true:

$$(1-\varphi B)(1-B)y_t = (1-\theta_1 B-\theta_2 B^2)a_t$$

where

$$\theta_1 = 1+\varphi-\alpha-\alpha\gamma\varphi$$

$$\theta_2 = (\alpha-1)\varphi$$

The moving average form of the model is defined as follows:

$$y_t = \alpha_t + \sum_{j=1}^{\infty} \left(\frac{\alpha + \alpha\gamma\varphi(\varphi^j - 1)}{\varphi - 1} \right) \alpha_{t-j}$$

Seasonal Exponential Smoothing

The model for seasonal exponential smoothing is $y_t = \mu_t + s(t) + a_t$.

The smoothing equations in terms of smoothing weights α and δ are defined as follows:

$$L_t = \alpha(y_t - S_{t-s}) + (1-\alpha)L_{t-1}$$

$$S_t = \delta(y_t - L_{t-s}) + (1-\delta)S_{t-s}$$

This model is equivalent to a seasonal ARIMA(0, 1, s+1)(0, 1, 0)$_s$ model:

$$(1-B)(1-B^s)y_t = (1-\theta_1 B - \theta_2 B^2 - \theta_3 B^{s+1})a_t$$

where

$$\theta_1 = 1-\alpha$$

$$\theta_2 = (1-\delta)(1-\alpha)$$

$$\theta_3 = (1-\alpha)(\delta-1)$$

The moving average form of the model is defined as follows:

$$y_t = a_t + \sum_{j=1}^{\infty} \psi_j a_{t-j} \quad \text{where} \quad \psi = \begin{cases} \alpha \text{ for } j \bmod s \neq 0 \\ \alpha + \delta(1-\alpha) \text{ for } j \bmod s = 0 \end{cases}$$

Winters Method (Additive)

The model for the additive version of the Winters method is $y_t = \mu_t + \beta_t t + s(t) + a_t$.

The smoothing equations in terms of weights α, γ, and δ are defined as follows:

$$L_t = \alpha(y_t - S_{t-s}) + (1-\alpha)(L_{t-1} + T_{t-1})$$

$$T_t = \gamma(L_t - L_{t-1}) + (1-\gamma)T_{t-1}$$

$$S_t = \delta(y_t - L_t) + (1-\delta)S_{t-s}$$

This model is equivalent to a seasonal ARIMA(0, 1, s+1)(0, 1, 0)s model defined as follows:

$$(1-\mathrm{B})(1-\mathrm{B}^2)y_t = \left(1 - \sum_{i=1}^{s+1} \theta_i \mathrm{B}^i\right) a_t$$

The moving average form of the model is defined as follows:

$$y_t = a_t + \sum_{j=1}^{\infty} \Psi_j a_{t-j}$$

where

$$\psi = \begin{cases} \alpha + j\alpha\gamma\,, & j \bmod s \neq 0 \\ \alpha + j\alpha\gamma + \delta(1-\alpha)\,, & j \bmod s = 0 \end{cases}$$

Statistical Details for ARIMA Models

ARIMA Model

For a response series $\{y_i\}$, the general form for the ARIMA model is defined as follows:

$$\phi(\mathrm{B})(w_t - \mu) = \theta(\mathrm{B})a_t$$

where

t is the time index

B is the backshift operator defined as $\mathrm{B}y_t = y_{t-1}$

$w_t = (1 - \mathrm{B})^d y_t$ is the response series after differencing

μ is the intercept or mean term

$\phi(\mathrm{B})$ and $\theta(\mathrm{B})$ are the autoregressive operator and the moving average operator, respectively, and are defined as follows:

$$\phi(\mathrm{B}) = 1 - \phi_1\mathrm{B} - \phi_2\mathrm{B}^2 - \ldots - \phi_p\mathrm{B}^p$$

$$\theta(\mathrm{B}) = 1 - \theta_1\mathrm{B} - \theta_2\mathrm{B}^2 - \ldots - \theta_q\mathrm{B}^q$$

where

a_t are the sequence of random shocks

The a_t are assumed to be independent and normally distributed with mean zero and constant variance.

The model can be rewritten as follows:

$$\phi(B)w_t = \delta + \theta(B)a_t$$

The constant estimate δ is given by the relation:

$$\delta = \phi(B)\mu = \mu - \phi_1\mu - \phi_2\mu - \ldots - \phi_p\mu$$

Seasonal ARIMA Model

In the case of **Seasonal ARIMA** modeling, the differencing, autoregressive, and moving average operators are the product of seasonal and nonseasonal polynomials:

$$w_t = (1-B)^d(1-B^s)^D y_t$$

$$\varphi(B) = (1-\varphi_{1,1}B - \varphi_{1,2}B^2 - \ldots - \varphi_{1,p}B^p)(1-\varphi_{2,s}B^s - \varphi_{2,2s}B^{2s} - \ldots - \varphi_{2,Ps}B^{Ps})$$

$$\theta(B) = (1-\theta_{1,1}B - \theta_{1,2}B^2 - \ldots - \theta_{1,q}B^q)(1-\theta_{2,s}B^s - \theta_{2,2s}B^{2s} - \ldots - \theta_{2,Qs}B^{Qs})$$

where s is the number of observations per period. The first index on the coefficients is the factor number (1 indicates nonseasonal, 2 indicates seasonal) and the second is the lag of the term.

Statistical Details for Transfer Functions

A typical transfer function model with m inputs can be represented as follows:

$$Y_t - \mu = \frac{\omega_1(B)}{\delta_1(B)}X_{1,t-d1} + \ldots + \frac{\omega_m(B)}{\delta_m(B)}X_{m,m-dm} + \frac{\theta(B)}{\varphi(B)}e_t$$

where

Y_t denotes the output series

X_1 to X_m denote m input series

e_t represents the noise series

$X_{1,\, t-d1}$ indicates the series X_1 is indexed by t with a $d1$-step lag

μ represents the mean level of the model

$\varphi(B)$ and $\theta(B)$ represent autoregressive and moving average polynomials from an ARIMA model

$\omega_k(B)$ and $\delta_k(B)$ represent numerator and denominator factors (or polynomials) for individual transfer functions, with k representing an index for the 1 to m individual inputs.

Each polynomial in the above model can contain two parts, either nonseasonal, seasonal, or a product of the two as in seasonal ARIMA. When specifying a model, leave the default 0 for any part that you do not want to include in the model.

Chapter 19

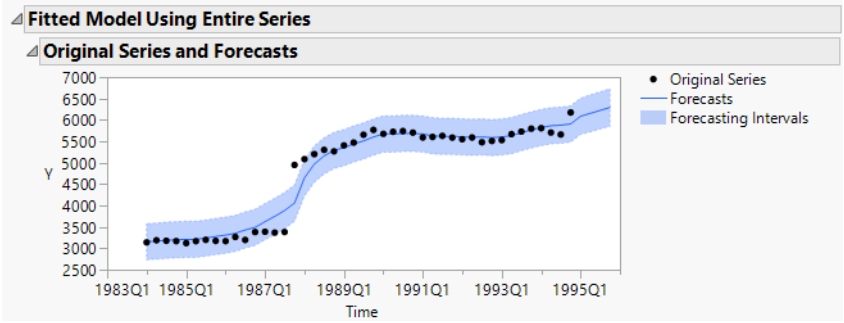
Time Series Forecast

Forecast Multiple Time Series

The Time Series Forecast platform enables you to forecast multiple time series, where the number of series could be very large. The platform provides compact summaries of the multiple time series and provides options for efficiently fitting up to 30 different forecasting models in the exponential smoothing family. The best fitting model is automatically selected.

The Time Series Forecast platform is designed to select a best fitting model automatically from a set of exponential smoothing models. This is in contrast to the Time Series platform which requires manual selection of a best fitting model and uses ARIMA models.

Figure 19.1 Original Series and Forecasts



Contents

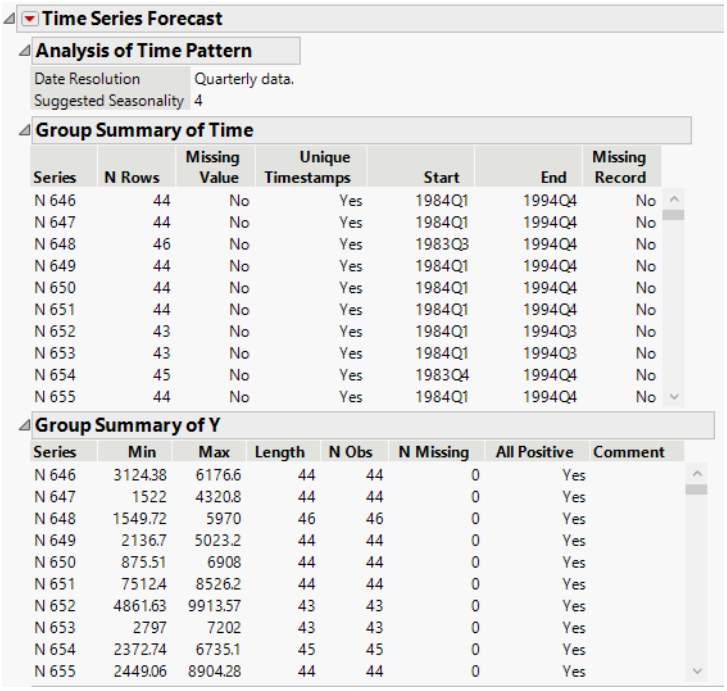
Example of Time Series Forecast 379
Launch the Time Series Forecast Platform 381
Data Format 381
The Time Series Forecast Report 383
Analysis of Time Pattern 383
Group Summary of Time 383
Group Summary of Y 384
Modeling Specifications 384
Time Series Forecast Platform Options 387
Model Reports 389

Example of Time Series Forecast

1. Select **Help > Sample Data Library** and open Time Series/M3C Quarterly.jmp.
2. Select **Analyze > Specialized Modeling > Time Series Forecast**.
3. Select Y and click **Y**.
4. Select Series and click **Grouping**.
5. Select Time and click **Time**.
6. Click **OK**.

   The initial Time Series Forecast report appears. It contains summaries of the Time and Y variables. There is also a Modeling Specifications report that provides options for fitting forecast models.

Figure 19.2 Initial Report for Time Series Forecast



Time Series Forecast

Analysis of Time Pattern

Date Resolution	Quarterly data.
Suggested Seasonality	4

Group Summary of Time

Series	N Rows	Missing Value	Unique Timestamps	Start	End	Missing Record
N 646	44	No	Yes	1984Q1	1994Q4	No
N 647	44	No	Yes	1984Q1	1994Q4	No
N 648	46	No	Yes	1983Q3	1994Q4	No
N 649	44	No	Yes	1984Q1	1994Q4	No
N 650	44	No	Yes	1984Q1	1994Q4	No
N 651	44	No	Yes	1984Q1	1994Q4	No
N 652	43	No	Yes	1984Q1	1994Q3	No
N 653	43	No	Yes	1984Q1	1994Q3	No
N 654	45	No	Yes	1983Q4	1994Q4	No
N 655	44	No	Yes	1984Q1	1994Q4	No

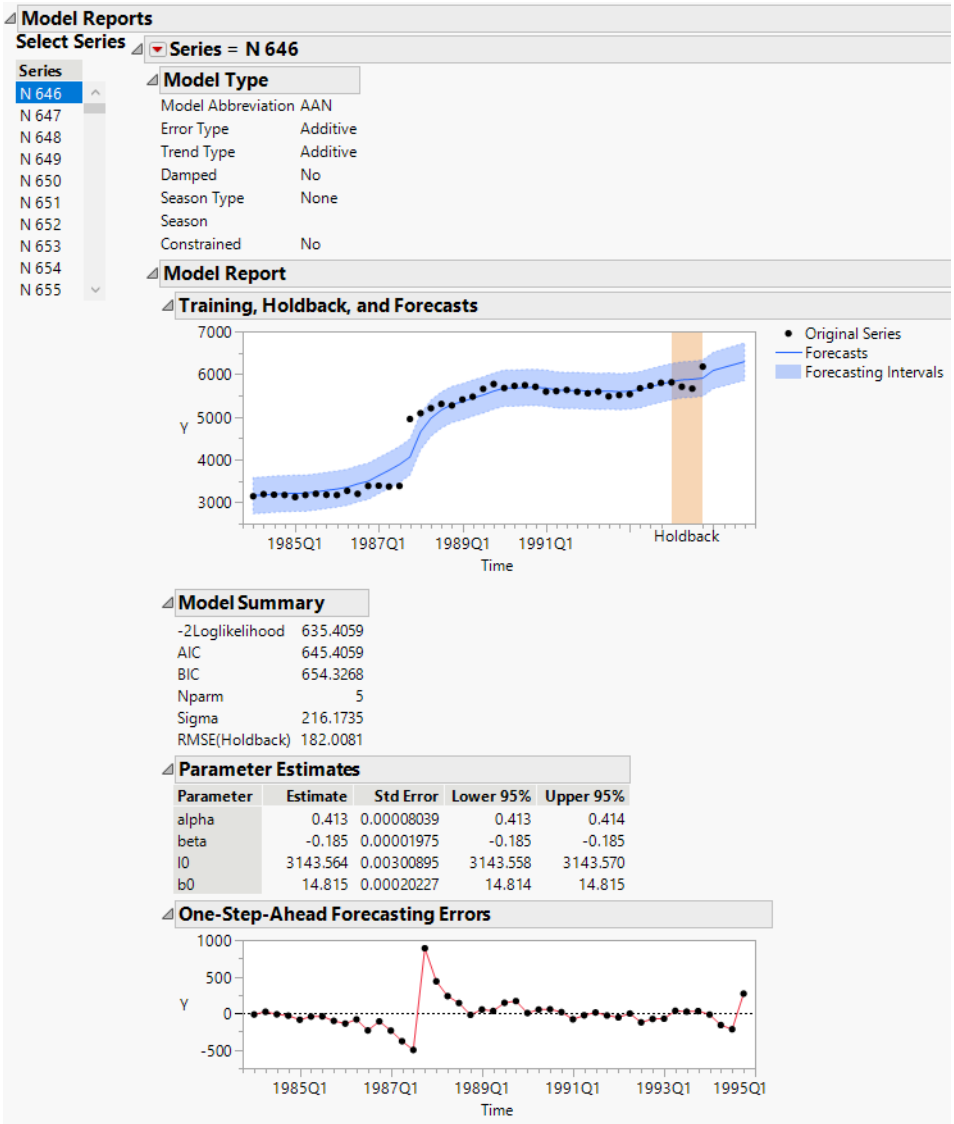
Group Summary of Y

Series	Min	Max	Length	N Obs	N Missing	All Positive	Comment
N 646	3124.38	6176.6	44	44	0	Yes	
N 647	1522	4320.8	44	44	0	Yes	
N 648	1549.72	5970	46	46	0	Yes	
N 649	2136.7	5023.2	44	44	0	Yes	
N 650	875.51	6908	44	44	0	Yes	
N 651	7512.4	8526.2	44	44	0	Yes	
N 652	4861.63	9913.57	43	43	0	Yes	
N 653	2797	7202	43	43	0	Yes	
N 654	2372.74	6735.1	45	45	0	Yes	
N 655	2449.06	8904.28	44	44	0	Yes	

7. In the Model Specifications report, click the Complete Specifications tab.
8. Click the menu below Model Selection Strategy and select Forecasting Performance.
9. Click **Run**.

   This fits a set of recommended models to the data. The best model for each level of the grouping variable is chosen and reported in the Model Reports. See "Model Reports".

Figure 19.3 Model Reports

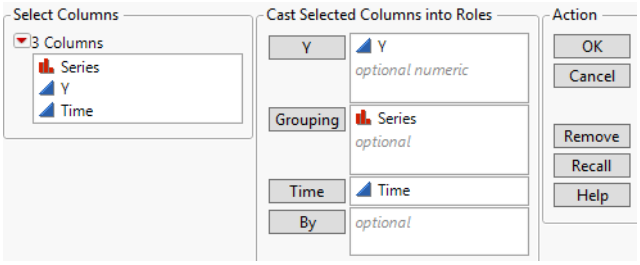


The model report for one series, Series N646, is shown. This corresponds to the series name that is selected in the Select Series list. You can use the down arrow key to scan through the model reports for each series. Alternatively, you can click a different name in the list to display the model report for the selected series.

Launch the Time Series Forecast Platform

Launch the Time Series Forecast platform by selection **Analyze > Specialized Modeling > Time Series Forecast**.

Figure 19.4 The Time Series Forecast Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Time Series Forecast platform launch window contains the following options:

Y Assigns one or more columns as time series variables. Multiple time series can be organized as individual columns in a table or they can be stacked in a single column with one or more grouping variables.

Grouping Assigns one or more columns as grouping variables. Grouping variables are used to identify individual time series when they are in stacked format.

Time Assigns a column for the time variable. If the individual time series are organized as individual columns, the timestamps in the Time column must not contain any duplicates.

By Assigns a column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Data Format

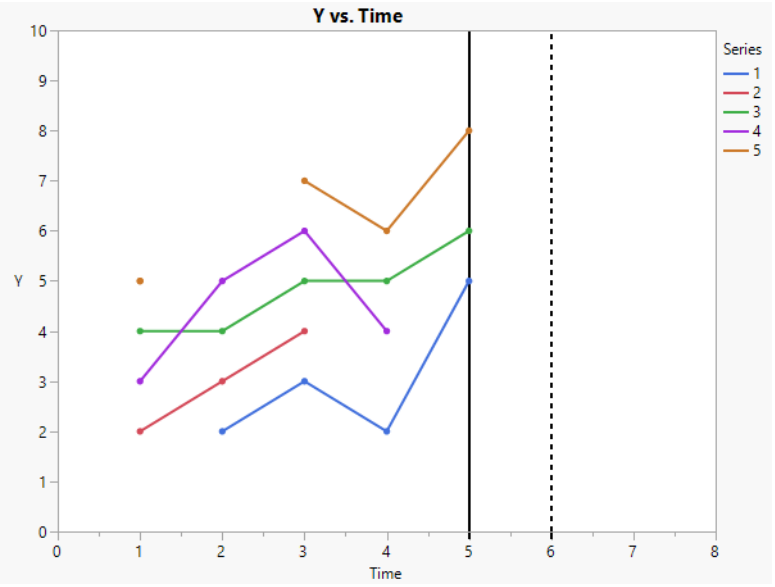
Multiple time series can be organized as individual columns in a table or they can be stacked in a single column with one or more grouping columns to identify the individual series. You can also have a combination of multiple Y columns and multiple grouping columns.

The platform makes the following assumptions about the data:

- All of the time series are recorded at the same time resolution.

- All of the time series are collected up to the same final observed time point. This final observed time point is defined as the latest time point that occurs in any of the individual time series. A series without data up to the final observed time point is assumed to have missing values. For example, in Figure 19.5, the final observed time point is 5, represented by the solid black line. It is assumed that Series 2 has missing observations at time points 4 and 5 and that Series 4 has a missing observation at time point 5.
- All of the forecasts start at the same time point. This starting time point is the next time point following the final observed time point. For example, in Figure 19.5, the final observed time point is 5, represented by the solid black line. Therefore, all of the forecasts start at time point 6, represented by the dashed black line.

Figure 19.5 Demonstration of Assumptions



The Time Series Forecast Report

The Time Series Forecast report contains summary tables based on the variables specified in the launch window. A summary table of the time series by group is always provided. If a Time column is specified, the report also contains a summary of the time pattern and a summary table of the time by group. The report also contains a Modeling Specifications report that enables you to fit models to each time series.

- "Analysis of Time Pattern"
- "Group Summary of Time"
- "Group Summary of Y"
- "Modeling Specifications"

Analysis of Time Pattern

If a time column is specified, the Time Series Forecast platform analyzes the entries of the column to determine a pattern. The platform is designed to recognize a variety of time patterns, including daily, business daily, monthly, bi-weekly, quarterly, and so on. The pattern is reported as the Date Resolution. A Suggested Seasonality is also provided based on the time pattern.

Group Summary of Time

If a time column is specified, the report contains summaries of the individual time series. The summaries contain the following information for each time series:

<Grouping Column Name> (Available only if a Grouping variable is specified.) The level of the grouping variable that defines the individual series when data is provided in the stacked format.

N Rows The number of observations in the series.

Missing Value Indicates whether any of the time values are missing values.

Unique Timestamps Indicates whether the time stamps are unique.

Start The starting time stamp.

End The ending time stamp.

Missing Record Indicates whether there are observations missing from the data table, based on the starting and ending time stamps and the data resolution/pattern.

Group Summary of Y

The report contains a summary table for each Y variable. The summaries contain the following information for each time series:

Variable (Available only if there are multiple Y columns and at least one grouping column.) The column name of the time series variables.

Time Series (Available only if there are multiple series.) The name of the individual time series. When the data are provided in the stacked format, this is the level of the grouping variable. When the data are provided in the wide format, with multiple columns, this is the column name of the individual series.

Note: When the data are provided in the stacked format, the column heading in the table is the name of the Grouping column.

Min The minimum value of Y for the individual series.

Max The maximum value of Y for the individual series.

Length The duration of the series. This is determined based on the starting and ending time stamps and the result of the Analysis of Time Pattern.

N Obs The number of observations. If there are no missing records, this number is the same as the Length.

N Missing The number of missing Y values.

All Positive Indicates whether all of the Y values are positive.

Comment Indicates if there is a problem with the series, such as non-unique time stamps.

Modeling Specifications

The Modeling Specifications report enables you to fit models to the individual time series. A variety of models are considered. Hyndman et al. (2008) defines state space smoothing models based on their error, trend component, and seasonal component:

- The errors can be additive (A) or multiplicative (M).
- The trend component can be none (N), additive (A), additive damped (A_d), multiplicative (M), or multiplicative damped (M_d).
- The seasonal component can be none (N), additive (A), or multiplicative (M).

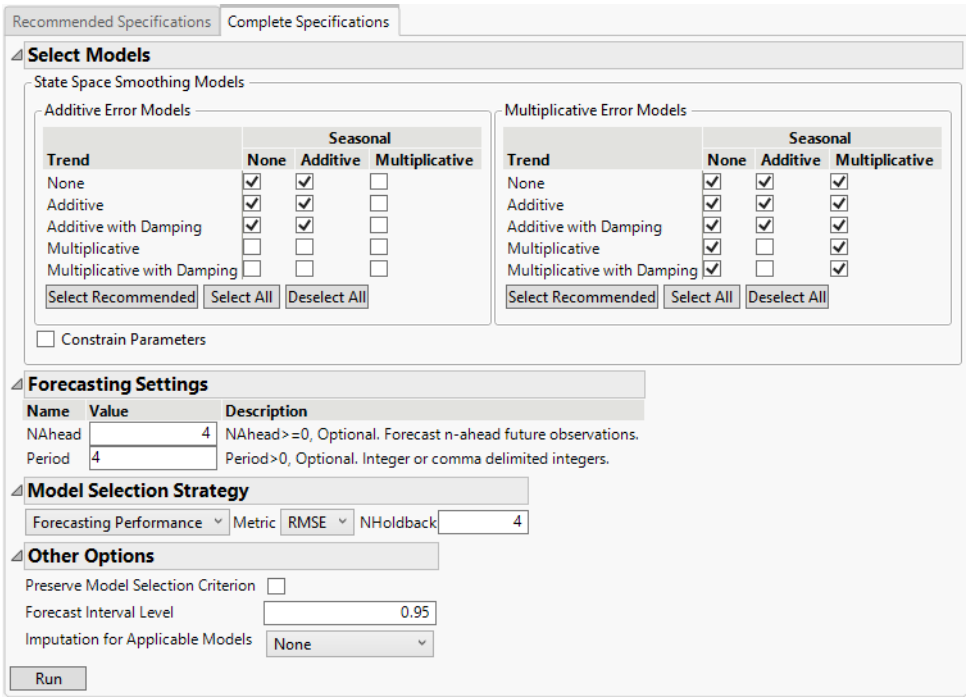
A specific model can be represented by its ETS (Error, Trend, Seasonal). These are the models used in Time Series Forecasting.

There are two tabs in the Modeling Specifications report. Use the Recommended Specifications tab to fit a set of state space smoothing models that is chosen by the platform. Use the Complete Specifications tab to select the specific state space smoothing models that you want to fit. For each individual time series, the best fitting model from the given set is then used for forecasting.

Select Models

Available only in the Complete Specifications tab, shown in "Complete Specifications Tab". The Select Models report enables you to specify the state space smoothing models to fit to the individual time series. Use the check boxes to select the error, trend, and seasonality for the models. Click **Select Recommended** to select the check boxes that correspond to the models recommended by the platform. Click **Select All** to select all check boxes or **Deselect All** to deselect all check boxes. Click **Constrain Parameters** to constrain the parameters in such a way that the further an observation is from the present, the less effect it has on the present state of the model. In State Space Smoothing models, a forecast at time *t*, given all previous observations, is the same as the weighted sum of all observations up to time *t*. The weights are a function of the parameters. Therefore, constraining the parameters ensures that the weights for past observations go to zero and that the further an observation is from the present, the faster the weight goes to zero.

Figure 19.6 Complete Specifications Tab



Forecast Settings

Enables you to specify the following optional settings:

NAhead Specifies the number of steps ahead to forecast. The number of steps must be nonnegative.

Period Specifies seasonality values to be considered in the model fitting process. By default, the period is set to the suggested seasonality from the Analysis of Time Pattern report. If you want to consider models with different seasonality, specify the additional periods here and separate them with a comma. The period must be greater than zero.

Model Selection Strategy

Enables you to specify how the best model is chosen.

Information Criteria Determines the best model based on the specified information criteria. The available information criteria are AIC and BIC.

> **Tip:** Use BIC if the time series is long.

Forecasting Performance Determines the best model based on a performance selection algorithm that involves a holdback set. First, the time series is partitioned into a training set and a holdback set. The value of NHoldback specifies the number of observations in the holdback set. Then, the algorithm fits all recommended or user-specified models on the data in the training set. Forecasts are made on the holdback set using the individual fitted models. These forecasts are compared to the actual holdback observations and the models are evaluated using the specified metric. The model with the best metric is selected for the final model. Last, the selected final model is fit to the entire time series (training set and holdback set) and the refitted model is used to forecast beyond the last observation of the time series.

Metric Specifies the metric to evaluate the forecasts made by the individual model fits. The available metrics are Root Mean Square Error (RMSE), Mean Square Error (MSE), and Mean Absolute Error (MAE).

NHoldback Specifies the number of observations used in the holdback set.

Other Options

Enables you to specify the following options:

Preserve Model Selection Criterion Saves the model selection criterion for the set of models that is fit to the data. After model fitting, this information is shown in the Information Criterion of All Models for All Series report.

Forecast Interval Level The prediction interval level for the forecasts. This changes the width of the shaded area in the forecasting plots.

Imputation for Applicable Models Specifies the imputation method used during the model fitting process.

Note: These options change only the data that are used for model fitting. The raw data are not changed.

None Does not impute missing values.

Last Value Imputes missing values by using the last value that is available before a sequence of missing observations.

Middle Value Imputes missing values by averaging the last value that is available before a sequence of missing observations and the first value that is available after a sequence of missing observations.

Linear Interpolation Imputes missing values by creating a linear interpolation between the last value that is available before a sequence of missing observations and the first value that is available after a sequence of missing observations.

When you click **Run**, the specified set of models is fit. A progress bar appears that reports the number of active threads, the number of finished and total tasks, and the percentage of finished tasks. Note that a task is defined as the fitting of all specified models for one series and response variable combination. When the fitting process is complete, the Model Reports report is shown. See "Model Reports". If you make any changes to the options in the Model Specifications report and click **Run** again, the Model Reports report is replaced with a new report.

Time Series Forecast Platform Options

The Time Series Forecast red triangle menu contains the following options:

Report Options (Effective only after fitting a model.) Contains the following report options:

Set Forecast Interval Level Enables you to change the forecast interval level. The individual model reports update automatically.

Set NAhead Enables you to change the number of steps ahead to forecast. The individual model reports update automatically.

Set NHoldout Enables you to change the duration of the holdout set. The individual model reports update automatically.

Save Results (Effective only after fitting a model.) Saves the forecasts to a separate data table named Forecasts. The Forecasts data table has the same layout and format as the original data table. If a holdout set is specified by NHoldout, an additional data table that saves the actual values from the holdout data and the forecasts is produced. You can use this table to calculate different types of error estimates.

The following additional save options are shown in the Request Outputs window:

Save Forecast Intervals Saves the lower and upper prediction intervals as columns in the Forecasts data table.

Save One-Step-Ahead Predictions Stacks the one-step-ahead predictions and the forecasts and saves them to a separate data table.

Save Original Series Stacks the original series and the forecasts and saves them to a separate data table.

Save Forecast Results to Original Data Table Saves the results to new columns in the original data table.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Model Reports

The Model Reports report contains a Select Series list and one or more individual series modeling reports. The individual reports that are displayed are determined by the Select Series list. Click on the name of a series in this list to show the report for the selected series. If you select more than one series name by using either Ctrl or Shift, an individual model fit report for each of the selected series is shown. If you deselect one or more series from the Select Series list, the reports for the corresponding series are removed.

The Select Series list is linked to the data table. When you select a series in the Select Series list, the observations for the series and response variable combination are also selected in the data table. The Select Series list is also linked to the Group Summary of Y table. If a Time variable is specified, the Select Series list is also linked to the Group Summary of Time table. When you select a series in the Select Series list, the corresponding entries are also selected in the tables.

Each individual model report contains the following information:

Model Type Describes the best fitting model for the series in terms of error, trend, and seasonality.

Model Report A group of reports based on the model that is fit to the series.

Series and Forecasts (Available only when Information Criteria is specified as the model selection strategy.) A plot of the original series and forecasts. During the time period of the original series, the one-step-ahead predictions are shown. Beyond the time period of the original series, the forecasts are shown. The shaded region represents the forecasting interval.

Note: All of the forecasts start at the time point that immediately follows the last time point across all individual series. See "Data Format".

Training, Holdback, and Forecasts (Available only when Forecast Performance is specified as the model selection strategy.) A plot of the training series, holdback series, and the forecasted series. A shaded vertical region represents the holdback series. Since this is the model refit on the entire time series, the one-step-ahead predictions are shown during the time period of the entire original series. During the time period past the last observation, the forecasts are shown. The shaded region represents the forecasting interval.

Model Summary Provides the -2Loglikelihood, the AIC, the number of parameters, and the sigma value. The sigma value is the estimate of the one-step-ahead forecast standard deviation. If Forecast Performance is specified as the model selection strategy, the root mean square error (RMSE) for the holdback set is also shown. See *Fitting Linear Models*.

Parameter Estimates Provides the parameter estimates for the model. The parameters correspond to the parameters in the best fitting model, which is described in the Model Type report.

One-Step-Ahead Forecasting Errors A plot of the one-step-ahead forecasting errors over time. For each time point t, the error is calculated as $e(t) = y(t) - \mu(t)$.

One-Step-Ahead Relative Forecasting Errors for Multiplicative Error Model (Available only for multiplicative error models.) A plot of the one-step-ahead relative forecasting errors over time. For each time point t, the error is calculated as $e(t) = [y(t) - \mu(t)]/\mu(t)$.

Chapter 20

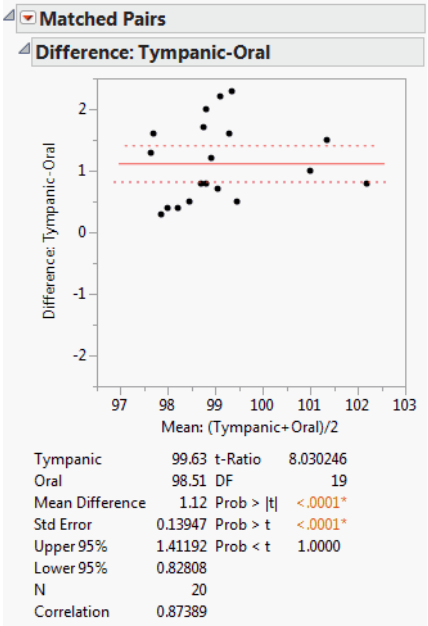
Matched Pairs Analysis

Compare Measurements on the Same Subject

The Matched Pairs platform compares the means between two or more correlated variables and assesses the differences. For example, you might compare a blood pressure measurement taken on the same subject before a treatment and again after the treatment. A statistical method called the *paired t test* takes the correlated responses into account.

The platform produces a graph of the paired differences by the paired means, and the paired *t* test results for all three alternative hypotheses. Additional features provide for more than two matched responses and for a grouping column to test across samples, in a simple version of repeated measures analysis.

Figure 20.1 Example of Matched Pairs Analysis



Contents

Overview of the Matched Pairs Platform 393
Example of Comparing Matched Pairs 393
Launch the Matched Pairs Platform 395
Multiple Y Columns 395
The Matched Pairs Report 396
Difference Plot and Report 397
Across Groups 397
Matched Pairs Platform Options 398
Example Comparing Matched Pairs across Groups 399
Statistical Details for the Matched Pairs Platform 400
Graphics for Matched Pairs 401
Correlation of Responses 402

Overview of the Matched Pairs Platform

The Matched Pairs platform compares row-by-row differences between two response columns using a paired *t* test. Often, the two columns represent measurements on the same subject before and after some treatment. Alternatively, the measurements could represent data taken on the same subject with two different instruments.

If you have paired data arranged in two data table columns, then you are ready to use the Matched Pairs platform. However, if all of your measurements are in a single column, then perform one of the following tasks:

- Use the **Split** option in the **Tables** menu to split the column of measurements into two columns. Then you can use the Matched Pairs platform.
- For two response columns, create a third column that calculates the difference between the two responses. Then test that the mean of the difference column is zero with the Distribution platform.
- For the two responses stored in a single column, you can do a two-way analysis of variance. One factor (the ID variable) identifies the two responses and the other factor identifies the subject. Use the Fit Y by X Oneway platform with a blocking variable (the subject column), or use the Fit Model platform to do a two-way ANOVA. The test on the ID factor is equivalent to the paired *t* test.

Note: If the data are paired, do not do a regular independent *t* test. Do not stack the data into one column and use the Fit Y by X One-way ANOVA on the ID without specifying a block variable. To do this has the effect of ignoring the correlation between the responses. This causes the test to overestimate the effect if responses are negatively correlated, or to underestimate the effect if responses are positively correlated.

Example of Comparing Matched Pairs

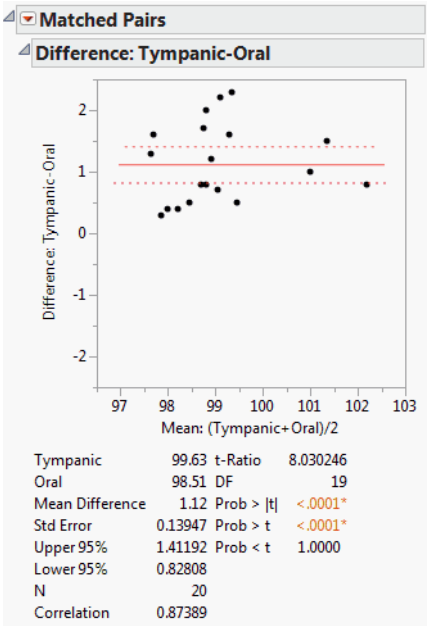
This example uses the Therm.jmp sample data table. The data contains temperature measurements on 20 people. Temperature is measured using two types of thermometers: oral and tympanic (ear). You want to determine whether the two types of thermometers produce equal temperature readings. Note that the differences in temperature between the different people are not important. The matched pairs analysis is testing the differences between the thermometers.

1. Select **Help > Sample Data Library** and open Therm.jmp.
2. Select **Analyze > Specialized Modeling > Matched Pairs**.
3. Select Oral and Tympanic and click **Y, Paired Response**.

4. Click **OK**.

   The report window appears.

Figure 20.2 The Matched Pairs Report Window



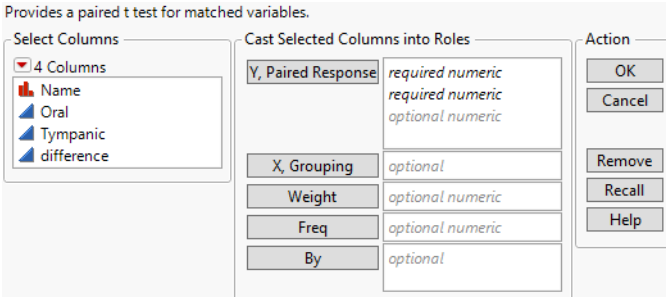
The results show that, on average, the tympanic thermometer measures 1.12 degrees higher than the oral thermometer. The small *p*-value (Prob > |t|) indicates that this difference is statistically significant, and not due to chance.

Note that this matched pairs analysis does not indicate which thermometer is correct (if either), but indicates only that there is a difference between the thermometers.

Launch the Matched Pairs Platform

Launch the Matched Pairs platform by selecting **Analyze > Specialized Modeling > Matched Pairs**.

Figure 20.3 The Matched Pairs Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Paired Response Provide the two response columns. For information about analyzing more than two responses, see "Multiple Y Columns".

X, Grouping Provide a grouping variable to compare the differences across groups. See "Across Groups".

Weight Identifies one column whose numeric values assign a weight to each row in the analysis.

Freq Identifies one column whose numeric values assign a frequency to each row in the analysis.

By Performs a separate matched pairs analysis for each level of the By variable.

After you click **OK**, the Matched Pairs report window appears. See "The Matched Pairs Report".

Multiple Y Columns

You can have more than two responses. If the number of responses is odd, all possible pairs are analyzed. The following table shows an example for three responses.

Y1 by Y2	Y1 by Y3

Y2 by Y3

If the number of responses is even, the Matched Pairs platform asks whether you want to do all possible pairs. If you do not do all possible pairs, adjacent responses are analyzed as a pair. The following table shows the arrangement of analyses for four responses.

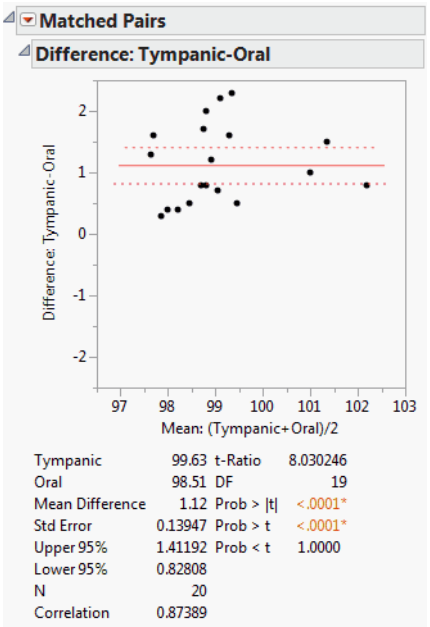
Y1 by Y2	Y3 by Y4

The Matched Pairs Report

Follow the instructions in "Example of Comparing Matched Pairs" to produce the report window shown in Figure 20.4.

The Matched Pairs report shows a Tukey mean-difference plot, summary statistics, and the results of the paired *t* test. See "Difference Plot and Report". If you specified an **X, Grouping** variable, the report also includes the Across Groups report. See "Across Groups".

Figure 20.4 Example of Matched Pairs Report



Note: The red triangle menu provides additional options that can add reports to the initial report window. See "Matched Pairs Platform Options".

Difference Plot and Report

The Difference plot shows differences by means. In the Difference plot, note the following:

- The mean difference is shown as the horizontal line, with the 95% confidence interval above and below shown as dotted lines. If the confidence region includes zero, then the means are not significantly different at the 0.05 level. In this example, the difference is significant.
- If you add a reference frame, the mean of pairs is shown by the vertical line. For more information about a reference frame, see "Matched Pairs Platform Options".

The Difference report shows the mean of each response, the difference of the means, and a confidence interval for the difference. The Difference report also shows the results of the paired *t* test.

Across Groups

Note: The Across Groups report appears only if you have specified an **X, Grouping** variable.

The Across Groups analysis corresponds to a simple repeated measures analysis. (You can get the same test results using the **Manova** personality of the Fit Model platform.)

Mean Difference Shows the mean of the difference across rows in each group between the two paired columns. In other words, this is the within-subject by across-subject interaction, or split-plot by whole-plot interaction.

Mean Mean Shows the mean of the mean across rows in each group across the two paired columns. In other words, this is the across-subject or whole-plot effect.

Test Across Groups Two *F* tests determine whether the across-groups values are different:

Mean Difference Tests that the change across the pair of responses is different in different groups.

Mean Mean Tests that the average response for a subject is different in different groups

Related Information

- "Example Comparing Matched Pairs across Groups"

Matched Pairs Platform Options

The Matched Pairs red triangle menu contains the following options:

Plot Dif by Mean Shows or hides the plot of the paired differences by paired means. For a detailed description of this plot, see "Difference Plot and Report".

Plot Dif by Row Shows or hides the plot of paired differences by row number.

Reference Frame Shows or hides the reference frame on the Plot Dif by Mean plot. A rectangle showing where a plot of Y2 by Y1 would be located inside the plot, tilted and possibly squished. A vertical red line is shown representing the mean of means. The reference frame is shown initially when the range of the differences is greater than half the range of the data.

Wilcoxon Signed Rank Shows or hides the Wilcoxon signed rank test. The Wilcoxon signed rank test is applied to the paired differences. It is a nonparametric test that compares the sizes of the positive differences to the sizes of the negative differences. The test uses the Pratt method to address zero differences. The test also assumes that the distribution of differences is symmetric. See *Basic Analysis*. See also Lehman (2006), Conover (1999, p. 350), and Cureton (1967).

Sign Test Shows or hides the sign test. This is a nonparametric version of the paired *t* test that uses only the sign (positive or negative) of the difference for the test.

Set α Level Changes the alpha level used in the analyses. Affects the confidence intervals in the report and on the plot.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Example Comparing Matched Pairs across Groups

This example uses the Dogs.jmp sample data table. This example shows you how to produce both a Matched Pairs Across Groups report and the corresponding MANOVA report using Fit Model.

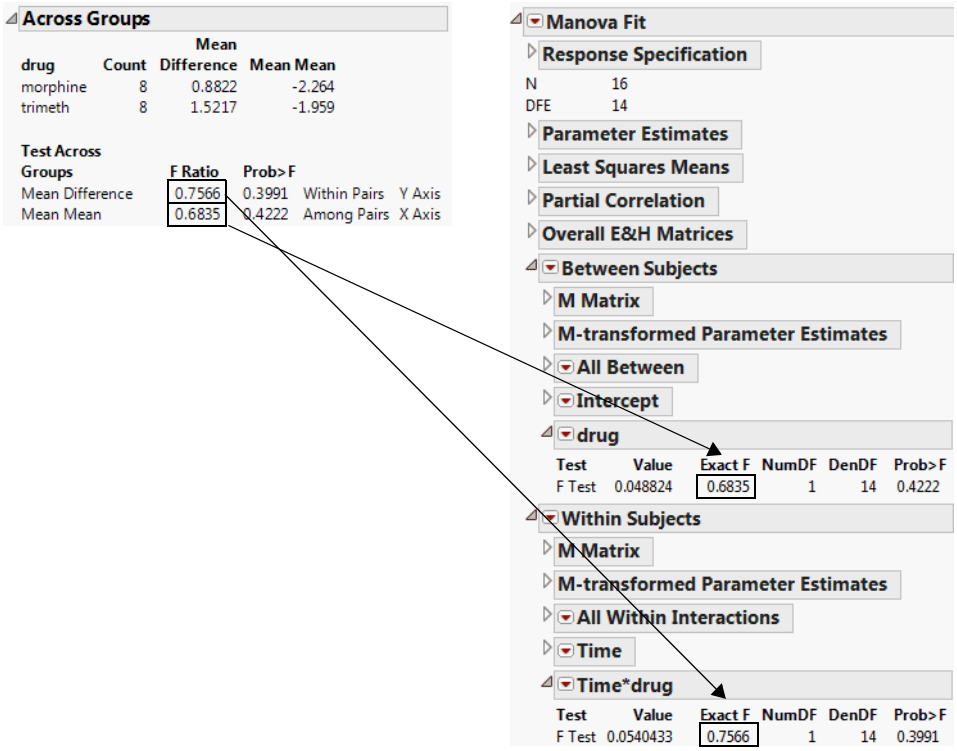
1. Select **Help > Sample Data Library** and open Dogs.jmp.
2. Select **Analyze > Specialized Modeling > Matched Pairs**.
3. Select LogHist0 and LogHist1 and click **Y, Paired Response**.
4. Select drug and click **X, Grouping**.
5. Click **OK**.

The report on the left in Figure 20.5 appears.

Now produce the Fit Model report using the same data table.

1. Select **Analyze > Fit Model**.
2. Select LogHist0 and LogHist1 and click **Y**.
3. Select drug and click **Add**.
4. Select the **Manova** personality.
5. Click **Run Model**.
6. In the Response Specification report, select **Repeated Measures** from the **Choose Response** menu.
7. Click **OK**.

Figure 20.5 Examples of Matched Pairs across Groups and Fit Model MANOVA with Repeated Measures



The F Ratio for the Mean Difference in the Across Groups report corresponds to the F Ratio for Time*drug under the Within Subjects report. The F Ratio for the Mean Mean in the Across Groups report corresponds to the F Ratio for drug under Between Subjects in the Manova Fit report.

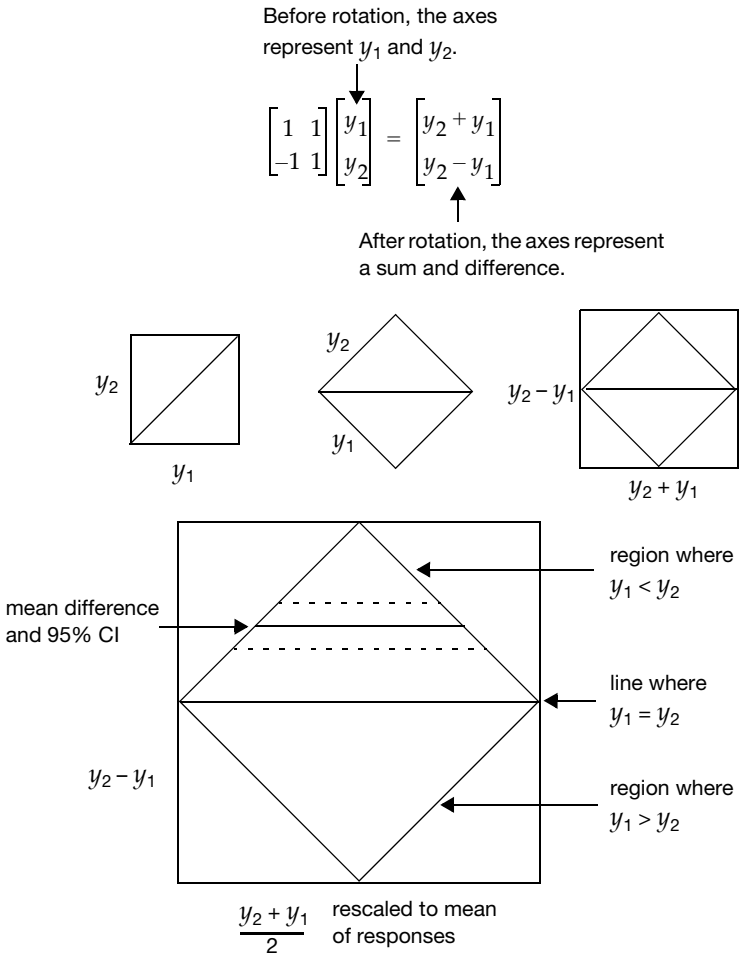
Statistical Details for the Matched Pairs Platform

- "Graphics for Matched Pairs"
- "Correlation of Responses"

Graphics for Matched Pairs

The primary graph in the platform is a Tukey mean-difference (Cleveland 1994, p. 130). This graph plots the difference of the two responses on the vertical axis against the mean of the two responses on the horizontal axis. This graph is the same as a scatterplot of the two original variables, but turned 45 degrees. A 45 degree rotation and rescaling turns the original coordinates into a difference and a mean.

Figure 20.6 Example of Transforming to Difference by Mean, Rotated by 45 Degrees

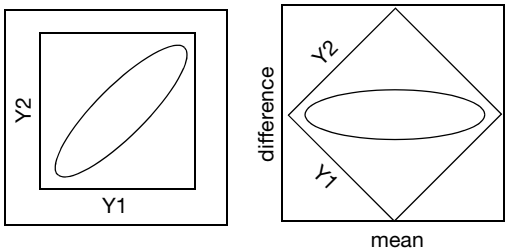


Correlation of Responses

In most cases where the pair of measurements is taken from the same individual at different times, they are positively correlated. However, if they represent competing responses, the correlation can be negative.

Figure 20.7 shows how the positive correlation of the two responses becomes the small variance on the difference (the vertical axis). If the correlation is negative, the ellipse is oriented in the other direction and the variance of the rotated graph is large on the vertical axis.

Figure 20.7 Examples of Positive Correlation Before and After Rotation



Chapter 21

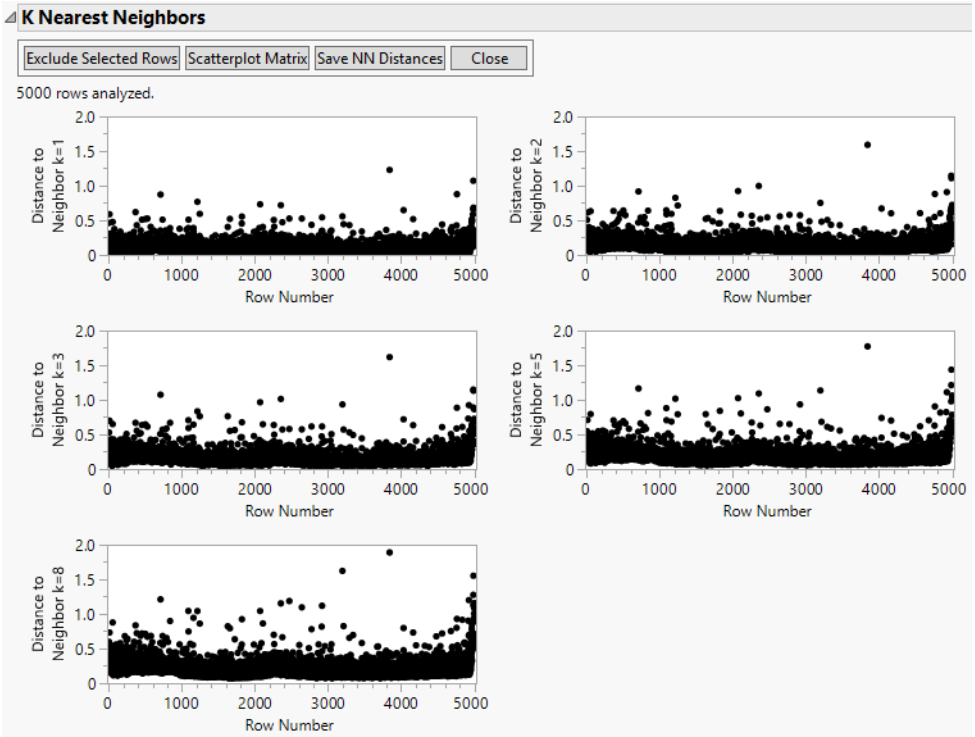
Modeling Utilities

Exploring Data for Outliers, Missing Values, and Patterns

Modeling Utilities is a collection of utilities that are designed to assist in the data cleaning and pre-processing stages of data analysis. Each utility has exploratory tools to give you a more thorough understanding of your data. With Modeling Utilities, you can do the following:

- Explore outliers in both the univariate and multivariate cases.
- Explore and impute missing values in your data.
- Explore and find unusual features or patterns in your data.

Figure 21.1 Multivariate k-Nearest Neighbor Outlier Example



Contents

Explore Outliers Utility 405
Example of the Explore Outliers Utility 405
Launch the Explore Outliers Utility 409
Quantile Range Outliers 410
Robust Fit Outliers 413
Robust PCA Outliers 415
K Nearest Neighbor Outliers 418
Explore Outliers Utility Options 420
Additional Examples of the Explore Outliers Utility 421
Explore Missing Values Utility 423
Example of the Explore Missing Values Utility 424
The Missing Value Report 426
Explore Missing Values Utility Options 432
Explore Patterns Utility 433
Example of the Explore Patterns Utility 433
Launch the Explore Patterns Utility 436
Explore Patterns Report 437
Explore Patterns Utility Options 438
Additional Example of the Explore Patterns Utility 442
Statistical Details for the Explore Patterns Utility 444

Explore Outliers Utility

The Explore Outliers tool provides four different options to identify, explore, and manage outliers. Exploring and understanding outliers in your data is an important part of analysis. Outliers in data can be due to mistakes in data collection or reporting, measurement systems failure, the inclusion of error or missing value codes in the data set, or simply an unusual value. The presence of outliers can distort estimates and bias results toward those outliers.

Outliers also inflate the sample variance. Sometimes retaining outliers in data is necessary, however, and removing them could underestimate the sample variance and bias the data in the opposite direction.

Whether you remove or retain outliers, it is a good practice to locate them. There are many ways to visually inspect for outliers. For example, box plots, histograms, and scatter plots can easily display these extreme values. See *Discovering JMP*.

The Explore Outliers tool provides the following options:

Univariate There are two options for exploring outliers in your univariate data.

Quantile Range Outliers Uses the quantile distribution of each column to identify outliers as extreme values. This tool is useful for discovering missing value or error codes within the data. This is the recommended method to begin exploring outliers in your data. See "Quantile Range Outliers".

Robust Fit Outliers Finds robust estimates of the center and spread of each column and identifies outliers as those data points that are far from those values. See "Robust Fit Outliers".

Multivariate There are two options for exploring outliers in your multivariate data.

Robust PCA Outliers Decomposes data into a low-rank matrix and residuals and uses the residuals to detect outliers. See "Robust PCA Outliers".

K Nearest Neighbor Outliers Identifies outliers as values that are far from their k-nearest neighbors. See "K Nearest Neighbor Outliers".

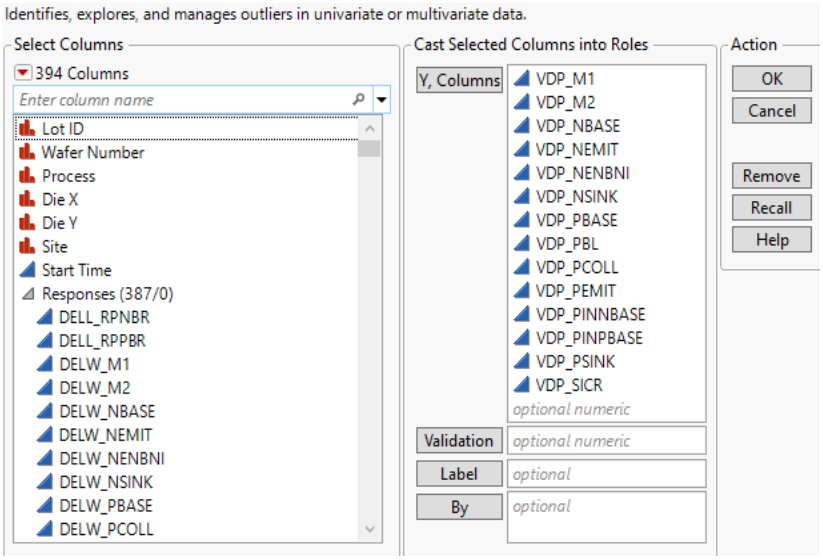
Example of the Explore Outliers Utility

Use the Explore Outliers utility to identify outliers that can then be examined using the Distribution platform. The Probe.jmp sample data table contains 387 characteristics (the Responses column group) measured on 5800 semiconductor wafers. The Lot ID and Wafer Number columns uniquely identify the wafer. You are interested in identifying outliers within a select group of columns of the data set.

1. Select **Help > Sample Data Library** and open the Probe.jmp sample data table.

2. Select **Analyze > Screening > Explore Outliers**.
3. Click the triangle next to Responses(387/0) to show all of the columns in the group.
4. Select columns VDP_M1 through VDP_SICR and click **Y, Columns**. There should be 14 columns selected.

Figure 21.2 Explore Outliers Launch Window



5. Click **OK**.
6. Click **Quantile Range Outliers**.

   The Quantile Range Outliers report shows each column and lists the number and identity of the outliers found.
7. In the Quantile Range Outliers report, select **Show only columns with outliers.** This limits the list of columns to only those that contain outliers.

   Note that several columns contain outlier values of 9999. Many industries use nines as a missing value code.
8. In the Nines report, select each column.
9. Click **Add Highest Nines to Missing Value Codes**.

   A JMP Alert indicates that you should use the **Save As** command to preserve your original data.
10. Click **OK**.
11. In the Quantile Range Outliers report, click **Rescan**.
12. Select **Restrict search to integers**.

In continuous data, integer values are often error codes or other coded data values. Notice that no additional error codes are included in this set of columns.

13. Deselect **Restrict search to integers**.

Examine the Data

1. Select all of the remaining columns in the Quantile Range Outliers report.
2. Click **Select Rows**.
3. Select **Analyze > Distribution**.
4. Assign the selected columns to the **Y, Columns** role. Because you selected these column names in the Quantile Range Outliers report, they are already selected in the Distribution launch window.
5. Click **OK**.

Figure 21.3 Distribution of Columns with Outliers Selected

In columns VDP_M1 and VDP_PEMIT, notice that some of the selected outliers are somewhat close to the majority of data. For the rest of the columns, the selected outliers appear distant from the majority of data. Investigate the data points and exclude them from your analyses.

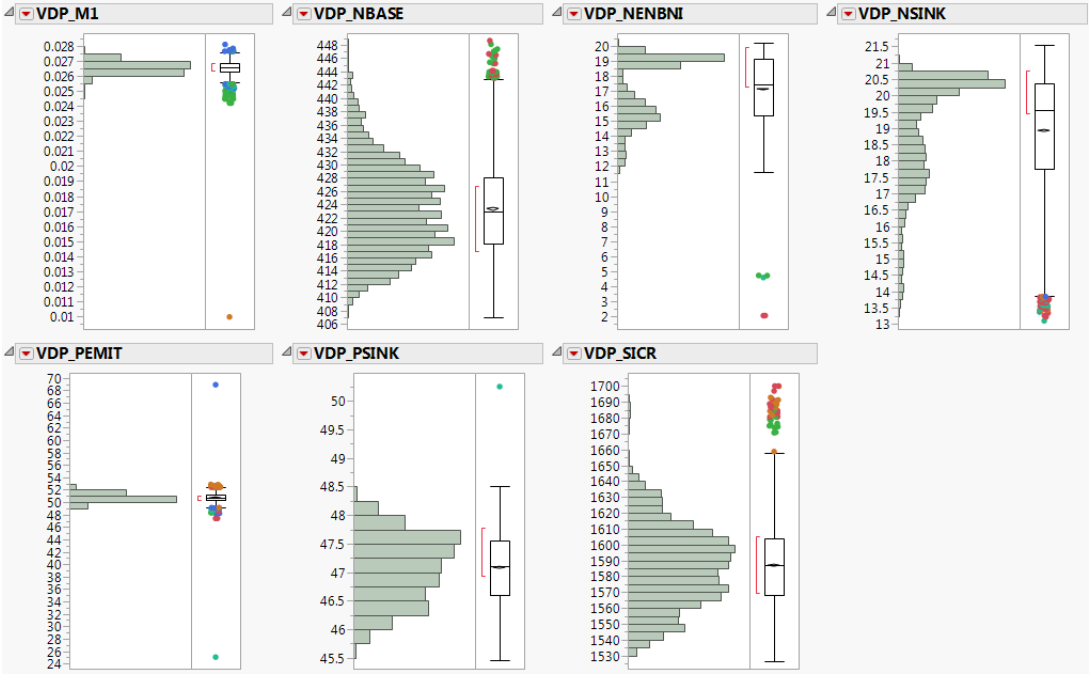
Refine Excluded Outliers

1. In the Quantile Range Outliers report, hold Ctrl and deselect columns VDP_M1 and VDP_PEMIT.
2. With the remaining columns selected in the report, click **Exclude Rows**.
3. Change Q to 20.
4. Click **Rescan**.
5. Select columns VDP_M1 and VDP_PEMIT in the report.
6. Click **Select Rows**.

Reexamine the Data

1. Examine the Distributions report again. Notice the selected outliers are now separate enough from the majority of the data to select and exclude them from your analyses.
2. In the Quantile Range Outliers report, click **Exclude Rows**.
3. In the Distributions report, click the Distributions red triangle and select **Redo > Redo Analysis**.

Figure 21.4 Distributions of Columns with Outliers Excluded



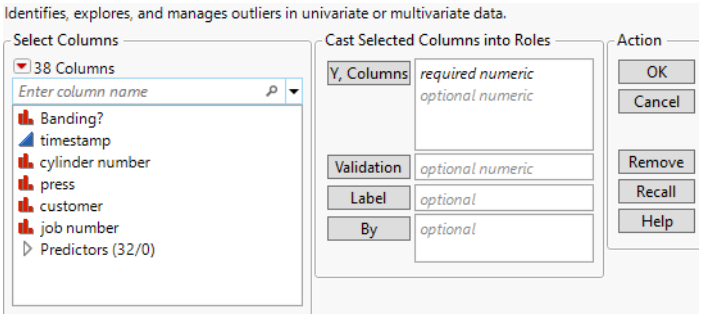
The displays of the distributions of the data are now more informative without the outliers.

Launch the Explore Outliers Utility

To launch Explore Outliers, select **Analyze > Screening > Explore Outliers**.

Note: The Explore Outliers commands analyze only columns with a Continuous modeling type. Other columns can be entered in the launch window but are ignored.

Figure 21.5 Explore Outliers Utility Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns Specifies the columns that you want to analyze.

Validation Specifies a validation column that is used for Robust PCA Outliers.

Label Specifies a column that replaces row numbers in multivariate analysis reports with labels.

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

After you click **OK**, the Explore Outliers report appears. You are presented with the following four outlier analysis commands:

- "Quantile Range Outliers"
- "Robust Fit Outliers"
- "Robust PCA Outliers"
- "K Nearest Neighbor Outliers"

Tip: To run an outlier analysis across all levels of a **By** variable, press Ctrl and click the desired outlier analysis command button.

Quantile Range Outliers

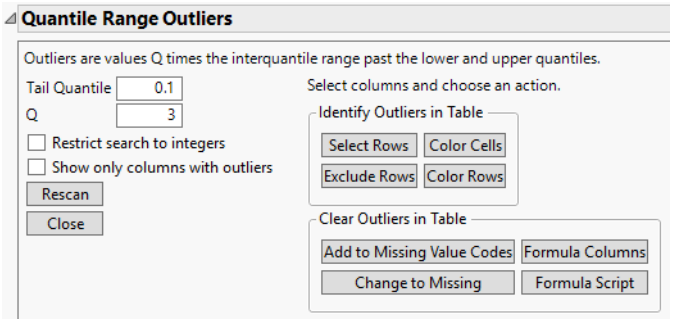
The Quantile Range Outliers method of outlier detection uses the quantile distribution of the values in a column to locate extreme values. Quantiles are useful for detecting outliers because there is no distributional assumption associated with them. Data are simply sorted from smallest to largest. For example, the 20th quantile is the value at which 20% of values are smaller. Extreme values are found using a multiplier of the interquantile range, the distance between two specified quantiles. For more information about how quantiles are computed, see *Basic Analysis*.

The Quantile Range Outliers utility is also useful for identifying missing value codes stored within the data. As noted earlier, in some industries, missing values are entered as nines (such as 999 or 9999). This utility finds any nines greater than the upper quartile as suspected missing value codes. The utility then enables you to add those missing value codes as a column property in the data table.

Quantile Range Outliers Options

The Quantile Range Outliers panel enables you to specify how outliers are to be identified and how you want to manage them.

Figure 21.6 Quantile Range Outliers Window



An outlier is considered any value more than *Q* times the interquantile range from the lower and upper quantiles. You can adjust the value of *Q* and the size of the interquantile range.

Tail Quantile The probability for the lower quantile that is used to calculate the interquantile range. The probability of the upper quantile is considered 1 - Tail Quantile. For example, a Tail Quantile value of 0.1 means that the interquantile range is between the 0.1 and 0.9 quantiles of the data. The default value is 0.1.

Q The multiplier that determines the outlier threshold. Values that fall beyond *Q* times the interquantile range past the Tail Quantile or 1 - Tail Quantile values are identified as outliers. Large values of *Q* provide a more conservative set of outliers than small values. The default is 3.

Restrict search to integers Restricts outlier values to only integer values. This setting limits the search for outliers in order to find industry-specific missing value codes and error codes.

Show only columns with outliers Limits the list of columns in the report to those that contain outliers.

After the report is displayed using your specifications, there are many ways to act on these extreme values. You can select the outliers in a column by selecting the specified column in the Quantile Range Outliers report.

Tip: If no columns are selected in the report and you click one of the following buttons, a JMP Alert appears that enables you to select all of the columns.

Select Rows Selects the rows of outliers in the selected columns in the data table.

Exclude Rows Turns on the exclude row state for outliers in the selected columns in the Quantile Range Outliers Report. Click **Rescan** to update the Quantile Range Outliers report.

Color Cells Colors the cells of the selected outliers in the data table.

Color Rows Colors the rows containing outliers for the selected columns in the data table

Add to Missing Value Codes Adds the selected outliers to the missing value codes column property. Use this option to identify known missing value or error codes within the data. Missing value and error codes are often integers and are sometimes a series of nines. Click **Rescan** to update the Quantile Range Outliers report.

Note: Add to Missing Value Codes is not available with Quantile Range Outliers if a By variable is specified in the launch window.

Change to Missing Changes the outlier value to a missing value in the data table. Use caution when changing values to missing. Change values to missing only if the data are known to be invalid or inaccurate. Click **Rescan** to update the Quantile Range Outliers report.

Note: If the selected outlier has been added to the missing value codes, the outlier is not changed to a missing value.

Formula Columns Creates a new formula column for each column that is specified in the launch window. Each new column contains the original column's value if the value is within the outlier limits and is set to missing otherwise. The new columns are prefixed or suffixed by a user specified name to distinguish them from the original columns. By default, the suffix is set to "Culled".

Formula Script Creates a script that is added to the data table. When the script is run, it creates a new formula column for each column that is specified in the launch window. Each new column contains the original column's value if the value is within the outlier limits and is set to missing otherwise. The new columns are prefixed or suffixed by a user specified name to distinguish them from the original columns. By default, the suffix is set to "Culled".

Rescan Rescans the data after outlier actions have been taken.

Note: Press Ctrl and click **Rescan** to rescan across all command groups.

Close Closes the Quantile Range Outliers panel.

Note: Press Ctrl and click **Close** to close all command windows.

Quantile Range Outliers Report

The Quantile Range Outliers report lists all columns with the outliers found using the specified options. The report shows values for the upper and lower quantiles along with their low and high thresholds. Values outside of these threshold limits are considered outliers. The number of outliers in each column is indicated. The values of each outlier are listed in the last column of the report. Outliers that occur more than once in a column are listed with their count in parentheses. To remove columns without outliers from the report, select **Show only columns with outliers**.

There are several things to look for when reading this report.

- Error codes. For some continuous data, suspiciously high integer values are likely to be error codes. For example, if your upper and lower quantile values are all less than 0.5, outliers such as 1049 or -777 are likely to be error codes.
- Zeros. Sometimes zeros can indicate missing values. If the majority of your data is reasonably large and you notice zeros as outliers, they are likely to be due to missing data.

Nines Report

The Nines report within the Quantile Range Outliers window shows a list of columns that contain probable missing value codes. These missing value codes are a series of nines (usually 9999) and are the highest number that is all nines and also higher than the upper quantile. If the count is high, it is likely that these outliers are actually missing value codes. If the count is very low, you should explore further to determine whether the value is an outlier or a missing value code. The Nines Report includes the upper quantile value.

This report is displayed only when probable missing value codes are identified.

Add Highest Nines to Missing Value Codes Adds the selected outlier values to the missing value codes column property. You must click **Rescan** to update the Quantile Range Outliers report.

Note: Add Highest Nines to Missing Value Codes is not available with Quantile Range Outliers if a By variable is specified in the launch window.

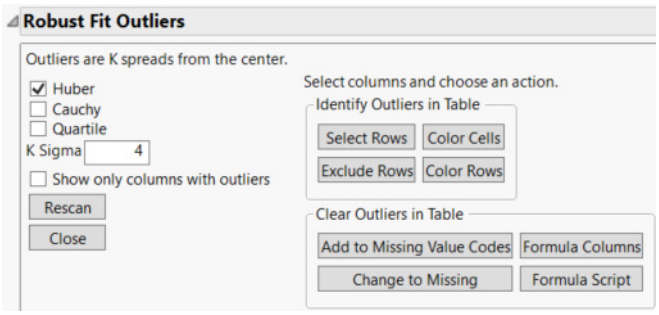
Change Highest Nines to Missing Replaces the selected outlier values with missing values in the data table.

Note: The first time you use choose an action (such as **Change to Missing** or **Exclude Rows**) to change your data, the alert window warns you to use the **Save As** command to save your data table as a new file to preserve a copy of your original data. When this window appears, click **OK**. If you decide to save your new data file, you will automatically be prompted to save the file with a new name.

Robust Fit Outliers

Robust estimates of parameters are less sensitive to outliers than non-robust estimates. Robust Fit Outliers provides several types of robust estimates of the center and spread of your data to determine thresholds for identifying outliers.

Figure 21.7 Robust Fit Outliers Window



Robust Fit Outliers Options

Given a robust estimate of the center and spread, outliers are defined as those values that are *K* times the robust spread from the robust center. The Robust Fit Outliers window provides several options for calculating the robust estimates and multiplier *K* as well as provides tools to manage the outliers found.

Huber Uses Huber M-Estimation to estimate center and spread. This option is the default. See Huber and Ronchetti (2009).

Cauchy Assumes a Cauchy distribution to calculate estimates for the center and spread. Cauchy estimates have a high breakdown point and are typically more robust than Huber estimates. However, if your data are separated into clusters, the Cauchy distribution tends to consider only the half of the data that makes closer clusters, ignoring the rest.

Quartile Uses the interquartile range (IQR) to estimate the spread. The estimate for the center is the median. The estimate for spread is the IQR divided by 1.34898. Dividing the IQR by this factor makes the spread correspond to one standard deviation if the data are normally distributed.

K The multiplier that determines outliers as *K* times the spread away from the center. Large values of *K* provide a more conservative set of outliers than small values. The default is 4.

Show only columns with outliers Limits the list of columns in the report to those that contain outliers.

Once the report is displayed using your specifications, there are many ways to explore these extreme values. You can select the outliers in a row by selecting the specified row in the Robust Estimates and Outliers report.

Tip: If no columns are selected in the report and you click one of the following buttons, a JMP Alert appears that enables you to select all of the columns.

Select Rows Selects the rows containing outliers for the selected columns in the data table.

Exclude Rows Sets the Exclude Row state for outliers in the selected columns in the data table. Click **Rescan** to update the Robust Estimates and Outliers report.

Color Cells Colors the cells of the selected outliers in the data table.

Color Rows Colors the rows containing outliers for the selected columns in the data table.

Add to Missing Value Codes Adds the selected outliers to the missing value codes column property for the selected columns. Use this option to identify known missing value or error codes within the data. Click **Rescan** to update the Robust Estimates and Outliers report.

Note: Add to Missing Value Codes is not available with Robust Fit Outliers if a By variable is specified in the launch window.

Change to Missing Changes the outlier value to a missing value in the data table. Click **Rescan** to update the Robust Estimates and Outliers report.

Formula Columns Creates a new formula column for each column that is specified in the launch window. Each new column contains the original column's value if the value is within the outlier limits and is set to missing otherwise. The new columns are prefixed or suffixed by a user specified name to distinguish them from the original columns. By default, the suffix is set to "Culled".

Formula Script Creates a script that is added to the data table. When the script is run, it creates a new formula column for each column that is specified in the launch window. Each new column contains the original column's value if the value is within the outlier limits and is set to missing otherwise. The new columns are prefixed or suffixed by a user specified name to distinguish them from the original columns. By default, the suffix is set to "Culled".

Rescan Rescans the data after outlier actions have been taken.

Note: Press Ctrl and click **Rescan** to rescan across all command groups.

Close Closes the Robust Fit Outliers panel.

Note: Press Ctrl and click **Close** to close all command windows.

Robust PCA Outliers

You can use the Robust PCA Outliers utility to quickly identify outlier cells in correlated multivariate data. This method is useful because many other multivariate approaches identify only the outlier rows. Before the method is applied to the data, the columns are first centered (optional) and scaled. The scaling factor is defined as follows:

max [Q(.75) - Q(.50), Q(.50) - Q(.25)] / [normalQuantile(0.75)]

where

Q(p) is the p^{th} quantile

Note: If Q(75) or Q(25) are equal to the median, then more extreme quantiles are used until there is a non-zero range.

After the data are centered and scaled, the Robust PCA Outliers utility performs a sequence of singular value decompositions and thresholding steps to decompose the data matrix. The data are decomposed into a low-rank matrix and a sparse matrix of residuals. The thresholding is done so that the residuals are either very large or outliers or very close to zero for non-outliers. The algorithm determines a matrix rank appropriate to capture the systematic variation without the outliers or small noise. Outliers that are not in the low-rank space are detected based on their residuals. See Candes et al (2009) and Lin et al (2013). If there are missing values, they are initially replaced with zeros after the centering and scaling steps. Then, after each singular value decomposition (SVD) iteration, the missing values are updated by their predicted values from the SVD.

Robust PCA Outliers Report

When you select Robust PCA Outliers from the list of commands, you must specify a value for Lambda and select if the data should be centered. If you Shift+Click the Robust PCA Outliers button, the following options are also available:

Lambda Specifies a value that determines the sparsity of the matrix of residuals. For larger values of Lambda, the matrix of residuals is more sparse. For a data table with n training rows and p columns, the default value of Lambda is defined as follows:

$$\lambda = \frac{2}{\sqrt{\max(n, p)}}$$

Max Iterations Specifies the maximum number of SVD iterations.

Converge Criterion Determines when to stop the algorithm.

Outlier Threshold Specifies the outlier threshold that determines which outliers are shown in the Cell Large Residuals table. An observation is shown if the scaled residuals is larger than the following:

$$\min[0.99 \times \max\{\text{abs}(\text{residuals})\}, \text{Outlier Threshold}]$$

By default, the Outlier Threshold is 2.

Center Determines if the data are centered before the Robust PCA Outlier algorithm is performed.

Note: If the number of rows is less than or equal to 10, the data are not centered.

Scale Determines if the data are scaled before the Robust PCA Outlier algorithm is performed.

Note: If the number of rows is less than or equal to 10, the data are not scaled.

The Robust PCA Outliers report contains a table with information about the method. This table includes the rank of the low-rank matrix, the number of SVD iterations, the convergence criterion, the value of Lambda, and the number of imputed missing values. The report also contains the following tables and reports:

Cell Large Residuals A table that shows the largest outlier cells. The number of cells shown is determined by the Outlier Threshold. The table contains the column name and row number of the cell, the residual value, and the scaled residual value.

Tip: To color specific outlier cells in the data table, select rows in the Cell Large Residuals table and click **Colorize**.

Row Root Mean Square A table that shows the root mean square value for each row in the data table. The root mean square is calculated using the scaled residuals.

Tip: If you select a row in the Row Root Mean Square table, the corresponding row is selected in the data table.

Column Root Mean Square A table that shows the root mean square value for each column specified in the launch window. The root mean square is calculated using the scaled residuals.

Tip: If you select a row in the Column Root Mean Square table and click **Select Columns**, the corresponding column is selected in the data table.

Snapshot A graphical representation of the outlier cells in the data table. The outlier cells are colored in red.

Residuals The matrix of residuals from the matrix decomposition. A cell is colored if the absolute value of the scaled residual is greater than the following:

$$\min[0.99 \times \max\{\text{abs}(\text{residuals})\}, \text{Outlier Threshold}]$$

Low Rank Approximation The matrix of scaled residuals from the matrix decomposition.

Singular Values The vector of singular values from the SVD.

Robust PCA Outliers Options

There are buttons at the bottom of the Robust PCA Outliers report that provide options to save different parts of the report.

Close Closes the Robust PCA Outliers report.

Save Large Outliers Saves the information in the Cell Large Residuals table to a new data table.

Save Cleaned Opens a window that provides several techniques to clean the outliers based on thresholds and save new columns to the data table.

Trim Trims outlier cells if the corresponding absolute scaled residual is greater than the specified threshold. By default, the threshold is 10. Select Color to color the outlier cells red. The trimmed cells are set to the value of the unscaled threshold.

Impute Sets outlier cells to the value of the low rank approximation if the corresponding absolute scaled residual is greater than the specified threshold. By default, the threshold is 100. Select Color to color these cells green.

Make Missing Sets outlier cells to missing if the corresponding absolute scaled residual is greater than the specified threshold. By default, the threshold is 1000. Select Color to color these cells blue.

Color imputed from missing If selected, colors cells that originally had missing values and were imputed.

Save Residuals Saves the residuals to new columns in the original data table.

Save Scaled Residuals Saves the scaled residuals to new columns in the original data table.

Save Low Rank Approx Saves the low-rank approximation to new columns in the original data table.

K Nearest Neighbor Outliers

Use K Nearest Neighbor Outliers to identify an outlier based on distance to its nearest neighbor. For each value of k, the K Nearest Neighbor Outliers utility displays a plot of the Euclidean distance from each point to its kth nearest neighbor. You specify the largest value of k, denoted as K. Plots are provided for $k = 1,2,3,5,...,K$, using the Fibonacci sequence to avoid displaying too many plots.

Before the nearest neighbors are calculated, the columns are centered and scaled. The scaling factor is as follows:

max [Q(.75) - Q(.50), Q(.50) - Q(.25)] / [normalQuantile(0.75)]

where

Q(p) is the p^{th} quantile

Note: If Q(75) or Q(25) are equal to the median, then more extreme quantiles are used until there is a non-zero range.

This approach is sensitive to the specified value of *k*. A small value of *k* can miss identifying points as outliers and a large value of *k* can falsely classify points as outliers:

- Suppose that the specified *K* is small, so that you are studying only a few neighbors. If there is a cluster of more than *K* points that is far from the rest of the points, then the points within the cluster have small distances to their nearest neighbors. You might be unable to detect the cluster of outliers.
- Suppose that the specified *K* is large, so that you are studying a large number of neighbors. If there are clusters with fewer than *K* data points, then the points within these clusters can appear to be outliers. You might overlook the fact that the points form a cluster, interpreting the individual cluster members as outliers instead.

K Nearest Neighbor Outliers Report

When you select K Nearest Neighbor Outliers from the list of commands, you must specify the value of *K* to use as an upper bound for the farthest neighbor to be considered. You must also specify whether missing values should be imputed. Notice that *K* is set to 8 and Impute Missing is selected by default.

The report shows plots for select values of *k* up to the value *K*. The value of *k* for each plot is displayed in its vertical axis label. It is of the form Distance to Neighbor k = <a>, where a is an integer denoting the a^{th} closest neighbor. Each plot shows the distance from the point in the i^{th} row to its a^{th} nearest neighbor. The points that have large distances from their neighbors, across multiple values of *k*, are likely to be outliers.

The buttons above the plots do the following:

Exclude Selected Rows Excludes rows corresponding to selected points from further analysis. The rows are assigned the Excluded row state in the data table. You are asked if you want to rerun or close the K Nearest Neighbors report. Rerunning the analysis identifies new nearest neighbors. The plots are updated and the excluded points are not shown.

Scatterplot Matrix Opens a separate window containing a scatterplot matrix for all columns in the analysis. You can explore potential outliers by selecting them in the K Nearest Neighbors plots and viewing them in the scatterplot matrix.

Save NN Distances Saves the distances from each row to its n^{th} nearest neighbor as new columns in the data table.

Close Closes the K Nearest Neighbors report.

Largest Outliers

The report also includes a Largest Outliers table. This table contains the 20 observations with the largest distances from their K^{th} nearest neighbor. The table has the following columns:

Row The row number of the observation.

Distance The distance from the observation in the specified row and its K^{th} nearest neighbor. The table is sorted by this column in descending order.

Nearest Neighbors Lists the row numbers for the k nearest neighbors. The first row number is the closest nearest neighbor. The last row number is the K^{th} nearest neighbor and the distance between this observation and the specified row is found in the Distance column.

Col<n> Specifies the column name for the corresponding RMS value.

RMS<n> Calculates the root mean squared differences across the k nearest neighbors for each column. The five largest RMS values are displayed in order, where RMS1 is the maximum RMS value. The p^{th} RMS value is calculated as follows:

$$\mathrm{RMS}_p = \sqrt{\frac{\sum_{k=1}^{K} (D_{p,i} - D_{p,i_k})^2}{K}}$$

where

D_p is the p^{th} column

$D_{p,i}$ is the value of the p^{th} column for row i

$D_{p,ik}$ is the value of the p^{th} column for the k^{th} nearest neighbor of row i

Note: The number of Col and RMS columns shown in the Largest Outliers table is the minimum of the number of columns specified in the launch and the number five.

Explore Outliers Utility Options

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Examples of the Explore Outliers Utility

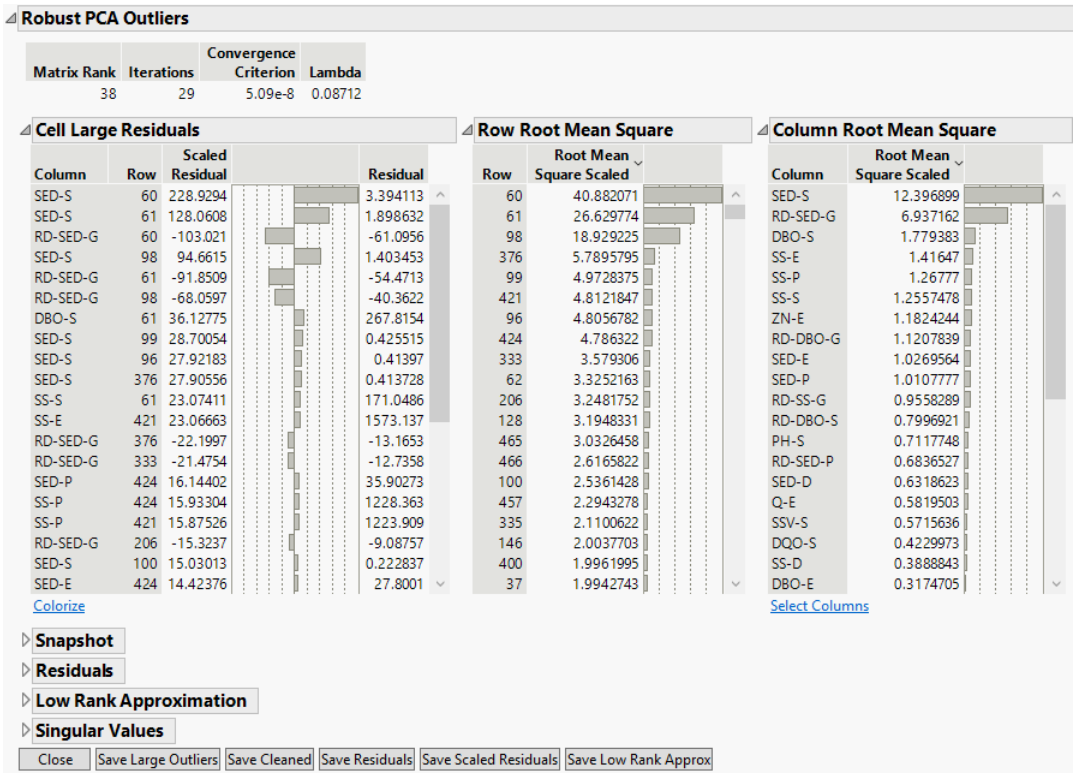
Robust PCA Outliers Example

The Water Treatment.jmp data set contains daily measurement values of 38 sensors in an urban waste water treatment plant. You are interested in exploring these data for potential outliers. Potential outliers could include sensor failures, storms, and other situations.

1. Select **Help > Sample Data Library** and open Water Treatment.jmp.
2. Select **Analyze > Screening > Explore Outliers**.
3. Select the Sensor Measurements column group and click **Y, Columns**.
4. Click **OK**.
5. Shift+Click **Robust PCA Outliers**.
6. Enter 10 next to Outlier Threshold.

   Use the default values for the other options.
7. Click **OK**.

Figure 21.8 Robust PCA Outliers Report



The Cell Large Residuals table shows the cells that have a scaled residual greater than 10 or less than -10. There are several cells with large scaled residuals from the SED-S column. If you look at the Column Root Mean Square table, it shows that the SED-S column does in fact have the largest scaled Root Mean Square value. The row with the largest scaled Root Mean Square value is row 60.

8. Click **Save Cleaned** at the bottom of the report.

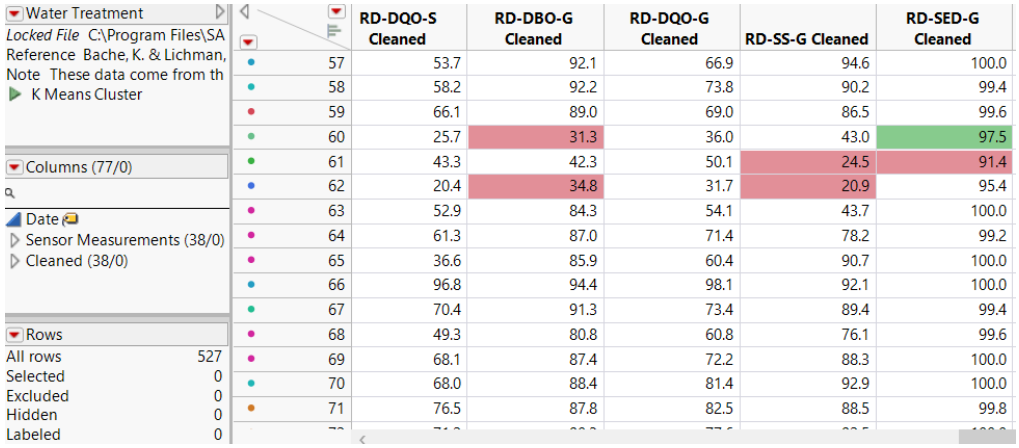
   This option enables you to adjust cells with large outliers by trimming the values, imputing new values, or setting the cells to missing. The Save Cleaned Columns report provides these options.

9. Select Impute. Trim is already selected by default.

10. Click **OK**.

    There are 38 new Cleaned columns added to the Water Treatment.jmp data table. Since you know row 60 had the largest scaled Root Mean Square value, scroll to this row to view some of the cells that were trimmed and imputed. You can now use the cleaned columns in any type of analysis.

Figure 21.9 Cleaned Columns from Robust PCA Outliers



	RD-DQO-S Cleaned	RD-DBO-G Cleaned	RD-DQO-G Cleaned	RD-SS-G Cleaned	RD-SED-G Cleaned
57	53.7	92.1	66.9	94.6	100.0
58	58.2	92.2	73.8	90.2	99.4
59	66.1	89.0	69.0	86.5	99.6
60	25.7	31.3	36.0	43.0	97.5
61	43.3	42.3	50.1	24.5	91.4
62	20.4	34.8	31.7	20.9	95.4
63	52.9	84.3	54.1	43.7	100.0
64	61.3	87.0	71.4	78.2	99.2
65	36.6	85.9	60.4	90.7	100.0
66	96.8	94.4	98.1	92.1	100.0
67	70.4	91.3	73.4	89.4	99.4
68	49.3	80.8	60.8	76.1	99.6
69	68.1	87.4	72.2	88.3	100.0
70	68.0	88.4	81.4	92.9	100.0
71	76.5	87.8	82.5	88.5	99.8

Explore Missing Values Utility

The presence of missing values in a data set can affect the conclusions made using the data. If, for example, several healthy participants dropped out of a longitudinal study and their data continued on as missing, the results of the study can be biased toward those unhealthy individuals who remained. Missing data values must not only be identified, they must also be understood before further analysis can be conducted.

The Explore Missing Values utility provides several ways to identify and understand the missing values in your data. It also provides methods for conducting multivariate normal imputation for missing values. These imputation methods assume that data are *missing at random*, which means that any differences between missing and non-missing data cannot be explained by the values of the other variables in the study. If you suspect that missing values are *not* missing at random, then consider using the Informative Missing procedure, which is available in a number of platforms. See *Fitting Linear Models*.

Caution: Be careful when analyzing data after imputing missing values, as the results have the potential to be biased.

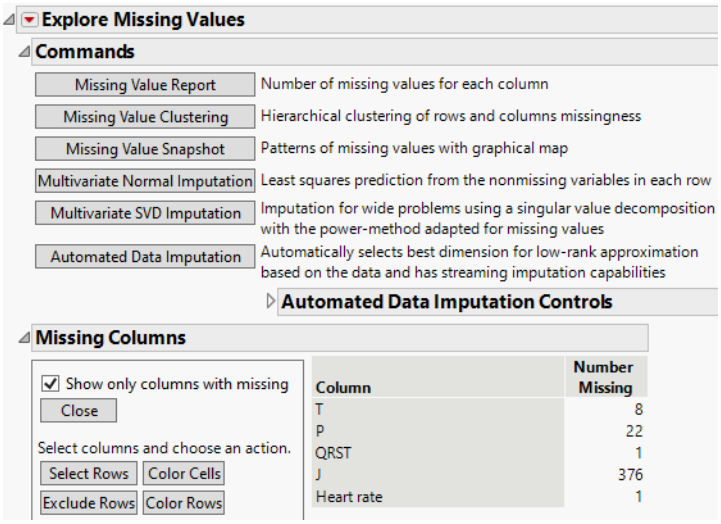
Example of the Explore Missing Values Utility

The Arrhythmia.jmp sample data table contains information from 452 patient electrocardiograms (ECGs). The data was originally collected to classify different patterns of ECGs as cardiac arrhythmia. However, there are missing values in this data table. You are primarily interested in exploring these missing values and imputing them when necessary. Since you can conduct missing value imputation only for columns that have a continuous modeling type, you conduct your analysis in two stages.

Examine Missing Values

1. Select **Help > Sample Data Library** and open Arrhythmia.jmp.
2. Select **Analyze > Screening > Explore Missing Values**.
3. Select all columns (280 in all) and click **Y, Columns**.
4. Click **OK**. Select the **Show only columns with missing** check box.

Figure 21.10 Missing Value Report



The Missing Columns report shown in Figure 21.10 indicates that only five columns have missing data. Out of a total of 452 rows, Column J has 376 missing values. Because it is largely missing, imputed values might not result in a meaningful analysis. For such data, you could explore a model using the Informative Missing option for Column J in a platform that supports the Informative Missing option.

Note that the two Imputation options, Multivariate Imputation and Multivariate SVD Imputation, are not shown. A message indicates that imputation is disabled because some columns included in the analysis were categorical. The data table contains several columns that are numeric, but have a nominal modeling type. These cannot be used for imputation.

Impute Missing Values

The five columns that have missing values are continuous. You proceed to impute values for the four columns other than Column J using multivariate imputation for the continuous columns in your data table. By doing so, you tacitly assume that the probabilities that values are missing depend only on the values of the continuous variables and not on the values of excluded nominal variables. To conduct this new analysis, you need to launch the Explore Missing Values utility again.

1. Select **Analyze > Screening > Explore Missing Values**.
2. In the launch window, click the red triangle next to **280 Columns**.

   Use the columns filter menu to view only the columns with a Continuous modeling type in the Select Columns list.
3. Select **Modeling Type** > **Uncheck All**.

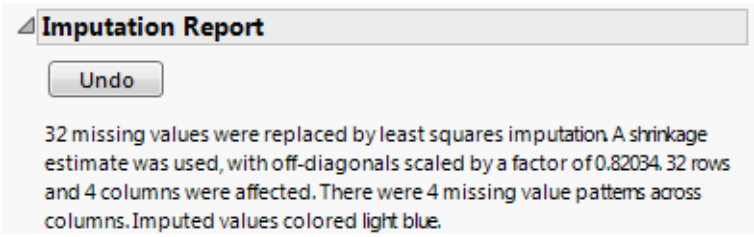
   This removes all columns from the Select Columns list.
4. Select **Modeling Type** > **Continuous**.

   The Select Columns list now contains only the 207 columns that are Continuous.
5. Select all 207 columns. Then Ctrl-click the J column (to deselect it) and click **Y, Columns**.
6. Click **OK**.
7. Click **Multivariate Normal Imputation.**

   A window appears and asks whether you want to use a Shrinkage estimator for covariances.
8. Click **Yes Shrinkage**.

   A JMP Alert appears, informing you that you should use the **Save As** command to preserve your original data.
9. Click **OK**.

Figure 21.11 Imputation Report



The Imputation Report indicates how many missing values were imputed and the specific imputation details. No missing data remain in the four columns that had missing values.

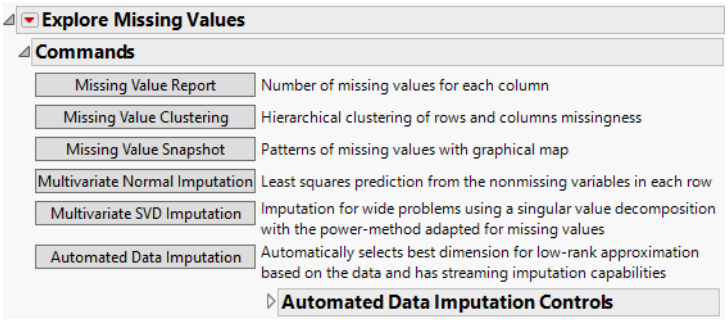
Launch the Explore Missing Values Utility

Launch the Explore Missing Values modeling utility by selecting **Analyze > Screening > Explore Missing Values**. Enter the columns of interest into the Y, Columns list. You can also specify a **By** variable.

Note: You can enter only columns with a Numeric modeling type in the Explore Missing Values utility.

The Missing Value Report

Figure 21.12 Missing Value Report for Continuous Variables in Arrhythmia.jmp.



After you click OK in the launch window, the report opens to show a Commands outline and a Missing Columns report. The commands are the following:

- "Missing Value Report"
- "Missing Value Clustering"
- "Missing Value Snapshot"
- "Multivariate Normal Imputation" (Not available if you entered a Numeric column with a Nominal or Ordinal modeling type in the launch window.)
- "Multivariate SVD Imputation" (Not available if you entered a Numeric column with a Nominal or Ordinal modeling type in the launch window.)
- JMP PRO "Automated Data Imputation" (Not available if you entered a Numeric column with a Nominal or Ordinal modeling type in the launch window.)

Tip: To run a missing value command across all levels of a **By** variable, press Ctrl and click the desired command button.

Missing Value Report

The Missing Value Report opens the Missing Columns report, which lists the name of each column and the number of missing values in that column.

Show only columns with missing Removes columns from the list that do not have missing values.

Close Closes the Missing Columns report.

Select Rows Selects the rows in the data table that contain missing values for the column(s) that you select in the Missing Columns report.

Exclude Rows Applies the excluded row state for rows in the data table that contain missing values for the column(s) that you select in the Missing Columns report.

Color Cells Colors the cells in the data table that contain missing values for the column(s) that you select in the Missing Columns report.

Color Rows Colors the rows in the data table that contain missing values for the column(s) that you select in the Missing Columns report.

Missing Value Clustering

Missing Value Clustering provides a hierarchical clustering analysis of the missing data.

- The dendrogram to the right of the plot shows clusters of missing data pattern rows. These are the rows that you would obtain by using Tables > Missing Data Pattern.
- The dendrogram beneath the plot shows clusters of variables.

Use this report to determine whether certain groups of columns tend to have similar patterns of missing values.

The rows of the plot are defined by the missing data patterns; there is a row for each pattern. The columns correspond to the variables. Each red cell indicates a group of missing values for the column listed beneath the plot. Hover over a cell to see the list of values represented. Click in the plot to select missing data pattern rows. Vertical bars are displayed to indicate the selected patterns.

Missing Value Snapshot

The Missing Value Snapshot shows a cell plot for the missing values. The columns represent the variables. Black cells indicate a missing value. This plot is especially useful in understanding missingness for longitudinal data, where subjects can withdraw from a study before the end of the data collection period.

Multivariate Normal Imputation

The Multivariate Normal Imputation utility imputes missing values based on the multivariate normal distribution. The procedure requires that all variables have a Continuous modeling type. The algorithm uses least squares imputation. The covariance matrix is constructed using pairwise covariances. The diagonal entries (variances) are computed using all nonmissing values for each variable. The off-diagonal entries for any two variables are computed using all observations that are nonmissing for both variables. In cases where the covariance matrix is singular, the algorithm uses minimum norm least squares imputation based on the Moore-Penrose pseudo-inverse.

Multivariate Normal Imputation allows the option to use a shrinkage estimator for the covariances. The use of shrinkage estimators is a way of improving the estimation of the covariance matrix. For more information about shrinkage estimators, see Schäfer and Strimmer (2005).

Note: If a validation column is specified, the covariance matrices are computed using observations from the Training set.

Multivariate Normal Imputation Report

The imputation report explains the results of the multivariate imputation process. Results include the following:

- Method of imputation (either least squares or minimum-norm least squares)
- How many values were replaced
- Shrinkage estimator on/off
- Factor by which the off-diagonals were scaled
- How many rows and columns were affected
- How many different missing value patterns there were

Once the imputation is complete, the cells corresponding to imputed values in the data table are colored in light blue. If the Missing Columns report is open, it is updated to show no missing values.

Click **Undo** to undo the imputation and replace the imputed data with missing values.

Multivariate SVD Imputation

The Multivariate SVD Imputation utility imputes missing values using the singular value decomposition (SVD). This utility is useful for data with hundreds or thousands of variables. Because SVD calculations do not require calculation of a covariance matrix, the SVD method is recommended for wide problems that contain large numbers of variables. The procedure requires that all variables have a Continuous modeling type.

The singular value decomposition represents a matrix of observations **X** as **X** = **UDV**′, where **U** and **V** are orthogonal matrices and **D** is a diagonal matrix.

The SVD algorithm used by default in the Multivariate SVD Imputation utility is the sparse Lanczos method, also known as the *implicitly restarted Lanczos bidiagonalization method* (IRLBA). See Baglama and Reichel (2005). The algorithm does the following:

1. Each missing value is replaced with its column's mean.
2. An SVD decomposition is performed on the matrix of observations, **X**.
3. Each cell that had a missing value is replaced by the corresponding element of the **UDV**′ matrix obtained from the SVD decomposition.
4. Steps 2 and 3 are repeated until the SVD converges to the matrix **X**.

Imputation Method Window

When you click Multivariate SVD Imputation, the Imputation Method window shows the recommended settings.

Number of Singular Vectors Number of singular vectors that are computed and used in the imputation.

> **Note:** It is important not to specify too many singular vectors, otherwise the SVD and the imputations do not change from iteration to iteration.

Maximum Iterations The number of iterations used in imputing the missing values.

Show Iteration Log Opens a Details report that shows the number of iterations and gives details about the criteria.

For large problems, a progress bar shows how many dimensions the SVD has completed. You can stop the imputation and use that number of dimensions at any time.

Multivariate SVD Imputation Report

The imputation report explains the results of the multiple imputation process.

- Method of imputation
- How many values were replaced
- How many rows and columns were affected

Once the imputation is complete, the Missing Columns report is automatically shown indicating no missing values in the columns that were imputed. Imputed values are displayed in light blue.

Click **Undo** to undo the imputation and replace the imputed data with missing values.

JMP PRO Automated Data Imputation

The Automated Data Imputation (ADI) utility imputes missing values using a low-rank matrix approximation method, also known as matrix completion. Once trained, the ADI model is capable of performing missing data imputations for streaming data through scoring formulas. Streaming data are added rows of observations that become available over time and were not used for tuning or validating the imputed model. This utility is flexible, robust, and automated to select the best dimension for the low-rank approximation. These features enable ADI to work well for many different types of data sets.

A low-rank approximation of a matrix is of the form $\mathbf{X} = \mathbf{UDV}'$ and can be viewed as an extension of singular value decomposition (SVD). ADI uses the Soft-Impute method as the imputation model and is designed such that the data determines the rank of the low-rank approximation.

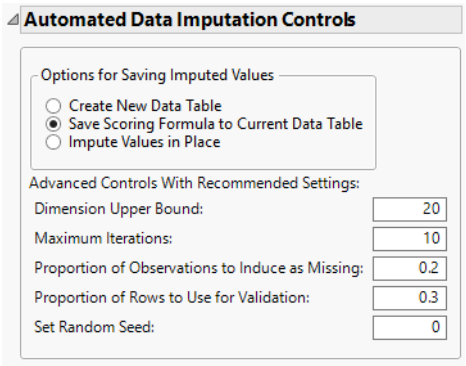
The ADI algorithm performs the following steps:

1. The data are partitioned into training and validation sets.
2. Each set is centered and scaled using the observed values from the training set.
3. For each partitioned data set, additional missing values are added within each column and are referred to as induced missing (IM) values.
4. The imputation model is fit on the training data set along a solution path of tuning parameters. The IM values are used to determine the best value for the tuning parameter.
5. Additional rank reduction is performed using the training data set by de-biasing the results from the chosen imputation model in step 4.
6. Final rank reduction is performed to calibrate the model for streaming data and to prevent overfitting. This is done by fitting the imputation model on the validation set, using the rank determined in step 5 as an upper bound.

Automated Data Imputation Controls

The ADI utility contains options for saving the imputed values and advanced controls.

Figure 21.13 ADI Controls



Options for Saving Imputed Values The following three options for saving the imputed values for the ADI method are available:

Create New Data Table Creates a new data table that has the same dimensions as the original data table. In the new data table, the columns selected in the launch window contain the imputed values.

Save Scoring Formula to Current Data Table Saves a column group, named Imputed_, to the current data table that contains the imputed columns specified in the launch window. A hidden column, ADI Impute Column, is also added to the current data table that contains the imputed vectors and the scoring formula used in the data imputation. The column formulas automatically update if any additional rows are added to the data table, enabling missing data imputation for streaming data. This is the default option.

Impute Values in Place Imputes the missing values in the current data table. The imputed values are displayed in light blue.

Advanced Controls Contains the following advanced controls, which default to recommended settings based on the data:

Dimension Upper Bound Determines the maximum rank allowed in the low-rank approximation. This is determined by the dimension of the matrix formed by the chosen columns.

Maximum Iterations Determines the number of values that are iterated over to determine the tuning parameter for the imputation model. The default is 10.

Proportion of Observations to Induce as Missing Determines the proportion of IM values that are added to the training and validation sets. The default proportion for each set is 0.2.

Proportion of Rows to Use for Validation Determines the proportion of rows to use in the training and validation sets. The default proportion for the validation set is 0.3.

Set Random Seed Determines the random seed for ADI. Use this option to obtain reproducible results.

Explore Missing Values Utility Options

ADI Loading Matrix (Available only for the ADI utility.) Shows or hides a report that contains the columns that correspond to the factor loading for each component.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Explore Patterns Utility

Data can be messy, so it is sometimes hard to determine the validity of your data by visual inspection alone. The Explore Patterns utility is a tool for detecting unusual or unexpected patterns in your data. Although some patterns occur by chance or some innocent mechanism, they might indicate data that has been falsified or tampered with. This utility is most useful when the values are precise so that matches and patterns are less likely to happen by coincidence. The Explore Patterns utility is also designed to handle large data tables efficiently.

The Explore Patterns utility looks for duplicate values and duplicate sequences of values. These types of patterns can indicate data tampering strategies such as pasting the same value many times, pasting the same value across a range of cells, or copying a range of data and pasting it somewhere else. The utility also analyzes properties of the formatted values to determine whether the data are actual measurements or might have been generated by a random number generator. There are also options to examine distribution properties and detect linear relationships between columns across groups of rows.

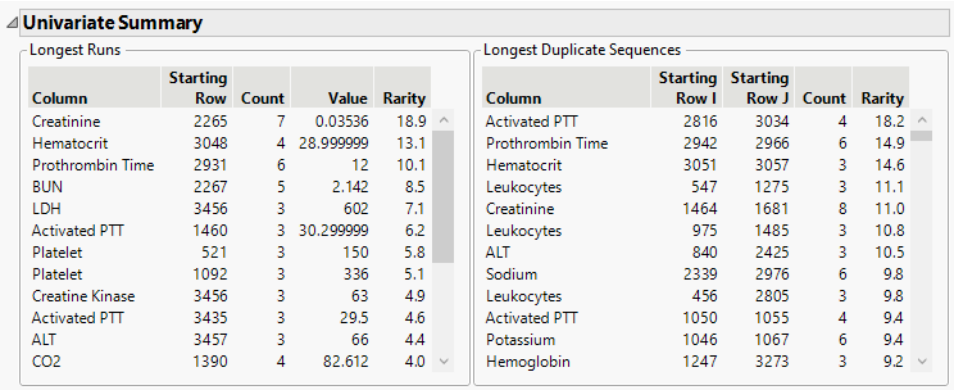
Note: The Explore Patterns utility ignores excluded rows, but the results are not necessarily equivalent to if you had deleted the row. For runs of rows, such as the Longest Runs and Longest Duplicated Sequences options, an excluded row stops the run.

Example of the Explore Patterns Utility

The Nicardipine Lab Patterns.jmp sample data table is adapted from a study of the drug Nicardipine. The data table contains 27 columns of numeric laboratory results. Use the Explore Patterns Utility to determine whether there are any unexpected patterns in the laboratory results.

1. Select **Help > Sample Data Library** and open the Nicardipine Lab Patterns.jmp sample data table.
2. Select **Analyze > Screening > Explore Patterns**.
3. Select the Laboratory Results column group and click **Y, Columns**.
4. Click **OK**.
5. Click the gray disclosure icon next to Univariate Summary to open the Univariate Summary report.

Figure 21.14 Explore Patterns Univariate Summary Report



Univariate Summary

Longest Runs

Column	Starting Row	Count	Value	Rarity
Creatinine	2265	7	0.03536	18.9
Hematocrit	3048	4	28.999999	13.1
Prothrombin Time	2931	6	12	10.1
BUN	2267	5	2.142	8.5
LDH	3456	3	602	7.1
Activated PTT	1460	3	30.299999	6.2
Platelet	521	3	150	5.8
Platelet	1092	3	336	5.1
Creatine Kinase	3456	3	63	4.9
Activated PTT	3435	3	29.5	4.6
ALT	3457	3	66	4.4
CO2	1390	4	82.612	4.0

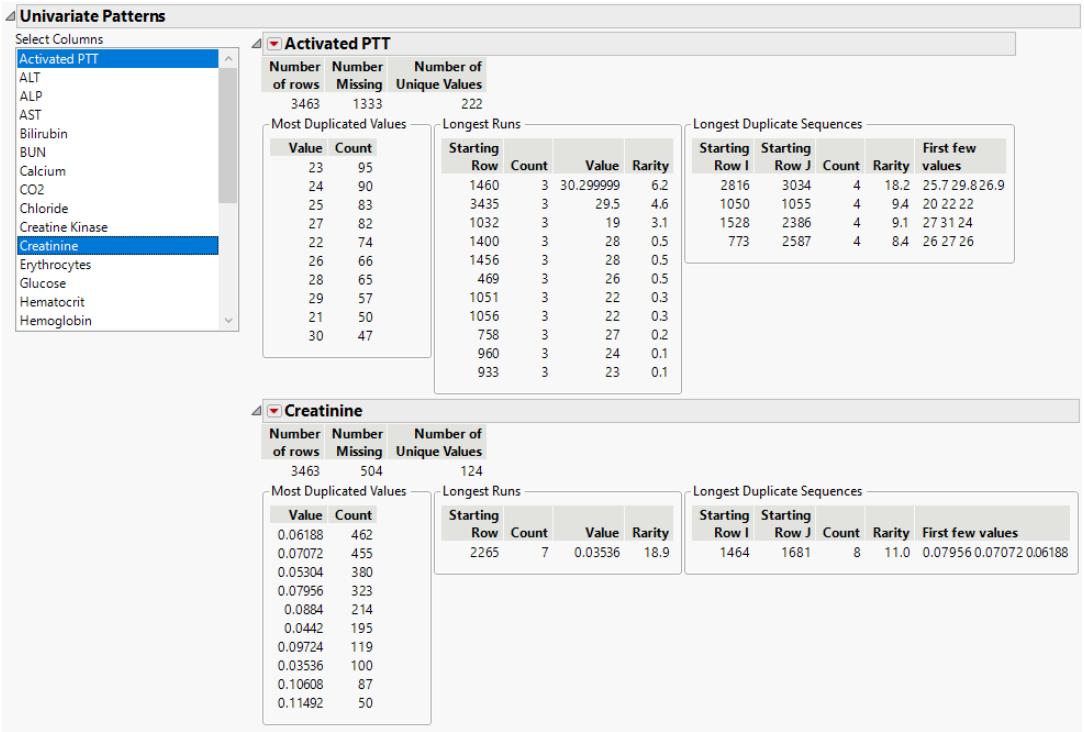
Longest Duplicate Sequences

Column	Starting Row I	Starting Row J	Count	Rarity
Activated PTT	2816	3034	4	18.2
Prothrombin Time	2942	2966	6	14.9
Hematocrit	3051	3057	3	14.6
Leukocytes	547	1275	3	11.1
Creatinine	1464	1681	8	11.0
Leukocytes	975	1485	3	10.8
ALT	840	2425	3	10.5
Sodium	2339	2976	6	9.8
Leukocytes	456	2805	3	9.8
Activated PTT	1050	1055	4	9.4
Potassium	1046	1067	6	9.4
Hemoglobin	1247	3273	3	9.2

The Univariate Summary report summarizes the longest runs and longest duplicate sequences found across all of the columns. Each table is sorted by Rarity, a calculation of how likely the run or sequence is to occur by chance. The column with the highest rarity score for longest runs is Creatinine, whereas the column with the highest rarity score for longest duplicate sequences is Activated PTT. You will further examine these columns in the Univariate Patterns Report.

6. In the Univariate Patterns report, the report for the first column, Activated PTT, is shown initially. Press Ctrl and select Creatinine from the Select Columns list.

Figure 21.15 Explore Patterns Univariate Patterns Report

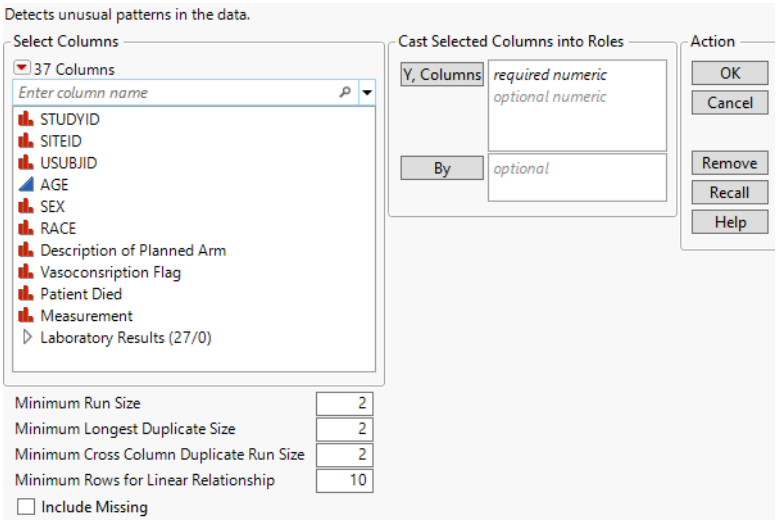


As you select column names from the Select Columns list, the corresponding column reports are shown. The Activated PTT report shows that this column contains several three count runs and four count sequences. The Creatinine report shows that although it is the column with the highest rarity score for Longest Runs, there is only one run in the column. This column also has one eight count sequence.

Launch the Explore Patterns Utility

Launch the Explore Patterns utility by selecting **Analyze > Screening > Explore Patterns**.

Figure 21.16 Explore Patterns Utility Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns Assigns the columns to analyze. The variables must be numeric and continuous.

By Assigns one or more columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Minimum Run Size Specifies the minimum number of values in a row for a sequence to be reported as a run.

Minimum Longest Duplicate Size Specifies the minimum length of a sequence for it to be reported as a duplicate sequence within a column.

Minimum Cross Column Duplicate Run Size Specifies the minimum length of a sequence for it to be reported as a duplicate sequence across columns.

Minimum Rows for Linear Relationship Specifies the minimum number of rows used for detecting linear relationships between columns across groups of rows.

Note: The minimum number of rows must be greater than or equal to three.

Include Missing Specifies that missing values be included in pattern detection.

Explore Patterns Report

The initial Explore Patterns report contains a Control Panel, Univariate Summary report, and Univariate Patterns report. Many of the tables in these reports contain rows that can be selected. If a row is selected and you right-click the first column, a menu appears that enables you to select or colorize the related rows and columns in the data table.

Control Panel

The control panel contains options that enable you to select which reports appear in the report window. These are the same options that are in the Explore Patterns red triangle menu. See "Explore Patterns Utility Options". There is also a section of the control panel to specify report settings. These are the same settings that are found in the launch window. See "Launch the Explore Patterns Utility". If you change the settings in the control panel, the reports are automatically updated.

Univariate Summary

The Univariate Summary report combines results from several patterns reports across columns. By default, there are summary tables for Longest Runs and Longest Duplicate Sequences. Longest Runs and Longest Duplicate Sequences with rarity ≥ 2 are shown in the tables.

If the Fraction Length option is selected a Fraction Length summary table is also shown. This table shows the proportion of values in each column that have a fraction length greater than or equal to 15. Columns that do not contain any values that have fraction lengths greater than or equal to 15 are not shown in the Fraction Length summary table.

If at least one of the Y columns has a Spec Limit column property and the Distribution wrt Spec Limits option is selected, a Spec Limits Distrib summary table is also shown. This table shows the columns that have a statistically significant number of observations outside of the specification limits.

For more information about the Univariate Summary report contents, see "Explore Patterns Utility Options".

Univariate Patterns

The Univariate Patterns report contains a Select Columns list and one or more column-specific patterns reports. The column-specific reports that are displayed are determined by the Select Columns list. To examine each column's report, select the first column and then successively use the down arrow key to advance to the next column. Alternatively, you can click a column name in this list to display the patterns report for the selected column. If you select more than one column name by using either Ctrl or Shift, a patterns report for each of the selected columns is displayed. If you deselect one or more columns from the Select Columns list, the reports for the corresponding columns are removed.

Each individual patterns report displays a summary table that contains the number of rows, number of missing observations, and number of unique values for the given column. By default, there are also tables for Most Duplicated Values, Longest Runs, and Longest Duplicate Sequences. If the Formatted Widths, Fraction Lengths, or Leading and Trailing Digits options are selected from either the control panel or the Explore Patterns red triangle menu, a corresponding table is added to each individual patterns report.

If you right-click on a row in any of the tables in the Univariate Patterns report, a menu of right-click options appears.

Select Rows and Column Selects the column and rows in the original data table that correspond to the selected row in a specific table in the Univariate Patterns report. For example, if you right-click the second row of the Leading Digits table for a specific column, the rows in that column that have a leading digit of 2 are selected in the original data table.

Colorize Cells Colors the cells in the original data table that correspond to the selected row in a specific table in the Univariate Patterns report. The rows of the colored cells are also selected.

There are red triangle options for the column-specific patterns report. See "Univariate Patterns Report Options".

Explore Patterns Utility Options

The control panel and Explore Patterns red triangle menu contain options for summary and pattern tables.

Most Duplicated Values Shows or hides a table of the values that appear most frequently in the data. Shows up to ten numbers for each column.

Longest Runs Shows or hides a table of the longest run of the same value. The table contains the following information:

Starting Row The row number that corresponds to the start of the run.

Count The length of the run.

Value The value of the entries in the run.

Rarity A measurement of how rare it would be for the run to occur by chance. A high rarity indicates that it is not likely that the run occurred by chance. See "Rarity in Longest Runs".

Longest Duplicated Sequences Shows or hides a table of the longest sequence of values that appears more than once in the same column. Sequences of repeated values are also considered. If a sequence is duplicated more than once, each instance appears in the table. The table contains the following information:

Starting Row I The row number that corresponds to the start of the first sequence.

Starting Row J The row number that corresponds to the start of the duplicate sequence.

Count The length of the sequence.

Rarity A measurement of how rare it would be for the duplicate sequence to occur by chance. A high rarity indicates that it is not likely that the duplicate sequence occurred by chance. See "Rarity in Longest Sequences".

First few values The first few values of the duplicate sequence. Shows up to three values for each sequence.

Formatted Widths Shows or hides a table of overall widths and decimal widths of the formatted values. The table contains the following information:

Width The width size.

Overall Count The number of observations in the column whose overall width equals the specified width.

Decimal Count The number of observations in the column whose decimal width equals the specified width.

Fraction Length Shows or hides a table of the continued fraction lengths for the values. A continued fraction is a representation of a number as a sequence of continually divided terms. The fraction length can indicate how rational a number is. Lengths of 15 or more indicate that the number is an irrational number, such as from a root function, a function involving an irrational number, or from a random number generator (Benford, 1938).

Note: In Explore Patterns, the maximum fraction length is 15. Numbers with fraction lengths of 15 or more are all reported as 15.

Length The length of the continued fraction.

Count The number of observations in the column whose continued fraction length equals the specified continued fraction length.

Leading and Trailing Digits Shows or hides a table of counts for each leading and trailing digit, 1 through 9. For the leading digits, there are also columns for the expected rate and the minimum and maximum values that contain the corresponding leading digit. The expected rate is based on Benford's Law, which states that the frequency that a number has a first digit that is equal to a is $F_a = \log_{10}(a+1/a)$. This law applies to data that have a large range of numbers that all have four or more digits.

Spec Limit Matches (Available when at least one of the columns has a Spec Limit column property.) Shows or hides a table of values that are exactly the lower specification limit, upper specification limit, or target value. If there are no exact matches, the report is not shown.

Distribution wrt Spec Limits (Available only when at least one of the columns has a Spec Limit column property.) Shows or hides a table of observed versus expected observations that are outside of the specification limits. The Chi-Square and PValue columns appear only if the number of observed observations is significantly higher than the number of expected observations.

Duplicates Across Columns Shows or hides the Duplicates Across Columns report. The report contains a table of sequences of values that appear in the same rows across more than one column. There is a note at the top of the report that states the minimum length of a sequence for it to be considered a duplicate. This number is specified in the launch window or control panel as Minimum Cross Column Duplicate Run Size. There is also a Colorize option at the top of the report. See "Colorize Duplicates Across Columns".

Colorize Duplicates Across Columns (Available only after Duplicates Across Columns has been selected.) Colors cells in the data table to correspond to the duplicate matches found in the Duplicates Across Columns report. The rows that contain the duplicates are also selected.

Linear Relationships Shows or hides the Linear relationships between variables report. The report contains a table of exact linear relationships across columns over a specified number of sequential rows. An exact linear relationship is defined as a linear relationship that has an R^2 value greater than or equal to 0.999999. There is a note at the top of the report that states the minimum number of rows considered. This number is specified in the launch window or control panel as Minimum Rows for Linear Relationships. There is also a Colorize option at the top of the report. See "Colorize Linear Relationships".

Colorize Linear Relationships (Available only after Linear Relationships has been selected.) Colors cells in the data table to correspond to the linear relationships found in the Linear relationships between variables table. The rows that contain the linear relationships are also selected.

Save Columns Contains the following save options:

Save Table of Runs Saves the longest runs for all columns to a new data table. The data table contains the same columns as the Longest Runs table in the Univariate Patterns report and is sorted by Starting Row.

Save Table of Duplicate Sequences Saves the longest duplicate sequences for all columns to a new data table. The data table contains the same columns as the Longest Duplicate Sequences table in the Univariate Patterns report and is sorted by Starting Row I.

Save Duplicates Across Columns Saves the Duplicates Across Columns table to a new data table.

Save Linear Relationships Saves the Linear relationships between values table to a new data table.

Ordering Columns Contains options for ordering. When you change the ordering, the column list in the Univariate Patterns report is reordered. The top column is automatically selected and the report for only that column is displayed.

Order by Runs Rarity Orders the column list by the rarity values of the longest runs. Higher rarity values are at the top.

Order by Sequence Rarity Orders the column list by the rarity values of the long duplicate sequences. Higher rarity values are at the top.

Order by Column Name Orders the column list in alphabetical order by column name.

Original Order Orders the column list in the original order in which the columns were specified in the launch window.

Clear Cell Colors Clears the cell colors for columns selected in the data table or for all columns if no columns are selected in the data table.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Univariate Patterns Report Options

Each Univariate Patterns red triangle menu contains the following options:

Select Most Duplicated Selects the observations in the data table that correspond to the values in the Most Duplicated Values table.

Select Longest Runs Selects the observations in the data table that correspond to the values in the Longest Runs table.

Select Longest Duplicate Sequences Selects the observations in the data table that correspond to the values in the Longest Duplicate Sequences table.

Select by Formatted Width Selects the observations in the data table that correspond to the values in the Formatted Widths table that have the user-specified overall width.

Select by Decimal Width Selects the observations in the data table that correspond to the values in the Formatted Widths that have the user-specified decimal width.

Select by Fraction Length Selects the observations in the data table that correspond to the values in the Fraction Length table that have the user-specified fraction length.

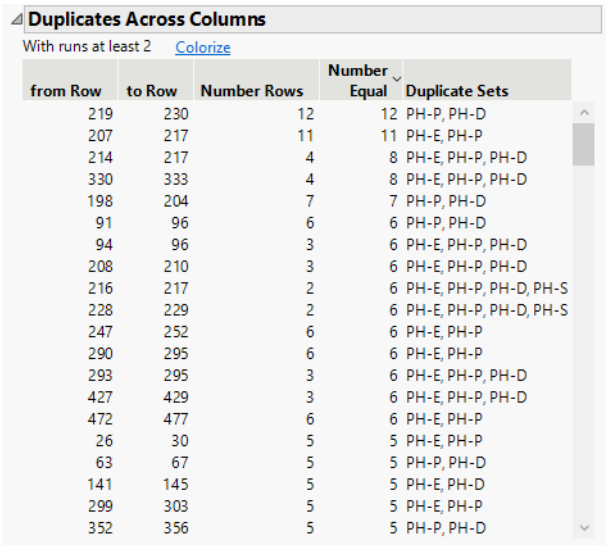
Select by Leading Digit Selects the observations in the data table that correspond to the values in the Leading and Trailing Digits table that have the user-specified leading digit.

Additional Example of the Explore Patterns Utility

The Water Treatment.jmp data table contains data from an urban waste water treatment plant. The columns are daily measurements of different sensors in the plant. Use the Explore Patterns Utility to determine whether there are any unexpected patterns in the sensor measurements.

1. Select **Help > Sample Data Library** and open the Water Treatment.jmp sample data table.
2. Select **Analyze > Screening > Explore Patterns**.
3. Select the Sensor Measurements column group and click **Y, Columns**.
4. Click **OK**.
5. In the Control Panel, under Across Columns, click the box next to Duplicates Across Columns.

Figure 21.17 Duplicates Across Columns Report



Duplicates Across Columns

With runs at least 2 Colorize

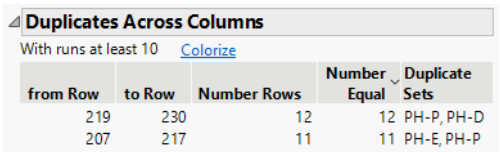
from Row	to Row	Number Rows	Number Equal	Duplicate Sets
219	230	12	12	PH-P, PH-D
207	217	11	11	PH-E, PH-P
214	217	4	8	PH-E, PH-P, PH-D
330	333	4	8	PH-E, PH-P, PH-D
198	204	7	7	PH-P, PH-D
91	96	6	6	PH-P, PH-D
94	96	3	6	PH-E, PH-P, PH-D
208	210	3	6	PH-E, PH-P, PH-D
216	217	2	6	PH-E, PH-P, PH-D, PH-S
228	229	2	6	PH-E, PH-P, PH-D, PH-S
247	252	6	6	PH-E, PH-P
290	295	6	6	PH-E, PH-P
293	295	3	6	PH-E, PH-P, PH-D
427	429	3	6	PH-E, PH-P, PH-D
472	477	6	6	PH-E, PH-P
26	30	5	5	PH-E, PH-P
63	67	5	5	PH-P, PH-D
141	145	5	5	PH-E, PH-D
299	303	5	5	PH-E, PH-P
352	356	5	5	PH-P, PH-D

The Duplicates Across Columns report shows a long list of duplicate columns. However, the minimum run size is 2 and there are many small runs in the list. You want to focus on the larger runs of at least 10.

6. In the Control Panel, under Settings, enter 10 in the box next to Minimum Cross Column Duplicate Run Size.

Tip: Click anywhere outside of the box to automatically update the report.

Figure 21.18 Duplicates Across Columns Report With Runs at Least 10



Duplicates Across Columns

With runs at least 10 Colorize

from Row	to Row	Number Rows	Number Equal	Duplicate Sets
219	230	12	12	PH-P, PH-D
207	217	11	11	PH-E, PH-P

The updated Duplicates Across Columns report shows that there are two duplicate sets with runs of at least 10. Both sets include the PH-P column.

7. Click **Colorize** in the Duplicates Across Columns Report.

Figure 21.19 Colorized Water Treatments.jmp Data Table

	PH-E	DBO-E	DQO-E	SS-E	SSV-E	SED-E	COND-E	PH-P	DBO-P	SS-P	SSV-P	SED-P	COND-P	PH-D
206	7.7	124.0	234.0	120.0	61.7	1.8	1550.0	7.9	123.0	220.0	59.0	4.0	1651.0	7.8
207	7.6	172.0	394.0	226.0	51.3	5.5	1324.0	7.6	197.0	302.0	47.0	4.0	1306.0	7.7
208	7.6	268.0	324.0	226.0	53.1	4.0	1390.0	7.6	265.0	236.0	53.4	4.5	1389.0	7.6
209	7.6	176.0	200.0	176.0	50.0	3.0	859.0	7.6	149.0	172.0	51.2	2.0	860.0	7.6
210	7.6	146.0	243.0	164.0	57.3	3.2	1095.0	7.6	193.0	208.0	55.8	3.0	1089.0	7.6
211	7.8	139.0	380.0	256.0	53.1	5.5	2040.0	7.8	201.0	340.0	47.6	8.5	2000.0	7.9
212	7.9	205.0	529.0	264.0	61.4	6.5	2750.0	7.9	220.0	324.0	60.5	6.0	2850.0	7.8
213	7.8	117.0	366.0	500.0	27.2	4.2	3230.0	7.8	135.0	620.0	32.3	7.0	3170.0	7.7
214	7.6	148.0	400.0	564.0	31.6	7.0	1554.0	7.6	235.0	798.0	28.6	11.0	1515.0	7.6
215	7.8	158.0	256.0	194.0	47.4	3.0	1831.0	7.8	218.0	240.0	46.7	4.0	1860.0	7.8
216	7.7	123.0	246.0	186.0	39.8	2.5	1288.0	7.7	241.0	300.0	25.3	4.5	1328.0	7.7
217	7.7	133.0	410.0	204.0	62.7	4.5	1490.0	7.7	161.0	256.0	57.8	4.5	1421.0	7.7
218	7.5	237.0	388.0	178.0	57.3	4.0	1620.0	7.6	300.0	240.0	53.3	4.5	1571.0	7.7
219	7.7	203.0	357.0	222.0	58.6	3.0	1828.0	7.6	215.0	208.0	73.1	5.0	1817.0	7.6
220	7.8	199.0	380.0	244.0	55.7	5.0	1453.0	7.8	110.0	196.0	60.2	3.5	1430.0	7.8
221	7.8	214.0	941.0	304.0	53.3	6.5	1454.0	7.8	396.0	320.0	57.5	7.0	1528.0	7.8
222	7.7	208.0	298.0	260.0	37.7	4.0	1236.0	7.8	245.0	284.0	46.5	5.0	1285.0	7.8
223	7.9	206.0	305.0	188.0	56.4	4.0	1200.0	7.9	220.9	272.0	50.0	6.0	1228.0	7.9
224	7.7	247.0	361.0	286.0	51.0	4.5	1180.0	7.7	146.0	212.0	56.4	2.5	1251.0	7.7
225	7.5	438.0	681.0	370.0	56.8	6.0	1396.0	7.6	212.0	348.0	64.4	6.0	1446.0	7.6
226	7.8	255.0	282.0	166.0	67.5	2.0	950.0	7.8	284.0	172.0	65.1	2.0	965.0	7.8
227	7.7	283.0	310.0	170.0	61.2	2.5	1065.0	7.8	235.0	270.0	51.1	5.5	1100.0	7.8
228	7.9	313.0	341.0	512.0	33.6	6.0	915.0	7.9	205.0	520.0	33.8	7.5	976.0	7.9
229	7.7	300.0	610.0	452.0	45.1	8.0	1313.0	7.7	375.0	556.0	43.9	10.0	1312.0	7.7
230	7.8	177.0	412.0	196.0	69.4	4.5	2190.0	7.8	330.0	476.0	52.1	7.5	2330.0	7.8

The Colorize feature colors the cells in the original data table that correspond to the duplicate sets in the report.

Statistical Details for the Explore Patterns Utility

Rarity in Longest Runs

To calculate the rarity for longest runs, first define the following variables:

n = the number of rows in the column

k = the number of times a specific value occurs in the column

$p = k/n$ = the probability of observing the specific value in the column

m = the length of the run

N = the number of unique runs

Then, the rarity for longest runs is calculated as follows:

$$Rarity = -\log_2(1 - (1 - p^{m-1})^N)$$

Rarity in Longest Sequences

To calculate the rarity for longest sequences, first define the following variables:

p = the probability of observing the specific sequence one time in the column

k = the number of times the starting value of the sequence occurs in the column

Then, the rarity for longest sequences is calculated as follows:

$$Rarity = -\log_2(1 - (1 - p)^k)$$

Chapter 22

Response Screening

Test Many Responses in Large-Scale Data

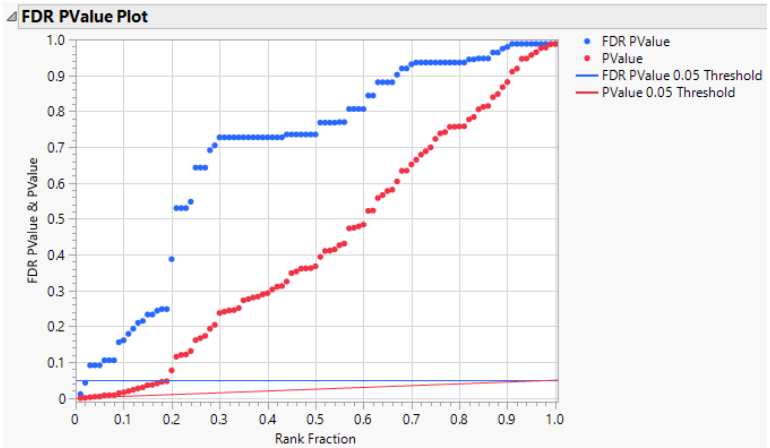
The analysis of large-scale data sets, where hundreds or thousands of measurements are taken on a part or an organism, requires innovative approaches. But testing many responses for the effects of factors can be challenging, if not misleading, without appropriate methodology.

Response Screening automates the process of conducting tests across a large number of responses. Your test results and summary statistics are presented in data tables, rather than reports, to enable data exploration. A False-Discovery Rate approach guards against incorrect declarations of significance. Plots of *p*-values are scaled using the LogWorth, making them easily interpretable.

Because large scale data sets are often messy, Response Screening presents methods that address irregularly distributed and missing data. A robust estimate method allows outliers to remain in the data, but reduces the sensitivity of tests to these outliers. Missing data options allow missing values to be included in the analysis. These features enable you to analyze your data without first conducting an extensive analysis of data quality.

When you have many observations, even differences that are of no practical interest can be statistically significant. Response Screening presents tests of practical difference, where you specify the difference that you are interested in detecting. On the other hand, you might want to know whether differences do not exceed a given magnitude, that is, if the means are equivalent. For this purpose, Response Screening presents equivalence tests.

Figure 22.1 Example of a Response Screening Plot



Contents

Overview of the Response Screening Platform . . . 449
Example of Response Screening . . . 450
Launch the Response Screening Platform . . . 453
The Response Screening Report . . . 455
Result Table . . . 456
FDR PValue Plot . . . 458
FDR LogWorth by Effect Size . . . 459
FDR LogWorth by RSquare . . . 460
The PValues Data Table . . . 460
PValues Data Table Scripts . . . 461
Response Screening Platform Options . . . 461
Means Data Table . . . 463
Compare Means Data Table . . . 464
The Response Screening Personality in Fit Model . . . 466
Launch Response Screening in Fit Model . . . 467
The Fit Response Screening Report . . . 468
Fit Response Screening Options . . . 469
PValues Data Table . . . 470
Y Fits Data Table . . . 471
Additional Examples of Response Screening . . . 472
Example of Tests of Practical Significance and Equivalence . . . 472
Example of the MaxLogWorth Option . . . 474
Example of Robust Fit . . . 475
Response Screening Personality . . . 478
Statistical Details for the Response Screening Platform . . . 479
The False Discovery Rate . . . 479

Overview of the Response Screening Platform

Response Screening automates the process of conducting tests across a large number of responses. It tests each response that you specify against each factor that you specify. Response screening addresses two main issues connected with large-scale data. These are the need to conduct many tests, and the requirement to deal effectively with outliers and missing values.

Response screening is available as a platform and as a Fit Model personality. In both cases, it performs tests analogous to those found in the Fit Y by X platform, as shown in Table 22.1. As a personality, it performs tests of the response against the individual model effects.

To facilitate and support the multiple inferences that are required, Response Screening provides these features:

Data Tables Results are shown in data tables, as well as in a report, to enable you to explore, sort, search, and plot your results. Statistics that facilitate plot interpretation are provided, such as the LogWorth of *p*-values.

False Discovery Rates Because you are conducting a large number of tests, you need to control the overall rate of declaring tests significant. Response screening controls the *false discovery rate*. The False Discovery Rate (FDR) is the expected proportion of significant tests that are incorrectly declared significant (Benjamini and Hochberg 1995; Westfall et al. 2011).

Tests of Practical Significance When you have many observations, even small effects that are of no practical consequence can result in statistical significance. To address this issue, you can define an effect size that you consider to be of *practical significance*. You then conduct tests of practical significance, thereby only detecting effects large enough to be of pragmatic interest.

Equivalence Tests When you are studying many factors, you are often interested in those that have essentially equivalent effects on the response. In this case, you can specify an effect size that defines practical equivalence and then conduct equivalence tests.

To address issues that arise when dealing with messy data, Response Screening provides features to deal with outliers and missing data. These features enable you to analyze your data directly, without expending effort to address data quality issues:

Robust Estimation Outliers in your data increase estimates of standard error, causing tests to be insensitive to real effects. Select the Robust option to conduct Huber M-estimation. Outliers remain in the data, but the sensitivity of tests to these outliers is reduced.

Missing Value Options The platform contains an option to treat missing values on categorical predictors in an informative fashion.

Table 22.1 Analyses Performed by Response Screening

Response	Factor	Fit Y by X Analysis	Description
Continuous	Categorical	Oneway	Analysis of Variance
Continuous	Continuous	Bivariate	Simple Linear Regression
Categorical	Categorical	Contingency	Chi-Square
Categorical	Continuous	Logistic	Simple Logistic Regression

The Response Screening platform generates a report and a data table: the Response Screening report and the PValues table. The Response Screening personality generates a report and two data tables: the Fit Response Screening report, the PValues table, and the Y Fits table.

The JSL command `Summarize Y by X` performs the same function as the Response Screening platform but without creating a platform window. See the *JSL Syntax Reference*.

Example of Response Screening

The Probe.jmp sample data table contains 387 characteristics (the Responses column group) measured on 5800 wafers. The Lot ID and Wafer Number columns uniquely identify the wafer. You are interested in which of the characteristics show different values across a process change (Process).

1. Select **Help > Sample Data Library** and open Probe.jmp.
2. Select **Analyze > Screening > Response Screening**.
3. Select the Responses column group and click **Y, Response**.
4. Select Process and click **X**.
5. Enter 100 in the **MaxLogWorth** box.

   A logworth ($-\log_{10}(p\text{-value})$) of 100 or larger corresponds to an extremely small p-value. Setting a value for the MaxLogWorth helps control the scale of plots as it limits the reported logworth value.
6. Click **OK**.

   The report shows a table of p-value results and the FDR PValue Plot, and also contains two other plot reports.

The FDR PValue Plot shows two types of *p*-values, FDR PValue and PValue, for each of the 387 tests. These are plotted against Rank Fraction. PValue is the usual *p*-value for the test of a Y against Process. The FDR PValue is a *p*-value that is adjusted to guarantee a given false discover rate (FDR), here 0.05. The FDR PValues are plotted in blue and the PValues are plotted in red. The Rank Fraction ranks the FDR *p*-values from smallest to largest, in order of decreasing significance.

Both the horizontal blue line and the sloped red line on the plot are thresholds for FDR significance. Tests with FDR *p*-values that fall below the blue line are significant at the 0.05 level when adjusted for the false discovery rate. Tests with ordinary *p*-values that fall below the red line are significant at the 0.05 level. In this way, the plot enables you to read FDR significance from either set of *p*-values.

Figure 22.2 Response Screening Report for 387 Tests against Process

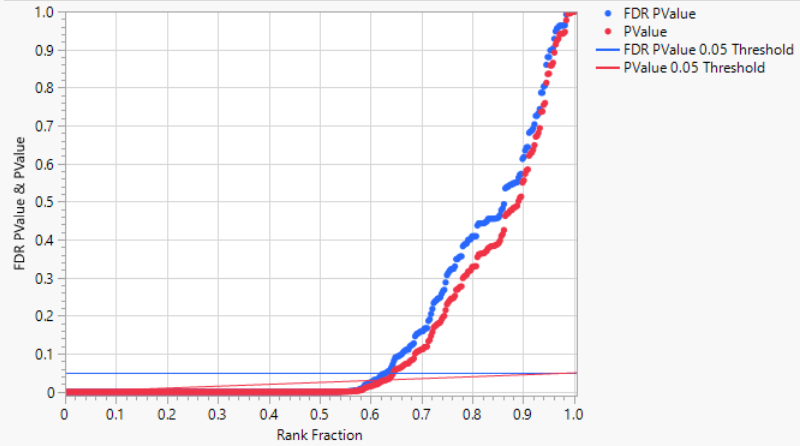
Response Screening

LogWorths limited to 100 for graph scaling purposes

Result Table

Y	X	Count	PValue	LogWorth	FDR PValue	FDR LogWorth	Effect Size	Rank Fraction	RSquare
DELW_M2	Process	5728	4e-322	100.000	0	100.000	0.47114	0.09115	0.2297
DELW_NENBNI	Process	5792	4e-322	100.000	0	100.000	0.48812	0.08594	0.2383
DELW_NSINK	Process	5782	4e-322	100.000	0	100.000	0.58572	0.07031	0.3431
DELW_PCOLL	Process	5785	4e-322	100.000	0	100.000	0.69477	0.00521	0.8866
NISO_TUB-TRENCH_VGATERTFF	Process	5800	2e-164	100.000	1.4e-163	100.000	0.34772	0.11458	0.1209
R_RPB_100X20	Process	5784	4e-128	100.000	3.3e-127	100.000	0.30889	0.1224	0.0954
R_TRENCH	Process	5800	4e-322	100.000	0	100.000	0.67328	0.0026	0.9748
VDP_NEMIT	Process	5788	2e-202	100.000	2.2e-201	100.000	0.36968	0.10417	0.1472
VDP_NENBNI	Process	5792	4e-322	100.000	0	100.000	0.8138	0.02344	0.6624
VDP_NSINK	Process	5787	4e-322	100.000	0	100.000	0.66372	0.02604	0.5487
VDP_PBASE	Process	5791	1e-251	100.000	1e-250	100.000	0.40048	0.09635	0.1800
VDP_PBL	Process	5793	4e-322	100.000	0	100.000	0.98523	0.07292	0.3365
VDP_PCOLL	Process	5800	4e-322	100.000	0	100.000	0.6806	0.01823	0.6950
30D1_7X100_VDON1U	Process	5794	8e-124	100.000	6.1e-123	100.000	0.68971	0.125	0.0922
30N1_12X100_BVCEO	Process	5800	4e-322	100.000	0	100.000	0.69387	0.03646	0.4815
30N1_12X100_HFE1UA	Process	5771	4e-322	100.000	0	100.000	0.59561	0.0599	0.3766
30N1_12X100_HFEPEAK*VA	Process	5795	4e-322	100.000	0	100.000	0.6609	0.08333	0.2698
30N1_12X100_ICHFEPEAK	Process	5764	4e-322	100.000	0	100.000	0.5898	0.03385	0.5048
30N1_12X100_ICHFEQ	Process	5711	3e-267	100.000	3.1e-266	100.000	0.50793	0.09375	0.1924
30N1_12X100_VA1.E-5	Process	5770	4e-246	100.000	4.3e-245	100.000	0.46474	0.09896	0.1769

FDR PValue Plot



The FDR PValue Plot shows that more than 60% of the tests are significant. A handful of tests are significant using the unadjusted *p*-value, but not significant using the FDR *p*-value. These tests correspond to the red points that are above the red line, but below the blue line.

To identify the characteristics that are significantly different across Process, you can drag a rectangle around the appropriate points in the plot. This selects the rows corresponding to these points in the Result table, where the names of the characteristics are given in the first column.

The Result table contains 387 rows, one for each response measure in the Responses group. The response is given in the first column, called Y. Each response is tested against the effect in the X column, namely, Process.

The remaining columns give information about the test of Y against X. Here the test is a Oneway Analysis of Variance. In addition to other information, the table gives the test's *p*-value, LogWorth, FDR (False Discovery Rate) *p*-value, and FDR LogWorth. Use this table to sort by the various statistics, select rows, or plot quantities of interest.

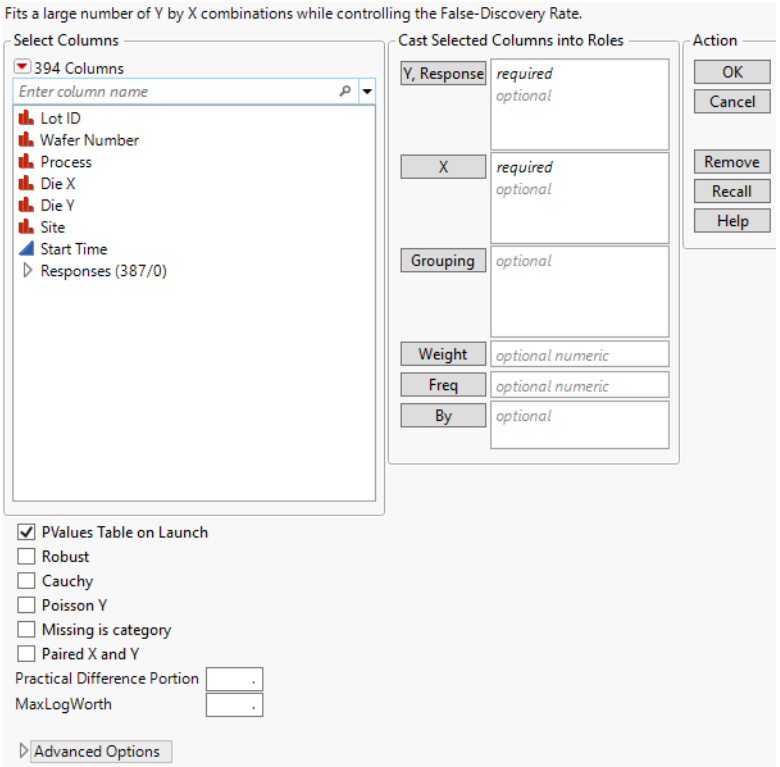
Notice that LogWorth and FDR LogWorth values that correspond to *p*-values of 1e-100 or less are reported as 100, because you set MaxLogWorth to 100 in the launch window. Also, cells corresponding to FDR LogWorth values greater than two are colored with an intensity gradient.

See "The Response Screening Report" for more information about the report and PValues table.

Launch the Response Screening Platform

Launch the Response Screening platform by selecting **Analyze > Screening > Response Screening**.

Figure 22.3 Response Screening Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Launch Window Roles

Y, Response Identifies the response columns containing the measurements to be analyzed.

X Identifies the columns against which you want to test the responses.

Grouping Analyzes the rows assigned to each level of the specified column separately. All results are presented in a single table and report.

Weight Identifies a column whose values assign a weight to each row. These values are used as weights in the analysis. See *Fitting Linear Models*.

Freq Identifies a column whose values assign a frequency to each row. These values enable you to account for pre-summarized data. See *Fitting Linear Models*.

By For each level of the specified column, analyzes the corresponding Ys and Xs and presents the results in separate tables and reports.

Launch Window Options

PValues Table on Launch Creates a data table for the *p*-values and individual model fit statistics. This data table is linked to the Result Table in the Response Screening report.

Robust (Available only for continuous responses.) Fits regression and ANOVA models using the Huber M-estimation method (Huber and Ronchetti 2009). This method estimates parameters by minimizing the Huber loss function. The Huber loss function penalizes outliers, which reduces the influence of the outliers on tests. If there are no outliers, the estimates are close to the least squares estimates. When you select the Robust option, several Robust measures and model fit statistics columns are added to the Result Table. See *Basic Analysis* for more information about Huber M-estimation. For an example, see "Example of Robust Fit".

Caution: The Robust option can increase processing time.

Cauchy (Available only for continuous responses.) Estimates parameters using maximum likelihood and a Cauchy link function. This estimation method assumes that the errors have a Cauchy distribution, which has fatter tails than the normal distribution, resulting in a reduced emphasis on outliers. This option can be useful if you have a large proportion of outliers in your data. However, if your data are close to normal with only a few outliers, this option can lead to incorrect inferences. When you select the Cauchy option, several Cauchy measures and model fit statistics columns are added to the Result Table.

Poisson Y Fits each Y response as a count having a Poisson distribution. The test is performed only for categorical X. This option is appropriate when your responses are counts.

Missing is category For any categorical X variable, treats missing values on X as a category.

Paired X and Y Performs tests only for Y columns paired with X columns according to their order in the **Y, Response** and **X** lists. The first Y is paired with the first X, the second Y with the second X, and so on.

Practical Difference Portion The fraction of the specification range, or of an estimated six standard deviation range, that represents a difference that you consider pragmatically meaningful. If Spec Limits is not set as a column property, a range of six standard

deviations is estimated for the response. The standard deviation estimate is computed from the interquartile range (IQR), as $\hat{\sigma} = (IQR)/(1.3489795)$.

If no Practical Difference Proportion is specified, its value defaults to 0.10. Tests of practical significance and equivalence tests use this difference to determine the practical difference. See "Compare Means Data Table".

MaxLogWorth Controls the scale of plots involving LogWorth values ($-\log_{10}$ of p-values). LogWorth values that exceed MaxLogWorth are plotted as MaxLogWorth to prevent extreme scales in LogWorth plots. See "Example of the MaxLogWorth Option" for an example.

Advanced Options Contains the following options:

Kappa Adds a new column called Kappa to the data table. If Y and X are both categorical and have the same levels, kappa is provided. This is a measure of agreement between Y and X.

Corr The Corr option computes the Pearson product-moment correlation in terms of the indices defined by the value ordering.

If X and Y are both binary, the Pearson product-moment calculation gives Spearman's rho and Kendall's Tau-b. Otherwise, a value of Corr that is large in magnitude indicates an association; a Corr value that is small in magnitude does not preclude an association.

Force X Categorical Ignores the modeling type and treats all X columns as categorical.

Force X Continuous Ignores the modeling type and treats all X columns as continuous.

Force Y Categorical Ignores the modeling type and treats all Y columns as categorical.

Force Y Continuous Ignores the modeling type and treats all Y columns as continuous.

Unthreaded Suppresses multithreading used for computational speed.

The Response Screening Report

The Response Screening report consists of a results table and several plots. These plots focus on False Discovery Rate (FDR) statistics. See "The False Discovery Rate".

The default plots are the FDR PValue Plot, the FDR LogWorth by Effect Size, and the FDR LogWorth by RSquare. If you select the Robust option on the launch window, Robust versions of each of these reports are also presented. In addition, a Robust LogWorth by FDR LogWorth plot is presented to help assess the impact of using the robust fit.

Result Table

The Response Screening Result table contains a row for each pair of Y and X variables. The columns of the table contain measures and model fit statistics that are specific to the selected fit and Y and X modeling types.

Group (Appears only if there is a grouping variable.) The level of the grouping column.

Y The specified response columns.

X The specified factor columns.

Count The number of rows used for testing, or the corresponding sum of the Freq or Weight variable.

PValue The *p*-value for the significance test corresponding to the pair of Y and X variables. For more information about Fit Y by X statistics, see *Basic Analysis*.

LogWorth The quantity $-\log_{10}(p\text{-value})$. This transformation adjusts *p*-values to provide an appropriate scale for graphing. A value that exceeds 2 is significant at the 0.01 level ($-\log_{10}(0.01) = 2$).

FDR PValue The False Discovery Rate *p*-value calculated using the Benjamini-Hochberg technique. This technique adjusts the *p*-values to control the false discovery rate for multiple tests. If there is no Group variable, the set of multiple tests includes all tests displayed in the table. If there is a Group variable, the set of multiple tests consists of all tests conducted for each level of the Group variable. For more information about the FDR correction, see Benjamini and Hochberg (1995). For more information about the false discovery rate, see "The False Discovery Rate".

FDR LogWorth The quantity $-\log_{10}(\text{FDR PValue})$. This is the statistic to use for plotting and assessing significance. Note that small *p*-values result in high FDR LogWorth values. Cells corresponding to FDR LogWorth values greater than two (*p*-values less than 0.01) are colored with an intensity gradient.

Effect Size Indicates the extent to which response values differ across the levels or values of X. Effect sizes are scale invariant.

- When Y is continuous, the effect size is the square root of the average sum of squares from the hypothesis test divided by a robust estimate of the response standard deviation. If the interquartile range (IQR) is nonzero and IQR > range/20, the standard deviation estimate is IQR/1.3489795. Otherwise the sample standard deviation is used.
- When Y is categorical and X is continuous, the effect size is the square root of the average ChiSquare value for the whole model test.
- When Y and X are both categorical, the effect size is the square root of the average Pearson ChiSquare.

Rank Fraction The rank of the FDR LogWorth expressed as a fraction of the number of tests. If the number of tests is *m*, the largest FDR LogWorth value has Rank Fraction 1/*m*, and the smallest has Rank Fraction 1. The Rank Fraction is used in plotting the PValues and FDR PValues in rank order of decreasing significance.

RSquare (Appears only when Y is continuous.) The coefficient of determination, which measures the proportion of total variation explained by the model.

Kappa (Appears only when Y and X are both categorical and have the same number of levels.) A measure of agreement between Y and X.

Corr (Appears only when Y and X are both categorical.) The Pearson product-moment correlation in terms of the indices defined by the value ordering.

The following columns are added to the Result table when the Robust option is selected in the launch window. The Robust option applies only when Y is continuous, so Robust column cells are empty when Y is categorical.

Robust PValue The *p*-value for the significance test corresponding to the pair of Y and X variables using a robust fit.

Robust LogWorth The quantity $-\log_{10}$(Robust PValue).

Robust FDR PValue The False Discovery Rate calculated for the Robust PValues using the Benjamini-Hochberg technique. If there is no Group variable, the multiple test adjustment applies to all tests displayed in the table. If there is a Group variable, the multiple test adjustment applies to all tests conducted for each level of the Group variable.

Robust FDR LogWorth The quantity $-\log_{10}$(Robust FDR PValue).

Robust Rank Fraction The rank of the Robust FDR LogWorth expressed as a fraction of the number of tests.

Robust Chisq The chi-square value associated with the robust test.

Robust Sigma The robust estimate of the error standard deviation.

Robust Outlier Portion The portion of the values whose distance from the robust mean exceeds three times the Robust Sigma.

The following columns are added to the Result table when the Cauchy option is selected in the launch window. The Cauchy option applies only when Y is continuous, so Cauchy column cells are empty when Y is categorical.

Cauchy PValue The *p*-value for the significance test corresponding to the pair of Y and X variables using a Cauchy fit.

Cauchy LogWorth The quantity $-\log_{10}$(Cauchy PValue).

Cauchy FDR PValue The False Discovery Rate calculated for the Cauchy PValues using the Benjamini-Hochberg technique. If there is no Group variable, the multiple test adjustment applies to all tests displayed in the table. If there is a Group variable, the multiple test adjustment applies to all tests conducted for each level of the Group variable.

Cauchy FDR LogWorth The quantity $-\log_{10}$(Cauchy FDR PValue).

Cauchy Rank Fraction The rank of the Cauchy FDR LogWorth expressed as a fraction of the number of tests.

Cauchy ChiSquare The chi-square value associated with the test from the Cauchy fit.

Cauchy Sigma The estimate of the error standard deviation from the Cauchy fit.

Cauchy Outlier Portion The portion of the values whose distance from the mean of the Cauchy fit exceeds three times the Cauchy Sigma.

FDR PValue Plot

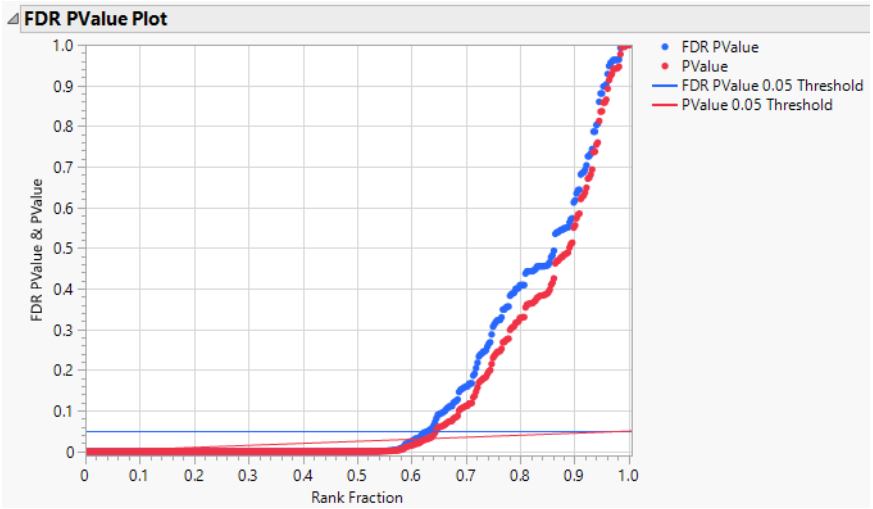
The FDR PValue Plot report shows a plot of FDR PValues and PValues against the Rank Fraction. The Rank Fraction ranks the PValues in order of decreasing significance. FDR PValues are plotted in blue and PValues in red.

A blue horizontal line shows the 0.05 significance level. Note that you can change this level by double-clicking the vertical axis, removing the current reference line, and adding a new reference line.

A red increasing line provides an FDR threshold for unadjusted *p*-values. A *p*-value falls below the red line precisely when the FDR-adjusted *p*-value falls below the blue line. This enables you to read significance relative to the FDR from either the adjusted or unadjusted *p*-values.

Figure 22.4 shows the FDR PValue Plot for the Probe.jmp sample data table. Note that some tests are significant according to the usual *p*-value but not according to the FDR *p*-value.

Figure 22.4 FDR PValue Plot



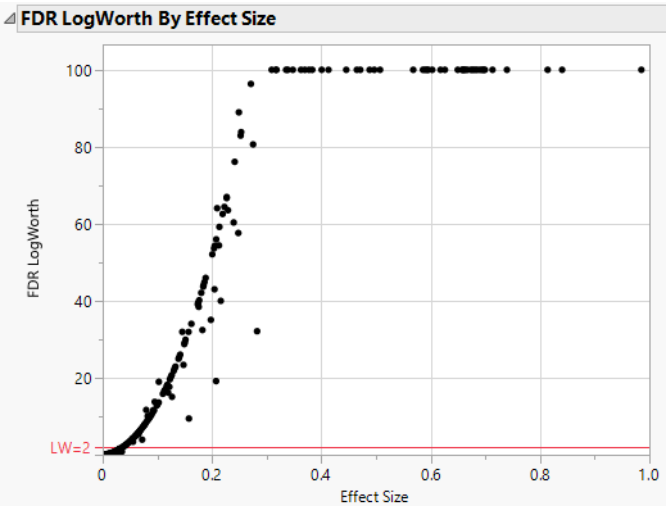
FDR LogWorth by Effect Size

When you have large effects, the associated *p*-values are often very small. Visualizing these small values graphically can be challenging. When transformed to the LogWorth ($-\log_{10}$(*p*-value)) scale, highly significant *p*-values have large LogWorths and nonsignificant *p*-values have low LogWorths. A LogWorth of zero corresponds to a nonsignificant *p*-value of 1. Any LogWorth above 2 corresponds to a *p*-value below 0.01.

In the FDR LogWorth by Effect Size plot, the vertical axis is the FDR LogWorth and the horizontal axis is the Effect Size. Generally, larger effects lead to more significant *p*-values and larger LogWorths. However, this relationship is not necessarily strong because significance also depends on the error variance. In fact, large LogWorths can be associated with small effects, and small LogWorths can be associated with large effects, because of the size of the error variance. The FDR LogWorth by Effect Size plot enables you to explore this relationship.

Figure 22.5 shows the FDR LogWorth by Effect size plot for the Probe.jmp sample data table with MaxLogWorth set to 100. Most FDR LogWorth values exceed 2, which indicates that most effects are significant at the 0.01 level. The FDR LogWorth values of 100 correspond to extremely small *p*-values.

Figure 22.5 FDR LogWorth by Effect Size



FDR LogWorth by RSquare

The FDR LogWorth by RSquare plot shows the FDR LogWorth on the vertical axis and RSquare values on the horizontal axis. Larger LogWorth values tend to be associated with larger RSquare values, but this relationship also depends on the number of observations.

The PValues Data Table

If the PValues Tables on Launch option is selected in the launch window, a PValues data table is created when the Response Screening platform is launched. This data table contains the same information as the Result table that is in the Response Screening report. The PValues data table also contains the additional, model specific columns:

YMean The mean of Y.

SSE Appears when Y is continuous. The sum of squares for error.

DFE Appears when Y is continuous. The degrees of freedom for error.

MSE Appears when Y is continuous. The mean squared error.

F Ratio Appears when Y is continuous. The F Ratio for the analysis of variance or regression test.

Intercept Appears when Y and X are both continuous. The intercept of the regression model relating the corresponding pair of Y and X variables.

Slope Appears when Y and X are both continuous. The slope of the regression model relating the corresponding pair of Y and X variables.

DF Appears when Y and X are both categorical. The degrees of freedom for the ChiSquare test.

LR Chisq Appears when Y and X are both categorical. The value of the Likelihood Ratio ChiSquare statistic.

Robust Elapsed Time (Appears only when Robust is specified in the launch window.) Computing time, in seconds, required to create the Robust report.

PValues Data Table Scripts

Relevant scripts are saved to the PValues data table. All but one of these reproduce plots provided in the report. When you select rows in the PValues table, the Fit Selected script produces the appropriate Fit Y by X analyses.

Response Screening Platform Options

The Response Screening red triangle menu contains options to customize the display and to compute and save calculated data.

Results Table Shows or hides the table of results.

Fit Selected Items For selected relationships, adds the appropriate Fit Y by X reports to the Response Screening report. You can select relationships by selecting points in the plots, rows in the Result Table, or if opened, rows in the PValue data table.

Select Columns Selects the columns in the original data table that correspond to rows that you select in the Result Table, to points that you select in plots in the Response Screening report window, or if opened, rows in the PValues table. Select the rows or points first, then select Select Columns. The corresponding columns in the data table are selected. You can select columns corresponding to additional rows in the PValues table or points in plots by first selecting them and then selecting Select Columns again. To select columns corresponding to different rows or points, first clear the current column selection in the original data table.

Select Where Opens the Select Where window. You can select specific responses in the Result table that correspond to a particular condition by using the Comparison menu and Value text box. For example, you can select all responses such that Effect Size > 0.80. After you click OK, the responses are selected in the Result table.

Tip: You can also access the Select Where window by right-clicking anywhere in the Result table.

Save PValues Creates the PValues data table. See "The PValues Data Table".

Save Means For continuous Ys and categorical Xs, creates a data table with the counts, means, and standard deviations for each level of the categorical variable. If the Robust option is selected, the robust mean is included.

Save Compare Means For continuous Ys and categorical Xs, tests all pairwise comparisons across the levels of the categorical variable. For each comparison, the data table gives the usual *t* test, a test of practical significance, an equivalence test, and a column that uses color coding to summarize the results. The data table also contains a script that plots Practical LogWorth by Relative Practical Difference. See "Compare Means Data Table". For an example, see "Example of Tests of Practical Significance and Equivalence".

Save Std Residuals Saves a new group of columns to the original data table and places these in a column group call Residual Group. For each continuous Y and categorical X, a column is constructed containing the residuals divided by their estimated standard deviation. In other words, the column contains standardized residuals. The column is defined by a formula.

If the Robust option is selected, standardized residual columns are constructed using robust fits and robust estimates.

Save Outlier Indicator Saves a new group of columns to the original data table and places these in a column group call Outlier Group. Save Outlier Indicator is most effective when you have selected the Robust option.

For each continuous Y and categorical X, a column that indicates outliers is constructed. An outlier is a point whose distance to the predicted value exceeds three times an estimate of sigma. In other words, an outlier is a point whose standardized residual exceeds three. The column is defined by a formula.

If the Robust option is selected, robust fits and robust estimates are used. An outlier is a point whose distance to the predicted value exceeds three times the robust estimate of sigma.

The Cluster Outliers script is added to the original data table. The script shows outliers on a hierarchical cluster plot of the data.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Means Data Table

The Means data table contains a row for each combination of response and X level. For the Probe.jmp sample data table, there are 387 response variables, each tested against Process at two levels. The Means table contains 387x2 = 774 rows.

Figure 22.6 Means Data Table

Save Means

Columns (6/0)

Y
X
Level
Count
Mean
StdDev

Rows

All rows	774
Selected	0
Excluded	0
Hidden	0
Labeled	0

	Y	X	Level	Count	Mean	StdDev
1	DELL_RPNBR	Process	New	3044	0.2035840106	0.1916031302
2	DELL_RPNBR	Process	Old	2750	0.2346329436	0.1278299414
3	DELL_RPPBR	Process	New	3044	-0.068072506	0.1784161107
4	DELL_RPPBR	Process	Old	2750	-0.043321135	1.9165048167
5	DELW_M1	Process	New	3039	-0.04387197	1.0774346893
6	DELW_M1	Process	Old	2750	-0.071400861	0.0389764703
7	DELW_M2	Process	New	3039	0.8014275806	0.0698264259
8	DELW_M2	Process	Old	2689	0.7274958056	0.0649383327
9	DELW_NBASE	Process	New	3028	0.1957655669	77.801881218
10	DELW_NBASE	Process	Old	2750	1.4765185729	0.1326787074
11	DELW_NEMIT	Process	New	3038	0.3411755212	0.0940762694
12	DELW_NEMIT	Process	Old	2750	0.3813085954	0.0976561441
13	DELW_NENBNI	Process	New	3043	3.7172025841	1.1419433505
14	DELW_NENBNI	Process	Old	2749	4.6556238745	0.1893626325
15	DELW_NSINK	Process	New	3032	9.2876152816	1.2840573691
16	DELW_NSINK	Process	Old	2750	7.8902297629	0.3781489659
17	DELW_PBASE	Process	New	3036	1.1651478779	0.1351193171
18	DELW_PBASE	Process	Old	2750	1.2087969753	0.1486923744

The Means data table includes the following columns:

Y The continuous response variables.

X The categorical variables.

Level The level of the categorical X variable.

Count The count of values in the corresponding Level.

Mean The mean of the Y variable for the specified Level.

StdDev The standard deviation of the Y variable for the specified Level.

Robust Mean The robust M-estimate of the mean. Appears when you select the Robust option on the launch window.

Compare Means Data Table

When your data table consists of a large number of rows (large *n*), the standard error used in testing can be very small. As a result, tests might be statistically significant, when in fact, the observed difference is too small to be of practical consequence. Tests of practical significance enable you to specify the size of the difference that you consider worth detecting. This difference is called the *practical difference*. Instead of testing that the difference is zero, you test whether the difference exceeds the practical difference. As a result, the tests are more meaningful, and fewer tests need to be scrutinized.

Equivalence tests enable you to determine whether two levels have essentially the same effect, from a practical perspective, on the response. In other words, an equivalence test tests whether the difference is smaller than the practical difference.

The Compare Means data table provides results for both tests of practical difference and tests of practical equivalence. Each row compares a response across two levels of a categorical factor. Results of the pairwise comparisons are color-coded to facilitate interpretation. See "Practical Difference" for a description of how the practical difference is specified. See "Example of Tests of Practical Significance and Equivalence" for an example.

Figure 22.7 Compare Means Data Table

Untitled 23
Practical Log...cal Difference

Columns (11/0)
Y
Level i
Level j
X
Difference
Std Err Dif
Plain Dif PValue
Practical Difference
Practical Dif PValue
Practical Equiv PValue
Practical Result

Rows
All rows 387
Selected 0
Excluded 0
Hidden 0
Labeled 0

	Y	Level i	Level j	X	Difference	Std Err Dif	Plain Dif PValue	Practical Difference	Practical Dif PValue	Practical Equiv PValue	Practical Result
1	DELL_RPNBR	New	Old	Process	-0.031048933	0.004326425	8.044596e-13	0.0990979657	1	6.407405e-55	Equivalent
2	DELL_RPPBR	New	Old	Process	-0.024751371	0.0349025251	0.4782555718	0.7959556871	1	3.07185e-104	Equivalent
3	DELW_M1	New	Old	Process	0.0275288913	0.0205582035	0.1806001982	0.4687012219	1	1.160101e-98	Equivalent
4	DELW_M2	New	Old	Process	0.0739317751	0.0017890895	0	0.0469881911	1.350706e-50	1	Different
5	DELW_NBASE	New	Old	Process	-1.280753006	1.4836374126	0.388034756	33.792861061	1	1.28251e-102	Equivalent
6	DELW_NEMIT	New	Old	Process	-0.040133074	0.002521398	7.306034e-56	0.0565587625	1	3.954083e-11	Equivalent
7	DELW_NENBNI	New	Old	Process	-0.93842129	0.0220490008	0	0.5760161527	1.142115e-59	1	Different
8	DELW_NSINK	New	Old	Process	1.3973855188	0.0254310074	0	0.7148733981	6.27067e-150	1	Different
9	DELW_PBASE	New	Old	Process	-0.043649097	0.0037311377	2.908704e-31	0.0864253619	1	2.094533e-30	Equivalent
10	DELW_PCOLL	New	Old	Process	1.8169399617	0.0085444762	0	0.78359742	0	1	Different
11	DELW_PEMIT	New	Old	Process	-0.022697392	0.0534170363	0.6709186367	1.2176717085	1	1.10189e-106	Equivalent
12	DELW_PSINK	New	Old	Process	0.911093371	0.0678503246	1.666427e-40	1.5705731641	1	1.83559e-22	Equivalent
13	DELW_RPNBR	New	Old	Process	-0.101421251	0.0100452789	9.008231e-24	0.2309922348	1	7.505304e-38	Equivalent
14	DELW_RPPBR	New	Old	Process	0.3015474189	0.0280757905	1.169725e-26	0.6465385194	1	1.402789e-34	Equivalent
15	DELW_SICR	New	Old	Process	0.0275370732	0.0176173979	0.1180928699	0.4016073762	1	1.042102e-96	Equivalent
16	M1_COMB_VGAT...	New	Old	Process	-0.532453055	0.6037284633	0.3778451214	13.77484916	1	8.16456e-103	Equivalent
17	M1_TRENCH_VG...	New	Old	Process	-0.61657228	0.1695447206	0.0002786205	3.872532637	1	4.896901e-80	Equivalent
18	M2/M1_CAP_VG...	New	Old	Process	-0.047230999	0.0978790012	0.6294372314	2.2331316201	1	2.36705e-106	Equivalent
19	M2_COMB_BB_V...	New	Old	Process	0.0001788376	0.1270241159	0.9988767052	2.8980258306	1	1.08191e-110	Equivalent
20	M2_COMB_VGAT...	New	Old	Process	0.2971147541	0.2860872543	0.2990592705	6.527621621	1	1.91844e-101	Equivalent
21	NISO_TUB-TREN...	New	Old	Process	7.0118092399	0.2482751793	1.58784e-164	6.0414047752	0.0000469484	0.9999530516	Different
22	NISO_TUB-TUB_V...	New	Old	Process	-0.074789866	0.1081535234	0.4892691678	2.4675993783	1	1.70833e-104	Equivalent
23	PS_RPNBR	New	Old	Process	-64.69026527	15.2943887	0.0000237641	353.51614685	1	1.545634e-77	Equivalent

The Compare Means data table contains a script that plots Practical LogWorth by Relative Practical Difference. Relative Practical Difference is defined as the actual difference divided by the practical difference.

Y The continuous response variables.

X The categorical variables.

Leveli The level of the categorical X variable.

Levelj The level of the categorical X variable being compared to Leveli.

Difference The estimated difference in means across the two levels. If the Robust option is selected, robust estimates of the means are used.

Std Err Diff The standard error of the difference in means. This is a robust estimate if the Robust option is selected.

Plain Dif PValue The *p*-value for the usual Student's *t* test for a pairwise comparison. This is the robust version of the *t* test when the Robust option is selected. Tests that are significant at the 0.05 level are highlighted.

Practical Difference The difference in means that is considered to be of practical interest. If you assign a Spec Limit property to the Y variable, the practical difference is computed as the difference between the specification limits multiplied by the Practical Difference Proportion. If no Practical Difference Proportion has been specified, the Practical Difference is the difference between the specification limits multiplied by 0.10.

If you do not assign a Spec Limit property to the Y variable, an estimate of its standard deviation is computed from its interquartile range (IQR). This estimate is $\hat{\sigma} = (IQR)/(1.3489795)$. The Practical Difference is computed as $6\hat{\sigma}$ multiplied by the Practical Difference Proportion. If no Practical Difference Proportion has been specified, the Practical Difference is computed as $6\hat{\sigma}$ multiplied by 0.10.

Practical Dif PValue The *p*-value for a test of whether the absolute value of the mean difference in Y between Leveli and Levelj is less than or equal to the Practical Difference. A small *p*-value indicates that the absolute difference exceeds the Practical Difference. This indicates that Leveli and Levelj account for a difference that is of practical consequence.

Practical Equiv PValue Uses the Two One-Sided Tests (TOST) method to test for a practical difference between the means (Schuirmann 1987). The Practical Difference specifies a threshold difference for which smaller differences are considered practically equivalent. One-sided *t* tests are constructed for two null hypotheses: the true difference exceeds the Practical Difference; the true difference is less than the negative of the Practical Difference. If both tests reject, this indicates that the absolute difference in the means falls within the Practical Difference. Therefore, the groups are considered practically equivalent.

The Practical Equivalence PValue is the largest *p*-value obtained on the one-sided *t* tests. A small Practical Equiv PValue indicates that the mean response for Leveli is equivalent, in a practical sense, to the mean for Levelj.

Practical Result A description of the results of the tests for practical difference and equivalence. Values are color-coded to help identify significant results.

- Different (Pink): Indicates that the absolute difference is significantly greater than the practical difference.
- Equivalent (Green): Indicates that the absolute difference is significantly within the practical difference.
- Inconclusive (Gray): Indicates that neither the test for practical difference nor the test for practical equivalence is significant.

The Response Screening Personality in Fit Model

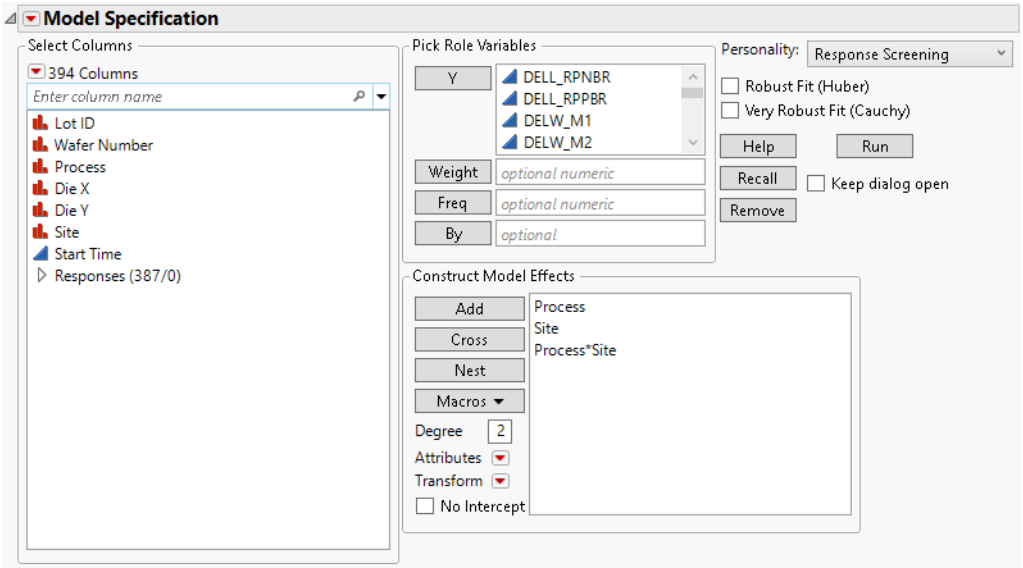
If you are interested in univariate tests against linear model effects, you can fit the Response Screening personality in Fit Model. The report and tables produced test all responses against all model effects.

- "Launch Response Screening in Fit Model"
- "The Fit Response Screening Report"
- "Fit Response Screening Options"
- "PValues Data Table"
- "Y Fits Data Table"

Launch Response Screening in Fit Model

Select **Analyze > Fit Model**. Enter your Ys and model effects. Select **Response Screening** from the Personality list.

Figure 22.8 Response Screening from the Fit Model Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

There are two options for robust estimation:

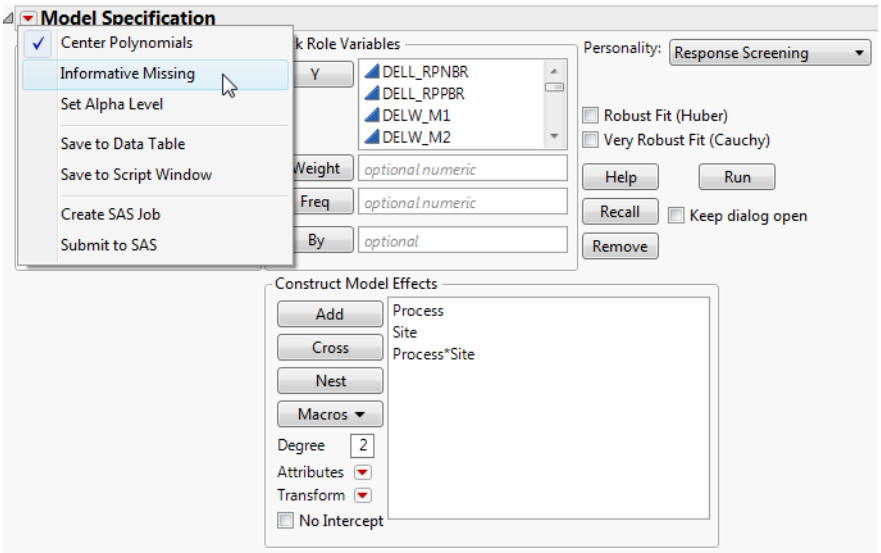
Robust Fit (Huber) Specifies robust (Huber) estimation to down weight outliers for continuous responses. If there are no outliers, these estimates are close to the least squares estimates. Note that this option increases processing time.

Very Robust Fit (Cauchy) Specifies very robust (Cauchy) estimation to down weight outliers for continuous responses. Assumes that the errors have a Cauchy distribution. A Cauchy distribution has fatter tails than the normal distribution, resulting in a reduced emphasis on outliers. This option can be useful if you have a large proportion of outliers in your data. However, if your data are close to normal with only a few outliers, this option can lead to incorrect inferences. The Cauchy option estimates parameters using maximum likelihood and a Cauchy link function.

Tip: If both robust options are selected, the platform uses only Cauchy estimation.

The **Informative Missing** option provides a coding system for missing values. The Informative Missing coding allows estimation of a predictive model despite the presence of missing values. It is useful in situations where missing data are informative. Select this option from the Model Specification red triangle menu.

Figure 22.9 Informative Missing Option



For more information about the Fit Model window, see *Fitting Linear Models*.

The Fit Response Screening Report

The Fit Response Screening report shows two plots:

- The FDR PValue Plot
- The FDR LogWorth by Rank Fraction Plot

The FDR PValue Plot is interpreted in the same way as for the platform itself. See "The Response Screening Report".

The FDR LogWorth by Rank Fraction plot shows FDR LogWorth values plotted against the ranks of the *p*-values. The plotted points decrease or remain constant as rank fraction increases. The plot gives an indication of what proportion of tests are significant. An example using the Response Screening personality is given in "Response Screening Personality".

Fit Response Screening Options

The Fit Response Screening red triangle menu contains options to save the calculated data.

Model Dialog Opens a window containing the model dialog that you have run to obtain the given report.

Save Estimates Opens a data table in which each row corresponds to a response and the columns correspond to the model terms. The entries are the parameter estimates obtained by fitting the specified model. This data table also contains a table variable called Original Data that gives the name of the data table that was used for the analysis. If you specified a By variable, JMP creates an estimates table for each level of the By variable, and the Original Data variable gives the By variable and its level.

Save Prediction Formula Adds columns to the original data table containing prediction equations for all responses.

Save Least Squares Means Opens a data table where each row corresponds to a response and a combination of effect settings. The row contains the least squares mean and standard error for that combination of settings.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

PValues Data Table

The PValues data table contains a row for each pair consisting of a Y variable and a model Effect. The columns in the table include the following. If you select the Robust Fit option on the launch window, the models are fit using Huber M-estimation.

Y The specified response columns.

Effect The specified model effects.

FRatio The test statistic for a test of the Effect. This is the value found in the Effect Tests report in Least Squares Fit.

PValue The *p*-value for the significance test corresponding to the FRatio. See *Fitting Linear Models* for more information about Effect Tests.

LogWorth The quantity $-\log_{10}(p\text{-value})$. This transformation adjusts *p*-values to provide an appropriate scale for graphing. A value that exceeds 2 is significant at the 0.01 level (because $-\log_{10}(0.01) = 2$).

FDR PValue The False Discovery Rate *p*-value calculated using the Benjamini-Hochberg technique. This technique adjusts the *p*-values to control the false discovery rate for multiple tests. For more information about the FDR correction, see Benjamini and Hochberg (1995). For more information about the false discovery rate, see "The False Discovery Rate" or Westfall et al. (2011).

FDR LogWorth The quantity $-\log_{10}(\text{FDR PValue})$. This is the best statistic for plotting and assessing significance. Note that small *p*-values result in high FDR LogWorth values.

Rank Fraction The rank of the FDR LogWorth expressed as a fraction of the number of tests. If the number of tests is *m*, the largest FDR LogWorth value has Rank Fraction 1/*m*, and the smallest has Rank Fraction 1. Equivalently, the Rank Fraction ranks the *p*-values in increasing order, as a fraction of the number of tests. The Rank Fraction is used in plotting the PValues and FDR PValues in rank order of decreasing significance.

Test DF The degrees of freedom for the effect test.

The PValues data table also contains a table variable called Original Data that gives the name of the data table that was used for the analysis. If you specified a By variable, JMP creates a PValues table for each level of the By variable, and the Original Data variable gives the By variable and its level.

Y Fits Data Table

The Y Fits data table contains a row for Y variable. For each Y, the columns in the table summarize information about the model fit. If you select the Robust Fit option on the launch window, the models are fit using Huber M-estimation.

Y The specified response columns.

RSquare The multiple correlation coefficient.

RMSE The Root Mean Square Error.

Count The number of observations (or sum of the Weight variable).

Overall FRatio The test statistic for model fit from the Analysis of Variance report in Least Squares Fit.

Overall PValue The *p*-value for the overall test of model significance.

Overall LogWorth The LogWorth of the *p*-value for the overall test of model significance.

Overall FDR PValue The overall *p*-value adjusted for the false discovery rate. (See "The Response Screening Report".)

Overall FDR LogWorth The LogWorth of the Overall FDR PValue.

Overall Rank Fraction The rank of the Overall FDR LogWorth expressed as a fraction of the number of tests. If the number of tests is *m*, the largest Overall FDR LogWorth value has Rank Fraction 1/*m*, and the smallest has Rank Fraction 1.

<Effect> PValue These columns contain *p*-values for tests of each model effect. These columns are arranged in a group called PValue in the columns panel.

<Effect> LogWorth These columns contain LogWorths for the *p*-values for tests of each model effect. These columns are arranged in a group called LogWorth in the columns panel.

<Effect> FDR LogWorth These columns contain FDR LogWorths for tests of each model effect. These columns are arranged in a group called FDR LogWorth in the columns panel.

The Y Fits data table also contains a table variable called Original Data that gives the name of the data table that was used for the analysis. If you specified a By variable, JMP creates a Y Fits table for each level of the By variable, and the Original Data variable gives the By variable and its level.

Additional Examples of Response Screening

- "Example of Tests of Practical Significance and Equivalence"
- "Example of the MaxLogWorth Option"
- "Example of Robust Fit"
- "Response Screening Personality"

Example of Tests of Practical Significance and Equivalence

This example tests for practical differences using the Probe.jmp sample data table.

1. Select **Help > Sample Data Library** and open Probe.jmp.
2. Select **Analyze > Screening > Response Screening**.

   The Response Screening Launch window appears.
3. Select the Responses column group and click **Y, Response**.
4. Select Process and click **X**.
5. Type 0.15 in the Practical Difference Portion box.
6. Click **OK**.
7. Click the Response Screening red triangle and select **Save Compare Means**.

   Figure 22.10 shows a portion of the data table. For each response in Y, the corresponding row gives information about tests of the New and the Old levels of Process.

Figure 22.10 Compare Means Table, Partial View

	Y	Level i	Level j	X	Difference	Std Err Diff	Plain Dif PValue	Practical Difference	Practical Dif PValue	Practical Equiv PValue	Practical Result
1	DELL_RPNBR	New	Old	Process	-0.031048933	0.004326425	8.044596e-13	0.1486469486	1	1.57079e-153	Equivalent
2	DELL_RPPBR	New	Old	Process	-0.024751371	0.0349025251	0.4782555718	1.1939335307	1	2.40983e-225	Equivalent
3	DELW_M1	New	Old	Process	0.0275288913	0.0205582035	0.1806001982	0.7030518329	1	1.40119e-217	Equivalent
4	DELW_M2	New	Old	Process	0.0739317751	0.0017890895	0	0.0704822866	0.0269479498	0.9730520502	Different
5	DELW_NBASE	New	Old	Process	-1.280753006	1.4836374126	0.388034756	50.689291592	1	6.62236e-223	Equivalent
6	DELW_NEMIT	New	Old	Process	-0.040133074	0.002521398	7.306034e-56	0.0848381437	1	7.74847e-69	Equivalent
7	DELW_NENBNI	New	Old	Process	-0.93842129	0.0220490008	0	0.864024229	0.0003726256	0.9996273744	Different
8	DELW_NSINK	New	Old	Process	1.3973855188	0.0254310074	0	1.0723100972	3.228438e-37	1	Different
9	DELW_PBASE	New	Old	Process	-0.043649097	0.0037311377	2.908704e-31	0.1296380428	1	8.21675e-113	Equivalent
10	DELW_PCOLL	New	Old	Process	1.8169399617	0.0085444762	0	1.17539613	0	1	Different
11	DELW_PEMIT	New	Old	Process	-0.022697392	0.0534170363	0.6709186367	1.8265075627	1	1.2499e-228	Equivalent
12	DELW_PSINK	New	Old	Process	0.911093371	0.0678503246	1.666427e-40	2.3558597461	1	3.147655e-97	Equivalent
13	DELW_RPNBR	New	Old	Process	-0.101421251	0.0100452789	9.008231e-24	0.3464883523	1	1.6441e-125	Equivalent
14	DELW_RPPBR	New	Old	Process	0.3015474189	0.0280757905	1.169725e-26	0.9698077792	1	7.48088e-120	Equivalent
15	DELW_SICR	New	Old	Process	0.0275370732	0.0176173979	0.1180928699	0.6024110643	1	7.77189e-215	Equivalent
16	M1_COMB_VGAT...	New	Old	Process	-0.532453055	0.6037284633	0.3778451214	20.662273741	1	1.83138e-223	Equivalent
17	M1_TRENCH_VG...	New	Old	Process	-0.61657228	0.1695447206	0.0002786205	5.8087989555	1	2.39165e-191	Equivalent
18	M2/M1_CAP_VG...	New	Old	Process	-0.047230999	0.0978790012	0.6294372314	3.3496974301	1	2.54456e-228	Equivalent
19	M2_COMB_BB_V...	New	Old	Process	0.0001788376	0.1270241159	0.9988767052	4.3470387459	1	3.07472e-234	Equivalent
20	M2_COMB_VGAT...	New	Old	Process	0.2971147541	0.2860872543	0.2990592705	9.7914324315	1	1.42412e-221	Equivalent
21	NISO_TUB-TREN...	New	Old	Process	7.0118092399	0.2482751793	1.58784e-164	9.0621071628	1	9.077234e-17	Equivalent
22	NISO_TUB-TUB_V...	New	Old	Process	-0.074789866	0.1081535234	0.4892691678	3.7013990674	1	8.97914e-226	Equivalent
23	PS_RPNBR	New	Old	Process	-64.69026527	15.2943887	0.0000237641	530.27422027	1	3.12381e-189	Equivalent

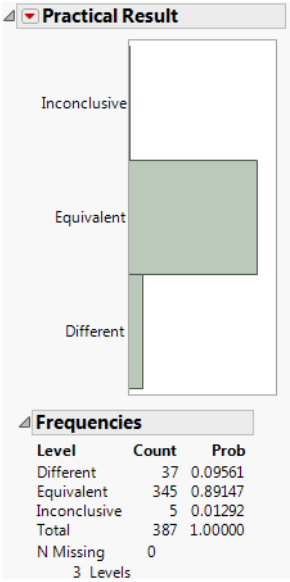
Because specification limits are not saved as column properties in Probe.jmp, JMP calculates a value of the practical difference for each response. The practical difference of 0.15 that you specified is multiplied by an estimate of the 6σ range of the response. This value is used in testing for practical difference and equivalence. It is shown in the Practical Difference column.

The Plain Difference column shows responses whose *p*-values indicate significance. The Practical Diff PValue and Practical Equiv PValue columns give the *p*-values for tests of practical difference and practical equivalence. Note that many columns show *statistically* significant differences, but do not show *practically* significant differences.

8. Display the Compare Means data table and select **Analyze > Distribution**.
9. Select Practical Result and click **Y, Columns**.
10. Click **OK**.

Figure 22.11 shows the distribution of results for practical significance. Only 37 tests are different, as determined by testing for the specified practical difference. For 5 of the responses, the tests were inconclusive. You cannot tell whether the responses result in a practical difference across Process.

Figure 22.11 Distribution of Practical Significance Results



The 37 responses can be selected for further study by clicking on the corresponding bar in the plot.

Example of the MaxLogWorth Option

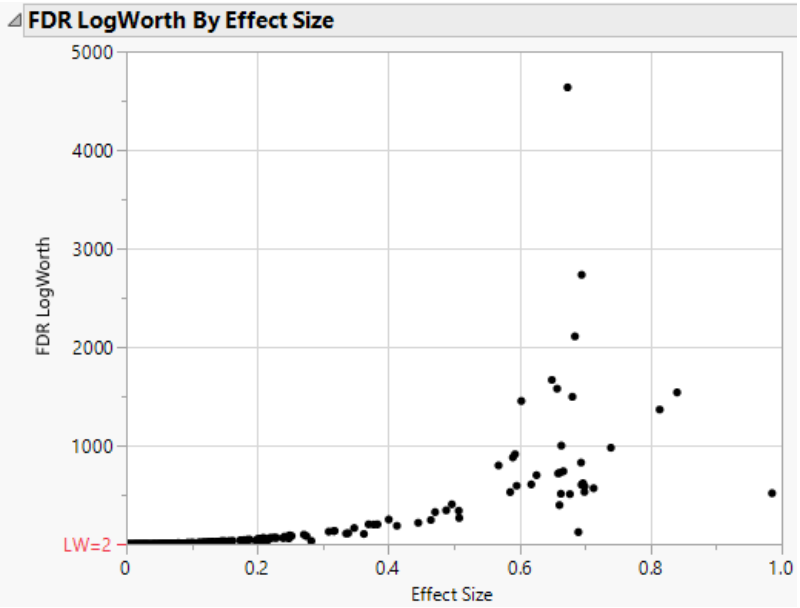
Use the Response Screening MaxLogWorth option to control the LogWorth scale from being distorted by very large values. When data sets have a large number of observations, *p*-values can be very small. LogWorth values provide a useful way to study *p*-values graphically in these cases. In some data sets the *p*-values are so small that the LogWorth scale is distorted by very large values.

1. Select **Help > Sample Data Library** and open Probe.jmp.
2. Select **Analyze > Screening > Response Screening**.
3. In the Response Screening Launch window, select the Responses column group and click **Y, Response**.
4. Select Process and click **X**.
5. Select the **Robust** check box.
6. Click **OK**.

   The analysis is numerically intensive and might take some time to complete.
7. In the Response Screening report, open the Robust FDR LogWorth by Effect Size report.

   The detail in the plot is difficult to see, because of the huge Robust FDR LogWorth value of about 58,000. To ensure that your graphs show sufficient detail, you can set a maximum value of the LogWorth.

Figure 22.12 Robust FDR LogWorth vs. Effect Size, MaxLogWorth Not Set

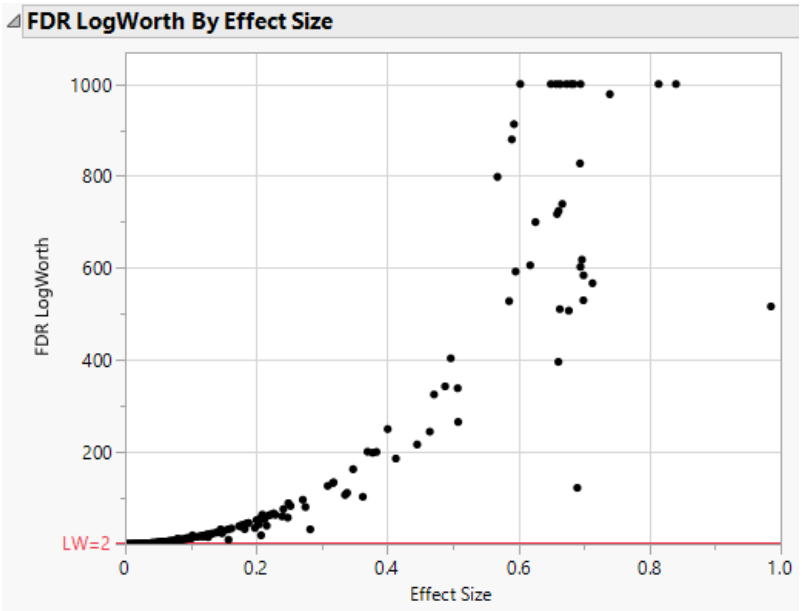


8. Repeat step 1 through step 5.
9. Type 1000 in the MaxLogWorth box at the bottom of the launch window.
10. Click **OK**.

    The analysis might take some time to complete.
11. In the Response Screening report, open the Robust FDR LogWorth by Effect Size report.

    Now the detail in the plot is apparent.

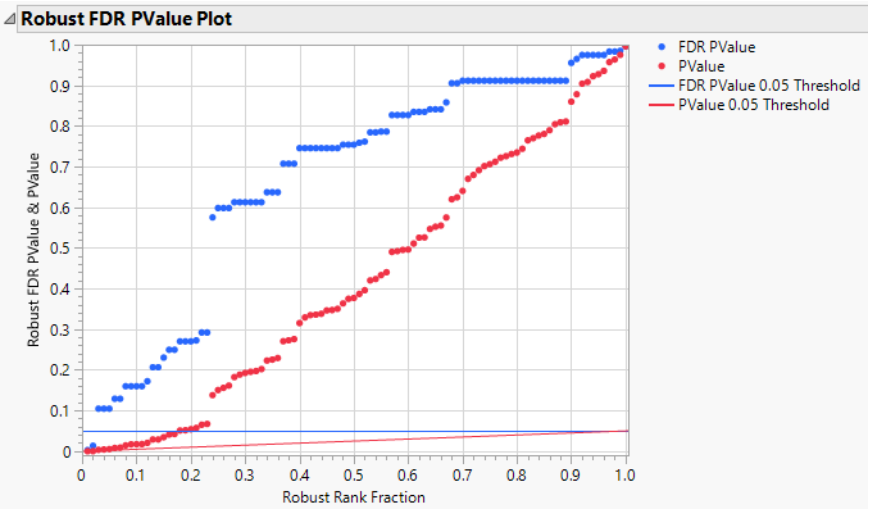
Figure 22.13 Robust FDR LogWorth vs. Effect Size, MaxLogWorth = 1000



Example of Robust Fit

1. Select **Help > Sample Data Library** and open Drosophila Aging.jmp.
2. Select **Analyze > Screening > Response Screening**.
3. Select all of the continuous columns and click **Y, Response**.
4. Select line and click **X**.
5. Check **Robust**.
6. Click **OK**.

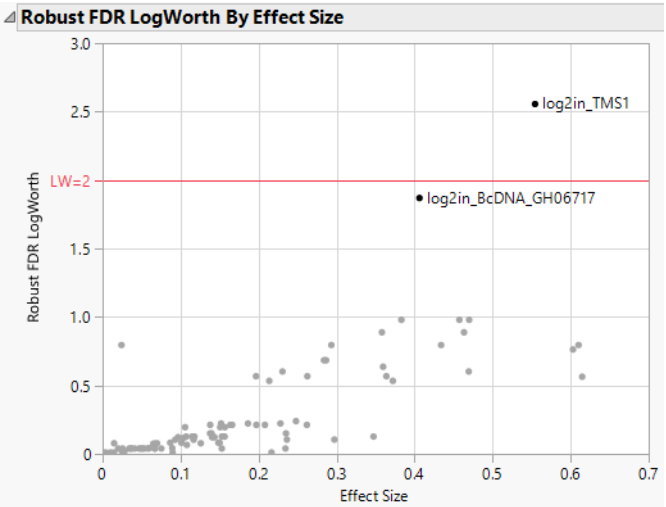
Figure 22.14 Robust FDR PValue Plot for Drosophila Data



Note that a number of tests are significant using the unadjusted robust *p*-values, as indicated by the red points that are less than 0.05. However, only two tests are significant according to the robust FDR *p*-values. These two points are more easily identified in a plot that shows FDR LogWorths.

7. Click the Robust FDR LogWorth by Effect Size disclosure icon.
8. Drag a rectangle around the point with a Robust FDR LogWorth value that exceeds 1.5.
9. In the graph, right click and select **Row Label**.

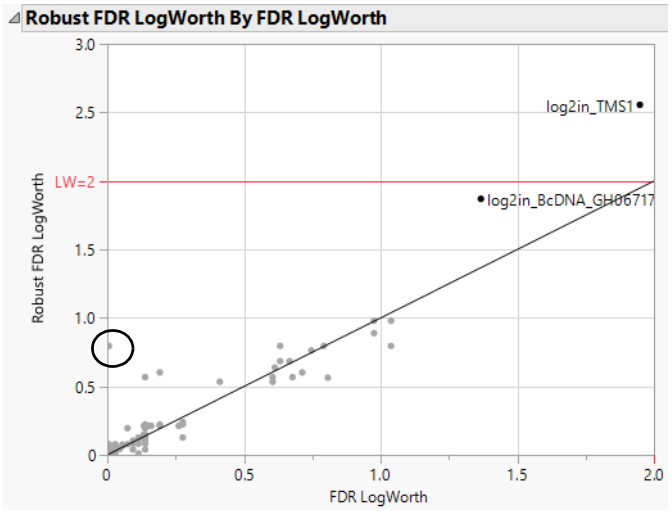
Figure 22.15 Robust LogWorth by Effect Size for Drosophila Data



Points above the red line at 2 have significance levels below 0.01 ($-\log_{10}(0.01) = 2$). A horizontal line at about 1.3 corresponds to a 0.05 significance level.

10. Click the Robust LogWorth by LogWorth disclosure icon.

Figure 22.16 Robust LogWorth by LogWorth for Drosophila Data



If the robust test for a response were identical to the usual test, its corresponding point would fall on the diagonal line in Figure 22.16. The circled point in the plot does not fall near the line, because it has a Robust LogWorth value that exceeds its LogWorth value.

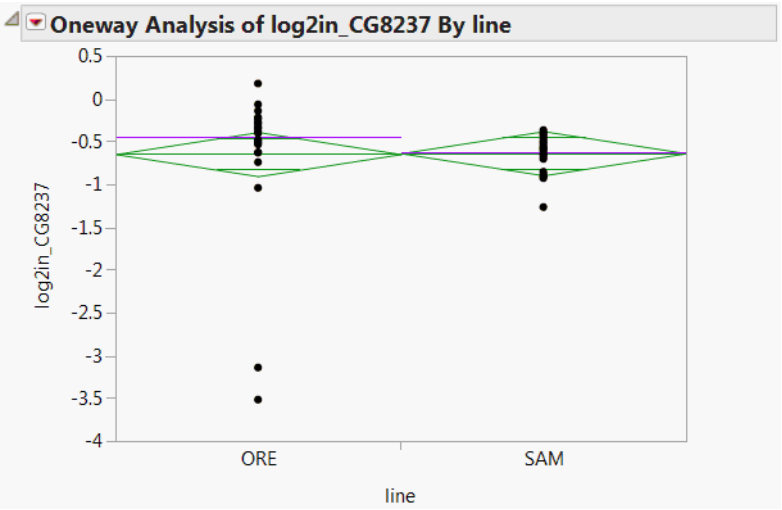
11. Drag a rectangle around this point in the plot.
12. Find the row for this point in the Result table.

    Note that the response, log2in_CG8237 has PValue 0.9568 and Robust PValue 0.0176.

13. Click the Response Screening red triangle and select **Fit Selected Items**.

    A Fit Selected Items report is displayed containing a Oneway Analysis for the response log2in_CG8237. The plot shows two outliers for the ORE line (Figure 22.17). These outliers indicate why the robust test and the usual test give disparate results. The outliers inflate the error variance for the non-robust test, which makes it more difficult to see a significant effect. In contrast, the robust fit down-weights these outliers, thereby reducing their contribution to the error variance.

Figure 22.17 Oneway Analysis for log2in_CG8237



Response Screening Personality

The Response Screening personality in Fit Model enables you to study tests of multiple responses against linear model effects.

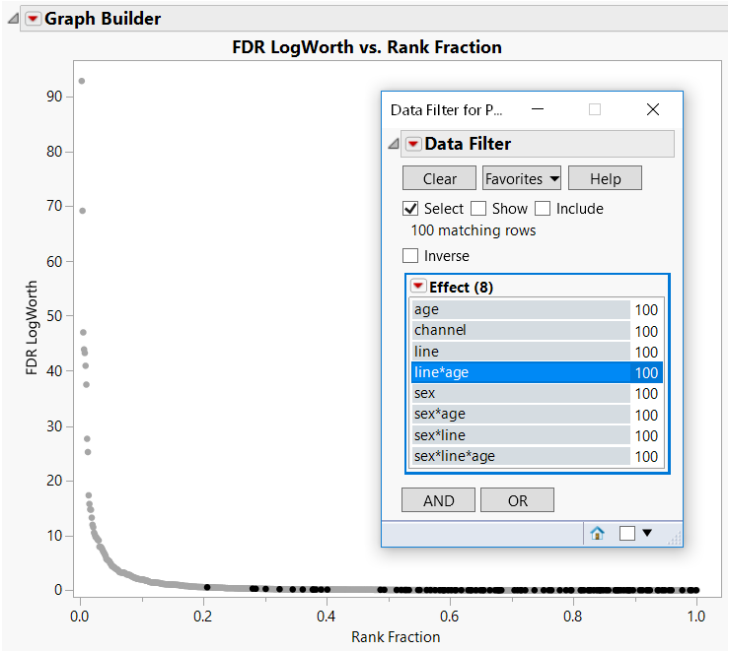
1. Select **Help > Sample Data Library** and open Drosophila Aging.jmp.
2. Select **Analyze > Fit Model**.
3. Select all the continuous columns and click **Y**.
4. Select channel and click **Add**.
5. Select sex, line, and age and select **Macros > Full Factorial**.
6. Select **Response Screening** from the Personality list.
7. Click **Run**.

   The Fit Response Screening report appears. Two data tables are also presented: Y Fits summarizes the overall model tests, and PValues tests the individual effects in the model for each Y.

   To get a general idea of which effects are important, do the following:
8. Run the FDR LogWorth by Rank Fraction script in the PValues data table.
9. Select **Rows > Data Filter**.
10. In the Data Filter window, select Effect and click **Add**.
11. In the Data Filter, click through the list of the model effects while you view the selected points in the FDR LogWorth by Rank Fraction plot.

Keep in mind that values of LogWorth that exceed 2 are significant at the 0.01 level. The Data Filter helps you see that, with the exception of sex and channel, the model effects are rarely significant at the 0.01 level. Figure 22.18 shows a reference line at 2. The points for tests of the line*age interaction effect are selected. None of these are significant at the 0.01 level.

Figure 22.18 FDR LogWorth vs Rank Fraction Plot with line*age Tests Selected



Statistical Details for the Response Screening Platform

The False Discovery Rate

All of the Response Screening plots involve p-values for tests conducted using the FDR technique described in Benjamini and Hochberg (1995). See also Westfall et al. (2011). This method assumes that the p-values are independent and uniformly distributed.

JMP uses the following procedure to control the false discovery rate at level α:

1. Conduct the m hypothesis tests of interest to obtain p-values $p_1, p_2, \ldots, p_m$.
2. Rank the p-values from smallest to largest. Denote these by $p_{(1)} \leq p_{(2)} \leq \ldots \leq p_{(m)}$.

3. Find the largest p-value for which $p_{(i)} \leq (i/m)\alpha$. Suppose this first p-value is the k^{th} largest, $p_{(k)}$.
4. Reject the k hypotheses associated with p-values less than or equal to $p_{(k)}$.

This procedure ensures that the false discovery rate does not exceed α.

The p-values adjusted for the false discovery rate, denoted $p_{(i),\,FDR}$, are computed as follows:

$$p_{(i),\,FDR} = \begin{cases} p_{(m)} & \text{for } i = m \\ min\left[p_{(i+1),\,FDR}, \left(\frac{m}{i}\right)p_{(i)}\right] & \text{for } i = m-1, \ldots, 1 \end{cases}$$

If a hypothesis has an FDR-adjusted p-value that falls below α, then it is rejected by the procedure.

Chapter 23

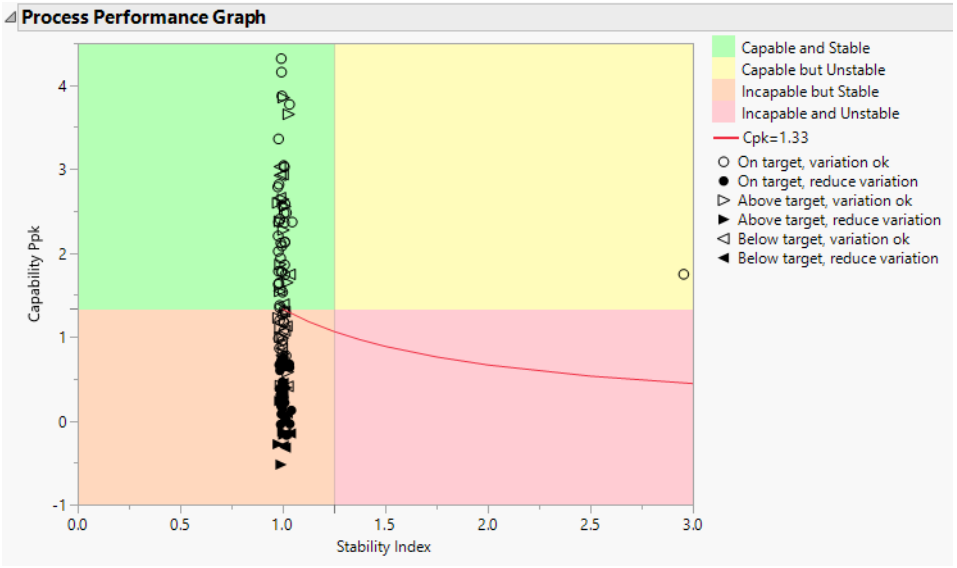
Process Screening

Screen Many Processes for Stability and Capability

Use the Process Screening platform for exploring a large number of processes across time. The platform calculates process stability and process capability metrics. The platform creates control charts and detects large process shifts. The platform is intended to expedite the evaluation of a very large number of processes by enabling you to quickly focus on the processes that are unstable, not capable of meeting specification limits, or subject to shifts in the mean.

Based on your initial results, you can choose to explore specific processes graphically or in greater analytical depth. You can easily access the Control Chart Builder and Process Capability platforms. You can save detailed results for all of your processes or for specific processes.

Figure 23.1 Example of a Process Performance Graph



Contents

Overview of the Process Screening Platform 483
Example of Process Screening 483
Launch the Process Screening Platform 486
Launch Window Roles 486
Launch Window Options 487
Limits Table 489
The Process Screening Report 491
Process Screening Platform Options 497
Process Performance Graph 502
Additional Example of Process Screening 504
Statistical Details for the Process Screening Platform 506
Scaling Factors for Using Medians to Estimate Sigma 506

Overview of the Process Screening Platform

The Process Screening platform facilitates the task of assessing data from a large number of processes for stability and capability. The results are largely based on control chart calculations to determine when a process is out of control. The Process Screening platform enables you to do the following:

- Specify a constant subgroup size or use a variable containing subgroup identifiers for control chart calculations using subgroups.
- Use grouping variables. For each combination of values of the group variables, an analysis is provided for each process variable.
- Use medians to make your centerline and sigma calculations robust to outliers.
- Obtain information about the location of large shifts in the mean of your processes.

You can customize the Summary report to show specific control chart tests, including tests for changes in process mean and spread. The report also provides capability information when you supply specification limits. The Process Performance Graph gives you a visual representation of the performance of your processes in terms of stability and capability. The Shift Graph shows locations of upshifts and downshifts.

Tip: For information about adding specification limits, see *Using JMP*.

Process Screening makes it easy to select specific processes for further analysis. The platform provides small run charts for these processes - the size of the plots makes it easy for you to view a substantial number at a time. You can also link to Control Chart Builder and the Process Capability platform for analyses of select processes.

You can save data tables containing results in various forms, either for your entire set of processes or only for select processes.

Example of Process Screening

The Semiconductor Capability.jmp sample data table contains 128 columns of process measurements. Each column contains 1,455 measurements. You are interested in identifying unstable processes. Also, each column contains a Spec Limits column property. If a process is stable, it is appropriate to calculate its process capability. You proceed to assess both stability and capability for this data table.

1. Select **Help > Sample Data Library** and open Semiconductor Capability.jmp.
2. Select **Analyze > Screening > Process Screening**.
3. Select the Processes column group and click **Process Variables**.

Notice that the Control Chart Type is set to Indiv and MR.

4. Click **OK**.

Figure 23.2 Partial View of Initial Report

Process Screening

Indiv and MR

	Variability			Summary		Control Chart Alarms			Capability				
Column	Stability Index	Within Sigma	Overall Sigma	Mean	Count	Alarm Rate	Test1	Latest Alarm	Ppk	Cpk	Out of Spec Count	Out of Spec Rate	Latest Out of Spec
PMS1	2.96	6.4e-15	1.9e-14	-50	1455	0.03643	53	3	1.746	5.159	0	0	.
IVP8	1.05	0.45226	0.47293	-21.69	1455	0.00550	8	49	2.369	2.477	0	0	.
B1	1.04	5.70202	5.94318	125.972	1455	0.00275	4	43	0.124	0.130	604	0.4151	1
IVP7	1.04	5.92093	6.14904	-79.865	1455	0.00275	4	114	-0.151	-0.157	1032	0.7093	1
SIT2	1.03	3.59488	3.71963	41.1722	1455	0.00550	8	21	1.745	1.806	0	0	.
CAP	1.03	0.92619	0.95805	38.1075	1455	0.00412	6	12	-0.037	-0.039	1149	0.7897	1
VPM7	1.03	3.20356	3.3107	85.321	1455	0.00550	8	107	0.634	0.655	54	0.0371	13
NPN10	1.03	1.76238	1.82043	73.5168	1455	0.00481	7	38	3.771	3.895	0	0	.
INV2	1.03	0.6307	0.65112	17.3705	1455	0.00481	7	11	0.676	0.697	33	0.0227	25
NPN9	1.03	2.40507	2.47045	47.8068	1455	0.00344	5	9	3.653	3.752	0	0	.
IVP9	1.03	1.61702	1.65762	-30.721	1455	0.00344	5	137	0.410	0.420	145	0.0997	1
VTN210	1.02	2.30557	2.35977	0.09116	1455	0.00344	5	9	0.582	0.595	55	0.0378	6
SIT1	1.02	15.3977	15.7506	149.659	1455	0.00481	7	16	0.088	0.090	581	0.3993	2
INM1	1.02	3.28224	3.34868	82.4373	1455	0.00412	6	3	1.649	1.682	0	0	.
IVP3	1.02	3.08312	3.14263	-49.493	1455	0.00206	3	3	0.395	0.403	168	0.1155	4
VTP210	1.02	3.31312	3.3769	1.29622	1455	0.00344	5	25	1.128	1.150	1	0.0007	1052
PBL1	1.02	0.26689	0.27188	2.71456	1455	0.00481	7	65	0.669	0.681	31	0.0213	53
NPN6	1.02	1.12479	1.14543	43.2968	1455	0.00206	3	69	-0.173	-0.176	1111	0.7636	2
E2A1	1.02	0.00132	0.00134	0.62966	1455	0.00344	5	21	2.478	2.522	0	0	.
VDP1	1.02	0.23752	0.24127	28.8111	1455	0.00137	2	17	0.775	0.787	15	0.0103	27
M1_M1	1.02	0.20527	0.20846	0.23703	1455	0.00412	6	26	-0.149	-0.151	1007	0.6921	1
A2N	1.01	8.53741	8.66111	56.0799	1455	0.00344	5	25	0.019	0.019	925	0.6357	1

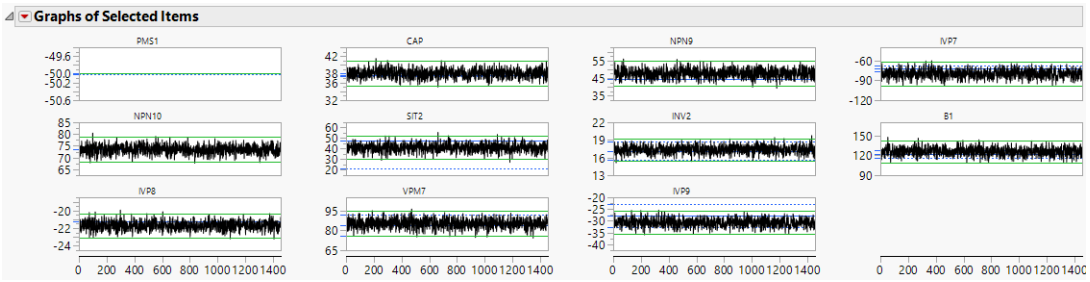
The Process Screening window appears, showing a table of results for each process. The table is sorted by Stability Index. This is a measure of the stability of a process, where a stable process has a stability index near 1. Higher values of the stability index indicate a less stable process. (The sorting is indicated by the caret beside Stability Index in the report.) You want to take a closer look at processes with a stability index value greater than or equal to 1.03.

5. In the report window, select processes PMS1 through IVP9.

   Each of these first 11 processes has a value of 1.03 or larger in the Stability Index column.

6. Right-click the selected processes and select **Quick Graph for Selected Items**.

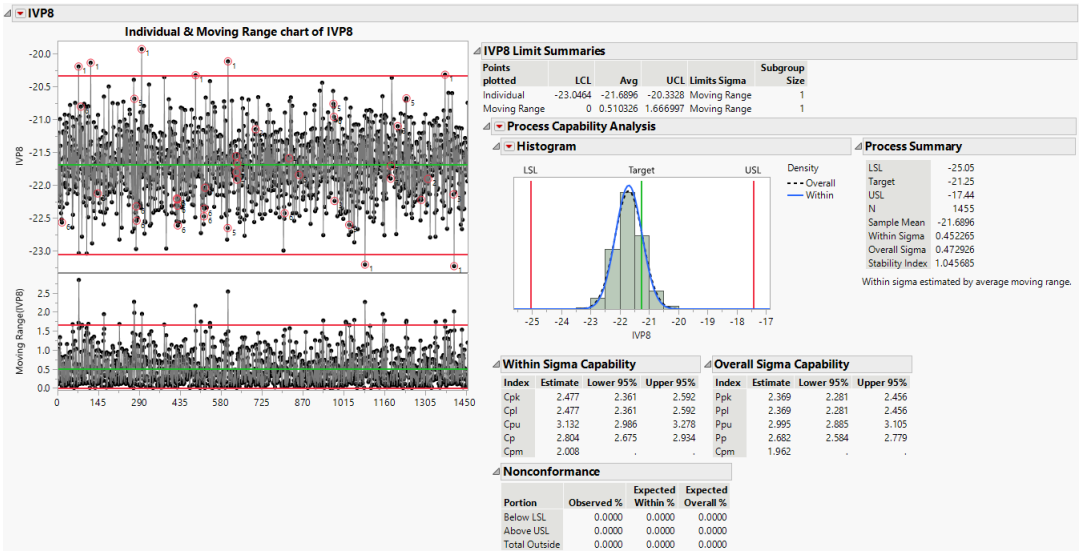
Figure 23.3 Quick Graphs for Highest Alarm Rate Processes



You decide to take a closer look at IVP8 (row 3, column 1 in the Graphs of Selected Items).

7. Select the second process in the Summary table, which corresponds to IVP8.
8. Right-click the selected process and select **Control Charts for Selected Items**.

Figure 23.4 Control Chart Builder Report for IVP8

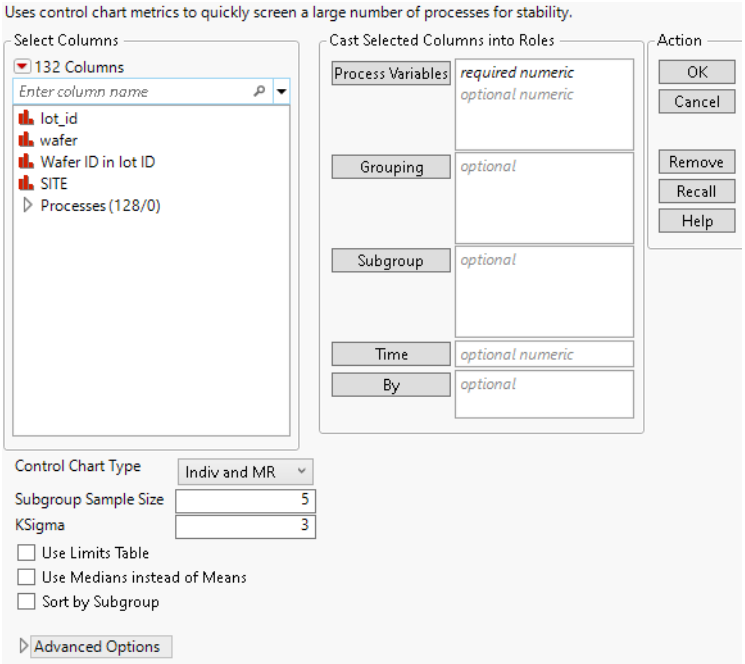


A Control Chart Builder report appears. Because IVP8 has a Spec Limits column property, the report also includes a capability analysis.

Launch the Process Screening Platform

Launch the Process Screening platform by selecting **Analyze > Screening > Process Screening**.

Figure 23.5 Process Screening Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Launch Window Roles

Process Variables The columns of process data containing the measurements to be analyzed. The columns must have a Numeric data type. If a process variable has a Control Limits column property, the Process Screening platform uses those limits to calculate the Specified Sigma.

Note: The platform does not support control limits that are specified in a Control Limits column property for dispersion (R) charts.

Grouping Columns assigned as grouping variables. Each process variable is analyzed at each combination of levels of the grouping variables. The results are presented in a single report.

Subgroup Assigns one or more subgroup variables. If more than one subgroup variable is assigned, the subgroup levels are defined as a combination of the multiple columns.

Note: The Subgroup role is ignored for Indiv and MR charts.

Time A numeric column whose values are used for the time order for the data. Use the Time role for data that are time-stamped. The time stamp is used for the time axis in quick graphs and shift graphs. The process data are sorted by the Time variable before calculations are performed.

By A column whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate tables and reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Launch Window Options

Control Chart Type Select one of five control chart types: Indiv and MR (Individual Measurement and Moving Range); XBar and R; XBar and S; XBar, MR, and R; or XBar, MR, and S. The XBar, MR, and R control chart and the XBar, MR, and S control chart are also referred to as three way control charts. For more information about statistical details, see *Quality and Process Methods*.

Note: If the subgroup size for a process is 1, the chart automatically switches to an Indiv and MR chart.

Subgroup Sample Size Specifies a constant sample size for subgroups. The minimum subgroup size is 2. A subgroup size of 5 is the default. The Subgroup Sample Size specification is ignored for Indiv and MR charts or when a subgroup variable is specified.

KSigma Specifies the sigma multiplier. KSigma is the value that is multiplied by sigma in the calculation of control limits. By default, KSigma is 3.

Use Limits Table Enables you to import historical control limits and specification limits from a data table. When you select this option and click OK in the launch window, a Choose limits table window appears. Once you choose a limits tables and click OK, a Limits Specification window appears. Assign columns in your limits table to appropriate roles and click OK. See "Limits Table".

Note: If you do not select this option, limits are obtained from the Control Limit and Specification Limit column properties for the Process Variables. If you do not select this option and the Process Variables do not have control limit column properties, the control limits are calculated from the data.

Use Medians instead of Means Estimates the center line using the median of the observations. Sigma is estimated using scaling factors obtained using Monte Carlo simulation. The table of factors is given in "Statistical Details for the Process Screening Platform". The calculation depends on the type of chart selected:

- For XBar and R chart or Indiv and MR chart calculations, sigma is estimated using the scaled median of the ranges.
- For XBar and S chart calculations, sigma is estimated using the scaled median of the standard deviations.
- For unequal subgroup sizes, the scaling factor corresponds to the average subgroup size rounded to the nearest integer.

When one or more outliers influence the location of the center line, many subgroups can appear out of control. Using the median alleviates this problem.

Note: When Use Medians instead of Means is selected, the results obtained from the Control Charts for Selected Items or the Process Capability for Selected Items red triangle menu options do not match the Process Screening results.

Sort by Subgroup Sorts the process data by the subgroup variable, or combination of nested subgroup variables, before calculations are performed.

Advanced Options Contains the following options:

Shift Threshold Specify a value that controls the sensitivity of the Shift Graph. Shift Threshold is set to three by default. After outlier removal, the Shift Graph shows a plot of the time occurrence of all process shifts that exceed the number of within-sigma units specified by the Shift Threshold. See "Shift Graph".

Outlier Threshold Specify a value that controls the sensitivity of outlier removal for detection of large recent shifts and for the Shift Graph. Outlier Threshold is set to five by default. If the number of within sigma units from an observation to both of its neighboring observations exceeds the specified Outlier Threshold, that observation is replaced with a value that is one within-sigma unit away from its closest neighboring observation. See "Shift Magnitudes and Positions".

Shift Lambda Enables you to change the exponentially weighted moving average (EWMA) weight used in the Shift Graph. See "Shift Magnitudes and Positions".

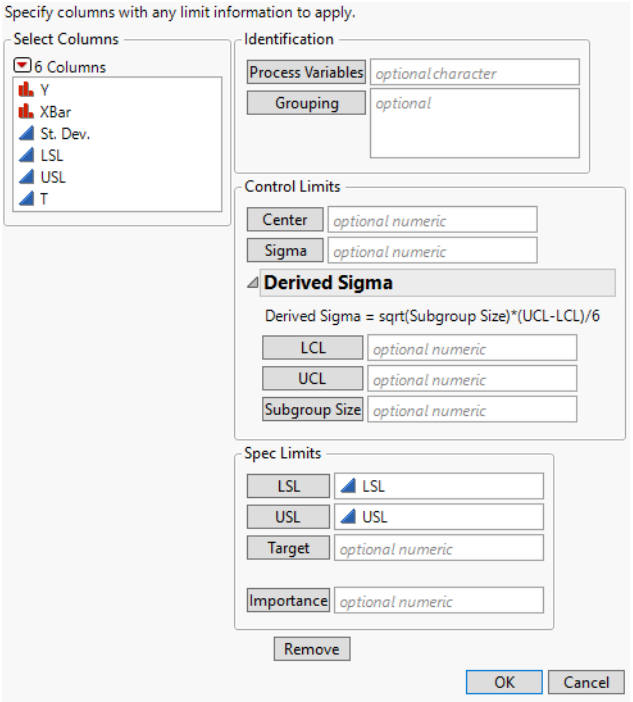
Drift Beta The weight used in the exponentially weighted moving average (EWMA) for drift detection. Drift Beta is set to 0.05 by default.

Minimum Process Length The minimum number of data values that a process must have in order to be included in the analysis. By default, this value is set to 3.

Limits Table

A Limits Table contains a row for each process defined by the Process Variables and Grouping variables in your table of process data. When you use a Limits Table, the Limits Specifications window enables you to specify variables with the roles listed below. You do not need to specify variables for all of these roles. All of these roles are optional.

Figure 23.6 Limits Specifications Window



Columns in the Limits Table that have appropriate names or names that match the role buttons are auto-filled. For example, any column called “Process”, “Column”, or “Parameter” is auto-filled into the Process Variables list.

If you have control limits but do not have columns for Center or Sigma, then you can use the Derived Sigma options.

Process Variables A column that contains values corresponding to the column names in your table of process data.

Grouping One or more columns that contain the values of the grouping variables for your table of process data.

Center A column containing values for the center line for each process. This is usually the historical process mean.

Sigma A column containing values for the within standard deviation for each process. This is usually the historical standard deviation. This is labeled as Specified Sigma in the Process Screening report.

Derived Sigma Calculates sigma based on the given control limits and subgroup size. This is labeled as Specified Sigma in the Process Screening report.

Note: If there is a specified sigma, the within-sigma, stability index, and stability ratio values do not match between the Process Screening and Control Chart Builder platforms. The Process Screening platform uses the specified sigma to calculate the control limits, but still calculates the within-sigma using the data. The Control Chart Builder platform uses the specified sigma as the historical sigma and in control charts, the historical sigma is used as the within-sigma.

The derived sigma is calculated as follows:

$$\text{Sigma} = d^*(UCL - LCL)/6$$

where d is the square root of the subgroup size.

LCL A column containing the lower control limits for each process.

UCL A column containing the upper control limits for each process.

Subgroup Size A column containing the subgroup size for each process.

LSL A column containing lower specification limits for each process.

USL A column containing upper specification limits for each process.

Target A column containing a target value for each process.

Importance A column containing an importance value for each process. The Importance column provides a mechanism to sort processes in the order that you prefer.

The Process Screening Report

The Process Screening report opens with a Summary table that contains results about process stability. There is a label above the summary table that shows the type of control chart on which the results are based. The sample size or subgroup variable is also shown above the summary table for every chart type except for Indiv and MR. The summary table also contains capability results if you have provided specification limits. The processes and groups are initially sorted in decreasing order by Stability Index. If Importance values are specified by the user, the processes and groups are sorted in decreasing order by Stability Index within Importance. The columns for Stability Index, Ppk, Cpk, Cp, and Target Index are colored as green, yellow, and red to indicate adequate, marginal, and poor stability or capability, respectively. This color coding scheme matches the Process Performance graph color coding scheme.

Tip: To sort the report by a column, click the column name. A caret appears to the right of the column name. The direction of the caret indicates whether the sorting is descending or ascending. To change the order of the sorting, click the column name again.

The control chart calculations in the Summary table include Nelson tests and a Range Limit Exceeded test. These tests assume the following about the control chart limits:

- The center line for the XBar or X control charts is given by the mean of all measurements. If you use the Medians instead of Means option, the center line is given by the median of the observations.
- Control limits are placed at *K* sigma units from the center line. Use the KSigma option in the launch window to specify *K*. By default, KSigma is 3.
- Sigma is estimated using the conventions that correspond to the control chart type that you specified or, if you use Medians instead of Means, as described in "Use Medians instead of Means".

Tip: The eight Nelson tests in the Process Screening platform follow the test settings in the Control Chart Builder platform preferences. You can customize the tests at File > Preferences > Platforms > Control Chart Builder.

The Summary table can contain the following information:

Column The columns that you entered as Process. There is a row for each distinct combination of Process and Grouping columns. This column is suppressed if there is only one process column.

Tip: To access options that operate on selected items, right-click the **Column** column.

Grouping Columns There is a report column for each column in the data table that you entered as Grouping. The levels of the Grouping columns are listed so that there is a unique row in the report table for each distinct combination of Process name and Grouping columns values.

Importance (Appears only when there are importance values specified by the user, as inputs from a limits table.) The user-specified importance value for the processes.

Variability Contains the following columns:

Stability Index A measure of stability of the process. A stable process has a stability index near one. Higher values indicate less stability. The stability index is defined as follows:

(Overall Sigma / Within Sigma)

If a three way control chart is selected in the launch, the stability index is defined as follows:

(Overall Sigma / Between-and-Within Sigma)

Stability Ratio A measure of stability of the process. A stable process has a stability ratio near one. Higher values indicate less stability. The stability ratio is defined as follows:

$(\text{Overall Sigma} / \text{Within Sigma})^2$

If a three way control chart is selected in the launch, the stability ratio is defined as follows:

$(\text{Overall Sigma} / \text{Between-and-Within Sigma})^2$

Within Sigma An estimate of the standard deviation based on within subgroup variation. The estimate is based on the control chart type that you specified, and is a short-term measure of variation. See *Quality and Process Methods* for statistical details. If you select Medians instead of Means, Within Sigma is computed as described in "Use Medians instead of Means".

Overall Sigma The usual estimate of standard deviation based on all observations.

Between Sigma (Appears only when three way control chart is selected in the launch window.) An estimate of the standard deviation based on the variation between subgroups. See *Quality and Process Methods* for statistical details.

Between-and-Within Sigma (Appears only when three way control chart is selected in the launch window.) An estimate of the standard deviation based on the variation between subgroups and the variation within subgroups. The Between-and-Within Sigma estimate is defined as follows:

$$\text{Between-and-Within Sigma} = \sqrt{\text{Within Sigma}^2 + \text{Between Sigma}^2}$$

Specified Sigma The standard deviation specified by a sigma control limit in the Limits Specification dialog or an estimate of the standard deviation derived from the control limits and subgroup size. See "Derived Sigma". The control limits and subgroup size can be specified using a limits table or a Control Limits column property.

Summary Contains the following columns:

Centerline (Appears if you do one of the following: select Use Medians instead of Means in the launch window, import a Center value using a limits table, or import control limits to derive sigma.) The value listed under Centerline is used in control chart calculations as the center line.

- If you select Use Medians instead of Means in the launch window, the overall median of the observations is displayed.
- If you import a Center value from a limits table, that value is displayed.
- If you import control limits, the value displayed is calculated as (UCL + LCL)/2.

Mean The average of all observations.

Count The number of observations.

N Subgroups The number of subgroups.

Control Chart Alarms Contains information about the subgroups that result in alarms for a variety of tests, including each of the 8 Western Electric rules. The standard deviation estimate is the Within Sigma value. By default, only the Alarm Rate, Test 1, and Latest Alarm columns are shown in the Summary table.

Alarm Rate The number of subgroups that resulted in alarms for any of the tests selected under the Choose Test option (Any Alarm) divided by the number of non-missing subgroups (Subgroups).

Any Alarm (Appears only when more than one Test column is shown.) The number of subgroups that trigger alarms for any of the tests selected under the Choose Test option. These are the eight Nelson tests and the test for Range Limit Exceeded.

Tip: The eight Nelson tests in the Process Screening platform follow the test settings in the Control Chart Builder platform preferences. You can customize the tests at File > Preferences > Platforms > Control Chart Builder.

Test 1 One point is more than three standard deviations from the center line. The subgroup associated with that point triggers the alarm.

Test 2 Nine or more consecutive points are on the same side of the center line. The subgroup associated with the ninth point triggers the alarm.

Test 3 Six or more consecutive points are continually increasing or decreasing. The subgroup associated with the sixth point triggers the alarm.

Test 4 Fourteen consecutive points alternate in direction: increasing and then decreasing or decreasing and then increasing. The subgroup associated with the 14th point triggers the alarm.

Test 5 Two out of three consecutive points on the same side of the center line are more than two standard deviations from the center line. The subgroup associated with the second point that exceeds two standard deviations triggers the alarm.

Test 6 Four out of five consecutive points on the same side of the center line are more than one standard deviation from the center line. The subgroup associated with the fourth point that exceeds one standard deviation triggers the alarm.

Test 7 Fifteen consecutive points, on either side of the center line, are all within one standard deviation of the center line. The subgroup associated with the 15th point triggers the alarm.

Test 8 Eight consecutive points, on either side of the center line, all fall beyond one standard deviation of the center line. The subgroup associated with the eighth point triggers the alarm.

Range Limit Exceeded The number of subgroups that exceed the upper control limit on the R, S, or MR chart calculation.

Moving Range Limit Exceeded The number of subgroups that exceed the moving range limit on the three way control chart calculation.

Latest Alarm The position of the subgroup, counting from the last subgroup, that signaled the most recent alarm for any of the Nelson or Range Limit Exceeded tests.

Capability (Appears only when there are Spec Limits specified for some processes.) Contains the following options:

Ppk Capability index based on Overall Sigma and assuming a normal distribution. See *Quality and Process Methods* for statistical details. On by default.

Cpk Capability index based on Within Sigma or Between-and-Within Sigma and assuming a normal distribution. See *Quality and Process Methods* for statistical details. On by default.

Cp The potential capability if target and drift issues are resolved. See *Quality and Process Methods* for details.

Target Index The number of short-term standard deviations that the process average differs from the target value. This measures the ability of the process to hit the target value. The Target Index is calculated as 3(Cp - Cpk). A target index is considered poor if above 1, marginal if between 0.5 and 1, and adequate if less than 0.5. See White et al. (2018).

Out of Spec Count The number of observations that fall outside the specification limits. On by default.

Out of Spec Rate The proportion of observations that fall outside the specification limits. On by default.

Expected Out of Spec Rate The expected proportion of observations that fall outside of the specification limits. The Expected Out of Spec Rate assumes a stable and normally distributed process and uses overall sigma.

Latest Out of Spec The number of observations, counting from the last observation to the most recent observation that falls outside the specification limits. On by default.

(Mean-Tgt)/SpecRange The spec centered mean. This is the same as the Mean Shift Standardized to Spec in Process Capability. See *Quality and Process Methods* for statistical details.

StdDev/SpecRange The spec scaled standard deviation. This is the same as the Std Deviation Standardized to Spec in Process Capability. See *Quality and Process Methods* for statistical details.

LSL The lower specification limits.

USL The upper specification limits.

Target The target value.

Shift Magnitudes and Positions (Shown only if you have selected a Shift Detection option from the Process Screening red triangle menu.) Shift detection is performed to identify shifts that exceed one within-sigma unit. The algorithm uses outlier-correction and an EWMA smoothing approach for the individual observations. This is the algorithm:

- Outliers are removed so that single outliers do not indicate shifts. The value specified as Outlier Threshold (five by default) on the launch window controls the sensitivity of outlier removal. If the number of within-sigma units from an observation to both of its neighboring observations exceeds the specified Outlier Threshold, that observation is replaced with a value that is one within-sigma unit away from its closest neighboring observation.
- An EWMA fit is constructed for the subgroup means in forward time order and another EWMA fit is constructed for the subgroup means in reverse time order. The EWMA fits have lambda equal to 0.3.

- The largest positive and negative differences between successive EWMA values that exceed one within-sigma unit are identified.
- The absolute values of these differences, divided by the within estimate of sigma, are the values reported as Largest Upshift and Largest Downshift.
- The locations of the first subgroups involved in these largest shifts define the Upshift Position and Downshift Position.

Largest Upshift The magnitude of the largest upward shift that exceeds one within-sigma unit, reported in within-sigma units.

Upshift Position or Upshift <Time Variable> The position of the subgroup having the largest Upshift. If you specify a Time variable, the column in the Summary table is named Upshift <Time Variable> and the position of the shift is given in terms of the Time variable.

Largest Downshift The magnitude of the largest downward shift that exceeds one within-sigma unit, reported in within-sigma units.

Downshift Position or Downshift <Time Variable> The position of the subgroup having the largest Downshift. If you specify a Time variable, the column in the Summary table is named Downshift <Time Variable> and the position of the shift is given in terms of the Time variable.

Drift Magnitudes and Positions (Shown only if you have selected the Drift Summaries option from the Process Screening red triangle menu.) Drift detection is performed to detect smaller, more gradual changes in processes. The algorithm is identical to the one used in shift detection, except that drift detection uses a Holt Double-Exponential Smoother instead of an EWMA. This is the algorithm:

- Outliers are removed through the same process that is used in shift detection. See "Shift Magnitudes and Positions".
- The drift detection algorithm fits a Holt Double-Exponential Smoothing model for the subgroup means in forward time order and fits another Holt Double-Exponential Smoothing model for the subgroup means in reverse time order. The two smoothing model fits each have two smoothing constants: α for the level and β for the slope. The β smoothing constant is set at 0.05 and the α smoothing constant is estimated to minimize the error.
- The means of the positive drift values, negative drift values, and absolute drift values are reported as Mean Up Drift, Mean Down Drift, and Mean Abs Drift, respectively.

Mean Up Drift The sum of the positive drift values divided by the count.

Mean Down Drift The sum of the negative drift values divided by the count.

Mean Abs Drift The sum of the absolute value of all drift values divided by the count.

Note: Process Screening does not utilize the Distribution or Process Capability Distribution column properties and always performs capability analyses assuming a normal distribution. If the distribution type for one of these column properties is non-normal, a Warnings outline is shown below the Summary table.

Process Screening Platform Options

The Process Screening red triangle menu contains options to customize the display and to save calculated statistics. The options that operate on selected items are also accessible by right-clicking the **Column** column in the Summary table.

Summary Shows or hides the Summary table. See "The Process Screening Report".

Find and Select Enter search strings for columns that you entered as Process Variables or Grouping in the launch window. A panel appears for each column. The corresponding processes are selected in the Summary table.

Select Where Opens the Select Where window. You can select specific processes in the Summary table that correspond to a particular condition by using the Comparison menu and Value text box. For example, you can select all processes such that Ppk < 1.33. After you click OK, the processes are selected in the Summary table.

Tip: You can also access the Select Where window by right-clicking anywhere in the Summary table.

Show Summary Columns Enables you to choose some of the summary columns that are shown in the Summary table.

Quick Graph for Selected Items Plots small graphs of the processes that you select in the Summary table in a Graphs of Selected Items report. The report makes it possible to view and compare many processes at once. The plots are ordered according to their order of entry on the launch window. Each quick graph also has lines for the spec limits. There is a solid blue line for the target value and dotted blue lines for the LSL and USL values. To remove the quick graphs, click the Graphs of Selected Items red triangle and select Remove.

Tip: To change the layout of the quick graphs, click the Graphs of Selected Items red triangle and select Number of Plots Across.

Control Charts for Selected Items Opens a Control Chart Builder report window for the processes that you selected in the Summary table. The control chart corresponds to your selections in the Process Screening launch window.

Notes:

- Only the tests selected in Process Screening are sent to Control Chart Builder. Tests that have been turned off are not sent.
- If you have multiple subgroup variables where the inner subgroup variable is not sorted within the outer subgroup variable, a new subgroup variable column is created and used when Control Charts for Selected Items is launched. This does not apply if Sort by Subgroup is selected in the Process Screening launch window.

Process Capability for Selected Items Opens a Process Capability report window showing Individual Detail Reports for the processes that you select in the Summary table. If you select a process for which specification limits are not specified, a Spec Limits window appears. In this window, you can specify specification limits by selecting a data table or entering values directly.

The Process Capability analysis assumes normal distributions and uses within sigma values that correspond to your Control Chart Type selection in the Process Screening launch window:

- Moving range for Indiv and MR
- Average of ranges for XBar and R
- Average of unbiased standard deviations for XBar and S

Notes:

- If you specify sigma using a sigma control limit or the Derived Sigma option in the Limits Specifications window, this value is not used in Process Capability. This is because the Process Screening and Process Capability platforms use these control limits differently.
- If you have multiple subgroup variables where the inner subgroup variable is not sorted within the outer subgroup variable, a new subgroup variable column is created and used when Process Capability for Selected Items is launched. This does not apply if Sort by Subgroup is selected in the Process Screening launch window.

Color Selected Items Applies a color of your choosing to the values in the selected rows of the Summary table.

Remove Selected Items Removes the rows selected in the Summary table and reruns the analysis without those processes.

Show Tests Shows or hides test results for Nelson tests that are selected under the Choose Tests option in the Process Screening report's summary table.

Choose Tests Enables you to choose the tests that you want to include in the calculation of Alarm Rate and Any Alarm.

Tip: To select multiple tests, press Alt and click the Process Screening red triangle to open a menu of all platform options.

Shift Detection Provides options for detecting shifts after outlier removal. See "Shift Magnitudes and Positions".

Largest Upshift Adds columns for Largest Upshift and Upshift Position to the Summary table. The largest upward shift in the series that exceeds one within-sigma unit is identified. See "Shift Magnitudes and Positions".

Largest Downshift Adds columns for Largest Downshift and Downshift Position to the Summary table. The largest downward shift in the series that exceeds one within-sigma unit is identified. See "Shift Magnitudes and Positions".

Shift Graph (Available only when some processes have shifts that exceed the Shift Threshold.) Shows a plot of the time occurrence of all process shifts that exceed the number of within-sigma units specified by the Shift Threshold (three by default). You can also change the Shift Threshold by using the slider bar on the Shift Graph. Hover over the diamond to see the value. Green markers indicate upshifts and red markers indicate downshifts. The markers are located at the local peaks of the shifts.

To identify the processes that correspond to one or more shift occurrences, select the points and click Select Process. The corresponding processes are selected in the Summary table. Processes that have no shifts exceeding the Shift Threshold number of within-sigma units are not plotted.

Note: The Shift Graph does not show the positions of Largest Upshift and Largest Downshift values that appear in the Summary table if the shifts are less than the specified Shift Threshold number of within-sigma units in magnitude. See "Additional Example of Process Screening".

Show Shifts in Quick Graphs (Available only when a Quick Graph has been added to the report window.) Shows the location of the shifts in the Quick Graphs using green and red vertical lines.

Drift Summaries Adds columns for Mean Up Drift, Mean Down Drift, and Mean Abs Drift to the Summary Table. See "Drift Magnitudes and Positions".

Drift Graph Selected Displays a drift graph for each process that you select in the Summary Table. Drift graphs enable you to detect smaller, more gradual changes in the selected processes. The values plotted are the slope estimates from a Holt Double-Exponential Smoothing model. See "Drift Magnitudes and Positions". Each drift graph displays the within-sigma value and the estimates for the two smoothing parameters in the Holt Double-Exponential Smoothing model.

Process Performance Graph (Available only when specification limits are defined for at least one process variable.) Shows a four-quadrant graph that assesses the performance of processes in terms of stability and capability. See "Process Performance Graph".

Process Performance Graph Boundaries (Available only once the Process Performance Graph is in the report window.) Opens a window where you can set values for the Process Performance Graph's stability index and Ppk capability boundaries.

Tip: You can set preferences for your desired boundaries for Stability Index and Ppk Capability in File > Preferences > Platforms > Process Screening.

Goal Plot (Available only when specification limits are defined for at least one process variable.) The Goal Plot shows, for each variable, the spec-normalized mean shift on the X axis, and the spec-normalized standard deviation on the Y axis. If you define importance values for the processes, the goal plot points are sized by importance. It is useful for getting a quick, summary view of how the variables are conforming to specification limits. See *Quality and Process Methods* for more information about goal plots and their features.

Tip: Hover over a point in the Goal Plot to view the Quick Graph for that process. Click the Quick Graph to add it to the report window.

The Goal Plot red triangle menu contains the following options:

Show Within Sigma Points Shows or hides the points calculated using the within sigma estimate.

Show Within or Between-and-Within Sigma Points (Available only when three way control chart is selected in the launch window.) Shows or hides the points calculated using the between-and-within sigma estimate.

Show Overall Sigma Points Shows or hides the points calculated using the overall sigma estimate.

Shade Levels Shows or hides the Ppk level shading. This is turned on by default. The shaded areas depend on the relationship between p and Ppk, with p representing the value shown in the box beneath Ppk. By default, $p = 1.33$.

- Points in the red area have Ppk < p.
- Points in the yellow area have p < Ppk < $2p$.
- Points in the green area have $2p$ < Ppk.

Label Within Sigma Points Shows or hides labels for points calculated using the within sigma estimate.

Label Within or Between-and-Within Sigma Points (Available only when a three way control chart is selected in the launch window.) Shows or hides labels for points calculated using the between-and-within sigma estimate.

Label Overall Sigma Points Shows or hides labels for points calculated using the overall sigma estimate.

Defect Rate Contour Shows or hides a contour representing a specified defect rate.

Show Capability (Available only when specification limits are defined for at least one process variable.) Displays a submenu that enables you to show or hide the Capabilities in the Summary table.

Note: Spec Centered Mean and Spec Scaled Std Dev are displayed in the Summary table as (Mean-Tgt)/SpecRange and StdDev/SpecRange. The Spec Limits option displays the LSL, USL, and Target columns in the Summary table.

The following option is also available in the submenu:

Color Out of Spec Values Colors the cells in the data table that correspond to values that are out of spec. The cell is colored blue if the value is above the USL and red if the value is below the LSL.

Tip: To remove colors in specific cells, select all cells of interest. Right-click in one of the cells and select Clear Color.

Save Summary Table Saves all of the information that can appear in the Summary table to a Process Summary data table. The Process Summary data table also contains specification limit details, if these are specified for at least one process. As in the summary table, the columns for Stability Index, Ppk, Cpk, Cp, and Target Index are colored as green, yellow, and red to indicate adequate, marginal, and poor stability or capability, respectively.

Save Summary Table with Graphs Creates the Process Summary data table with an additional column titled Graph. The Graph column contains a quick graph for each process in the Process Summary table.

Save Details Table Saves detailed information about control chart calculations to a Process Details data table. For each combination of process and grouping variables, the table contains a row for each subgroup showing:

- The value of the subgroup variable
- The values of the subgroup sample statistics.
- The control limits.
- The subgroup size.

- A list of indicators for which, if any, alarms were triggered. Alarms for the Nelson tests are indicated with the numbers for the tests. An alarm for the Range Limit Exceeded test is indicated with an R.
- The Drift

Save Selected Details For the selected rows in the Summary table, saves a Process Details table with the information that is saved when you select Save Details Table.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Process Performance Graph

The Process Performance Graph is a four-quadrant plot of capability versus stability. Each process for which specification limits are provided is represented by a marker. The marker type depends on if the process violates guidelines for the Target Index and Cp values. A left-pointing triangle indicates the process is below target, a circle indicates the process is on target, and a right-pointing triangle indicates the process is above target. An open marker indicates that the variation of the process is adequate and a filled marker indicates that the variation of the process could be reduced. A complete description of the marker combinations is provided in Table 23.1. Additionally, if you define importance values for the processes, the markers on the Process Performance Graph are sized by importance.

Table 23.1 Process Performance Graph Markers

Graph Marker	Target Index	Cp
○	On Target	Adequate Variation
●	On Target	Reduce Variation
▷	Above Target	Adequate Variation
▶	Above Target	Reduce Variation
◁	Below Target	Adequate Variation
◀	Below Target	Reduce Variation

On the graph, the horizontal coordinates represent the stability index of the process and the vertical coordinates represent the capability of the process, given as Ppk. The graph is divided into four quadrants based on the following default boundaries:

– A stability index that exceeds 1.25 indicates that the process is unstable.
– A Ppk that is smaller than 1.33 indicates that the process is not capable.

Additionally, there is a red line on the graph at 1.33 divided by Stability Index that indicates where the Cpk value is 1.33. This line categorizes the processes in the unstable and incapable quadrants in terms of how they might be fixed. Processes below the red line are fixable by special causes; processes above the red line are fixable by reducing variability. Selecting points in the graph selects the corresponding processes in the Summary table.

Tip: Hover over a point in the Process Performance Graph to view the Quick Graph for that process. Click the Quick Graph to add it to the report window.

Additional Example of Process Screening

This example illustrates the use of a Grouping column and the construction of a Shift Graph using Consumer Prices.jmp.

The consumer price index data table contains monthly data on 17 products. The time periods vary by product. The data are arranged so that all 17 products are listed in a single column called Series. To separate the products, you must treat Series as a Grouping column.

1. Select **Help > Sample Data Library** and open Consumer Prices.jmp.
2. Select **Analyze > Screening > Process Screening**.
3. Select Price and click **Process Variables.**
4. Select Series and click **Grouping**.

   This ensures that each level of Series is treated as a separate process.
5. Select Date and click **Time**.
6. Set the **Control Chart Type** to XBar and R.
7. Set the **Subgroup Sample Size** to 3.

   Because the data are given monthly, subgroups of size three represent quarters.
8. Click **OK**.
9. Press Alt and click the red triangle next to Process Screening.

   This opens a window showing the available red triangle options. You can select multiple options at once in this window.
10. Check the following: **Largest Upshift**, **Largest Downshift**, and **Shift Graph**.
11. Click **OK**.

    The columns Largest Upshift, Upshift Date, Largest Downshift, and Downshift Date are added to the Summary table. The shifts are the largest shifts exceeding *one* within-sigma unit. The position of each shift is given in terms of the Time variable, Date. See "Shift Magnitudes and Positions".

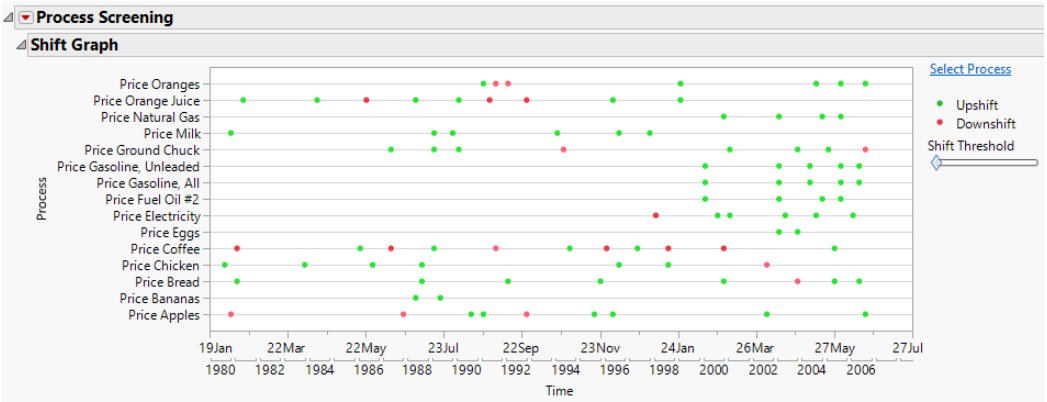
    A Shift Graph also appears. The Shift Graph shows all shifts that exceed the number of *Shift Threshold* within-sigma units, which is set to three by default. See "Shift Graph". Green points correspond to upshifts and red points correspond to downshifts.

    Notice that Gasoline, All has values for both Largest Upshift and Largest Downshift in the Summary table. The Largest Downshift value, 1.8296, is less than three. Because the Shift Graph shows shifts of only three or more within-sigma units, the Largest Downshift value for Gasoline, All is not plotted on the Shift Graph.

    Also notice that Tomatoes is not included on the Shift Graph. For Tomatoes, no shifts of three or more within-sigma units were found.

12. Double-click the horizontal axis of the Shift Graph to open the X Axis Settings window.
13. In the Tick/Bin Increment panel, set **# Minor Ticks** to 1.
14. Set **Label Row Nesting** to 2.
15. Click **OK**.

Figure 23.7 Shift Graph

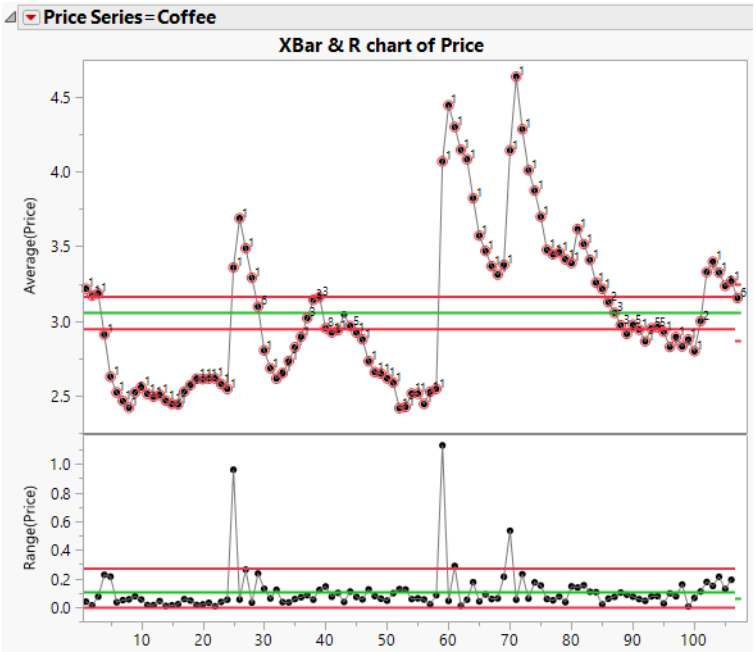


Most series show primarily upshifts. Price Coffee, however, has several alternating downshifts and upshifts. To better understand this series, obtain a control chart.

16. Select any point to the right of Price Coffee in the Shift Graph and click **Select Process**.

    This action selects the row of the Summary table corresponding to Coffee.
17. Right-click the selected process in the Summary table and select **Control Charts for Selected Items**.

Figure 23.8 Control Chart for Coffee



The control chart shows the upshifts and downshifts that are identified in the Shift Graph.

The Summary table indicates that the largest upshift (25.399 within-sigma units) occurs for the subgroup that includes September 1994. In the control chart in Figure 23.8, this is the subgroup in position 59. The Summary table also indicates that the largest downshift (9.1674 within-sigma units) occurs for the subgroup that includes March 1981. This is the subgroup in position 5 in the control chart.

Because shifts are calculated using EWMA-smoothed series and an outlier-correction algorithm, the shift positions might not precisely correspond to the subgroups that seem to start the shifts on a Shewhart control chart.

Statistical Details for the Process Screening Platform

Scaling Factors for Using Medians to Estimate Sigma

When you select Use Medians instead of Means, sigma is estimated using a scaled median range or median standard deviation. The table below gives the scaling factors, which were obtained using Monte Carlo simulation.

For subgroups of size *n* drawn from a normal distribution, the following are true:

- The theoretical median of the ranges is approximately *d2_Median**σ, where *d2_Median* is the value corresponding to *n*.
- The theoretical median of the standard deviations is approximately *c4_Median**σ, where *c4_Median* is the value corresponding to *n*.

Table 23.2 Scaling Constants for Median Range and Median Standard Deviation

n	d2_Median	c4_Median
2	0.953	0.675
3	1.588	0.833
4	1.978	0.888
5	2.257	0.917
6	2.471	0.933
7	2.646	0.944
8	2.792	0.952
9	2.915	0.959
10	3.024	0.963
11	3.118	0.967
12	3.208	0.969
13	3.286	0.972
14	3.357	0.975
15	3.422	0.976
16	3.483	0.978
17	3.539	0.979
18	3.590	0.980
19	3.640	0.981
20	3.685	0.982
21	3.731	0.983
22	3.770	0.984

Table 23.2 Scaling Constants for Median Range and Median Standard Deviation *(Continued)*

n	d2_Median	c4_Median
23	3.811	0.984
24	3.846	0.985
25	3.883	0.986

Chapter 24

Predictor Screening

Screen Many Predictors for Significant Effects

The analysis of large data sets, where hundreds to thousands of measurements on a part, process, or sample are taken requires innovative approaches. The Predictor Screening platform provides a method of screening many predictors for their ability to predict an outcome. For example, predictor screening can be used to help identify biomarkers from thousands tested in samples from patients with and without a condition to predict the condition.

Predictor screening differs from response screening. Response screening tests factors one at a time as a predictor of the response. Predictor screening uses bootstrap forest partitioning to evaluate the contribution of predictors on the response. The partition models are built on multiple predictors. Predictor screening can identify predictors that might be weak alone but strong when used in combination with other predictors. See "Response Screening".

Figure 24.1 Example of a Predictor Screening Report

Predictor Screening

Banding? Copy Selected

Predictor	Contribution	Portion	Rank
ink pct	27.3767	0.1609	1
solvent pct	14.5981	0.0858	2
press	14.0203	0.0824	3
varnish pct	13.6577	0.0803	4
roller durometer	11.8470	0.0696	5
press speed	11.0130	0.0647	6
ESA Voltage	7.1478	0.0420	7
viscosity	6.9979	0.0411	8
press type	6.5136	0.0383	9
ESA Amperage	6.4143	0.0377	10
grain screened	4.6011	0.0270	11
unit number	4.3955	0.0258	12
humidity	4.1744	0.0245	13
blade pressure	4.0893	0.0240	14
paper mill location	3.7328	0.0219	15
ink type	3.6759	0.0216	16
ink temperature	3.4103	0.0200	17
type on cylinder	2.8272	0.0166	18
proof cut	2.8073	0.0165	19
hardener	2.7894	0.0164	20
anode space ratio	2.2329	0.0131	21
current density	1.8053	0.0106	22
caliper	1.6249	0.0095	23
cylinder size	1.3343	0.0078	24
blade mfg	1.2237	0.0072	25
roughness	1.1170	0.0066	26
proof on ctd ink	1.1017	0.0065	27
paper type	1.0461	0.0061	28
plating tank	0.9256	0.0054	29
wax	0.8162	0.0048	30
solvent type	0.7797	0.0046	31
direct steam	0.0556	0.0003	32
chrome content	0.0000	0.0000	33

Contents

Overview of the Predictor Screening Platform 511
Example of Predictor Screening 511
Launch the Predictor Screening Platform 513
The Predictor Screening Report 514
Predictor Screening Platform Options 514

Overview of the Predictor Screening Platform

Predictor screening is useful for the identification of significant predictors from a large number of candidates. Suppose you had hundreds of Xs and needed to determine which of those were most significant as predictors of an outcome.

The predictor screening platform uses a bootstrap forest to identify potential predictors of your response. For each response, a bootstrap forest model using 100 decision trees is built. The column contributions to the bootstrap forest model for each predictor are ranked. Because the bootstrap forest method involves a random component, column contributions can differ when you rerun the report. For more information about decision trees, see "Partition Models".

Example of Predictor Screening

The Bands Data.jmp data table contains measurements from machinery in the rotogravure printing business. The data set contains 539 records and 38 variables. The response Y is the column Banding? and its values are "BAND" and "NOBAND". You are interested in understanding what properties are most likely to contribute to the response.

1. Select **Help > Sample Data Library** and open Bands Data.jmp.
2. Select **Analyze > Screening > Predictor Screening**.
3. Select Banding? as **Y, Response**.
4. Select the grouped columns grain screened to chrome content and click **X**.
5. Click **OK**.

Figure 24.2 Ranked Column Contributions

Predictor Screening

Banding? Copy Selected

Predictor	Contribution	Portion	Rank
ink pct	27.3767	0.1609	1
solvent pct	14.5981	0.0858	2
press	14.0203	0.0824	3
varnish pct	13.6577	0.0803	4
roller durometer	11.8470	0.0696	5
press speed	11.0130	0.0647	6
ESA Voltage	7.1478	0.0420	7
viscosity	6.9979	0.0411	8
press type	6.5136	0.0383	9
ESA Amperage	6.4143	0.0377	10
grain screened	4.6011	0.0270	11
unit number	4.3955	0.0258	12
humidity	4.1744	0.0245	13
blade pressure	4.0893	0.0240	14
paper mill location	3.7328	0.0219	15
ink type	3.6759	0.0216	16
ink temperature	3.4103	0.0200	17
type on cylinder	2.8272	0.0166	18
proof cut	2.8073	0.0165	19
hardener	2.7894	0.0164	20
anode space ratio	2.2329	0.0131	21
current density	1.8053	0.0106	22
caliper	1.6249	0.0095	23
cylinder size	1.3343	0.0078	24
blade mfg	1.2237	0.0072	25
roughness	1.1170	0.0066	26
proof on ctd ink	1.1017	0.0065	27
paper type	1.0461	0.0061	28
plating tank	0.9256	0.0054	29
wax	0.8162	0.0048	30
solvent type	0.7797	0.0046	31
direct steam	0.0556	0.0003	32
chrome content	0.0000	0.0000	33

Note: Because this analysis is based on the Bootstrap Forest method that has a random selection component, your results can differ slightly from those in Figure 24.2. See "Bootstrap Forest".

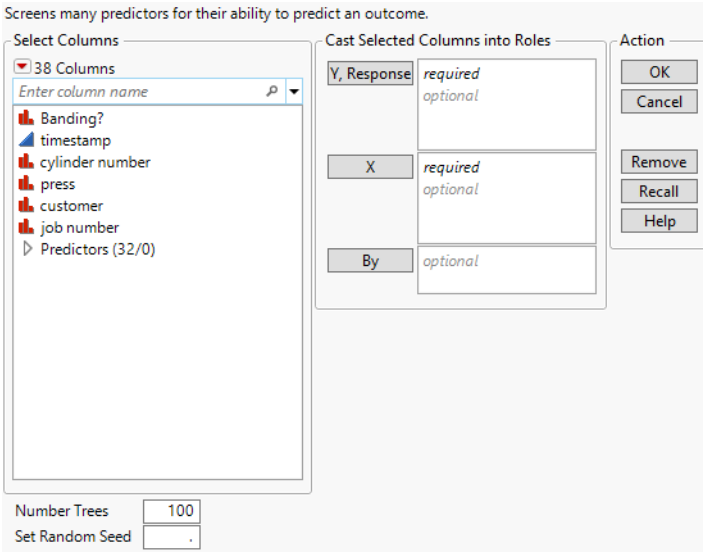
The columns are sorted and ranked in order of contribution in the bootstrap forest model. Predictors with the highest contributions are strong candidate predictors for the response of interest.

Tip: Click on a column heading to change how the columns are sorted in the table.

Launch the Predictor Screening Platform

Launch the Predictor Screening platform by selecting **Analyze > Screening > Predictor Screening**.

Figure 24.3 Predictor Screening Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Response The response columns.

X Predictor columns.

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Number of Trees The number of decision trees in the bootstrap forest model.

Set Random Seed Sets the seed for the starting values used in the fitting procedures. Set Random Seed is useful if you want to reproduce an analysis. If you set a random seed and save the script, the seed is automatically saved in the script.

The Predictor Screening Report

The report (Figure 24.2) shows the list of predictors with their respective contributions and rank. Predictors with the highest contributions are likely to be important in predicting Y.

The Contribution column shows the contribution of each predictor to the bootstrap forest model. The Portion column in the report shows the percent contribution of each variable.

You can select the important predictors in the Predictor Screening report. Selecting the important predictors selects the corresponding columns in the data table, enabling you to easily enter these columns into the launch windows for modeling platforms. The Copy Selected button enables you to copy the selected columns to the clipboard so that you can paste them into a model as well. In this way, the Predictor Screening enhances the modeling process.

Predictor Screening Platform Options

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Chapter 25

JMP PRO Association Analysis

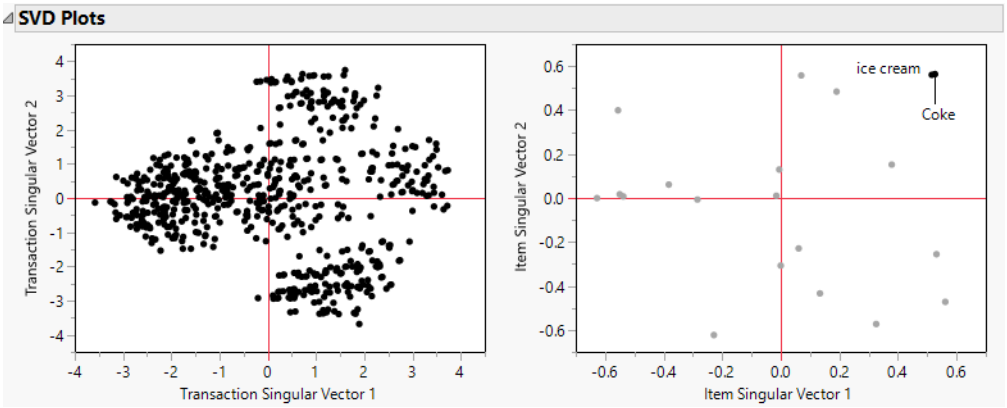
Perform Market Basket Analysis

JMP PRO *The Association Analysis platform is available only in JMP Pro.*

Association analysis enables you to identify items that have an affinity for each other. It is frequently used to analyze transactional data (also called *market baskets*) to identify items that often appear together in transactions. For example, grocery stores and online merchants use association analysis to strategically organize and recommend products that tend to be purchased together.

Association analysis is also used for identifying dependent or associated events. For example, you can identify car parts that seem to fail around the same time. In this application, car inspections are treated as the market baskets and you analyze the associations among groups of faulty parts found in each inspection.

Figure 25.1 Example of Singular Value Decomposition Plots



Contents

Overview of the Association Analysis Platform . . . 517
Example of the Association Analysis Platform . . . 518
Launch the Association Analysis Platform . . . 520
Data Format . . . 521
The Association Analysis Report . . . 522
Frequent Item Sets . . . 522
Rules . . . 522
Association Analysis Platform Options . . . 523
Singular Value Decomposition . . . 524
SVD Report . . . 525
SVD Report Options . . . 526
Topic Analysis . . . 527
Topic Analysis Report . . . 527
Topic Analysis Report Options . . . 528
Additional Example: SVD Analysis . . . 528
Statistical Details for the Association Analysis Platform . . . 531
Frequent Item Set Generation . . . 531
Association Analysis Performance Measures . . . 531

JMP PRO Overview of the Association Analysis Platform

The Association Analysis platform identifies connections among groups of items in an independent event or *transaction*. In association analysis, an *item* is the basic object of interest. For example, an item could be a product, a web page, or a service. An *item set* is a list of one or more items.

The relationship between two item sets is defined by an *association rule*. An association rule consists of a *condition* item set and a *consequent* item set. *Antecedents* are the individual items in the condition item set. Association analysis identifies association rules, which predict that a consequent item set will be in a transaction, given that the condition item set is already in the transaction. Some association rules are stronger, and therefore more useful, than others. The following three performance measures describe the strength of an association rule:

- *Support* is the proportion of transactions in which an item set appears. A high value for support indicates that the item set occurs frequently.
- *Confidence* is the proportion of transactions that contain the consequent item set, given that the condition item set is in the transaction. Confidence measures the strength of implication, or the predictive power, of an association rule. Note that confidence in association analysis is not related to the concept of confidence intervals.
- *Lift* is the ratio of an association rule's confidence to its expected confidence, assuming that the condition and consequent item sets appear in transactions independently. Lift measures how much the consequent item set depends on the presence of the condition item set. The minimum value for lift is 0.
  - A lift ratio less than 1 indicates that the condition and consequent repel each other, because they occur together less frequently than one would expect by chance alone.
  - A lift ratio close to 1 indicates that the consequent occurs at the same rate in transactions that contain the condition as one would expect from chance alone.
  - A lift ratio greater than 1 indicates that the consequent item set has an affinity for the condition item set. The consequent item set occurs more often with the condition item set than one would expect by chance alone.

For more information about these performance measures, see "Association Analysis Performance Measures".

The Association Analysis platform also enables you to perform singular value decomposition. Singular value decomposition (SVD) groups similar transactions and also groups similar items using a matrix reducing methodology that is different from what is used in association analysis. Use the SVD methodology to gain insights that complement what you learn from association analysis.

For more information about association analysis, see Hastie et al. (2009) and Shmueli et al. (2010). For more information about singular value decomposition, see Jolliffe (2002).

JMP PRO Example of the Association Analysis Platform

This example uses the Grocery Purchases.jmp sample data table, which contains transactional data reported by a grocery store. The data table lists the items purchased by 1001 customers, each assigned a unique customer ID. You want to explore the associations among items in order to identify patterns in consumer behavior.

1. Select **Help > Sample Data Library** and open Grocery Purchases.jmp.
2. Select **Analyze > Screening > Association Analysis**.
3. Select Product and click **Item**.
4. Select Customer ID and click **ID**.
5. Click **OK**.

   By default, the Rules report is sorted in decreasing order by Confidence. However, association rules with extremely high confidence also tend to have a higher number of items in the condition set. Since you want to view association rules with smaller condition sets, sort the report by Confidence, but in increasing order.
6. Right-click in the Rules report and select **Sort By Column**.

   The Select Columns window appears.
7. Select Confidence and then check the Ascending option.
8. Click **OK**.

Figure 25.2 Association Analysis Report

Rules

Rule			
Condition	**Consequent**	**Confidence**	**Lift**
apples	sardines	40%	1.357
Coke	turkey	40%	1.422
sardines	chicken, Heineken	40%	2.002
herring, olives	ham, turkey	40%	3.006
ice cream	turkey	40%	1.424
artichoke	baguette, Heineken	40%	1.547
ham	corned beef, herring	40%	1.648
ham	corned beef, olives	40%	1.703
peppers	sardines	41%	1.371
sardines	peppers	41%	1.371
Coke	Heineken, sardines	41%	2.23
sardines	Coke, Heineken	41%	2.332
avocado, Heineken	crackers, ham	41%	2.707
soda	baguette, herring	41%	1.631
corned beef	ham	41%	1.343
Heineken, soda	bourbon, crackers, olives	41%	3.152
artichoke, Heineken	avocado, baguette	41%	1.921
Heineken, herring	avocado, baguette	41%	1.924

9. Scroll down the report to where the Confidence values are 58%.

There is an entry in the Rules report table that indicates that 58% of customers who bought an avocado also bought an artichoke. The value of Lift is 1.908, indicating that there is a likely dependency. You want to verify that avocados and artichokes occur in a significant portion of transactions.

10. Click the disclosure icon next to Frequent Item Sets.

Figure 25.3 Frequent Item Sets Report

Frequent Item Sets

Item Set	Support	N Items
{Heineken}	60%	1
{crackers}	49%	1
{herring}	49%	1
{olives}	47%	1
{bourbon}	40%	1
{baguette}	39%	1
{corned beef}	39%	1
{crackers, Heineken}	37%	2
{avocado}	36%	1
{soda}	32%	1
{chicken}	31%	1
{apples}	31%	1
{ice cream}	31%	1
{artichoke}	30%	1
{ham}	30%	1
{Coke}	30%	1
{peppers}	30%	1
{sardines}	30%	1
{Heineken, herring}	29%	2
{turkey}	28%	1
{baguette, Heineken}	26%	2
{Heineken, soda}	26%	2
{herring, olives}	26%	2
{artichoke, Heineken}	25%	2

The Frequent Item Sets report shows that 36% of customers purchased avocados. The Rules report in Figure 25.2 shows that 58% of these customers also bought artichokes. Because of the large proportion of customers who follow this behavior, the grocery store management might use this information to strategically locate avocados and artichokes.

You also decide to look at the association rules with the highest lift.

11. Right-click in the Rules report table and select **Sort By Column**.

    The Select Columns window appears.

12. Select Lift and click **OK**.

    The Rules table is sorted by decreasing values of lift. Notice that the second association rule has a lift of 5.642 and 83% confidence. You want to verify that both the condition set, {chicken, ice cream}, and the consequent item set, {Coke, sardines}, have adequate support.

13. Right-click in the Frequent Item Sets report and select **Sort By Column**.

    The Select Columns window appears.

14. Select Item Set and then check the Ascending option.

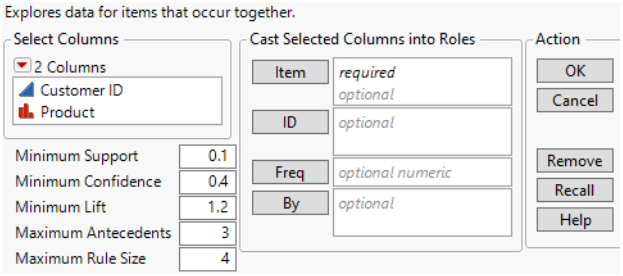
15. Click **OK**.

The Frequent Item Sets table is sorted alphabetically by item set. Scroll through the list to see that the condition item set, {chicken, ice cream}, has 14% support. The consequent item set, {Coke, sardines}, has 15% support. This association rule has high lift, but represents fewer transactions than the first association rule that you examined.

JMP PRO Launch the Association Analysis Platform

Launch the Association Analysis platform by selecting **Analyze > Screening > Association Analysis**.

Figure 25.4 Association Analysis Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Item The categorical column or columns that contain the item data to be analyzed. If the column has the Multiple Response modeling type, each of the multiple responses in each row is treated as an item. If multiple columns are specified, each column in each row is treated as an item.

ID The column that identifies the transaction to which an item belongs. The ID role is required, unless the Item role is filled by a column with the Multiple Response modeling type or the Item role is filled by multiple columns.

Note: If an ID role is specified, all rows with the same ID are combined into one transaction.

Freq The column that identifies a frequency for the transactions in the Item column. For example, if a transaction has a frequency of 3, that is treated as three identical transactions that contain the items in the Item column.

Note: The Freq role is ignored if the Item role is filled by a column that does not have the Multiple Response modeling type.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Minimum Support Specifies a minimum value for the proportion of occurrences of an item set. This value must be between 0 and 1. Only item sets with support equal to or exceeding this value are considered in the analysis.

Minimum Confidence Specifies a minimum value for the proportion of occurrences that a consequent item set occurs within transactions that contain the conditional item set. This value must be between 0 and 1. Only association rules with confidence equal to or exceeding this value appear in the report.

Minimum Lift Specifies a minimum dependency ratio. Lift values must be 0 or greater. Only association rules with lift equal to or exceeding this value appear in the report.

Maximum Antecedents Specifies the maximum number of items in the condition item set. Association rules with more than this number of items in the condition set are not considered in the analysis.

Maximum Rule Size Specifies the maximum number of items that appear in the union of the condition and consequent item sets. Association rules with more than this combined number of items are not considered in the analysis. The default value is 4.

Note: You can use the minimum support, maximum antecedent, and maximum rule size options in the launch window to reduce computational time for large data sets. For more information about these measures, see "Statistical Details for the Association Analysis Platform".

Data Format

There are three data formats that are accepted in the Association Analysis platform:

- You can specify a single item response in each row in the Item role and identify which items are included in each transaction using the ID role. In this format, the Item and ID roles are required and the Freq role is ignored.
- You can specify a Multiple Response modeling type column for the Item role. In this format, the Item role is required, the ID and Freq roles are optional. If an ID role is specified, all rows with the same ID are combined into one transaction. If a Freq role is specified, the frequency applies to the transaction, not the specific items in the transaction.
- You can specify multiple item response columns in the Item role. In this format, the ID and Freq roles are optional. If an ID role is specified, all rows with the same ID are combined into one transaction. If a Freq role is specified, the frequency applies to the transaction, not the specific items in the transaction.

JMP PRO The Association Analysis Report

By default, the Association Analysis report contains the following reports:

- "Frequent Item Sets"
- "Rules"

Tip: To order the contents of a table in a report by any of its columns, right-click in the table and select **Sort by Column**.

JMP PRO Frequent Item Sets

The Frequent Item Sets report lists item sets in decreasing order of support. The listed item sets meet the Minimum Support value that you specified in the launch window. Each item set is considered as a conditional and as a consequent item set to form association rules. The table contains the following columns:

Item Set The item sets that are considered as conditional or consequent sets for the association rules.

Support The proportion of transactions in which all of the items in the Item Set occur.

N Items The number of items in the Item Set.

JMP PRO Rules

The Rules report shows a table of association rules that are sorted in decreasing order of confidences. Only association rules that meet the Minimum Support, Minimum Confidence, Minimum Lift, Maximum Antecedents, and Maximum Rule Size requirements that you specified in the launch window appear in this report.

The Rules report table contains the following columns:

Rule The association rules formed by combining Condition and Consequent item sets.

Condition The item set that is thought to influence the presence of a Consequent item set within transactions.

Consequent The item set whose presence is thought to be influenced by the presence of a Condition item set.

Confidence The proportion of transactions that contain the Consequent item set, given that the condition item set is in the transaction. Confidence measures the strength of implication, or the predictive power, of an association rule.

Lift The ratio of an association rule's confidence to its expected confidence, assuming that the condition and consequent item sets appear in transactions independently. Lift measures how much the Consequent item set depends on the presence of the Condition item set. The minimum value for lift is 0.

- A lift ratio less than 1 indicates that the Condition and Consequent item sets repel each other, because they occur together less frequently than one would expect by chance alone.
- A lift ratio close to 1 indicates that the Consequent item set occurs at the same rate in transactions that contain the Condition item set as one would expect from chance alone.
- A lift ratio greater than 1 indicates that the Consequent item set has an affinity for the Condition item set. The Consequent item set occurs more often with the Condition item set than one would expect by chance alone.

JMP PRO Association Analysis Platform Options

The Association Analysis red triangle menu contains the following options:

Transaction Listing Shows or hides a table listing each Transaction ID value and the items included in that transaction. The table is sorted by the Transaction ID column.

Frequent Item Sets Shows or hides a list of item sets whose support exceeds the Minimum Support value specified in the launch window. See "Frequent Item Sets".

Rules Shows or hides a table of association rules that meet the Minimum Support, Minimum Confidence, Minimum Lift, Maximum Antecedents, and Maximum Rule Size requirements specified in the launch window. See "Rules".

SVD Shows or hides a report of a partial singular value decomposition (SVD) of the incidence matrix for the items. This decomposition reduces the incidence matrix into a user-specified number of dimensions for analysis. The SVD report also contains options to visualize the singular vectors, cluster transactions and items, save out singular vectors, and perform a rotated SVD analysis. See "Singular Value Decomposition".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Singular Value Decomposition

Singular value decomposition (SVD) complements association analysis by providing another method to identify items that have an affinity for each other. Singular value decomposition of the transaction item matrix reduces the matrix to a manageable number of dimensions, thereby enabling you to group similar transactions and similar items. The SVD analysis is equivalent to performing principal components analysis (PCA) on a correlation matrix.

The *transaction item matrix* is a matrix for which each row corresponds to a transaction and each column corresponds to an item. The entries of the matrix are zeros and ones. If an item occurs in a transaction, the corresponding row and column entry is one. Otherwise, the row and column entry is zero. Because the transaction item matrix usually contains more values of zero than one, it is called a *sparse matrix*.

The partial singular value decomposition approximates the column-standardized transaction item matrix using three matrices: **U**, **S**, and **V**′. The relationship between these matrices is defined as follows:

Transaction Item Matrix ≈ **U** * **S** * **V**′

Define *nTran* as the number of transactions (rows) in the transaction item matrix, *nItem* as the number of items (columns) in the transaction item matrix, and *nVec* as the specified number of singular vectors. Note that *nVec* must be less than or equal to min(*nTran*, *nItem*). It follows that **U** is an *nTran* by *nVec* matrix that contains the left singular vectors of the transaction item matrix. **S** is a diagonal matrix of dimension *nVec*. The diagonal entries in **S** are the singular values in the transaction item matrix. **V**′ is an *nVec* by *nItem* matrix. The rows in **V**′ (or columns in **V**) are the right singular vectors.

The right singular vectors capture connections among different items with similar functions or topic areas. If three items tend to appear in the same transactions, the SVD is likely to produce a singular vector in **V**′ with large values for those three items. The **U** singular vectors represent the transactions projected into this new item space.

The SVD also captures indirect connections. If two items never appear together in the same transaction, but they generally appear in transactions with another third item, the SVD is able to capture some of that connection. If two transactions have no items in common but contain items that are connected in the dimension-reduced space, they map to similar vectors in the SVD plots.

The SVD transforms transaction data into a fixed-dimensional vector space, making it amenable to clustering, classification, and regression techniques. The Save options enable you to export this vector space to be analyzed in other JMP platforms.

The transaction item matrix is centered, scaled, and divided by *nTran* minus 1 before the singular value decomposition is carried out. This analysis is equivalent to a PCA of the correlation matrix of the transaction item matrix. The SVD implementation takes advantage of the sparsity of the transaction item matrix.

JMP PRO SVD Report

The SVD option produces two SVD plots and a table of the singular values from the singular value decomposition.

JMP PRO SVD Plots

The first plot contains a point for each transaction. For a given transaction, the point that is plotted is defined by the transaction's values in the first two singular vectors (the first two columns of the **U** matrix) multiplied by the diagonal singular values matrix (**S**). This plot is equivalent to the Score Plot in the Principal Components platform. Each point in this plot represents a transaction. In the Transaction SVD plot, points that are visibly grouped together indicate transactions with a similar composition.

The second plot contains a point for each item. For a given item, the point that is plotted is defined by the item's values in the first two singular vectors (the first two rows of the **V**′ matrix) multiplied by the diagonal singular values matrix (**S**). This plot is equivalent to the Loadings Plot in the Principal Components platform. Each point in this plot represents an item. In the Item SVD plot, items that are visibly grouped together indicate items that have similar functions or topic areas.

JMP PRO Singular Values

Below the transaction and item SVD plots, a table of the singular values appears. These are the diagonal entries of the **S** matrix in the singular value decomposition of the transaction item matrix. The Singular Values table also contains a column of corresponding eigenvalues for the equivalent principal components analysis. Like in the Principal Components platform, there are columns for the percent and cumulative percent of variation explained by each eigenvalue (or singular value). You can use the Cum Percent column to decide what percent of variance from the transaction item matrix you want to preserve, and then use the corresponding number of singular vectors.

JMP PRO SVD Report Options

The SVD red triangle menu contains the following options:

SVD Scatterplot Matrix Shows or hides a scatterplot matrix of the item and transaction singular value decomposition vectors. You must select the size of the scatterplot matrix when you select this option. This scatterplot matrix enables you to visualize more than the first two dimensions of the singular value decomposition.

Topic Analysis, Rotated SVD Performs a varimax rotated partial singular value decomposition of the item transaction matrix to produce groups of items called topics. You can select this option multiple times to find different numbers of topics. See "Topic Analysis".

Cluster Items Shows or hides a hierarchical clustering analysis of the items in the data. The following options are available to the right of the dendrogram:

Set Clusters Specifies the number of clusters that are colored in the dendrogram and used in the Save Clusters option. Points in the Item SVD plot are also colored by cluster.

Save Clusters Saves a data table that contains the items and their cluster assignments. You must set the number of clusters before clicking Save Clusters.

For more information about clustering and dendrograms, see *Multivariate Methods*.

Cluster Transactions Shows or hides a hierarchical clustering analysis of the transactions in the data. The following options are available to the right of the dendrogram:

Set Clusters Specifies the number of clusters that are colored in the dendrogram and used in the Save Clusters option. Points in the Transaction SVD plot are also colored by cluster.

Save Clusters Saves a data table that contains the transactions and their cluster assignments. You must set the number of clusters before clicking Save Clusters.

For more information about clustering and dendrograms, see *Multivariate Methods*.

Save Transaction Singular Vectors Saves a user-specified number of singular vectors from the transaction singular value decomposition as columns to a new data table where each row corresponds to a transaction. The first two saved columns represent the points plotted in the transaction SVD plot. See "Singular Value Decomposition".

Save Item Singular Vectors Saves a user-specified number of singular vectors from the items singular value decomposition as columns to a new data table where each row corresponds to an item. The first two saved columns represent the points plotted in the item SVD plot. See "Singular Value Decomposition".

JMP PRO Topic Analysis

The Topic Analysis, Rotated SVD option performs a varimax rotation on the partial singular value decomposition (SVD) of the transaction item matrix. See "Singular Value Decomposition". You must specify a number of rotated singular vectors, which corresponds to the number topics that you want to retain from the transaction item matrix. After you specify a number of topics, the Topic Analysis report appears.

Topic analysis is equivalent to a rotated principal component analysis (rotated PCA). The varimax rotation takes a set of singular vectors and rotates them to make them point more directly in the coordinate directions (toward the items). This rotation makes the vectors help explain the composition of the transactions as each rotated vector orients toward a set of items. Negative values indicate a repulsion force. The items with negative values occur in a topic less frequently compared to the items with positive values.

JMP PRO Topic Analysis Report

The Topic Analysis report shows the items that have the largest loadings in each topic after rotation. There are additional reports that show the components of the rotated singular value decomposition.

The Top Loadings by Topic report shows a table for each topic. The items in each table are the ones that have the largest loadings in absolute value for each topic. Each table is sorted in descending order by the absolute value of the loading. These tables can be used to determine conceptual themes that correspond to each topic.

The Topic Analysis report also contains the following reports:

Topic Loadings Contains a matrix of the loadings across topics for each item. This matrix is equivalent to the factor loading matrix in a rotated PCA.

Word Clouds by Topic Contains a matrix of word clouds, one for each topic.

Topic Scores Contains a matrix of transaction scores for each topic. Transactions with higher scores in a topic are more likely to be associated with that topic.

Topic Scores Plots Shows the topic scores for all transactions in one-dimensional scatterplots. Negative values indicate transactions that are negatively associated with a topic. Use these plots to explore the distribution of transactions within each topic.

Variance Explained by Each Topic Contains a table of the variance explained by each topic. The table also contains columns for the percent and cumulative percent of the variation explained by each topic.

Rotation Matrix Contains the rotation matrix for the varimax rotation.

JMP PRO Topic Analysis Report Options

The Topic Analysis red triangle menu contains the following options:

Topic Scatterplot Matrix Shows or hides a scatterplot matrix of the rotated singular value decomposition vectors.

Display Options Contains options to show or hide content that appears in the Topic Analysis report. See "Topic Analysis Report".

Rename Topics Shows a dialog that enables you to add descriptive names for one or more of the topics.

Save Transaction Topic Vectors Saves a user-specified number of singular vectors from the rotated singular value decomposition as columns to the data table.

Save Item Topic Vectors Saves the topic vectors as columns to a new Item Topic Scores data table.

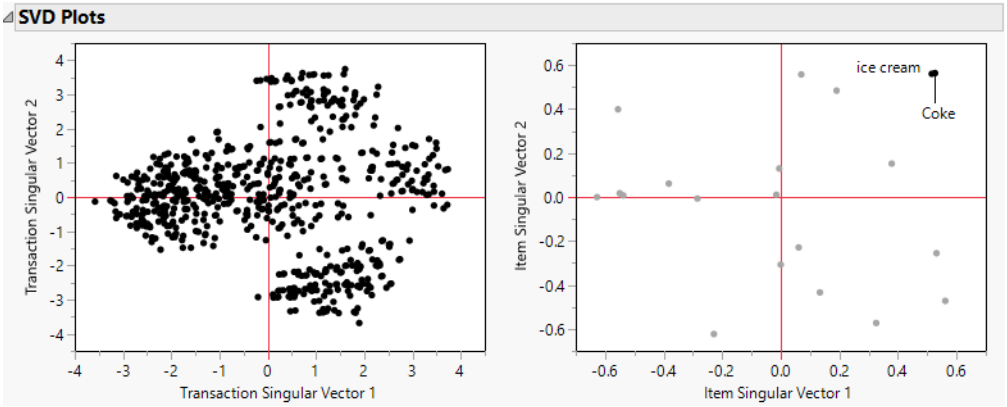
Remove Removes the Topic Analysis report from the SVD report.

JMP PRO Additional Example: SVD Analysis

In this example, you use singular value decomposition of the transaction item matrix to gain further insight into the Grocery Purchases.jmp sample data.

1. Select **Help > Sample Data Library** and open Grocery Purchases.jmp.
2. Select **Analyze > Screening > Association Analysis**.
3. Select Product and click **Item**.
4. Select Customer ID and click **ID**.
5. Click **OK**.
6. Click the Association Analysis red triangle and select **SVD**.
7. Click **OK**.

Figure 25.5 SVD Plots



The transaction SVD plot suggests that there might be two or three groups of transactions. In the upper right corner of the item SVD plot, notice that the points that represent Coke and ice cream overlap. The proximity of these two items indicates that there is a strong affinity between them.

8. Click the SVD red triangle and select **Topic Analysis, Rotated SVD**.
9. Type 3 next to Number of topics (rotated singular vectors) and click **OK**.

   The Topic Items and Topic Scores reports appear.

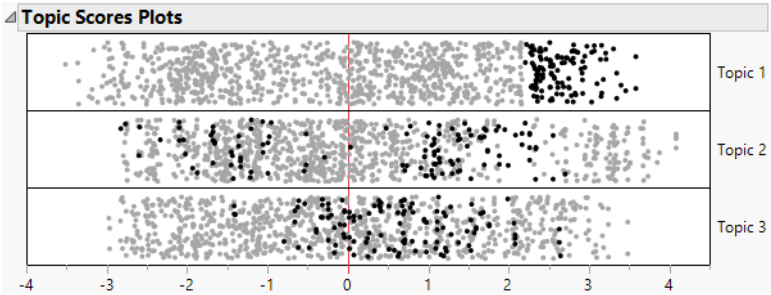
Figure 25.6 Topic Items Report

Top Loadings by Topic

Topic 1		Topic 2		Topic 3	
Item	Loading	Item	Loading	Item	Loading
avocado	-0.7191	Coke	0.7017	crackers	0.7050
olives	0.7142	ice cream	0.6945	apples	-0.6437
baguette	-0.6089	herring	-0.6550	soda	0.6411
turkey	0.5697	sardines	0.6122	Heineken	0.6409
artichoke	-0.4889	chicken	0.5158	sardines	-0.3618
bourbon	0.4860	corned beef	-0.4257	steak	-0.3468
Heineken	-0.4329	steak	-0.3395	corned beef	-0.3409
corned beef	0.3836	olives	-0.3035	bourbon	0.3340
Coke	0.3096	ham	-0.3022		

Three groups, or topics, are created and shown in the Topic Items report. The first items listed in the Topic Item tables represent the primary items for that group. For example, Topic 1 is a group that is identified primarily by transactions that do not contain avocados, but do contain olives.

Figure 25.7 Topic Scores



The topic scores that are assigned to each of the 1001 transactions are plotted in the Topic Scores report. Select groups of points for a topic to see how those transactions relate to other topics. For example, transactions with very high values on Topic 1 tend to have low values on Topics 2 and 3.

10. Open the Singular Values report.

Figure 25.8 Singular Values Table

Singular Values

Number	Singular Value	Eigenvalue	Percent	Cum Percent
1	1.7399	3.0272	15.1362	15.1362
2	1.6494	2.7204	13.6021	28.7383
3	1.5093	2.2780	11.3901	40.1284
4	1.4077	1.9816	9.9079	50.0363
5	1.2565	1.5788	7.8938	57.9301
6	1.1331	1.2839	6.4196	64.3497
7	1.0333	1.0676	5.3381	69.6877
8	0.9927	0.9854	4.9270	74.6148
9	0.9265	0.8583	4.2916	78.9063
10	0.7888	0.6223	3.1114	82.0177
11	0.7247	0.5253	2.6263	84.6440
12	0.6647	0.4419	2.2094	86.8534
13	0.6465	0.4179	2.0897	88.9431
14	0.6379	0.4069	2.0346	90.9777
15	0.6266	0.3927	1.9634	92.9411
16	0.6140	0.3770	1.8848	94.8259
17	0.6077	0.3693	1.8465	96.6724
18	0.5817	0.3383	1.6917	98.3641
19	0.5590	0.3125	1.5623	99.9264
20	0.1213	0.0147	0.0736	100.000

As seen in Figure 25.8, the first two singular values explain only about 30% of the variability in the grocery store data. Additional dimensions might be required to explain a sufficient amount of variability.

JMP PRO Statistical Details for the Association Analysis Platform

- "Frequent Item Set Generation"
- "Association Analysis Performance Measures"

JMP PRO Frequent Item Set Generation

The Association Analysis platform uses the *Apriori algorithm* to reduce computational time when generating frequent item sets. The Apriori algorithm leverages the fact that an item set's support is never larger than the support of its subsets. The platform generates larger item sets from combinations of smaller item sets that meet the minimum support level. In addition, the platform does not generate item sets that exceed either the specified maximum number of antecedents or the maximum rule size. These options are useful when working with large data sets, because the total possible number of rules increases exponentially with the number of items. For more information about the Apriori algorithm, see Agrawal and Srikant (1994).

JMP PRO Association Analysis Performance Measures

This section defines the performance measures used in the Association Analysis platform. Denote an association rule with condition item set X and consequent item set Y by $X \Rightarrow Y$. Hahsler (2015) contains a collection of performance measures used in association analysis, including support, confidence, and lift.

JMP PRO Support

Support is the proportion of transactions in which an item set occurs. Support can also be viewed as the probability that a transaction contains an item set.

The support S of a condition item set X is defined as follows:

$$supp(X) = \frac{N_X}{N} = Pr(X)$$

where:

N_X is the number of transactions that contain the item set X

N is the total number of transactions.

Support for an association rule is defined as follows:

$$supp(X \Rightarrow Y) = supp(X \cup Y) = Pr(X \cap Y)$$

In this case, support is equivalent to the probability that a transaction contains both item sets X and Y.

For both the item set and association rule cases, support ranges from 0% to 100%.

JMP PRO Confidence

Confidence is the proportion of transactions that contain the consequent item set, given that the transaction contains the condition item set. Confidence can also be viewed as the conditional probability that a transaction contains the consequent item set, given that the transaction contains the condition item set.

$$conf(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)} = \frac{Pr(X \cap Y)}{Pr(X)} = Pr(Y|X)$$

Confidence ranges from 0% to 100%. An association rule with a confidence of 0% has a consequent item set that does not appear in any transaction with the condition item set. A confidence of 100% indicates that every transaction that contains the condition item set also contains the consequent item set.

Note: Confidence in association analysis is not related to the concept of confidence intervals.

JMP PRO Lift

Lift measures dependency between X and Y.

$$lift(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X) \times supp(Y)} = \frac{Pr(X \cap Y)}{Pr(X)Pr(Y)}$$

The numerator for lift is the proportion of transactions where X and Y occur jointly. The denominator is an estimate of the expected joint occurrence of X and Y, assuming that they occur independently.

Lift ranges from 0 to ∞. A lift value of 1 indicates that X and Y jointly occur in transactions with the frequency that would be expected by chance alone. Increasing lift values suggest that Y occurs more often than expected when X is present.

Note: The lift for association rule $X \Rightarrow Y$ is equal to the lift for association rule $Y \Rightarrow X$.

Chapter 26

Process History Explorer

Identify Problem Components in Complex Process Histories

Note: The Process History Explorer platform is new and experimental. It is subject to change in future releases.

In complex manufacturing situations, such as in semiconductor factories, a defective tool or route combination might lead to poor yield or reduced quality. However, there might be hundreds of steps that a unit passes through that involve many different tools. It can be difficult to determine the exact combination of factors that leads to the poor yield.

The Process History Explorer platform organizes all of the production tracking data and enables you to identity factors that seem to be associated with poor yield. You can then perform stepwise regression. You can analyze the time that a unit spends waiting between steps or tools in the process and see how that waiting time is related to yield. You can also analyze the effect on yield of transitions from one tool to another.

Figure 26.1 Process History Explorer Report

Process History Explorer

1250 Units, using ID columns: Lot, Wafer
125 Steps, using columns: Layer, Operation
156250 Operations, using time columns: TimeIn, TimeOut
195 levels in X columns: Tool, Route
1 response: Yield

Response Yield

Levels with Lowest Yields

Term	Level	Count	Mean	Std Error	Std Dev
Tool	PlanarTool008	1000	0.75299	0.00188	0.05952
Tool	DepTool003	1375	0.75577	0.00168	0.06245
Tool	EtchTool019	750	0.76192	0.0021	0.05749
Tool	EtchTool012	500	0.76203	0.0026	0.05807
Tool	EtchTool034	850	0.76279	0.00199	0.0581
Tool	PlanarTool021	1050	0.7637	0.00186	0.06028
Tool	PlanarTool030	1025	0.76539	0.00184	0.05894
Tool	EtchTool016	900	0.76566	0.00191	0.05734
Tool	EtchTool009	925	0.76598	0.00197	0.05994
Tool	Dep2Tool018	1525	0.76766	0.00156	0.06088
Tool	Dep2Tool014	1825	0.76813	0.00138	0.05893
Tool	DepTool019	1650	0.76839	0.00143	0.05794
Tool	EtchTool020	1000	0.76909	0.00177	0.05586
Tool	Dep2Tool015	1475	0.76948	0.0015	0.05775
Tool	PhotoTool010	3525	0.76996	0.00098	0.05822
Tool	Dep2Tool013	1825	0.7705	0.00141	0.06025
Tool	DepTool012	1750	0.77063	0.00145	0.06051
Tool	Dep2Tool006	1400	0.77063	0.00158	0.05905
Tool	PlanarTool028	900	0.77103	0.00202	0.06071
Tool	Dep2Tool011	1600	0.77124	0.00145	0.05797
Tool	EtchTool010	850	0.77125	0.00213	0.06208

Contents

Example of Process History Explorer 537
Launch the Process History Explorer Platform 539
Data Format 540
The Process History Explorer Report 540
Process History Explorer Platform Options 541

Example of Process History Explorer

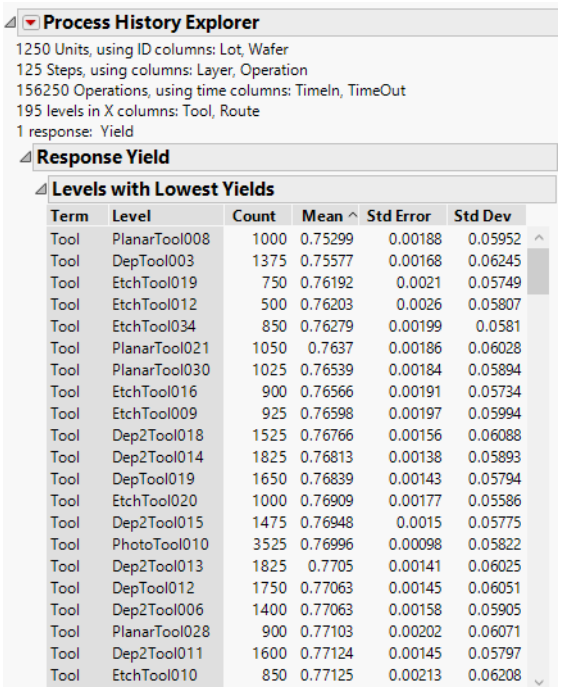
This example uses two data tables. The first data table contains step-by-step history of 25 wafers in 50 lots that went through a complex manufacturing process. The second data table contains the yield for each wafer in each lot.

1. Select **Help > Sample Data Library** and open Quality Control > Lot Wafer History.jmp and Quality Control > Lot Wafer Yield.jmp.
2. From the Lot Wafer History.jmp data table, select **Analyze > Screening > Process History Explorer**.
3. Select Lot and Wafer and click **ID**.
4. Select Tool and Route and click **X, Process**.
5. Select Layer and Operation and click **Step**.
6. Select TimeIn and TimeOut and click **Timestamp**.
7. Click **OK**.

   A window appears that enables you to select the corresponding yield table.
8. Select Lot Wafer Yield and click **OK**.

   A window appears that enables you to select the yield column.
9. Select Yield and click **OK**.
10. Click the Process History Explorer red triangle and select **Levels with Lowest Yield**.

Figure 26.2 Process History Explorer Report



Process History Explorer

1250 Units, using ID columns: Lot, Wafer
125 Steps, using columns: Layer, Operation
156250 Operations, using time columns: TimeIn, TimeOut
195 levels in X columns: Tool, Route
1 response: Yield

Response Yield

Levels with Lowest Yields

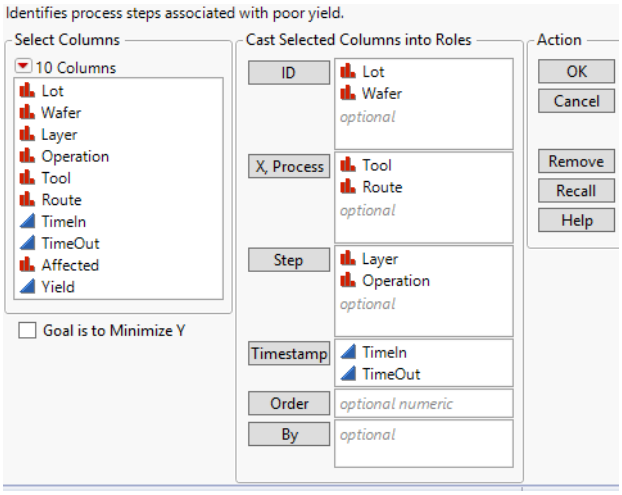
Term	Level	Count	Mean	Std Error	Std Dev
Tool	PlanarTool008	1000	0.75299	0.00188	0.05952
Tool	DepTool003	1375	0.75577	0.00168	0.06245
Tool	EtchTool019	750	0.76192	0.0021	0.05749
Tool	EtchTool012	500	0.76203	0.0026	0.05807
Tool	EtchTool034	850	0.76279	0.00199	0.0581
Tool	PlanarTool021	1050	0.7637	0.00186	0.06028
Tool	PlanarTool030	1025	0.76539	0.00184	0.05894
Tool	EtchTool016	900	0.76566	0.00191	0.05734
Tool	EtchTool009	925	0.76598	0.00197	0.05994
Tool	Dep2Tool018	1525	0.76766	0.00156	0.06088
Tool	Dep2Tool014	1825	0.76813	0.00138	0.05893
Tool	DepTool019	1650	0.76839	0.00143	0.05794
Tool	EtchTool020	1000	0.76909	0.00177	0.05586
Tool	Dep2Tool015	1475	0.76948	0.0015	0.05775
Tool	PhotoTool010	3525	0.76996	0.00098	0.05822
Tool	Dep2Tool013	1825	0.7705	0.00141	0.06025
Tool	DepTool012	1750	0.77063	0.00145	0.06051
Tool	Dep2Tool006	1400	0.77063	0.00158	0.05905
Tool	PlanarTool028	900	0.77103	0.00202	0.06071
Tool	Dep2Tool011	1600	0.77124	0.00145	0.05797
Tool	EtchTool010	850	0.77125	0.00213	0.06208

The report contains information about the process, including the number of units, steps, operations, and levels in the X columns. The Levels with Lowest Yields table contains counts and yield information for the 195 levels in the X columns. Notice that the PlanarTool008 tool had the lowest yield and 1,000 units went through that tool.

Launch the Process History Explorer Platform

Launch the Process History Explorer platform by selecting **Analyze > Screening > Process History Explorer**.

Figure 26.3 Process History Explorer Launch



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Launch Window Roles

ID Specifies one or more columns that identify a unit in the process.

> **Note:** The ID columns must have corresponding columns in the yield table.

X, Process Specifies one or more columns that identify the components or specific details that make up the steps of the process.

Step Optional columns that specify the steps and sub-steps that a unit goes through. For example, the layer and the type of operation within the layer.

Timestamp Specifies the time of the operation. The time can be specified as a single time point or as a pair of time points that indicate the times that the operation started and ended.

Order Optional column that specifies the order that a batch of units went through the components specified in the X, Process role.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Launch Window Options

Goal is to Minimize Y Specifies that the goal of the analysis is to minimize the response. By default, the analysis seeks to maximize the response.

Note: The response is specified in a second data table, not in the data table used to launch the platform.

Data Format

The data for Process History Explorer must be specified in two data tables. The first table is used to launch the platform and contains the steps and details of the process. The second table contains the response, or yield, for each level of the ID columns.

The sample data tables Quality Control > Lot Wafer History and Quality Control > Lot Wafer Yield provide an example of how the data are specified in two data tables.

The Process History Explorer Report

The initial Process History Explorer Report provides a summary of the information in the data table(s).

- The number of Units, determined by the ID columns. If there is more than one ID column, this is the number of unique combinations.
- The number of Steps, determined by the Step columns. If there is more than one Step column, this is the number of unique combinations.
- The number of Operations, determined by the Timestamp columns. This is the same as the number of rows in your data table.
- The number of levels, determined by the X, Process columns. If there is more than one X, Process column, this is the sum of the number of levels in each column.
- The number of responses, determined by the Yield columns.

Select options from the Process History Explorer red triangle menu to continue your analysis. See "Process History Explorer Platform Options".

Process History Explorer Platform Options

The Process History Explorer red triangle menu contains the following options:

Levels with Lowest Yield Displays a table of summary statistics for the yield for each level of each X, Process variable. The summary statistics provided for each level are the number of operations, and the mean, standard error, and standard deviation of the yield. By default, the table is sorted in ascending order by mean yield.

Levels with Lowest Yield with Time Filter Displays a table of summary statistics for the time intervals of the top 50 levels with the lowest yields. The routine searches for a time interval for each of the top 50 levels in the Levels with Lowest Yield table that minimizes the mean yield within the interval. The interval must contain at least 25% of the operations in that level. The summary statistics provided for each level are the number of operations in the identified interval, the mean yield in the identified interval, the timestamps for the first and last operations in the level (First Time and Last Time), and the timestamps for the first and last operations in the identified interval (RunStart Time and RunEnd Time).

Stepwise Regression Displays a Stepwise Regression window that enables you to specify options for a stepwise regression analysis. You can use stepwise regression to find problems in the process, where a problem is something that causes your yield to decrease. At each step in the regression, a term enters according to which term has the largest negative yield. This helps you identify process steps that most impact the yield. Stepwise Regression provides a table that shows the order in which terms entered the model at each step of the regression.

Note: If the Goal is to Minimize Y option is selected in the launch window, Stepwise Regression adjusts to detect terms that cause the yield to increase.

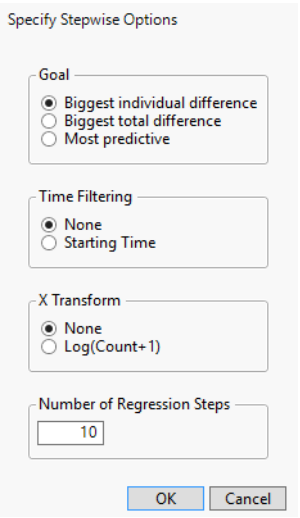
Goal Specifies the criterion used in the stepwise regression model. The Biggest individual difference option identifies the terms that individually cause the yield to change the most. The Biggest total difference option identifies the terms that cause the biggest total difference, which is the difference multiplied by the number of units affected. The Most predictive option produces a traditional stepwise regression model.

Time Filtering Enables you to use the starting time of a component in the stepwise regression model. If you choose the None option, the times are averaged over the entire time period. If you choose the Starting Time option, the component is allowed to fail at a particular starting time. The starting time is the time that maximizes the negative yield affect. If you choose the Starting Time option, the Stepwise Regression report table contains a Start Time column.

X Transform Enables you to use the regular count data or the Log(Count + 1) data for the X, Process variables in the stepwise regression. The Log(Count + 1) option transforms the variables in the regression such that high frequency counts have less of a linear influence in the regression. If you choose the None option, high frequency counts have a linear influence in the regression.

Number of Regression Steps The number of steps in the Stepwise Regression analysis. The default is ten.

Figure 26.4 Stepwise Regression Window



Waiting Time Analysis (Available only when two Timestamp variables are specified.) Each unique ID can go through only one level of a step at a time. In addition, each step can process only one ID at a time. Waiting Time Analysis provides a table of waiting time summary statistics for each ID. In this table, waiting time is the amount of time between when an ID is done with one step and when it starts another step. Additional Waiting Time Analysis tables are also provided. These tables contain waiting time summary statistics for each ID within the levels of each Step variable. In these tables, waiting time is the amount of time between when an ID is done with a particular level of a Step variable and when it goes through that same level again. The tables provide the following statistics:

Waiting Sum The sum of the waiting times for the given combination of ID and Step variable levels.

Waiting Count The number of waiting times for the given combination of ID and Step variable levels.

Waiting Mean The mean of the waiting times for the given combination of ID and Step variable levels.

Waiting Max The maximum waiting time for the given combination of ID and Step variable levels.

Transition Analysis Displays a report that helps you identify transitions from one component to another that are problematic for the resulting yield. This report can help you identify alignment and registration issues in the process. For example, some levels of the process column might be out of alignment with respect to other levels of the process column. If there is a transition problem, you would expect that items that repeat a specific level to have better yields than items that go through different levels of the process column.

When you select this option, you must choose one or more X, Process columns and one or more levels in each of those columns to include in the analysis. For each column that you specify for the analysis, a report appears that contains four tables. The first shows yields and counts for transitions from a specific level of the column to another specific level of the column. The second shows yields and counts for transitions away from specific levels of the column. The third shows yields and counts for transitions to specific levels of the column. The fourth shows yields and counts for transitions where the unit repeats a specific level of the column.

Order Interaction (Available only if an Order variable is specified in the launch window.) Displays a report of a one-way analysis of the yield on the Order column. The report contains a table for each X, Process column. In each table, the levels of the process column are listed in descending order of LogWorth of the corresponding analysis of variance model. The tables also contain the *p*-values and a heat map for each level of the process column.

Save Count Table Creates a new data table that contains counts of how many units went through each level of the X, Process variables. The count table has a row for each unique unit, determined by the ID values.

Save Log Count Table Creates a new data table that is the same format as the Count table, except with entries of Log(Count + 1).

Save Waiting Time (Available only when two Timestamp variables are specified.) Saves each table in the Waiting Time Analysis report to a separate data table.

Appendix A

Statistical Details

Predictive and Specialized Modeling

This appendix discusses measures of fit and the types of cross validation available in each platform.

Contents

Measures of Fit 547
Entropy RSquare 547
Validation in JMP Modeling 547
Validation Column Role 548
K-Fold and Holdback Validation 550

Measures of Fit

Entropy RSquare

The Entropy RSquare is computed for the training set and for the validation and test sets if validation is used. For the training set, Entropy RSquare is computed as follows:

- The specified model is fit using the training set.
- Predicted probabilities for each level of the response are obtained, based on the model.
- Using these predicted probabilities, the likelihood is computed for observations in the training set. Call this $Likelihood_Full_{Training}$.
- The reduced model (no predictors) is fit using the training set.
- The predicted probabilities for the levels of the response from the reduced model are used to compute the likelihood for observations in the training set. Call this quantity $Likelihood_Reduced_{Training}$.
- The Entropy RSquare for the training set is:

$$\text{Entropy RSquare}_{Training} = 1 - \frac{\log(Likelihood_Full_{Training})}{\log(Likelihood_Reduced_{Training})}$$

The Entropy RSquare for the validation and test sets are computed in a manner analogous to the Entropy RSquare for the training set.

Validation in JMP Modeling

Data can be partitioned into sets before modeling to avoid overfitting and to select a good predictive model. This process uses part of the original data to estimate parameters and uses the rest of the data to tune or evaluate the parameters, or do both. In JMP Pro, you can partition the data into two or three sets in the following ways:

Train and Evaluate Partitions the data into two sets, called Training and Validation. The training set is used to estimate the model parameters. The validation set is used to independently evaluate the performance of the fitted model.

Train and Tune Partitions the data into two sets, called Training and Validation. The training set is used to estimate the model parameters. The validation set is used in the model fitting algorithm to tune the model parameters and ultimately choose a model with good predictive ability. There is no independent model evaluation done in this case.

Train, Tune, and Evaluate Partitions the data into three sets, called Training, Validation, and Test. The training set is used to estimate the model parameters. The validation set is used in the model fitting algorithm to tune the model parameters and ultimately choose a model with good predictive ability. The test set is then used to independently evaluate the performance of the fitted model.

JMP PRO Validation Column Role

The Validation Column role is available only in JMP Pro. For JMP, see "Excluded Rows as Validation Holdback".

One way to create data partitions is to use the Validation Column role. The Validation Column role uses the column's values to divide the data into parts. The column is assigned using the Validation role in the platform's launch window. For information about how to create a validation column, see "Make Validation Column".

Caution: The use of a validation column is platform specific. Different platforms use the levels of the validation column differently. See notes in Table A.1.

Table A.1 Validation Column by Platform

Platform	Train & Evaluate	Train & Tune	Train, Tune, & Evaluate	Notes
Fit Model				
Fit Least Squares	Yes	No	No	If there are more than three levels, the validation column is ignored.
Stepwise Regression	No	Yes	Yes	If there are more than three levels, K-Fold Cross-Validation is used.
Logistic Regression	Yes	No	No	If there are more than three levels, the validation column is ignored.
Generalized Regression	No	Yes	Yes	If there are more than three levels, K-Fold Cross-Validation is used.
Partial Least Squares	No	Yes	Yes	If there are more than three levels, K-Fold Cross-Validation is used.
Predictive Models				
Neural	No	Yes	Yes	If there are more than three levels, K-Fold Cross-Validation is used.

Platform	Train & Evaluate	Train & Tune	Train, Tune, & Evaluate	Notes
Partition	No	Yes	Yes	If there are more than three levels, the platform only uses rows with the three smallest values.
Bootstrap Forest	No	Yes	Yes	If there are more than three levels, the platform only uses rows with the three smallest values.
Boosted Tree	No	Yes	Yes	If there are more than three levels, the platform only uses rows with the three smallest values.
K Nearest Neighbors	No	Yes	Yes	If there are more than three levels, the platform only uses rows with the three smallest values.
Naive Bayes	No	Yes	Yes	If there are more than three levels, the platform only uses rows with the three smallest values.
Support Vector Machines	Yes	Yes	Yes	If there are more than three levels, K-Fold Cross-Validation is used.
Specialized Models				
Functional Data Explorer	Yes	No	No	Must be created as a Grouped Random validation column. If there are more than two levels, the smallest value defines the training set and all other values define the validation set.
Multivariate Models				
Discriminant	Yes	Yes	Yes	If there are more than three levels, the platform only uses rows with the three smallest values.
Partial Least Squares	No	Yes	Yes	If there are more than three levels, K-Fold Cross-Validation is used.

Platform	Train & Evaluate	Train & Tune	Train, Tune, & Evaluate	Notes
Uplift	No	Yes	Yes	If there are more than three levels, the platform only uses rows with the three smallest values.

K-Fold and Holdback Validation

The Validation Column role provides a framework for partitioning data into cross validation sets. In addition, some JMP platforms also support K-Fold and various types of Holdback validation.

K-Fold Cross-Validation Divides the original data into *K* subsets. In turn, each of the *K* sets is used to validate the model fit on the rest of the data, fitting a total of *K* models. The model that produces the best validation statistic is chosen as the final model, and the fold that is not used in the building of that model provides the test set performance statistics.

Note: For some platforms, you must specify K-Fold Cross-Validation in the model control panel. For other platforms, you must specify K-Fold Cross-Validation in the platform launch window. For still other platforms, you must specify K-Fold Cross-Validation through a validation column that contains more than three levels.

Random Validation Holdback (Available as a launch option for specific platforms.) Randomly divides the original data into the training and validation sets. A test set can also be included. You can specify the proportions of the original data to use in each set.

Leave-One-Out Validation Holdback (Available as an option for specific platforms.) Repeatedly fits the model leaving out one observation at a time. Leave-one-out validation is also known as the jackknife procedure.

Excluded Rows as Validation Holdback Uses the excluded rows in the data table as a validation holdback set. For JMP Pro, this option is available by selecting in the platform preferences.

Note: For platforms that support using excluded rows as a validation holdback set, the excluded rows are used only when there is no validation column or validation proportion specified in the launch window.

Table A.2 K-Fold and Holdback Validation by Platform

Platform	Excluded Rows as Validation Holdback	Random Validation Holdback	Leave-One-Out Holdback	K-Fold Cross-Validation
Fit Model				
Fit Least Squares	No	No	No	No
Stepwise Regression	No	No	No	Yes (for continuous response models only)
Logistic Regression	No	No	No	No
JMP PRO Generalized Regression	No	Yes	Yes	Yes (though the model controls)
JMP PRO Partial Least Squares	No	Yes	Yes	Yes (through the model controls)
Predictive Model				
Neural	Yes	Yes	No	Yes (through model launch or validation column)
Partition	Yes	Yes	No	Yes (select the option in the platform preferences)
JMP PRO Bootstrap Forest	Yes	Yes	No	No
JMP PRO Boosted Tree	Yes	Yes	No	No
JMP PRO K Nearest Neighbors	Yes	Yes	No	No
JMP PRO Naive Bayes	Yes	Yes	No	No
JMP PRO Support Vector Machines	No	Yes	No	Yes (through model launch)
Specialized Models				

Platform	Excluded Rows as Validation Holdback	Random Validation Holdback	Leave-One-Out Holdback	K-Fold Cross-Validation
JMP PRO Functional Data Explorer	No	No	No	No
Multivariate Models				
Discriminant	Optional	No	No	No
Partial Least Squares	No	Yes	Yes	Yes (through model launch or validation column)
JMP PRO Uplift	No	Yes		No

Appendix B

References

Agrawal, R., and Srikant, R. (1994). "Fast Algorithms for Mining Association Rules." In *Proceedings of the 20th VLDB Conference*. Santiago, Chile: IBM Almaden Research Center. Accessed July 5, 2016. https://web.stanford.edu/class/cs345d-01/rl/ar-mining.pdf.

Baglama, J., and Reichel, L. (2005). "Augmented implicitly restarted Lanczos bidiagonalization methods." *SIAM Journal on Scientific Computing*, 27:19–42.

Bates, D. M., and Watts, D. G. (1988). *Nonlinear Regression Analysis and Its Applications*. New York: John Wiley & Sons.

Benford, F. (1938). "The law of anomalous numbers." *Proceedings of the American philosophical society*, 551–572.

Benjamini, Y., and Hochberg, Y. (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society*, *Series B* 57:289–300.

Box, G. E. P., Jenkins, G. M., and Reinsel, G. C. (1994). *Time Series Analysis: Forecasting and Control*. 3rd ed. Englewood Cliffs, NJ: Prentice-Hall.

Candes, E. J., Li, X., Ma, Y., and Wright, J. (2009). "Robust Principal Component Analysis?" *Journal of the ACM*, 58:1–37.

Cleveland, W. S. (1994). *Visualizing Data*, Summit, NJ: Hobart Press.

Conover, W. J. (1999). *Practical Nonparametric Statistics*. 3rd ed. New York: John Wiley & Sons.

Cureton, E. E. (1967). "The Normal Approximation to the Signed-Rank Sampling Distribution when Zero Differences are Present." *Journal of the American Statistical Association* 62: 319, 1068–1069.

Hahsler, M. (2015). "A Probabilistic Comparison of Commonly Used Interest Measures for Association Rules." Accessed September 17, 2018. http://michael.hahsler.net/research/association_rules/measures.html.

Hand, D. J., Mannila, H., and Smyth, P. (2001). *Principles of Data Mining*. Cambridge, MA: MIT Press.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer-Verlag.

Hawkins D. M., and Kass G. V. (1982). "Automatic Interaction Detection." In *Topics in Applied Multivariate Analysis*, edited by D. M. Hawkins, 267–300. Cambridge: Cambridge University Press.

Huber, P. J., and Ronchetti, E. M. (2009). *Robust Statistics*. 2nd ed. New York: John Wiley & Sons.

Hyndman, R. J., Koehler, A. B., Ord, J. K., and Snyder, R. D. (2008). *Forecasting with Exponential Smoothing: The State Space Approach*. Berlin: Springer-Verlag.

Jolliffe, I. T. (2002). *Principal Component Analysis*. New York: Springer-Verlag.

Kass, G. V. (1980). "An Exploratory Technique for Investigating Large Quantities of Categorical Data." *Journal of the Royal Statistical Society, Series C* 29:119–127

Lehman, E. L. (2006). *Nonparametrics: Statistical Methods Based on Ranks.* 2nd ed. New York: Springer.

Lin, Z., Chen, M., and Ma, Y. (2013). The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. arXiv preprint arXiv:1009.5055.

Mason, R. L., and Young, J. C. (2002). *Multivariate Statistical Process Control with Industrial Applications*. Philadelphia: SIAM.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models.* 2nd ed. London: Chapman & Hall.

Nagelkerke, N. J. D. (1991). "A Note on a General Definition of the Coefficient of Determination." *Biometrika* 78:691–692.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Parker, R. J. (2015). *Efficient Computational Methods for Large Spatial Data*. Ph.D. diss., Department of Statistics, North Carolina State University. Accessed June 30, 2016. https://repository.lib.ncsu.edu/ir/bitstream/1840.16/10572/1/etd.pdf.

Platt, J. (1998). *Sequential minimal optimization: A fast algorithm for training support vector machines*. Technical Report MST-TR-98-14, Microsoft Research. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-98-14.pdf.

Qian, P. Z., Huaiquing, W., and Wu, C. F. (2012). "Gaussian process models for computer experiments with qualitative and quantitative factors." *Technometrics* 50:383–396.

Ramsay, J. O., and Silverman, B. W. (2005). *Functional Data Analysis*. 2nd ed. New York: Springer.

Ratkowsky, D. A. (1990). *Handbook of Nonlinear Regression Models*. New York: Marcel Dekker.

Sall, J. (2002). "Monte Carlo Calibration of Distributions of Partition Statistics." SAS Institute Inc.,Cary, NC. Accessed July 29, 2015. https://www.jmp.com/content/dam/jmp/documents/en/white-papers/montecarlocal.pdf.

Santer, T., Williams, B., and Notz, W. (2003). *The Design and Analysis of Computer Experiments*. New York: Springer-Verlag.

SAS Institute Inc. (2020). *SAS/ETS 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/etsug/15.2/content/etsug.pdf.

Schäfer, J., and Strimmer, K. (2005). "A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics." *Statistical Applications in Genetics and Molecular Biology* 4, Article 32.

Schuirmann, D. J. (1987). "A Comparison of the Two One-sided Tests Procedure and the Power Approach for Assessing the Equivalence of Average Bioavailability." *Journal of Pharmacokinetics and Biopharmaceutics* 15:657–680.

Shiskin, J., Young, A. H., and Musgrave, J. C. (1967). *The X-11 Variant of the Census Method II Seasonal Adjustment Program.* Technical Report 15, US Department of Commerce, Bureau of the Census.

Shmueli, G., Patel, N. R., and Bruce, P. C. (2010). *Data Mining For Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Shmueli, G., Bruce, P. C., Stephens M. L., and Patel, N. R. (2017). *Data Mining For Business Intelligence: Concepts, Techniques, and Applications with JMP Pro*. Hoboken, NJ: John Wiley & Sons.

Westfall, P. H., Tobias, R. D., and Wolfinger, R. D. (2011). *Multiple Comparisons and Multiple Tests Using SAS*. 2nd ed. Cary, NC: SAS Institute Inc.

White, K., Szarka, J., and Jensen, W. (2018). "A Recommended Set of Indices for Evaluating Process Health." Presentation at the 2018 Fall Technical Conference, sponsored by the American Statistical Association and the American Society for Quality.

Appendix C

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
- You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Multivariate Methods

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Multivariate Methods*. Cary, NC: SAS Institute Inc.

JMP® 16 Multivariate Methods

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Multivariate Methods

1 Learn about JMP 13
Documentation and Additional Resources

Formatting Conventions in JMP Documentation 15
JMP Help 16
JMP Documentation Library 16
Additional Resources for Learning JMP 22
JMP Tutorials 22
Sample Data Tables 22
Learn about Statistical and JSL Terms 23
Learn JMP Tips and Tricks 23
JMP Tooltips 23
JMP User Community 24
Free Online Statistical Thinking Course 24
JMP New User Welcome Kit 24
Statistics Knowledge Portal 24
JMP Training 24
JMP Books by Users 25
The JMP Starter Window 25
JMP Technical Support 25

2 Introduction to Multivariate Analysis 27
Overview of Multivariate Techniques

3 Correlations and Multivariate Techniques 29
Explore the Multidimensional Behavior of Variables

Example of the Multivariate Platform 31
Launch the Multivariate Platform 33
The Multivariate Report 35
Multivariate Platform Options 35
Nonparametric Correlations 39
Scatterplot Matrix 40
Outlier Analysis 42
Item Reliability 45

Impute Missing Data ... 45
Example of Item Reliability ... 46
Computations and Statistical Details ... 47
Estimation Methods ... 47
Pearson Product-Moment Correlation ... 48
Nonparametric Measures of Association ... 48
Inverse Correlation Matrix ... 50
Distance Measures ... 50
Cronbach's α ... 52

4 Principal Components ... 55
Reduce the Dimensionality of Your Data
Overview of Principal Component Analysis Platform ... 57
Example of Principal Component Analysis ... 57
Launch the Principal Components Platform ... 59
Missing Data ... 61
Principal Components Report ... 62
Principal Components Report Options ... 63
Outlier Analysis ... 73
Statistical Details for the Principal Components Analysis Platform ... 75
Estimation Methods ... 75
DModX Calculation ... 76
Calculations for Outlier Analysis ... 77

5 Discriminant Analysis ... 79
Predict Classifications Based on Continuous Variables
Overview of the Discriminant Analysis Platform ... 81
Example of Discriminant Analysis ... 81
Launch the Discriminant Analysis Platform ... 83
Stepwise Variable Selection ... 85
Discriminant Methods ... 88
Shrink Covariances ... 91
Discriminant Analysis Report ... 91
Principal Components ... 92
Canonical Plot and Canonical Structure ... 93
Discriminant Scores ... 97
Score Summaries ... 98
Discriminant Analysis Options ... 100
Score Options ... 102
Canonical Options ... 104
Example of a Canonical 3D Plot ... 108

Specify Priors ... 109
Consider New Levels ... 109
Save Discrim Matrices ... 110
Scatterplot Matrix ... 110
Validation in JMP and JMP Pro ... 111
Statistical Details for the Discriminant Analysis Platform ... 112
Description of the Wide Linear Algorithm ... 112
Saved Formulas ... 112
Multivariate Tests ... 120
Approximate F-Tests ... 121
Between Groups Covariance Matrix ... 121

6 Partial Least Squares Models ... 123
Develop Models Using Correlations between Ys and Xs

Overview of the Partial Least Squares Platform ... 125
Example of Partial Least Squares ... 126
Launch the Partial Least Squares Platform ... 129
Centering and Scaling ... 133
Standardize X ... 133
Model Launch Control Panel ... 134
Partial Least Squares Options ... 135
Partial Least Squares Report ... 136
Model Comparison Summary ... 136
Cross Validation Report ... 137
Model Fit Report ... 142
Model Fit Options ... 143
Variable Importance Plot ... 145
VIP vs Coefficients Plots ... 146
Save Columns ... 146
Additional Example of Partial Least Squares ... 148
Statistical Details for the Partial Least Squares Platform ... 150
Partial Least Squares ... 150
van der Voet T^2 ... 151
T^2 Plot ... 152
Confidence Ellipses for X Score Scatterplot Matrix ... 153
Standard Error of Prediction and Confidence Limits ... 153
Standardized Scores and Loadings ... 154
PLS Discriminant Analysis (PLS-DA) ... 155

7 Multiple Correspondence Analysis 157
Identify Associations between Levels of Categorical Variables

Example of Multiple Correspondence Analysis 159
Launch the Multiple Correspondence Analysis Platform 162
The Multiple Correspondence Analysis Report 163
Multiple Correspondence Analysis Platform Options 164
Correspondence Analysis Options 166
Show Plot 167
Show Detail 167
Show Adjusted Inertia 168
Show Coordinates 168
Show Summary Statistics 169
Show Partial Contributions to Inertia 169
Show Squared Cosines 170
Cochran's Q Test 170
Cross Table 170
Additional Examples of the Multiple Correspondence Analysis Platform 171
Example Using a Supplementary Variable 172
Example Using a Supplementary ID 173
Example of Cochran's Q Test 174
Statistical Details for the Multiple Correspondence Analysis Platform 176
Details Report 176
Adjusted Inertia 177
Summary Statistics 177
Partial Contributions to Inertia 178
Cochran's Q Statistic 178

8 Structural Equation Models 179
Fit Structural Equation Models

Overview of Structural Equation Models 181
Example of a Structural Equation Model 184
Launch the Structural Equation Models Platform 187
The Structural Equation Models Report 189
Model Specification Report 189
Model Comparison Report 195
Chi-Square Difference Test Report 196
Structural Equation Model Fit Report 197
Structural Equation Models Platform Options 198
Model Options 200
Customize the SEM Path Diagram 205

Path Diagram Pop-Up Menu Options ... 205
Customize Diagram Appearance Options ... 207
Additional Examples of Structural Equation Models ... 208
Example of a Latent Path Variable Model ... 209
Example of a Latent Growth Curve Model ... 215
Example of Using the Assess Measurement Model Report ... 217
Statistical Details for the Structural Equation Models Platform ... 220
Statistical Details for Summary of Fit Measures ... 220

9 Factor Analysis ... 225
Identify Latent Variables in Your Data

Overview of the Factor Analysis Platform ... 227
Example of the Factor Analysis Platform ... 228
Launch the Factor Analysis Platform ... 230
The Factor Analysis Report ... 231
Model Launch ... 232
Rotation Methods ... 234
Factor Analysis Platform Options ... 235
Factor Analysis Model Fit Options ... 236

10 Multidimensional Scaling ... 241
Visualize Proximities among a Set of Objects

Overview of the Multidimensional Scaling Platform ... 243
Example of Multidimensional Scaling ... 243
Launch the Multidimensional Scaling Platform ... 246
The Multidimensional Scaling Report ... 247
Multidimensional Scaling Plot ... 247
Shepard Diagram ... 248
Fit Details ... 248
Multidimensional Scaling Platform Options ... 248
Waern Links ... 249
Additional Example of the Multidimensional Scaling Platform ... 251
Statistical Details for the Multidimensional Scaling Platform ... 253
Stress ... 253
Transformations ... 254
Attributes List Format ... 254

11 Item Analysis ... 255
Analyze Test Results by Item and Subject

Example of Item Analysis ... 257
Launch the Item Analysis Platform ... 260

Logistic 3PL Model Details . . . 261
Data Format . . . 261
The Item Analysis Report . . . 261
Characteristic Curves . . . 261
Information Plot . . . 262
Dual Plot . . . 263
Parameter Estimates . . . 264
Item Analysis Platform Options . . . 265
Statistical Details for the Item Analysis Platform . . . 265
Item Response Curves . . . 266
Item Response Curve Models . . . 267
IRT Model Assumptions . . . 269
Fitting the IRT Model . . . 269
Ability Formula . . . 271

12 Hierarchical Cluster . . . 273
Group Observations Using a Tree of Clusters

Overview of the Hierarchical Clustering Platform . . . 275
Overview of Platforms for Clustering Observations . . . 275
Example of Clustering . . . 277
Launch the Hierarchical Cluster Platform . . . 280
Clustering Method . . . 281
Method for Distance Calculation . . . 281
Data Structure . . . 282
Transformations to Y, Columns Variables . . . 284
Hierarchical Cluster Report . . . 286
Dendrogram Report . . . 286
Illustration of Dendrogram and Distance Graph . . . 287
Clustering History Report . . . 288
Hierarchical Cluster Options . . . 288
Additional Examples of the Hierarchical Clustering Platform . . . 292
Example of a Distance Matrix . . . 292
Example of Wafer Defect Classification Using Spatial Measures . . . 294
Statistical Details for the Hierarchical Clustering Platform . . . 296
Spatial Measures . . . 297
Distance Method Formulas . . . 299

13 K Means Cluster . . . 301
Group Observations Using Distances

Overview of the K Means Cluster Platform . . . 303
Overview of Platforms for Clustering Observations . . . 303

Example of K Means Cluster ... 305
Launch the K Means Cluster Platform ... 308
Iterative Clustering Report ... 309
Iterative Clustering Options ... 309
Iterative Clustering Control Panel ... 310
K Means Report ... 311
Cluster Comparison Report ... 311
K Means Report ... 311
K Means Report Options ... 312
Self Organizing Map ... 314
Self Organizing Map Control Panel ... 314
Self Organizing Map Report ... 315
Description of SOM Algorithm ... 315
Additional Example of K Means Cluster Platform ... 316
Example of a Self-Organizing Map ... 316

14 Normal Mixtures ... 319
Group Observations Using Probabilities

Overview of the Normal Mixtures Clustering Platform ... 321
Overview of Platforms for Clustering Observations ... 321
Example of Normal Mixtures Clustering ... 323
Launch the Normal Mixtures Clustering Platform ... 325
Model Based Clustering Report ... 326
Model Based Clustering Options ... 326
Model Based Clustering Control Panel ... 326
Normal Mixtures Report ... 327
Cluster Comparison Report ... 328
Normal Mixtures Report ... 328
Normal Mixtures Report Options ... 328
Statistical Details for the Normal Mixtures Clustering Platform ... 330

15 Latent Class Analysis ... 331
Group Observations of Categorical Variables

Overview of the Latent Class Analysis Platform ... 333
Example of Latent Class Analysis ... 333
Launch the Latent Class Analysis Platform ... 337
The Latent Class Analysis Report ... 338
Cluster Comparison Report ... 338
Latent Class Model Report ... 338
Latent Class Analysis Platform Options ... 341
Latent Class Analysis Options ... 341

Latent Class Model Options ... 341
Additional Example of the Latent Class Analysis Platform ... 342
Plot Probabilities of Cluster Membership ... 342
Statistical Details for the Latent Class Analysis Platform ... 344
Latent Class Model Fit ... 344
Maximum Number of Clusters ... 345

16 Cluster Variables ... 347
Group Similar Variables into Representative Groups
Overview of the Cluster Variables Platform ... 349
Example of the Cluster Variables Platform ... 349
Launch the Cluster Variables Platform ... 351
The Cluster Variables Report ... 351
Color Map on Correlations ... 352
Cluster Summary ... 352
Cluster Members ... 353
Standardized Components ... 353
Cluster Variables Platform Options ... 353
Additional Examples of the Cluster Variables Platform ... 354
Example of Color Map on Correlations ... 354
Example of Cluster Variables Platform for Dimension Reduction ... 356
Statistical Details for the Cluster Variables Platform ... 359
Variable Clustering Algorithm ... 359

A Statistical Details ... 361
Multivariate Methods
Wide Linear Methods and the Singular Value Decomposition ... 363
The Singular Value Decomposition ... 363
The SVD and the Covariance Matrix ... 364
The SVD and the Inverse Covariance Matrix ... 364
Calculating the SVD ... 365

B References ... 367

C Technology License Notices ... 373

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 15
JMP Help . . . 16
JMP Documentation Library . . . 16
Additional Resources for Learning JMP . . . 22
JMP Tutorials . . . 22
Sample Data Tables . . . 22
Learn about Statistical and JSL Terms . . . 23
Learn JMP Tips and Tricks . . . 23
JMP Tooltips . . . 23
JMP User Community . . . 24
Free Online Statistical Thinking Course . . . 24
JMP New User Welcome Kit . . . 24
Statistics Knowledge Portal . . . 24
JMP Training . . . 24
JMP Books by Users . . . 25
The JMP Starter Window . . . 25
JMP Technical Support . . . 25

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction to Multivariate Analysis

Overview of Multivariate Techniques

Multivariate Methods describes the following techniques for analyzing several variables simultaneously:

- The Multivariate platform examines multiple variables to see how they relate to each other. See "Correlations and Multivariate Techniques".
- The Principal Components platform derives a small number of independent linear combinations (principal components) of a set of measured variables that capture as much of the variability in the original variables as possible. It is a useful exploratory technique and can help you create predictive models. See "Principal Components".
- The Discriminant platform looks to find a way to predict a classification (X) variable (nominal or ordinal) based on known continuous responses (Y). It can be regarded as inverse prediction from a multivariate analysis of variance (MANOVA). See "Discriminant Analysis".
- The Partial Least Squares platform fits linear models based on factors, namely, linear combinations of the explanatory variables (Xs). PLS exploits the correlations between the Xs and the Ys to reveal underlying latent structures. See "Partial Least Squares Models".
- The Multiple Correspondence Analysis (MCA) platform takes multiple categorical variables and seeks to identify associations between levels of those variables. MCA is frequently used in the social sciences particularly in France and Japan. It can be used in survey analysis to identify question agreement. See "Multiple Correspondence Analysis".
- JMP PRO The Structural Equation Models platform enables you to fit a variety of models, including confirmatory factor analysis, path models with or without latent variables, measurement error models, and latent growth curve models. See "Structural Equation Models".
- The Factor Analysis platform enables you to construct factors from a larger set of observed variables. These factors are expressed as linear combinations of a subset of the observed variables. Factor analysis enables you to explore the number of factors that are explained by a set of measured, observed variables, and the strength of the relationship between factors and variables. See "Factor Analysis".
- The Multidimensional Scaling (MDS) platform enables you to create a visual representation of the pattern of proximities (similarities, dissimilarities, or distances) among a set of objects. See "Multidimensional Scaling".
- The Item Analysis platform enables you to fit item response theory models. The Item Response Theory (IRT) method is used for the analysis and scoring of measurement

instruments such as tests and questionnaires. IRT uses a system of models to relate a trait or ability to an individual's probability of endorsing or correctly responding to an item. IRT can be used to study standardized tests, cognitive development, and consumer preferences. See "Item Analysis".

- The Hierarchical Cluster platform groups rows together that share similar values across a number of variables. It is a useful exploratory technique to help you understand the clumping structure of your data. See "Hierarchical Cluster".
- The KMeans Clustering platform groups observations that share similar values across a number of variables. See "K Means Cluster".
- The Normal Mixtures platform enables you to cluster observations when your data come from overlapping normal distributions. See "Normal Mixtures".
- The Latent Class Analysis platform finds clusters of observations for categorical response variables. The model takes the form of a multinomial mixture model. See "Latent Class Analysis".
- The Cluster Variables platform groups similar variables into representative groups. You can use Cluster Variables as a dimension-reduction method. Instead of using a large set of variables in modeling, the cluster components of the most representative variable in the cluster can be used to explain most of the variation in the data. See "Cluster Variables".

Chapter 3

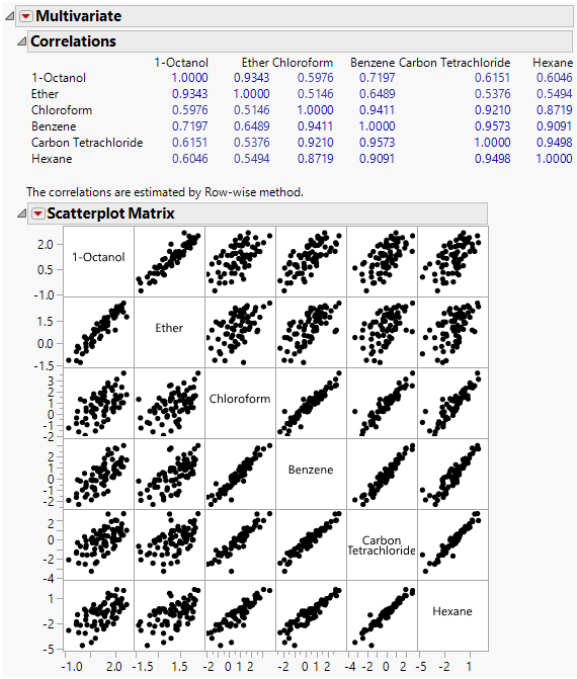
Correlations and Multivariate Techniques

Explore the Multidimensional Behavior of Variables

Multivariate data involve many variables instead of one (univariate) or two (bivariate). Use the Multivariate platform to explore how multiple variables relate to each other. The Multivariate platform provides many techniques to summarize and test the strength of the linear relationship between each pair of response variables. Both parametric and nonparametric correlations tests are available in the platform. You can also use graphical features, such as the Scatterplot Matrix and Color Maps, to identify dependencies, outliers, and clusters among the variables.

There are additional multivariate analysis techniques to further examine the relationship between variables, including principal components analysis, outlier analysis, and item reliability. These techniques are available through the Multivariate report. You can also use the Principal Components Analysis and Outlier Analysis platforms in JMP for more in-depth implementations of these techniques.

Figure 3.1 Example of a Multivariate Report



Contents

Example of the Multivariate Platform 31
Launch the Multivariate Platform. 33
The Multivariate Report 35
Multivariate Platform Options 35
Nonparametric Correlations 39
Scatterplot Matrix 40
Outlier Analysis. 42
Item Reliability 45
Impute Missing Data 45
Example of Item Reliability 46
Computations and Statistical Details 47
Estimation Methods 47
Pearson Product-Moment Correlation 48
Nonparametric Measures of Association. 48
Inverse Correlation Matrix. 50
Distance Measures 50
Cronbach's α 52

Example of the Multivariate Platform

This example uses the Body Measurements.jmp sample data table to examine the relationship between different measurements on the body.

1. Select **Help > Sample Data Library** and open Body Measurements.jmp.
2. Select **Analyze > Multivariate Methods > Multivariate**.
3. Select all columns except Mass and click **Y, Columns**.
4. Click **OK**.

   The initial multivariate report contains a correlation matrix and scatterplot matrix. There is also a note that tells you that the Row-wise estimation method was used. All of the variables are positively correlated, but at varying strengths.
5. Click the Multivariate red triangle and deselect **Scatterplot Matrix**.
6. Click the Multivariate red triangle and select **Correlation Probability**.
7. Click the Multivariate red triangle and select **Color Maps > Color Map on Correlations**.

Figure 3.2 Multivariate Report for Body Measurements Data

Multivariate

Correlations

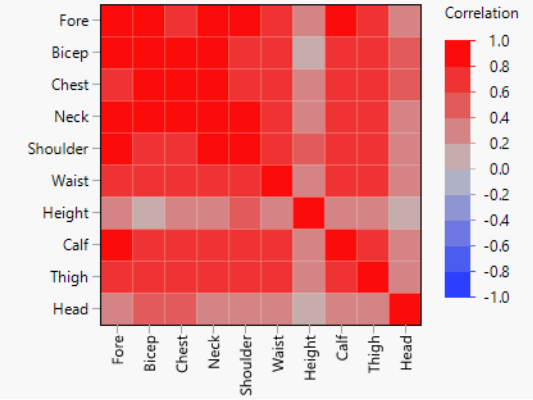
	Fore	Bicep	Chest	Neck	Shoulder	Waist	Height	Calf	Thigh	Head
Fore	1.0000	0.8134	0.7671	0.8134	0.8670	0.7577	0.3537	0.8130	0.7877	0.3029
Bicep	0.8134	1.0000	0.8552	0.8240	0.7202	0.6652	0.0657	0.6421	0.6974	0.4199
Chest	0.7671	0.8552	1.0000	0.8279	0.7999	0.7154	0.2017	0.6436	0.7802	0.4379
Neck	0.8134	0.8240	0.8279	1.0000	0.8312	0.7530	0.3912	0.6095	0.7055	0.3223
Shoulder	0.8670	0.7202	0.7999	0.8312	1.0000	0.7462	0.4839	0.6847	0.6825	0.2258
Waist	0.7577	0.6652	0.7154	0.7530	0.7462	1.0000	0.3513	0.6893	0.7403	0.2997
Height	0.3537	0.0657	0.2017	0.3912	0.4839	0.3513	1.0000	0.3977	0.2860	0.1576
Calf	0.8130	0.6421	0.6436	0.6095	0.6847	0.6893	0.3977	1.0000	0.7497	0.2742
Thigh	0.7877	0.6974	0.7802	0.7055	0.6825	0.7403	0.2860	0.7497	1.0000	0.2363
Head	0.3029	0.4199	0.4379	0.3223	0.2258	0.2997	0.1576	0.2742	0.2363	1.0000

The correlations are estimated by Row-wise method.

Correlation Probability

	Fore	Bicep	Chest	Neck	Shoulder	Waist	Height	Calf	Thigh	Head
Fore	<.0001	<.0001	<.0001	<.0001	<.0001	<.0001	0.1064	<.0001	<.0001	0.1706
Bicep	<.0001	<.0001	<.0001	<.0001	0.0002	0.0007	0.7715	0.0013	0.0003	0.0517
Chest	<.0001	<.0001	<.0001	<.0001	<.0001	0.0002	0.3681	0.0012	<.0001	0.0415
Neck	<.0001	<.0001	<.0001	<.0001	<.0001	<.0001	0.0718	0.0026	0.0002	0.1436
Shoulder	<.0001	0.0002	<.0001	<.0001	<.0001	<.0001	0.0225	0.0004	0.0005	0.3124
Waist	<.0001	0.0007	0.0002	<.0001	<.0001	<.0001	0.1089	0.0004	<.0001	0.1753
Height	0.1064	0.7715	0.3681	0.0718	0.0225	0.1089	<.0001	0.0668	0.1969	0.4836
Calf	<.0001	0.0013	0.0012	0.0026	0.0004	0.0004	0.0668	<.0001	<.0001	0.2168
Thigh	<.0001	0.0003	<.0001	0.0002	0.0005	<.0001	0.1969	<.0001	<.0001	0.2898
Head	0.1706	0.0517	0.0415	0.1436	0.3124	0.1753	0.4836	0.2168	0.2898	<.0001

Color Map on Correlations

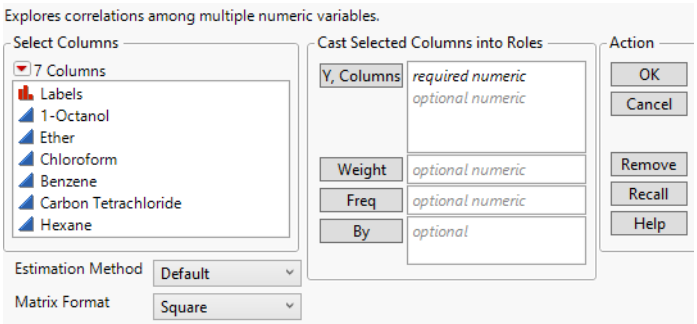


The Color Map on Correlations report provides a more concise version of the information in the Correlation Matrix. The majority of the plot is dark red colors indicating that most of the variables are highly correlated. The two lighter colored rows and columns indicate that height and head measurements are not highly correlated with other variables. This is further supported by the mostly non-significant *p*-values for Height and Head in the Correlation Probability table.

Launch the Multivariate Platform

Launch the Multivariate platform by selecting **Analyze > Multivariate Methods > Multivariate**.

Figure 3.3 The Multivariate Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns Identifies one or more response columns. The response columns must have a numeric data type, but the modeling type can be continuous or ordinal.

> **Note:** If you specify an ordinal response variable, a JMP alert appears after you click OK in the launch window. The alert indicates which variables are ordinal and confirms that you intended to include the ordinal variables in the analysis.

Weight Identifies one column whose numeric values assign a weight to each row in the analysis.

Freq Identifies one column whose numeric values assign a frequency to each row in the analysis.

By Produces a separate report of each level of By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Estimation Method Specifies the method for calculating the correlations. REML and Pairwise are the methods used most frequently. Several of these methods address the treatment of missing data. You can also estimate missing values by using a method other than Row-wise and then selecting the Impute Missing Data command. See "Impute Missing Data".

Default The Default option uses either the Row-wise, Pairwise, or REML methods.

– Row-wise estimation is used for data tables with no missing values.

– Pairwise estimation is used for data tables with missing values and either more than 10 columns, more than 5,000 rows, or more columns than rows.
– REML estimation is used otherwise.

Note: When the Default option would otherwise result in REML estimation, but the fit does not converge properly, the platform reverts to the Pairwise method. This can happen when there are missing values in your data table and at least one of the following situations applies: if your data table has fewer than 10 columns, fewer than 5,000 rows, or fewer columns than rows. If the estimation method shown is Pairwise, this means that the REML fit did not converge.

REML Restricted maximum likelihood (REML) estimation uses all of the data, even if missing values are present. Due to a bias-correction factor, this method is slow if the data set is large and there are many missing values. Therefore, REML is most useful for smaller data sets. If there are no missing cells in the data, then the REML and ML estimates are equivalent and equal to the sample covariance matrix. If there are missing cells, REML's variance and covariance estimates are less biased than the estimates from ML estimation. For statistical details, see "REML".

ML Maximum likelihood (ML) estimation uses all of the data, even if missing values are present. Because the estimates from ML are generated quickly, this method is most useful for large data tables with missing data.

Robust Robust estimation uses all of the data, even if missing values are present. This method down-weights extreme values and is therefore useful for data tables that might have outliers. For statistical details, see "Robust".

Row-wise Row-wise estimation calculates the Pearson correlation for each pair of columns. For statistical details, see "Pearson Product-Moment Correlation". Row-wise estimation does not use rows with missing values. This method is useful for excluding observations with missing data.

Pair-wise Pair-wise estimation uses all of the data, even if missing values are present. This estimation method calculates Pearson correlations for each pair of columns using all observations with nonmissing values for those two columns. For statistical details, see "Pearson Product-Moment Correlation". Pair-wise estimation is most useful when a data table has missing values and either more columns than rows, more than 10 columns, or more than 5,000 rows.

Note: If you select REML, ML, or Robust and your data table contains more columns than rows and has missing values, JMP switches the Estimation Method to Pairwise.

Matrix Format Select a format option for the Scatterplot Matrix. The Square option displays plots for all ordered combinations of columns. Lower Triangular displays plots below the

diagonal, with the first n - 1 columns on the horizontal axis. Upper Triangular displays plots above the diagonal, with the first n - 1 columns on the vertical axis.

The Multivariate Report

The default multivariate report shows the standard correlation matrix, the scatterplot matrix, and a note that indicates which method was used to estimate the correlations. In some cases, information that explains why the given method was used is also displayed. The platform menu lists additional correlation options and other techniques for looking at multiple variables. See "Multivariate Platform Options".

Multivariate Platform Options

Note: The correlation values and p-values in the Multivariate report are formatted using the conditional formatting settings in Preferences. See *Using JMP*.

The Multivariate red triangle menu contains the following options:

Correlations Multivariate Shows or hides the Correlations table. The table is a matrix of correlation coefficients that summarizes the strength of the linear relationships between each pair of response (Y) variables. This option is on by default. See "Pearson Product-Moment Correlation".

Note: This correlation matrix is calculated by the method that you select in the launch window.

Correlation Probability Shows or hides the Correlation Probability table. The table is a matrix of p-values. Each p-value corresponds to a test of the null hypothesis that the true correlation between the variables is zero. This is a test of no linear relationship between the two response variables.

CI of Correlations Shows or hides the two-tailed confidence intervals of the correlations.

Tip: The default confidence coefficient is 95%. Use the Set α Level option to change the confidence coefficient.

Inverse Correlations Shows or hides the Inverse Corr report, which is the inverse correlation matrix. The diagonal elements of the matrix are a function of how closely the variable is a linear function of the other variables. In the inverse correlation table, the diagonal is the value of $1/(1 - R^2)$. R^2 is calculated from the simple linear model regressing that variable

on all the other variables. If the multiple correlation is zero, the diagonal inverse element is 1. If the multiple correlation is 1, then the inverse element becomes infinite and is reported missing. For statistical details about inverse correlations, see the "Inverse Correlation Matrix".

Partial Correlations Shows or hides the Partial Corr report, which is the partial correlation matrix. The partial correlation matrix shows the measure of the relationship between a pair of variables after adjusting for the effects of all the other variables. The table is the negative of the inverse correlation matrix, scaled to unit diagonal. This means that the matrix is scaled such that the diagonal elements are equal to one.

Partial Correlation Probability Shows or hides the Partial Correlation Probability report, which is a matrix of *p*-values. Each *p*-value corresponds to a test of the null hypothesis that the true partial correlation between the variables is zero. This is a test of no linear relationship between the two response variables, after adjusting for the effects of the other variables.

Note: The Partial Correlation Probability option is not available when there are not enough degrees of freedom. This can occur when there are more variables than observations.

Covariance Matrix Shows or hides the covariance matrix, which measures the degree to which a pair of variables change together.

Pairwise Correlations Shows or hides the Pairwise Correlations table, which lists the Pearson product-moment correlations for each pair of Y variables. The correlations are calculated by the pairwise deletion method. The count values differ if any pair has a missing value for either variable. The Pairwise Correlations report also shows significance probabilities and compares the correlations in a bar chart. All results are based on the pairwise method.

Note: This option excludes rows that are missing for either of the variables under consideration.

Hotelling's T^2 Test Enables you to conduct a one-sample test for the mean of the multivariate distribution of the variables that you entered as Y. When you select the Hotelling's T^2 Test option, a window appears that enables you to specify the mean vector under the null hypothesis. Enter a hypothesized mean for each variable. The test assumes multivariate normality of the Y variables. The Hotelling's T^2 Test report gives the following information:

Variable The variables entered as Y.

Mean The sample mean for each variable.

Hypothesized Mean The null hypothesis means that you specified.

Test Statistic The value of Hotelling's T^2 statistic.

F Ratio The value of the test statistic. If you have n rows and k variables, the F ratio is defined as follows:

$$\frac{n-k}{k(n-1)}T^2$$

Prob > F The p-value for the test. Under the null hypothesis the F ratio has an F distribution with k and n - k degrees of freedom.

Note: To remove a report, click the red triangle next to Hotelling's T^2 Test and select **Remove Test**.

Simple Statistics This menu contains two options that each show or hide simple statistics (mean, standard deviation, sum, minimum, and maximum) for each column. The univariate and multivariate simple statistics can differ when there are missing values present, or when the Robust method is used.

Univariate Simple Statistics Shows statistics that are calculated on each column, regardless of values or missing values in other columns. These values match those produced by the Distribution platform.

Multivariate Simple Statistics Shows statistics that correspond to the estimation method selected in the launch window and the presence or absence of missing data. If there are no missing observations, this option is available only for the Robust method. If there are missing observations, this option is available for all estimation methods except for Pairwise. For the REML, ML, or Robust methods, the mean vector and covariance matrix are estimated by the selected method. For the Row-wise method, all rows with at least one missing value are excluded from the calculation of means and variances.

Nonparametric Correlations This menu contains three nonparametric measures: Spearman's Rho, Kendall's Tau, and Hoeffding's D. Each option shows or hides a nonparametric report for the corresponding measure. See "Nonparametric Correlations".

Set α Level You can specify any alpha value for the correlation confidence intervals. Four alpha values are listed: 0.01, 0.05, 0.10, and 0.50. Select **Other** to enter any other value.

Scatterplot Matrix Shows or hides a scatterplot matrix of each pair of response variables. This option is on by default. See "Scatterplot Matrix".

Color Maps The Color Maps menu contains three types of color maps. Each option in the menu shows or hides a color map of the corresponding type. The following types of color maps are available:

Color Map On Correlations Produces a cell plot that shows the correlations among variables on a scale from red (+1) to blue (-1).

Color Map On p-values Produces a cell plot that shows the significance of the correlations on a scale from $p = 0$ (red) to $p = 1$ (blue).

Cluster the Correlations Produces a cell plot that clusters together similar variables. The correlations are the same as for Color Map on Correlations, but the positioning of the variables might be different.

Color Map on Pairwise Correlations Produces a cell plot that shows the pairwise correlations among variables on a scale from red (+1) to blue (-1).

Color Map on Spearman's ρ Produces a cell plot that shows the Spearman's ρ nonparametric correlations among the variables on a scale from red (+1) to blue (-1).

Color Map on Kendall's τ Produces a cell plot that shows the Kendall's τ nonparametric correlations among the variables on a scale from red (+1) to blue (-1).

Color Map on Hoeffding's D Produces a cell plot that shows the Hoeffding's D nonparametric correlations among the variables on a scale from red (+1) to blue (-1).

Parallel Coord Plot Shows or hides a parallel coordinate plot of the variables.

Ellipsoid 3D Plot Shows or hides a three-dimensional scatterplot with a 95% confidence ellipsoid. You are prompted to specify three variables and their corresponding axes when you select this option.

Principal Components This menu contains options to show or hide a principal components report. You can select a principal components report based on correlations, covariances, or unscaled values. Selecting one of these options when a principal components report is already shown changes the report to the new option. Select **None** to remove the report.

Principal components is a technique that calculates linear combinations of the original variables. The first principal component has maximum variation, the second principal component has the next most variation, subject to being orthogonal to the first, and so on. See "Principal Components".

Outlier Analysis This menu contains options that each show or hide plots that measure distance in the multivariate sense using one of the following methods: the Mahalanobis distance, jackknife distances, and the T^2 statistic. See "Outlier Analysis".

Item Reliability This menu contains options that each show or hide an item reliability report. The reports indicate how consistently a set of instruments measures an overall response, using either Cronbach's α or Standardized α. See "Item Reliability".

Impute Missing Data (Available only if the data table contains missing values.) Produces a new data table that duplicates your data table and replaces all missing values with estimated values. See "Impute Missing Data".

Save Imputed Formulas (Available only if the data table contains missing values.) For columns that contain missing values, saves new columns to the data table that contain the formulas used to estimate the missing values. The new columns are called Imputed_<Column Name>.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Nonparametric Correlations

The Nonparametric Correlations menu offers three nonparametric measures for pairwise correlations. Each Nonparametric correlation report gives the significance probability for the chosen measure of association and displays the association value on a bar chart. There are three nonparametric correlation measures.

Spearman's Rho A correlation coefficient computed on the ranks of the data values instead of on the values themselves.

Kendall's Tau Based on the number of concordant and discordant pairs of observations. A pair is *concordant* if the observation with the larger value of X also has the larger value of Y. A pair is *discordant* if the observation with the larger value of X has the smaller value of Y. There is a correction for tied pairs, which are pairs of observations that have equal values of *X* or equal values of *Y*.

Hoeffding's D A statistical scale that ranges from –0.5 to 1. Large positive values indicate dependence. The statistic approximates a weighted sum over observations of chi-square statistics for two-by-two classification tables. The two-by-two tables are made by setting each data value as the threshold. This statistic detects more general departures from independence.

Note: The nonparametric correlations are calculated using the Pairwise method, even if you selected a different Estimation Method in the launch window.

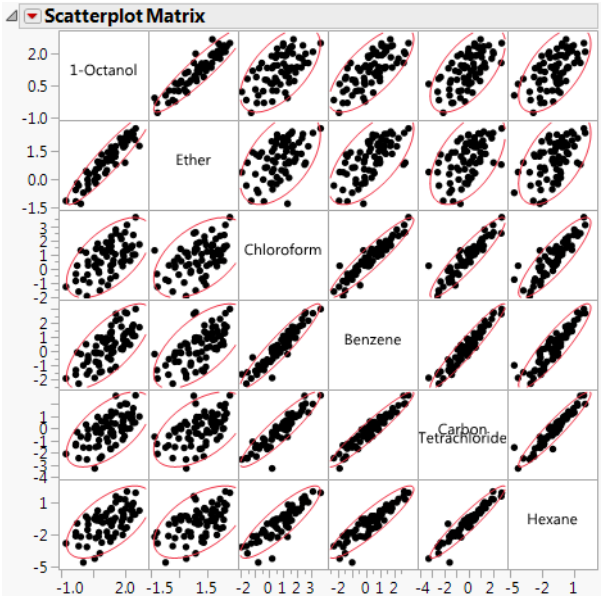
Note: When a Weight variable is specified, missing and zero-valued weights are excluded from the nonparametric correlation calculations. All other weight values are treated as 1.

For statistical details about these three methods, see the "Nonparametric Measures of Association".

Scatterplot Matrix

A scatterplot matrix helps you visualize the correlations between each pair of response variables.

Figure 3.4 Scatterplot Matrix



Turning on the Density Ellipses option shows a 95% bivariate normal density ellipse in each scatterplot. Assuming that each pair of variables has a bivariate normal distribution, this ellipse encloses approximately 95% of the points. The narrowness of the ellipse reflects the degree of correlation of the variables. If the ellipse is fairly round and is not diagonally oriented, the variables are uncorrelated. If the ellipse is narrow and diagonally oriented, the variables are correlated.

Tips:

- Re-sizing any cell resizes all the cells.
- Drag a label cell to another label cell to reorder the matrix.
- When you look for patterns in the scatterplot matrix, you can see the variables cluster into groups based on their correlations. Figure 3.4 shows two clusters of correlations: the first two variables (top, left), and the next four (bottom, right).

Scatterplot Matrix Options

The red triangle menu for the Scatterplot Matrix lets you customize the matrix with color and density ellipses and by setting the α-level.

Show Points Shows or hides the points in the scatterplot.

Fit Line Shows or hides the regression line and 95% level confidence curves for the fitted regression line.

Density Ellipses Shows or hides the 95% density ellipses in the scatterplots. Use the Ellipse α menu to change the α-level.

Shaded Ellipses Colors each ellipse. Use the Ellipses Transparency and Ellipse Color menus to change the transparency and color.

Show Correlations Shows or hides the correlation of each pair of variables in the upper left corner of each scatterplot.

Matrix Options (Available only when the Square Matrix Format is selected on the launch window.) Shows a submenu of options to change the appearance of the upper right triangle of the scatterplot matrix. Only one of the following options can be selected at a time.

Significance Circles Shows or hides correlation circles in the upper right triangle of the scatterplot matrix. The color of each circle represents the correlation between each pair of variables on a scale from red (+1) to blue (-1). The size of each circle represents the significance test between the variables. A larger circle indicates a more significant relationship.

Heat Map Shows or hides a correlation heat map in the upper right triangle of the scatterplot matrix. The color of each cell in the heat map represents the correlation between each pair of variables on a scale from red (+1) to blue (-1).

Show Histograms Shows or hides horizontal or vertical histograms in the label cells. Once histograms have been added, select **Show Counts** to label each bar of the histogram with its count. Select **Horizontal** or **Vertical** to either change the orientation of the histograms or remove the histograms.

Ellipse α Sets the α-level used for the ellipses. Select one of the standard α-levels in the menu, or select **Other** to enter a different value.

Ellipses Transparency Sets the transparency of the ellipses if they are colored. Select one of the default levels, or select **Other** to enter a different value. The default value is 0.2.

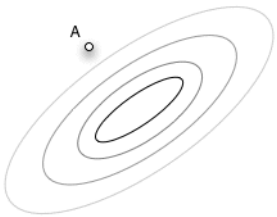
Ellipse Color Sets the color of the ellipses if they are colored. Select one of the colors in the palette, or select **Other** to use another color. The default value is red.

Nonpar Density Shows or hides shaded density contours based on a smooth nonparametric bivariate surface that describes the density of data points. Contours for the 10% and 50% quantiles of the nonparametric surface are shown.

Outlier Analysis

The Outlier Analysis menu contains options that each show or hide plots that measure distance in the multivariate sense, with respect to the correlation structure. For example, in Figure 3.5, Point A is an outlier because it is outside the correlation structure, even though it is not an outlier in any of the coordinate directions.

Figure 3.5 Example of an Outlier



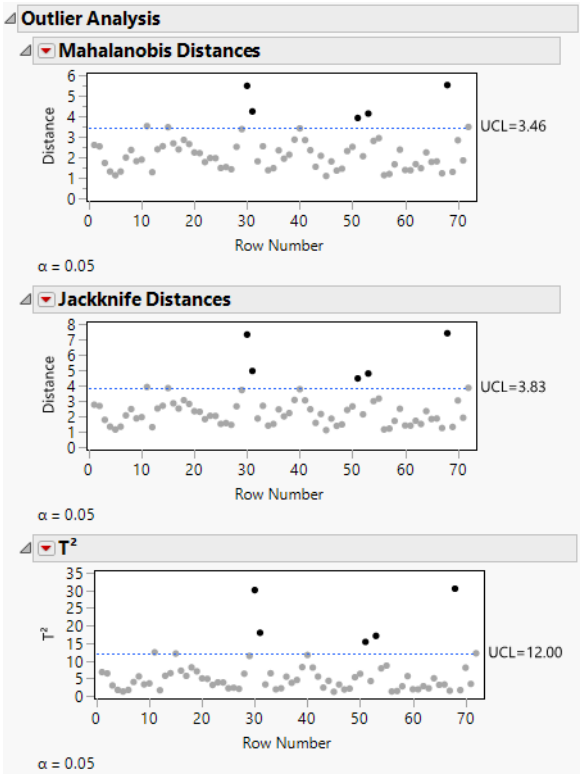
Each option in the Outlier Analysis menu shows or hides a plot of the corresponding distances. Testing is done at the alpha level that appears at the bottom of the plot. The following options are available:

Mahalanobis Distances Shows or hides the Mahalanobis distance of each point from the multivariate mean (centroid). The standard Mahalanobis distance depends on estimates of the mean, standard deviation, and correlation for the data. The distance is plotted for each observation number. Extreme multivariate outliers can be identified by highlighting the points with the largest distance values. See "Mahalanobis Distance Measures".

Jackknife Distances Shows or hides distances that are calculated using a jackknife technique. The distance for each observation is calculated with estimates of the mean, standard deviation, and correlation matrix that do not include the observation itself. The jack-knifed distances are useful when there is an outlier. In this case, the Mahalanobis distance is distorted and tends to disguise the outlier or make other points look more outlying than they are. See "Jackknife Distance Measures".

T^2 Shows or hides distances that are the square of the Mahalanobis distance. This plot is preferred for multivariate control charts. The plot includes the value of the calculated T^2 statistic, as well as its upper control limit. Values that fall outside this limit might be outliers. See "T^2 Distance Measures".

Figure 3.6 Outlier Analysis Plots



Saving Distances and Values

You can save any of the distances to the data table by selecting the **Save** option from the red triangle menu for the plot.

Note: There is no formula saved with the jackknife distance column. This means that the distance is *not* recomputed if you modify the data table. If you add or delete columns, or change values in the data table, select **Analyze > Multivariate Methods > Multivariate** again to compute new jackknife distances.

In addition to saving the distance values for each row, a column property is created that holds the upper control limit (UCL) value for the Outlier Analysis type specified.

Item Reliability

Item reliability indicates how consistently a set of instruments measures an overall response. Cronbach's α (Cronbach 1951) is one measure of reliability. Two primary applications for Cronbach's α are industrial instrument reliability and questionnaire analysis.

Cronbach's α is based on the average correlation of items in a measurement scale. It is equivalent to computing the average of all split-half correlations in the data table. The Standardized α can be requested if the items have variances that vary widely.

Note: Cronbach's α is not related to a significance level α. Also, item reliability is unrelated to survival time reliability analysis.

To look at the influence of an individual item, JMP excludes it from the computations and shows the effect of the Cronbach's α value. If α increases when you exclude a variable (item), that variable is not highly correlated with the other variables. If the α decreases, you can conclude that the variable is correlated with the other items in the scale.

See "Cronbach's α" for more information about computations.

Impute Missing Data

To impute missing data, click the Multivariate red triangle and select **Impute Missing Data**. A new data table is created. This data table is a duplicate of your data table except that the missing values are replaced with estimated values.

Imputed values are expectations conditional on the nonmissing values for each row. The mean and covariance matrix, estimated by the method chosen in the launch window, are used for the imputation calculation. All multivariate tests and options are then available for the imputed data set.

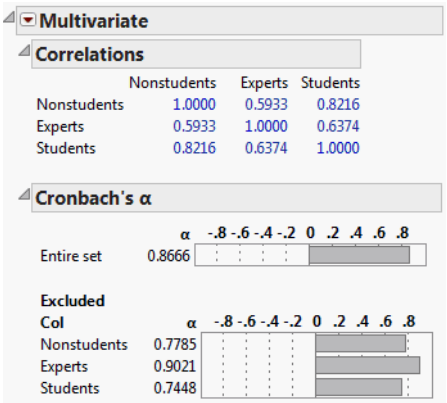
Example of Item Reliability

This example uses the Danger.jmp data in the sample data folder. This table lists 30 items having some level of inherent danger. Three groups of people (students, non-students, and experts) ranked the items according to perceived level of danger. Note that Nuclear power is rated as very dangerous (1) by both students and non-students, but is ranked low (20) by experts. On the other hand, motorcycles are ranked either fifth or sixth by all three judging groups.

You can use Cronbach's α to evaluate the agreement in the perceived way the groups ranked the items. Note that in this type of example, where the values are the same set of ranks for each group, standardizing the data has no effect.

1. Select **Help > Sample Data Library** and open Danger.jmp.
2. Select **Analyze > Multivariate Methods > Multivariate**.
3. Select all the columns except for Activity and click **Y, Columns**.
4. Click **OK**.
5. Click the Multivariate red triangle and select **Item Reliability > Cronbach's α**.
6. (Optional) Click the Multivariate red triangle and select **Scatterplot Matrix** to hide that plot.

Figure 3.7 Cronbach's α Report



The Cronbach's α results show an overall α of 0.8666, which indicates a high correlation of the ranked values among the three groups. Further, when you remove the experts from the analysis, the Nonstudents and Students ranked the dangers nearly the same, with Cronbach's α scores of 0.7785 and 0.7448, respectively.

Computations and Statistical Details

- "Estimation Methods"
- "Pearson Product-Moment Correlation"
- "Nonparametric Measures of Association"
- "Inverse Correlation Matrix"
- "Distance Measures"
- "Cronbach's α"

Estimation Methods

REML

REML (restricted maximum likelihood) estimates are less biased than the ML (maximum likelihood) estimation method when the data contains missing values. The REML method maximizes marginal likelihoods based on error contrasts. The REML method is often used for estimating variances and covariances. The REML method in the Principal Components platform is the same as the REML estimation of mixed models for repeated measures data with an unstructured covariance matrix. See the MIXED Procedure chapter in SAS Institute Inc. (2020e) about REML estimation of mixed models.

Robust

This method essentially ignores any outlying values by substantially down-weighting them. A sequence of iteratively reweighted fits of the data is done using the following weight:

$w_i = 1.0$ if $Q < K$ and $w_i = K/Q$ otherwise

Here, K is a constant equal to the 0.75 quantile of a chi-square distribution with the degrees of freedom equal to the number of columns in the data table. Q is defined as follows:

$$Q = (y_i - \mu)^T (S^2)^{-1} (y_i - \mu)$$

In this equation, y_i = the response for the i^{th} observation, μ = the current estimate of the mean vector, S^2 = current estimate of the covariance matrix, and T = the transpose matrix operation. The final step is a bias reduction of the variance matrix.

The trade off of this method is that you can have higher variance estimates when the data do not have many outliers, but can have a much more precise estimate of the variances when the data do have outliers.

Pearson Product-Moment Correlation

The Pearson product-moment correlation coefficient measures the strength of the linear relationship between two variables. For response variables X and Y, it is denoted as r and computed as follows:

$$r = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2}\sqrt{\sum(y-\bar{y})^2}}$$

If there is an exact linear relationship between two variables, the correlation is 1 or –1, depending on whether the variables are positively or negatively related. If there is no linear relationship, the correlation tends toward zero.

Nonparametric Measures of Association

For the Spearman, Kendall, or Hoeffding correlations, the data are first ranked. Computations are then performed on the ranks of the data values. Average ranks are used in case of ties.

Note: When a Weight variable is specified, missing and zero-valued weights are excluded from the nonparametric correlation calculations. All other weight values are treated as 1.

Spearman's ρ (rho) Coefficients

Spearman's ρ correlation coefficient is computed on the ranks of the data using the formula for the Pearson's correlation previously described.

Kendall's τ_b Coefficients

Kendall's τ_b coefficients are based on the number of concordant and discordant pairs. A pair of rows for two variables is *concordant* if they agree in which variable is greater. Otherwise, they are discordant, or tied.

The formula

$$\tau_b = \frac{\sum_{i<j} \operatorname{sgn}(x_i - x_j)\operatorname{sgn}(y_i - y_j)}{\sqrt{(T_0 - T_1)(T_0 - T_2)}}$$

computes Kendall's τ_b where:

$$T_0 = (n(n-1))/2$$
$$T_1 = \sum((t_i)(t_i-1))/2$$
$$T_2 = \sum((u_i)(u_i-1))/2$$

Note the following:

- The sgn(z) is equal to 1 if z>0, 0 if z=0, and –1 if z<0.
- The t_i (the u_i) are the number of tied x (respectively y) values in the ith group of tied x (respectively y) values.
- The n is the number of observations.
- Kendall's τ_b ranges from –1 to 1. If a weight variable is specified, it is ignored.

Computations proceed in the following way:

- Observations are ranked in order according to the value of the first variable.
- The observations are then re-ranked according to the values of the second variable.
- The number of interchanges of the first variable is used to compute Kendall's τ_b.

Hoeffding's D Statistic

The formula for Hoeffding's D (1948) is

$$D = 30\left(\frac{(n-2)(n-3)D_1 + D_2 - 2(n-2)D_3}{n(n-1)(n-2)(n-3)(n-4)}\right)$$

where:

$$D_1 = \Sigma_i(Q_i-1)(Q_i-2)$$
$$D_2 = \Sigma_i(R_i-1)(R_i-2)(S_i-1)(S_i-2)$$
$$D_3 = \Sigma_i(R_i-2)(S_i-2)(Q_i-1)$$

Note the following:

- The R_i and S_i are ranks of the x and y values.
- The Q_i (sometimes called bivariate ranks) are one plus the number of points that have both x and y values less than the ith points.
- A point that is tied on its x value or y value, but not on both, contributes 1/2 to Q_i if the other value is less than the corresponding value for the ith point. A point tied on both x and y contributes 1/4 to Q_i.

When there are no ties among observations, the *D* statistic has values between –0.5 and 1, where 1 indicates complete dependence. If a weight variable is specified, it is ignored.

Inverse Correlation Matrix

The inverse correlation matrix provides useful multivariate information. The diagonal elements of the inverse correlation matrix, sometimes called the variance inflation factors (VIF), are a function of how closely the variable is a linear function of the other variables. Specifically, if the correlation matrix is denoted **R** and the inverse correlation matrix is denoted $\mathbf{R}^{-1}$, the diagonal element is denoted r_{ii} and is computed as follows:

$$r_{ii} = \mathrm{VIF}_i = \frac{1}{1 - \mathrm{R}_i^2}$$

where R_i^2 is the coefficient of variation from the model regressing the i^{th} explanatory variable on the other explanatory variables. Thus, a large r_{ii} indicates that the i^{th} variable is highly correlated with any number of the other variables.

Distance Measures

The Outlier Analysis plots show the specified distance measure for each point in the data table.

Mahalanobis Distance Measures

The Mahalanobis distance takes into account the correlation structure of the data and the individual scales. For each value, the Mahalanobis distance is denoted M_i and is computed as follows:

$$M_i = \sqrt{(\mathrm{Y}_i - \overline{\mathrm{Y}})'\mathrm{S}^{-1}(\mathrm{Y}_i - \overline{\mathrm{Y}})}$$

where:

Y_i is the data for the i^{th} row

$\overline{Y}$ is the row of means

S is the estimated covariance matrix for the data

The UCL reference line (Mason and Young 2002) drawn on the Mahalanobis Distances plot is computed as follows:

$$UCL_{Mahalanobis} = \sqrt{\frac{(n-1)^2}{n}\beta_{\left[1-\alpha;\frac{p}{2};\frac{n-p-1}{2}\right]}}$$

where:

n = number of observations

p = number of variables (columns)

$\beta_{\left[1-\alpha;\frac{p}{2};\frac{n-p-1}{2}\right]}$ = $(1-\alpha)^{th}$ quantile of a Beta $\left(\frac{p}{2}, \frac{n-p-1}{2}\right)$ distribution

If a variable is an exact linear combination of other variables, then the correlation matrix is singular and the row and the column for that variable are zeroed out. The generalized inverse that results is still valid for forming the distances.

Jackknife Distance Measures

The jackknife distance is calculated with estimates of the mean, standard deviation, and correlation matrix that do not include the observation itself. For each value, the jackknife distance is computed as follows:

$$J_i = \sqrt{\frac{(n-2)n^2}{(n-1)^3} \times \frac{M_i^2}{1-\frac{nM_i^2}{(n-1)^2}}}$$

where:

n = number of observations

p = number of variables (columns)

M_i = Mahalanobis distance for the i^{th} observation

The UCL reference line (Penny 1996) drawn on the Jackknife Distances plot is calculated as follows:

$$UCL_{Jackknife} = \sqrt{\frac{(n-2)n^2}{(n-1)^3} \times \frac{UCL_{Mahalanobis}^2}{1-\frac{n \bullet UCL_{Mahalanobis}^2}{(n-1)^2}}}$$

T^2 Distance Measures

The T^2 distance is the square of the Mahalanobis distance, so $T_i^2 = M_i^2$.

The UCL on the T^2 distance is:

$$UCL_{T^2} = \frac{(n-1)^2}{n}\beta_{\left[1-\alpha;\frac{p}{2};\frac{n-p-1}{2}\right]} = (UCL_{Mahalanobis})^2$$

where

n = number of observations

p = number of variables (columns)

$\beta_{\left[1-\alpha;\frac{p}{2};\frac{n-p-1}{2}\right]}$ = $(1-\alpha)^{th}$ quantile of a Beta $\left(\frac{p}{2}, \frac{n-p-1}{2}\right)$ distribution

Multivariate distances are useful for spotting outliers in many dimensions. However, if the variables are highly correlated in a multivariate sense, then a point can be seen as an outlier in multivariate space without looking unusual along any subset of dimensions. In other words, when the values are correlated, it is possible for a point to be unremarkable when seen along one or two axes but still be an outlier by violating the correlation.

Cronbach's α

Cronbach's α is defined as follows:

$$\alpha = \frac{kc}{v + (k-1)c}$$

where

k = the number of items in the scale

c = the average covariance between items

v = the average variance between items

If the items are standardized to have a constant variance, the formula becomes

$$\alpha = \frac{k(r)}{1 + (k-1)r}$$

where

r = the average correlation between items

The larger the overall α coefficient, the more confident you can feel that your items contribute to a reliable scale or test. The coefficient can approach 1.0 if you have many highly correlated items.

Chapter 4

Principal Components

Reduce the Dimensionality of Your Data

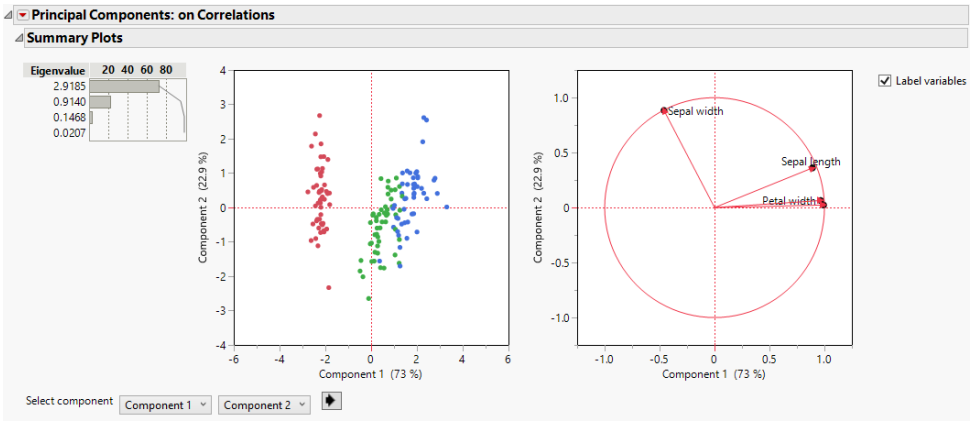
The purpose of principal component analysis is to derive a small number of independent linear combinations (*principal components*) of a set of measured variables that capture as much of the variability in the original variables as possible. Principal component analysis is a dimension-reduction technique, as well as an exploratory data analysis tool. Principal component analysis is also useful for constructing predictive models, as in *principal components analysis regression* (also known as PCA regression or PCR).

For data with a very large number of variables, the Principal Components platform provides an estimation method called the Wide method. The Wide method enables you to calculate principal components in short computing times. These principal components can then be used in PCA regression.

For data that contain mostly zeros, also called *sparse data*, the Principal Components platform provides the Sparse estimation method. Similar to the Wide method, the Sparse method calculates principal components in short computing times. Unlike the Wide method, the Sparse method calculates a fixed, user-defined number of principal components rather than the full set.

The Principal Components platform also supports factor analysis. JMP offers several types of orthogonal and oblique factor analysis-style rotations to help interpret the extracted components. For factor analysis, see "Factor Analysis".

Figure 4.1 Example of Principal Components



Contents

Overview of Principal Component Analysis Platform 57
Example of Principal Component Analysis 57
Launch the Principal Components Platform 59
Missing Data 61
Principal Components Report 62
Principal Components Report Options 63
Outlier Analysis 73
Statistical Details for the Principal Components Analysis Platform 75
Estimation Methods 75
DModX Calculation 76
Calculations for Outlier Analysis 77

Overview of Principal Component Analysis Platform

A principal component analysis models the variation in a set of variables in terms of a smaller number of independent linear combinations (*principal components*) of those variables.

If you want to see the arrangement of points across many correlated variables, you can use principal component analysis to show the most prominent directions of the high-dimensional data. Using principal component analysis reduces the dimensionality of a set of data. Principal components is a way to picture the structure of the data as completely as possible by using as few variables as possible.

For p variables, this is how the p principal components are formed:

- The first principal component is the linear combination of the standardized original variables that has the greatest possible variance.
- Each subsequent principal component is the linear combination of the variables that has the greatest possible variance and is uncorrelated with all previously defined components.

Each principal component is calculated by taking a linear combination of an eigenvector of the correlation matrix (or covariance matrix or sum of squares and cross products matrix) with the variables. The eigenvalues represent the variance of each component.

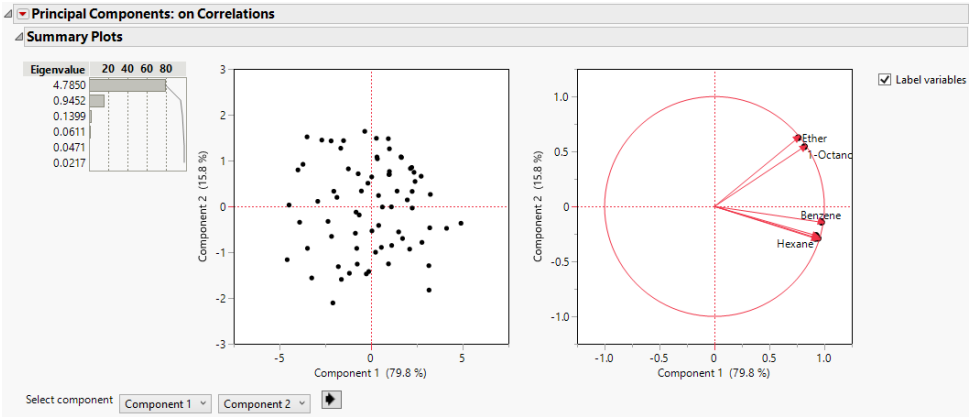
The Principal Components platform enables you to conduct your analysis on the correlation matrix, the covariance matrix, or the unscaled data. You can also conduct Factor Analysis within the Principal Components platform. See "Factor Analysis".

Example of Principal Component Analysis

To view an example Principal Component Analysis report for a data table for two factors:

1. Select **Help > Sample Data Library** and open Solubility.jmp.
2. Select **Analyze > Multivariate Methods > Principal Components**.
3. Select all of the continuous columns and click **Y, Columns**.
4. Keep the default Estimation Method and then click **OK**.

Figure 4.2 Principal Components on Correlations Report



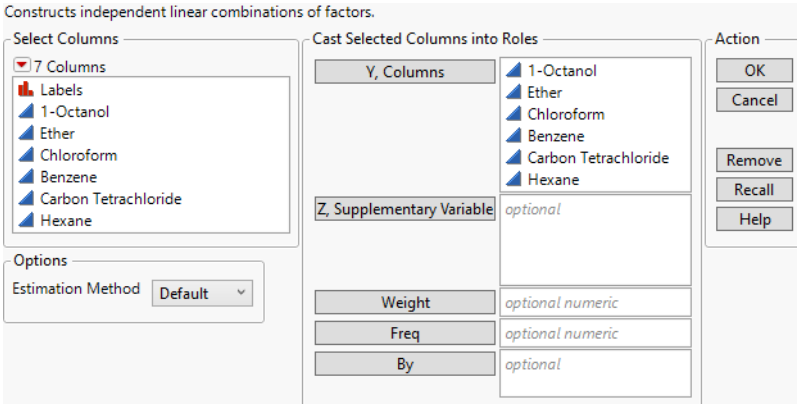
The report gives the eigenvalues and a bar chart of the percent of the variation accounted for by each principal component. In this example, the first principal component accounts for almost 80% of the variation in the data. Together, the first two principal components account for almost all of the variation in the data (95.5%). There is a Score Plot and a Loadings Plot as well. See "Principal Components Report".

Launch the Principal Components Platform

Launch the Principal Components platform by selecting **Analyze > Multivariate Methods > Principal Components**. Principal Component analysis is also available using the Multivariate and the Scatterplot 3D platforms.

The example described in "Example of Principal Component Analysis" uses all of the continuous variables from the Solubility.jmp sample data table.

Figure 4.3 Principal Components Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns The variables to analyze for components.

Z, Supplementary Variable The supplementary variables to be displayed. Supplementary variables are not included in the calculation of principal components and including them does not affect the results. Supplementary variables that are continuous can be projected on to the loading plot and used to enhance interpretation.

Weight Identifies one column whose numeric values assign a weight to each row in the analysis.

Note: The Weight role is ignored for the Wide and Sparse estimation methods.

Freq Identifies one column whose numeric values assign a frequency to each row in the analysis.

Note: The Freq role is ignored for the Wide and Sparse estimation methods.

By Creates a Principal Component report for each value specified by the By column so that you can perform separate analyses for each group.

Estimation Method Specifies the method for calculating the correlations. Several of these methods address the treatment of missing data.

Default The **Default** option uses either the Row-wise, Pairwise, or REML methods. A JMP Alert also recommends switching to the Wide method when appropriate.

- **Row-wise** estimation is used for data tables with no missing values.
- **Pairwise** estimation is used for data tables with missing values and either more than 10 columns, more than 5,000 rows, or more columns than rows.
- **REML** estimation is used otherwise.
- **Wide** estimation is recommended by a JMP Alert window for data tables with more than 500 columns and more columns than rows. This is because computation time can be considerable when you use the other methods with a large number of columns. Click **Wide** to switch to the Wide method or click **Continue** to use the method you originally selected.

REML Restricted maximum likelihood (REML) estimation uses all of the data, even if missing values are present. Due to a bias-correction factor, this method is slow if the dataset is large and there are many missing values. Therefore, REML is most useful for smaller datasets. If there are no missing cells in the data, then the REML and ML estimates are equivalent and equal to the sample covariance matrix. If there are missing cells, REML's variance and covariance estimates are less biased than the estimates from ML estimation. For statistical details, see "REML".

ML Maximum likelihood (ML) estimation uses all of the data, even if missing values are present. Because the estimates from ML are generated quickly, this method is most useful for large data tables with missing data.

Robust Robust estimation uses all of the data, even if missing values are present. This method down-weights extreme values and is therefore useful for data tables that might have outliers. For statistical details, see "Robust".

Row-wise Row-wise estimation calculates the Pearson correlation for each pair of columns. For statistical details, see "Pearson Product-Moment Correlation". Row-wise estimation does not use observations with missing values. This method is useful for excluding any observations that have missing data.

Pair-wise Pair-wise estimation uses all of the data, even if missing values are present. This estimation method calculates Pearson correlations for each pair of columns using all observations with nonmissing values for those two columns. For statistical details, see "Pearson Product-Moment Correlation". Pair-wise estimation is most useful when a data table has missing values and either more columns than rows, more than 10 columns, or more than 5,000 rows.

Wide Wide estimation does not use observations with missing values, so rows that contain missing cells are deleted before the method is applied. This estimation method uses an algorithm based on the full singular value decomposition. The algorithm avoids calculating the covariance matrix and is therefore computationally efficient. It is useful when you have a very large number of columns in your data. For statistical details, see "Wide".

JMP PRO **Sparse** Sparse estimation uses all of the data, even if missing values are present. This estimation method uses an algorithm based on the partial singular value decomposition, which computes only the first specified number of singular values and singular value vectors. The algorithm avoids calculating the covariance matrix, as well as unnecessary principal components and is therefore computationally efficient. It is useful when your data are sparse, meaning they contain many zeros, or when there are a large number of columns in the data. For statistical details, see "Sparse".

Note: If you select REML, ML, or Robust and your data table contains more columns than rows and has missing values, JMP switches the Estimation Method to Pairwise.

Number of Components (Available only when Sparse is specified as the Estimation Method.) Specifies the number of components to be estimated. Typically, the Number of Components is much smaller than the dimension of your data.

Missing Data

The different estimation methods are equipped to handle missing data in a variety of ways. You can also estimate missing values in the following ways:

- Use the Impute Missing Data option found under **Multivariate Methods > Multivariate**. See "Impute Missing Data".
- Use the Multivariate Normal Imputation or Multivariate SVD Imputation utilities found in **Analyze > Screening > Explore Missing Values**. See *Predictive and Specialized Modeling*.

Principal Components Report

If you selected any estimation method other than Wide or Sparse, the Principal Components: on Correlations report initially appears. (The title of this report changes if you select on Covariances or on Unscaled for the Principal Components option in the Principal Components red triangle menu.)

If you select the Wide method, the Wide Principal Components report appears. If you select the Sparse method, the Sparse Principal Components report appears.

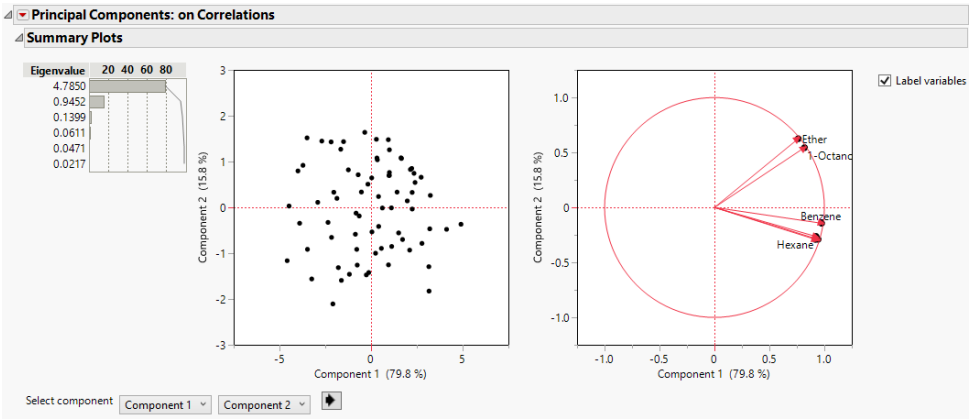
The initial Principal Components report is for an analysis on Correlations. It summarizes the variation of the specified *Y* variables with principal components (Figure 4.4). You can switch to an analysis based on the covariance matrix or unscaled data by selecting the Principal Components option from the red triangle menu.

Based on your selection, the principal components are derived from an eigenvalue decomposition of one of the following:

- the correlation matrix
- the covariance matrix
- the sum of squares and cross products matrix for the unscaled and uncentered data

The details in the report show how the principal components absorb the variation in the data. The principal component points are derived from the eigenvector linear combination of the variables.

Figure 4.4 Principal Components on Correlations Report



The report gives the eigenvalues and a bar chart of the percent of the variation accounted for by each principal component. There is a Score Plot and a Loadings Plot as well. The eigenvalues indicate the total number of components extracted based on the amount of variance contributed by each component.

The Score Plot graphs each component's calculated values in relation to the other, adjusting each value for the mean and standard deviation.

The Loadings Plot graphs the unrotated loading matrix between the variables and the components. The closer the value is to 1, the greater the effect of the component on the variable.

By default, the report shows the Score Plot and the Loadings Plot for the first two principal components. Use the list next to Select component to specify the principal components that are graphed on the Score Plot and the Loadings Plot.

Principal Components Report Options

The Principal Components red triangle menu contains the following options:

Note: Some of the options are not available for the Wide or Sparse estimation methods.

Principal Components (Not available for the Wide or Sparse estimation methods.) Enables you to create the principal components based on **Correlations**, **Covariances**, or **Unscaled**.

Correlations (Not available for the Wide or Sparse estimation methods.) The matrix of correlations between the variables.

Note: The values on the diagonals are 1.0.

Covariance Matrix (Not available for the Wide or Sparse estimation methods.) Shows or hides the covariances of the variables.

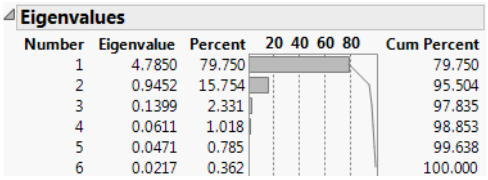
Eigenvalues Lists the eigenvalue that corresponds to each principal component in order from largest to smallest. The eigenvalues represent a partition of the total variation in the multivariate sample.

The scaling of the eigenvalues depends on which matrix you select for extraction of principal components:

- For the on Correlations option, the eigenvalues are scaled to sum to the number of variables.
- For the on Covariances options, the eigenvalues are not scaled.
- For the on Unscaled option, the eigenvalues are divided by the total number of observations.

If you select the **Bartlett Test** option from the red triangle menu, hypothesis tests (Figure 4.6) are given for each eigenvalue (Jackson 2003).

Figure 4.5 Eigenvalues



Eigenvalues

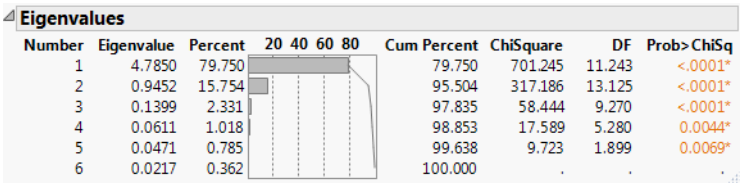
Number	Eigenvalue	Percent	20 40 60 80	Cum Percent
1	4.7850	79.750		79.750
2	0.9452	15.754		95.504
3	0.1399	2.331		97.835
4	0.0611	1.018		98.853
5	0.0471	0.785		99.638
6	0.0217	0.362		100.000

Eigenvectors Shows or hides a table of the eigenvectors for each of the principal components, in order, from left to right. Using these coefficients to form a linear combination of the original variables produces the principal component variables. Following the standard convention, eigenvectors have norm 1.

Note: The number of eigenvectors shown is equal to the rank of the correlation matrix, or, if the Sparse method is selected, the number of components specified on the launch window.

Bartlett Test (Not available for the Wide or Sparse estimation methods.) Shows or hides the results of the homogeneity test (appended to the Eigenvalues table). The test determines whether the eigenvalues have the same variance by calculating the Chi-square, degrees of freedom (DF), and the *p*-value (prob > ChiSq) for the test. See Bartlett (1937, 1954).

Figure 4.6 Bartlett Test



Eigenvalues

Number	Eigenvalue	Percent	20 40 60 80	Cum Percent	ChiSquare	DF	Prob>ChiSq
1	4.7850	79.750		79.750	701.245	11.243	<.0001*
2	0.9452	15.754		95.504	317.186	13.125	<.0001*
3	0.1399	2.331		97.835	58.444	9.270	<.0001*
4	0.0611	1.018		98.853	17.589	5.280	0.0044*
5	0.0471	0.785		99.638	9.723	1.899	0.0069*
6	0.0217	0.362		100.000	.	.	.

Loading Matrix Shows or hides a table of the loadings for each component. These values are graphed in the loading plot. The degree of transparency for the table values indicates the distance of the absolute loading value from zero. Absolute loading values that are closer to zero are more transparent than absolute loading values that are farther from zero.

If you specify a supplementary variable, an additional table of coordinates is shown for each supplementary continuous variable and each level of supplementary categorical variable. These values are graphed in the loading plot for continuous supplementary variables.

The scaling of the loadings and coordinates depends on which matrix you select for extraction of principal components:

– For the on Correlations option, the i^{th} column of loadings is the i^{th} eigenvector multiplied by the square root of the i^{th} eigenvalue. The i,j^{th} loading is the correlation between the i^{th} variable and the j^{th} principal component.

– For the on Covariances option, the j^{th} entry in the i^{th} column of loadings is the i^{th} eigenvector multiplied by the square root of the i^{th} eigenvalue and divided by the standard deviation of the j^{th} variable. The i,j^{th} loading is the correlation between the i^{th} variable and the j^{th} principal component.
– For the on Unscaled option, the j^{th} entry in the i^{th} column of loadings is the i^{th} eigenvector multiplied by the square root of the i^{th} eigenvalue and divided by the standard error of the j^{th} variable. The standard error of the j^{th} variable is the j^{th} diagonal entry of the sum of squares and cross products matrix divided by the number of rows ($\mathbf{X}'\mathbf{X}$/n).

Note: When you are analyzing the unscaled data, the i,j^{th} loading is *not* the correlation between the i^{th} variable and the j^{th} principal component.

Formatted Loading Matrix Shows or hides a table of the loadings for each component. The table is sorted in order of decreasing loadings on the first principal component. Therefore, the variables are listed in the order of decreasing loadings on the first component.

Figure 4.7 Formatted Loading Matrix

Formatted Loading Matrix

	Prin1	Prin2	Prin3	Prin4	Prin5	Prin6
Benzene	0.974761	-0.143460	-0.081914	-0.023369	-0.108186	-0.101488
Carbon Tetrachloride	0.942849	-0.289099	0.069134	-0.059654	-0.099757	0.095746
Hexane	0.923483	-0.263641	0.256572	0.026770	0.099672	-0.034523
Chloroform	0.917422	-0.290351	-0.242516	0.075629	0.093454	0.027695
1-Octanol	0.819019	0.544318	-0.041397	-0.162739	0.068711	0.002762
Ether	0.761978	0.625283	0.044774	0.155130	-0.045337	0.016883

Suppress Absolute Loading Value Less Than 0.3
Dim Text 0.4

Suppress Absolute Loading Value Less Than The value that determines which loadings are unavailable in the Formatted Loading Matrix report. You can use the text box or the slider to dim the loadings whose absolute values fall below the selected value.

Dim Text The transparency of the dimmed values in the Formatted Loading Matrix report. You can use the text box or the slider to set the degree of transparency for the dimmed loadings. The degree of transparency ranges from 0 to 1, where lower values are more transparent than higher values. For example, setting the transparency to 0 completely removes the unavailable loadings from the matrix, while the loadings are still available when you set the transparency to 1.

Squared Cosines of Variables Shows or hides a table that contains the squared cosines of variables. If you specify a supplementary variable, an additional table of squared cosines of supplementary variables is shown. The sum of the squared cosine values across principal components is equal to one for each variable. The squared cosines enable you to see how well the variables are represented by the principal components. You can also determine how many principal components are necessary to represent certain variables.

This option also shows a plot of the squared cosines for the first three principal components.

Note: If the Sparse estimation method is used and the number of components selected is less than three, only the specified number of components are displayed in the plot.

Partial Contribution of Variables Shows or hides a table that contains the partial contributions of variables. The partial contributions enable you to see the percentage that each variable contributes to each principal component. This option also shows a plot of the partial contributions for the first three principal components.

Note: If the Sparse estimation method is used and the number of components selected is less than three, only the specified number of components are displayed in the plot.

Summary Plots Shows or hides the summary information produced in the default report. This summary information includes a plot of the eigenvalues, a score plot, and a loading plot. By default, the report shows the score and loading plots for the first two principal components. There are options in the report to specify which principal components to plot. See "Principal Components Report".

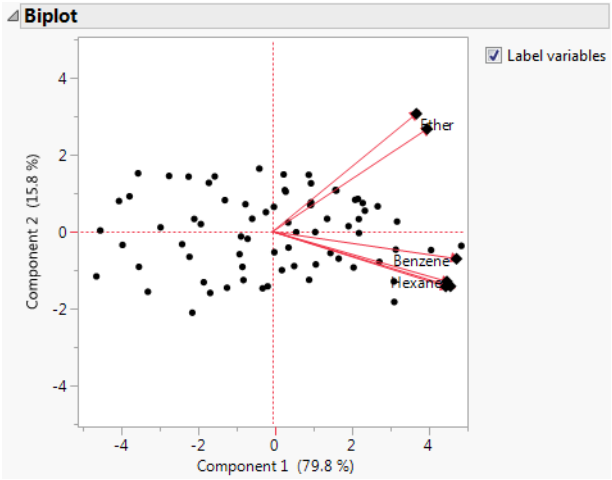
Note: If your data contain missing values, the imputed scores are plotted on the score plot.

Tip: Select the tips of arrows in the loading plot to select the corresponding columns in the data table. Press Ctrl and click an arrow tip to deselect the column.

Biplot Shows or hides a plot that overlays the score plot and the loading plot for the specified number of components.

Note: If your data contain missing values, the imputed scores are plotted on the Biplot.

Figure 4.8 Biplot

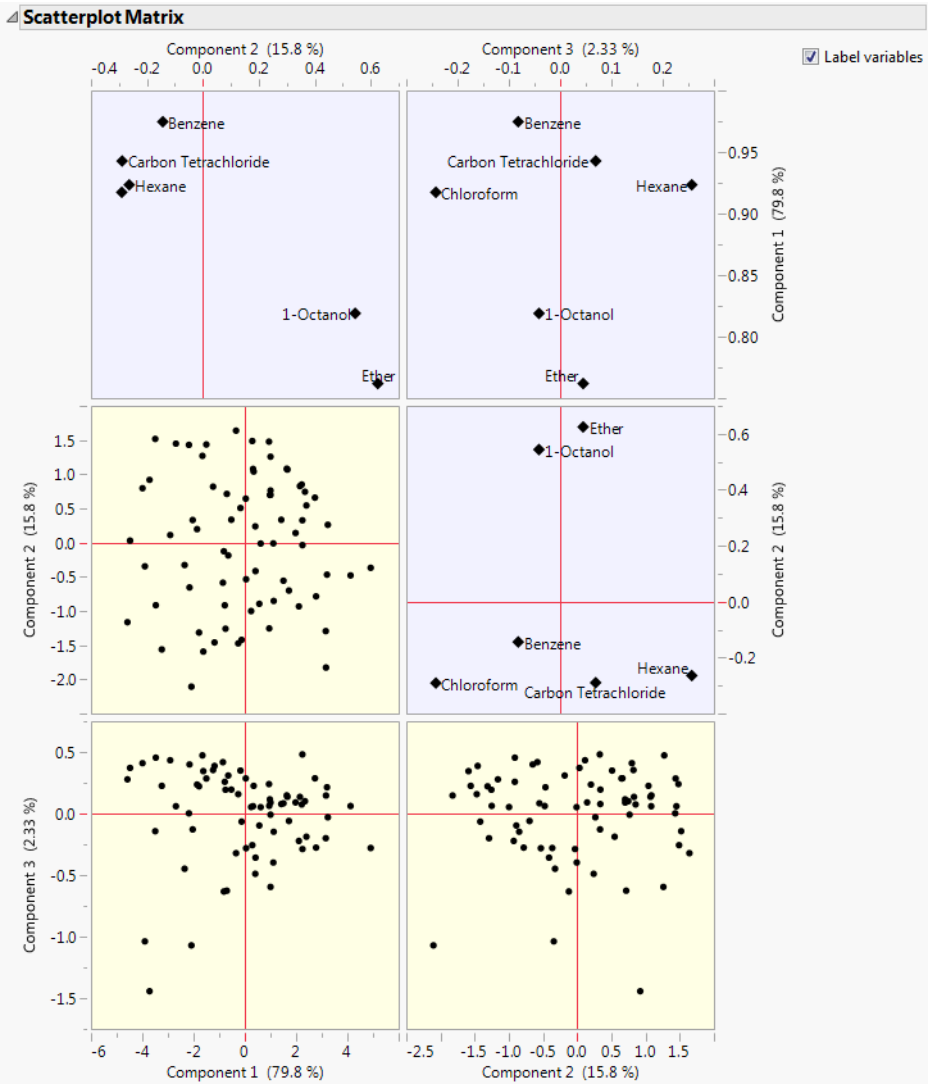


Note: The score plot markers are dots and the loading plot markers are diamonds.

Scatterplot Matrix Shows or hides a matrix of score and loading plots for a specified number of principal components. The scatterplot matrix arranges both the score plots and the loading plots in one space. The score plots have a yellow shaded background. The loading plots have a blue shaded background.

Note: If your data contain missing values, the imputed scores are plotted in the Scatterplot Matrix.

Figure 4.9 Scatterplot Matrix



Note: The loading plot matrix displayed in the Scatterplot Matrix is the transpose of the loading plot matrix that you obtain when you select the Loading Plot option.

Scree Plot Shows or hides a graph of the eigenvalue for each component. This scree plot helps in visualizing the dimensionality of the data space.

Score Plot Shows or hides a matrix of scatterplots of the scores for pairs of principal components for the specified number of components. This plot is shown in Figure 4.4 (left-most plot).

Loading Plot Shows or hides a matrix of two-dimensional representations of factor loadings for the specified number of components. The loading plot labels variables if the number of variables is 30 or fewer. If there are more than 30 variables, the labels are off by default. This information is shown in Figure 4.4 (right-most plot).

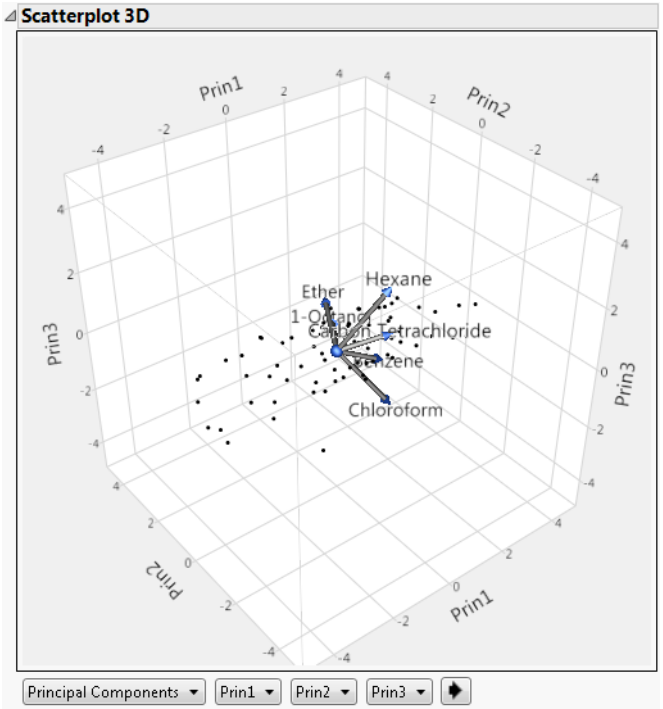
Tip: Select the tips of arrows in the loading plot to select the corresponding columns in the data table. Press Ctrl and click an arrow tip to deselect the column.

Score Plot with Imputation (Not available for the Wide or Sparse estimation methods.) Imputes any missing values and creates a score plot. This option is available only if there are missing values.

3D Score Plot (Not available for the Wide or Sparse estimation methods.) Shows or hides a 3D scatterplot of any three principal component scores. When you first invoke the command, the first three principal components are presented.

Note: If your data contain missing values, the imputed scores are plotted on the 3D Score Plot.

Figure 4.10 Scatterplot 3D Score Plot



Plot Source The source of the data points in the plot. The available options are Principal Components, Rotated Principal Components, and Data Columns.

Axis Controls The contents of each axis. If the Principal Components option or the Rotated Components option is selected, the options for the Axis Controls are principal components. If the Data Columns option is selected, the options are variables from the analysis.

Cycle Button Cycles through all axis content possibilities.

The variables show as rays in the plot. These rays, called *biplot rays*, approximate the variables as a function of the principal components on the axes. If there are only two or three variables, the rays represent the variables exactly. The ray corresponds to the principal component loadings.

Score Ellipses Shows or hides ellipses on the summary score plot for each pair of principal components. The ellipses are constructed as either confidence ellipses based on the alpha level or a control limit ellipses based on how far the observations are from the center. By default, the ellipses are 95% confidence ellipses.

Score Ellipse Coverage Displays a submenu that enables you to change how the score ellipses are constructed. Specify the score ellipses by confidence level or the distance from the center in terms of *k*-sigma. The relationship between the confidence level, *p*, and *k*-sigma is $p = 1 - \exp(-k^2/2)$.

Display Options

Arrow Lines Enables you to show or hide arrows on all plots that can display arrows. Arrows are shown if the number of variables is 1000 or fewer. If there are more than 1000 variables, the arrows are off by default.

Show Supplementary Variable (Available only if you specify a supplementary variable.) Shows or hides the arrow lines for continuous supplementary variables or label markers for categorical supplementary variables in the biplot, score plot, and loading plots.

Outlier Analysis Shows or hides the Outlier Analysis report, which enables you to detect outliers in the data through T^2 and contribution statistics. See "Outlier Analysis".

Factor Analysis (Not available for the Wide or Sparse estimation methods.) Performs factor analysis-style rotations of the principal components, or factor analysis. See "Factor Analysis".

JMP PRO **Cluster Variables** (Not available for the Wide or Sparse estimation methods.) Performs a cluster analysis on the variables by dividing the variables into non-overlapping clusters. Variable clustering provides a method for grouping similar variables into representative groups. Each cluster can then be represented by a single component or variable. The

component is a linear combination of all variables in the cluster. Alternatively, the cluster can be represented by the variable identified to be the most representative member in the cluster. See "Cluster Variables".

Note: Cluster Variables uses correlation matrices for all calculations, even when you select the on Covariance or on Unscaled options.

Model Driven Multivariate Control Chart Saves the formulas for the specified number of principal components and launches the Model Driven Multivariate Control Chart (MDMCC) launch window. In the MDMCC launch window, the principal components formulas are assigned as the process columns. You can add or remove processes, add a time ID, or set where historical data ends before clicking OK. See *Quality and Process Methods*.

Save Principal Components Saves the number of principal components that you specify to the data table with a formula for computing each component. The formula cannot evaluate rows with missing values.

The calculation for the principal components depends on which matrix you select for extraction of principal components:

- For the on Correlations option, the i^{th} principal component is a linear combination of the centered and scaled observations using the entries of the i^{th} eigenvector as coefficients.
- For the on Covariances options, the i^{th} principal component is a linear combination of the centered observations using the entries of the i^{th} eigenvector as coefficients.
- For the on Unscaled option, the i^{th} principal component is a linear combination of the raw observations using the entries of the i^{th} eigenvector as coefficients.

Note: If the specified number of components exceeds the rank of the correlation matrix, then the number of components saved is set to the rank of the correlation matrix.

Save Predicteds Saves the predicted variables, given the specified number of principal components, to new columns in the data table.

Save Predicteds As Component Formulas Saves the predicted variables as formulas that are linear combinations of the specified number of components. The principal components used in the formulas are also saved to the data table.

Save DModX Saves the observation distance to the principal components model (DModX), given the specified number of principal components, to a new column in the data table. Larger DModX values indicate mild to moderate outliers in the data. See "DModX Calculation".

Save Individual Squared Cosines Saves the individual squared cosines, given the specified number of principal components, to new columns in the data table.

Save Individual Partial Contributions Saves the individual partial contributions, given the specified number of principal components, to new columns in the data table.

Save Rotated Components (Not available for the Wide or Sparse estimation methods.) Saves the rotated components to the data table, with a formula for computing the components. This option is available only after the Factor Analysis option is used. The formula cannot evaluate rows with missing values.

Save Principal Components with Imputation (Not available for the Wide or Sparse estimation methods.) Imputes missing values, and saves the principal components to the data table. The column contains a formula for doing the imputation and computing the principal components. This option is available only if there are missing values.

Save Rotated Components with Imputation (Not available for the Wide or Sparse estimation methods.) Imputes missing values and saves the rotated components to the data table. The column contains a formula for doing the imputation and computing the rotated components. This option is available only after the Factor Analysis option is used and if there are missing values.

Save Imputed Formulas (Available only if the data table contains missing values.) For columns that contain missing values, saves new columns to the data table that contain the formulas used to estimate the missing values. The new columns are called Imputed_<Column Name>.

JMP PRO **Publish Components Formulas** Creates a specified number of principal component formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

JMP PRO **Publish DModX Formula** Saves the DModX formula based on a specified number of principal components as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

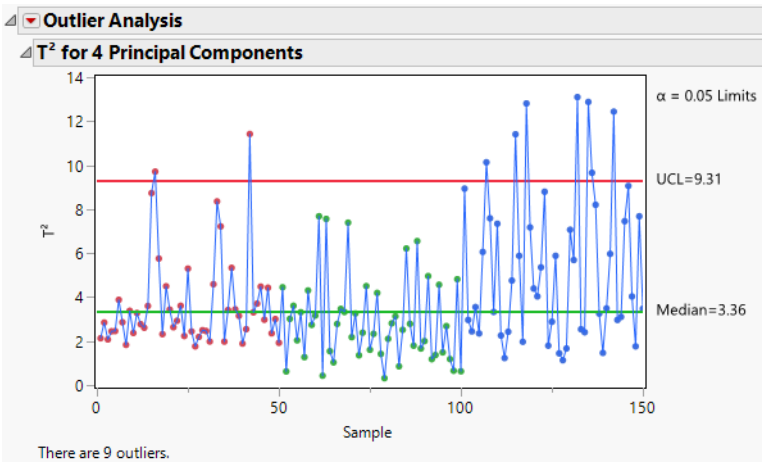
Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Outlier Analysis

Figure 4.11 Outlier Analysis Report



By default, the Outlier Analysis report displays the T^2 for <*A*> Principal Components plot. The plot shows the T^2 value for each observation and horizontal lines at the Median and Upper Control Limit (UCL). For more information about how the T^2 values, median, and UCL are calculated, see "Calculations for Outlier Analysis".

The α level used to calculate the UCL is displayed next to the plot. The number of outliers detected is shown below the plot. This number is the number of observations with T^2 values that are greater than the UCL.

Tip: Hover over a point in the T^2 Plot to view the T^2 Contribution Proportion Plot for that observation. Click the T^2 Contribution Proportion Plot to add it to the report window.

Outlier Analysis Report Options

T² Plot Shows or hides the T^2 plot. On by default.

Contribution Heat Map Shows or hides a heat map of the T^2 contribution values for all observations.

Contribution Proportions Heat Map Shows or hides a heat map of the T^2 contribution values for all observations expressed as a proportion of the individual row's T^2. The proportions for an individual row are obtained by computing the square of the contribution and dividing by the sum of the squares of all contributions for that individual row.

Contribution Plots for Selected Samples (Available only if one or more points is selected in the T^2 plot.) Shows a report with a T^2 contribution plot for each selected sample. A T^2 contribution plot shows the contribution of each variable to the sample's T^2 statistic. For more information about how the contributions are calculated, see "Calculations for Outlier Analysis". Use contribution plots to investigate outliers. The variables with the largest positive or negative contributions are those that contribute most to a sample having a large T^2 value. See "Contribution Plots Report Options" for information on red triangle menu options.

Contribution Proportion Plots for Selected Samples (Available only if one or more points is selected in the T^2 plot.) Shows a report with a T^2 contribution proportion plot for each selected sample. A T^2 contribution proportion plot shows the contribution values for the selected observations expressed as a proportion of the individual row's T^2. This is a different presentation of the information found in the Contribution Proportions Heat Map. See "Contribution Plots Report Options" for information on red triangle menu options.

Normalized DModX Plot (Available only when the number of components is less than the number of variables.) Shows or hides a plot of the Normalized DModX values. DModX values are useful for detecting moderate outliers in the data.

Number of Components Enables you to specify the number of principal components used in the T^2 and T^2 contribution statistics. When you change the number of components, the T^2 plot, heap map, and normalized DModX plot automatically update.

Set α level Enables you to specify the α level.

Save T² Saves the T^2 values to a new column in the data table.

Save Contributions Saves the T^2 Contributions to new columns in the data table. There is one column for each Y variable.

Save Normalized DModX (Available only when the number of components is less than the number of variables.) Saves the normalized DModX values to a new column in the data table.

Contribution Plots Report Options

The T^2 Contribution Plots for Selected Samples, T^2 Contribution Proportion Plots for Selected Samples, and T^2 Mean Contribution Proportion Plots for Selected Samples contain the following red triangle menu options.

Bar Label Displays a submenu of options to label the bars in the contribution plot. The label options are No Label, Label by Value, and Label by Column.

Remove Plot Removes the contribution plot from the report.

Control Charts for Selected Items (Available only if one or more bar segments are selected in the contribution plot.) Opens a Control Chart Builder window, with control chart results for each of the selected processes and groups.

Tip: You can also view the control chart by hovering over a bar segment in a contribution plot. Click the control chart to open the Control Chart Builder window.

Statistical Details for the Principal Components Analysis Platform

- "Estimation Methods"
- "DModX Calculation"
- "Calculations for Outlier Analysis"

Estimation Methods

REML

REML (restricted maximum likelihood) estimates are less biased than the ML (maximum likelihood) estimation method when the data contains missing values. The REML method maximizes marginal likelihoods based on error contrasts. The REML method is often used for estimating variances and covariances. The REML method in the Principal Components platform is the same as the REML estimation of mixed models for repeated measures data with an unstructured covariance matrix. See the documentation for SAS PROC MIXED about REML estimation of mixed models.

Wide

The Wide method uses a computationally efficient algorithm that avoids calculating the covariance matrix. The algorithm is based on the singular value decomposition. Consider the following notation:

- n = number of rows
- p = number of variables
- $\mathbf{X}$ = n by p matrix of data values

The number of nonzero eigenvalues, and consequently the number of principal components, equals the rank of the correlation matrix of **X**. The number of nonzero eigenvalues cannot exceed the smaller of n and p.

When you select the Wide method, the data are standardized. To standardize a value, subtract its mean and divide by its standard deviation. Denote the n by p matrix of standardized data values by $\mathbf{X}_s$. Then the covariance matrix of the standardized data is the correlation matrix of **X** and it is defined as follows:

$$Cov = \mathbf{X}_s'\mathbf{X}_s/(n-1)$$

Using the singular value decomposition, $\mathbf{X}_s$ is written as $\mathbf{U}Diag(\Lambda)\mathbf{V}'$. This representation is used to obtain the eigenvectors and eigenvalues of $\mathbf{X}_s'\mathbf{X}_s$. The principal components, or scores, are given by $\mathbf{X}_s\mathbf{V}$. For additional background information, see "Wide Linear Methods and the Singular Value Decomposition".

JMP PRO Sparse

Similar to the Wide method, the Sparse method is based on singular value decomposition. Therefore, the algorithm for the Sparse method avoids computing the covariance matrix and is computationally efficient.

Consider the same notation and standardization of **X** that is described in "Wide". The correlation matrix of **X** is represented by the covariance matrix of $\mathbf{X}_s$:

$$Cov(\mathbf{X}_s) = \mathbf{X}_s'\mathbf{X}_s/(n-1)$$

The Sparse method differs from the Wide method in the calculation of the singular value decomposition. The Wide method performs a full singular value decomposition. However, the Sparse method uses an algorithm that computes only the first specified number of singular values and singular vectors in the singular value decomposition. Therefore, only the first specified number of eigenvalues and principal components are returned. For more information about the algorithm, see Baglama and Reichel (2005).

DModX Calculation

DModX is the observed distance to the principal components model and is defined as follows:

$$\text{DModX} = \sqrt{\frac{\sum_k e_{ik}^2}{K-A}}$$

where

e_{ik} = the residuals from the model

K = the number of variables

A = the number of principal components

Larger values of DModX indicate mild to moderate outliers in the data.

Calculations for Outlier Analysis

The calculations in the Outlier Analysis report use the following notation:

n = number of observations

A = number of principal components

X_{ci} = the standardized data for the i^{th} observation

T^2 Statistic

The T^2 statistic for the i^{th} observation is calculated as follows:

$$T_i^2 = X_{ci} P_A L^{-1} P_A^{\mathrm{T}} X_{ci}^{\mathrm{T}}$$

where P_A is a matrix containing the first A eigenvectors and L is a diagonal matrix containing the first A eigenvalues.

The median and UCL for the T^2 plot are calculated as follows:

$$CL_{T^2, q} = \frac{(n-1)^2}{n} \beta\left[q, \frac{A}{2}, \frac{n-A-1}{2}\right]$$

where

$$\beta\left[q, \frac{A}{2}, \frac{n-A-1}{2}\right] = \text{the } q^{\text{th}} \text{ quantile of the Beta } \left(\frac{A}{2}, \frac{n-A-1}{2}\right) \text{ distribution.}$$

To calculate the median, use $q = 0.5$. To calculate the UCL, use $q = (1 - \alpha)$.

Contribution Statistic

The vector of T^2 contribution statistics for the ith observation is calculated as follows:

$$con_i = X_{ci} P_A L^{-1/2} P_A^{\mathrm{T}}$$

Note: The sum of the squared contributions for an individual is equal to T_i^2 for that individual.

Chapter 5

Discriminant Analysis

Predict Classifications Based on Continuous Variables

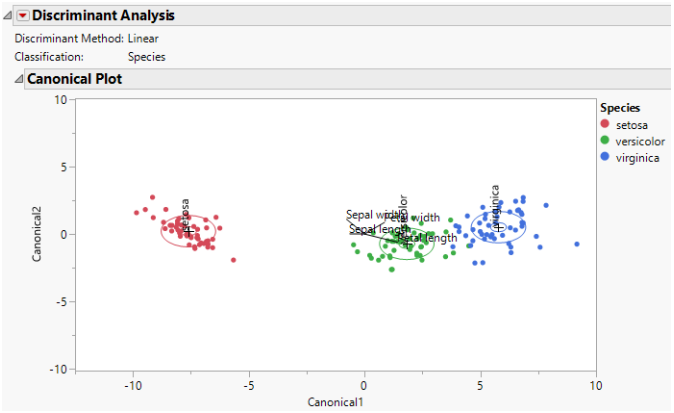
Discriminant analysis predicts membership in a group or category based on observed values of several continuous variables. Specifically, discriminant analysis predicts a classification (X) variable (categorical) based on known continuous responses (Y). The data for a discriminant analysis consist of a sample of observations with known group membership together with their values on the continuous variables.

For example, you might attempt to classify loan applicants into three loan categories (X) based on expected profitability: low interest rate loan, long term loan, or no loan. You might use continuous variables such as current salary, years in current job, age, and debt burden, (Ys) to predict an individual's most profitable loan category. You could build a predictive model to classify an individual into a loan category using discriminant analysis.

Features of the Discriminant platform include the following:

- A stepwise selection option to help choose variables that discriminate well.
- A choice of fitting methods: Linear, Quadratic, Regularized, and Wide Linear.
- A canonical plot and a misclassification summary.
- Discriminant scores and squared distances to each group.
- Options to save prediction distances and probabilities to the data table.

Figure 5.1 Canonical Plot



Contents

Overview of the Discriminant Analysis Platform 81
Example of Discriminant Analysis 81
Launch the Discriminant Analysis Platform 83
Stepwise Variable Selection 85
Discriminant Methods 88
Shrink Covariances 91
Discriminant Analysis Report 91
Principal Components 92
Canonical Plot and Canonical Structure 93
Discriminant Scores 97
Score Summaries 98
Discriminant Analysis Options 100
Score Options 102
Canonical Options 104
Example of a Canonical 3D Plot 107
Specify Priors 109
Consider New Levels 109
Save Discrim Matrices 109
Scatterplot Matrix 110
Validation in JMP and JMP Pro 111
Statistical Details for the Discriminant Analysis Platform 112
Description of the Wide Linear Algorithm 112
Saved Formulas 112
Multivariate Tests 119
Approximate F-Tests 120
Between Groups Covariance Matrix 121

Overview of the Discriminant Analysis Platform

Discriminant analysis attempts to classify observations described by values on continuous variables into groups. Group membership, defined by a categorical variable *X*, is predicted by the continuous variables. These variables are called *covariates* and are denoted by *Y*.

Discriminant analysis differs from logistic regression. In logistic regression, the classification variable is random and predicted by the continuous variables. In discriminant analysis, the classifications are fixed, and the covariates (*Y*) are realizations of random variables. However, in both techniques, the categorical value is predicted by the continuous variables.

The Discriminant platform provides four methods for fitting models. All methods estimate the distance from each observation to each group's multivariate mean (*centroid*) using Mahalanobis distance. You can specify prior probabilities of group membership and these are accounted for in the distance calculation. Observations are classified into the closest group.

Fitting methods include the following:

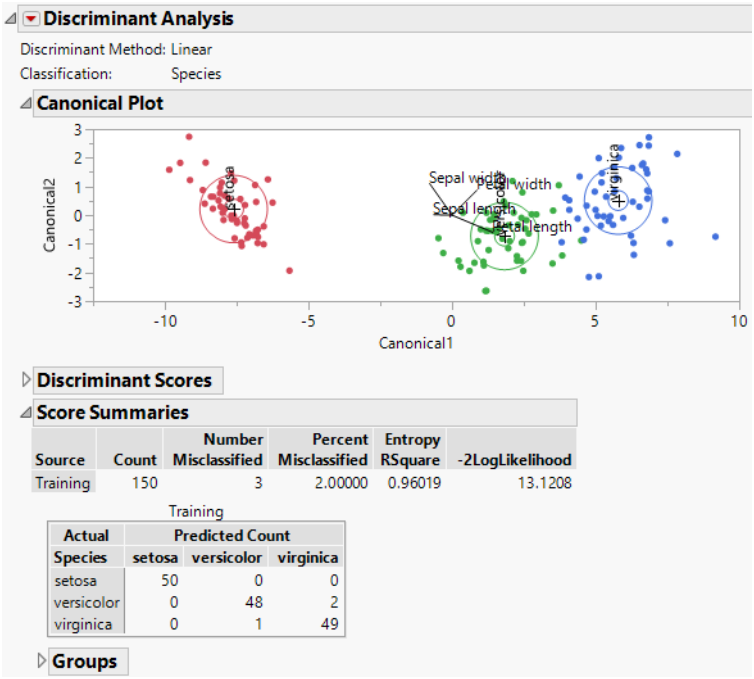
- Linear—Assumes that the within-group covariance matrices are equal. The covariate means for the groups defined by *X* are assumed to differ.
- Quadratic—Assumes that the within-group covariance matrices differ. This requires estimating more parameters than does the Linear method. If group sample sizes are small, you risk obtaining unstable estimates.
- Regularized—Provides two ways to impose stability on estimates when the within-group covariance matrices differ. This is a useful option if group sample sizes are small.
- Wide Linear—Useful in fitting models based on a large number of covariates, where other methods can have computational difficulties. It assumes that all covariance matrices are equal.

Example of Discriminant Analysis

In Fisher's Iris data set, four measurements are taken from a sample of Iris flowers consisting of three different species. The goal is to identify the species accurately using the values of the four measurements.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Analyze > Multivariate Methods > Discriminant**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Covariates**.
4. Select Species and click **X, Categories**.
5. Click **OK**.

Figure 5.2 Discriminant Analysis Report Window

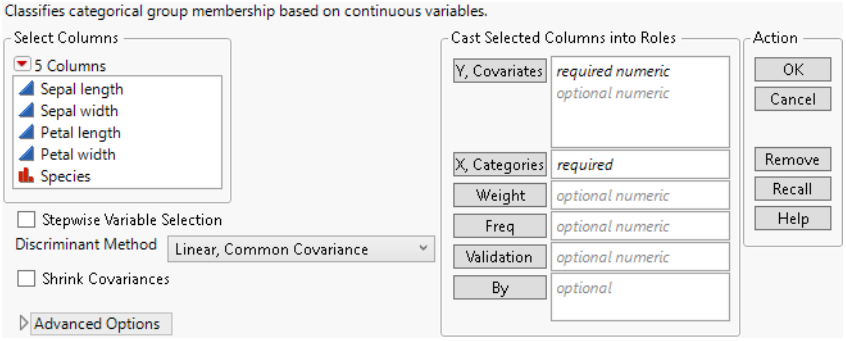


Because there are three classes for Species, there are two canonical variables. In the Canonical Plot, each observation is plotted against the two canonical coordinates. The plot shows that these two coordinates separate the three species. Since there was no validation set, the Score Summaries report shows a panel for the Training set only. When there is no validation set, the entire data set is considered the Training set. Of the 150 observations, only three are misclassified.

Launch the Discriminant Analysis Platform

Launch the Discriminant platform by selecting **Analyze > Multivariate Methods > Discriminant**.

Figure 5.3 Discriminant Launch Window for Iris.jmp



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Note: The Validation button appears in JMP Pro only. In JMP, you can define a validation set using excluded rows. See "Validation in JMP and JMP Pro".

Y, Covariates Columns that contain the continuous variables used to classify observations into categories.

X, Categories A column that contains the categories or groups into which observations are to be classified.

Weight A column whose values assign a weight to each row for the analysis.

Freq A column whose values assign a frequency to each row for the analysis. In general terms, the effect of a frequency column is to expand the data table, so that any row with integer frequency k is expanded to k rows. You can specify fractional frequencies.

JMP PRO **Validation** A numeric column that defines the validation sets. This column should contain at most three distinct values:

- If there are two values, the smaller value defines the training set and the larger value defines the validation set.
- If there are three values, these values define the training, validation, and test sets in order of increasing size.

– If the validation column has more than three levels, the rows that contain the smallest three values define the validation sets. All other rows are excluded from the analysis.

The Discriminant platform uses the validation column to train and evaluation the model, unless Stepwise Variable Selection is used. If the Stepwise Variable Selection option is selected in the launch, the Discriminant platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see *Predictive and Specialized Modeling*.

Tip: If Stepwise Variable Selection is not used, the validation column should contain only two distinct values.

If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, see *Predictive and Specialized Modeling*.

By Performs a separate analysis for each level of the specified column.

Stepwise Variable Selection Performs stepwise variable selection using covariance analysis and *p*-values. See "Stepwise Variable Selection".

If you have specified a validation set, statistics for the validation set also appear. The validation set statistics are used to determine how many steps to take if you use the Go button.

Note: This option is not provided for the Wide Linear discriminant method.

Discriminant Method Provides four methods for conducting discriminant analysis. See "Discriminant Methods".

Shrink Covariances Shrinks the off-diagonal elements of the pooled within-group covariance matrix and the within-group covariance matrices. This can improve stability and reduce the variance of prediction. See "Shrink Covariances".

Advanced Options Contains the following options:

Uncentered Canonical Suppresses centering of canonical scores for compatibility with older versions of JMP.

Use Pseudoinverses Uses Moore-Penrose pseudoinverses in the analysis when the covariance matrix is singular. The resulting scores involve all covariates. If left unchecked, the analysis drops covariates that are linear combinations of covariates that precede them in the list of **Y, Covariates**.

Cross Validate by Excluded Rows Specifies that the excluded rows form a validation set for which statistics of fit are calculated.

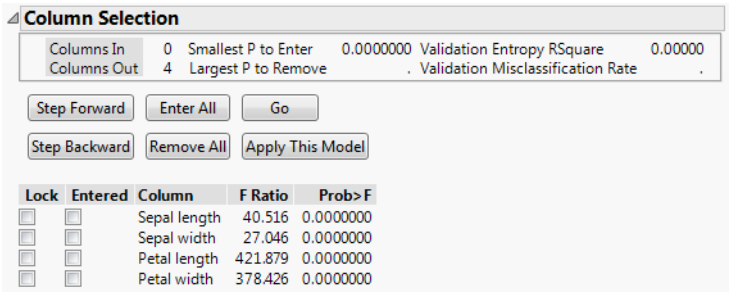
Stepwise Variable Selection

Note: Stepwise Variable Selection is not available for the Wide Linear method.

If you select the Stepwise Variable Selection option in the launch window, the Discriminant Analysis report opens, showing the Column Selection panel. You can perform stepwise analysis using the buttons to select variables or selecting them manually with the Lock and Entered check boxes. Based on your selection *F* ratios and *p*-values are updated. For more information about how these are updated, see "Updating the F Ratio and Prob>F".

If you specify any type of validation set, a Go button appears. When you click Go, JMP uses the validation set statistics to determine how many steps to take.

Figure 5.4 Column Selection Panel for Iris.jmp with a Validation Set



Updating the F Ratio and Prob>F

When you enter or remove variables from the model, the F Ratio and Prob>F values are updated based on an analysis of covariance model with the following structure:

- The covariate under consideration is the response.
- The covariates already entered into the model are predictors.
- The group variable is a predictor.

The values for F Ratio and Prob>F given in the Stepwise report are the *F* ratio and *p*-value for the analysis of covariance test for the group variable. The analysis of covariance test for the group variable is an indicator of its discriminatory power relative to the covariate under consideration.

Statistics

Columns In The number of columns currently selected for entry into the discriminant model.

Columns Out The number of columns currently available for entry into the discriminant model.

Smallest P to Enter The smallest *p*-value among the *p*-values for all covariates available to enter the model.

Largest P to Remove The largest *p*-value among the *p*-values for all covariates currently selected for entry into the model.

Validation Entropy RSquare Entropy RSquare for the validation set. Larger values indicate better fit. An Entropy RSquare value of 1 indicates that the classifications are perfectly predicted. Because uncertainty in the predicted probabilities is typical for discriminant models, Entropy RSquare values tend to be small.

See "Entropy RSquare". Available only if a validation set is used.

Note: It is possible for the Validation Entropy RSquare to be negative.

Validation Misclassification Rate Misclassification rate for the validation set. Smaller values indicate better classification. Available only if a validation set is used.

Buttons

Step Forward Enters the most significant covariate from the covariates not yet entered. If a validation set is used, the Prob>F values are based on the training set.

Step Backward Removes the least significant covariate from the covariates entered but not locked. If a validation set is used, Prob>F values are based on the training set.

Enter All Enters all covariates by checking all covariates that are not locked in the Entered column.

Remove All Removes all covariates that are not locked by deselecting them in the Entered column.

Apply this Model Produces a discriminant analysis report based on the covariates that are checked in the Entered columns. The Select Columns outline is closed and the Discriminant Analysis window is updated to show analysis results based on your selected Discriminant Method.

Tip: After you click **Apply this Model**, the columns that you select appear at the top of the Score Summaries report.

Go Enters covariates in forward steps until the Validation Entropy RSquare begins to decrease. Entry terminates when two forward steps are taken without improving the Validation Entropy RSquare. Available only with excluded rows in JMP or a validation column in JMP Pro.

Columns

Lock Forces a covariate to stay in its current state regardless of any stepping using the buttons.

Note the following:

- If you enter a covariate and then select **Lock** for that covariate, it remains in the model regardless of selections made using the control buttons. The **Entered** box for the locked covariate shows a dimmed check mark to indicate that it is in the model.
- If you select **Lock** for a covariate that is not Entered, it is not entered into the model regardless of selections made using the control buttons.

Entered Indicates which columns are currently in the model. You can manually select columns in or out of the model. A dimmed check mark indicates a locked covariate that has been entered into the model.

Column The covariate of interest.

F Ratio The *F* ratio for a test for the group variable obtained using an analysis of covariance model. See "Updating the F Ratio and Prob>F".

Prob > F The *p*-value for a test for the group variable obtained using an analysis of covariance model. See "Updating the F Ratio and Prob>F".

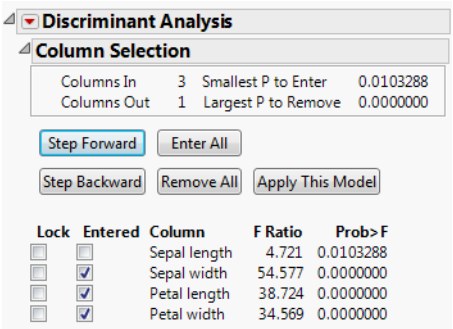
Stepwise Example

For an illustration of how to use Stepwise, consider the Iris.jmp sample data table.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Analyze > Multivariate Methods > Discriminant**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Covariates**.
4. Select Species and click **X, Categories**.
5. Select **Stepwise Variable Selection**.
6. Click **OK**.
7. Click **Step Forward** three times.

   Three covariates are entered into the model. The Smallest P to Enter appears in the top panel. It is 0.0103288, indicating that the remaining covariate, Sepal length, might also be valuable in a discriminant analysis model for Species.

Figure 5.5 Stepped Model for Iris.jmp

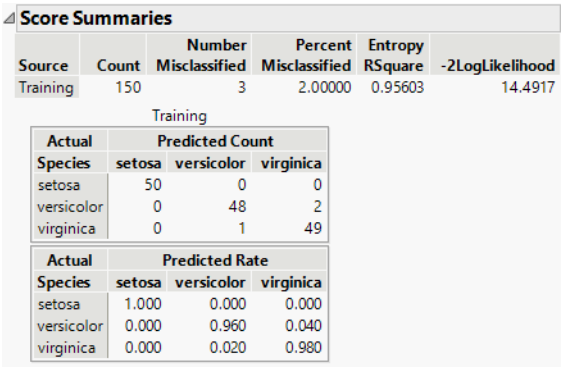


8. Click **Apply This Model**.

   The Column Selection outline is closed. The window is updated to show reports for a fit based on the entered covariates and your selected discriminant method.

   Note that the covariates that you selected for your model are listed at the top of the Score Summaries report.

Figure 5.6 Score Summaries Report Showing Selected Covariates



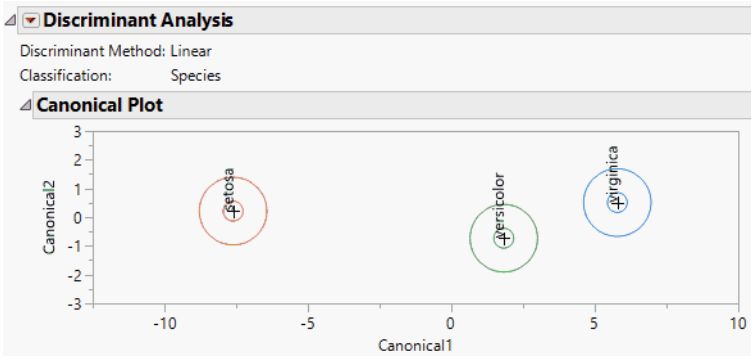
Discriminant Methods

JMP offers these methods for conducting Discriminant Analysis: Linear, Quadratic, Regularized, and Wide Linear. The first three methods differ in terms of the underlying model. The Wide Linear method is an efficient way to fit a Linear model when the number of covariates is large.

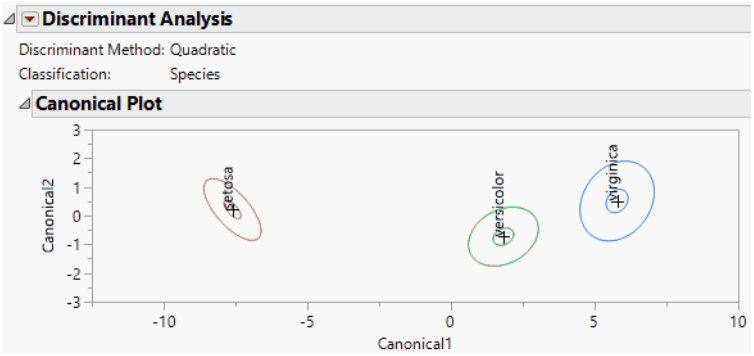
Note: When you enter more than 500 covariates, a JMP Alert recommends that you switch to the Wide Linear method. This is because computation time can be considerable when you use the other methods with a large number of columns. Click **Wide Linear, Many Columns** to switch to the Wide Linear method. Click **Continue** to use the method you originally selected.

Figure 5.7 Linear, Quadratic, and Regularized Discriminant Analysis

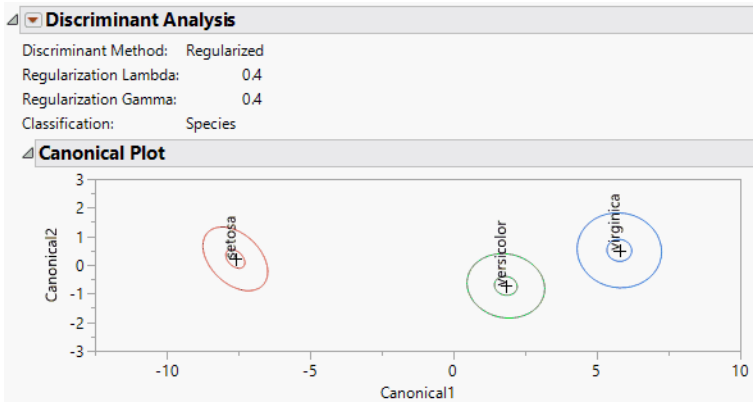
Linear



Quadratic



Regularized
(λ=0.4, γ=0.4)



The Linear, Quadratic, and Regularized methods are illustrated in Figure 5.7. The methods are described here briefly. See "Saved Formulas".

Linear, Common Covariance Performs linear discriminant analysis. This method assumes that the within-group covariance matrices are equal. See "Linear Discriminant Method".

Quadratic, Different Covariances Performs quadratic discriminant analysis. This method assumes that the within-group covariance matrices differ. This method requires estimating

more parameters than the Linear method requires. If group sample sizes are small, you risk obtaining unstable estimates. See "Quadratic Discriminant Method".

If a covariate is constant across a level of the X variable, then its related entries in the within-group covariance matrix have zero covariances. To enable matrix inversion, the zero covariances are replaced with the corresponding pooled within covariances. When this is done, a note appears in the report window identifying the problematic covariate and level of X.

Tip: A shortcoming of the quadratic method surfaces in small data sets. It can be difficult to construct invertible and stable covariance matrices. The Regularized method ameliorates these problems, still allowing for differences among groups.

Regularized, Compromise Method Provides two ways to impose stability on estimates when the within-group covariance matrices differ. This is a useful option when group sample sizes are small. See "Regularized, Compromise Method" and "Regularized Discriminant Method".

Wide Linear, Many Columns Useful in fitting models based on a large number of covariates, where other methods can have computational difficulties. This method assumes that all within-group covariance matrices are equal. This method uses a singular value decomposition approach to compute the inverse of the pooled within-group covariance matrix. See "Description of the Wide Linear Algorithm".

Note: When you use the Wide Linear option, a few of the features that normally appear for other discriminant methods are not available. This is because the algorithm does not explicitly calculate the very large pooled within-group covariance matrix.

Regularized, Compromise Method

Regularized discriminant analysis is governed by two nonnegative parameters.

- The first parameter (**Lambda, Shrinkage to Common Covariance**) specifies how to mix the individual and group covariance matrices. For this parameter, 1 corresponds to Linear Discriminant Analysis and 0 corresponds to Quadratic Discriminant Analysis.
- The second parameter (**Gamma, Shrinkage to Diagonal**) is a multiplier that specifies how much deflation to apply to the non-diagonal elements (the covariances across variables). If you choose 1, then the covariance matrix is forced to be diagonal.

Assigning 0 to each of these two parameters is identical to requesting quadratic discriminant analysis. Similarly, assigning 1 to Lambda and 0 to Gamma requests linear discriminant analysis. Use Table 5.1 to help you decide on the regularization. See Figure 5.7 for examples of linear, quadratic, and regularized discriminant analysis.

Table 5.1 Regularized Discriminant Analysis

Use Smaller Lambda	Use Larger Lambda	Use Smaller Gamma	Use Larger Gamma
Covariance matrices differ	Covariance matrices are identical	Variables are correlated	Variables are uncorrelated
Many rows	Few rows		
Few variables	Many variables		

Shrink Covariances

In the Discriminant launch window, you can select the option to Shrink Covariances. This option is recommended when some groups have a small number of observations. Discriminant analysis requires inversion of the covariance matrices. Shrinking off-diagonal entries improves their stability and reduces prediction variance. The Shrink Covariances option shrinks the off-diagonal entries by a factor that is determined using the method described in Schafer and Strimmer (2005).

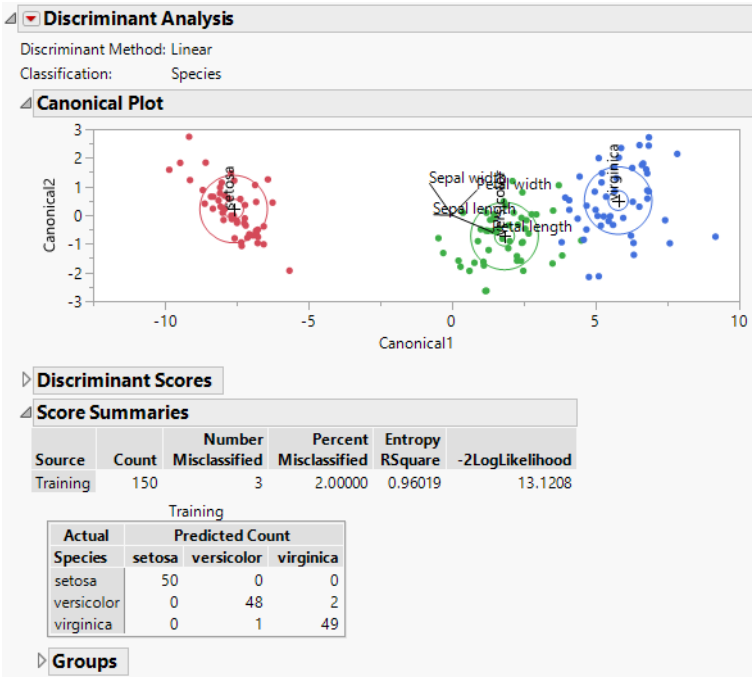
If you select the Shrink Covariances option with the Linear discriminant method in the launch window, this provides a shrinkage of the covariance matrices that is equivalent to the shrinkage provided by the Regularized discriminant method with appropriate Lambda and Gamma values. When you select the Shrink Covariances option and run your analysis, the Shrinkage report gives you an Overall Shrinkage value and an Overall Lambda value. To obtain the same analysis using the Regularized method, enter 1 as Lambda and the Overall Lambda from the Shrinkage report as Gamma in the Regularization Parameters window.

Discriminant Analysis Report

The Discriminant Analysis report provides discriminant results based on your selected Discriminant Method. The Discriminant Method and the Classification variable are shown at the top of the report. If you selected the Regularized method, its associated parameters are also shown.

You can change Discriminant Method by selecting the option from the Discriminant Analysis red triangle menu. The results in the report update to reflect the selected method.

Figure 5.8 Example of a Discriminant Analysis Report



The default Discriminant Analysis report contains the following sections:

- When you select the Wide Linear discriminant method, a Principal Components report appears. See "Principal Components".
- The Canonical Plot shows the points and multivariate means in the two dimensions that best separate the groups. See "Canonical Plot and Canonical Structure".
- The Discriminant Scores report provides details about how each observation is classified. See "Discriminant Scores".
- The Score Summaries report provides an overview of how well observations are classified. See "Discriminant Scores".

Principal Components

This report appears only for the Wide Linear method. Consider the following notation:

- Denote the n by p matrix of covariates by $\mathbf{Y}$, where n is the number of observations and p is the number of covariates.
- For each observation in $\mathbf{Y}$, subtract the covariate mean and divide the difference by the pooled standard deviation for the covariate. Denote the resulting matrix by $\mathbf{Y}_S$.

The report gives the following:

Number The number of eigenvalues extracted. Eigenvalues are extracted until Cum Percent is at least 99.99%, indicating that 99.99% of the variation has been explained.

Eigenvalue The eigenvalues of the covariance matrix for $\mathbf{Y}_S$, namely $(\mathbf{Y}_S'\mathbf{Y}_S)/(n - p)$, arranged in decreasing order.

Cum Percent The cumulative sum of the eigenvalues as a percentage of the sum of all eigenvalues. The eigenvalues sum to the rank of $\mathbf{Y}_S'\mathbf{Y}_S$.

Singular Value The singular values of $\mathbf{Y}_S$ arranged in decreasing order.

Canonical Plot and Canonical Structure

The Canonical Plot is a biplot that describes the canonical correlation structure of the variables.

Canonical Structure

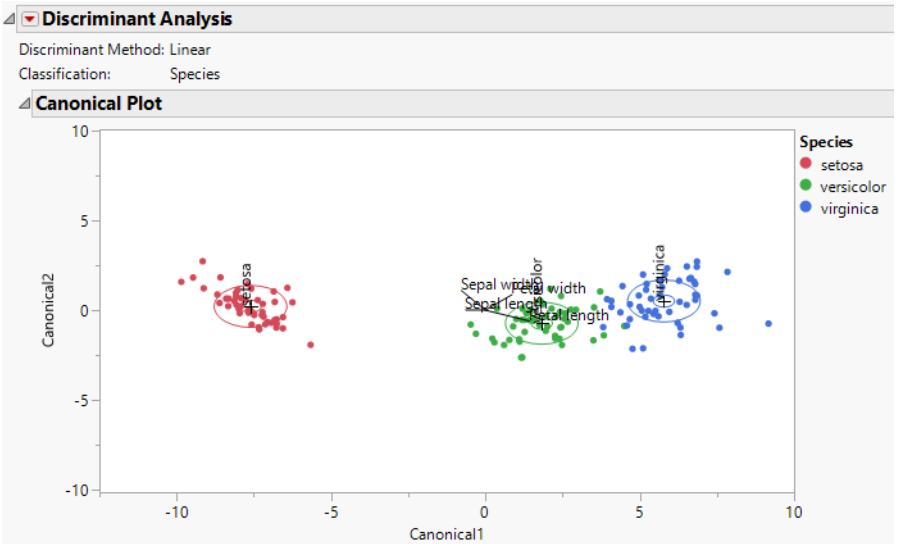
Each of the levels of the X, Categories column defines an indicator variable. A canonical correlation is performed between the set of indicator variables representing the categories and the covariates. Linear combinations of the covariates, called *canonical variables*, are derived. These canonical variables attempt to summarize the between-category variation.

The first canonical variable is the linear combination of the covariates that maximizes the multiple correlation between the category indicator variables and the covariates. The second canonical variable is a linear combination uncorrelated with the first canonical variable that maximizes the multiples correlation with the categories. If the X, Categories column has k levels, then $k - 1$ canonical variables are obtained.

Canonical Plot

Figure 5.9 shows the Canonical Plot for a linear discriminant analysis of the data table Iris.jmp. The points have been colored by Species.

Figure 5.9 Canonical Plot for Iris.jmp



The biplot axes are the first two canonical variables. These define the two dimensions that provide maximum separation among the groups. Each canonical variable is a linear combination of the covariates. (See "Canonical Structure".) The biplot shows how each observation is represented in terms of canonical variables and how each covariate contributes to the canonical variables.

- The observations and the multivariate means of each group are represented as points on the biplot. They are expressed in terms of the first two canonical variables.
  - The point corresponding to each multivariate mean is denoted by a plus ("+") marker.
  - A 95% confidence level ellipse is plotted for each mean. If two groups differ significantly, the confidence ellipses tend not to intersect.
  - An ellipse denoting a 50% contour is plotted for each group. This depicts a region in the space of the first two canonical variables that contains approximately 50% of the observations, assuming normality.
- The set of rays that appears in the plot represents the covariates.
  - For each canonical variable, the coefficients of the covariates in the linear combination can be interpreted as *weights*.
  - To facilitate comparisons among the weights, the covariates are standardized so that each has mean 0 and standard deviation 1. The coefficients for the standardized covariates are called the *canonical weights*. The larger the canonical weight of a covariate, the greater its association with the canonical variable.

- The length and direction of each ray in the biplot indicates the degree of association of the corresponding covariate with the first two canonical variables. The length of the rays is a multiple of the canonical weights.
- The rays emanate from the point (0,0), which represents the grand mean of the data in terms of the canonical variables.
- You can obtain the values of the weight coefficients by selecting **Canonical Options > Show Canonical Details** from the Discriminant Analysis red triangle menu. At the bottom of the Canonical Details report, click Standardized Scoring Coefficients. See "Standardized Scoring Coefficients".

Modifying the Canonical Plot

Additional options enable you to modify the biplot:

- Show or hide the 95% confidence ellipses by selecting **Canonical Options > Show Means CL Ellipses** from the Discriminant Analysis red triangle menu.
- Show or hide the rays by selecting **Canonical Options > Show Biplot Rays** from the Discriminant Analysis red triangle menu.
- Drag the center of the biplot rays to other places in the graph. Specify their position and scaling by selecting **Canonical Options > Biplot Ray Position** from the Discriminant Analysis red triangle menu. The default Radius Scaling shown in the Canonical Plot is 1.5, unless an adjustment is needed to make the rays visible.
- Show or hide the 50% contours by selecting **Canonical Options > Show Normal 50% Contours** from the Discriminant Analysis red triangle menu.
- Color code the points to match the ellipses by selecting **Canonical Options > Color Points** from the Discriminant Analysis red triangle menu.

Classification into Three or More Categories

For the Iris.jmp data, there are three Species, so there are only two canonical variables. The plot in Figure 5.9 shows good separation of the three groups using the two canonical variables.

The rays in the plot indicate the following:

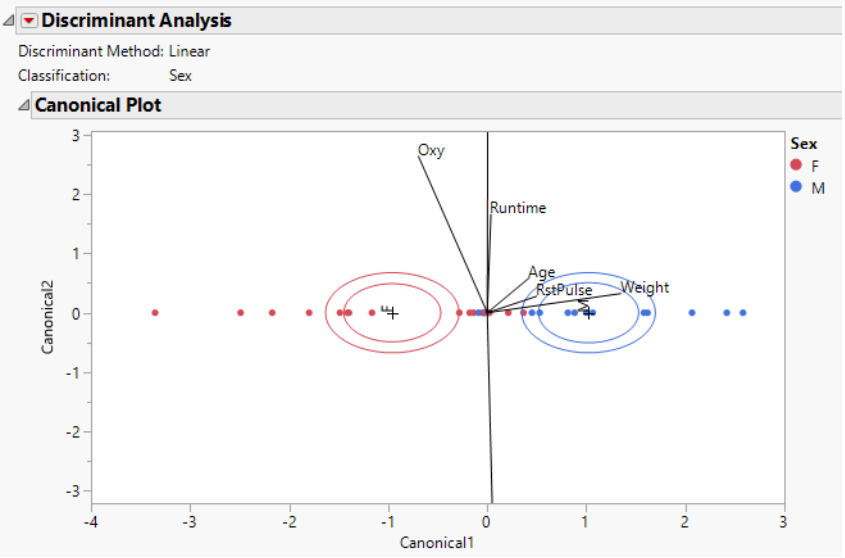
- Petal length is positively associated with Canonical1 and negatively associated with Canonical2. It carries more weight in defining Canonical1 than Canonical2.
- Petal width is positively associated with both Canonical1 and Canonical2. It carries about the same weight in defining both canonical variates.
- Sepal width is negatively associated with Canonical1 and positively associated with Canonical2. It carries more weight in defining Canonical2 than Canonical1.
- Sepal length is negatively weighted in terms of defining Canonical1 and very weakly associated in defining Canonical2.

Classification into Two Categories

When the classification variable has only two levels, the points are plotted against the single canonical variable, denoted by Canonical1 in the plot. The canonical weights for each covariate relate to Canonical1 only. The rays are shown with a vertical component only in order to separate them. Project the rays onto the Canonical1 axis to compare their relative association with the single canonical variable.

Figure 5.10 shows a Canonical Plot for the sample data table Fitness.jmp. The seven continuous variates are used to classify an individual into the categories M (male) or F (female). Since the classification variable has only two categories, there is only one canonical variable.

Figure 5.10 Canonical Plot for Fitness.jmp



The points in the Canonical Plot have been colored by Sex. Note that the two groups are well separated by their values on Canonical1.

Although the rays corresponding to the seven covariates have a vertical component, in this case you must interpret the rays only in terms of their projection onto the Canonical1 axis. You note the following:

- MaxPulse, Runtime, and RunPulse have little association with Canonical1.
- Weight, RstPulse, and Age are positively associated with Canonical1. Weight has the highest degree of association. The covariates RstPulse and Age have a similar, but smaller, degree of association.
- Oxy is negatively associated with Canonical1.

Discriminant Scores

The Discriminant Scores report provides the predicted classification of each observation and supporting information.

Row Row of the observation in the data table.

Actual Classification of the observation as given in the data table.

SqDist(Actual) Value of the saved formula SqDist[<level>] for the classification of the observation given in the data table. See "Score Options".

Note: Due to an offset term in the formula, SqDist(Actual) can be negative.

Prob(Actual) Estimated probability of the observation's actual classification.

-Log(Prob) Negative of the log of Prob(Actual). Large values of this negative log-likelihood identify observations that are poorly predicted in terms of membership in their actual categories.

A plot of -Log(Prob) appears to the right of the -Log(Prob) values. A large bar indicates a poor prediction. An asterisk(*) indicates observations that are misclassified.

If you are using a validation or a test set, observations in the validation set are marked with a "v" and those in the test set are marked with a "t".

Predicted Predicted classification of the observation. The predicted classification is the category with the highest predicted probability of membership.

Prob(Pred) Estimated probability of the observation's predicted classification.

Others Lists other categories, if they exist, that have a predicted probability that exceeds 0.1.

Figure 5.11 shows the Discriminant Scores report for the Iris.jmp sample data table using the Linear discriminant method. The option **Score Options > Show Interesting Rows Only** option is selected, showing only misclassified rows or rows with predicted probabilities between 0.05 and 0.95.

Figure 5.11 Show Interesting Rows Only

Discriminant Scores

Row	Actual	SqDist(Actual)	Prob(Actual)	-Log(Prob)		Predicted	Prob(Pred)	Others
71	versicolor	8.66970	0.2532	1.373	*	virginica	0.7468	
73	versicolor	4.87619	0.8155	0.204		versicolor	0.8155	virginica 0.18
78	versicolor	4.66698	0.6892	0.372		versicolor	0.6892	virginica 0.31
84	versicolor	8.43926	0.1434	1.942	*	virginica	0.8566	
120	virginica	8.19641	0.7792	0.249		virginica	0.7792	versicolor 0.22
124	virginica	3.57858	0.9029	0.102		virginica	0.9029	
127	virginica	3.90184	0.8116	0.209		virginica	0.8116	versicolor 0.19
128	virginica	3.31470	0.8658	0.144		virginica	0.8658	versicolor 0.13
130	virginica	9.08495	0.8963	0.109		virginica	0.8963	versicolor 0.10
134	virginica	7.23593	0.2706	1.307	*	versicolor	0.7294	
135	virginica	15.83301	0.9340	0.068		virginica	0.9340	
139	virginica	4.09385	0.8075	0.214		virginica	0.8075	versicolor 0.19

'*' indicates misclassified

Score Summaries

The Score Summaries report provides an overview of the discriminant scores. The table in Figure 5.12 shows Actual and Predicted classifications. If all observations are correctly classified, the off-diagonal counts are zero.

Figure 5.12 Score Summaries for Iris.jmp

Score Summaries

Source	Count	Number Misclassified	Percent Misclassified	Entropy RSquare	-2LogLikelihood
Training	65	1	1.53846	0.94487	7.81888
Validation	49	3	6.12245	0.86644	
Test	36	0	0.00000	0.98410	

Training

Actual Species	Predicted Count setosa	versicolor	virginica
setosa	18	0	0
versicolor	0	22	1
virginica	0	0	24

Actual Species	Predicted Rate setosa	versicolor	virginica
setosa	1.000	0.000	0.000
versicolor	0.000	0.957	0.043
virginica	0.000	0.000	1.000

Validation

Actual Species	Predicted Count setosa	versicolor	virginica
setosa	16	0	0
versicolor	0	15	2
virginica	0	1	15

Actual Species	Predicted Rate setosa	versicolor	virginica
setosa	1.000	0.000	0.000
versicolor	0.000	0.882	0.118
virginica	0.000	0.063	0.938

Test

Actual Species	Predicted Count setosa	versicolor	virginica
setosa	16	0	0
versicolor	0	10	0
virginica	0	0	10

Actual Species	Predicted Rate setosa	versicolor	virginica
setosa	1.000	0.000	0.000
versicolor	0.000	1.000	0.000
virginica	0.000	0.000	1.000

The Score Summaries report provides the following information:

Columns If you used Stepwise Variable Selection to construct the model, the columns entered into the model are listed (Figure 5.6).

Source If no validation is used, all observations comprise the Training set. If validation is used, a row is shown for the Training and Validation sets, or for the Training, Validation, and Test sets.

Number Misclassified Provides the number of observations in the specified set that are incorrectly classified.

Percent Misclassified Provides the percent of observations in the specified set that are incorrectly classified.

Entropy RSquare A measure of fit. Larger values indicate better fit. An Entropy RSquare value of 1 indicates that the classifications are perfectly predicted. Because uncertainty in the predicted probabilities is typical for discriminant models, Entropy RSquare values tend to be small.

See "Entropy RSquare".

Note: It is possible for Entropy RSquare to be negative.

-2LogLikelihood Twice the negative log-likelihood of the observations in the training set, based on the model. Smaller values indicate better fit. Provided for the training set only. See *Fitting Linear Models*.

Confusion Matrices Shows matrices of actual by predicted counts for each level of the categorical X. If you are using JMP Pro with validation, a matrix is given for each set of observations. If you are using JMP with excluded rows, the excluded rows are considered the validation set and a separate Validation matrix is given. See "Validation in JMP and JMP Pro".

Entropy RSquare

The Entropy RSquare is a measure of fit. It is computed for the training set and for the validation and test sets if validation is used.

Entropy RSquare for the Training Set

For the training set, Entropy RSquare is computed as follows:

- A discriminant model is fit using the training set.
- Predicted probabilities based on the model are obtained.
- Using these predicted probabilities, the likelihood is computed for observations in the training set. Call this $Likelihood_Full_{Training}$.
- The reduced model (no predictors) is fit using the training set.
- The predicted probabilities for the levels of X from the reduced model are used to compute the likelihood for observations in the training set. Call this quantity $Likelihood_Reduced_{Training}$.
- The Entropy RSquare for the training set is:

$$\text{Entropy RSquare}_{Training} = 1 - \frac{\log(Likelihood_Full_{Training})}{\log(Likelihood_Reduced_{Training})}$$

Entropy RSquare for Validation and Test Sets

For the validation set, Entropy RSquare is computed as follows:

- A discriminant model is fit using only the training set.
- Predicted probabilities based on the training set model are obtained for all observations.
- Using these predicted probabilities, the likelihood is computed for observations in the validation set. Call this $Likelihood_Full_{Validation}$.
- The reduced model (no predictors) is fit using only the training set.
- The predicted probabilities for the levels of X from the reduced model are used to compute the likelihood for observations in the validation set. Call this quantity $Likelihood_Reduced_{Validation}$.
- The Validation Entropy RSquare is:

$$\text{Validation Entropy RSquare} = 1 - \frac{\log(Likelihood_Full_{Validation})}{\log(Likelihood_Reduced_{Validation})}$$

The Entropy RSquare for the test set is computed in a manner analogous to the Entropy RSquare for the Validation set.

Discriminant Analysis Options

The Discriminant Analysis red triangle menu contains the following options:

Stepwise Variable Selection (Not available for the Wide Linear method.) Shows or hides the Column Selection control panel. This control panel contains options that enable you to perform stepwise variable selection using covariance analysis and *p*-values. See "Stepwise Variable Selection".

Discriminant Method Specifies the discriminant method. Choose from Linear, Quadratic, Regularized, and Wide Linear. See "Discriminant Methods".

Discriminant Scores Shows or hides a table of the discriminant scores for each row.

Score Options Provides options for the scoring of the observations. In particular, you can save the scoring formulas. See "Score Options".

Canonical Plot Shows or hides the Canonical Plot. See "Canonical Plot and Canonical Structure".

Canonical Options Provides options that affect the Canonical Plot. See "Canonical Options".

Canonical 3D Plot Shows a three-dimensional canonical plot. This option is available only when there are four or more levels of the categorical X. See "Example of a Canonical 3D Plot".

Specify Priors Enables you to specify prior probabilities for each level of the *X* variable. See "Specify Priors".

Consider New Levels Specifies that some points might not fit into any known group and should be considered to be from an unscored new group. See "Consider New Levels".

Show Within Covariances Shows or hides these reports:

- A Covariance Matrices report that gives the pooled-within covariance and correlation matrices.
- For the Quadratic and Regularized methods, a Correlations for Each Group report that shows the within-group correlation matrices.

  For each group, the log of the determinant of the within-group covariance matrix is also shown.
- For the Quadratic discriminant method, adds a Group Covariances outline to the Covariance Matrices report that shows the within-group covariance matrices.

Show Within Covariances is not available for the Wide Linear discriminant method.

Show Group Means Shows or hides the Group Means report that provides the mean of each covariate. Means for each level of the *X* variable and overall means appear.

Save Discrim Matrices Saves a script called `Discrim Results` to the data table. The script is a list of the following objects for use in JSL:

- a list of the covariates (Ys)
- the categorical variable X
- a list of the levels of X
- a matrix of the means of the covariates by the levels of X
- the pooled-within covariance matrix

Save Discrim Matrices is not available for the Wide Linear discriminant method. See "Save Discrim Matrices".

Scatterplot Matrix Opens a Scatterplot Matrix report that shows a matrix with a scatterplot for each pair of covariates. The option invokes the Scatterplot Matrix platform with shaded density ellipses for each group. The scatterplots include all observations in the data table, even if validation is used. See "Scatterplot Matrix".

Not available for the Wide Linear discriminant method.

Profiler Shows or hides an interactive profiler report where the categorical probabilities are combined into a single profiler row. Changes in the factor values are reflected in the estimated classification probabilities. For more information about the options in the red triangle menu, see *Profilers*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Score Options

Score Options provides the following selections that deal with scores:

Show Interesting Rows Only In the Discriminant Scores report, shows only rows that are misclassified and those with predicted probability between 0.05 and 0.95.

Show Classification Counts Shows or hides the confusion matrices and confusion rates matrices in the Score Summaries report. A confusion matrix is a two-way classification of actual and predicted responses. A confusion rates matrix is equal to the confusion matrix, with the numbers divided by the row totals. By default, the Score Summaries report shows a confusion matrix and a confusion rates matrix for each level of the categorical X. If you are using JMP Pro with validation, a matrix is given for each set of observations. If you are using JMP with excluded rows, these rows are considered the validation set and a separate Validation matrix is given. See "Validation in JMP and JMP Pro".

Show Distances to Each Group Shows or hides a report that contains each observation's squared Mahalanobis distance to each group mean.

Show Probabilities to Each Group Shows or hides a report that contains the probability that an observation belongs to each of the groups defined by the categorical X.

ROC Curve Shows or hides a Receiver Operating Characteristic (ROC) curve to the Score Summaries report. For more information about the ROC Curve, see *Predictive and Specialized Modeling*.

Select Misclassified Rows Selects the misclassified rows in the data table and in report windows that display a listing by row.

Select Uncertain Rows Selects rows with uncertain classifications in the data table and in report windows that display a listing by row. An uncertain row is one whose probability of group membership for *any* group is neither close to 0 nor close to 1.

When you select this option, a window opens where you can specify the range of predicted probabilities that reflect uncertainty. By default, any row whose probability differs from 0 or 1 by more than 0.1 is defined to be uncertain. Therefore, the default selects rows with probabilities between 0.1 and 0.9.

Save Formulas Saves distance, probability, and predicted membership formulas to the data table. See "Saved Formulas".

- The distance formulas are SqDist0 and SqDist[<level>], where <level> represents a level of X. The distance formulas produce intermediate values connected with the Mahalanobis distance calculations.
- The probability formulas are Prob[<level>], where <level> represents a level of X. Each probability column gives the posterior probability of an observation's membership in that level of X. The Response Probability column property is saved to each probability column. For more information about the Response Probability column property, see *Using JMP*.
- The predicted membership formula is Pred <X> and contains the "most likely level" classification rule.
- The Wide Linear method also saves a Discrim Data Matrix column containing the vector of covariates and a Discrim Prin Comp formula. See "Wide Linear Discriminant Method".

Note: For any method other than Wide Linear, when you Save Formulas, a RowEdit Prob script is saved to the data table. This script selects uncertain rows in the data table. The script defines any row whose probability differs from 0 or 1 by more than 0.1 as uncertain. The script also opens a Row Editor window that enables you to examine the uncertain rows. If you fit a new model (other than Wide Linear) and select Save Formulas, any existing RowEdit Prob script is replaced with a script that applies to the new fit.

Make Scoring Script (Available only in JMP Standard.) Creates a script that constructs the formula columns saved by the Save Formulas option. You can save this script and use it, perhaps with other data tables, to create the formula columns that calculate membership probabilities and predict group membership.

JMP PRO **Publish Probability Formulas** (Available only in JMP Pro.) Creates probability formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula

Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Canonical Options

The first options listed below relate to the appearance of the Canonical Plot or the Canonical 3D Plot. The remaining options provide detail on the calculations related to the plot.

Note: The Canonical 3D Plot is available only when there are three or more covariates and when the grouping variable has four or more categories.

Options Relating to Plot Appearance

Show Points Shows or hides the points in the Canonical Plot and the Canonical 3D Plot.

Show Means CL Ellipses Shows or hides 95% confidence ellipses for the mean of each group in the Canonical Plot and the Canonical 3D Plot, assuming normality.

Show Normal 50% Contours Shows or hides a 50% prediction ellipse or ellipsoid for each group. In the Canonical Plot, each ellipse depicts a region in the space of the first two canonical variables that estimates where 50% of the observations for each group should fall, assuming multivariate normality. In the Canonical 3D Plot, each ellipsoid depicts a region in the space of the first three canonical variables that estimates where 50% of the observations should fall, assuming multivariate normality.

Show Biplot Rays Shows or hides the biplot rays in the Canonical Plot and the Canonical 3D Plot. The labeled rays show the directions of the covariates in the canonical space. They represent the degree of association of each covariate with each canonical variable.

Biplot Ray Position Enables you to specify the position and radius scaling of the biplot rays in the Canonical Plot and in the Canonical 3D Plot.

- By default, the rays emanate from the point (0,0), which represents the grand mean of the data in terms of the canonical variables. In the Canonical Plot, you can drag the rays or use this option to specify coordinates.
- The default Radius Scaling in the canonical plots is 1.5, unless an adjustment is needed to make the rays visible. Radius Scaling is done relative to the Standardized Scoring Coefficients.

Color Points Colors the points in the Canonical Plot and the Canonical 3D Plot based on the levels of the *X* variable. Color markers are added to the rows in the data table. This option is equivalent to selecting **Rows > Color or Mark by Column** and selecting the *X* variable. It is also equivalent to right-clicking the graph and selecting **Row Legend**, and then coloring by the classification column.

Options Relating to Calculations

Show Canonical Details Shows or hides the Canonical Details report. See "Show Canonical Details".

Show Canonical Structure Shows or hides the Canonical Structures report. See "Show Canonical Structure". Not available for the Wide Linear discriminant method.

Save Canonical Scores Creates columns in the data table that contain canonical score formulas for each observation. The column for the k^{th} canonical score is named Canon[<k>].

Tip: In a script, sending the scripting command **Save to New Data Table** to the Discriminant object saves the following to a new data table: group means on the canonical variables; the biplot rays with 1.5 Radius Scaling of the Standardized Scoring Coefficients; and the canonical scores. Not available for the Wide Linear discriminant method.

Show Canonical Details

The Canonical Details report shows tests that address the relationship between the covariates and the grouping variable *X*. Relevant matrices are presented at the bottom of the report.

Figure 5.13 Canonical Details for Iris.jmp

Canonical Details

Canonical Details calculated from the overall pooled within-group covariance matrix.

Eigenvalue	Percent	Cum Percent	Canonical Corr	Likelihood Ratio	Approx. F	NumDF	DenDF	Prob>F
32.1919292	99.1213	99.1213	0.98482089	0.02343863	199.1453	8	288	<.0001*
0.28539104	0.8787	100.0000	0.47119702	0.77797337	13.7939	3	145	<.0001*

Test	Value	Approx. F	NumDF	DenDF	Prob>F
Wilks' Lambda	0.0234386	199.1453	8	288	<.0001*
Pillai's Trace	1.1918988	53.4665	8	290	<.0001*
Hotelling-Lawley	32.47732	582.1970	8	203.4	<.0001*
Roy's Max Root	32.191929	1166.9574	4	145	<.0001*

▷ Within Matrix
▷ Between Matrix
▷ Scoring Coefficients
▷ Standardized Scoring Coefficients

Note: The matrix used in computing the results in the report is the pooled within-covariance matrix (given as the Within Matrix). This matrix is used as a basis for the Canonical Details report for all discriminant methods. The statistics and tests in the Canonical Details report are the same for all discriminant methods.

Statistics and Tests

The Canonical Details report lists eigenvalues and gives a likelihood ratio test for zero eigenvalues. Four tests are provided for the null hypothesis that the canonical correlations are zero.

Eigenvalue Eigenvalues of the product of the Between Matrix and the inverse of the Within Matrix. These are listed from largest to smallest. The size of an eigenvalue reflects the amount of variance explained by its associated discriminant function.

Percent Proportion of the sum of the eigenvalues represented by the given eigenvalue.

Cum Percent Cumulative sum of the proportions.

Canonical Corr Canonical correlations between the covariates and the groups defined by the categorical *X*. Suppose that you define numeric indicator variables to represent the groups defined by *X*. Then perform a canonical correlation analysis using the covariates as one set of variables and the indicator variables representing the groups in *X* as the other. The Canonical Corr values are the canonical correlation values that result from this analysis.

Likelihood Ratio Likelihood ratio statistic for a test of whether the population values of the corresponding canonical correlation and all smaller correlations are zero. The ratio equals the product of the values (1 - Canonical Corr^2) for the given and all smaller canonical correlations.

Test Lists four standard tests for the null hypothesis that the means of the covariates are equal across groups: Wilk's Lambda, Pillai's Trace, Hotelling-Lawley, and Roy's Max Root. See "Multivariate Tests" and "Approximate F-Tests".

Approx. F The *F* value associated with the corresponding test. For certain tests, the *F* value is approximate or an upper bound. See "Approximate F-Tests".

NumDF The numerator degrees of freedom for the corresponding test.

DenDF The denominator degrees of freedom for the corresponding test.

Prob>F The *p*-value for the corresponding test.

Matrices

Four matrices that relate to the canonical structure are presented at the bottom of the report. To view a matrix, click the disclosure icon beside its names. To hide it, click the name of the matrix.

Within Matrix Pooled within-covariance matrix.

Between Matrix Between groups covariance matrix, $\mathbf{S}_B$. See "Between Groups Covariance Matrix".

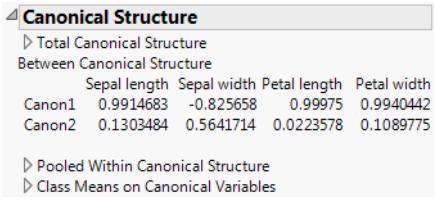
Scoring Coefficients Coefficients used to compute canonical scores in terms of the raw data. These are the coefficients used for the option **Canonical Options > Save Canonical Scores**. For more information about how these are computed, see the CANDISC Procedure chapter in SAS Institute Inc. (2020b).

Standardized Scoring Coefficients Coefficients used to compute canonical scores in terms of the standardized data. Often called *canonical weights*. For more information about how these are computed, see the CANDISC Procedure chapter in SAS Institute Inc. (2020b).

Show Canonical Structure

The Canonical Structure report gives three matrices that provide correlations between the canonical variables and the covariates. Another matrix shows means across the levels of the group variable. To view a matrix, click the disclosure icon beside its names. To hide it, click the name of the matrix.

Figure 5.14 Canonical Structure for Iris.jmp Showing between Canonical Structure



Total Canonical Structure Correlations between the canonical variables and the covariates. Often called *loadings*.

Between Canonical Structure Correlations between the group means on the canonical variables and the group means on the covariates.

Pooled Within Canonical Structure Partial correlations between the canonical variables and the covariates, adjusted for the group variable.

Class Means on Canonical Variables Provides means across the levels of the group variable for each canonical variable.

Example of a Canonical 3D Plot

1. Select **Help > Sample Data Library** and open Owl Diet.jmp.
2. Select rows 180 through 294.

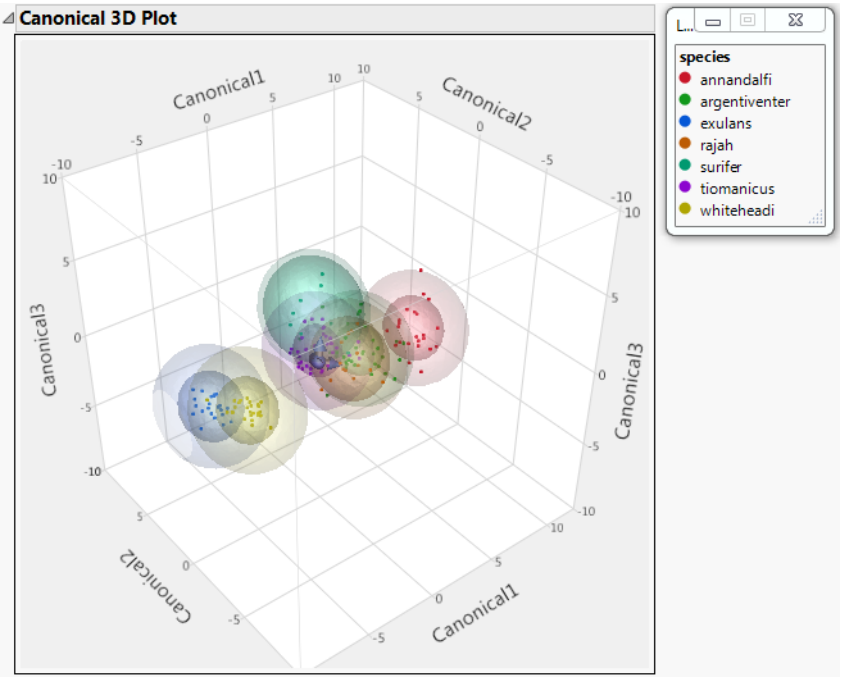
   These are the rows for which species is missing. Hide and exclude these rows.
3. Select **Rows > Hide and Exclude**.
4. Select **Rows > Color or Mark by Column**.

5. Select species.
6. From the Colors menu, select **JMP Dark**.
7. Check **Make Window with Legend**.
8. Click **OK**.

   A small Legend window appears. The rows in the data table are assigned colors by species.
9. **Select Analyze > Multivariate Methods > Discriminant**.
10. Specify skull length, teeth row, palatine foramen, and jaw length as **Y, Covariates**.
11. Specify species as **X, Categories**.
12. Click **OK**.
13. Click the Discriminant Analysis red triangle and click **Canonical 3D Plot**.

Tip: Click categories in the Legend to highlight those points in the Canonical 3D plot. Click and drag inside the 3D plot to rotate it.

Figure 5.15 Canonical 3D Plot with Legend Window



Specify Priors

The following options are available for specifying priors:

Equal Probabilities Assigns equal prior probabilities to all groups. This is the default.

Proportional to Occurrence Assigns prior probabilities to the groups that are proportional to their frequency in the observed data.

Other Enables you to specify custom prior probabilities.

Consider New Levels

Use the Consider New Levels option if you suspect that some of your observations are outliers with respect to the specified levels of the categorical variable. When you select the option, a menu asks you to specify the prior probability of the new level.

Observations that would be better fit using a new group are assigned to the new level, called "Other". Probability of membership in the Other group assumes that these observations have the distribution of the entire set of observations where *no group structure* is assumed. This leads to correspondingly wide normal contours associated with the covariance structure. Distance calculations are adjusted by the specified prior probability.

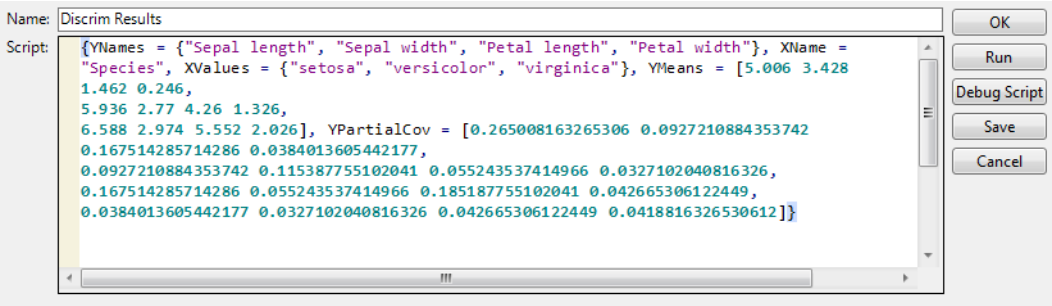
Save Discrim Matrices

Save Discrim Matrices creates a global list (`DiscrimResults`) for use in the JMP scripting language. The list contains the following, calculated for the training set:

- YNames, a list of the covariates (Ys)
- XName, the categorical variable
- XValues, a list of the levels of X
- YMeans, a matrix of the means of the covariates by the levels of X
- YPartialCov, the within covariance matrix

Consider the analysis obtained using the **Discriminant** script in the Iris.jmp sample data table. If you select **Save Discrim Matrices** from the Discriminant Analysis red triangle menu, the script **Discrim Results** is saved to the data table.

Figure 5.16 Discrim Results Table Script for Iris.jmp



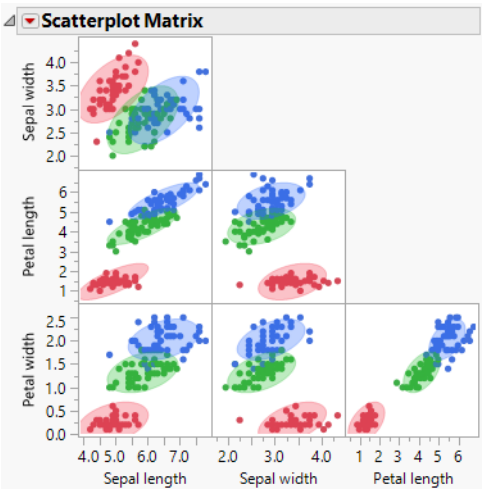
Note: In a script, you can send the scripting command Get Discrim Matrices to the Discriminant platform object. This obtains the same values as Save Discrim Matrices, but does not store them in the data table.

Scatterplot Matrix

The Scatterplot Matrix command invokes the Scatterplot Matrix platform in a separate window containing a lower triangular scatterplot matrix for the covariates. Points are plotted for all observations in the data table.

Ellipses with 90% coverage are shown for each level of the categorical variable X. For the Linear discriminant method, these are based on the pooled within covariance matrix. Figure 5.17 shows the Scatterplot Matrix window for the Iris.jmp sample data table.

Figure 5.17 Scatterplot Matrix for Iris.jmp



The options in the Scatterplot Matrix red triangle menu are described in *Essential Graphing*.

Validation in JMP and JMP Pro

In JMP, you can specify a validation set by excluding the rows that form the validation set. Select the rows that you want to use as your validation set and then select **Rows > Exclude/Unexclude**. The unexcluded rows are treated as the training set.

Note: In JMP Pro, you can specify a Validation column in the Discriminant launch window. A validation column must have a numeric data type and should contain at least two distinct values.

Notice the following:

- If the column contains two values, the smaller value defines the training set and the larger value defines the validation set.
- If the column contains three values, the values define the training, validation, and test sets in order of increasing size.
- If the column contains four or more distinct values, only the smallest three values and their associated observations are used to define the training, validation, and test sets, in that order.

When a validation set is specified, the Discriminant platform does the following:

- Models are fit using the training data.
- The Stepwise Variable Selection option gives the Validation Entropy RSquare and Validation Misclassification Rate statistics for the model. See "Statistics" and "Entropy RSquare for Validation and Test Sets".
- The Discriminant Scores report shows an indicator identifying rows in the validation and test sets.
- The Score Summaries report shows actual by predicted classifications for the training, validation, and test sets.

Statistical Details for the Discriminant Analysis Platform

- "Description of the Wide Linear Algorithm"
- "Saved Formulas"
- "Multivariate Tests"
- "Approximate F-Tests"
- "Between Groups Covariance Matrix"

Description of the Wide Linear Algorithm

Wide Linear discriminant analysis is performed as follows:

- The data are standardized by subtracting group means and dividing by pooled standard deviations.
- The singular value decomposition is used to obtain a principal component transformation matrix from the set of singular vectors.
- The number of components retained represents a minimum of 0.9999 of the sum of the squared singular values.
- A linear discriminant analysis is performed on the transformed data, where the data are not shifted by group means. This is a fast calculation because the pooled-within covariance matrix is diagonal.

Saved Formulas

This section gives the derivation of formulas saved by **Score Options > Save Formulas**. The formulas depend on the Discriminant Method.

For each group defined by the categorical variable X, observations on the covariates are assumed to have a p-dimensional multivariate normal distribution, where p is the number of covariates. The notation used in the formulas is given in Table 5.2.

Table 5.2 Notation for Formulas Given by Save Formulas Options

p	number of covariates
T	total number of groups (levels of X)

Table 5.2 Notation for Formulas Given by Save Formulas Options *(Continued)*

$t = 1,..., T$	subscript to distinguish groups defined by X
n_t	number of observations in group t
$n = n_1 + n_2 + ... + n_T$	total number of observations
$\mathbf{y}$	p by 1 vector of covariates for an observation
$\mathbf{y}_{it} = (y_{i1t}, y_{i2t}, \ldots, y_{ipt})$	i^{th} observation in group t, consisting of a vector of p covariates
$\bar{\mathbf{y}}_t$	p by *1* vector of means of the covariates $\mathbf{y}$ for observations in group t
$\mathbf{y}_{bar}$	p by *1* vector of means for the covariates across all observations
$\mathbf{S}_t = \frac{1}{n_t - 1} \sum_{i=1}^{n_t} (\mathbf{y}_{it} - \bar{\mathbf{y}}_t)(\mathbf{y}_{it} - \bar{\mathbf{y}}_t)'$	estimated (p by p) within-group covariance matrix for group t
$\mathbf{S}_p = \frac{1}{n - T} \sum_{t=1}^{T} (n_t - 1)\mathbf{S}_t$	estimated (p by p) pooled within covariance matrix
q_t	prior probability of membership for group t
$p(t \mid \mathbf{y})$	posterior probability that $\mathbf{y}$ belongs to group t
$\lvert \mathbf{A} \rvert$	determinant of a matrix A

Linear Discriminant Method

In linear discriminant analysis, all within-group covariance matrices are assumed equal. The common covariance matrix is estimated by $\mathbf{S}_p$. See Table 5.2 for notation.

The Mahalanobis distance from an observation $\mathbf{y}$ to group t is defined as follows:

$$d_t^2 = (\mathbf{y} - \bar{\mathbf{y}}_t)' \mathbf{S}_p^{-1} (\mathbf{y} - \bar{\mathbf{y}}_t)$$

The likelihood for an observation $\mathbf{y}$ in group t is estimated as follows:

$$\begin{aligned} l_t(\mathbf{y}) &= (2\pi)^{-T/2} \left|\mathbf{S}_p\right|^{-1/2} \exp(-(\mathbf{y} - \bar{\mathbf{y}}_t)' \mathbf{S}_p^{-1} (\mathbf{y} - \bar{\mathbf{y}}_t)/2) \\ &= (2\pi)^{-T/2} \left|\mathbf{S}_p\right|^{-1/2} \exp(-d_t^2/2) \end{aligned}$$

Note that the number of parameters that must be estimated for the pooled covariance matrix is $p(p+1)/2$ and for the means is Tp. The total number of parameters that must be estimated is $p(p+1)/2 + Tp$.

The posterior probability of membership in group t is defined as follows:

$$p(t|\mathbf{y}) = \frac{q_t l_t(\mathbf{y})}{\sum_{u=1}^{T} q_u l_u(\mathbf{y})} = \frac{1}{1 + \sum_{u \neq t} \exp(-[(d_u^2 - 2\log(q_u)) - (d_t^2 - 2\log(q_t))]/2)}$$

An observation $\mathbf{y}$ is assigned to the group for which its posterior probability is the largest.

The formulas saved by the Linear discriminant method are defined as follows:

SqDist[0]	$\mathbf{y}' \mathbf{S}_p^{-1} \mathbf{y}$
SqDist[<group *t*>]	$d_t^2 - 2\log(q_t)$
Prob[<group *t*>]	$p(t\|\mathbf{y})$
Pred <*X*>	t for which $p(t\|\mathbf{y})$ is maximum, $t = 1, \ldots, T$

Quadratic Discriminant Method

In quadratic discriminant analysis, the within-group covariance matrices are not assumed equal. The within-group covariance matrix for group t is estimated by $\mathbf{S}_t$. This means that the number of parameters that must be estimated for the within-group covariance matrices is $Tp(p+1)/2$ and for the means is Tp. The total number of parameters that must be estimated is $Tp(p+3)/2$.

When group sample sizes are small relative to p, the estimates of the within-group covariance matrices tend to be highly variable. The discriminant score is heavily influenced by the smallest eigenvalues of the inverse of the within-group covariance matrices. See Friedman (1989). For this reason, if your group sample sizes are small compared to p, you might want to consider the Regularized method, described in "Regularized Discriminant Method".

See Table 5.2 for notation. The Mahalanobis distance from an observation $\mathbf{y}$ to group t is defined as follows:

$$d_t^2 = (\mathbf{y} - \bar{\mathbf{y}}_t)' \mathbf{S}_t^{-1} (\mathbf{y} - \bar{\mathbf{y}}_t)$$

The likelihood for an observation $\mathbf{y}$ in group t is estimated as follows:

$$\begin{aligned} l_t(\mathbf{y}) &= (2\pi)^{-T/2} |\mathbf{S}_t|^{-1/2} \exp(-(\mathbf{y} - \bar{\mathbf{y}}_t)' \mathbf{S}_t^{-1} (\mathbf{y} - \bar{\mathbf{y}}_t)/2) \\ &= (2\pi)^{-T/2} |\mathbf{S}_t|^{-1/2} \exp(-d_t^2/2) \end{aligned}$$

The posterior probability of membership in group t is the following:

$$\begin{aligned} p(t|\mathbf{y}) &= (q_t l_t(\mathbf{y})) / \left(\sum_{u=1}^{T} q_u l_u(\mathbf{x}) \right) \\ &= \frac{1}{1 + \sum_{u \neq t} \exp(-[(d_u^2 + \log|\mathbf{S}_u| - 2\log(q_u)) - (d_t^2 + \log|\mathbf{S}_t| - 2\log(q_t))]/2)} \end{aligned}$$

An observation $\mathbf{y}$ is assigned to the group for which its posterior probability is the largest.

The formulas saved by the Quadratic discriminant method are defined as follows:

SqDist[<group t>]	$d_t^2 + \log\|\mathbf{S}_t\| - 2\log(q_t)$

Prob[<group *t*>]	$p(t\|\mathbf{y})$
Pred <*X*>	t for which $p(t\|\mathbf{y})$ is maximum, $t = 1, \ldots, T$

Note: SqDist[<group *t*>] can be negative.

Regularized Discriminant Method

Regularized discriminant analysis allows for two parameters: λ and γ.

- The parameter λ balances weights assigned to the pooled covariance matrix and the within-group covariance matrices, which are not assumed equal.
- The parameter γ determines the amount of shrinkage toward a diagonal matrix.

This method enables you to leverage two aspects of regularization to bring stability to estimates for quadratic discriminant analysis. See Friedman (1989). See Table 5.2 for notation.

For the regularized method, the covariance matrix for group t is:

$$\Sigma_t = (1-\gamma)(\lambda \mathbf{S}_p + (1-\lambda)\mathbf{S}_t) + \gamma Diag((\lambda \mathbf{S}_p + (1-\lambda)\mathbf{S}_t))$$

The Mahalanobis distance from an observation $\mathbf{y}$ to group t is defined as follows:

$$d_t^2 = (\mathbf{y} - \bar{\mathbf{y}}_t)' \Sigma_t^{-1} (\mathbf{y} - \bar{\mathbf{y}}_t)$$

The likelihood for an observation $\mathbf{y}$ in group t is estimated as follows:

$$\begin{aligned} l_t(\mathbf{y}) &= (2\pi)^{-T/2} \left|\Sigma_t\right|^{-1/2} \exp(-(\mathbf{y} - \bar{\mathbf{y}}_t)' \Sigma_t^{-1} (\mathbf{y} - \bar{\mathbf{y}}_t)/2) \\ &= (2\pi)^{-T/2} \left|\Sigma_t\right|^{-1/2} \exp(-d_t^2/2) \end{aligned}$$

The posterior probability of membership in group t given by the following:

$$p(t|\mathbf{y}) = (q_t l_t(\mathbf{y}))/\left(\sum_{u=1}^{T} q_u l_u(\mathbf{x})\right)$$

$$= \frac{1}{1 + \sum_{u \neq t} \exp(-[(d_u^2 + \log|\Sigma_u| - 2\log(q_u)) - (d_t^2 + \log|\Sigma_t| - 2\log(q_t))]/2)}$$

An observation $\mathbf{y}$ is assigned to the group for which its posterior probability is the largest.

The formulas saved by the Regularized discriminant method are defined below:

SqDist[<group *t*>]	$d_t^2 + \log\|\Sigma_t\| - 2\log(q_t)$
Prob[<group *t*>]	$p(t\|\mathbf{y})$
Pred <*X*>	t for which $p(t\|\mathbf{y})$ is maximum, $t = 1, \ldots, T$

Note: SqDist[<group *t*>] can be negative.

Wide Linear Discriminant Method

The Wide Linear method is useful when you have a large number of covariates and, in particular, when the number of covariates exceeds the number of observations ($p > n$). This approach centers around an efficient calculation of the inverse of the pooled within-covariance matrix $\mathbf{S}_p$ or of its transpose, if $p > n$. It uses a singular value decomposition approach to avoid inverting and allocating space for large covariance matrices.

The Wide Linear method assumes equal within-group covariance matrices and is equivalent to the Linear method if the number of observations equals or exceeds the number of covariates.

Wide Linear Calculation

See Table 5.2 for notation. The following steps are used in the Wide Linear calculation:

1. Compute the T by p matrix $\mathbf{M}$ of within-group sample means. The $(t,j)^{\text{th}}$ entry of $\mathbf{M}$, m_{tj}, is the sample mean for members of group t on the j^{th} covariate.

2. For each covariate j, calculate the pooled standard deviation across groups. Call this s_{jj}.
3. Denote the diagonal matrix with diagonal entries s_{jj} by $\mathbf{S}_{diag}$.
4. Center and scale values for each covariate:
   - Subtract the mean for the group to which the observation belongs.
   - Divide the difference by the pooled standard deviation.

   Using notation, for an observation i in group t, the group-centered and scaled value for the j^{th} covariate is:

$$y^*_{ij} = \frac{y_{ij} - m_{t(i)j}}{s_{jj}}$$

   The notation $t(i)$ indicates the group t to which observation i belongs.
5. Denote the matrix of y^*_{ij} values by $\mathbf{Y}_S$.
6. Denote the pooled within-covariance matrix for the group-centered and scaled covariates by $\mathbf{R}$. The matrix $\mathbf{R}$ is given by the following:

$$\mathbf{R} = (\mathbf{Y}_S'\mathbf{Y}_S)/(n - T)$$

7. Apply the singular value decomposition to $\mathbf{Y}_S$:

$$\mathbf{Y}_S = \mathbf{UDV}'$$

   where $\mathbf{U}$ and $\mathbf{V}$ are orthonormal and $\mathbf{D}$ is a diagonal matrix with positive entries (the singular values) on the diagonal. See "The Singular Value Decomposition".

   Then $\mathbf{R}$ can be written as follows:

$$\mathbf{R} = (\mathbf{Y}_S'\mathbf{Y}_S)/(n - T) = (\mathbf{VD}^2\mathbf{V}')/(n - T)$$

8. If $\mathbf{R}$ is of full rank, obtain $\mathbf{R}^{-1/2}$ as follows:

$$\mathbf{R}^{-1/2} = (\mathbf{VD}^{-1}\mathbf{V}')/\sqrt{n - T}$$

   where $\mathbf{D}^{-1}$ is the diagonal matrix whose diagonal entries are the inverses of the diagonal entries of $\mathbf{D}$.

   If $\mathbf{R}$ is not of full rank, define a pseudo-inverse for $\mathbf{R}$ as follows:

$$\mathbf{R}^- = (\mathbf{VD}^{-2}\mathbf{V}')/(n - T)$$

Then define the inverse square root of $\mathbf{R}$ as follows:

$$(\mathbf{R}^{-})^{1/2} = (\mathbf{V}\mathbf{D}^{-1}\mathbf{V}')/\sqrt{n-T}$$

9. If $\mathbf{R}$ is of full rank, it follows that $\mathbf{R}^{-} = \mathbf{R}^{-1}$. So, for completeness, the discussion continues using pseudo-inverses.

   Define a p by p matrix $\mathbf{T}_s$ as follows:

   $$\mathbf{T}_s = (\mathbf{S}_{diag}^{-1}\mathbf{V}\mathbf{D}^{-})/(\sqrt{n-T})$$

   Then:

   $$(\mathbf{T}_s\mathbf{T}_s') = \left(\mathbf{S}_{diag}^{-1}\mathbf{V}(\mathbf{D}^{-})^2\mathbf{V}'\mathbf{S}_{diag}^{-1}\right)/(n-T) = \mathbf{S}_{diag}^{-1}\mathbf{R}^{-}\mathbf{S}_{diag}^{-1} = \mathbf{S}_p^{-}$$

   where $\mathbf{S}_p^{-}$ is a generalized inverse of the pooled within-covariance matrix for the original data that is calculated using the SVD.

Mahalanobis Distance

The formulas for the Mahalanobis distance, the likelihood, and the posterior probabilities are identical to those in "Linear Discriminant Method". However, the inverse of $\mathbf{S}_p$ is replaced by a generalized inverse computed using the singular value decomposition.

When you save the formulas, the Mahalanobis distance is given in terms of the decomposition. For an observation $\mathbf{y}$, the squared distance to group t is the following, where *SqDist*[0] and *Discrim Prin Comp* in the last equality are defined in "Saved Formulas":

$$\begin{aligned}
d_t^2 &= (\mathbf{y}-\bar{\mathbf{y}}_t)'\mathbf{S}_p^{-}(\mathbf{y}-\bar{\mathbf{y}}_t) \\
&= (\mathbf{y}-\bar{\mathbf{y}}_t)'\mathbf{T}_s\mathbf{T}_s'(\mathbf{y}-\bar{\mathbf{y}}_t) \\
&= ((\mathbf{y}-\bar{\mathbf{y}})-(\bar{\mathbf{y}}_t-\bar{\mathbf{y}}))'\mathbf{T}_s\mathbf{T}_s'((\mathbf{y}-\bar{\mathbf{y}})-(\bar{\mathbf{y}}_t-\bar{\mathbf{y}})) \\
&= (\mathbf{T}_s'(\mathbf{y}-\bar{\mathbf{y}}))'(\mathbf{T}_s'(\mathbf{y}-\bar{\mathbf{y}})) - 2(\mathbf{T}_s'(\bar{\mathbf{y}}_t-\bar{\mathbf{y}}))'(\mathbf{T}_s'(\mathbf{y}-\bar{\mathbf{y}})) + (\mathbf{T}_s'(\bar{\mathbf{y}}_t-\bar{\mathbf{y}}))'(\mathbf{T}_s'(\bar{\mathbf{y}}_t-\bar{\mathbf{y}})) \\
&= SqDist[0] - 2(\mathbf{T}_s'(\bar{\mathbf{y}}_t-\bar{\mathbf{y}}))'Discrim\ Prin\ Comp + (\mathbf{T}_s'(\bar{\mathbf{y}}_t-\bar{\mathbf{y}}))'(\mathbf{T}_s'(\bar{\mathbf{y}}_t-\bar{\mathbf{y}}))
\end{aligned}$$

Saved Formulas

These are the formulas saved by the Wide Linear discriminant method:

Discrim Data Matrix	Vector of observations on the covariates

Discrim Prin Comp	The data transformed by the principal component scoring matrix, which renders the data uncorrelated within groups. Given by $\mathbf{T}_s'(\mathbf{y} - \bar{\mathbf{y}})$, where $\bar{\mathbf{y}}$ is a p by *1* vector containing the overall means.
SqDist[0]	$(\mathbf{y} - \bar{\mathbf{y}})'\mathbf{T}'_s\mathbf{T}_s'(\mathbf{y} - \bar{\mathbf{y}})$
SqDist[<group *t*>]	The Mahalanobis distance from the from observation to the group centroid. See "Mahalanobis Distance".
Prob[<group *t*>]	$p(t\|\mathbf{y})$, given in "Linear Discriminant Method"
Pred <*X*>	t for which $p(t\|\mathbf{y})$ is maximum, $t = 1, \ldots, T$

Multivariate Tests

In the following, **E** is the residual cross product matrix and **H** is the model cross product matrix. Diagonal elements of **E** are the residual sums of squares for each variable. Diagonal elements of **H** are the sums of squares for the model for each variable. In the discriminant analysis literature, **E** is often called **W**, where **W** stands for *within*.

Test statistics in the multivariate results tables are functions of the eigenvalues λ of $\mathbf{E}^{-1}\mathbf{H}$. The following list describes the computation of each test statistic.

Note: After specification of a response design, the initial **E** and **H** matrices are premultiplied by **M**′ and postmultiplied by **M**.

- Wilks' Lambda

$$\Lambda = \frac{\det(\mathbf{E})}{\det(\mathbf{H}+\mathbf{E})} = \prod_{i=1}^{n}\left(\frac{1}{1+\lambda_i}\right)$$

- Pillai's Trace

$$V = \text{Trace}[\mathbf{H}(\mathbf{H}+\mathbf{E})^{-1}] = \sum_{i=1}^{n}\frac{\lambda_i}{1+\lambda_i}$$

- Hotelling-Lawley Trace

$$U = \text{Trace}(\mathbf{E}^{-1}\mathbf{H}) = \sum_{i=1}^{n}\lambda_i$$

- Roy's Max Root

$\Theta = \lambda_1$, the maximum eigenvalue of $\mathbf{E}^{-1}\mathbf{H}$.

E and **H** are defined as follows:

$$\mathbf{E} = \mathbf{Y}'\mathbf{Y} - \mathbf{b}'(\mathbf{X}'\mathbf{X})\mathbf{b}$$

$$\mathbf{H} = (\mathbf{L}\mathbf{b})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{L}\mathbf{b})$$

where **b** is the estimated vector for the model coefficients and $\mathbf{A}^{-}$ denotes the generalized inverse of a matrix **A**.

The whole model **L** is a column of zeros (for the intercept) concatenated with an identity matrix having the number of rows and columns equal to the number of parameters in the model. **L** matrices for effects are subsets of rows from the whole model **L** matrix.

Approximate F-Tests

To compute *F*-values and degrees of freedom, let p be the rank of $\mathbf{H} + \mathbf{E}$. Let q be the rank of $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{L}'$, where the **L** matrix identifies elements of $\mathbf{X}'\mathbf{X}$ associated with the effect being tested. Let v be the error degrees of freedom and s be the minimum of p and q. Also let $m = 0.5(|p-q|-1)$ and $n = 0.5(v-p-1)$.

Table 5.3, gives the computation of each approximate F from the corresponding test statistic.

Table 5.3 Approximate *F*-statistics

Test	Approximate F	Numerator DF	Denominator DF
Wilks' Lambda	$F = \left(\frac{1-\Lambda^{1/t}}{\Lambda^{1/t}}\right)\left(\frac{rt-2u}{pq}\right)$	pq	$rt-2u$
Pillai's Trace	$F = \left(\frac{V}{s-V}\right)\left(\frac{2n+s+1}{2m+s+1}\right)$	$s(2m+s+1)$	$s(2n+s+1)$
Hotelling-Lawley Trace	$F = \frac{2(sn+1)U}{s^2(2m+s+1)}$	$s(2m+s+1)$	$2(sn+1)$

Table 5.3 Approximate *F*-statistics *(Continued)*

Test	Approximate F	Numerator DF	Denominator DF
Roy's Max Root	$F = \frac{\Theta(v - \max(p,q) + q)}{\max(p,q)}$	$\max(p,q)$	$v - \max(p,q) + q$

Between Groups Covariance Matrix

Using the notation in Table 5.2, this matrix is defined as follows:

$$\mathbf{S}_B = \frac{1}{T-1} \sum_{t=1}^{T} T\left(\frac{n_t}{n}\right)(\bar{\mathbf{y}}_t - \mathbf{y}_{bar})(\bar{\mathbf{y}}_t - \mathbf{y}_{bar})'$$

Chapter 6

Partial Least Squares Models

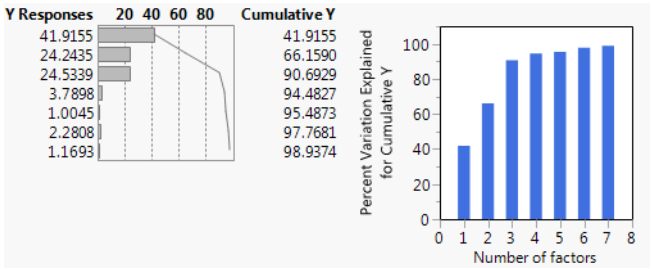
Develop Models Using Correlations between Ys and Xs

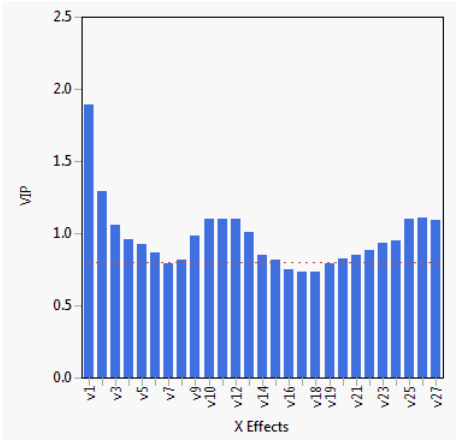
The Partial Least Squares (PLS) platform fits linear models based on factors, namely, linear combinations of the explanatory variables (*X*s). These factors are obtained in a way that attempts to maximize the covariance between the *X*s and the response or responses (*Y*s). PLS exploits the correlations between the *X*s and the *Y*s to reveal underlying latent structures.

JMP PRO JMP Pro provides additional functionality, enabling you to conduct PLS Discriminant Analysis (PLS-DA), include a variety of model effects, use several validation methods, impute missing data, and obtain bootstrap estimates of the distributions of various statistics.

Partial least squares performs well in situations such as the following, where the use of ordinary least squares does not produce satisfactory results: More X variables than observations; highly correlated X variables; a large number of X variables; several Y variables and many X variables.

Figure 6.1 A Portion of a Partial Least Squares Report





Contents

Overview of the Partial Least Squares Platform . . . 125
Example of Partial Least Squares . . . 126
Launch the Partial Least Squares Platform . . . 129
Centering and Scaling . . . 133
Standardize X . . . 133
Model Launch Control Panel . . . 134
Partial Least Squares Options . . . 135
Partial Least Squares Report . . . 136
Model Comparison Summary . . . 136
Cross Validation Report . . . 137
Model Fit Report . . . 142
Model Fit Options . . . 143
Variable Importance Plot . . . 145
VIP vs Coefficients Plots . . . 146
Save Columns . . . 146
Additional Example of Partial Least Squares . . . 148
Statistical Details for the Partial Least Squares Platform . . . 150
Partial Least Squares . . . 150
van der Voet T^2 . . . 151
T^2 Plot . . . 152
Confidence Ellipses for X Score Scatterplot Matrix . . . 153
Standard Error of Prediction and Confidence Limits . . . 153
Standardized Scores and Loadings . . . 154
PLS Discriminant Analysis (PLS-DA) . . . 155

Overview of the Partial Least Squares Platform

In contrast to ordinary least squares, PLS can be used when the predictors outnumber the observations. PLS is used widely in modeling high-dimensional data in areas such as spectroscopy, chemometrics, genomics, psychology, education, economics, political science, and environmental science.

The PLS approach to model fitting is particularly useful when there are more explanatory variables than observations or when the explanatory variables are highly correlated. You can use PLS to fit a single model to several responses simultaneously. See Garthwaite (1994), Wold (1994), Wold et al. (2001), Eriksson et al. (2006), and Cox and Gaudard (2013).

Two model fitting algorithms are available: nonlinear iterative partial least squares (NIPALS) and a "statistically inspired modification of PLS" (SIMPLS). For more information about NIPALS, see Wold (1980). For more information about SIMPLS, see De Jong (1993). For a description of both methods, see Boulesteix and Strimmer (2007). The SIMPLS algorithm was developed with the goal of solving a specific optimality problem. For a single response, both methods give the same model. For multiple responses, there are slight differences.

In JMP, the PLS platform is accessible only through Analyze > Multivariate Methods > Partial Least Squares. In JMP Pro, you can also access the Partial Least Squares personality through Analyze > Fit Model.

JMP PRO In JMP Pro, you can do the following:

- Conduct PLS-DA (PLS discriminant analysis) by fitting responses with a nominal modeling type, using the Partial Least Squares personality in Fit Model.
- Fit polynomial, interaction, and categorical effects, using the Partial Least Squares personality in Fit Model.
- Select among several methods for validation and cross validation.
- Impute missing data.
- Obtain bootstrap estimates of the distributions of various statistics. Right-click in the report of interest. For more information about bootstrap estimates, see *Basic Analysis*.

Partial Least Squares uses the van der Voet T^2 test and cross validation to help you choose the optimal number of factors to extract.

- In JMP, the platform uses the leave-one-out method of cross validation. You can also choose not to use validation.
- JMP PRO In JMP Pro, you can choose KFold, Leave-One-Out, or random holdback cross validation, or you can specify a validation column. You can also choose not to use validation.

Example of Partial Least Squares

This example is from spectrometric calibration, which is an area where partial least squares is very effective. Suppose you are researching pollution in the Baltic Sea. You would like to use the spectra of samples of sea water to determine the amounts of three compounds that are present in these samples.

The three compounds of interest are:

- lignin sulfonate (ls), which is pulp industry pollution
- humic acid (ha), which is a natural forest product
- an optical whitener from detergent (dt)

The amounts of these compounds in each of the samples are the responses. The predictors are spectral emission intensities measured at a range of wavelengths (v1–v27).

For the purposes of calibrating the model, samples with known compositions are used. The calibration data consist of 16 samples of known concentrations of lignin sulfonate, humic acid, and detergent. Emission intensities are recorded at 27 equidistant wavelengths. Use the Partial Least Squares platform to build a model for predicting the amount of the compounds from the spectral emission intensities.

1. Select **Help > Sample Data Library** and open Baltic.jmp.

   Note: The data in the Baltic.jmp data table are reported in Umetrics (1995). The original source is Lindberg, Persson, and Wold (1983).

2. Select **Analyze > Multivariate Methods > Partial Least Squares**.
3. Assign ls, ha, and dt to the **Y, Response** role.
4. Assign Intensities, which contains the 27 intensity variables v1 through v27, to the **X, Factor** role.
5. Click **OK**.

   The Partial Least Squares Model Launch control panel appears.
6. Select **Leave-One-Out** as the Validation Method.
7. Click **Go**.

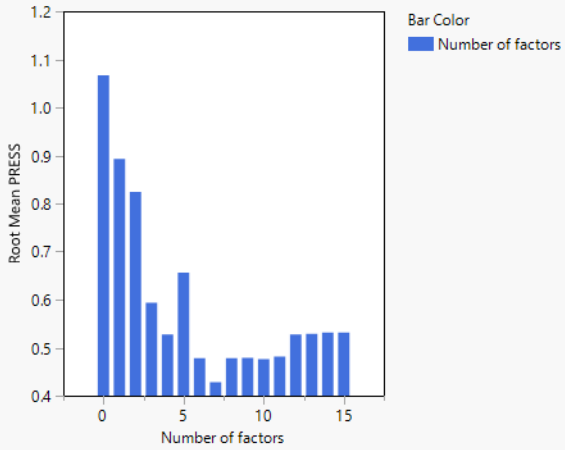
   Since the van der Voet test is a randomization test, your Prob > van der Voet T^2 values may differ slightly.

Figure 6.2 Partial Least Squares Report

Leave-One-Out Cross Validation With Method=NIPALS Using Fast SVD

Number of factors	Root Mean PRESS	van der Voet T^2	Prob > van der Voet T^2	Q^2	Cumulative Q^2	R^2X	Cumulative R^2X	R^2Y	Cumulative R^2Y
0	1.066667	10.070473	0.0030*	-0.137778	-0.137778	0.000000	0.000000	0.000000	0.000000
1	0.892851	11.141942	<.0001*	-0.098077	-0.098077	0.974480	0.974480	0.421421	0.421421
2	0.824551	8.605908	0.0010*	0.773385	0.751159	0.022034	0.996514	0.247724	0.669146
3	0.593922	8.034522	0.0140*	0.774570	0.943904	0.001772	0.998286	0.241683	0.910829
4	0.527742	7.879423	0.0030*	0.656229	0.980716	0.001153	0.999438	0.037043	0.947872
5	0.656013	6.883024	0.0230*	0.356257	0.987586	0.000386	0.999825	0.012502	0.960374
6	0.478479	3.446269	0.3040	0.640049	0.995532	0.000113	0.999938	0.019350	0.979724
7	0.428141	0.000000	1.0000	0.809320	0.999148	0.000017	0.999955	0.010643	0.990367
8	0.478562	7.031041	0.0080*	0.818516	0.999845	0.000011	0.999967	0.004666	0.995032
9	0.478739	3.066627	0.4190	0.810850	0.999971	0.000012	0.999979	0.001452	0.996485
10	0.476024	3.847943	0.2620	0.845631	0.999995	0.000007	0.999986	0.001417	0.997902
11	0.481440	4.822874	0.1170	0.820847	0.999999	0.000004	0.999990	0.001248	0.999150
12	0.527436	3.968884	0.2020	0.810490	1.000000	0.000004	0.999994	0.000637	0.999787
13	0.529249	4.028960	0.2220	0.812180	1.000000	0.000003	0.999997	0.000158	0.999945
14	0.531553	4.408278	0.1790	0.816103	1.000000	0.000003	1.000000	0.000055	1.000000
15	0.531553	4.408278	0.1450	0.816103	1.000000	0.000000	1.000000	0.000000	1.000000

Root Mean PRESS Plot

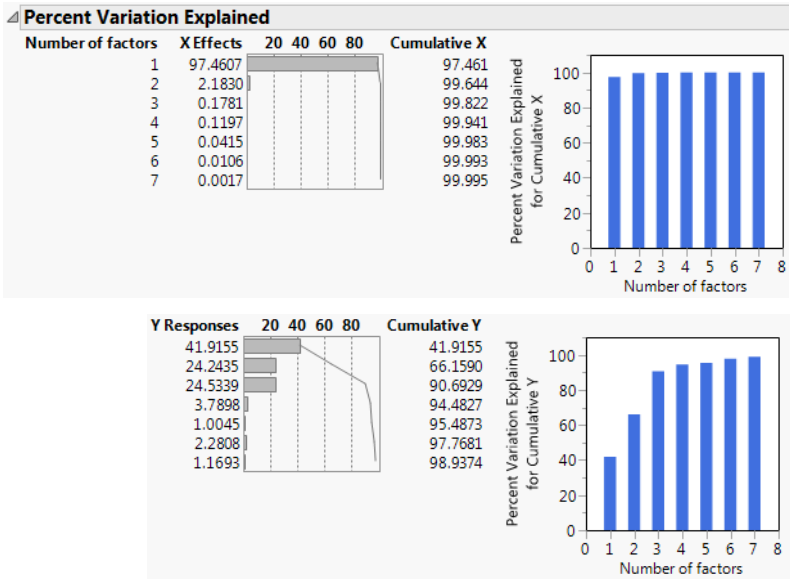


Note: The minimum root mean PRESS is 0.42814 and the minimizing number of factors is 7.

The Root Mean PRESS (predicted residual sum of squares) Plot shows that Root Mean PRESS is minimized when the number of factors is 7. This is stated in the note beneath the Root Mean PRESS Plot. A report called **NIPALS Fit with 7 Factors Using Fast SVD** is produced. A portion of that report is shown in Figure 6.3.

The van der Voet T^2 statistic tests to determine whether a model with a different number of factors differs significantly from the model with the minimum PRESS value. A common practice is to extract the smallest number of factors for which the van der Voet significance level exceeds 0.10 (SAS Institute Inc 2020f; Tobias 1995). If you were to apply this thinking here, you would fit a new model by entering 6 as the **Number of Factors** in the **Model Launch** panel.

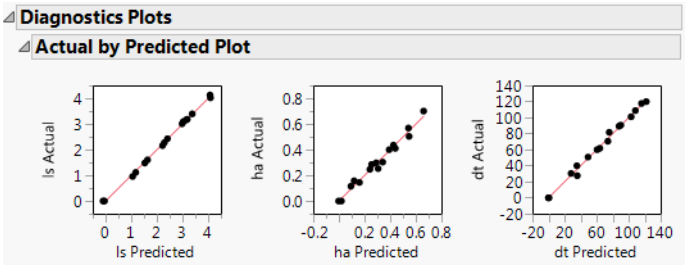
Figure 6.3 Seven Extracted Factors



8. Click the NIPALS Fit with 7 Factors Using Fast SVD red triangle and select **Diagnostics Plots**.

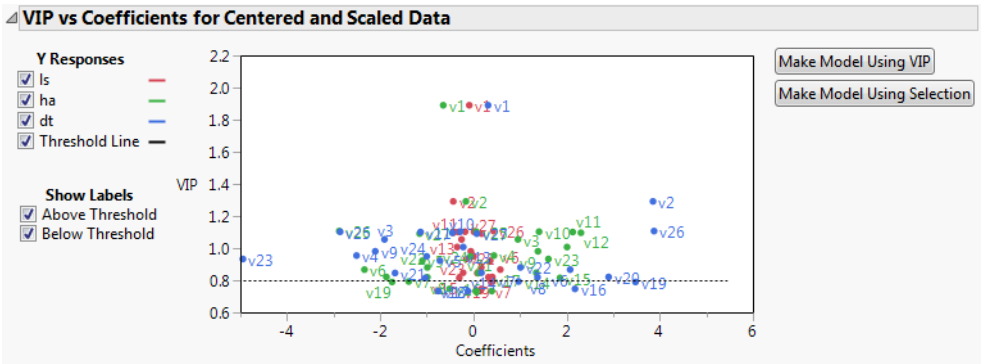
   This gives a report showing actual by predicted plots and three reports showing various residual plots. The Actual by Predicted Plot shows the degree to which predicted compound amounts agree with actual amounts.

Figure 6.4 Diagnostics Plots



9. Click the NIPALS Fit with 7 Factors Using Fast SVD red triangle and select **VIP vs Coefficients Plot**.

Figure 6.5 VIP vs Coefficients Plot



The VIP vs Coefficients plot helps identify variables that are influential relative to the fit for the various responses. For example, v23, v2, and v26 have both VIP values that exceed 0.8 and relatively large coefficients.

Launch the Partial Least Squares Platform

There are two ways to launch the Partial Least Squares platform:

- Select **Analyze > Multivariate Methods > Partial Least Squares**.
- JMP PRO Select **Analyze > Fit Model** and select **Partial Least Squares** from the Personality menu. This approach enables you to do the following:
  - Enter categorical variables as *Y*s or *X*s. Conduct PLS-DA by entering categorical *Y*s.
  - Add interaction and polynomial terms to your model.
  - Use the Standardize X option to construct higher-order terms using centered and scaled columns.
  - Save your model specification script.

Some features on the Fit Model launch window are not applicable for the Partial Least Squares personality:

- Weight, Nest, Attributes, Transform, and No Intercept.

  Tip: You can transform a variable by right-clicking it in the Select Columns box and selecting a Transform option.

- The following Macros: Mixture Response Surface, and Scheffé Cubic.

Figure 6.6 JMP Pro Partial Least Squares Launch Window (Imputation Method EM Selected)

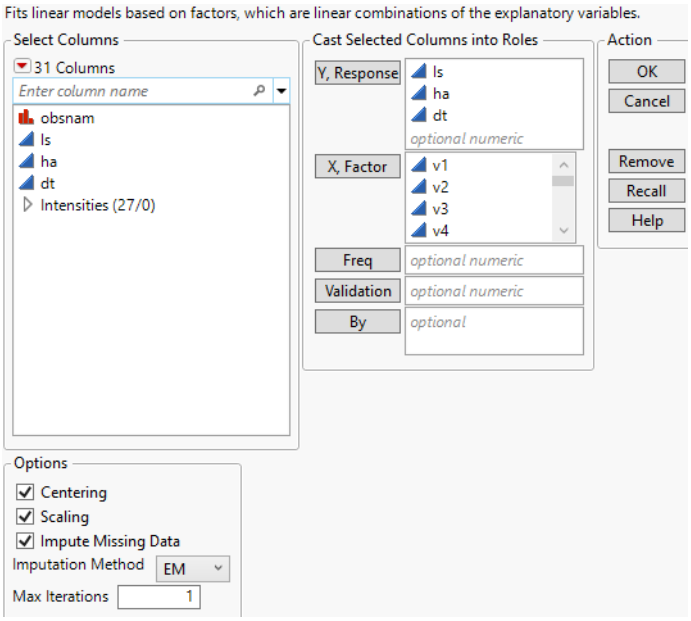
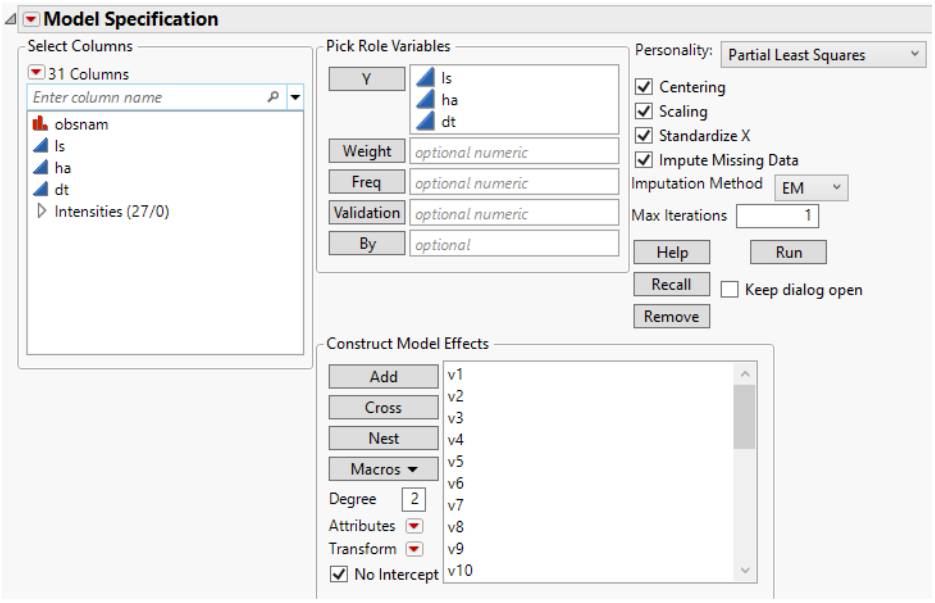


Figure 6.7 JMP Pro Fit Model Partial Least Squares Launch Window



For more information about the Fit Model launch options, see *Fitting Linear Models*.

For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Partial Least Squares launch window contains the following options:

Y, Response Enter numeric response columns. If you enter multiple columns, they are modeled jointly.

JMP PRO In JMP Pro, you can enter nominal response columns in the Fit Model launch window to conduct PLS-DA. See "PLS Discriminant Analysis (PLS-DA)".

X, Factor Enter the predictor columns. The Partial Least Squares launch window allows only numeric predictors.

JMP PRO In JMP Pro, you can enter nominal and ordinal model effects in the Fit Model launch window. Ordinal effects are treated as nominal.

Freq If your data are summarized, enter the column whose values contain counts for each row.

JMP PRO **Validation** A numeric column that defines the validation sets. A validation column must contain only consecutive integer values:

- If the validation column has two levels, the smaller value defines the training set and the larger value defines the validation set.
- If the validation column has three levels, the values define the training, validation, and test sets in order of increasing size.
- If the validation column has more than three levels, then KFold Cross Validation is used. For information about other validation options, see "Validation Method".

The PLS platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see "Validation in JMP Modeling".

Note: If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. For more information about the Make Validation Column utility, *Predictive and Specialized Modeling*.

By Enter a column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Centering Centers all Y variables and model effects by subtracting the mean from each column. See "Centering and Scaling".

Scaling Scales all Y variables and model effects by dividing each column by its standard deviation. See "Centering and Scaling".

JMP PRO **Standardize X** (Available only in the Fit Model launch window.) Centers and scales all columns that are used in the construction of model effects. If this option is not selected,

higher-order effects are constructed using the original data table columns. Then each higher-order effect is centered or scaled, based on the selected Centering and Scaling options. Note that Standardize X does not center or scale Y variables. See "Standardize X".

JMP PRO **Impute Missing Data** Replaces missing data values in Ys or Xs with nonmissing values. Select the appropriate method from the **Imputation Method** list.

If **Impute Missing Data** is not selected, rows that are missing observations on any X variable are excluded from the analysis and no predictions are computed for these rows. Rows with no missing observations on X variables but with missing observations on Y variables are also excluded from the analysis, but predictions are computed.

JMP PRO **Imputation Method** (Appears only when **Impute Missing Data** is selected.) Select from the following imputation methods:

Mean For each model effect or response column, replaces the missing value with the mean of the nonmissing values.

EM Uses an iterative Expectation-Maximization (EM) approach to impute missing values. On the first iteration, the specified model is fit to the data with missing values for an effect or response replaced by their means. Predicted values from the model for Y and the model for X are used to impute the missing values. For subsequent iterations, the missing values are replaced by their predicted values, given the conditional distribution using the current estimates.

For the purpose of imputation, polynomial terms are treated as separate predictors. When a polynomial term is specified, that term is calculated from the original data, or, if Standardize X is checked, from the standardized column values. If a row has a missing value for a column involved in the definition of the polynomial term, then that entry is missing for the polynomial term. Imputation is conducted using polynomial terms defined in this way.

For more information about the EM approach, see Nelson, Taylor, and MacGregor (1996).

JMP PRO **Max Iterations** (Appears only when EM is selected as the Imputation Method.) Enables you to set the maximum number of iterations used by the algorithm. The algorithm terminates if the maximum difference between the current and previous estimates of missing values is bounded by 10^{-8}.

After completing the launch window and clicking **OK**, the Model Launch control panel appears. See "Model Launch Control Panel".

Centering and Scaling

The Centering and Scaling options are selected by default. This means that predictors and responses are centered and scaled to have mean 0 and standard deviation 1. Centering the predictors and the responses places them on an equal footing relative to their variation. Without centering, both the variable's mean and its variation around that mean are involved in constructing successive factors. To illustrate, suppose that Time and Temp are two of the predictors. Scaling them indicates that a change of one standard deviation in Time is approximately equivalent to a change of one standard deviation in Temp.

Standardize X

JMP PRO When the Partial Least Square personality is selected in the Fit Model window, the Standardize X option is selected by default. This ensures that all columns entered as model effects and that all columns that are involved in an interaction or polynomial term are standardized.

Suppose that you have two columns, X1 and X2, and you enter the interaction term X1*X2 as a model effect in the Fit Model window. When the Standardize X option is selected, both X1 and X2 are centered and scaled before forming the interaction term. The interaction term that is formed is calculated as follows:

$$\left(\frac{X1 - mean(X1)}{std(X1)}\right) \times \left(\frac{X2 - mean(X2)}{std(X2)}\right)$$

All model effects are then centered or scaled, in accordance with your selections of the **Centering** and **Scaling** options, prior to inclusion in the model.

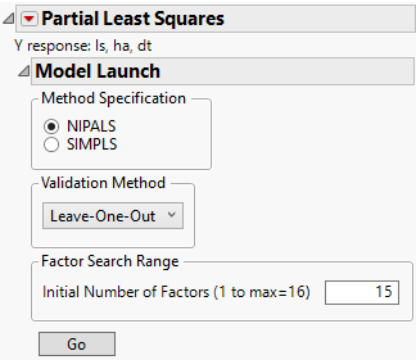
If the Standardize X option is not selected, and **Centering** and **Scaling** are both selected, then the term that is entered into the model is calculated as follows:

$$\frac{X1 \times X2 - mean(X1 \times X2)}{std(X1 \times X2)}$$

Model Launch Control Panel

After you click **OK** in the platform launch window (or **Run** in the Fit Model window), the Model Launch control panel appears.

Figure 6.8 Partial Least Squares Model Launch Control Panel



Note: The Validation Method portion of the Model Launch control panel appears differently in JMP Pro.

The Model Launch control panel contains the following selections:

Method Specification Select the type of model fitting algorithm. There are two algorithm choices: **NIPALS** and **SIMPLS**. The two methods produce the same coefficient estimates when there is only one response variable. See "Statistical Details for the Partial Least Squares Platform" for more information about differences between the two algorithms.

Validation Method Select the validation method. Validation is used to determine the optimum number of factors to extract. For JMP Pro, if a validation column is specified on the platform launch window, these options do not appear.

JMP PRO **Holdback** Randomly selects the specified proportion of the data for a validation set, and uses the other portion of the data to fit the model. The random selection is based on stratified sampling across the model factors to attempt to create training and validation sets that are more balanced than ones based on simple random sampling.

JMP PRO **KFold** Partitions the data into *K* subsets, or *folds*. In turn, each fold is used to validate the model that is fit to the rest of the data, fitting a total of *K* models. This method is best for small data sets because it makes efficient use of limited amounts of data.

Leave-One-Out Performs leave-one-out cross validation.

None Does not use validation to choose the number of factors to extract. The number of factors is specified in the Factor Search Range.

Factor Search Range Specifies how many latent factors to extract if not using validation. If validation is being used, this is the maximum number of factors the platform attempts to fit before choosing the optimum number of factors. The maximum number of factors is the minimum of the number of non-missing rows and the number of factors. Then, the initial number of factors is the minimum of 15 and the maximum number of factors.

Factor Specification Appears once you click **Go** to fit an initial model. Specify a number of factors to be used in fitting a new model.

Go Launches the model fit with the given specifications.

Partial Least Squares Options

The Partial Least Squares red triangle menu contains some options that are useful before clicking Go in the Model Launch Control Panel.

JMP PRO **Set Random Seed** Sets the seed for the randomization process used for KFold and Holdback validation. This is useful if you want to reproduce an analysis. Set the seed to a positive value, save the script, and the seed is automatically saved in the script. Running the script always produces the same cross validation analysis. This option does not appear when Validation Method is set to None, or when a validation column is used.

JMP PRO **SVD** Shows a submenu that enables you to choose between a fast or classical implementation of the SVD algorithm in the model fitting algorithm. The default is Fast SVD.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Partial Least Squares Report

The first time you click **Go** in the Model Launch control panel (Figure 6.8), the Validation Method panel is removed from the Model Launch window. If you specified a Validation column or if you selected Holdback in the Validation Method panel, all model fits in the report are based on the training data. Otherwise, all model fits are based on the entire data set.

If you used validation, three reports appear:

- Model Comparison Summary
- Cross Validation Report
- NIPALS (or SIMPLS) Fit with <N> Factors

If you selected **None** as the CV method, two reports appear:

- Model Comparison Summary
- NIPALS (or SIMPLS) Fit with <N> Factors

To fit additional models, specify the desired numbers of factors in the Model Launch panel.

Model Comparison Summary

The Model Comparison Summary shows summary results for each fitted model.

Figure 6.9 Model Comparison Summary

Model Comparison Summary

Method	SVD	Number of rows	Number of factors	Percent Variation Explained for Cumulative X	Percent Variation Explained for Cumulative Y	Number of VIP > 0.8
NIPALS	Fast	16	7	99.995152	98.937438	22
NIPALS	Fast	16	6	99.993471	97.768092	22

The report includes the following summary information:

Method Shows the analysis method that you specified in the Model Launch control panel.

SVD Shows the implementation of the SVD algorithm that you specified.

Number of rows Shows the number of observations used in the training set.

Number of factors Shows the number of extracted factors.

Percent Variation Explained for Cumulative X Shows the percent of variation in X that is explained by the model.

Percent Variation Explained for Cumulative Y Shows the percent of variation in Y that is explained by the model.

Number of VIP>0.8 Shows the number of model effects with VIP (variable importance for projection) values greater than 0.8. The VIP score is a measure of a variable's importance relative to modeling both X and Y (Wold 1994; Eriksson et al. 2006).

Cross Validation Report

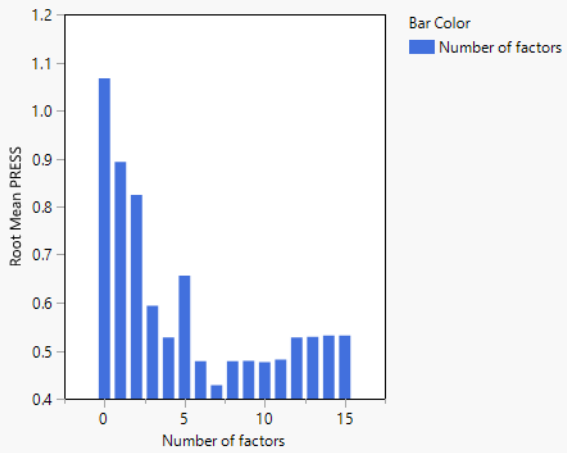
This report appears only when a form of cross validation is selected as a Validation Method in the Model Launch control panel. The report title is dynamically named <Cross Validation Method> With <Method = Method Specification>, depending on the cross validation and method options selected in the control panel. It shows summary statistics for models fit, using from 0 to the maximum number of extracted factors, as specified in the Model Launch control panel. The report also provides a plot of Root Mean PRESS values. See "Root Mean PRESS Plot". An optimum number of factors is identified using the minimum Root Mean PRESS statistic.

Figure 6.10 Cross Validation Report

Leave-One-Out Cross Validation With Method=NIPALS Using Fast SVD

Number of factors	Root Mean PRESS	van der Voet T^2	Prob > van der Voet T^2	Q^2	Cumulative Q^2	R^2X	Cumulative R^2X	R^2Y	Cumulative R^2Y
0	1.066667	10.070473	0.0030*	-0.137778	-0.137778	0.000000	0.000000	0.000000	0.000000
1	0.892851	11.141942	<.0001*	-0.098077	-0.098077	0.974480	0.974480	0.421421	0.421421
2	0.824551	8.605908	0.0010*	0.773385	0.751159	0.022034	0.996514	0.247724	0.669146
3	0.593922	8.034522	0.0140*	0.774570	0.943904	0.001772	0.998286	0.241683	0.910829
4	0.527742	7.879423	0.0030*	0.656229	0.980716	0.001153	0.999438	0.037043	0.947872
5	0.656013	6.883024	0.0230*	0.356257	0.987586	0.000386	0.999825	0.012502	0.960374
6	0.478479	3.446269	0.3040	0.640049	0.995532	0.000113	0.999938	0.019350	0.979724
7	0.428141	0.000000	1.0000	0.809320	0.999148	0.000017	0.999955	0.010643	0.990367
8	0.478562	7.031041	0.0080*	0.818516	0.999845	0.000011	0.999967	0.004666	0.995032
9	0.478739	3.066627	0.4190	0.810850	0.999971	0.000012	0.999979	0.001452	0.996485
10	0.476024	3.847943	0.2620	0.845631	0.999995	0.000007	0.999986	0.001417	0.997902
11	0.481440	4.822874	0.1170	0.820847	0.999999	0.000004	0.999990	0.001248	0.999150
12	0.527436	3.968884	0.2020	0.810490	1.000000	0.000004	0.999994	0.000637	0.999787
13	0.529249	4.028960	0.2220	0.812180	1.000000	0.000003	0.999997	0.000158	0.999945
14	0.531553	4.408278	0.1790	0.816103	1.000000	0.000003	1.000000	0.000055	1.000000
15	0.531553	4.408278	0.1450	0.816103	1.000000	0.000000	1.000000	0.000000	1.000000

Root Mean PRESS Plot



Note: The minimum root mean PRESS is 0.42814 and the minimizing number of factors is 7.

JMP PRO When the **Standardize X** option is selected, the standardization is applied once to the entire data table. It is not reapplied to the individual training sets. However, when any combination of the **Centering** or **Scaling** options are selected, this combination of selections is applied to each cross validation training set. Cross validation proceeds by using the training sets, which are individually centered and scaled if these options are selected.

The following statistics are shown in the report. If any form of validation or cross validation is used, the reported results are summaries of the training set statistics.

Number of Factors The number of factors used in fitting the model.

Root Mean PRESS The square root of the average of the PRESS values across all responses. See "Root Mean PRESS".

van der Voet T^2 The test statistic for the van der Voet test, which tests whether models with different numbers of extracted factors differ significantly from the optimum model. The null hypothesis for each van der Voet T^2 test states that the model based on the corresponding number of factors does not differ from the optimum model. The alternative hypothesis is that the model does differ from the optimum model. See "van der Voet T^2".

Prob > van der Voet T^2 The *p*-value for the van der Voet T^2 test. See "van der Voet T^2".

Q^2 Dimensionless measure of predictive ability defined by subtracting the ratio of the PRESS value divided by the total sum of squares for Y from one:

$$1 - PRESS/SSY$$

See "Calculation of Q^2".

Cumulative Q^2 Indicator of the predictive ability of models with the given number of factors or fewer. For a given number of factors, *f*, Cumulative Q^2 is defined as follows:

$$1 - \prod_{i=1}^{f} (Press_i/SYY_i)$$

Here $PRESS_i$ and SSY_i correspond to their values for *i* factors.

R^2X Percent of X variation explained by the specified factor. A component with a large R^2X explains a large amount of the variation in the X variables. See "Calculation of R^2X and R^2Y When Validation Is Used".

Cumulative R^2X Percent of X variation explained by the model with the given number of factors. This is the sum of the R^2X values for $i = 1$ to the given number of factors.

R^2Y Percent of Y variation explained by the specified factor. A component with a large R^2Y explains a large amount of the variation in the Y variables. See "Calculation of R^2X and R^2Y When Validation Is Used".

Cumulative R^2Y Percent of Y variation explained by the model with the given number of factors. This is the Sum of the R^2Y values for $i = 1$ to the given number of factors.

Interpretation of Q^2 and Cumulative R^2Y

The statistics Q^2 and Cumulative R^2Y both measure the predictive ability of the model, but in different ways.

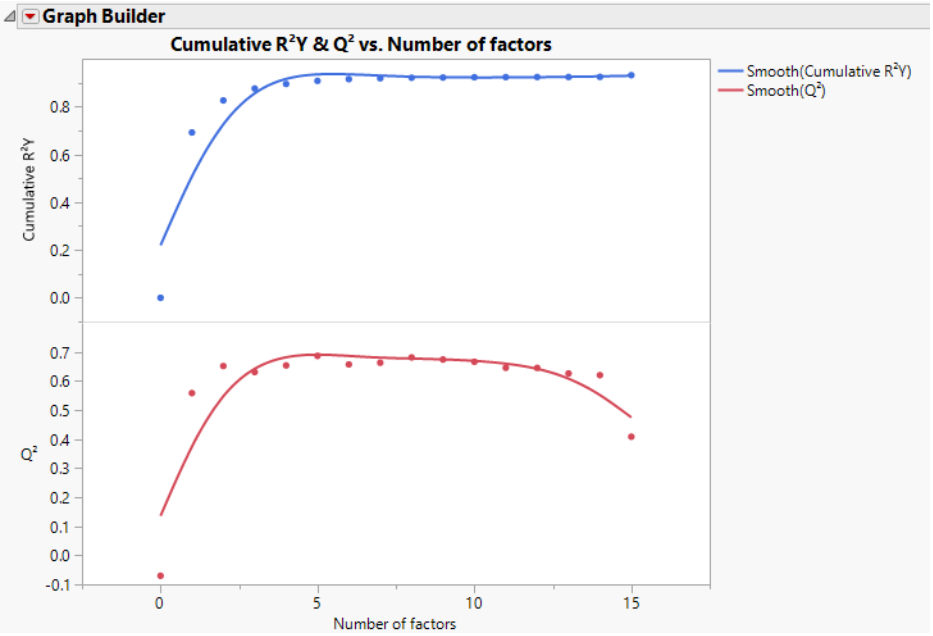
- Cumulative R^2Y increases as the number of factors increases. This is because, as factors are added to the model, more variation is explained.
- Q^2 tends to increase and then decrease, or at least discontinue increasing, as the number of factors increases. This is because, as more factors are added, the model becomes tuned to

the training set and does not generalize well to new data, causing the PRESS statistic to decrease.

Analysis of Q^2 and Cumulative R^2Y provides an alternative to using the van der Voet test for determining how many factors to include in your model. Select a number of factors for which Q^2 is large and has not started decreasing. You also want Cumulative R^2Y to be large.

Figure 6.11 shows plots of Cumulative R^2Y and Q^2 against the number of factors for the Penta.jmp data table, using Leave-One-Out as the validation method. Cumulative R^2Y increases and levels off for about four factors. The statistic Q^2 is largest for two factors and then begins to level off. The plot suggests that a model with two factors explains a large portion of the variation in Y without overfitting the data.

Figure 6.11 Cumulative R^2Y and Q^2 for Penta.jmp



Root Mean PRESS Plot

This bar chart shows the number of factors along the horizontal axis and the Root Mean PRESS values on the vertical axis. It is equivalent to the horizontal bar chart that appears to the right of the Root Mean PRESS column in the Cross Validation report (Figure 6.10).

Root Mean PRESS

For a specified number of factors, *a*, the Root Mean PRESS is calculated using the following steps:

1. Fit a model with *a* factors to each training set.
2. Apply the resulting prediction formula to the observations in the validation set.
3. For each Y:
   - For each validation set, compute the squared difference between each observed validation set value and its predicted value (the squared prediction error).
   - For each validation set, average these squared differences and divide the result by a variance estimate for the response. For the KFold and Leave-One-Out validation methods, divide by the variance of the entire response column. For Holdback validation, divide by the variance of the response values in the training set.
   - Sum these means and, in the case of more than one validation set, divide their sum by the number of validation sets minus one. This is the PRESS statistic for the given Y.
4. Root Mean PRESS for *a* factors is the square root of the average of the PRESS values across all responses.
5. The PRESS statistic for multiple Ys is obtained by averaging the PRESS statistic, obtained in step 3, across all responses.

Calculation of Q^2

The statistic Q^2 is defined as 1 - *PRESS* / *SSY*. The *PRESS* statistic is the predicted error sum of squares averaged across all responses for the model developed based on the training data, but evaluated on the validation set. The value of *SSY* is the sum of squares for Y averaged across all responses and based on the observations in the validation set.

The statistic Q^2 in the Cross Validation report is computed in the following ways, depending on the selected Validation Method:

Leave-One-Out Q^2 is the values 1 - mean(*PRESS*) / mean(*SSY*). *PRESS* for each *Y* is calculated from the models constructed leaving out one observation at a time. *SSY* is calculated from all of the values in each *Y* column.

KFold Q^2 is the average of the values 1 - *PRESS* / *SSY* computed for the validation sets based on the *K* models constructed by leaving out each of the *K* folds.

Holdback or Validation Set Q^2 is the value of 1 - *PRESS* / *SSY* computed for the validation set based on the model constructed using the single set of training data.

Calculation of R^2X and R^2Y When Validation Is Used

The statistics R^2X and R^2Y in the Cross Validation report are computed in the following ways, depending on the selected Validation Method:

Note: For all of these computations, R^2Y is calculated analogously.

Leave-One-Out R^2X is the average of the Percent Variation Explained for X Effects for the models constructed by leaving out one observation at a time.

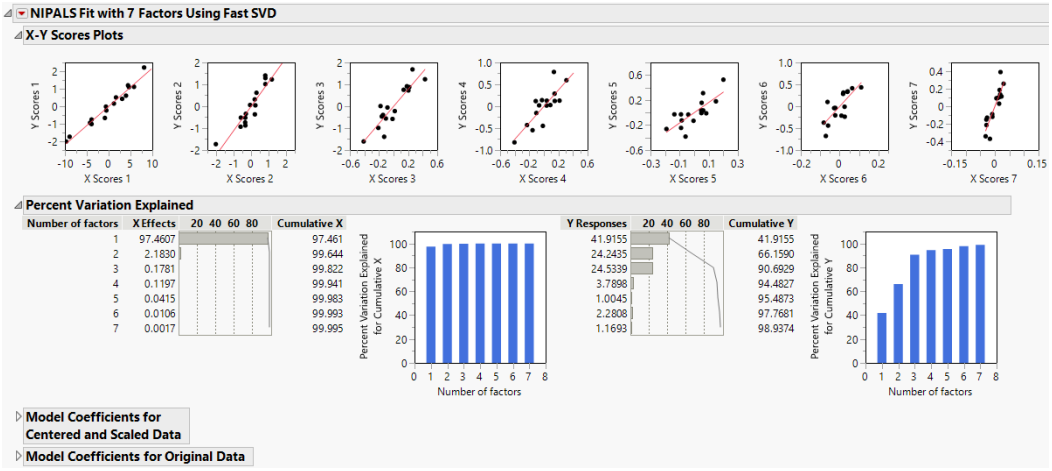
KFold R^2X is the average of the Percent Variation Explained for X Effects for the K models constructed by leaving out each fold.

Holdback or Validation Set R^2X is the Percent Variation Explained for X Effects for the model constructed using the training data.

Model Fit Report

The Model Fit Report shows detailed results for each fitted model. The fit uses either the optimum number of factors based on cross validation, or the specified number of factors if no cross validation methods are specified. The report title indicates whether NIPALS or SIMPLS was used, whether Fast SVD or Classical SVD was used, and gives the number of extracted factors.

Figure 6.12 Model Fit Report



The Model Fit report includes the following summary information:

X-Y Scores Plots Scatterplots of the X and Y scores for each extracted factor.

Percent Variation Explained Shows the percent variation and cumulative percent variation explained for both X and Y. Results are given for each extracted factor.

Model Coefficients for Centered and Scaled Data For each Y, shows the coefficients of the Xs for the model based on the centered and scaled data.

Model Fit Options

The Model Fit red triangle menu contains the following options:

Percent Variation Plots Adds two plots entitled Percent Variation Explained for X Effects and Percent Variation Explained for Y Effects. These show stacked bar charts representing the percent variation explained by each extracted factor for the *X*s and *Y*s.

Variable Importance Plot Plots the VIP values for each X variable. VIP scores appear in the Variable Importance Table. See "Variable Importance Plot".

VIP vs Coefficients Plots Plots the VIP statistics against the model coefficients. You can show only those points corresponding to your selected *Y*s. Additional labeling options are provided. There are plots for both the centered and scaled data and the original data. See "VIP vs Coefficients Plots".

Set VIP Threshold Sets the threshold level for the Variable Importance Plot, Variance Importance Table, and the VIP vs Coefficients Plots.

Coefficient Plots Plots the model coefficients for each response across the X variables. You can show only those points corresponding to your selected *Y*s. There are plots for both the centered and scaled data and the original data.

Loading Plots Plots X and Y loadings for each extracted factor. There are separate plots for the *X*s and *Y*s.

Loading Scatterplot Matrices Shows scatterplot matrices of the X loadings and the Y loadings.

Correlation Loading Plot Shows either a single scatterplot or a scatterplot matrix of the X and Y loadings overlaid on the same plot. When you select the option, you specify how many factors you want to plot.

- If you specify two factors, a single correlation loading scatterplot appears. Select the two factors that define the axes beneath the plot. Click the right arrow button to successively display each combination of factors on the plot.

– If you specify more than two factors, a scatterplot matrix appears with a cell for pair of factors up to the number that you selected.

In both cases, use check boxes to control labeling.

X-Y Score Plots Includes the following options:

Fit Line Shows or hides a fitted line through the points on the X-Y Scores Plots.

Show Confidence Band Shows or hides 95% confidence bands for the fitted lines on the X-Y Scores Plots.

Score Scatterplot Matrices Shows a scatterplot matrix of the X scores and a scatterplot matrix of the Y scores. Each X score scatterplot displays a 95% confidence ellipse, which can be used for outlier detection. For statistical details about the confidence ellipses, see "Confidence Ellipses for X Score Scatterplot Matrix".

Distance Plots Shows plots of the following:

– the distance from each observation to the X model
– the distance from each observation to the Y model
– a scatterplot of distances to both the X and Y models

In a good model, both X and Y distances are small, so the points are close to the origin (0,0). Use the plots to look for outliers relative to either X or Y. If a group of points clusters together, then they might have a common feature and could be analyzed separately. When a validation set or a validation and test set are in use, separate reports are provided for these sets and for the training set.

T Square Plot Shows a plot of T^2 statistics for each observation, along with a control limit. An observation's T^2 statistic is calculated based on that observation's scores on the extracted factors. For more information about the computation of T^2 and the control limit, see "T^2 Plot".

Diagnostics Plots Shows diagnostic plots for assessing the model fit. Four plot types are available: Actual by Predicted Plot, Residual by Predicted Plot, Residual by Row Plot, and a Residual Normal Quantile Plot. Plots are provided for each response. When a validation set or a validation and test set are in use, separate reports are provided for these sets and for the training set.

Profiler Shows a profiler for each Y variable.

Spectral Profiler Shows a single profiler where all of the response variables appear in the first cell of the plot. This profiler is useful for visualizing the effect of changes in the X variables on the Y variables simultaneously.

Save Columns Includes options for saving various formulas and results. See "Save Columns".

Remove Fit Removes the model report from the main platform report.

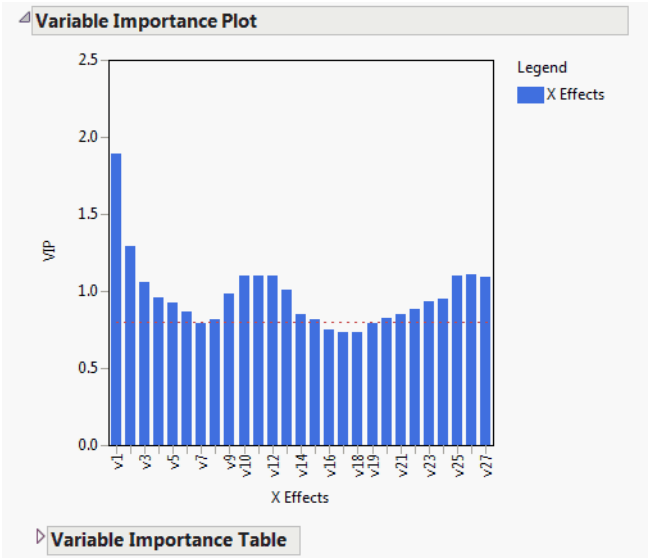
Make Model Using VIP Opens and populates a launch window with the appropriate responses entered as *Y*s and the variables whose VIPs exceed the specified threshold entered as *X*s. Performs the same function as the button in the VIP vs Coefficients for Centered and Scaled Data report. See "VIP vs Coefficients Plots".

Model Driven Multivariate Control Chart for Saved X Scores Saves the formulas for each X Score and launches the Model Driven Multivariate Control Chart (MDMCC) launch window. In the MDMCC launch window, the score formulas are assigned as the process columns. You can add or remove processes, add a time ID, or set where historical data ends before clicking OK. See *Quality and Process Methods*.

Variable Importance Plot

The Variable Importance Plot graphs the VIP values for each X variable. The Variable Importance Table shows the VIP scores. A VIP score is a measure of a variable's importance in modeling both X and Y. If a variable has a small coefficient and a small VIP, then it is a candidate for deletion from the model (Wold 1994). A value of 0.8 is generally considered to be a small VIP (Eriksson et al. 2006) and a red dashed line is drawn on the plot at 0.8.

Figure 6.13 Variable Importance Plot



VIP vs Coefficients Plots

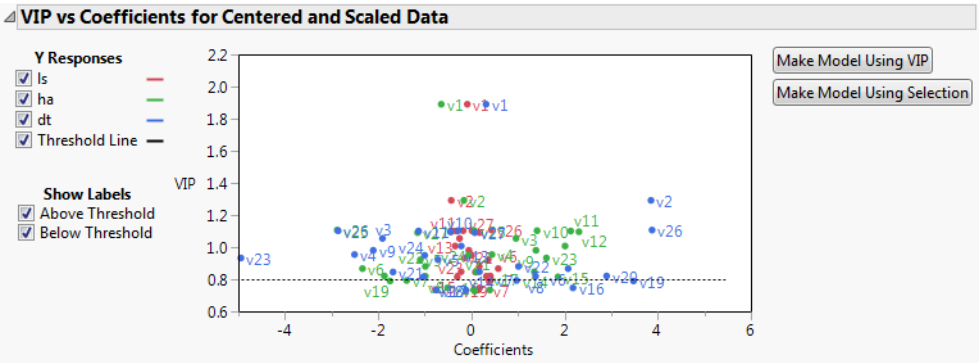
Two options to the right of the plot facilitate variable reduction and model building:

Make Model Using VIP Opens and populates a launch window with the appropriate responses entered as *Y*s and the variables whose VIPs exceed the specified threshold entered as *X*s.

Make Model Using Selection Enables you to select *X*s directly in the plot and then enters the *Y*s and only the selected *X*s into a launch window.

To use another platform based on your current column selection, open the desired platform. Notice in the launch window that the selections are retained. Click the role button and the selected columns are populated.

Figure 6.14 VIP vs Coefficients Plot for Centered and Scaled Data



Save Columns

Save Prediction Formula For each Y variable, saves a column to the data table called Pred Formula <response> that contains the prediction formula.

Save Prediction as X Score Formula For each Y variable, saves a column to the data table called Pred Formula <response> that contains the prediction formula in terms of the X scores.

Save Standard Errors of Prediction Formula For each Y variable, saves a column to the data table called PredSE <response> that contains the standard error of the predicted mean. See "Standard Error of Prediction and Confidence Limits".

Save Mean Confidence Limit Formula For each Y variable, saves two columns to the data table called Lower 95% Mean <response> and Upper 95% Mean <response>. These columns contain 95% confidence limits for the response mean. See "Standard Error of Prediction and Confidence Limits".

Save Indiv Confidence Limit Formula For each Y variable, saves two columns to the data table called Lower 95% Indiv <response> and Upper 95% Indiv <response>. These columns contain 95% prediction limits for individual values. See "Standard Error of Prediction and Confidence Limits".

Save Score Formula Saves two sets of columns to the data table:

- Columns called X Score <N> Formula containing the formulas for each X Score.
- Columns called Y Score <N> Formula containing the formulas for each Y Score

See "Partial Least Squares".

Save Y Predicted Values Saves the predicted values for the Y variables to columns in the data table.

Save Y Residuals Saves the residual values for the Y variables to columns in the data table.

Save X Predicted Values Saves the predicted values for the X variables to columns in the data table.

Save X Residuals Saves the residual values for the X variables to columns in the data table.

Save Percent Variation Explained For X Effects Saves the percent variation explained for each X variable across all extracted factors to a new data table.

Save Percent Variation Explained For Y Responses Saves the percent variation explained for each Y variable across all extracted factors to a new data table.

Save Scores Saves the X and Y scores for each extracted factor to the data table.

Save Loadings Saves the X and Y loadings to new data tables.

Save Standardized Scores Saves the X and Y standardized scores used in constructing the Correlation Loading Plot to the data table. For the formulas, see "Standardized Scores and Loadings".

Save Standardized Loadings Saves the X and Y standardized loadings used in constructing the Correlation Loading Plot to new data tables. For the formulas, see "Standardized Scores and Loadings".

Save T Square Saves a column to the data table that contains the T^2 formulas. The values in this column are also used in the T Square Plot.

Save Distance Saves the Distance to X Model (DModX) and Distance to Y Model (DModY) values to the data table. These are the values used in the Distance Plots.

Save X Weights Saves the weights for each X variable across all extracted factors to a new data table.

JMP PRO **Save Validation** Saves a new column to the data table describing how each observation was used in validation. For Holdback validation, the column identifies if a row was used for training or validation. For KFold validation, the column identifies the number of the subgroup to which the row was assigned.

JMP PRO **Save Imputation** If Impute Missing Data is selected, opens a new data table that contains the data table columns specified as X and Y, with missing values replaced by their imputed values. Columns for polynomial terms are not shown. If a Validation column is specified, the validation column is also included.

JMP PRO **Publish Prediction Formula** Creates prediction formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

JMP PRO **Publish Score Formula** Creates X and Y score formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

JMP PRO Additional Example of Partial Least Squares

In this example, you will use the Partial Least Squares personality in Fit Model to perform an analysis for a categorical response and a large number of variables. Using Partial Least Squares through Fit Model also enables you to use K Fold Crossvalidation. The example data set consists of serum samples collected from 165 men. 84 of the men have prostate cancer and 81 of the men do not. The goal of the analysis is to be able to determine which men have prostate cancer based on the serum samples.

1. Select **Help > Sample Data Library** and open Prostate Cancer.jmp.
2. Select **Analyze > Fit Model**.
3. Select Status and click **Y**.
4. Select the Proteins column group and click **Add**.
5. From the Personality list, select **Partial Least Squares**.
6. Click **Run**.
7. (Optional) Click the Partial Least Squares red triangle and select **Set Random Seed**.
8. (Optional) Next to Specify random seed, enter 1234.

   Specifying a random seed enables you to reproduce the results shown in this example.
9. (Optional). Click **OK**.
10. In the Model Launch, click **Go**.

Figure 6.15 Partial Least Squares Report

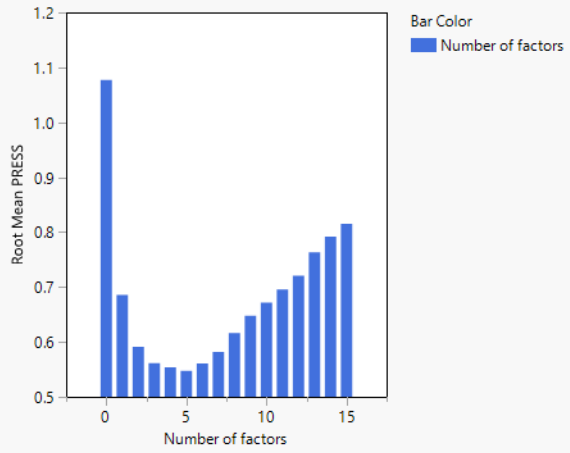
Model Comparison Summary

Method	SVD	Number of rows	Number of factors	Percent Variation Explained for Cumulative X	Percent Variation Explained for Cumulative Y	Number of VIP > 0.8
NIPALS	Fast	165	5	52.886468	85.671251	626

KFold Cross Validation with K=7 and Method=NIPALS Using Fast SVD

Number of factors	Root Mean PRESS	van der Voet T^2	Prob > van der Voet T^2	Q^2	Cumulative Q^2	R^2X	Cumulative R^2X	R^2Y	Cumulative R^2Y
0	1.076926	130.46053	<.0001*	-0.000625	-0.000625	0.000000	0.000000	0.000000	0.000000
1	0.685794	16.150335	0.0010*	0.593265	0.593265	0.342651	0.342651	0.608907	0.608907
2	0.591463	2.927629	0.0790	0.696945	0.876737	0.077029	0.419680	0.132190	0.741097
3	0.561730	0.480087	0.4740	0.726683	0.966310	0.039013	0.458693	0.066929	0.808026
4	0.553949	0.278455	0.5910	0.734125	0.991043	0.043112	0.501805	0.034490	0.842515
5	0.547744	0.000000	1.0000	0.740188	0.997673	0.055223	0.557027	0.017535	0.860051
6	0.560870	1.357190	0.2350	0.728045	0.999367	0.048092	0.605120	0.015012	0.875063
7	0.581726	3.490817	0.0600	0.707162	0.999815	0.034371	0.639491	0.012864	0.887927
8	0.616081	7.673681	0.0010*	0.671501	0.999939	0.023573	0.663064	0.013162	0.901089
9	0.647871	12.869719	<.0001*	0.636867	0.999978	0.020427	0.683491	0.011323	0.912412
10	0.671625	14.823639	<.0001*	0.610463	0.999991	0.015562	0.699053	0.010699	0.923112
11	0.695867	17.385858	<.0001*	0.581816	0.999996	0.019040	0.718092	0.007945	0.931057
12	0.720865	19.399541	<.0001*	0.551351	0.999998	0.021476	0.739568	0.007050	0.938107
13	0.763351	22.554517	<.0001*	0.496903	0.999999	0.011706	0.751274	0.008857	0.946963
14	0.791689	26.405173	<.0001*	0.458986	1.000000	0.011510	0.762784	0.006236	0.953199
15	0.815424	28.291824	<.0001*	0.426187	1.000000	0.011048	0.773832	0.005424	0.958623

Root Mean PRESS Plot



Note: The minimum root mean PRESS is 0.54774 and the minimizing number of factors is 5.

The Root Mean PRESS Plot shows that Root Mean PRESS is minimized when the number of factors is 5. This is stated in the note beneath the Root Mean PRESS Plot. A report called **NIPALS Fit with 5 Factors Using Fast SVD** is produced.

Statistical Details for the Partial Least Squares Platform

This section provides details about some of the methods used in the Partial Least Squares platform. See Hoskuldsson (1988), Garthwaite (1994), or Cox and Gaudard (2013).

- "Partial Least Squares"
- "van der Voet T^2"
- "T^2 Plot"
- "Confidence Ellipses for X Score Scatterplot Matrix"
- "Standard Error of Prediction and Confidence Limits"
- "Standardized Scores and Loadings"
- "PLS Discriminant Analysis (PLS-DA)"

Partial Least Squares

Partial least squares fits linear models based on linear combinations, called factors, of the explanatory variables (*X*s). These factors are obtained in a way that attempts to maximize the covariance between the *X*s and the response or responses (*Y*s). In this way, PLS exploits the correlations between the *X*s and the *Y*s to reveal underlying latent structures. The factors address the combined goals of explaining response variation and predictor variation. Partial least squares is particularly useful when you have more X variables than observations or when the X variables are highly correlated.

NIPALS

The NIPALS method works by extracting one factor at a time. Let $\mathbf{X} = \mathbf{X}_0$ be the centered and scaled matrix of predictors and $\mathbf{Y} = \mathbf{Y}_0$ the centered and scaled matrix of response values. The PLS method starts with a linear combination $\mathbf{t} = \mathbf{X}_0\mathbf{w}$ of the predictors, where $\mathbf{t}$ is called a *score vector* and $\mathbf{w}$ is its associated *weight vector*. The PLS method predicts both $\mathbf{X}_0$ and $\mathbf{Y}_0$ by regression on $\mathbf{t}$:

$$\hat{\mathbf{X}}_0 = \mathbf{t}\mathbf{p}', \text{ where } \mathbf{p}' = (\mathbf{t}'\mathbf{t})^{-1}\mathbf{t}'\mathbf{X}_0$$

$$\hat{\mathbf{Y}}_0 = \mathbf{t}\mathbf{c}', \text{ where } \mathbf{c}' = (\mathbf{t}'\mathbf{t})^{-1}\mathbf{t}'\mathbf{Y}_0$$

The vectors $\mathbf{p}$ and $\mathbf{c}$ are called the *X*- and *Y*-*loadings*, respectively.

The specific linear combination $\mathbf{t} = \mathbf{X}_0\mathbf{w}$ is the one that has maximum covariance $\mathbf{t}'\mathbf{u}$ with some response linear combination $\mathbf{u} = \mathbf{Y}_0\mathbf{q}$. Another characterization is that the X- and Y-weights, $\mathbf{w}$ and $\mathbf{q}$, are proportional to the first left and right singular vectors of the covariance matrix $\mathbf{X}_0'\mathbf{Y}_0$. Or, equivalently, the first eigenvectors of $\mathbf{X}_0'\mathbf{Y}_0\mathbf{Y}_0'\mathbf{X}_0$ and $\mathbf{Y}_0'\mathbf{X}_0\mathbf{X}_0'\mathbf{Y}_0$ respectively.

This accounts for how the first PLS factor is extracted. The second factor is extracted in the same way by replacing $\mathbf{X}_0$ and $\mathbf{Y}_0$ with the X- and Y-residuals from the first factor:

$$\mathbf{X}_1 = \mathbf{X}_0 - \hat{\mathbf{X}}_0$$

$$\mathbf{Y}_1 = \mathbf{Y}_0 - \hat{\mathbf{Y}}_0$$

These residuals are also called the *deflated* X and Y blocks. The process of extracting a score vector and deflating the data matrices is repeated for as many extracted factors as desired.

SIMPLS

The SIMPLS algorithm was developed to optimize a statistical criterion: it finds score vectors that maximize the covariance between linear combinations of Xs and Ys, subject to the requirement that the X-scores are orthogonal. Unlike NIPALS, where the matrices $\mathbf{X}_0$ and $\mathbf{Y}_0$ are deflated, SIMPLS deflates the cross-product matrix, $\mathbf{X}_0'\mathbf{Y}_0$.

In the case of a single Y variable, these two algorithms are equivalent. However, for multivariate Y, the models differ. SIMPLS was suggested by De Jong (1993).

van der Voet T^2

The van der Voet T^2 test helps determine whether a model with a specified number of extracted factors differs significantly from a proposed optimum model. The test is a randomization test based on the null hypothesis that the squared residuals for both models have the same distribution. Intuitively, one can think of the null hypothesis as stating that both models have the same predictive ability.

To obtain the van der Voet T^2 statistic given in the Cross Validation report, the calculation below is performed on each validation set. In the case of a single validation set, the result is the reported value. In the case of Leave-One-Out and KFold validation, the results for each validation set are averaged.

Denote by $R_{i,jk}$ the jth predicted residual for response k for the model with i extracted factors. Denote by $R_{opt,jk}$ is the corresponding quantity for the model based on the proposed optimum number of factors, opt. The test statistic is based on the following differences:

$$D_{i,jk} = R^2_{i,jk} - R^2_{opt,jk}$$

Suppose that there are K responses. Consider the following notation:

$$\mathbf{d}_{i,j} = (D_{i,j1}, D_{i,j2}, \ldots, D_{i,jK})'$$

$$\mathbf{d}_{i,.} = \sum_j \mathbf{d}_{i,j}$$

$$\mathbf{S_i} = \sum_j \mathbf{d}_{i,j}\mathbf{d}_{i,j}'$$

The van der Voet statistic for i extracted factors is defined as follows:

$$C_i = \mathbf{d}_{i,.}'\mathbf{S_i}^{-1}\mathbf{d}_{i,.}$$

The significance level is obtained by comparing C_i with the distribution of values that results from randomly exchanging $R^2_{i,jk}$ and $R^2_{opt,jk}$. A Monte Carlo sample of such values is simulated and the significance level is approximated as the proportion of simulated critical values that are greater than or equal to C_i.

T^2 Plot

The T^2 value for the i^{th} observation is computed as follows:

$$T^2_i = (n-1)\sum_{j=1}^{p}\left(t^2_{ij} \Big/ \sum_{k=1}^{n} t^2_{kj}\right)$$

where t_{ij} = X score for the i^{th} row and j^{th} extracted factor, p = number of extracted factors, and n = number of observations used to train the model. If validation is not used, n = total number of observations.

The control limit for the T^2 Plot is computed as follows:

$((n\text{-}1)^2/n)$*BetaQuantile(0.95, p/2, $(n\text{-}p\text{-}1)/2$)

where p = number of extracted factors, and n = number of observations used to train the model. If validation is not used, n = total number of observations. See Tracy et al. (1992).

Confidence Ellipses for X Score Scatterplot Matrix

The Score Scatterplot Matrices option adds 95% confidence ellipses to the X Score scatterplots. The X scores are uncorrelated because both the NIPALS and SIMPLE algorithms produce orthogonal score vectors. The ellipses assume that each pair of X scores follows a bivariate normal distribution with zero correlation.

Consider a scatterplot for score i on the vertical axis and score j on the horizontal axis. The coordinates of the top, bottom, left, and right extremes of the ellipse are defined as follows:

- the top and bottom extremes are +/-sqrt(var(score i)*z)
- the left and right extremes are +/-sqrt(var(score j)*z)

where z = ((n-1)*(n-1)/n)*BetaQuantile(0.95, 1, (n-3)/2). For background on the z value, see Tracy et al. (1992).

Standard Error of Prediction and Confidence Limits

Let **X** denote the matrix of predictors and **Y** the matrix of response values, which might be centered and scaled based on your selections in the launch window. Assume that the components of **Y** are independent and normally distributed with a common variance σ^2.

Hoskuldsson (1988) observes that the PLS model for **Y** in terms of scores is formally similar to a multiple linear regression model. He uses this similarity to derive an approximate formula for the variance of a predicted value. See also Umetrics (1995). However, Denham (1997) points out that any value predicted by PLS is a non-linear function of the Ys. He suggests bootstrap and cross validation techniques for obtaining prediction intervals. The PLS platform uses the normality-based approach described in Umetrics (1995).

Denote the matrix whose columns are the scores by **T** and consider a new observation on **X**, $\mathbf{x}_0$. The predictive model for **Y** is obtained by regressing **Y** on **T**. Denote the score vector associated with $\mathbf{x}_0$ by $\mathbf{t}_0$.

Let a denote the number of factors. Define s^2 to be the sum of squares of residuals divided by $df = n - a - 1$ if the data are centered and $df = n - a$ if the data are not centered. The value of s^2 is an estimate of σ^2.

Standard Error of Prediction Formula

The standard error of the predicted mean at $\mathbf{x}_0$ is estimated by the following:

$$SE(\overline{Y}_{x_0}) = s\sqrt{\left(\frac{1}{n} + \mathbf{t}_0(\mathbf{T}'\mathbf{T})^{-1}\mathbf{t}_0'\right)}$$

Mean Confidence Limit Formula

Let $t_{0.975,\,df}$ denote the 0.975 quantile of a t distribution with degrees of freedom $df = n - a - 1$ if the data are centered and $df = n - a$ if the data are not centered.

The 95% confidence interval for the mean is computed as follows:

$$\overline{Y}_{x_0} \pm t_{0.975,\,df} SE(\overline{Y}_{x_0})$$

Indiv Confidence Limit Formula

The standard error of a predicted individual response at $\mathbf{x}_0$ is estimated by the following:

$$SE(\hat{Y}_{x_0}) = s\sqrt{\left(\frac{1}{n} + 1 + \mathbf{t}_0(\mathbf{T}'\mathbf{T})^{-1}\mathbf{t}_0'\right)}$$

Let $t_{0.975,\,df}$ denote the 0.975 quantile of a t distribution with degrees of freedom $df = n - a - 1$ if the data are centered and $df = n - a$ if the data are not centered.

The 95% prediction interval for an individual response is computed as follows:

$$\overline{Y}_{x_0} \pm t_{0.975,\,df} SE(\hat{Y}_{x_0})$$

Standardized Scores and Loadings

Consider the following notation:

- n_{tr} is the number of observations in the training set
- m is the number of effects in X
- k is the number of responses in Y
- $VarX_i$ is the percent variation in X explained by the i^{th} factor
- $VarY_i$ is the percent variation in Y explained by the i^{th} factor
- $\mathbf{XScore}_i$ is the vector of X scores for the i^{th} factor

- $\mathbf{YScore}_i$ is the vector of Y scores for the i^{th} factor
- $\mathbf{XLoad}_i$ is the vector of X loadings for the i^{th} factor
- $\mathbf{YLoad}_i$ is the vector of Y loadings for the i^{th} factor

Standardized Scores

The vector of i^{th} Standardized X Scores is defined as follows:

$$\frac{\mathbf{XScore}_i}{(n_{tr}-1)\sqrt{mVarX_i/n_{tr}}}$$

The vector of i^{th} Standardized Y Scores is defined as follows:

$$\frac{\mathbf{YScore}_i}{(n_{tr}-1)\sqrt{kVarY_i/n_{tr}}}$$

Standardized Loadings

The vector of i^{th} Standardized X Loadings is defined as follows:

$$\mathbf{XLoad}_i\sqrt{mVarX_i}$$

The vector of i^{th} Standardized Y Loadings is defined as follows:

$$\mathbf{YLoad}_i\sqrt{kVarY_i}$$

PLS Discriminant Analysis (PLS-DA)

When a categorical variable is entered as Y in the launch window, it is coded using indicator coding. If there are k levels, each level is represented by an indicator variable with the value 1 for rows in that level and 0 otherwise. The resulting k indicator variables are treated as continuous and the PLS analysis proceeds as it would with continuous Ys.

Chapter 7

Multiple Correspondence Analysis

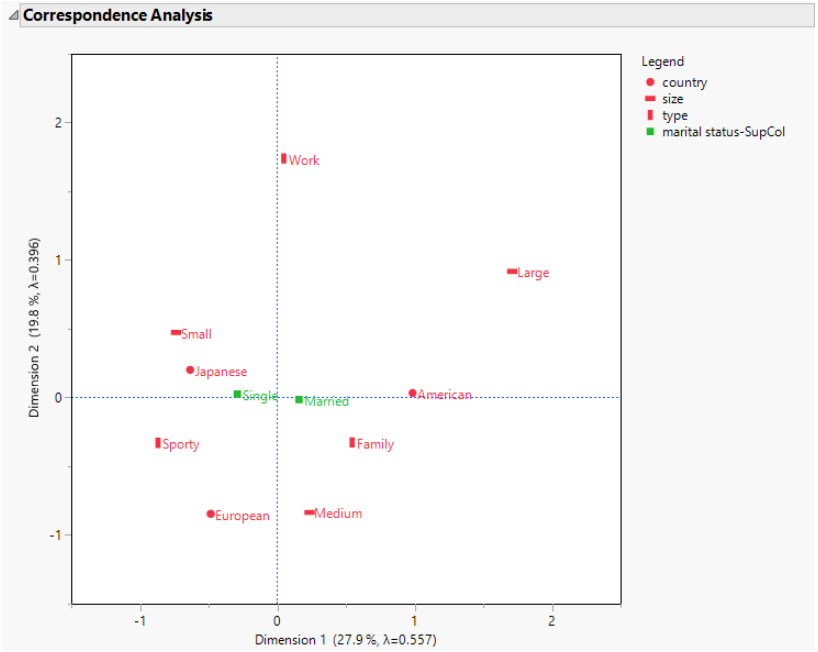
Identify Associations between Levels of Categorical Variables

Multiple Correspondence Analysis (MCA) takes multiple categorical variables and seeks to identify associations between levels of those variables. MCA extends correspondence analysis from two variables to many. It can be thought of as analogous to principal component analysis for quantitative variables. Similar to other multivariate methods, it is a dimension reducing method; it represents the data as points in 2- or 3-dimensional space.

Multiple correspondence analysis is frequently used in the social sciences. It can be used in survey analysis to identify question agreement. It is also used in consumer research to identify potential markets for products.

For more information about multiple correspondence analysis, see LeRoux and Rouanet (2010).

Figure 7.1 Multiple Correspondence Analysis



Contents

Example of Multiple Correspondence Analysis . . . 159
Launch the Multiple Correspondence Analysis Platform . . . 162
The Multiple Correspondence Analysis Report . . . 163
Multiple Correspondence Analysis Platform Options . . . 164
Correspondence Analysis Options . . . 166
Show Plot . . . 167
Show Detail . . . 167
Show Adjusted Inertia . . . 168
Show Coordinates . . . 168
Show Summary Statistics . . . 169
Show Partial Contributions to Inertia . . . 169
Show Squared Cosines . . . 170
Cochran's Q Test . . . 170
Cross Table . . . 170
Additional Examples of the Multiple Correspondence Analysis Platform . . . 171
Example Using a Supplementary Variable . . . 172
Example Using a Supplementary ID . . . 173
Example of Cochran's Q Test . . . 174
Statistical Details for the Multiple Correspondence Analysis Platform . . . 176
Details Report . . . 176
Adjusted Inertia . . . 177
Summary Statistics . . . 177
Partial Contributions to Inertia . . . 178
Cochran's Q Statistic . . . 178

Example of Multiple Correspondence Analysis

This example uses data collected from 55 JMP employees about their preferences, or tastes, in these areas:

- Preferred TV program (8 categories): news, comedy, police, nature, sport, film, drama, or soap operas.
- Preferred film (8 categories): action, comedy, costume drama, documentary, horror, musical, romance, or SciFi.
- Preferred type of art (7 categories): performance, landscape, renaissance, still life, portrait, modern, or impressionism.
- Preferred place to eat out (6 categories): burgers & fries, pub, Indian restaurant, Italian restaurant, French restaurant, or steak house.

The data also include demographics about the individuals polled. You want to explore the associations between the 55 responses to the four questions. This can lead to an understanding of different groups of employees.

Note: The measures for preferences are based on questions from LeRoux and Rouanet (2010).

1. Select **Help > Sample Data Library** and open Employee Taste.jmp.
2. Select **Analyze > Multivariate Methods > Multiple Correspondence Analysis**.
3. Select TV, Film, Art, and Restaurant and click **Y, Response**.

   In MCA, as in principal component analysis, factors are typically considered responses rather than some being responses and others explanatory.
4. Click **OK**.

Tip: Click and drag overlapping labels to rearrange them.

5. Below the plot, select **Proportional marker size**.

   The marker size indicates the relative proportion of responses in that category.

Figure 7.2 Part of the Initial Multiple Correspondence Analysis Report

The Correspondence Analysis report shows the projection of categories of the four variables onto the first two principal axes, or dimensions. Use the controls to change the dimensions plotted. The distance between points represents the dissimilarity between response patterns of the employees.

From this graph, you can use your knowledge of the subject matter to interpret the findings:

- Grouped near the Burgers and Fries restaurant preference, there is a cluster of the Sport and Police TV program preferences. This cluster could be called the “popular culture” group.
- Looking at Dimension 2, individual tastes move from the popular culture group to what could be classified as more “sophisticated” taste - those who like films such as documentaries, prefer TV dramas, and specialized restaurants, such as French and Indian cuisines.

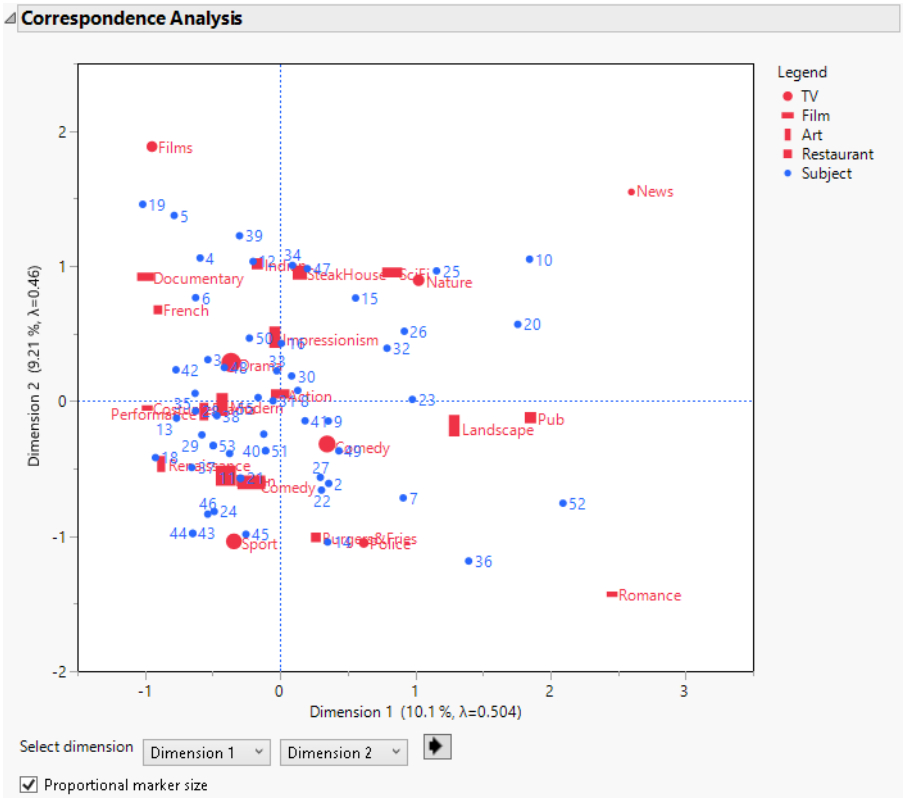
To obtain the scores for individuals, add the subject as the **X, Factor**:

6. Open the Variable Summary outline.

   The Variable Summary panel allows you to modify your analysis without relaunching the platform. It also provides a concise view of the completed analysis.

7. Select Subject and click **Add X**. The analysis automatically updates.

Figure 7.3 Multiple Correspondence Analysis Report with Subjects

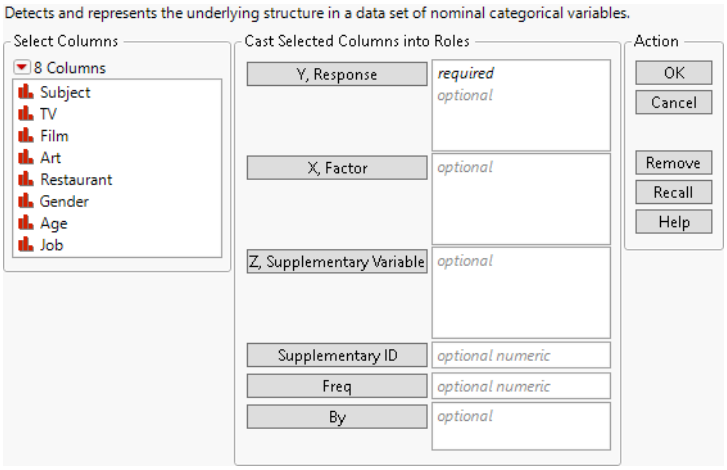


Including the subjects highlights a cluster of employees with similar taste in the lower left quadrant of the plot. These employees fall in the area we identified as popular culture.

Launch the Multiple Correspondence Analysis Platform

Launch the Multiple Correspondence Analysis platform by selecting **Analyze > Multivariate Methods > Multiple Correspondence Analysis**.

Figure 7.4 Multiple Correspondence Analysis Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Response Assigns the categorical columns to be analyzed. In MCA, you are generally interested in the associations between variables, but there are not explicit "explanatory" and "response" variables.

X, Factor Assigns one or more categorical columns to be used as factor, or explanatory, variables.

Note: Use a subject ID column as a single X to obtain scores on individuals.

Z, Supplementary Variable Assigns the columns to be used as supplementary variables. These variables are those you are interested in identifying associations with but not include in the calculations. Supplementary variables are used to improve data interpretation.

Supplementary ID Assigns the column that identifies rows to be used as supplementary. A supplementary ID column usually has 1s and 0s. The rows associated with ID 0 are treated as supplementary rows. The Supplementary ID column is ignored if there are levels of the

X or Y variables present in the supplementary rows that are not present in the non-supplementary rows.

Note: Only one of the Supplementary ID and Z, Supplementary Variable roles can be specified at one time.

Freq Assigns a frequency variable to this role. This is useful if your data are summarized.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

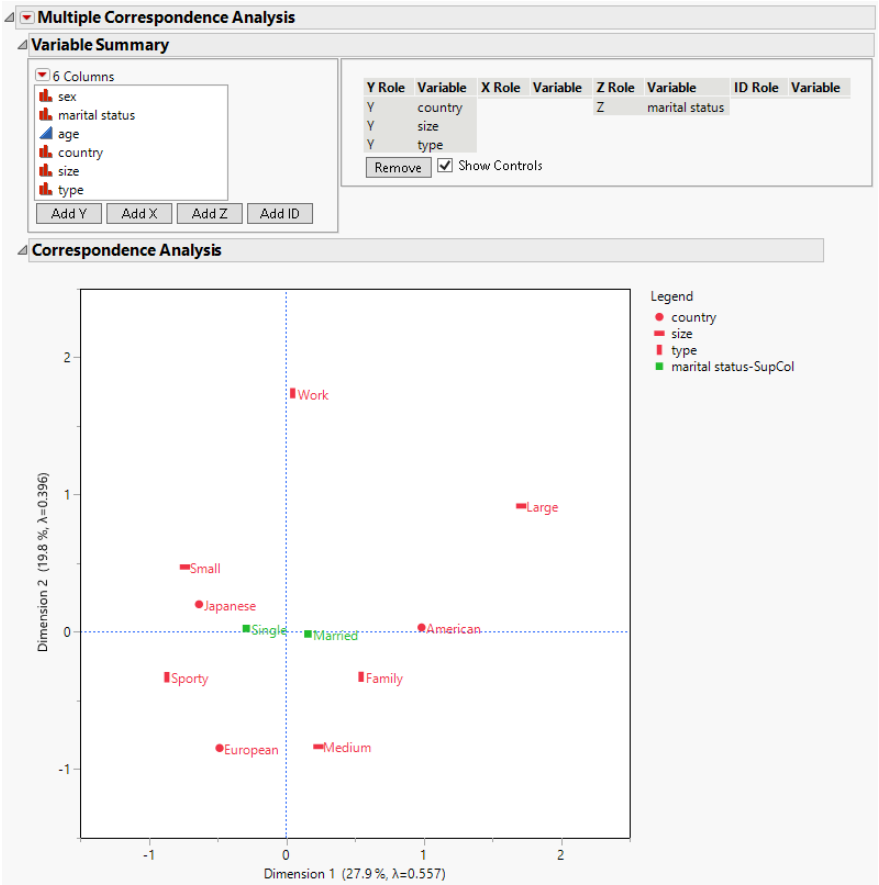
Note: The Multiple Correspondence Analysis platform handles missing values differently than many other JMP platforms. The analysis uses all nonmissing pairs of cells in a row. It does not remove entire rows from the computation.

The Multiple Correspondence Analysis Report

The initial Multiple Correspondence Analysis report shows the variable summary, correspondence analysis plot, and details of the dimensions of the data in order of importance. From the plot of the cloud of categories or individuals, you can identify associations that exist within the data. The details provide information about whether the two dimensions shown in the plot are sufficient to interpret the relationships within the data.

The Variable Summary in an interactive panel that allows you to modify your analysis without relaunching the platform. The panel shows the columns and roles used in the analysis. If you select the Show Controls check box, a list of the columns in the data table appears to the left. You can change the columns in the analysis either by selecting a column and clicking Add Y, Add X, Add Z, or Add ID. Or you can drag the column to the header in the variable summary table. This enables you to modify the analysis without returning to the launch window.

Figure 7.5 Multiple Correspondence Analysis Report with Show Controls Selected



Multiple Correspondence Analysis Platform Options

The Multiple Correspondence Analysis red triangle menu options give you the ability to customize reports according to your needs. The reports available are determined by the type of analysis that you conduct.

Cross Table Shows or hides a Burt or contingency table as appropriate for variable roles selected. See "Cross Table".

Correspondence Analysis Shows or hides the correspondence analysis reports. These reports give the plots, details, coordinates, and summary statistics. See "Additional Examples of the Multiple Correspondence Analysis Platform".

Cross Table of Supplementary Rows (Available when one or more supplementary Z and X variables are specified or when an ID is specified.) Shows or hides a Burt or contingency table, dependent on the following rules:

- When one or more X and Z variables are specified, shows or hides a contingency table of the X variable(s) versus the response variable(s).
- When an ID and no X variables are specified, shows or hides a Burt table for the supplementary observations. The supplementary observations are those observations for which the ID variable is equal to 0.
- When an ID and one or more X variables are specified, shows or hides a contingency table for the supplementary observations. The supplementary observations are those observations for which the ID variable is equal to 0.

Cross Table of Supplementary Columns (Available when one or more supplementary Z variables are specified.) Shows or hides a contingency table dependent on the following rules:

- When no X variable is specified, shows or hides a contingency table of the Response variable(s) versus the supplementary variable(s).
- When one or more X variables are specified, shows or hides a contingency table of the X variable(s) versus the supplementary variable(s).

Mosaic Plot (Available only when there is one X variable and one Y variable.) Shows or hides a mosaic plot Y versus X. A mosaic plot is a stacked bar chart where each segment is proportional to its group's frequency count.

Tests for Independence (Available only when there is one X variable and one Y variable.) Shows or hides the tests for independence between the X and Y variables. There are two versions of this test, the Pearson form and the Likelihood Ratio form, both with chi-square statistics.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Correspondence Analysis Options

The reports available under Correspondence Analysis are determined by the type of multiple correspondence analysis that you conduct. Several of these reports are shown by default.

Show Plot Shows the two-dimensional cloud of categories in the plane described by the first two principal axes. This plot appears by default. The plot uses isometric scaling.

Show Detail Provides the details of the analysis including the singular values, inertias, ChiSquare statistics, percent, and cumulative percent. This report appears by default. See "Show Detail".

Show Adjusted Inertia Provides reports of the Benzecri and Greenacre adjusted inertia. See Benzécri (1979) and Greenacre (1984). This option is not available when there are one or more X variables. See "Show Adjusted Inertia".

Show Coordinates Provides a report of up to the first three principal coordinates for the categories in the analysis, as appropriate. See "Show Coordinates".

Show Summary Statistics Provides a report of the summary statistics, Quality, Mass, and Inertia, for each category in the analysis. See "Show Summary Statistics".

Show Partial Contributions to Inertia Provides a report of the contribution of each category to the inertia for each of up to the first three dimensions. See "Show Partial Contributions to Inertia".

Show Squared Cosines Provides a report of the squared cosines of each category for each of up to the first three dimensions. The report includes a bar chart that shows, for each level of each Y variable, the squared cosine values for each of up to the first three dimensions. See "Show Squared Cosines".

Cochran's Q Test (Available only when all of the Y variables have the same set of only two levels and the X variable has a unique value for each row.) Provides Cochran's *Q* statistic, which tests that the marginal probability of a specific response is unchanged across the Y variables. Cochran's *Q* statistic is a generalization of McNemar's statistic for more than two response variables. See Agresti (2013). See "Cochran's Q Test".

3D Correspondence Analysis Shows the three-dimensional cloud of categories of the Y, X, and Z variables in the space described by the first three principal axes. This option is not available if there are less than three dimensions.

Save Coordinates Saves the principal coordinates to one or more JMP data tables. Column coordinates, row coordinates, supplementary column coordinates, and supplementary row coordinates are saved to separate JMP data tables. You can choose how many columns to save.

Save Coordinate Formula Saves formula columns to the data table for the principal coordinates in multiple dimensions. The value for each observation is the average of the coordinates for the Y variables scaled by the singular value for each dimension. You can choose how many columns to save.

Show Plot

The plot displays a projection of the projection of categories or individuals onto the plane described by the first two principal axes. The plot uses isometric scaling. You can toggle the dimensions shown in the plot using the Select Dimension controls below the plot. The first control defines the horizontal axis of the plot, and the second control defines the vertical axis of the plot. Click the arrow button to cycle through the dimensions shown in the plot. Use the **Proportional marker size** check box to specify if the size of the points in the plot should be proportional to the count of observations corresponding to each point.

Note: Selecting a point in the correspondence analysis plot also selects the corresponding rows in other tables in the report window. However, rows in the data table are not selected. To select all of the points in the plot associated with a particular variable, select the name of the variable in the plot legend.

Show Detail

Shows the table of singular values.

Singular Value Shows the singular values in the singular value decomposition of the contingency table or Burt table. For the formula, see "Details Report".

Inertia Lists the square of the singular values, reflecting the relative variation accounted for in the canonical dimensions.

ChiSquare Lists the portion of the overall Chi-square for the Burt or contingency table represented by the dimension.

Percent Portion of inertia with respect to the total inertia.

Cumulative Percent Shows the cumulative portion of inertia. If the first two singular values capture the bulk of the inertia, then the 2-D correspondence analysis plot is sufficient to show the relationships in the table.

Show Adjusted Inertia

The principal inertias of a Burt table in MCA are the eigenvalues. The problem with these inertias is that they provide a pessimistic indication of fit. Benzécri proposed an inertia adjustment. Greenacre argued that the Benzécri adjustment overestimates the quality of fit and proposed an alternate adjustment. Both adjustments are calculated for your reference. See "Adjusted Inertia".

Inertia Lists the square of the singular values, reflecting the relative variation accounted for in the canonical dimensions.

Adjusted Inertia Lists the adjusted inertia according to either the Benzécri or Greenacre adjustment.

Percent Portion of adjusted inertia with respect to the total inertia.

Cumulative Percent Shows the cumulative portion of adjusted inertia. If the first two singular values capture the bulk of the inertia, then the 2-D correspondence analysis plot is sufficient to show the relationships in the table.

Show Coordinates

Shows the Column Coordinates table or the Row and Column Coordinates tables.

X Lists the columns specified as X, Factor variables.

Y Lists the columns specified as Y, Response variables.

Z Lists the columns specified as Z, Supplementary Variables.

Category Lists the levels of the X, Y, or Z variables.

Dimension 1, Dimension 2, Dimension 3 For each level or each response, lists its coordinate along the indicated principal axis. By default, the tables show coordinates for up to the first three dimensions. Coordinate columns for additional dimensions are hidden. To show these optional columns, right-click in a table and select the dimension columns from the **Columns** submenu.

Note: If there are columns specified as X, Factor variables, the Coordinates report displays tables of both X and Y with the same report headings. If a Z, Supplementary Variable is specified, the coordinates are listed below the X and Y coordinates as applicable.

Show Summary Statistics

Shows the Summary Statistics for the Column Points table or the Summary Statistics for the Row and Column Points tables. The Y table gives Quality, Mass, and Inertia for each level of each response, called a *column point*. The X table gives Quality, Mass, and Inertia for each level of the X, Factor variables. See "Summary Statistics".

X Lists the columns specified as X, Factor variables.

Y Lists the columns specified as Y, Response variables.

Category Lists the levels of the X or Y variables.

Quality(dim=2) Lists the quality of the representation of the level by the solution.

Mass Lists the row frequency for the level of the response divided by the total frequency. In the Burt table, this is the Total % for each row.

Inertia Lists the proportion of the total inertia accounted for by the level of the response. The inertia values sum to one across the levels and their responses.

Note: If there are columns specified as X, Factor variables, the Summary Statistics report displays tables of both X and Y with the same report headings.

Show Partial Contributions to Inertia

Shows the Partial Contributions to Inertia for the Column Points table or the Partial Contributions to Inertia for the Row and Column Points tables. Also shows the Plot of Partial Contributions to Inertia for the Column Points. This is a bar chart that shows, for each level of each Y variable, its partial contributions to each of the dimensions shown in the table.

X Lists the columns specified as X, Factor variables.

Y Lists the columns specified as Y, Response variables.

Category Lists the levels of the X or Y variables.

Dimension 1, Dimension 2, Dimension 3 Lists the contribution of the response or factor level to the inertia of the indicated dimension. By default, the tables show columns for up to the first three dimensions. Additional columns are hidden. To show these optional columns, right-click a table and select the dimension columns from the **Columns** submenu.

Each level of each response contributes to the inertia of each dimension. The partial contributions within each dimension sum to 1.

Note: If there are columns specified as X, Factor variables, the Partial Contributions to Inertia report displays tables of both X and Y with the same report headings. See "Partial Contributions to Inertia".

Show Squared Cosines

Shows the Squared Cosines for the Column Points table or the Squared Cosines for the Row and Column Points. Also shows the Plot of Squared Cosines for the Column Points. This is a bar chart that shows, for each level of each Y variable, the squared cosine values for each of up to the first three dimensions shown.

X Lists the columns specified as X, Factor variables.

Y Lists the columns specified as Y, Response variables.

Category Lists the levels of the X or Y variables.

Dimension 1, Dimension 2, Dimension 3 Lists the *quality* of the representation of the level by the indicated dimension. By default, the tables show results for up to the first three dimensions. Additional columns are hidden. To show these optional columns, right-click a table and select the dimension columns from the **Columns** submenu.

The values indicate the quality of each point for the indicated dimension. The squared cosine can be interpreted as the squared correlation of the point with the dimension. The sum of the squared cosines of the first two dimensions equalsQuality(dim=2) in the Summary Statistics report. See "Summary Statistics".

Note: If there are columns specified as X, Factor variables, the Squared Cosines report displays tables of both X and Y with the same report headings.

Cochran's Q Test

Cochran's *Q* test is a nonparametric test for homogeneity across matched samples for three or more binary outcomes. You can use it to test for differences in proportions in matched pairs. Cochran's *Q* test is an extension of the McNemar test used for two outcomes.

Cross Table

The cross table option hides or shows a Burt table or a contingency table. When you select multiple Y, Response columns with no X, Factor columns, the Burt table is created. If you select any X, Factor columns, a traditional contingency table is created instead of a Burt table. The outline node title reflects the structure of the cross table.

The Burt table is the basis of the multiple correspondence analysis. It is a partitioned symmetric table of all pairs of categorical variables. The diagonal partitions are diagonal matrices (a cross-table of a variable with itself). The off-diagonal partitions are ordinary contingency tables.

The red triangle menu for the Burt or contingency table contains the following options of statistics to display in the table.

Count Cell frequency, margin total frequencies, and grand total (total sample size). This appears by default.

Total % Percent of cell counts and margin totals to the grand total. This appears by default.

Cell Chi Square Chi-square values computed for each cell as $(O - E)^2 / E$.

Col % Percent of each cell count to its column total.

Row % Percent of each cell count to its row total.

Expected Expected frequency (E) of each cell under the assumption of independence. Computed as the product of the corresponding row total and column total divided by the grand total.

Deviation Observed cell frequency (O) minus the expected cell frequency (E).

Col Cum Cumulative column total.

Col Cum % Cumulative column percentage.

Row Cum Cumulative row total.

Row Cum % Cumulative row percentage.

Make Into Data Table Creates one data table for each statistic shown in the table.

Additional Examples of the Multiple Correspondence Analysis Platform

- "Example Using a Supplementary Variable"
- "Example Using a Supplementary ID"
- "Example of Cochran's Q Test"

Example Using a Supplementary Variable

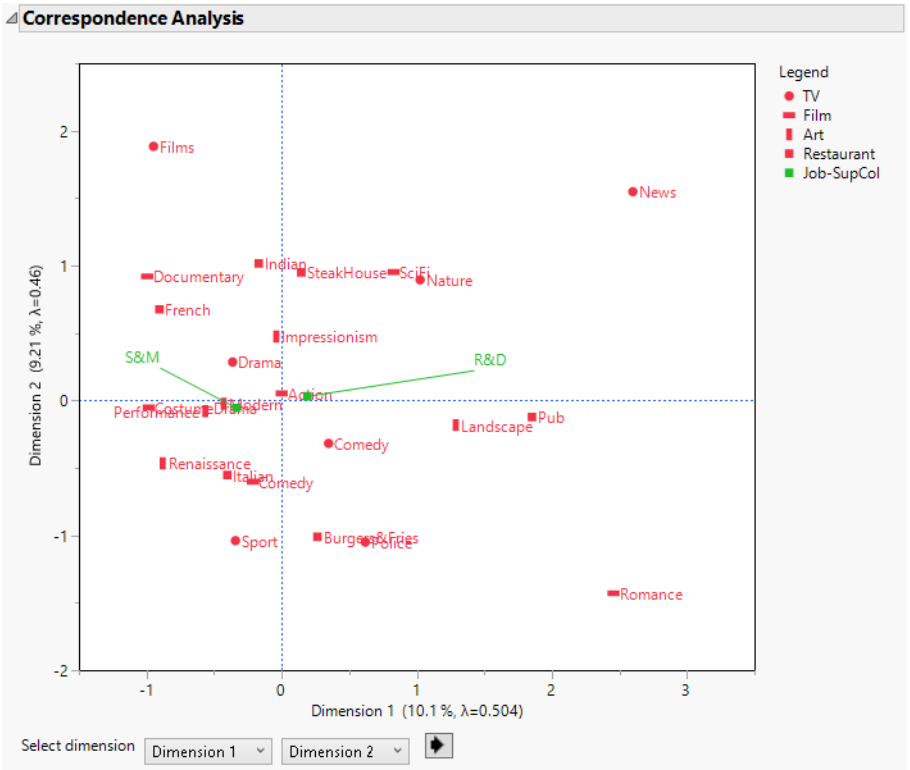
This example uses the sample data on JMP employee preferences described in "Example of Multiple Correspondence Analysis".

You want to explore relationships between employee job and their preferences.

1. Select **Help > Sample Data Library** and open Employee Taste.jmp.
2. Select **Analyze > Multivariate Methods > Multiple Correspondence Analysis**.
3. Select TV, Film, Art, and Restaurant and click **Y, Response**.
4. Select Job and click **Z, Supplementary Variable**.
5. Click **OK**.

Note: Since Job is a supplementary variable, it is not used in the calculations. Job type is plotted after the calculations are complete.

Figure 7.6 MCA with Supplementary Variable Report



You can see from the plot that the data points for the R&D (research and development) and the S&M (sales and marketing) employees are close together. We interpret this to mean that there is not a strong difference in terms of preferences due to employee job function. In other words, the employee's job role is not a significant factor in their preferences.

Example Using a Supplementary ID

The United States census allows for examining population growth over the last century. The US Regional Population.jmp sample data table contains populations of the 50 US states grouped into regions for each of the census years from 1920 to 2010. Alaska and Hawaii are treated as supplementary regions because they were not states during the entire time, and they are not part of the contiguous United States. You are interested in whether the population growth in these two states differs from the rest of the US.

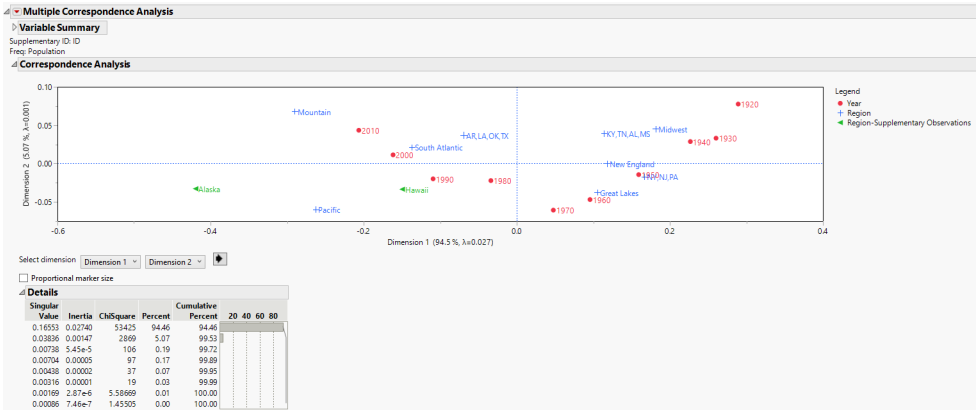
1. Select **Help > Sample Data Library** and open US Regional Population.jmp.
2. Select **Analyze > Multivariate Methods > Multiple Correspondence Analysis**.
3. Select Year and click **Y, Response**.
4. Select Region and click **X, Factor**.
5. Select ID and click **Supplementary ID**.
6. Select Population and click **Freq**.
7. Click **OK**.

The Details report shows that the association between years and regions is almost entirely explained by the first dimension. The plot shows that years are in the correct order on the first dimension. This ordering occurs naturally through the correspondence analysis; there is no information about the order provided to the analysis. The plot highlights the isometric scale used to plot the data.

Notice that the ordering of the regions reflects the population shift from the Midwest to the Northeast to the South and finally to the Mountain and West.

Alaska and Hawaii were not used in the computation of the analysis but are plotted based on the results. Their growth pattern is most similar to the Pacific states. Alaska's growth is even more extreme than the Pacific region.

Figure 7.7 MCA with Supplementary ID Report

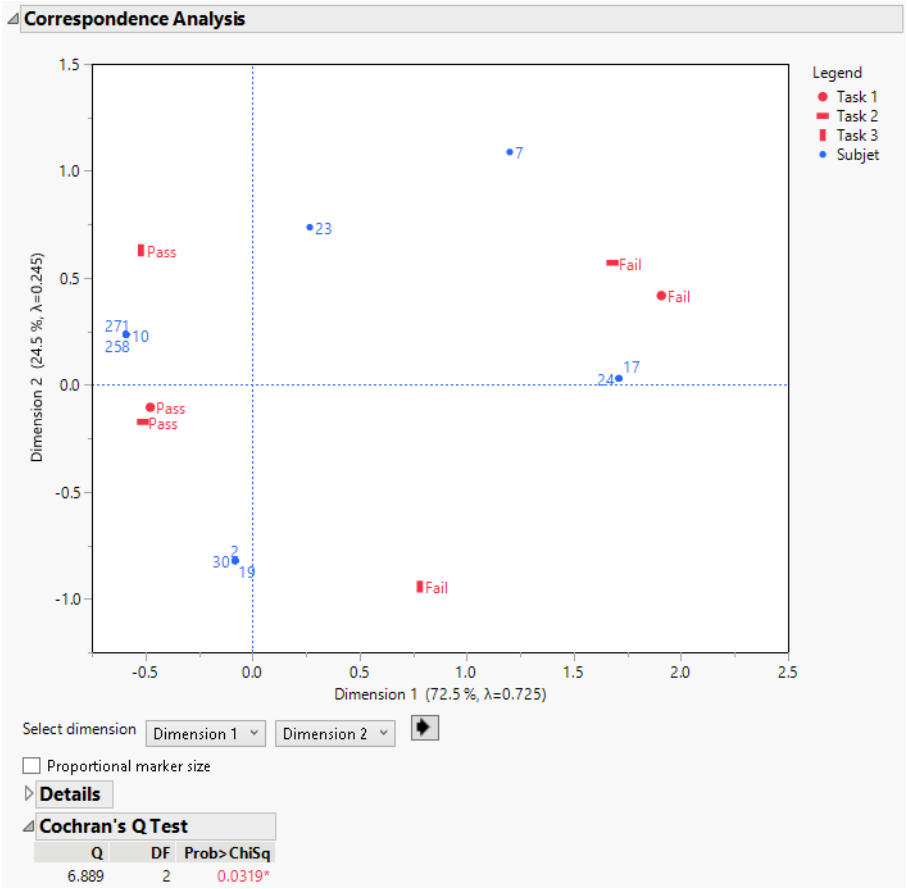


Example of Cochran's Q Test

This example uses a simulated data table to evaluate the ease of completion of three tasks for 30 subjects. Task 1 was simulated to have an 80% completion rate. Tasks 2 and 3 were simulated to have the same result as Task 1 for 95% and 80% of subjects respectively.

1. Select **Help > Sample Data Library** and open Cohcrans Q.jmp.
2. Select **Analyze > Multivariate Methods > Multiple Correspondence Analysis**.
3. Select Task 1, Task 2, and Task 3 and click **Y, Response**.
4. Select Subject and click **X, Factor**.
5. Click **OK**.
6. Click the Multiple Correspondence Analysis red triangle menu and select **Correspondence Analysis > Cochran's Q Test**.

Figure 7.8 Cochran's *Q* Test



The correspondence plot shows the similarity between Task 1 and Task 2 as the Pass and Fail outcomes cluster together. However, Task 3 is farther away from Tasks 1 and 2. Cochran's *Q* test statistic of 6.889 has an associated *p*-value of 0.0319. This *p*-value supports the rejection of the hypothesis that the pass rates are the same across all tasks at a 0.05 significance level.

Statistical Details for the Multiple Correspondence Analysis Platform

- "Details Report"
- "Adjusted Inertia"
- "Summary Statistics"
- "Partial Contributions to Inertia"
- "Cochran's Q Statistic"

Details Report

When a simple Correspondence Analysis is performed, the report lists the singular values from the following singular value decomposition:

$$\mathbf{D}_r^{-0.5}(\mathbf{P} - rc')\mathbf{D}_c^{-0.5} = \mathbf{U}Diag(\Lambda)\mathbf{V}'$$

where:

- $\mathbf{P}$ is the matrix of counts divided by the total frequency
- r and c are the row and column sums of $\mathbf{P}$
- the $\mathbf{D}$ matrices are diagonal matrices of the values of r and c
- Λ is the column vector of singular values

When Multiple Correspondence Analysis is performed, the singular value decomposition extends to the following equation:

$$\mathbf{D}^{-0.5}(\mathbf{C} - dd')\mathbf{D}^{-0.5} = \mathbf{U}Diag(\Lambda)\mathbf{V}'$$

where:

- $\mathbf{C}$ is the Burt table
- d is a column vector of the column sums of $\mathbf{C}$ (d is also the row sums, since $\mathbf{C}$ is symmetric)
- $\mathbf{D}$ is a diagonal matrix of the values of d

In the Details report, the inertia is the column vector Λ. The singular value is the square root of the inertia vector. The column coordinates are calculated as follows:

$$\mathbf{D}^{-0.5}\mathbf{V}Diag\left(\Lambda^{0.5}\right)$$

Adjusted Inertia

The usual principal inertias of a Burt table constructed from m categorical variables in MCA are the eigenvalues λ_k from Λ^2. These inertias provide a pessimistic indication of fit. Benzécri (1979) proposed the following inertia adjustment; it is also described by Greenacre (1984, p. 145):

$$\left(\frac{m}{m-1}\right)^2 \times \left(\lambda_k - \frac{1}{m}\right)^2 \quad \text{for} \quad \lambda_k > \frac{1}{m}$$

This adjustment computes the percent of adjusted inertia relative to the sum of the adjusted inertias for all inertias greater than $1/m$.

Greenacre (1984, p. 156) argues that the Benzécri adjustment overestimates the quality of fit. Greenacre proposes instead to compute the percentage of adjusted inertia relative to:

$$\frac{m}{m-1}\left(\text{tr}\left(Diag\left(\Lambda^4\right)\right) - \frac{n_c - m}{m^2}\right)$$

for all inertias greater than $1/m$, where $\text{tr}(Diag(\Lambda^4))$ is the sum of squared inertias and n_c is the total number of categories across the m variables.

Summary Statistics

Quality is the ratio of the squared distance of a point from the origin in the space defined by the specified number of dimensions to the distance from the origin in the space with the maximum number of dimensions. For the Chi-Square metric, a point's quality in a given dimension can be obtained from the cosine that its vector makes with the vector that defines the dimension. Quality is also equal to the ratio of the sum of inertias in the specified dimensions to the sum of the inertias in all dimensions. Quality indicates how well the point is represented in the lower-dimensional space.

Mass is the proportion of row or column total frequency to the total frequency.

Inertia is analogous to variance in principal component analysis. The overall inertia is the total Pearson Chi-square for a two-way frequency table divided by the sum of all observations in the table.

Relative inertia is the proportion of the contribution of the point to the overall inertia. In the summary statistics table, the relative inertia is listed in the column labeled Inertia.

Partial Contributions to Inertia

The contribution of a row or column to the inertia of a dimension is calculated as follows:

contribution = (mass)(coordinate)2/(dimension inertia)

Cochran's Q Statistic

The Cochran's Q test statistic is calculated as follows:

$$Q = k(k-1)\frac{\sum_{j=1}^{k}\left(\sum_{i=1}^{b} X_{ij} - \frac{N}{k}\right)^2}{\sum_{i=1}^{b}\left(\sum_{j=1}^{k} X_{ij}\left(k - \sum_{j=1}^{k} X_{ij}\right)\right)}$$

where

k is the number of treatments

b is the number of blocks

X_{ij} is the response (0 or 1) for the i^{th} subject for the k^{th} treatment

N is the grand total of positive responses across all subjects and treatments

Chapter 8

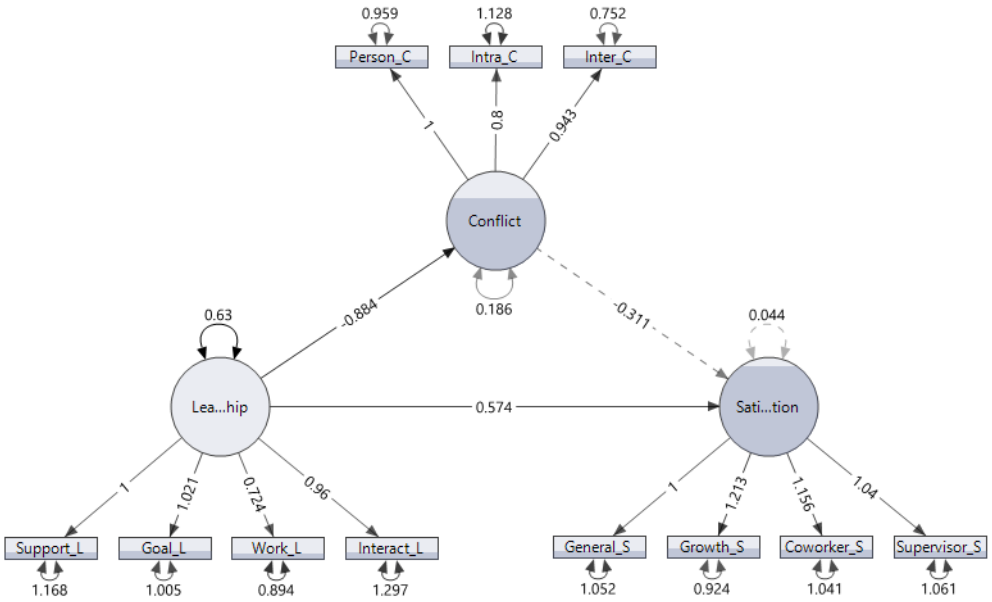
JMP PRO Structural Equation Models

Fit Structural Equation Models

The Structural Equation Models platform is available only in JMP Pro.

The Structural Equation Models platform enables you to fit a variety of models. These models include confirmatory factor analysis, path models with or without latent variables, measurement error models, and latent growth curve models. You can specify the model by selecting *From* and *To* variables and indicating how they are linked. The link is represented by either a unidirectional or a bidirectional arrow in the diagram. These selections populate two alternative views of the model: a dynamic path diagram and a list of model edges. The platform also checks for model identification as you specify the model.

Figure 8.1 Structural Equation Model Path Diagram



Contents

Overview of Structural Equation Models 181
Example of a Structural Equation Model 184
Launch the Structural Equation Models Platform 187
The Structural Equation Models Report 189
Model Specification Report 189
Model Comparison Report 195
Chi-Square Difference Test Report 196
Structural Equation Model Fit Report 197
Structural Equation Models Platform Options 198
Model Options 200
Customize the SEM Path Diagram 205
Path Diagram Pop-Up Menu Options 205
Customize Diagram Appearance Options 207
Additional Examples of Structural Equation Models 208
Example of a Latent Path Variable Model 208
Example of a Latent Growth Curve Model 215
Example of Using the Assess Measurement Model Report 217
Statistical Details for the Structural Equation Models Platform 220
Statistical Details for Summary of Fit Measures 220

JMP PRO Overview of Structural Equation Models

The Structural Equation Models (SEM) platform enables you to fit a wide variety of models that can be used to test theories of relationships among variables. The variables in the models can be observed (*manifest* variables) or unobserved (*latent* variables). Structural equation modeling is popular in the social and behavioral sciences.

By default, the platform specifies a model with means and variances for all variables. The platform then provides a model-building interface that enables you to see multiple views of the model while it is being built. It also provides some model details during the model construction process that alert you to untenable models prior to running the model.

After you fit one or more models, you can compare the fitted models and two baseline models in the Model Comparison report. The baseline models are an unrestricted model and an independence model. The unrestricted model is a fully saturated model, which fits all means, variances, and covariances of the specified Model Variables without imposing any structure on the data. The independence model fits all means and variances of the specified Model Variables. All covariances among the specified Model Variables are fixed to zero, which leads to a highly restrictive model.

The SEM platform uses the full information maximum likelihood (Finkbeiner 1979) method. This enables you to fully use all available information from the data even when there is a high proportion of observations with random missing values.

For more information about structural equation modeling, see the CALIS Procedure chapter in SAS Institute Inc. (2020a), Bollen (1989), and Kline (2016).

Note: All models in the Structural Equation Models platform are estimated with a mean structure, which means that a Constant term is included. If you do not want to place a structure on the means of the observed variables, then the means should be freely estimated as in the default model specification.

Types of Models

This section describes some of the various types of models that can be fit in the Structural Equation Models platform:

- *Path Analysis* enables you to test alternative explanatory models of the associations between observed variables. This technique is often used when only one variable is available per construct of interest in a study. Perhaps the simplest Path Analysis model is a standard regression model, in which X predicts Y. The SEM platform enables you to fit this simple regression model but you can also specify more interesting models as well. For example, you might have variable Z that is presumed, based on theory or previous research, to be a mediator of the X $\Rightarrow$ Y relationship. In other words, X predicts Z, which

then predicts Y. Thus, the original X ⇒ Y relationship might exist only because Z is excluded in the original model. Path analysis can be carried out in the SEM platform by performing the following steps:

1. Select all the observed variables in the launch window, click Model Variables, and click OK.
2. In the Model Specification report, select the predictors in the From List and the corresponding outcomes in the To List, and click the unidirectional arrow button.

Note: All exogenous variables (those that do not have any unidirectional arrows pointing at them) must be freely correlated in the model, unless a hypothesis of zero correlation is being tested. These covariances are specified with the bidirectional arrow button.

- *Confirmatory Factor Analysis* (CFA) enables you to test alternative measurement models. CFA is often used in survey development and as an initial step prior to fitting structural regression models. The SEM platform enables fitting of confirmatory factor analysis models by performing the following steps:
  1. Select all the observed variables in the launch window, click Model Variables, and click OK.
  2. Using the To List under Model Specification, select the variables that are presumed to load onto a latent variable.
  3. Enter the name for the latent variable in the box below the To List, and click the add latent ⊞ button.
  4. Repeat this process until all the latent variables for the model have been specified.

  Note that the SEM platform always includes a mean structure, so all of the observed variables are listed in the Means/Intercepts list as outcomes of the Constant term. Moreover, all latent variables are automatically identified by setting the loading of their first indicator to 1 (default) or their variance to 1 if the Standardized Latent Variables option was selected in the launch window. Finally, the traditional CFA model allows all latent variables to covary. You can specify these covariances by selecting all the latent variables in the From and To Lists and then clicking the bidirectional arrow button.
- *Structural Regression* (SR) models are also known as *path analysis with latent variables*. These models are often used after having identified an appropriate measurement model through confirmatory factor analysis (CFA). SR models enable you to test specific patterns of relationships between latent variables. In other words, while CFA does not impose any directionality in the effects between latent variables (all latent variables are allowed to freely covary), SR models do. In an example where management Leadership is hypothesized to lead to less team Conflict and more employee Satisfaction in the workplace, the Leadership latent variable can predict the Conflict and Satisfaction latent variables. You can specify these directional effects (regressions) after performing a CFA by performing the following steps:

1. Select all the observed variables in the launch window, click Model Variables, and click OK.
2. Using the To List under Model Specification, select the variables that are presumed to load onto a latent variable.
3. Enter the name for the latent variable next in the box below the To List, and click the add latent ⊞ button.
4. Repeat this process until all the latent variables for the model have been specified.
5. In the Model Specification report, select the predictors in the From List and the corresponding outcomes in the To List, and click the unidirectional arrow button.

- *Latent Growth Curve* (LGC) models enable you to fit and test alternative latent trajectories to repeated measures data. These models are very similar to random effects models in the mixed models framework. Often, you want to compare a no-growth model with a linear model. In a no-growth model, individuals can vary in their starting point but have flat trajectories. In a linear model, individuals can vary in both their starting point and linear slope over time. If enough data are available, you can also compare these models with a quadratic model where individuals can vary in their starting point and their linear and quadratic rates of change over time. You can fit LGC models using the SEM platform model specification, but the platform streamlines the fitting of LGC models by performing the following steps:

  1. Select all the observed variables (repeated measures) in the launch window, click Model Variables, and click OK.

     Note: For the Latent Growth model shortcuts to specify the model correctly, the observed variables must be listed in ascending time order and must have equal time intervals.

  2. Using the Model Shortcuts option, select the Longitudinal Analysis > Intercept-Only Growth Curve model, and then click Run.
  3. Using the Model Shortcuts option, select the Longitudinal Analysis > Linear Latent Growth Curve model, and then click Run.
  4. Using the Model Shortcuts option, select the Longitudinal Analysis > Quadratic Latent Growth Curve model, and then click Run.

  The Model Comparison table shows the alternative fit indices and the best model can be selected.

- *Conditional Latent Growth Curve* models can be used after identifying an ideal growth trajectory following the steps above. At this point, predictors of the intercept and change factors can be added to the model. These predictors might prove to be important factors for determining initial scores on a growth process and ensuing changes. To fit a conditional LGC, select all of the observed variables (repeated measures), including the

hypothesized predictors of the latent variables, in the launch window. Make sure that the predictors are the last variables in the Model Variables list to facilitate the following steps:

1. Use the Model Shortcuts option to select the appropriate growth trajectory. This option specifies all variables in the LGC model, including the predictors. Thus, you need to exclude the predictors from the growth process and correctly specify them as predictors.
2. Find the predictors in the Loadings list, select all the effects that involve them, and click Remove.
3. Select the predictors in the From List and the Intercept or Slope in the To List.
4. Click the unidirectional arrow to specify the conditional LGC.

Note: If you have more than one predictor, their covariances must be specified by selecting the predictors in the From and To Lists and clicking the bidirectional arrow button.

JMP PRO Example of a Structural Equation Model

In this example, an employee in a human resources department wants to develop a survey to measure key workplace constructs. The example uses the Structural Equation Models platform to analyze responses to a survey of 200 individuals regarding aspects of their workplace. The survey contains responses to 11 questions related to their job satisfaction. You seek to build a confirmatory factor analysis model that relates the answers to the survey questions to the latent variables of leadership characteristics, role conflict, and overall job satisfaction.

1. Select **Help > Sample Data Library** and open Job Satisfaction.jmp.
2. Select **Analyze > Multivariate Methods > Structural Equation Models**.
3. Select Support_L through Supervisor_S and click **Model Variables**.
4. Click **OK**.

   The Structural Equation Models report Model Specification outline appears.

Create Latent Variables

5. Select Support_L through Interact_L in the To List, type Leadership in the box below the To List, and click the add latent ⊞ button.
6. Select Person_C through Inter_C in the To List, type Conflict in the box below the To List, and click the add latent ⊞ button.
7. Select General_S through Supervisor_S in the To List, type Satisfaction in the box below the To List, and click the add latent ⊞ button.

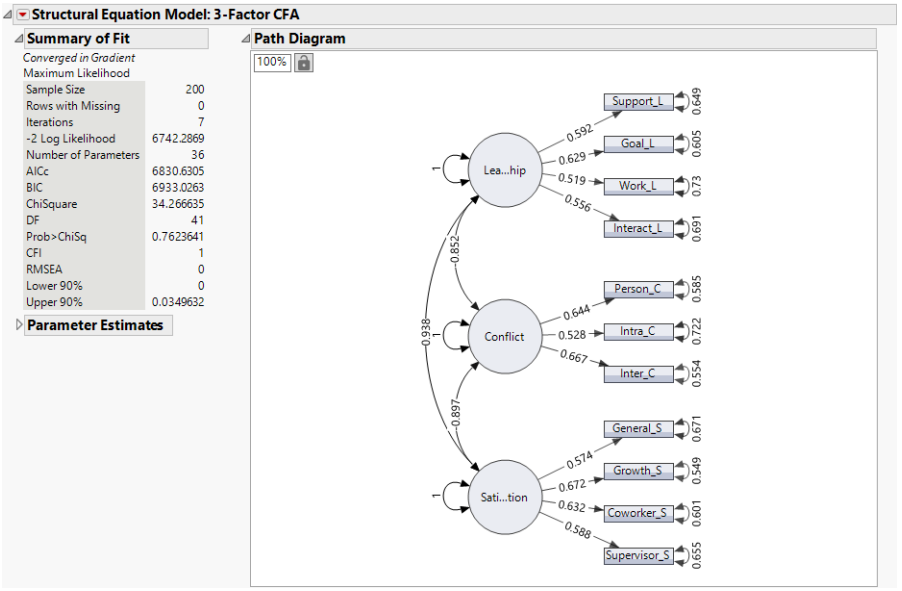
Add Covariances and Fit the Model

8. Select Leadership, Conflict, and Satisfaction in the From List, select Leadership, Conflict, and Satisfaction in the To List, and click the bidirectional arrow ↔ button.
9. In the text box below Model Name (in the top left of the Model Specification report), type 3-Factor CFA.
10. Click **Run**.
11. (Optional.) Click the red triangle next to Structural Equation Model: Path Analysis with Latent Variables and select **Path Diagram Settings > Layout > Top to Bottom**.
12. (Optional.) Click the gray disclosure icon next to Parameter Estimates.

    Closing the Parameter Estimates report enables you to see the entire path diagram.
13. Right click the path diagram and select **Show > Show Estimates > Standardized**.

    The numbers in the Path Diagram now represent the standardized parameter estimates for the model.

Figure 8.2 Structural Equation Model Report



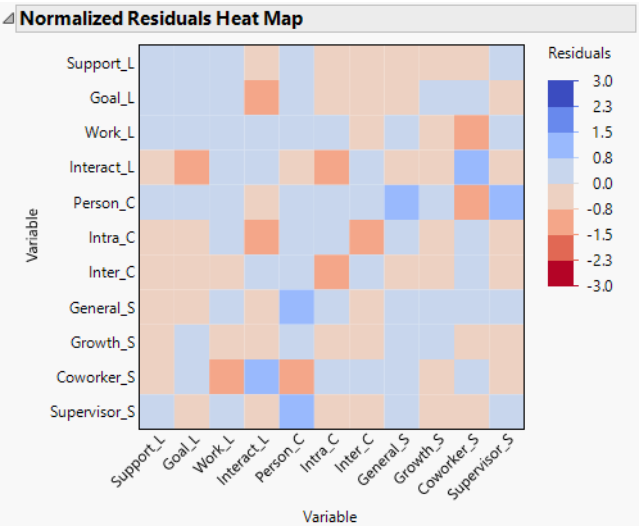
The chi-square statistic for this model, listed in the Summary of Fit report, is 34.27 with 41 degrees of freedom. Note that the corresponding *p*-value is 0.7624, which is not significant. This indicates that there is not evidence to reject the null hypothesis that the model fits well. Therefore, you conclude that this model fits the data reasonably well.

The chi-square value depends on the sample size, and thus, some well-fitting models can still produce a significant chi-square value. The comparative fit index (CFI) and root mean square error of approximation (RMSEA) provide additional guidance for determining

model fit. These indices are bounded between 0 and 1. CFI values greater than 0.90 and RMSEA values less than 0.10 are preferred (Browne and Cudeck 1993; Hu and Bentler 1999). Here, the CFI of 1 and RMSEA of 0 indicate an excellent fit. You conclude that the survey is a good instrument for measuring the latent variables for leadership, conflict, and satisfaction.

14. Click the red triangle next to Structural Equation Model: 3-Factor CFA and select **Normalized Residuals Heat Map**.

Figure 8.3 Normalized Residuals Heat Map

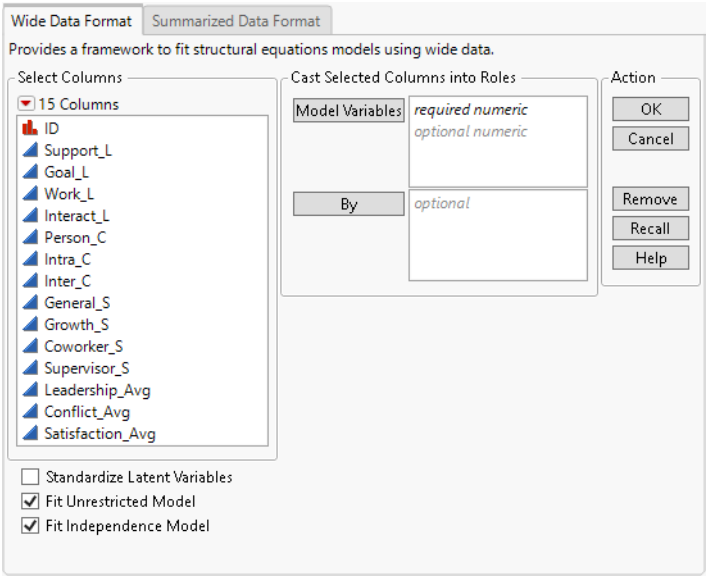


The Normalized Residuals Heat Map has no values that exceed 2.0 units in the positive or negative direction; this is further evidence that the model fits the data well. The residuals can also diagnose ill-fitting models at a more granular level. The normalized residuals in this model do not show evidence of local misfit.

JMP PRO Launch the Structural Equation Models Platform

Launch the Structural Equation Models platform by selecting **Analyze > Multivariate Methods > Structural Equation Models**.

Figure 8.4 Structural Equation Models Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Launch Window includes tabs for two different data formats.

Wide Data Format Select for data tables where each row corresponds to a single observation and the columns contain variables to be used in the model. Rows that contain only missing values are excluded from the analysis.

Summarized Data Format Select for data tables where the data are summarized as a correlation or covariance matrix. Means and standard deviations can also be specified as columns. If they are not specified, the means are assumed to be zero and the standard deviations are assumed to be the square root of the diagonal of the input matrix.

JMP PRO Launch Window Options

Model Variables The columns that you want to include in the model. You must specify at least one column. All columns must have numeric data type and continuous modeling type.

For the Summarized Data Format, the Model Variables columns are the columns that contain the correlation or covariance matrix for the summarized data.

By (Available only for Wide Data Format.) A column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Caution: Using a By variable does not perform a multiple-group analysis.

Mean (Available only for Summarized Data Format.) A column of means that correspond to the variables in each row of the correlation or covariance matrix. If they are not specified, the means are assumed to be zero.

Std Dev (Available only for Summarized Data Format.) A column of standard deviations that correspond to the variables in each row of the correlation or covariance matrix. If they are not specified, the standard deviations are assumed to be the square root of the diagonal of the input matrix.

Label (Available only for Summarized Data Format.) A column of labels that correspond to the variables in each row of the correlation or covariance matrix. If they are not specified, the variables in the Model Specification use the names of the columns that contain the input matrix.

Standardize Latent Variables If selected, this option sets the scale of the latent variables by fixing their variance to one and allowing free estimation of all loadings.

Fit Unrestricted Model If selected, the unrestricted model is automatically fit when you fit your first model. You can then compare your fitted model to the unrestricted model in the Model Comparison report. The unrestricted model is a fully saturated model, which fits all means, variances, and covariances of the specified Model Variables without imposing any structure on the data.

Fit Independence Model If selected, the independence model is automatically fit when you fit your first model. You can then compare your fitted model to the independence model in the Model Comparison report. The independence model fits all means and variances of the specified Model Variables. All covariances among the specified Model Variables are fixed to zero, which leads to a highly restrictive model.

Sample Size (Available only for Summarized Data Format.) Specifies the number of observations represented by the summarized data.

JMP PRO The Structural Equation Models Report

- "Model Specification Report"
- "Model Comparison Report"
- "Chi-Square Difference Test Report"
- "Structural Equation Model Fit Report"

JMP PRO Model Specification Report

The Model Specification report contains controls for specifying your model. When you click OK in the launch window, the default model of independence appears in the Diagram and Lists tabs of the Model Specification report. The independence model includes all means and variances of the specified Model Variables.

JMP PRO Specification Panel

The Specification panel contains controls for building your model.

Model Name Enables you to specify a name for the model. When you click Run in the Action panel, a model report that uses the text in Model Name is created.

From List Lists the model variables that were specified in the launch window as well as a Constant term. The Constant term estimates means of any observed or latent variable by regressing each variable on the Constant. Select one or more variables in this list and one or more variables in the To List before clicking one of the arrow buttons to add terms to the model. If you create a latent variable, the latent variable is added to the From List and the To List. You can filter the items in this list using the search filter controls at the top of the list. See "Search Filter Options".

To List Lists the model variables that were specified in the launch window as well as a Constant term. The Constant term estimates means of any observed or latent variable by regressing each variable on the Constant. Select one or more variables in the From List and one or more variables in this list before clicking one of the arrow buttons to add terms to the model. If you create a latent variable, the latent variable is added to the From List and the To List. Variables in the To List must be selected to add latent variables to the model. You can filter the items in this list using the search filter controls at the top of the list. See "Search Filter Options".

Unidirectional Arrow → Specifies the type of relationship between the selected variables in the From List and the To List. The unidirectional arrow is equivalent to a regression effect.

Bidirectional Arrow ↔ Specifies the type of relationship between the selected variables in the From List and the To List. The bidirectional arrow is equivalent to a covariance effect.

Add Latent + Adds a latent variable to the From and To lists. The variable is named based on the text in the box to the left of the + button.

Note: You must select indicators for the latent variable in the To List prior to clicking the + button.

Remove Latent − Removes any latent variables that are currently selected in either the From List or the To List. Removing a latent variable from the From and To lists also removes the latent variable from the model.

Model Shortcuts Contains three categories of options to quickly build common models. Note that when you select one of these options, all current model specifications are cleared.

Specification Assist Contains options that shortcut the process of adding covariances for either exogenous or endogenous variables.

Cross-Sectional Classics Contains options for confirmatory factor analysis and mediation analysis.

Longitudinal Analysis Contains options for common latent growth curve models: intercept-only, linear, quadratic, and latent basis growth curve. Additionally, the Fit and Compare Growth Models option enables you to fit no growth, linear growth, and quadratic growth models in one click.

Note: If variables in the To List are selected when you select one of the latent growth curve options, the model is applied only to the selected variables. Otherwise, the model is applied to all of the variables in the To List.

JMP PRO Action Panel

The buttons in the Action panel operate on the lists of specified model variables in the Lists tab. The following operations are available:

Fix To Enables you to fix the parameter value of the currently selected effect to a constant value. When you fix the parameter value for a variable, the fixed value appears in parentheses after the name of the term.

Note: If you keep the default value of 0 when you click OK, the selected effects are removed from the model.

Set Equal Enables you to constrain the parameter values of two or more selected effects to be equal. When you set two or more effects to have equal parameter values, an arbitrary alphanumeric label ("c1") appears in parentheses after the names of the terms. In the case of multiple sets of effects that have parameter values that are set equal to each other, sequential numbering is used ("c1", "c2", and so on).

Note: Equality constraints are allowed only within the same type of parameters.

Free Enables you to remove restrictions on the selected effects in the model. Restrictions include effects that have been set to a fixed value or effects that have been set equal to other effects.

Remove Removes the selected effects from the model.

Undo Undoes the last modification to the model.

Redo Redoes the last undone modification to the model.

Run Fits and creates a report for the currently specified model.

Reset Resets the model specification to the independence model, which is the default.

JMP PRO Diagram Panel

The Diagram panel contains buttons that enable you to adjust the layout of the model diagram. The first button enables you to rotate the manifest variables (represented by rectangles) in the diagram. The second button enables you to cycle through two different arrangements of the diagram. The third button enables you to customize the diagram. See "Customize the SEM Path Diagram".

JMP PRO Details Panel

The Details panel contains information about the model that is currently specified in the Model Specification report. Information about the model includes the number of manifest variables, number of latent variables, number of freely estimated parameters, and number of degrees of freedom. You can also adjust the maximum number of iterations.

JMP PRO Diagram Tab

The Diagram tab contains a model diagram that enables you to visualize the currently specified model. Latent variables are represented by circles, and manifest variables are represented by rectangles. Unidirectional arrows represent loadings and regressions. Bidirectional arrows represent variances and covariances. A variance is specified by a bidirectional arrow from one variable to itself. A covariance is specified by a bidirectional arrow between two variables.

The path diagram is interactive, so you can drag items to arrange them. You can also use the arrow keys to move selected items in the path diagram. You can use the buttons in the Diagram panel to rearrange items in the path diagram. You can use the options in the path diagram pop-up menu to add or remove items in the path diagram. The path diagram pop-up menu appears when you right-click in the path diagram itself. The contents of the pop-up menu differs based on whether you click on part of the blank canvas or on an element of the path diagram. You can also zoom in or out in the path diagram by adjusting the zoom setting in the upper left corner of the path diagram. The path diagram also contains a Lock button in the top left corner that enables you to lock the positions of the items in the path diagram. When the path diagram is locked, the Lock button is blue. After making a selection in the path diagram, you can hide the selection by selecting the **Selection > Hide Selection** option in the pop-up menu. When items in the path diagram are hidden, an Unhide All button appears at the top of the path diagram that enables you to show all hidden items.

See "Customize the SEM Path Diagram" for more information about path diagram customization options. See "Diagram Panel", "Details Panel", and "Path Diagram Settings".

Tip: To copy the path diagram as an image, right-click the diagram and select **Copy Diagram**. To retain the highest possible quality, paste the clipboard image as a vector graphic.

JMP PRO Lists Tab

The Lists tab contains lists for each type of variable in the model. They are categorized by the arrows that are used to represent them in the model diagram. Unidirectional arrows are used to designate means or intercepts, loadings, and regressions. Bidirectional arrows are used to designate variances and covariances. Means and intercepts for the model appear only in the Lists tab and not in the Diagram tab.

You can filter the items in each list using the search filter controls at the top of each list. See "Search Filter Options".

JMP PRO Status Tab

The Status tab contains checks for model identifiability. Structural equation models must be identified to obtain reliable estimates. Identification in the context of SEM means that one can derive unique solutions for every parameter in the model based on the population covariance matrix of the input data. Because you can specify such a variety of models in the SEM framework, there are a variety of rules for identifiability. The Status tab consists of three panels of information: Identification Rules, Model Details, and Data Details.

- The Identification Rules panel contains a list of up to eight rules that are appropriate for a specified model. Some rules are necessary and others are sufficient to guarantee model identification. If necessary rules fail, steps must be taken to correct them before fitting the model. If sufficient rules fail, you do not necessarily need to address any issues. Sometimes sufficient rules can fail without being evidence of anything wrong with the model. Note that all of the rules assume a positive definite covariance matrix. If the covariance matrix is not positive definite, a warning appears below the Model Details panel.

  Tip: For more information about a particular identification rule, click on the row of the table for the rule and a description of the rule appears to the right of the table.

- The Model Details panel contains a list of descriptive values for the currently specified model. These values can be used to detect potential issues with the model.
- The Data Details panel contains a list of descriptive values for the input data. These values can be used to detect potential issues with the data.

If there are singularities in the specified data columns, the report contains a Singularity Details table.

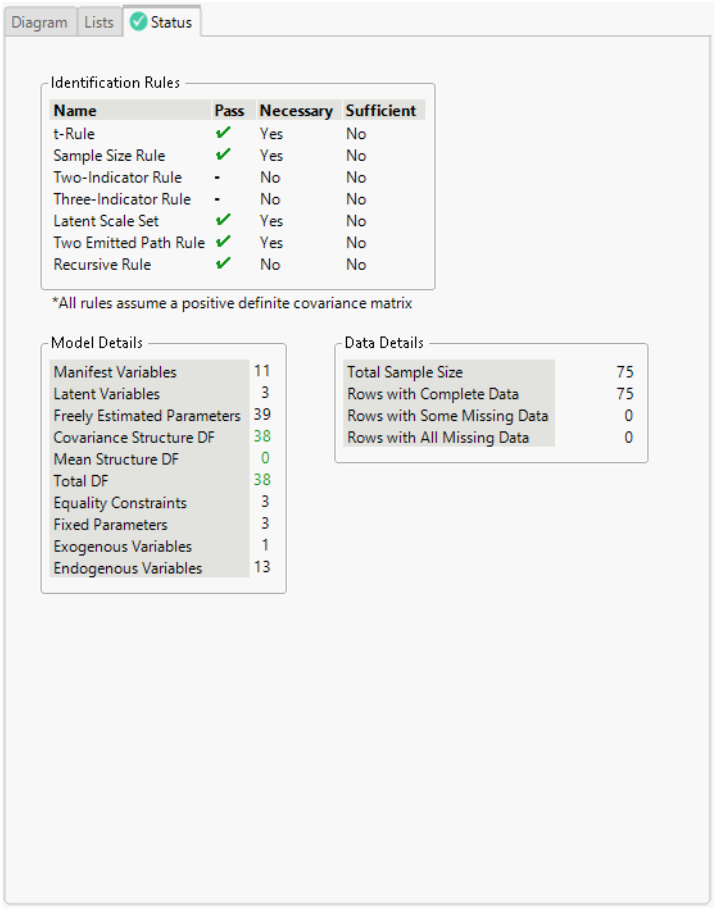
The Status tab itself contains a dynamic icon that shows the current status of the specified model. The icon designates the status:

All applicable identification rules pass.

At least one necessary identification rule does not pass and you must take steps to correct the issues before fitting the model.

At least one non-necessary identification rule does not pass and further investigation is needed to determine whether the model is correctly specified. Oftentimes, advanced applications of SEM lead to this situation; this does not suggest a problem with the model. Rather, it suggests that the identification rules cannot guarantee that the model is identified.

Figure 8.5 Example Status Tab



JMP PRO Search Filter Options

The From List, To List, and the list boxes that appear in the Lists tab in the Model Specification report contain search filters that enable you to filter the items in a specific list.

Click the down arrow button next to the search box to refine your search.

Contains Terms Returns items that contain a part of the search criteria. A search for “ease oom” returns messages such as “Release Zoom”.

Contains Phrase Returns items that contain the exact search criteria. A search for “text box” returns entries that contain “text” followed directly by “box” (for example, “Context Box” and “Text Box”).

Starts With Phrase Returns items that start with the search criteria.

Ends With Phrase Returns items that end with the search criteria.

Whole Phrase Returns items that consist of the entire string. A search for "text box" returns entries that contain only "text box".

Regular Expression Enables you to use the wildcard (*) and period (.) in the search box. Searching for "get.*name" looks for items that contain "get" followed by one or more words. It returns "Get Color Theme Names", "Get Name Info", and "Get Effect Names", and so on.

Invert Result Returns items that do not match the search criteria.

Match All Terms Returns items that contain both strings. A search for "t test" returns elements that contain either or both of the search strings: "Pat Test", "Shortest Edit Script" and "Paired t test".

Ignore Case Ignores the case in the search criteria.

Match Whole Words Returns items that contain each word in the string based on the Match All Terms setting. If you search for "data filter", and Match All Terms is selected, entries that contain both "data" and "filter" are returned.

JMP PRO Model Comparison Report

The Model Comparison report contains a table of all the models that have been fit. Use the icons in the second column to control which model reports appear below the Model Comparison report. The rest of the columns to the right of Model Name enable you to compare models based on various criteria.

The AICc Weight value for a model can be interpreted as the probability that a particular model is the true model given that one of the fitted models is the truth. Therefore, the model with the AICc weight closest to one is the best fit. The AICc weights are calculated using only nonmissing AICc values and are defined as follows:

AICcWeight = exp[-0.5(AICc-min(AICc))] / sum(exp[-0.5(AICc-min(AICc))])

where min(AICc) is the smallest AICc value among the fitted models.

For information about the other criteria in the Model Comparison report, see "Structural Equation Model Fit Report".

Note: If a model does not converge, an asterisk appears at the beginning of the model name in the Model Comparison report.

To provide context for the performance of the fitted models, the following two models are shown by default in the Model Comparison report:

Unrestricted Fits all means, variances, and covariances of the specified Model Variables without imposing any structure on the data.

Independence Fits all means and variances of the specified Model Variables and fixes all covariances to zero.

Model Comparison Report Options

Two options appear below the Model Comparison table. These options are available after you click on rows to highlight them in the Model Comparison table.

Compare Selected Models (Available only when two or more rows of the table are highlighted.) Computes nested chi-square difference tests between all nested model combinations of the selected rows.

Tip: If any of the models that you select are not nested, a warning appears and the non-nested combination of models does not appear in the Chi-Square Difference Test report.

Clear Selection (Available only when one or more rows of the table are highlighted.) Clears all selections from the rows of the table.

JMP PRO Chi-Square Difference Test Report

The Chi-Square Difference Test report contains a table of nested chi-square tests. The table contains two columns that define the models. The first column contains the more constrained of the two models, and the second column contains the less constrained of the two models. The smaller model is nested within the larger model. The remaining columns show the differences in chi-square values, degrees of freedom, CFI, and RMSEA, as well as the *p*-values for the nested chi-square tests. The Δ notation in the column names indicates differences. A significant Δ chi-square value indicates that the additional constraints in the nested model produce a statistically significant increase in misfit and that the less constrained model should be retained. Because chi-square tests are influenced by sample size, such that they are more likely to be significant as sample sizes increase, the ΔCFI and ΔRMSEA should also be considered; ideally, the ΔCFI should not exceed –0.01 and ΔRMSEA should not exceed 0.015 (Chen 2007).

Caution: Difference tests in this report are meaningful only for nested models.

You can remove any row of the report by clicking the red X button. To remove the report entirely, you must remove all of the rows of the table. If you remove a Structural Equation Model node in the main report, any difference tests involving that model are removed from the table.

JMP PRO Structural Equation Model Fit Report

Each time you click Run, a Structural Equation Model report for the specified model appears. By default, this report contains a Summary of Fit report, a Parameter Estimates report, and a Path Diagram.

Summary of Fit Table of information about the model fit, including the convergence status and the estimation method. The following statistics are reported in this table:

Sample Size The number of observations (rows) used to fit the model.

Rows with Missing The number of observations (rows) that contained at least one missing value. All missing values are handled using full information maximum likelihood (Finkbeiner 1979).

Iterations The number of iterations used to fit the model.

-2 Log Likelihood The log-likelihood of the fitted model multiplied by -2. This value can be used to compare nested models; the difference between two models' -2 Log Likelihood values is chi-square distributed with degrees of freedom equal to the difference of degrees of freedom between the models. See *Fitting Linear Models*.

Number of Parameters The number of freely estimated parameters in the model.

AICc The corrected Akaike information criterion. This value can be used to compare models, where a smaller number indicates a better model fit. See "AICc and BIC".

BIC The Bayesian information criterion. This value can be used to compare models, where a smaller number indicates a better model fit. See "AICc and BIC".

ChiSquare The chi-square statistic for the model.

DF The degrees of freedom for the chi-square test for model fit.

Prob>ChiSq The *p*-value of the chi-square statistic for the model.

CFI The Bentler's comparative fit index (CFI) provides additional guidance for determining model fit. The CFI is bounded between 0 and 1. Values greater than 0.90 are preferred (Browne and Cudeck 1993; Hu and Bentler 1999). See "CFI".

RMSEA The root mean square error of approximation (RMSEA) provides additional guidance for determining model fit. The RMSEA is bounded between 0 and 1. Values less than 0.10 are preferred (Browne and Cudeck 1993; Hu and Bentler 1999). See "RMSEA".

Lower 90% The 90% lower confidence limit for the RMSEA. See "RMSEA".

Upper 90% The 90% upper confidence limit for the RMSEA. See "RMSEA".

Parameter Estimates Table of estimates for the model parameters. The table is organized in sections for Means/Intercepts, Loadings, Regressions, and Variances. For each estimate, a standard error (Std Error), Wald test statistic (Wald Z), and a corresponding *p*-value (Prob>|Z|) are given.

Tip: The Parameter Estimates table contains hidden columns that identify the From and To components for each model parameter. To show these columns, right-click the table and select **Columns > From** and **Columns > To**.

Path Diagram Shows the path diagram representation of the fitted model. See "Diagram Tab".

JMP PRO Structural Equation Models Platform Options

The Structural Equation Models red triangle menu contains the following options:

Path Diagram Settings Contains the following options to modify the path diagram display:

Customize Diagram Enables you to customize many aspects of the path diagram. See "Customize Diagram Appearance Options".

Layout Contains two options that change the overall shape of the path diagram. You can choose between a Left to Right layout or a Top to Bottom layout.

Tip: You can also drag items in the path diagram to change the arrangement of specific items.

Copy Diagram Saves an image of the path diagram to the clipboard. To retain the highest possible quality, paste the clipboard image as a vector graphic.

Copy Diagram Properties Copies the current path diagram properties to the clipboard. You can then paste the properties into another SEM path diagram.

Paste Diagram Properties Pastes the path diagram properties from the clipboard into the current SEM path diagram.

Descriptive Statistics Contains the following options to produce descriptive statistics:

Univariate Simple Statistics Shows or hides a univariate simple statistics report. The statistics in this report are estimated for each column independently from other columns that might have missing data.

Full Information Multivariate Statistics Shows or hides a multivariate simple statistics report. The statistics in this report are estimated with full information maximum likelihood to account for missing data.

Launch Explore Outliers Launches the Explore Outliers platform. See *Predictive and Specialized Modeling*.

Launch Explore Missing Values (Not available when there are no missing values.) Launches the Explore Missing Values platform. See *Predictive and Specialized Modeling*.

Add Manifest Variables Enables you to add new manifest variables to the current model. After you select new manifest variables to add, a new Structural Equation Models report is launched. The new report uses the existing model specification with the newly added manifest variables.

Remove Manifest Variables Enables you to launch a new Structural Equation Models report with the specified manifest variables removed. After you select manifest variables to remove, a new report is launched. The new report uses the existing model specification with the specified manifest variables removed.

Copy Model Specification Copies the current structural equation model specifications to the clipboard. You can then paste the model specifications into another SEM platform report.

Paste Model Specification Pastes the model specifications from the clipboard into the current model specifications.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO Model Options

After you click Run, a Structural Equation Model report for the specified model appears. This report has a red triangle menu that contains the following options:

Show Path Diagram Shows or hides the path diagram in the model report.

Path Diagram Settings Contains the following options to modify the path diagram for the model:

Customize Diagram Enables you to customize many aspects of the path diagram. See "Customize Diagram Appearance Options".

Layout Contains two options that change the overall shape of the path diagram. You can choose between a Left to Right layout or a Top to Bottom layout.

Tip: You can also drag items in the path diagram to change the arrangement of specific items.

Copy Diagram Saves an image of the path diagram to the clipboard. To retain the highest possible quality, paste the clipboard image as a vector graphic.

Copy Diagram Properties Copies the current path diagram properties to the clipboard. You can then paste the properties into another SEM path diagram.

Paste Diagram Properties Pastes the path diagram properties from the clipboard into the current SEM path diagram.

Fit Indices Shows or hides a report that contains a variety of index values that enable you to evaluate the fitted model. In addition to values that appear in the Summary of Fit report, the Fit Indices report also contains the following index values:

TLI The Tucker-Lewis index (TLI) provides additional guidance for determining model fit. This index is also known as the non-normed fit index (NNFI). The TLI is bounded between 0 and 1. Values greater than 0.95 are preferred (West et al. 2012). See "TLI".

NFI The normed fit index (NFI) provides additional guidance for determining model fit. The NFI is bounded between 0 and 1. Values greater than 0.95 are preferred (West et al. 2012). See "NFI".

Revised GFI The revised goodness-of-fit index provides additional guidance for determining model fit. The revised GFI is bounded between 0 and 1. Values greater than 0.95 are preferred (West et al. 2012). See "Revised GFI and Revised AGFI".

Revised AGFI The revised adjusted goodness-of-fit index provides additional guidance for determining model fit. The revised AGFI is bounded between 0 and 1 (West et al. 2012). See "Revised GFI and Revised AGFI".

RMR The root mean square residual (RMR) provides additional guidance for determining model fit. The residuals for the RMR are from the differences between the observed and model-implied covariances. The RMR is positive and smaller values are preferred (West et al. 2012). See "RMR and SRMR".

SRMR The standardized root mean square residual (SRMR) provides additional guidance for determining model fit. The residuals for the SRMR are from the standardized differences between the observed and model-implied covariances. The SRMR is positive and smaller values are preferred (West et al. 2012). See "RMR and SRMR".

Note: For a description of the other values in the Fit Indices report, see "Structural Equation Model Fit Report".

Summary of Fit Shows or hides a report that contains details of the model fit.

Parameter Estimates Shows or hides a report that contains the unstandardized parameter estimates for the model.

Standardized Parameter Estimates Shows or hides a report that contains the standardized parameter estimates for the model.

Confidence Intervals Shows or hides confidence intervals in the Parameter Estimates and Standardized Parameter Estimates reports.

Total Effects (Available only when a model contains at least one regression or loading variable and the effects converge.) Shows or hides a table of unstandardized and standardized coefficients of the total effects in the model. Standard errors are also included. The test for convergence of the effects is described in Bentler and Freeman (1983).

Indirect Effects (Available only when a model contains mediating variables and the effects converge.) Shows or hides a table of unstandardized and standardized coefficients of the indirect effects in the model. Standard errors are also included. The test for convergence of the effects is described in Bentler and Freeman (1983).

Prediction Profiler Enables you to view the effects of a set of predictors on the conditional expected values of a set of outcome variables. When you select this option, a window appears in which you must select one or more predictors and one or more outcomes. The predictions and 95% confidence intervals are based on the model-implied covariance matrix. For more information about the prediction profiler, see *Profilers*.

Note: The initial list of variables in the setup window is limited to variables that are consistent with the model. For example, the Select Predictors list contains only variables that predict something in the model and the Select Outcomes list contains only variables that are predicted by some other variable in the model. Check the **Show All Variables** box to see all model variables in both lists.

Model Implied Covariances Shows or hides a report that contains the covariance matrix that is implied by the model.

Model Implied Means Shows or hides a report that contains the means for each variable that are implied by the model.

Residuals Shows or hides a report that contains a matrix of the residuals for the model. This matrix is the difference between the model implied covariance matrix and the sample covariance matrix.

Normalized Residuals Shows or hides a report that contains a matrix of the normalized residuals for the model.

Normalized Residuals Heat Map Shows or hides a report that contains a heat map of the normalized residuals for the model.

RAM Matrices Shows or hides a report that contains the model matrices used in reticular action model (RAM) notation.

Covariance of Estimates Shows or hides a report that contains the covariance matrix of the parameter estimates for the model.

R^2 for Endogenous Variables (Available only when the model is recursive and contains endogenous variables.) Shows or hides a report that contains the R^2 values for each endogenous variable in the model. This value is calculated as 1 minus the ratio of the residual variance and the model-implied variance for each endogenous variable. The R^2 values represent how much variance is explained by the model in an endogenous variable. An endogenous variable is one that has a path directed at it in the path diagram.

Modification Indices Enables you to show all or a subset of the estimates of model modification indices. These values can be used to determine which parameters might be added to the model to improve model fit. Each table is sorted by the ChiSquare column in descending order.

All Modification Indices Shows or hides a table that contains the estimates of all the model modification indices. This table contains a column that indicates the parameter type for each estimate.

Modification Indices for Means Shows or hides a table that contains the estimates of the model modification indices for the means and intercepts.

Modification Indices for Loadings Shows or hides a table that contains the estimates of the model modification indices for the loading parameters.

Modification Indices for Regressions Shows or hides a table that contains the estimates of the model modification indices for the regression parameters.

Modification Indices for Covariances Shows or hides a table that contains the estimates of the model modification indices for the covariance parameters.

Assess Measurement Model (Available only for confirmatory factor models that do not have covariances among unique factors.) Shows or hides a variety of statistics and graphs for quantifying the reliability and validity of tests and measures, including indicator reliability, coefficients omega and H, and a construct validity matrix.

The Indicator Reliability plot shows the squared standardized loadings of the latent variables along with a suggested minimum threshold for acceptable reliability (0.25). Low values for a variable indicate that the variable does not do a very good job of capturing variability in the corresponding latent variable.

The Composite Reliability and Construct Maximal Reliability reports show the coefficients Omega (McDonald 1999) and H (Hancock and Mueller 2001), respectively, for each latent variable. These values range from 0 to 1, and it is recommended that they are about 0.70 or greater. Omega represents the proportion of variance of the latent variable(s) in the observed composite score(s). H represents the proportion of latent variable variance represented by the indicators. These estimates are model-dependent; if a one factor model is fit, the resulting Omega is known as general omega. If a factor model with more than one latent variable is fit, the resulting omega estimates are known as subscale omegas. However, if a bi-factor model is fit, the omega for the general factor estimate is known as hierarchical omega, whereas the group factors are known as hierarchical subscale omegas (Rodriguez et al. 2015). The suggested thresholds should be used in the context of the goals for the survey; if you plan to use composite scores to make decisions about individuals, then reliability should be higher than the suggested threshold (around 0.90 or greater) but if you plan to use composite scores for research purposes, then the lower end of the threshold is acceptable (Nunnally 1978).

The Construct Validity Matrix report helps you determine whether latent variables are measuring what you think they are measuring:

- The lower triangular entries contain the latent variable correlations. These entries enable you to check how strongly correlated the latent variables are with each other and compare that to the hypothesized strength of correlation.
- The upper triangular entries are the squared latent variable correlations. These entries enable you to focus on the overlap in variance across latent variables. These statistics are particularly valuable when compared against the diagonal entries in the matrix.

– The diagonal entries contain the average amount of variance extracted by each latent variable, which is equivalent to the average of the indicator reliabilities for each latent variable. A good latent variable should have high values in the diagonal because its indicators have sufficient systematic variance to define it properly. Ideally, the diagonal entry for each latent variable should be higher than the entries above and to the right of it.

The visualization of the construct validity matrix enables you to compare the diagonal entries to the upper triangular entries.

See "Example of Using the Assess Measurement Model Report".

Save Columns Enables you to save columns based on the fitted structural equation model to the data table.

Save Factor Scores (Available only when there are latent variables in the model.) Saves a column with the factor score computed using the regression method for each latent variable to the data table. The factor scores are calculated in a hidden column that is also added to the data table. This hidden column uses the `Estimate Factor Score()` JSL function. For more information about this function, see Help > Scripting Index.

Save Bartlett Factor Scores (Available only when there are latent variables in the model.) Saves a column with the factor score computed using the Bartlett method for each latent variable to the data table. The factor scores are calculated in a hidden column that is also added to the data table. This hidden column uses the `Estimate Bartlett Factor Score()` JSL function. For more information about this function, see Help > Scripting Index.

Save Prediction Formulas (Available only when there is at least one endogenous or dependent variable in the model.) Saves a column with a formula for the predicted values of the observed outcomes for each variable to the data table. When there are latent variables in the model, factor scores computed using the Bartlett method are also saved to the data table.

Save Observational Residuals (Available only when there is at least one endogenous or dependent variable in the model.) Saves a column with the residual values of the observed outcomes for each variable to the data table. When there are latent variables in the model, factor scores computed using the Bartlett method are also saved to the data table.

Copy Model Specification Copies the current structural equation model specifications to the clipboard. You can then paste the model specifications into another SEM platform report.

Recall in Model Specification Sets the model in the Model Specification report to the specified model.

Remove Fit Removes the specified model report from the report window.

JMP PRO Customize the SEM Path Diagram

You can customize the path diagram in many ways. Most of the customization options are in one or both of the following places: the pop-up menu in the path diagram and the Customize Diagram Appearance window.

- "Path Diagram Pop-Up Menu Options"
- "Customize Diagram Appearance Options"

JMP PRO Path Diagram Pop-Up Menu Options

The path diagram pop-up menu appears when you right-click in the path diagram itself. The specific set of options that are available in the menu depends on where you right-click in the diagram and if there are any elements of the path diagram selected when you right-click.

Selection Contains options to make selections in the From and To Lists, add unidirectional and bidirectional arrows to the diagram, create new latent variables, hide selected items, and show all hidden items. Many of the options in this submenu are not enabled until you select items in the path diagram.

Tip: When items in the path diagram are hidden, an Unhide All button appears at the top of the path diagram that enables you to show all hidden items.

Show Contains options to show or hide various elements of the path diagram. Select **Show Default** to return to the original set of shown elements.

Note: The Show Default option does not affect the setting of the Show Means/Intercepts option.

For path diagrams in fitted model reports, this submenu also contains options for changing which estimates appear on the arrows in the path diagram. You can choose between showing the unstandardized parameter estimates, the standardized parameter estimates, or no estimates.

Select All of This Type (Available only when you right-click an item in the path diagram.) Selects all other items in the path diagram that are the same type as the item that you right-clicked.

Select Latent Group(s) (Available only when you right-click on a latent variable item in the path diagram.) Selects the entire group of items associated with the selected latent variables. If no latent variables are selected, this option selects only the group of items associated with the latent variable that you right-clicked.

Rename Variable (Available only when you right-click on a latent variable item in the path diagram.) Enables you to change the name of the latent variable in the path diagram.

Note: When you rename a latent variable in the path diagram, the name is updated in the From and To Lists.

Add Regression(s) (Available only when you right-click on a variable in the path diagram.) Enables you to add a unidirectional arrow that represents a regression between the variable that you right-clicked and another variable in the path diagram. When you select this option, click the variable where you want the arrow to point.

Add Covariance(s) (Available only when you right-click on a variable in the path diagram.) Enables you to add a bidirectional arrow that represents a covariance between the variable that you right-clicked and another variable in the path diagram. When you select this option, click the variable where you want the arrow to point.

Customize Diagram Launches the Customize Diagram Appearance window. See "Customize Diagram Appearance Options".

Rotate Latent Groups Rotates latent variable indicators clockwise in 90-degree angle increments for the selected latent variables in the path diagram. If no latent variables are selected, this option rotates all latent variable indicators in the path diagram.

Layout Enables you to select one of two arrangement options for the items in the path diagram.

Copy Diagram Saves an image of the path diagram to the clipboard. To retain the highest possible quality, paste the clipboard image as a vector graphic.

Copy Diagram Properties Copies the current path diagram properties to the clipboard. You can then paste the properties into another SEM path diagram.

Paste Diagram Properties Pastes the path diagram properties from the clipboard into the current SEM path diagram.

Undo Undoes the last change to the path diagram.

Redo Redoes the last undone change to the path diagram.

Edit Contains standard options for graphs in JMP. For more information about these options, see *Using JMP*.

Reset Layout Resets the path diagram to the original settings.

JMP PRO Customize Diagram Appearance Options

The Customize Diagram Appearance window appears when you click the customize button in the Diagram panel, select Customize Diagram from the red triangle menu, or select Customize Diagram from the path diagram pop-up menu. The Customize Diagram Appearance window contains four panels as well as a drop-down menu that enables you to select from two preset color themes.

Theme Preset Contains options for preset color themes. You can choose between a Black and White theme or a Blue theme. Select one of these to quickly update the settings to follow either theme.

Variable Appearance Contains options for setting the fill color, border color, text color, and size of the items in the path diagram. You can set the appearance for manifest, latent, and constant variables, as well as the font for each of these items.

Path Settings Contains options for the appearance of the arrows. You can set the thickness and transparency to fixed values or base them on the standardized estimates for each path arrow. You can also represent non-significant *p*-values with dashed lines, set the α level for non-significance, and change the color and font associated with the arrows. By default, the thickness of the arrows is fixed and the transparency of the arrows is proportional to the standardized parameter estimates.

Other Settings Contains the following options:

Enable Grid Shows or hides a grid in the path diagram that can be used to help align items in the diagram.

Lock Diagram Locks or unlocks the placement of items in the path diagram. When the path diagram is locked, you cannot drag items around and the Lock indicator at the top of the path diagram is highlighted.

Show Means/Intercepts Shows or hides the means and intercepts in the path diagram. The mean structure is not displayed in the path diagram by default.

Show Latent Indicators Shows or hides the latent indicators in the path diagram.

Show Regressions Shows or hides the arrows for regressions in the path diagram.

Show Variances Shows or hides the arrows for variances in the path diagram.

Show Covariances Shows or hides the arrows for covariances in the path diagram.

Show Equality Constraints Shows or hides equality constraints in the path diagram.

Show R Squared Values Shows or hides R^2 values for each node in the path diagram.

Fill Nodes with R Squared Shows or hides a fill coloring for each node in the path diagram that corresponds to the R^2 value for each node.

Preview Contains a preview of the path diagram appearance based on your current settings in the window.

Tip: Use the **Save to Preferences** button at the bottom of the Customize Diagram Appearance window to save the current settings in the window to your platform preferences so that future path diagrams use the same settings.

JMP PRO Additional Examples of Structural Equation Models

- "Example of a Latent Path Variable Model"
- "Example of a Latent Growth Curve Model"
- "Example of Using the Assess Measurement Model Report"

Example of a Latent Path Variable Model

In this example, you are building the structural equation model for industrialization and political democracy described in Bollen (1989), which uses data from 75 developing countries. The variables in the data table include four measures of democracy in 1960 and 1965, and three measures of industrialization in 1960. These variables are described in the Notes column property in each column of the data table. To view the Notes column property, right-click a column name, select Column Info, and select Notes under Column Properties. The type of structural equation model that you are building is a structural regression model.

There are four main steps to the model specification process: creating latent variables, adding the loading and regression variables, adding the covariance terms, and placing constraints on the loading variables.

1. Select **Help > Sample Data Library** and open Political Democracy.jmp.
2. Select **Analyze > Multivariate Methods > Structural Equation Models**.
3. Select Prod60 through Legis65 and click **Model Variables**.
4. Click **OK**.

   The Structural Equation Models report Model Specification outline appears.
5. Click the **Lists** tab in the View panel box.

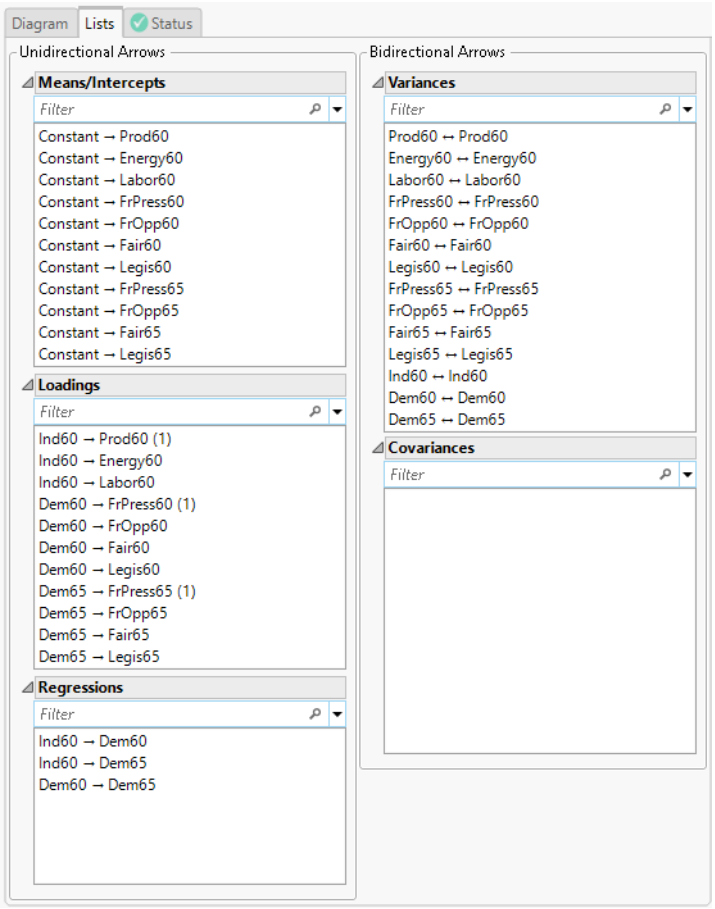
Create Latent Variables

6. Select Prod60 through Labor60 in the To List, type Ind60 in the box below the To List, and click the add latent button.
7. Select FrPress60 through Legis60 in the To List, type Dem60 in the box below the To List, and click the add latent button.
8. Select FrPress65 through Legis65 in the To List, type Dem65 in the box below the To List, and click the add latent button.

Add Loading and Regression Variables

9. Select Ind60 in the From List, select Dem60 in the To List, and click the unidirectional arrow button.
10. Select Ind60 in the From List, select Dem65 in the To List, and click the unidirectional arrow button.
11. Select Dem60 in the From List, select Dem65 in the To List, and click the unidirectional arrow button.

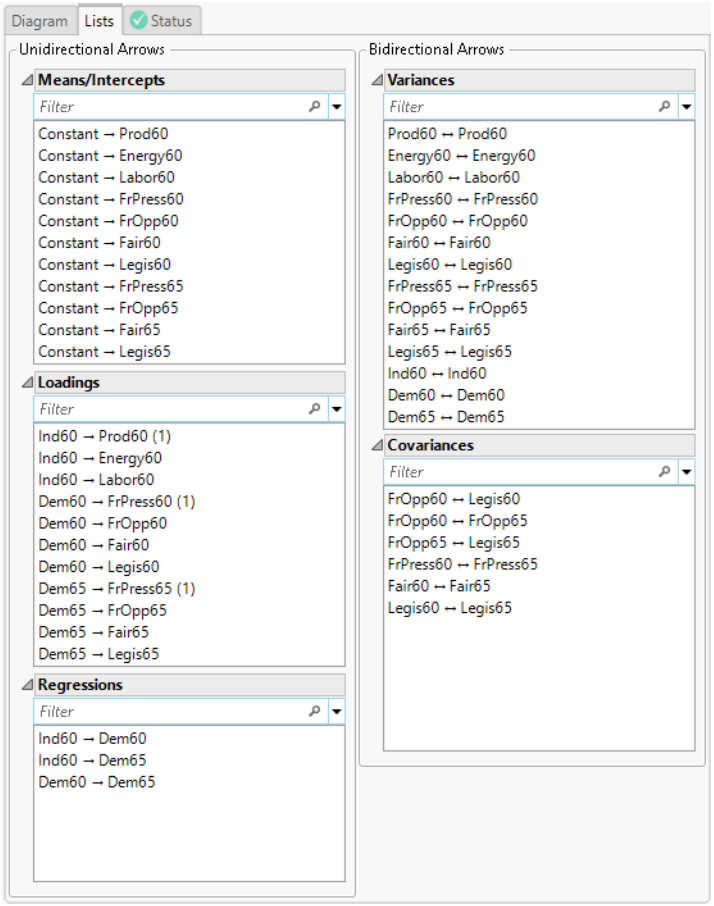
Figure 8.6 Loadings and Regressions



Add Covariances

12. Select FrOpp60 in the From List, select Legis60 and FrOpp65 in the To List, and click the bidirectional arrow ↔ button.
13. Select FrOpp65 in the From List, select Legis65 in the To List, and click the bidirectional arrow ↔ button.
14. Select FrPress60 in the From List, select FrPress65 in the To List, and click the bidirectional arrow ↔ button.
15. Select Fair60 in the From List, select Fair65 in the To List, and click the bidirectional arrow ↔ button.
16. Select Legis60 in the From List, select Legis65 in the To List, and click the bidirectional arrow ↔ button.

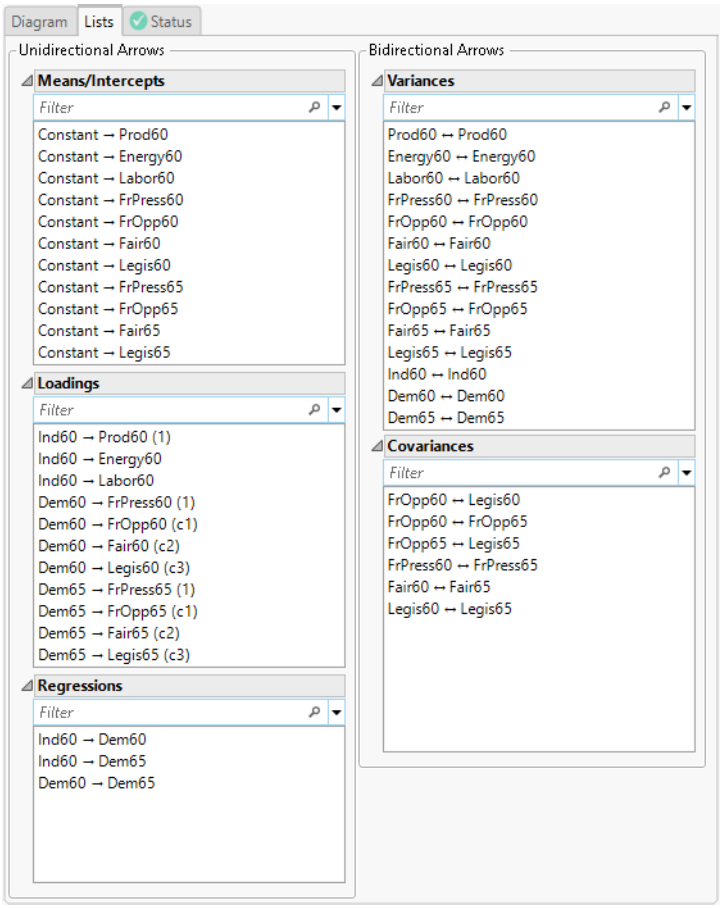
Figure 8.7 Covariances



Add Constraints on Loadings

17. Select Dem60->FrOpp60 and Dem65->FrOpp65 in the Loadings list and click **Set Equal**.
18. Select Dem60->Fair60 and Dem65->Fair65 in the Loadings list and click **Set Equal**.
19. Select Dem60->Legis60 and Dem65->Legis65 in the Loadings list and click **Set Equal**.

Figure 8.8 Completed Model Specification



The constraints on the loadings are designated by alphanumeric labels. For example, you can see that Dem60->FrOpp60 and Dem65->FrOpp65 are set equal because they both are labeled "c1".

20. In the text box below Model Name, type Industrialization and Political Democracy.
21. Click **Run**.

Figure 8.9 Structural Equation Model Summary of Fit Report

Summary of Fit

Converged in Gradient
Maximum Likelihood

Sample Size	75
Rows with Missing	0
Iterations	7
-2 Log Likelihood	3097.6362
Number of Parameters	39
AICc	3264.779
BIC	3266.0182
ChiSquare	40.179489
DF	38
Prob>ChiSq	0.3738824
CFI	0.9967743
RMSEA	0.0276538
Lower 90%	0
Upper 90%	0.0870678

The chi-square statistic for this model, which is listed in the Summary of Fit report, is 40.18 with 38 degrees of freedom. Note that the corresponding p-value is 0.3739, which is not significant. This indicates that there is not evidence to reject the null hypothesis that the model fits well. Therefore, you conclude that this model fits the data reasonably well.

The chi-square value depends on the sample size, and thus, some well-fitting models can still produce a significant chi-square value. The comparative fit index (CFI) and root mean square error of approximation (RMSEA) provide additional guidance for determining model fit. These indices are bounded between 0 and 1. CFI values greater than 0.90 and RMSEA values less than 0.10 are preferred (Browne and Cudeck 1993; Hu and Bentler 1999). Here, the CFI of 0.9968 and RMSEA of 0.0277 indicate a very good fit.

Figure 8.10 Structural Equation Model Parameter Estimates Report

Parameter Estimates

Means/Intercepts	Estimate	Std Error	Wald Z	Prob>|Z|
Constant → Prod60	5.0543838	0.0840624	60.126559	<.0001*
Constant → Energy60	4.7921946	0.1732697	27.657432	<.0001*
Constant → Labor60	3.5576898	0.1612318	22.065682	<.0001*
Constant → FrPress60	5.4646667	0.2989114	18.281892	<.0001*
Constant → FrOpp60	4.2564429	0.4389904	9.6959807	<.0001*
Constant → Fair60	6.5631103	0.3939809	16.658446	<.0001*
Constant → Legis60	4.452533	0.3796307	11.728591	<.0001*
Constant → FrPress65	5.1362519	0.3044457	16.870832	<.0001*
Constant → FrOpp65	2.9780741	0.3923504	7.5903426	<.0001*
Constant → Fair65	6.1962639	0.3643999	17.004021	<.0001*
Constant → Legis65	4.0433897	0.3753526	10.772243	<.0001*
Loadings	**Estimate**	**Std Error**	**Wald Z**	**Prob>|Z|**
Ind60 → Prod60	1	.	.	.
Ind60 → Energy60	2.1796569	0.1389146	15.690626	<.0001*
Ind60 → Labor60	1.8182095	0.152128	11.951843	<.0001*
Dem60 → FrPress60	1	.	.	.
Dem60 → FrOpp60	1.1907835	0.1416451	8.4068115	<.0001*
Dem60 → Fair60	1.1745402	0.1197987	9.8042797	<.0001*
Dem60 → Legis60	1.2509806	0.1229262	10.176679	<.0001*
Dem65 → FrPress65	1	.	.	.
Dem65 → FrOpp65	1.1907835	0.1416451	8.4068115	<.0001*
Dem65 → Fair65	1.1745402	0.1197987	9.8042797	<.0001*
Dem65 → Legis65	1.2509806	0.1229262	10.176679	<.0001*
Regressions	**Estimate**	**Std Error**	**Wald Z**	**Prob>|Z|**
Ind60 → Dem60	1.471331	0.3914721	3.7584565	0.0002*
Ind60 → Dem65	0.6004747	0.2382837	2.5199993	0.0117*
Dem60 → Dem65	0.8650414	0.0756835	11.429724	<.0001*
Variances	**Estimate**	**Std Error**	**Wald Z**	**Prob>|Z|**
Prod60 ↔ Prod60	0.0813877	0.0196996	4.13144	<.0001*
Energy60 ↔ Energy60	0.1204277	0.0699024	1.7227975	0.0849
Labor60 ↔ Labor60	0.4666599	0.0891233	5.2361183	<.0001*
FrPress60 ↔ FrPress60	1.8546493	0.4569855	4.0584422	<.0001*
FrOpp60 ↔ FrOpp60	7.5813412	1.3449672	5.6368223	<.0001*
Fair60 ↔ Fair60	4.9556729	0.96122	5.1556073	<.0001*
Legis60 ↔ Legis60	3.2244878	0.7417156	4.3473373	<.0001*
FrPress65 ↔ FrPress65	2.3130434	0.4834138	4.7848104	<.0001*
FrOpp65 ↔ FrOpp65	4.9681894	0.8945151	5.554059	<.0001*
Fair65 ↔ Fair65	3.5600372	0.7379206	4.8244178	<.0001*
Legis65 ↔ Legis65	3.3076951	0.7128724	4.6399537	<.0001*
Ind60 ↔ Ind60	0.448599	0.0867473	5.1713335	<.0001*
Dem60 ↔ Dem60	3.8753207	0.8886194	4.3610581	<.0001*
Dem65 ↔ Dem65	0.1644634	0.2333453	0.7048069	0.4809
Covariances	**Estimate**	**Std Error**	**Wald Z**	**Prob>|Z|**
FrOpp60 ↔ Legis60	1.4401257	0.6909886	2.0841528	0.0371*
FrOpp60 ↔ FrOpp65	2.1830097	0.7311078	2.9858931	0.0028*
FrOpp65 ↔ Legis65	1.3717876	0.5781788	2.3726008	0.0177*
FrPress60 ↔ FrPress65	0.5825448	0.3644286	1.5985156	0.1099
Fair60 ↔ Fair65	0.7115828	0.6194143	1.1487994	0.2506
Legis60 ↔ Legis65	0.362804	0.4607898	0.7873525	0.4311

Next, the parameter estimates under Regressions suggest positive effects of Ind60 on Dem60 and Dem65, as well as a positive effect of Dem60 on Dem65. Thus, higher scores on Ind60 are associated with higher Dem60 and Dem65, and higher scores in Dem60 are associated with higher scores in Dem65. The corresponding *p*-values for the parameter estimates are shown under Regressions. All 3 regression parameters are significant at the $\alpha = 0.05$ level. Therefore, you conclude that there are nonzero relationships between the latent variables.

Example of a Latent Growth Curve Model

Latent growth curve (LGC) models enable analysts to model longitudinal data and characterize trajectories over time. The most common latent growth curve model is one of linear growth, where intercept and slope latent variables are specified to capture the overall trajectory of the process and individual observations' deviations from the overall trajectory. This model is very similar to a random coefficients model. A key feature of LGC models is that they place constraints on the mean structure of the data. Modeling the mean structure enables you to test hypotheses related to growth.

In this example, you are modeling the achievement scores of students over 4 years of an academic program. You want to test the hypothesis that a linear trajectory characterizes this process.

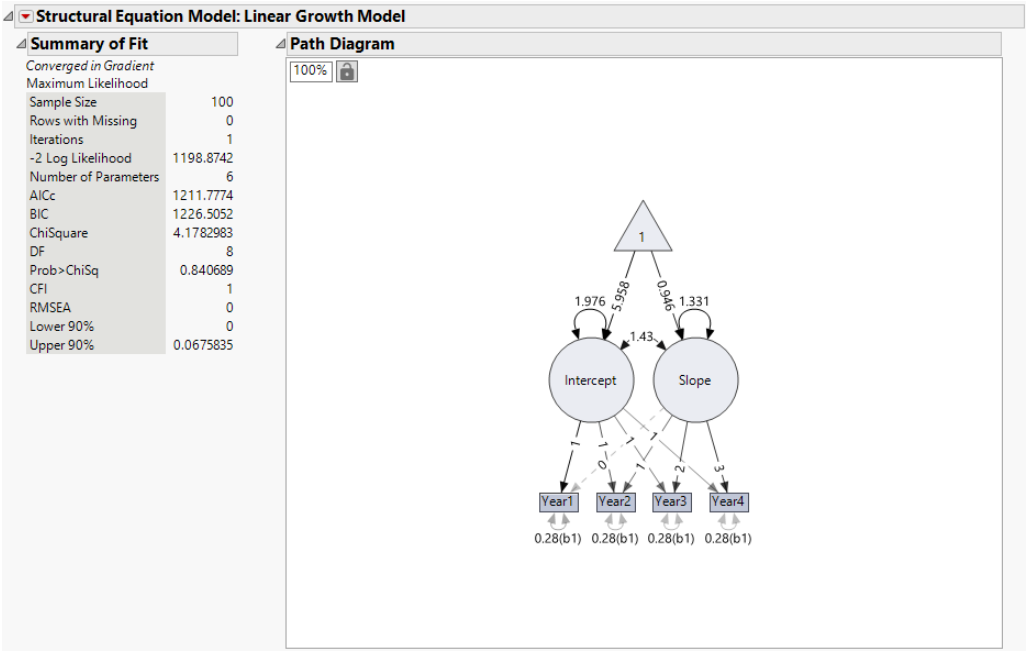
1. Select **Help > Sample Data Library** and open Academic Achievement.jmp.
2. Select **Analyze > Multivariate Methods > Structural Equation Models**.
3. Select Year1 through Year4 and click **Model Variables**.
4. Click **OK**.

   The Structural Equation Models report Model Specification outline appears.
5. Select **Model Shortcuts > Longitudinal Analysis > Linear Latent Growth Curve**.

   The Model Specification diagram now shows a linear latent growth curve model.
6. Click the **Lists** tab in the View panel box.
7. Select the first 4 items in the Variances list and click **Set Equal**.

   This constrains the residual variances for each variable to be equal, which is analogous to a homogeneity of variance assumption. In SEM, this assumption is testable, such that another model can be fit without equality constraints and compared with a chi-square difference test.
8. In the text box below Model Name, type Linear LGC with Equal Variances.
9. Click **Run**.

Figure 8.11 Summary of Fit and Path Diagram for Linear LGC Model



The chi-square statistic for this model, listed in the Summary of Fit report, is 4.18 with 8 degrees of freedom. Note that the corresponding *p*-value is 0.8407, which is not significant. This indicates that there is not evidence to reject the null hypothesis that the model fits well. Therefore, you conclude that this model fits the data reasonably well. The CFI and RMSEA fit indices also indicate excellent fit as they are greater than 0.9 and lower than 0.1, respectively.

Figure 8.12 Parameter Estimates for Linear LGC Model

Structural Equation Model: Linear Growth Model

Parameter Estimates

Means/Intercepts	Estimate	Std Error	Wald Z	Prob>|Z|
Constant → Intercept	5.9579957	0.1473906	40.423176	<.0001*
Constant → Slope	0.9461125	0.1177755	8.0331884	<.0001*
Loadings	**Estimate**	**Std Error**	**Wald Z**	**Prob>|Z|**
Intercept → Year1	1	.	.	.
Intercept → Year2	1	.	.	.
Intercept → Year3	1	.	.	.
Intercept → Year4	1	.	.	.
Slope → Year1	0	.	.	.
Slope → Year2	1	.	.	.
Slope → Year3	2	.	.	.
Slope → Year4	3	.	.	.
Variances	**Estimate**	**Std Error**	**Wald Z**	**Prob>|Z|**
Year1 ↔ Year1	0.2803392	0.0280339	10	<.0001*
Year2 ↔ Year2	0.2803392	0.0280339	10	<.0001*
Year3 ↔ Year3	0.2803392	0.0280339	10	<.0001*
Year4 ↔ Year4	0.2803392	0.0280339	10	<.0001*
Intercept ↔ Intercept	1.9761611	0.3078496	6.4192413	<.0001*
Slope ↔ Slope	1.3310383	0.1962465	6.7824803	<.0001*
Covariances	**Estimate**	**Std Error**	**Wald Z**	**Prob>|Z|**
Intercept ↔ Slope	1.429507	0.2197848	6.5041213	<.0001*

The mean estimate of the intercept is 5.96, which indicates the overall achievement score at year 1. This means that, on average, students' academic achievement is 5.96 on the first occasion of measurement. The mean of the intercept is centered at the first occasion because the slope factor has a zero loading on that variable.

The mean estimate of the slope is 0.95, which indicates that the overall achievement scores increase by 0.95 each year. The significant variance estimates of the intercept and slope factors point to significant variability around the average trajectory; not everyone starts at the same place and not everyone increases at the same rate. External variables can be used as predictors of the intercept and slope to understand what leads to differences in the trajectories of individuals.

Lastly, the positive covariance between intercept and slope suggests that those who start with higher levels of academic achievement on the first year tend to increase more rapidly over time.

LGC models provide a flexible approach for investigating how processes change over time. The Model Shortcuts menu provides alternative trajectories that can be fit and tested against each other to ascertain the best model fit, including a no-growth trajectory, a quadratic trajectory, and even a non-linear trajectory with the Latent Basis option.

Example of Using the Assess Measurement Model Report

A confirmatory factor analysis model (CFA) enables you to test alternative measurement models. The Assess Measurement Model report provides tools for quantifying the reliability and validity of tests and measures. The results include indicator reliability, coefficients Omega and H, and a construct validity matrix.

In this example, you are assessing the validity and reliability of a consumer data survey. You fit a confirmatory factor analysis model for five latent variables: Privacy, Security, Reputation, Trust, and Purchase Intent. You then use the Assess Measurement Model option to test the survey reliability.

1. Select **Help > Sample Data Library** and open Online Consumer Data.jmp.
2. Click the green triangle next to the **SEM: CFA** script.

   The script runs a confirmatory factor analysis model for the survey data.
3. Click the gray disclosure icons next to Model Specification and Model Comparison to hide those parts of the report.

Figure 8.13 Summary of Fit for CFA Model

Structural Equation Model: CFA

Summary of Fit

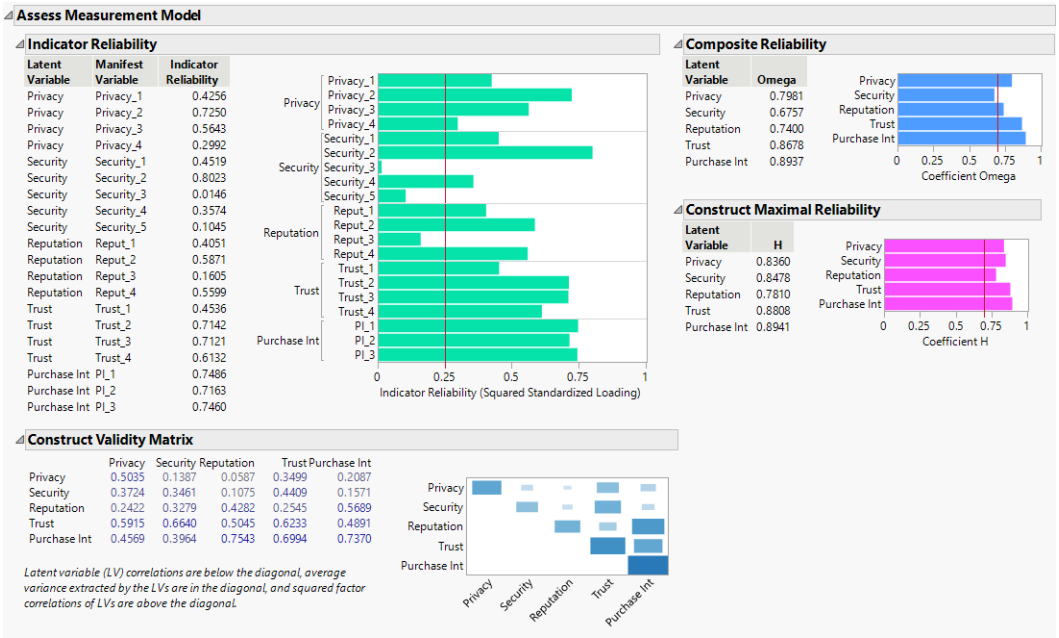
Converged in Gradient
Maximum Likelihood

Sample Size	843
Rows with Missing	0
Iterations	7
-2 Log Likelihood	45106.777
Number of Parameters	70
AICc	45259.653
BIC	45578.365
ChiSquare	808.10917
DF	160
Prob>ChiSq	3.738e-87
CFI	0.9160319
RMSEA	0.0693187
Lower 90%	0.0646117
Upper 90%	0.0741036

The chi-square statistic for this model, listed in the Summary of Fit report, is 808.11 with 160 degrees of freedom. Note that the corresponding *p*-value is significant. This indicates that there is some evidence to reject the null hypothesis that the model fits well. However, the chi-square statistic is strongly influenced by sample size and has been shown to be significant even when the model provides good fit to the data in samples of over 200 to 300 observations. In this example, the sample size is 843. Thus, the fit of the model should also be assessed with the CFI and RMSEA fit indices. In this example, both indicate good fit as they are greater than 0.9 and lower than 0.1, respectively.

4. Click the red triangle next to Structural Equation Model: CFA and select **Assess Measurement Model**.

Figure 8.14 Assess Measurement Model Report for CFA Model



The Indicator Reliability plot shows the squared standardized loadings of the latent variables along with a suggested minimum threshold for acceptable reliability (0.25). You can see that two of the Security-related questions and one of the Reputation-related questions do not do a very good job of capturing variability in the corresponding latent variable. The Security_3 question has such a low value that it could be removed without nearly any loss to the reliability of the construct, whereas the Security_5 and Reput_3 questions could be revised in an attempt to improve their reliability.

The Composite Reliability and Construct Maximal Reliability reports show the coefficients Omega and H, respectively, for each latent variable. These values range from 0 to 1, and it is recommended that they are about 0.70 or greater. Omega represents the proportion of variance of the latent variable(s) in the observed composite score(s). H represents the proportion of latent variable variance represented by the indicators. By these measures, the Omega coefficient for Security is just below the suggested threshold of 0.70, but it is close enough to deem the composite reliable for research purposes. The suggested thresholds should be used in the context of the goals for the survey; if you plan to use composite scores to make decisions about individuals, then reliability should be higher than the suggested threshold (around 0.90 or greater) but if you plan to use composite scores for research purposes, then the lower end of the threshold is acceptable (Nunnally 1978). Composite reliability for Security and Reputation could be improved by focusing on improving the indicator reliability for the Security_3, Security_5, and Reput_4 questions.

The Construct Validity Matrix report provides a way to determine whether latent variables are measuring what you think they are measuring. Notice in the matrix visualization that the diagonal values for Privacy, Trust, and Purchase Intent are larger than the values above and to the right of them. However, this is not the case for Security and Reputation. This is further evidence that the Security-related and Reputation-related questions can be improved to measure the Security and Reputation latent variables.

You conclude that the survey could be improved by removing or revising some of the questions related to Security and Reputation.

For more information about the components of the report, see "Assess Measurement Model".

JMP PRO Statistical Details for the Structural Equation Models Platform

JMP PRO Statistical Details for Summary of Fit Measures

JMP PRO AICc and BIC

The formulas for AICc and BIC are defined as follows:

$$\text{AICc} = \text{-2logL} + 2k + \frac{2k(k+1)}{n-(k+1)}$$

$$\text{BIC} = \text{-2logL} + k\ln(n)$$

where:

- -2logL is twice the negative log-likelihood.
- n is the sample size.
- k is the number of parameters.

For more information about the likelihood-based measures in the Model Comparisons report, see *Fitting Linear Models*.

JMP PRO CFI

The comparative fit index (CFI) is calculated as follows:

$$\text{CFI} = \frac{max(\chi_0^2 - df_0, 0) - max(\chi_{min}^2 - df_{min}, 0)}{max(\chi_0^2 - df_0, 0)}$$

where:

- χ_0^2 is the chi-square statistic of the independence model.
- df_0 is the degrees of freedom of the independence model.
- χ_{min}^2 is the chi-square statistic of the fitted model.
- df_{min} is the degrees of freedom of the fitted model.

For more information about the CFI, see Bentler (1990).

JMP PRO TLI

The Tucker-Lewis index (TLI) is defined as follows:

$$\text{TLI} = \frac{\frac{\chi_0^2}{df_0} - \frac{\chi_{min}^2}{df_{min}}}{\frac{\chi_0^2}{df_0} - 1}$$

where:

- χ_0^2 is the chi-square statistic of the independence model.
- df_0 is the degrees of freedom of the independence model.
- χ_{min}^2 is the chi-square statistic of the fitted model.
- df_{min} is the degrees of freedom of the fitted model.

For more information, see West et al. (2012).

JMP PRO NFI

The Bentler-Bonett normed fit index (NFI) is defined as follows:

$$\text{NFI} = \frac{\chi_0^2 - \chi_{min}^2}{\chi_0^2}$$

where:

- χ_0^2 is the chi-square statistic of the independence model.
- χ_{min}^2 is the chi-square statistic of the fitted model.

For more information, see West et al. (2012).

JMP PRO Revised GFI and Revised AGFI

The revised goodness-of-fit index (Revised GFI) is defined as follows:

$$\text{Revised GFI} = \frac{p}{p + 2\left(\frac{\chi_{min}^2 - df_{min}}{n-1}\right)}$$

where:

- χ_{min}^2 is the chi-square statistic of the fitted model.
- df_{min} is the degrees of freedom of the fitted model.
- p is number of observed variables in the fitted model.
- n is the sample size.

The revised adjusted goodness-of-fit index (Revised AGFI) is defined as follows:

$$\text{Revised AGFI} = 1 - \frac{p^*}{df_{min}}(1 - RevisedGFI)$$

where:

- p^* is the number of unique entries in the covariance matrix and the mean vector of the observed variables.
- df_{min} is the degrees of freedom of the fitted model.

For more information, see West et al. (2012).

JMP PRO RMSEA

The root mean square error of approximation (RMSEA) is calculated as follows:

$$\text{RMSEA} = \sqrt{\frac{\max(\chi^2_{min} - df_{min}, 0)}{n \times df_{min}}}$$

where:

- n is the sample size.
- df_{min} is the degrees of freedom of the fitted model.
- χ^2_{min} is the chi-square statistic of the fitted model.

The confidence limits for RMSEA are computed using the cumulative distribution function of the noncentral chi-square distribution $\Phi(x \mid \lambda, d)$. The 90% confidence limits are computed as follows:

$$\text{Lower limit} = \sqrt{\frac{\lambda_L}{n \times df_{min}}}$$

$$\text{Upper limit} = \sqrt{\frac{\lambda_U}{n \times df_{min}}}$$

where:

- λ_{L} satisfies $\Phi(\chi^2_{min} \mid \lambda_{\text{L}}, df_{\text{min}}) = 0.95$.
- λ_{U} satisfies $\Phi(\chi^2_{min} \mid \lambda_{\text{U}}, df_{\text{min}}) = 0.05$.

For more information, see Maydeu-Olivares et al. (2017).

JMP PRO RMR and SRMR

The formulas for RMR and SRMR are defined as follows:

$$\text{RMR} = \sqrt{\frac{1}{b}\left[\sum_{i}^{p}\sum_{j}^{i}(s_{ij} - \hat{\sigma}_{ij})^2 + \sum_{i}^{p}(\bar{x}_i - \hat{\mu}_i)^2\right]}$$

$$\text{SRMR} = \sqrt{\frac{1}{b}\left[\sum_{i}^{p}\sum_{j}^{i}\frac{(s_{ij}-\hat{\sigma}_{ij})^2}{s_{ii}s_{jj}}+\sum_{i}^{p}\frac{(\bar{x}_i-\hat{\mu}_i)^2}{s_{ii}}\right]}$$

where:

- p is the number of manifest variables.
- b is the number of unique entries in the covariance matrix and the mean vector of the observed variables:

$$b = \frac{p(p+1)}{2} + p$$

- s_{ij} is the $(i, j)^{\text{th}}$ element of the input covariance matrix.
- $\hat{\sigma}_{ij}$ is the $(i, j)^{\text{th}}$ element of the predicted covariance matrix.
- $\bar{x}_i$ is the i^{th} element of the vector of sample means.
- $\hat{\mu}_i$ is the i^{th} element of the vector predicted means.

For more information, see the CALIS Procedure chapter in SAS Institute Inc. (2020a).

Chapter 9

Factor Analysis

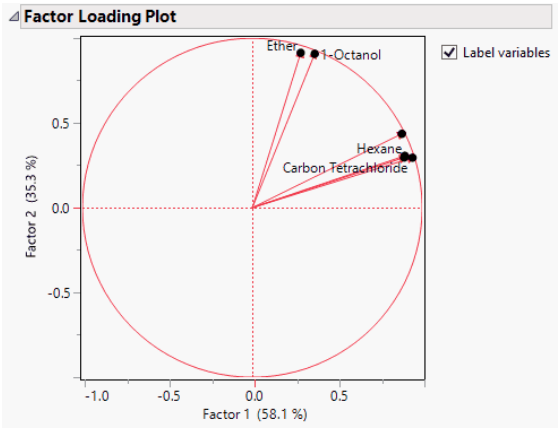
Identify Latent Variables in Your Data

Factor analysis seeks to describe observed variables in terms of a smaller number of (unobservable) latent variables, or factors. Factor analysis is also known as common factor analysis and exploratory factor analysis. The factors are defined as linear combinations of the observed variables (plus error). They are constructed to explain variation that is *common* to the observed variables. A goal of factor analysis is to find a meaningful interpretation of the observed variables in terms of the unobserved factors. Another goal is to reduce the number of variables.

Factor analysis is used in many areas, with roots in psychology, sociology, and education. In these areas, factor analysis is used to understand how observed behavior can be interpreted in terms of underlying patterns and structures. For example, measures of participation in outdoor activities, hobbies, exercise, and travel, might all relate to a factor that can be described as "active versus inactive personality type".

Use factor analysis when you need to explore or interpret underlying patterns and structure in your data. Also consider using it to summarize the information in your variables using a smaller number of latent variables.

Figure 9.1 Rotated Factor Loading



Contents

Overview of the Factor Analysis Platform 227
Example of the Factor Analysis Platform 228
Launch the Factor Analysis Platform 230
The Factor Analysis Report 231
Model Launch 232
Rotation Methods 234
Factor Analysis Platform Options 235
Factor Analysis Model Fit Options 236

Overview of the Factor Analysis Platform

Factor analysis models a set of observable variables in terms of a smaller number of unobservable factors. The factors are constructed to account for the correlation or covariance between the observed variables. Factor rotation is used to change the reference axes of the factors to increase their interpretability.

Consider a situation where you have ten observed variables, X_1, X_2, …, X_{10}. Suppose you want to model these ten variables in terms of two latent factors, F_1 and F_2. For convenience, it is assumed that the factors are uncorrelated and that each has mean zero and variance one. The model that you want to derive is of the form:

$$X_i = \beta_{i0} + \beta_{i1}F_1 + \beta_{i2}F_2 + \varepsilon_i$$

It follows that $Var(X_i) = \beta_{i1}^2 + \beta_{i2}^2 + Var(\varepsilon_i)$. The portion of the variance of X_i that is attributable to the factors, the common variance or *communality*, is $\beta_{i1}^2 + \beta_{i2}^2$. The remaining variance, $Var(\varepsilon_i)$, is the unique variance, and is considered to be a combination of specific and error variances that are unique to X_i.

The platform provides a scree plot for the eigenvalues of the correlation or covariance matrix. You can use this as a guide in determining the number of factors to extract. The platform's default number of factors is the number of eigenvalues that exceed one.

The platform provides two factoring methods for estimating the parameters of this model: Principal Axis and Maximum Likelihood. There are two Prior Communality options for estimating the proportion of variance contributed by common factors for each variable. These options impose assumptions on the diagonal of the correlation (or covariance) matrix. The Principal Components option treats the correlation matrix, which has ones on its diagonal (or the covariance matrix with variances on its diagonal), as the structure to be analyzed. The Common Factor Analysis option sets the diagonal entries to square multiple correlations. These are values that reflect the proportion of the variation shared with other variables.

Factor rotation is used to support interpretability of the extracted factors. The Factor Analysis platform provides a variety of rotation methods that encompass both orthogonal and oblique rotations.

In contrast to factor analysis, which looks at common variance, principal component analysis accounts for the total variance of the observed variables. See "Principal Components".

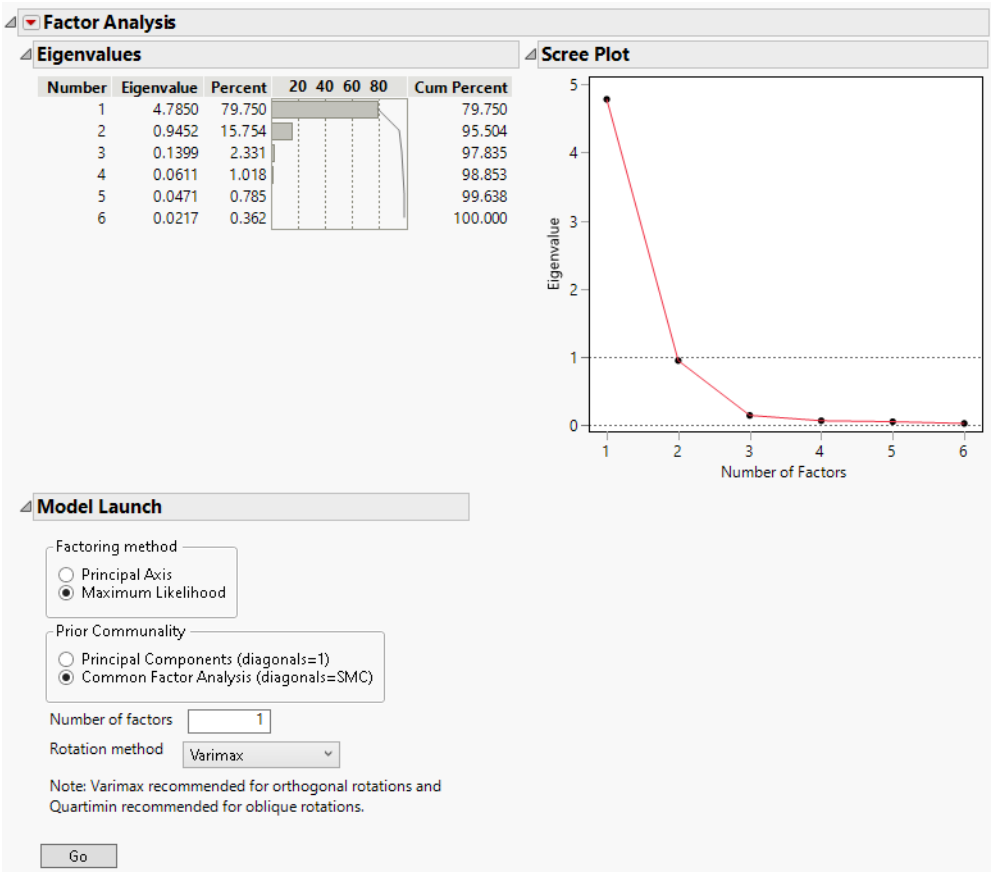
For more information about factor analysis, see Jöreskog (1977) or Cudeck and MacCallum (2007).

Example of the Factor Analysis Platform

This example uses the Solubility.jmp sample data table to extract two factors explained by six solvents.

1. Select **Help > Sample Data Library** and open Solubility.jmp.
2. Select **Analyze > Multivariate Methods > Factor Analysis**.
3. Select 1-Octanol through Hexane and click **Y, Columns**.
4. Click **OK**.

Figure 9.2 Initial Factor Analysis Report



The elbow on the scree plot indicates that a model with two factors is appropriate.

5. In the Model Launch outline, make the following selections:
   - Factoring Method= **Maximum Likelihood**

- Prior Communality= **Common Factor Analysis**
- Number of factors = 2
- Rotation Method= **Varimax**

6. Click **Go**.

Figure 9.3 Factor Analysis Report

Factor Analysis

Eigenvalues **Scree Plot**

Model Launch

Factor Analysis on Correlations with 2 Factors: Maximum Likelihood / Varimax

Final Communality Estimates

Variance Explained by Each Factor

Significance Test

Test	DF	ChiSquare	Prob>ChiSq
H0: no common factors. HA: at least one common factor.	15	691.106	<.0001*

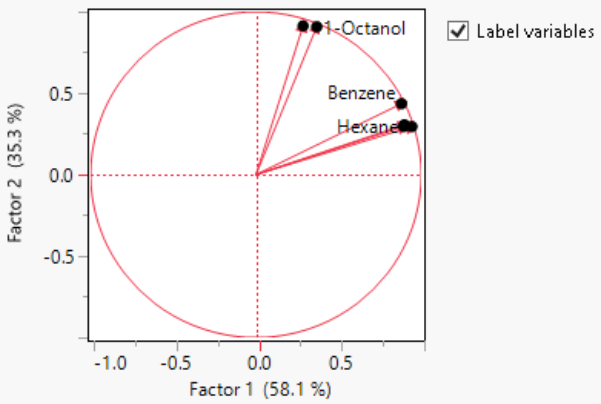
Test	DF	Criterion	ChiSquare	Prob>ChiSq
H0: 2 factors are sufficient. HA: more factors are needed.	4	0.389	26.031	<.0001*

Measures of Fit

Measures of Fit	Fit Index
Chi-Square without Bartlett's Correction	27.654
AIC	19.654
BIC	10.547
Tucker and Lewis's Index	0.878
Root Mean Square Error of Approximation	0.289

Rotated Factor Loading

Factor Loading Plot



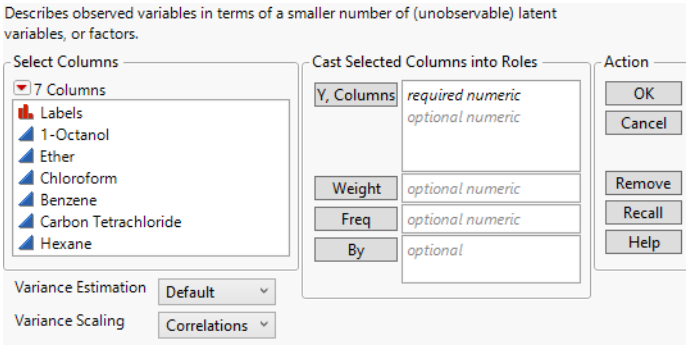
The report lists the communality estimates, variance estimates, significance tests, measures of fit, rotated factor loadings, and a factor loading plot. The Rotated Factor Loadings and Factor Loading Plot suggest that Factor 1 relates to the Carbon Tetrachloride-Chloroform-Benzene-Hexane cluster of variables, and that Factor 2 relates to the Ether–1-Octanol cluster of variables. See "Factor Analysis Model Fit Options" for details of the information shown in the report.

Tip: Click the points in the Factor Loading Plot to select and move the labels. Click in the lower right corner to increase the plot size to more easily view the labels.

Launch the Factor Analysis Platform

Launch the Factor Analysis platform by selecting **Analyze > Multivariate Methods > Factor Analysis**.

Figure 9.4 Factor Analysis Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns The columns to be analyzed. These must have a Numeric data type.

Weight A column containing a weight for each observation in the data table. A row is included in the analyses only when its value is greater than zero.

Freq Assigns a frequency to each row in the analysis. This is useful when your data are summarized.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Variance Estimation Lists the methods for estimating the variance-covariance matrix for the analysis. For more information about the methods, see "The Multivariate Report".

Variance Scaling Lists the scaling methods for performing the factor analysis.

Correlations Default method that enables analysis on correlations.

Covariances Enables the analysis on a weighted correlation matrix where the weights are the variables' variances.

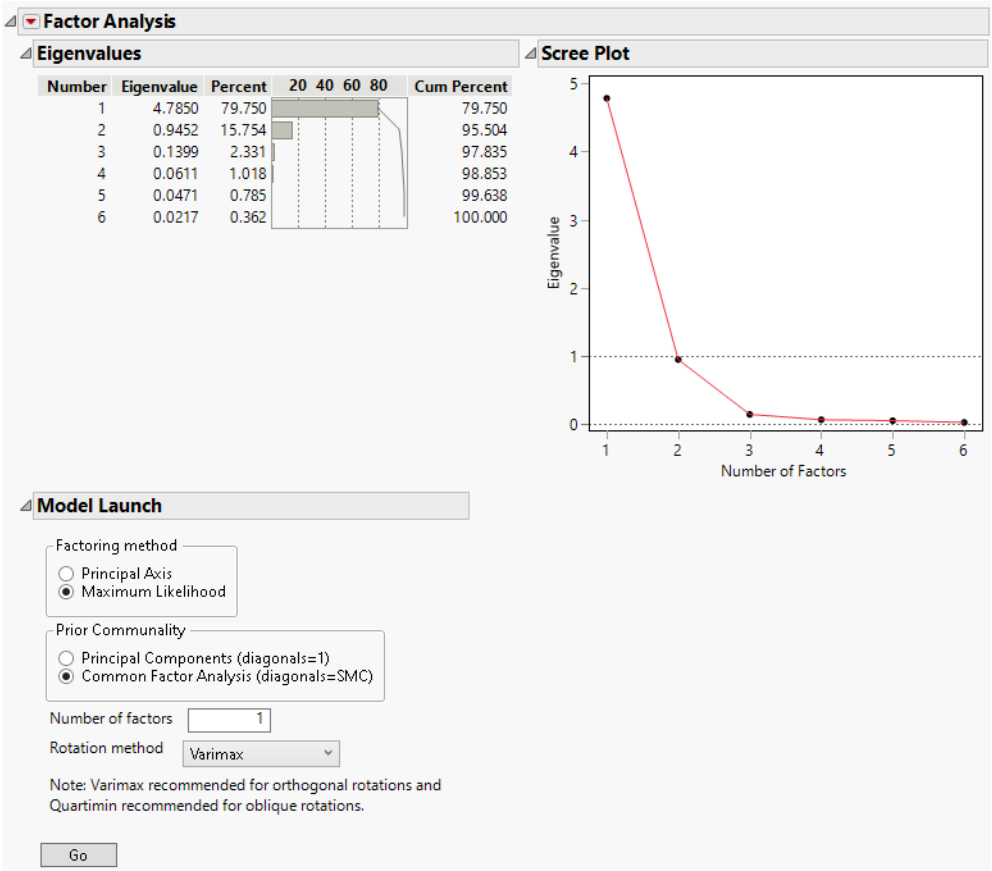
Unscaled Enables the analysis of variables that are already centered or scaled.

The Factor Analysis Report

The initial Factor Analysis report shows Eigenvalues and the Scree Plot. The eigenvalues are obtained from a principal components analysis. The scree plot graphs these eigenvalues. The initial number of factors in the Model Launch equals the number of eigenvalues that exceed 1.0. You can change the number of factors to extract.

Use the scree plot as a guide for the number of factors. The number of eigenvalues before the scree plot levels out provides an upper bound on the number of factors. For this example, two components (factors) are suggested by the scree plot.

Figure 9.5 Factor Analysis Report

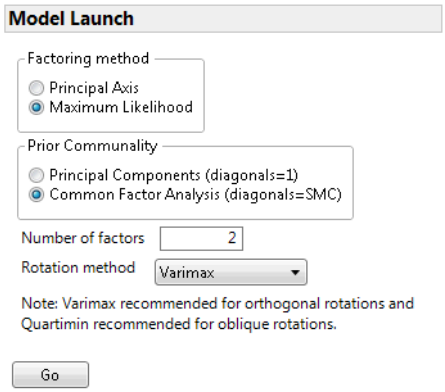


The Eigenvalues table indicates that the first eigenvalue accounts for 79.75% of the variation and the second eigenvalue accounts for 15.75%. Therefore, the first two eigenvalues account for 95.50% of the total variation. The third eigenvalue explains only 2.33% of the variation, and the contributions from the remaining eigenvalues are negligible. Although the **Number of factors** box is initially set to 1, this analysis suggests that it is appropriate to extract 2 factors.

Model Launch

Configure the Factor Analysis from the Model Launch control panel. Click **Go** to obtain the results of the factor analysis.

Figure 9.6 Model Launch



Factoring method Defines the method for extracting factors:

Principal Axis Performs eigenvalue decomposition on a reduced correlation or covariance matrix, where the diagonal of the matrix is replaced by an estimate of the communality of the variables. This method is a computationally efficient method, but it does not allow for hypothesis testing.

Maximum Likelihood Enables you to test hypotheses about the number of common factors and to obtain model fit statistics.

Note: The Maximum Likelihood method requires a positive definite correlation matrix. If your correlation matrix is not positive definite, select the Principal Axis method.

Prior Communality Defines the method for estimating the proportion of variance contributed by common factors for each variable.

Principal Components (diagonals = 1) Sets all communalities equal to 1, indicating that 100% of each variable's variance is explained by all of the factors.

Tip: Using this option with Factoring Method set to Principal Axis results in principal component analysis.

Common Factor Analysis (diagonals = SMC) Sets the communalities equal to squared multiple correlation (SMC) coefficients. For a given variable, the SMC is the RSquare for a regression of that variable on all other variables.

Number of factors Specifies the number of factors to extract from the analysis. The default is the number of eigenvalues that are greater than or equal to 1.0. You can set the number of factors to at least one and no more than the number of variables.

Rotation method Defines the method for rotation. The default is Varimax. See "Rotation Methods" for a description of the available rotation methods.

Rotation Methods

Rotations are used to change the reference axes of the factors to make the factors more interpretable. Rotations are applied to the factors extracted from the data. Rotation methods are based on various complexity or simplicity functions. For more information about rotations see the FACTOR Procedure chapter in SAS Institute Inc. (2020c), Browne (2001), or Frank and Todeschini (1994).

After the initial extraction, the factors are uncorrelated with each other. If the factors are rotated by an orthogonal transformation, the rotated factors are also uncorrelated. If the factors are rotated by an oblique transformation, the rotated factors become correlated. Oblique rotations often produce more interpretable factors than do orthogonal rotations. However, a consequence of correlated factors is that there is no single unambiguous measure of the importance of a factor in explaining a variable.

Orthogonal Rotation Methods

Varimax Maximizes the sum of the variances of the squared loadings of a factor on all variables. This common method results in each variable having either a small or large loading on each factor. (Orthomax with $\gamma = 1$.)

Biquartimax An equally weighted solution of the Varimax and Quartimax rotations. (Orthomax with $\gamma = 0.5$.)

Equamax A weighted solution between the Varimax rotation and the Quartimax rotation. (Orthomax with $\gamma = N/2$, where N = number of factors.)

Factorparsimax A solution that aims to minimize the complexity of factors. This method might result in cross-loadings as variable complexity is not considered in the algorithm. (Orthomax with $\gamma = N$, where N = number of factors.)

Orthomax A general weighted rotation method where the weight is denoted by γ. Many specific orthogonal rotation methods are Orthomax rotations with a specific γ.

Parsimax Balances the variable and the factor complexity. (Orthomax with $\gamma = (I(N-1))/(I+N-2)$, where I = the number of items and N = number of factors.)

Quartimax Minimizes the number of factors needed to explain each variable. (Orthomax with $\gamma = 1$.)

Oblique Rotation Methods

Biquartimin A rotation to minimize the ratio of the covariances (Oblimin with $\tau = 0.5$.).

Covarimin Oblique Varimax rotation. (Oblimin with $\tau = 1.$).

Obbiquartimax Oblique Biquartimax rotation.

Obequamax Oblique Equamax rotation.

Obfactorparsimax Oblique factor Parsimax rotation.

Oblimin A general weighted oblique rotation method where the weight is denoted by τ. Many specific oblique rotation methods are Oblimin rotations with a specific τ.

Obparsimax Oblique Parsimax rotation

Obquartimax Oblique Quartimax rotation, equivalent to the Quartimin method.

Obvarimax Oblique Varimax rotation.

Quartimin Oblique Quartimin rotation, equivalent to oblique Quartimax (Oblimin with $\tau = 0.$)

Promax A two step rotation in which Varimax is performed first and then the Procrustes rotation is used to attain simple structure. This is a computationally efficient method that is an alternative to Oblimin.

Factor Analysis Platform Options

The Factor Analysis red triangle menu contains the following options:

Eigenvalues Shows or hides a table of the eigenvalues of the original correlation, covariance, or unscaled matrix. The table includes the percent of the total variance represented by each eigenvalue, a bar chart illustrating the percent contribution, and the cumulative percent contributed by each successive eigenvalue. The number of eigenvalues that are greater than or equal to 1.0 can be taken as a guideline of the number of factors for analysis.

Scree Plot Shows or hides a plot of the eigenvalues versus the number of components (or factors). The plot can be used as an additional guideline to determine the number of factors that contribute to the maximum amount of variance. The point at which the plotted line levels out can be used as the number of sufficient factors for analysis.

Kaiser-Meyer-Olkin Test Shows or hides the results of the Kaiser-Meyer-Olkin (KMO) test. The test is an indicator of the proportion of variance that might be common variance, potentially due to underlying factors. The test statistic is calculated for each individual variable and for the set of all variables and is referred to as the Measure of Sampling Adequacy (MSA) in JMP. The following values indicate the suitability of the variables for factor analysis:

- 0.00 to 0.49 unacceptable
- 0.50 to 0.59 miserable
- 0.60 to 0.69 mediocre
- 0.70 to 0.79 middling
- 0.80 to 0.89 meritorious
- 0.90 to 1.00 marvelous

Note: The KMO test is not available if the correlation matrix is singular.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Factor Analysis Model Fit Options

The Factor Analysis Model Fit red triangle menu contains the following options:

Prior Communality (Available only for Common Factor Analysis.) Shows or hides an initial estimate of the communality for each variable. For a given variable, this estimate is the squared multiple correlation coefficient (SMC), or RSquare, for a regression of that variable on all other variables.

Eigenvalues (Available only for Common Factor Analysis.) Shows or hides the eigenvalues of the reduced correlation matrix and the percent of the common variance for which they account. The reduced correlation matrix is the correlation matrix with its diagonal entries replaced by the communality estimates. The eigenvalues indicate the common variance explained by the factors. The Cum Percent can exceed 100% because the reduced correlation matrix is not necessarily positive definite and can have negative eigenvalues.

Note the table indicates the number of factors retained for analysis.

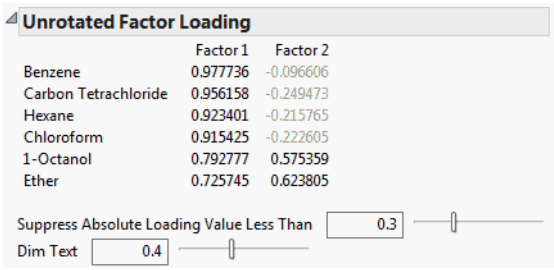
Unsorted and Unrotated Factor Loading Shows or hides the factor loading matrix before sorting and rotation.

Unrotated Factor Loading Shows or hides the factor loading matrix before rotation. Factor loadings measure the influence of a common factor on a variable. Because the unrotated factors are orthogonal, the factor loading matrix is the matrix of correlations between the variables and the factors. The closer the absolute value of a loading is to 1, the stronger the effect of the factor on the variable.

Use the slider and value to **Suppress Absolute Loading Values Less Than** the specified value in the table. Suppressed values appear dimmed according to the setting specified by **Dim Text**.

Use the **Dim Text** slider and value to control the table's font transparency gradient for factor values less in absolute value than the specified **Suppress Absolute Loading Values Less Than** value.

Figure 9.7 Unrotated Factor Loading with Dim Text Controls



Note: The Unrotated Factor Loading matrix is ordered so that variables associated with the same factor appear next to each other.

Rotation Matrix Shows or hides the values used for rotating the factor loading plot and the factor loading matrix.

Interfactor Correlations (Available only for oblique rotations.) Shows or hides the matrix of correlations between factors.

Target Matrix (Available only for the Promax rotation.) Shows or hides the matrix to which the varimax factor pattern is rotated.

Factor Structure (Available only for oblique rotations.) Shows or hides the matrix of correlations between variables and common factors.

Final Communality Estimates Shows or hides estimates of the communalities after the factor model has been fit. When the factors are orthogonal, the final communality estimate for a variable equals the sum of the squared loadings for that variable.

Standard Score Coefficients Shows or hides a table of the multipliers used to estimate factor scores when saving rotated factors to the source data table.

Variance Explained by Each Factor (Available only for orthogonal rotations.) Shows or hides the variance, percent, and cumulative percent, of common variance explained by each rotated factor.

Variance Explained by Each Factor Ignoring Other Factors (Available only for oblique rotations.) Shows or hides the variance and percent of common variance explained by each rotated factor regardless of other factors.

Significance Test (Available only for the Maximum Likelihood factoring method.) Provides the results of two Chi-square tests.

The first test is for H_0: No common factors. This null hypothesis indicates that none of the common factors explain the intercorrelations among the variables. This test is Bartlett's Test for Sphericity, whose null hypothesis is that the correlation matrix of the factors is an identity matrix (Bartlett, 1954).

The second test is for H_0: N factors are sufficient, where N is the specified number of factors. Rejection of this null hypothesis indicates that more factors might be required to explain the intercorrelations among the variables (Bartlett, 1954). The Criterion is the log-likelihood objective function value.

Measures of Fit (Available only for the Maximum Likelihood factoring method.) Shows or hides measures of fit: Chi-Squared without Bartlett's Correction, AIC, BIC, Tucker-Lewis's Index, and the root mean square error of approximation.

Measures of Factor Scores Shows or hides measures of factor score determinacy: Multiple R, Multiple R Square, and Minimum Correlation. These measures are used to evaluate if the factor scores might be useful for secondary analyses.

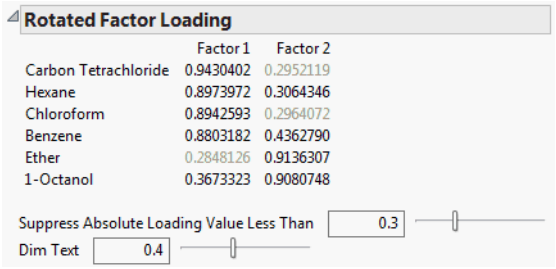
Unsorted and Rotated Factor Loading Shows or hides the unsorted factor loading matrix after rotation.

Rotated Factor Loading Shows or hides the factor loading matrix after rotation. If the rotation is orthogonal, these values are the correlations between the variables and the rotated factors.

Use the slider and value to **Suppress Absolute Loading Values Less Than** the specified value in the table. Suppressed values appear dimmed according to the setting specified by **Dim Text**.

Use the **Dim Text** slider and value to control the table's font transparency gradient. The lower the value, the more transparency the font for factor values less in absolute value than the specified **Suppress Absolute Loading Values Less Than** value.

Figure 9.8 Rotated Factor Loading with Dim Text Controls



Note: The Rotated Factor Loading matrix is ordered so that variables associated with the same factor appear next to each other.

Factor Loading Plot Shows or hides a plot of the rotated factor loadings. When more than 2 factors are modeled, the loading plot is a matrix of plots.

Score Plot Shows or hides a scatterplot of the estimated factor scores. When more than 2 factors are modeled, the score plot is a matrix of plots.

Score Plot with Imputation (Available only if there are missing values.) Shows or hides a scatterplot of the estimated factor scores with imputed values for missing values.

Display Options Enables you to show or hide arrows on the loading plots.

Save Rotated Factors Saves the rotated factor scores and formulas to the data table.

Note: The formula cannot evaluate rows with missing values.

Remove Fit Removes the fit model results from the Factor Analysis report. This option enables you to change the Model Launch configuration for a new report.

Chapter 10

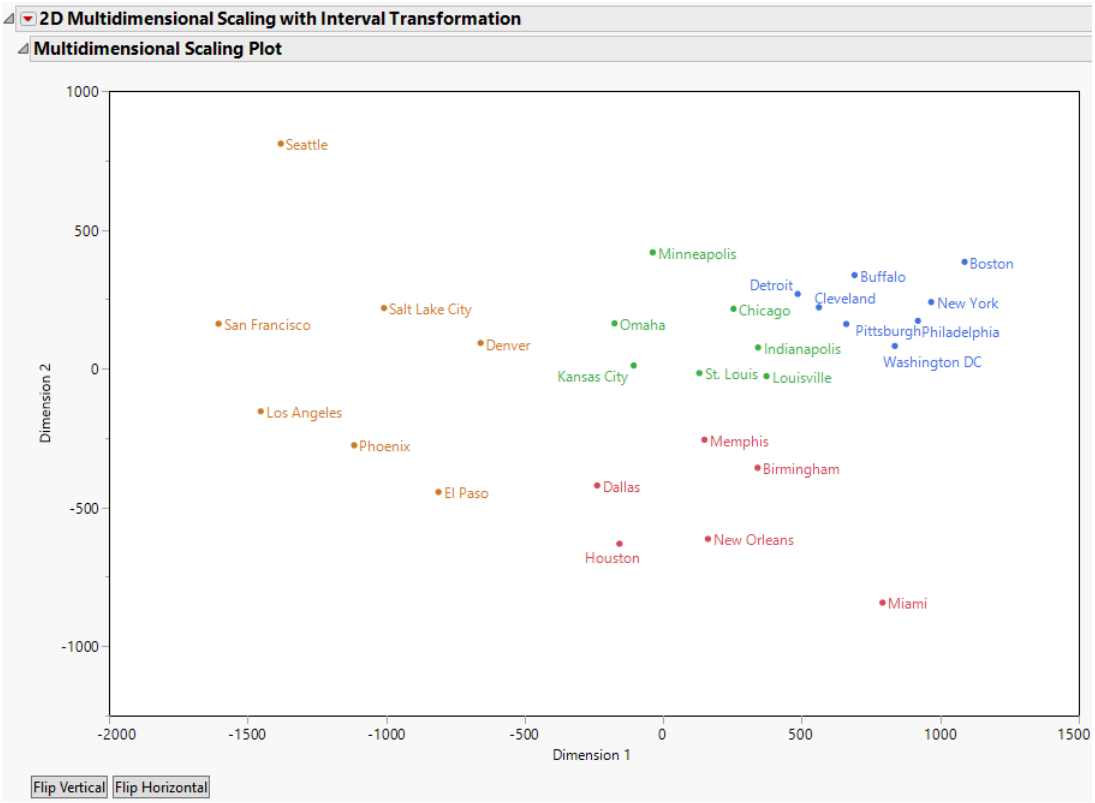
Multidimensional Scaling

Visualize Proximities among a Set of Objects

Multidimensional Scaling (MDS) is a technique that is used to create a visual representation of the pattern of proximities (similarities, dissimilarities, or distances) among a set of objects. For example, given a matrix of distances between cities, MDS can be used to generate a map of the cities in two dimensions.

Multidimensional Scaling is frequently used in consumer research where researchers have measures of perceptions about brands, tastes, or other product attributes. MDS is applicable to many other areas where one is interested in visualizing the proximity of objects based on a set of attributes or proximities.

Figure 10.1 Multidimensional Scaling Example



Contents

Overview of the Multidimensional Scaling Platform 243
Example of Multidimensional Scaling 243
Launch the Multidimensional Scaling Platform 246
The Multidimensional Scaling Report 247
Multidimensional Scaling Plot 247
Shepard Diagram 248
Fit Details 248
Multidimensional Scaling Platform Options 248
Waern Links 249
Additional Example of the Multidimensional Scaling Platform 251
Statistical Details for the Multidimensional Scaling Platform 253
Stress 253
Transformations 254
Attributes List Format 254

Overview of the Multidimensional Scaling Platform

The Multidimensional Scaling platform generates a plot of proximities among a set of objects. This plot can be used to visually explore structure in a data set. MDS is a multivariate technique that is used to visualize the patterns of proximities (distances, similarities) among a set of objects in a small number of dimensions. MDS is applied to a distance matrix. The coordinates for the MDS plot are obtained by minimizing a stress function (the difference between the actual and predicted proximities).

The term distance can refer to a measure of physical distance, such as between cities. More often distance is a subjective assessment rather than a precise measurement. Proximities can measure perceived similarities between brands of a product, correlations of crime rates, or economic similarities for a sample of countries. Distance can also be called proximity or similarity (dissimilarity). If the data are given as an attribute list, then a distance matrix is first constructed from the attribute list.

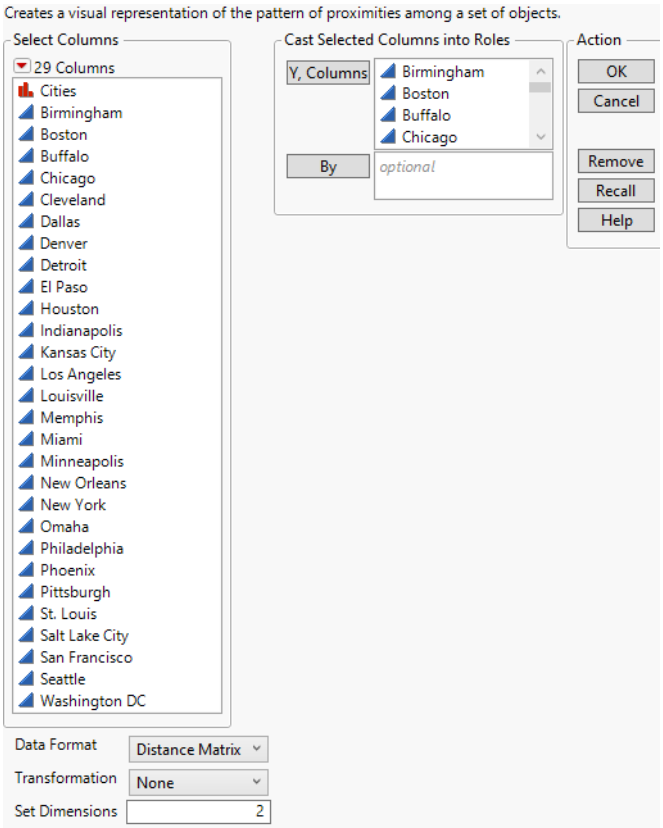
For more information about multidimensional scaling, see Borg and Groenen (2005) or Jackson (2003).

Example of Multidimensional Scaling

This example uses the Flight Distances.jmp sample data table, which is a distance matrix of flight distances between 28 US cities. You can use MDS to construct a map of the cities in two dimensions that is based on the pairwise distances in the data table.

1. Select **Help > Sample Data Library** and open Flight Distances.jmp.
2. Select **Analyze > Multivariate Methods > Multidimensional Scaling.**
3. Select Birmingham through Washington DC and click **Y, Columns.**

Figure 10.2 Completed Multidimensional Scaling Launch Window



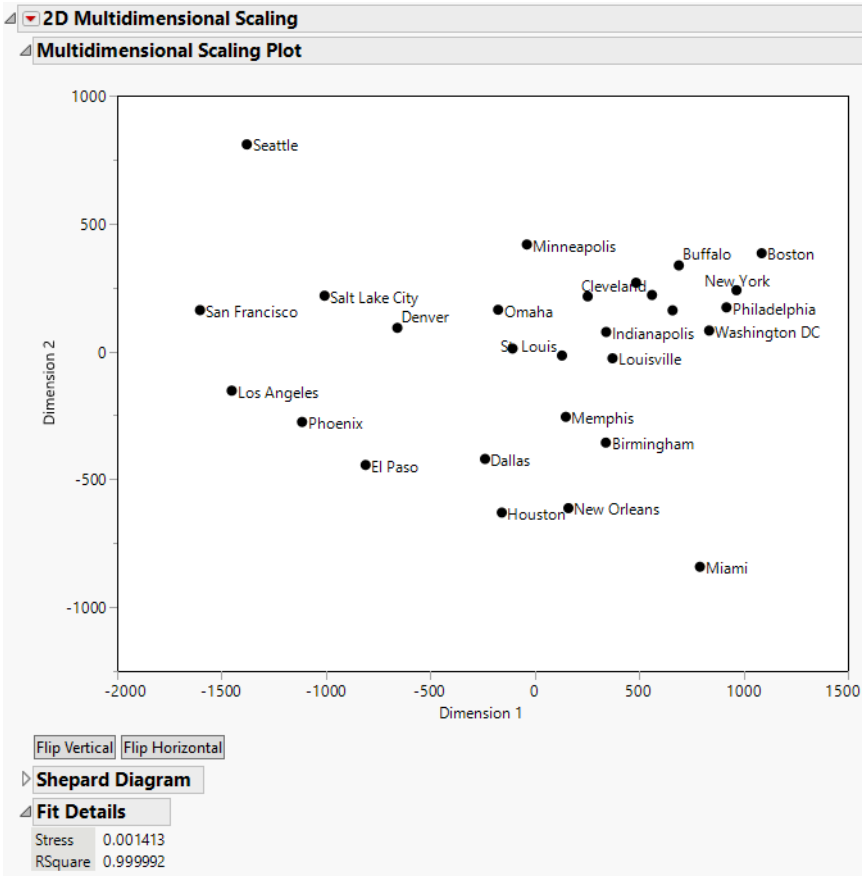
4. Click **OK.**

   In the Multidimensional Scaling Plot, hover over data points to view the row number or row label. The next 7 steps are to label and rotate your MDS Plot.

5. Select the Flight Distances data table.
6. Right-click the column Cities and select **Label/Unlabel**.
7. Select **Rows > Row Selection> Select all Rows**.
8. Select **Rows > Label/Unlabel**.
9. Select the Multidimensional Scaling Plot.
10. Click the **Flip Vertical** button.
11. Click the **Flip Horizontal** button.

    The Flip Vertical and Flip Horizontal buttons enable you to change the orientation of the MDS Plot. The MDS results are invariant to orientation. When the results have a known orientation, such as physical locations, then you might want to rotate or flip your plot.

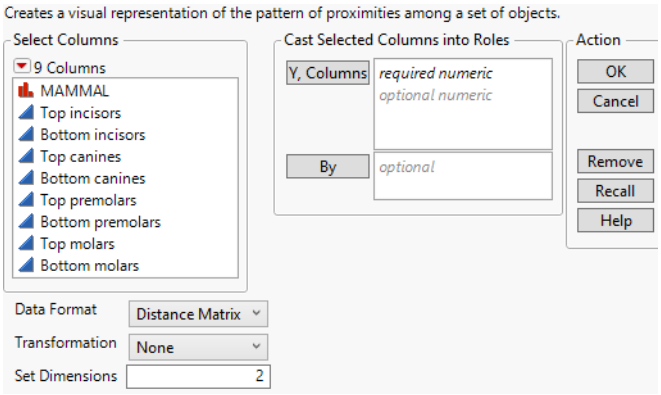
Figure 10.3 Multidimensional Scaling Plot



Launch the Multidimensional Scaling Platform

Launch the Multidimensional Scaling Platform by selecting **Analyze > Multivariate Methods > Multidimensional Scaling**.

Figure 10.4 Multidimensional Scaling Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns The columns to be analyzed. These must have a Numeric data type.

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Note: When using a distance matrix, the By variable requires a full matrix for each level of the By variable.

Data Format MDS supports two data formats:

Distance Matrix A full symmetric, lower, or upper triangular matrix where the number of rows equals the number of columns.The diagonal entries can either be zeros or missing.

Attribute List A set of columns that contain measures of a quality or characteristic of an object. The objects are typically named in a column. The object column is not used in the analysis but rather is used as a label for the data points on the MDS plot.

Transformation Supported transformations are Ratio, Interval, and Ordinal.

None No transformation used.

Ratio Data has an ordering from smallest to largest, the differences between values have meaning, and the scale has a true zero. Used to scale the MDS plot.

Interval Data has an ordering from smallest to largest and the differences between values have meaning. Used to scale and shift the MDS plot.

Ordinal Data has an ordering from smallest to largest. Used for ordinal data.

Set Dimensions The number of dimensions for the visual representation of the proximities among your objects. Typically, two or three dimensions are used. With greater than three dimensions, the visualization becomes complex.

Note: The dimension selected can be between 1 to n - 1 where n = the number of objects, otherwise the dimension is set to 2.

The Multidimensional Scaling Report

The initial Multidimensional Scaling report shows these reports: Multidimensional Scaling Plot, the Shepard Diagram, and the Fit Details. If you specify three or more dimensions for the fit in the launch window, then the Multidimensional Scaling Plot provides controls for selecting the dimensions that you view.

Objects that are close together on the MDS plot share similar attributes. Adding labels and colors to the plot can help in the identification of similar groups. The Shepard diagram and summary of fit statistics provide measures of how well the MDS plot represents the proximities of the objects.

Multidimensional Scaling Plot

The MDS plot displays the multidimensional scaling in two dimensions. Below the plot are two buttons to flip the axis either in the vertical or horizontal direction. The MDS solution can be reflected, rotated, or translated without changing the inter-point proximities. The rotating or reflection of the axes is most common when working with geographical objects that have a known map orientation.

If more than two dimensions were used in the analysis, then you can toggle the dimensions shown in the plot using the **Select Dimension** controls below the plot. The first control defines the horizontal axis of the plot, and the second control defines the vertical axis of the plot.

Shepard Diagram

The Shepard plot is a plot of the actual or transformed proximities versus the predicted proximities. The plot indicates how well the Multidimensional Scaling Plot reflects the actual proximities. The Shepard is analogous to an Actual by Predicted plot. Ideally the points fall on the Y = X line, which is shown in red.

Fit Details

The Fit Details provides statistics that summarize how well the MDS proximities match the actual proximities as well as details about transformations when used.

Stress The value of the stress function (Stress1) that was minimized in the fitting procedure. Stress can be between 0 and 1 with lower values indicating a better fit.

RSquare The R^2 value for linear fit of the actual or transformed proximities versus the predicted proximities.

Slope If a ratio or interval transformation was used, the slope for the transformation is provided. It is the slope of the linear regression of the actual against transformed proximities.

Intercept If an interval transformation was used, the intercept for the transformation is provided. It is the intercept of the linear regression of the actual against transformed proximities.

Multidimensional Scaling Platform Options

The Multidimensional Scaling red triangle menu options give you the ability to customize reports according to your needs. The options available are determined by the type of data and the number of dimensions that you use for your analysis.

MDS Plot Shows or hides the MDS Plot.

Diagnostics Provides diagnostics for the MDS.

Shepard Diagram Shows a plot of actual proximity (or transformed proximity if a transformation is used) versus the predicted proximity. This report appears by default. See "Shepard Diagram".

Waern Links Displays the Waern links on the MDS plot. Controls for the portion (smallest or largest) are available when this option is selected. See "Waern Links".

Show Coordinates Provides a report of the solution coordinates. These are the coordinates of the points on the Multidimensional Scaling Plot. The report shows the coordinates of up to three dimensions. Right-click in the report and select **Columns** to add additional dimensions to the report. The maximum number of dimensions is the number of dimensions set in the launch window.

Show Proximity Provides a report of the proximities. The original and derived proximities (distances) are provided between each pair of objects. The pairs are identified in the From and To object columns. If a transformation was used, the transformed proximities are also included in the table.

Save Proximity (Available only if Attribute List is the data format.) Saves the distance matrix to the data table.

3D Plot (Available only if three or more dimensions are specified for Set Dimensions in the launch window.) Shows a 3-D plot of the first three dimensions.

Save Coordinates Saves the solution coordinates to the data table in separate columns.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Waern Links

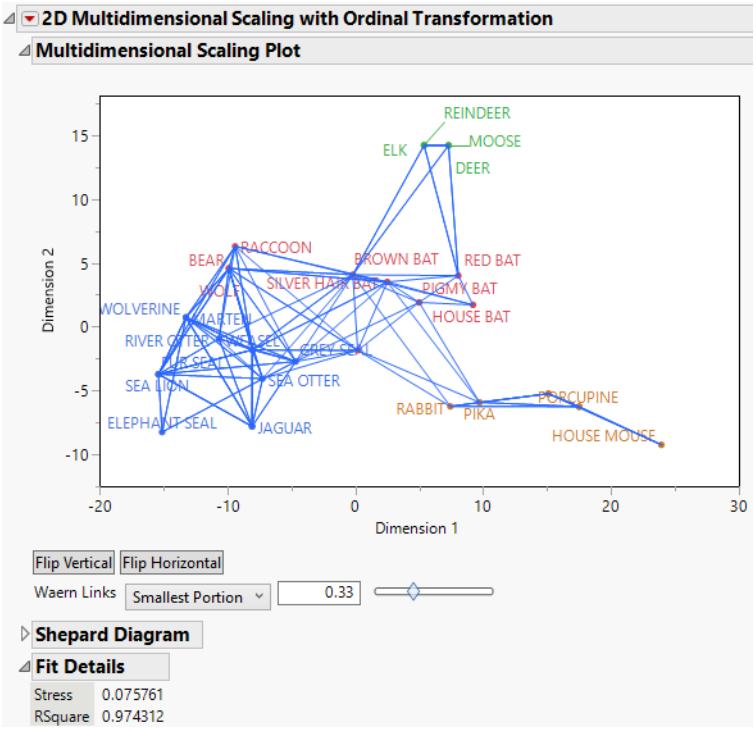
Waern links provide a visual check of the MDS results by comparing actual proximities to predicted proximities. The links join points on the Multidimensional Scaling Plot based on their actual proximities. The objects with the smallest (largest) proximities are connected. A typical scenario to consider is the smallest 33% of the proximities between objects. If the MDS Plot is a good representation of the proximities, then the links for the smallest actual proximities should connect the closest objects in the plot. If a link for a small proximity stretches across the plot connecting distant objects, then the MDS fit would be questioned.

Waern Link Controls

There is a list from which you can choose to show the Smallest Portion or the Largest Portion of links on the plot. The portion of links shown is controlled by entering a value in the box or by using the slider. Figure 10.5 shows Waern links for the Teeth.jmp data table for the 33% smallest portion.

For more information about Waern links, see Waern (1972).

Figure 10.5 MDS Plot with Waern Links

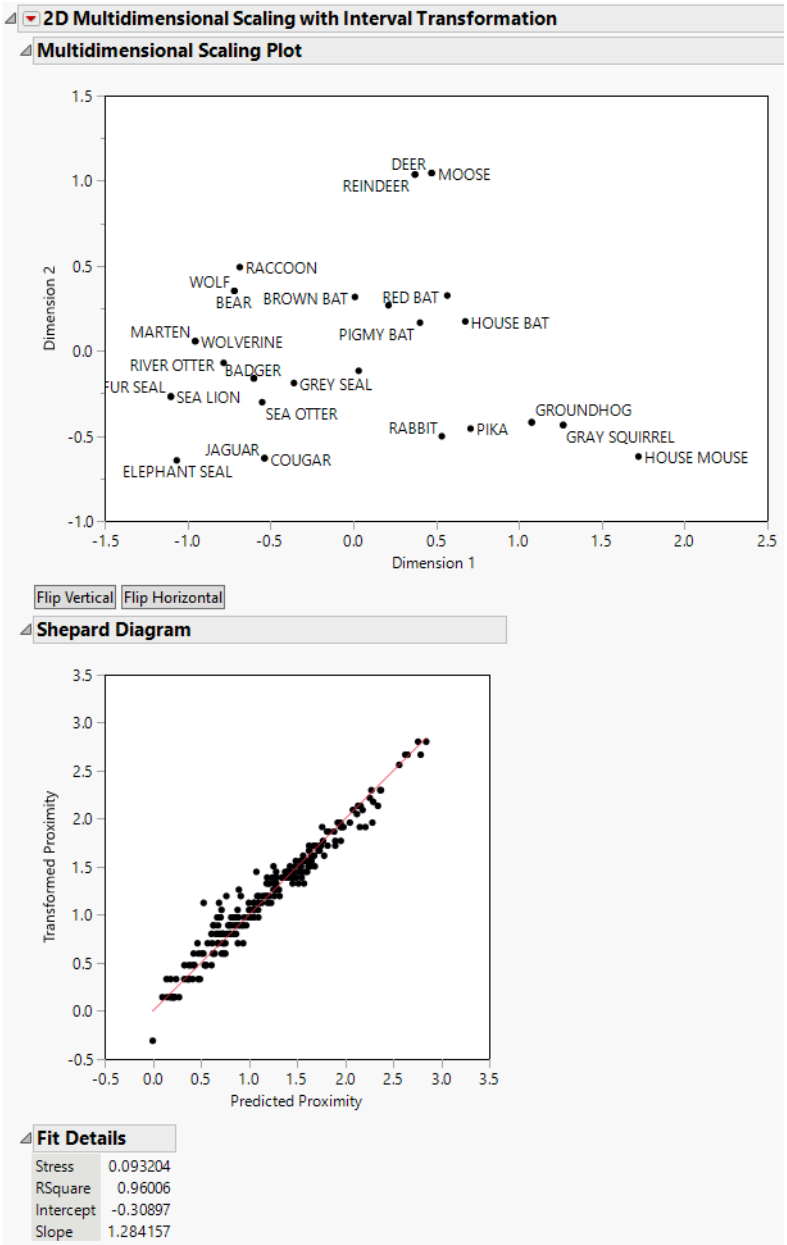


Additional Example of the Multidimensional Scaling Platform

This example uses the Teeth.jmp sample data table, which is an attribute list of the counts of eight teeth types in 32 mammals. You can use MDS to explore the similarities of mammals based on their teeth. An interval transformation is used to illustrate the output from that transformation. The data do have an ordering that has a meaning (four teeth are twice as many as two teeth). One might explore other transformations such as the ordinal transformation.

1. Select **Help > Sample Data Library** and open Teeth.jmp.
2. Right-click the column MAMMAL and select **Label/Unlabel**.
3. Select **Rows > Row Selection> Select all Rows.**
4. Select **Rows > Label/Unlabel**
5. Select **Analyze > Multivariate Methods > Multidimensional Scaling**.
6. Select Top incisors through Bottom molars and click **Y, Columns**.
7. Select **Data Format > Attribute List**.
8. Select **Transformation > Interval**.
9. Click **OK**.

Figure 10.6 Multidimensional Scaling Report



The Shepard Diagram and the Fit Details indicate that the MDS Plot is a good representation of similarities of animals due to similarities in their teeth. The Stress statistics of 0.093 is low and the R^2 fit of the transformed versus predicted proximities is

high at 0.96. In addition, the Fit Details provides the intercept and slope for the transformation of the actual proximities.

Statistical Details for the Multidimensional Scaling Platform

JMP uses a quasi Newton optimization method to minimize the Stress function to determine the MDS coordinates. This minimization leads to a set of coordinates in a predetermined number of dimensions that minimize the derived proximity measures for each pairwise set of the dimensions. When the data is ordinal, monotonic regression is used. Otherwise, standard least squares regression is used.

- "Stress"
- "Transformations"
- "Attributes List Format"

Stress

The following notation is used to define Stress:

- i, j - indexes for the number of objects
- d_{ij} - the derived distance between objects i and j

  δ_{ij} - the observed relative distance between objects i and j
- $f(\delta_{rs})$ - transformation function for the distance

The Stress function is defined as follows:

$$\text{Stress} = \left[\frac{\sum_{i<j} [f(\delta_{ij}) - d_{ij}]^2}{\sum_{i<j} d_{ij}^2} \right]^{\frac{1}{2}}$$

This measure of stress is also known as Kruskal's Stress, Type I, or simply Stress1.

Transformations

The section uses the notation described in "Stress". Transformations are used to scale the actual proximities. Transformations would be considered to improve the MDS representation of the actual proximities by taking into account specific structures in the data. The parameters in the transformation functions become additional parameters in the minimization algorithm.

Ratio Transformation

For ratio data:

$$f(\delta_{rs}) = b\delta_{rs}$$

Interval Transformation

For interval data:

$$f(\delta_{rs}) = a + b\delta_{rs}$$

Ordinal Transformation

For ordinal data the data is not transformed, rather the algorithm uses monotone regression rather than least squares regression.

Attributes List Format

When the data is an attributes list, it is converted to a distance matrix and then MDS is applied. The distance matrix is calculated using Euclidean distance. For each pair of items define the distance between the items by:

$$\delta_{ij} = \sqrt{\sum_k \frac{(x_{ki} - x_{kj})^2}{k}}$$

where k is the number of attributes.

Note: For an advanced example of the MDS platform, see the San Francisco Crime Distances.jmp sample data table and the source script for that table. The script creates the distance matrix using pairwise correlations. The resulting distance matrix is then used to explore the relationships between crime categories.

Chapter 11

Item Analysis

Analyze Test Results by Item and Subject

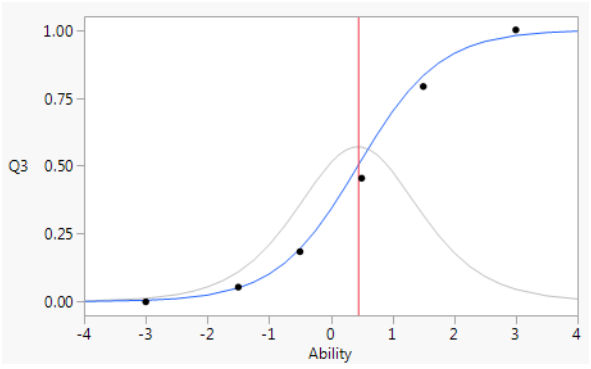
The Item Analysis platform enables you to fit *item response theory* models. The Item Response Theory (IRT) method is used for the analysis and scoring of measurement instruments such as tests and questionnaires. Item response theory uses a system of models to relate a trait or ability to an individual's probability of endorsing or correctly responding to an item. Frequently, the trait or ability of interest is not directly measurable and is therefore called *latent*. IRT can be used to study standardized tests, cognitive development, and consumer preferences. IRT is an alternative method to classical test theory (CTT) where the focus is on the total observed score rather than the item scores.

The Item Analysis platform implements the IRT method with the following outcomes:

- Measurement instruments are scored at the item level, providing insight into the contributions of each item on the latent response.
- Scores for both the responders and the items are produced on the same scale.
- Respondent and item scores are shown on a single plot.
- Item characteristic curves are shown. These curves can be used to explore the relationship between items and respondent's underlying trait or ability.

For more information about item response theory, see de Ayala (2009).

Figure 11.1 Item Analysis Characteristic Plot



Contents

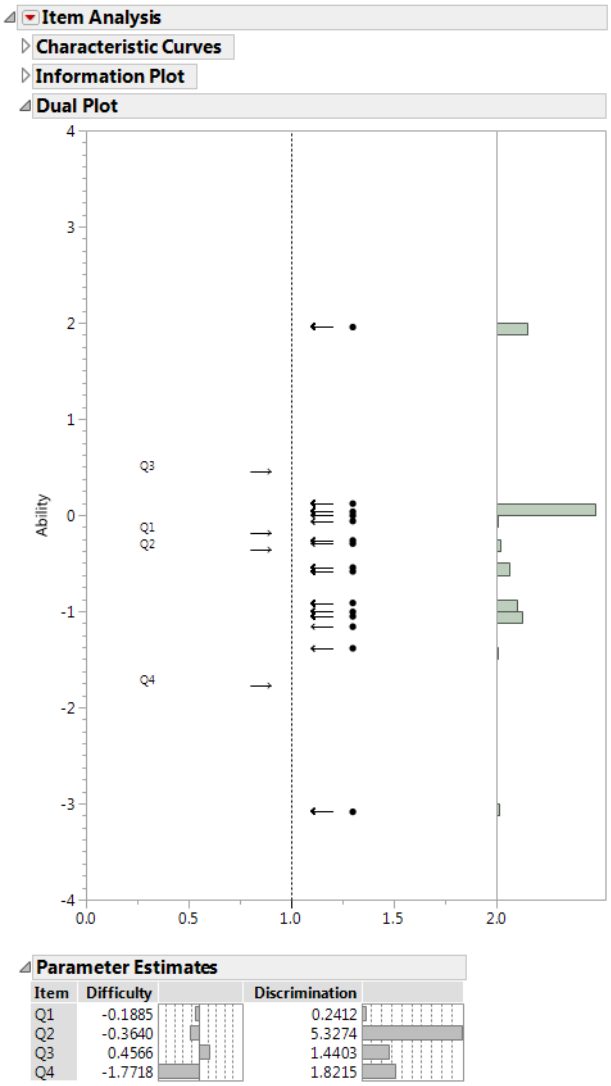
Example of Item Analysis 257
Launch the Item Analysis Platform 260
Logistic 3PL Model Details 261
Data Format 261
The Item Analysis Report 261
Characteristic Curves 261
Information Plot 262
Dual Plot 263
Parameter Estimates 264
Item Analysis Platform Options 265
Statistical Details for the Item Analysis Platform 265
Item Response Curves 266
Item Response Curve Models 267
IRT Model Assumptions 269
Fitting the IRT Model 269
Ability Formula 271

Example of Item Analysis

This example uses the MathScienceTest.jmp sample data table, which is a subset of the data from the Third International Mathematics and Science Study (TIMSS) conducted in 1996. The data table contains scores (1 = correct or 0 = incorrect) for 1263 students on 14 questions. You examine the first four questions to understand the relationship between questions and respondent's mathematical ability. The questions on the test are the items that are used to measure the latent mathematical ability. Fit a 2PL model to this data.

1. Select **Help > Sample Data Library** and open MathScienceTest.jmp.
2. Select **Analyze > Multivariate Methods > Item Analysis**.
3. Select Q1 through Q4, click **Y, Test Items** and click **OK**.

Figure 11.2 Item Response Report

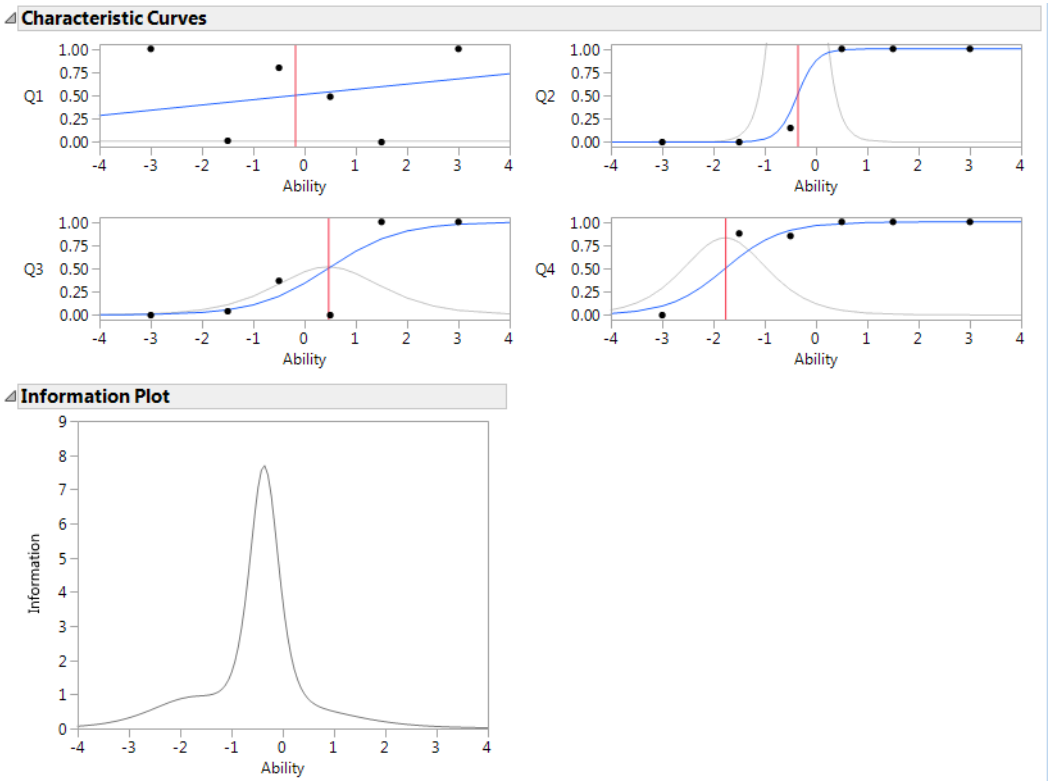


From the dual plot you note that Q4 is the easiest of the four questions to answer as it has the lowest difficulty score at -1.78. Q3 is the hardest with a difficulty score of 0.46. Most of the respondents fall in the middle to lower end of the ability scale as shown by the data points in the center part of the graph. In the histogram, you can see that approximately 40% of the respondents fall slightly above zero on the ability scale.

Note: Ability scores are not computed for individuals with all incorrect or all correct answers. See "Fitting the IRT Model".

4. Click the gray Characteristic Curves report disclosure icon to open.
5. Click the Item Analysis red triangle and select **Number of Plots Across**.
6. Enter 2 and click **OK**.
7. Click the gray Information Plot report disclosure icon to open.

Figure 11.3 Item Response Example



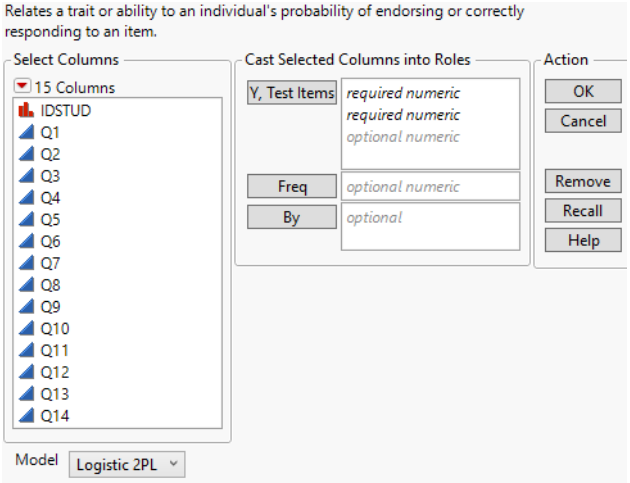
Q1 has a flat characteristic curve and a flat information curve. This suggests that Q1 does not provide much information to discriminate respondents' mathematical ability. The characteristic curve for Q2 is steep, which indicates that Q2 is useful for discriminating respondent ability. The vertical line in each plot is at the inflection point for the characteristic curve. This vertical line is the ability level at which the respondent has a 50% probability of answering the specified question correctly.

The information plot indicates that together the four questions analyzed provide the most information about ability levels between about -1 and 0. Including more questions of higher difficulty in the model could broaden the information curve.

Launch the Item Analysis Platform

Launch the Item Analysis platform by selecting **Analyze > Multivariate Methods > Item Analysis**.

Figure 11.4 Item Analysis Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Test Items Assigns two or more columns to be analyzed. The columns must be numeric, continuous, and contain only 0s and 1s.

> **Tip:** Use **Cols > Recode** if you need to recode your data to 0s and 1s. See *Using JMP*.

Freq Assigns a frequency variable to this role. This is useful if your data are summarized.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variable.

Model Specifies the desired model from the following options:

Logistic 2PL The 2-parameter logistic model.

Logistic 3PL The 3-parameter logistic model.

Logistic 1PL The 1-parameter logistic model with a Rasch parameterization.

Logistic 3PL Model Details

If you select Logistic 3PL for Model, you are prompted to enter a penalty for the guessing parameters after you click OK. For a 3PL model, the default value of the penalty is zero. However, you can enter a non-zero penalty for the *c* parameters (the guessing for each item). This penalty is similar to the type of penalty parameter that you would use in ridge regression. The penalty is on the variance of the estimated guessing parameters. The use of the penalty has the following benefits:

- Stabilizes the estimation of model parameters.
- Speeds up computations.
- Reduces the variability of the guessing parameter across items at the expense of some bias.

Large values of the penalty force the guessing parameters to zero while smaller values help reduce the variability of the guessing parameter across items. A value of zero can be used for no penalty.

Data Format

The Item Analysis platform requires a data table that contains a row for each individual and a column for each item. The item columns must be numeric and contain only 0s and 1s to indicate incorrect or correct responses, respectively. The MathScienceTest.jmp sample data table illustrates the required data format for an item response analysis of 1,263 individuals responses to 14 test questions.

The Item Analysis Report

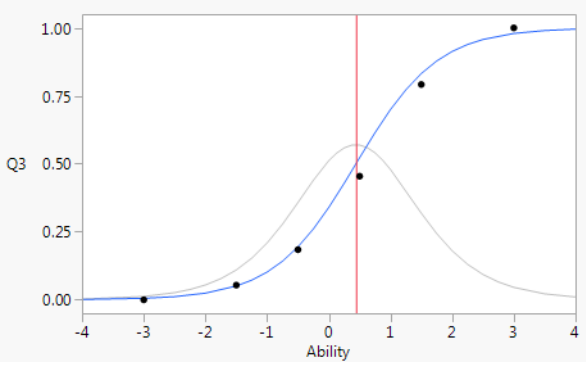
- "Characteristic Curves"
- "Information Plot"
- "Dual Plot"
- "Parameter Estimates"

Characteristic Curves

The Characteristic Curves contains an item characteristic curve (ICC) for each item that you specified in the launch window. The Characteristic Curves are initially closed.

The item characteristic curve plots the probability of answering an item correctly versus ability. Ability is measured on a standardized scale, so a respondent with ability equal to 0 is a respondent of average ability. Data points for the observed probability of correct answers for fixed ability levels are plotted. Comparing the fitted characteristic curve to the data points provides a visual measure of goodness of fit of the model for each individual item. In addition, the characteristic plots have a background information curve and a vertical line at the characteristic curve inflection point. The background information curve is a plot of the slope of the item characteristic curve, which is maximized at the inflection point.

Figure 11.5 Item Characteristic Curve

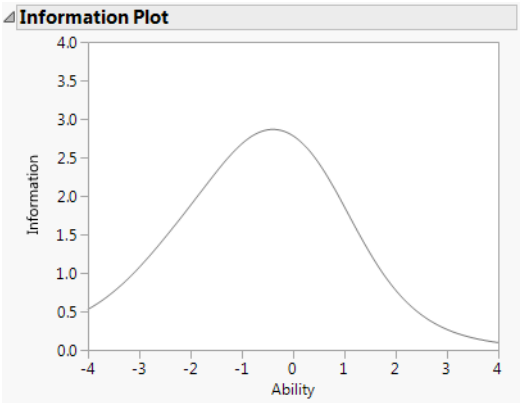


Tip: You can adjust the number of characteristic curves that appear in each row of the report using the Number of Plots Across option in the Item Analysis red triangle menu.

Information Plot

The Information Plot report contains a plot of the overall information curve, which is constructed by summing the individual item information curves. The information plot provides insight into the appropriate ability levels that the test is able to measure. Figure 11.6 describes a test with items that are appropriate for assessing individuals with average to low levels of the ability more so than individuals with high levels of ability. This plot is initially closed.

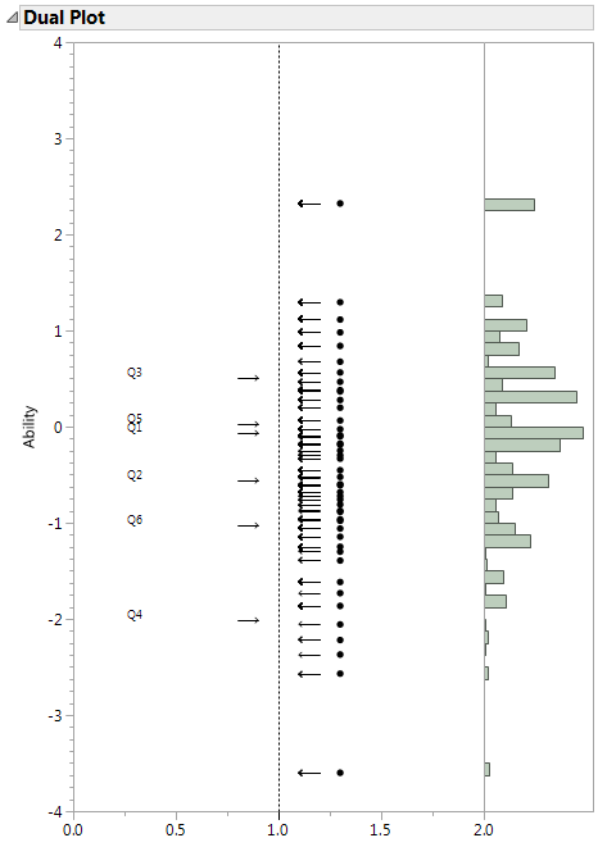
Figure 11.6 Information Plot



Dual Plot

The Dual Plot report contains a plot that shows item difficulty and subject ability in one plot. Difficulty and ability use a common standardized scale shown on the vertical axis. The items are plotted by their difficulty on the left side of the plot. The subjects are plotted to the right with data points and a histogram. The dual plot enables you to relate the difficulty of each item to the ability of each respondent.

Figure 11.7 Dual Plot



Parameter Estimates

The Parameter Estimates report contains a table of estimated parameters for each item. The parameters provided depend on the model used in your analysis (1PL, 2PL, or 3PL).

Item The test item.

Difficulty The *b* parameter or the measure of the difficulty of the item. A histogram of the difficulty parameters is shown beside the difficulty estimates.

Discrimination (Available only for 2PL and 3PL models.) The *a* parameter or the measure of the item discrimination. A histogram of the discrimination parameters is shown beside the discrimination estimates.

Lower Asymptote (Available only for 3PL models.) The *c* parameter or a measure of guessing.

Item Analysis Platform Options

Number of Plots Across Enables you to specify how many ICC plots to display in each row of plots in the Characteristic Curve report. The default is one ICC plot per row.

Save Ability Formula Saves the ability formula to a new column in the data table. This options uses the `IRT Ability()` JSL function. For more information about this function, see Help > Scripting Index.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Statistical Details for the Item Analysis Platform

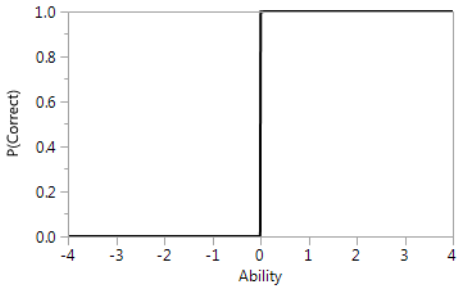
Item response theory (IRT) uses a series of equations to relate items to an unobserved (latent) trait or ability. Items, or questions, are indicators of an underlying latent construct that cannot be directly observed. At the time of data collection, both the subject abilities and the item characteristics are unknown.

- "Item Response Curves"
- "Item Response Curve Models"
- "IRT Model Assumptions"
- "Fitting the IRT Model"
- "Ability Formula"

Item Response Curves

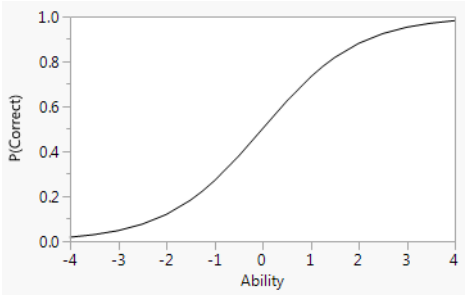
Item response curves (item characteristic curves) are used to describe the relationship between the ability, defined on an *ability* scale, and each item. An item response curve plots the probability of correctly answering an item against different levels of ability. An item with perfect discrimination has a 0% probability of correct answers for respondents with ability below a threshold and a 100% probability of a correct response for subjects with ability above the threshold.

Figure 11.8 Characteristic Curve for an Item with Perfect Discrimination



A typical relationship between the probability of correctly answering an item and ability is an S-shaped function with lower and upper asymptotes. As a respondent's ability increases, their probability of correctly answering the item increases to 100%. The shape of the curve for a specific item is related to the difficulty and discriminatory properties of the item.

Figure 11.9 Typical Item Response Curve



Item Response Curve Models

One-, two-, and three-parameter logistic models can be used to model the item response curves. The three-parameter logistic (3PL) model is defined as follows:

$$P(\theta) = c + \frac{1-c}{1+e^{-(a)(\theta-b)}}$$

- $P(\theta)$ is the probability of answering the item correctly for an ability level θ. For more information about fitting the item response theory model, see "Fitting the IRT Model".
- The a parameter defines the steepness of the curve at its inflection point. It provides an estimate of the discriminatory power of the item.
- The b parameter defines the location of the inflection point on the Ability axis. It provides an estimate of the difficulty of an item.
- The c parameter is the lower asymptote. It provides an estimate of the probability that an item is answered correctly by guessing.
- For the 2PL model, the c parameter is set to 0.

$$P(\theta) = \frac{1}{1+e^{-(a)(\theta-b)}}$$

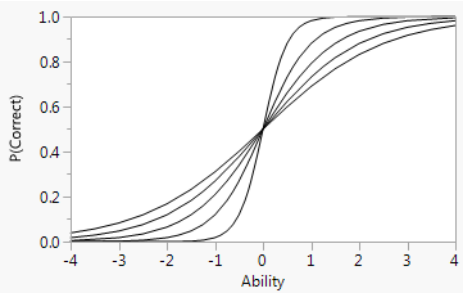
- For the 1PL model, the c parameter is set to 0 and the a parameter is set to 1. This parameterization is also known as the Rasch model (Rasch 1980).

$$P(\theta) = \frac{1}{1+e^{-(\theta-b)}}$$

The a Parameter: Item Discrimination

In the 2PL and 3PL models, the a parameter, or the steepness of the curve at its inflection point, provides a measure of the discriminatory power of an item. The discriminatory power, or discrimination, of an item refers to how well an item can distinguish between respondents with low ability levels versus those with high ability levels. A steep item response curve indicates that the item has strong discrimination. Respondents with low ability levels have a low probability of a correct response to the item while respondents with high ability have a high probability of a correct response. Items whose curves are relatively flat have low discrimination. Items with low discrimination are candidates to be dropped from the measurement instrument.

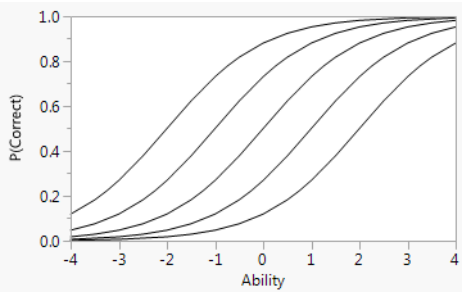
Figure 11.10 Logistic Model for Several Values of *a*



The b Parameter: Item Difficulty

The *b* parameter, or the location of the inflection point with respect to ability, provides a measure of item difficulty. Item response curves with inflection points farther to the right on the ability scale are indicative of items that are more difficult to answer than items with inflection points to the left. In the 1PL and 2PL models, the *b* parameter provides an estimate of the ability level required for a 50% probability of correctly answering the item.

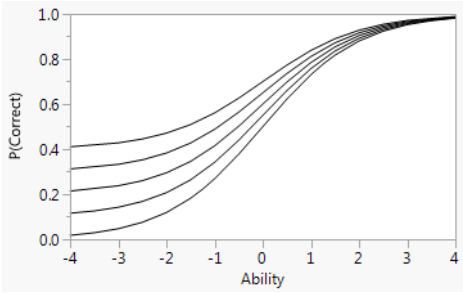
Figure 11.11 Logistic Curve for Several Values of *b*



The c Parameter: Guessing

In the 3PL model, the *c* parameter, or the lower asymptote of the item response curve, provides a measure of the guessing parameter. A nonzero lower asymptote represents the nonzero probability of a person with a very low ability level answering an item correctly.

Figure 11.12 Logistic Model for Several Values of *c*



IRT Model Assumptions

The 2PL model is the default model in the Item Analysis platform. The 1PL model is appropriate when you can assume that all items have equal discriminating power. When this assumption is not appropriate, the 2PL or 3PL model should be used. The 2PL model has greater numerical stability than the 3PL model, especially for small data sets. In addition, in the 2PL model, b can be interpreted as the ability level required for a 50% chance of a responder answering an item correctly.

The IRT model assumes that the underlying trait is unidimensional. That is, there is a single underlying latent construct. If there are several traits that have complex interactions with each other being measured, then a unidimensional model is not appropriate. The IRT model is appropriate for continuous latent variables. For a categorical latent variable, you should consider a latent class model. See "Latent Class Analysis". IRT models are assumed to be item-invariant. Item-invariance means that $P(\theta)$ is interpreted as the probability of a correct response for a set of individuals with ability level θ. If a large group of individuals with equal ability levels answered the item, $P(\theta)$ predicts the proportion who would answer the item correctly. This implies that IRT models would have the same parameters regardless of the group of subjects tested. In addition, the IRT model assumes local independence, which means that once the latent construct has been accounted for, the items are independent of one another.

Fitting the IRT Model

The IRT model is fit using Marginal Maximum Likelihood estimation (MMLE). MMLE is an alternative method to Joint Maximum Likelihood estimation (JLE). MMLE treats the subjects as random effects. The items and abilities are related as conditional probabilities. The formulas are defined as follows:

$$p(\mathbf{x}|\theta, \vartheta) = \prod_{j=1}^{L} p_j(\theta)^{x_j}(1 - p_j(\theta))^{1 - x_j}$$

where $p(\mathbf{x}|\theta, \vartheta)$ is the probability of a response vector $\mathbf{x}$ given the subject ability θ and the vector of item parameters ϑ. The number of item parameters depends on the model used (1PL, 2PL, or 3PL).

MMLE integrates out the subject effects using Gaussian quadrature to obtain item parameter estimates. The probability of response vector $\mathbf{x}$ is calculated as follows:

$$p(\mathbf{x}) = \int_{-\infty}^{\infty} p(\mathbf{x}|\theta, \vartheta) g(\theta|\upsilon) d\theta$$

where $g(\theta|\nu)$ is the distribution of the subjects and ν is a vector of the population location and scale parameters. The normal distribution with mean 0 and standard deviation 1 is used for $g(\theta|\nu)$ in JMP.

Note: A missing value for a test question is treated as an incorrect response. Ability scores are not calculated for individuals with all incorrect or all correct answers. The patterns of the responses for these subjects are included in the model estimation.

The MMLE procedure for fitting the IRT model can be compared to fitting a random effects model in two stages. The ability parameters are treated as random effects with variance of 1. In the first step, these random effects are integrated out using Gaussian quadrature. The item parameters are treated as fixed effects that are estimated using ML from the marginal likelihood with the ability parameters integrated out. The ability parameters are in essence best linear unbiased predictions that are estimated using the full unintegrated (joint) likelihood, treating the item parameters as known and held fixed at the values obtained in the first stage.

There are 2^L patterns of responses for L items. The ability level for each pattern can be calculated by finding the ability level with the highest probability for the response pattern by applying the following until θ converges:

$$\theta_i^{t+1} = \theta_i^{t} - \frac{X_i - \sum_{j=1}^{L} p_{ij}(t)}{-\sum_{j=1}^{L} p_{ij}(t)(1 - p_{ij}(t))}$$

where:

θ maximizes the likelihood of obtaining the response pattern

t is the number of iterations

L is the number of items

X_i is the observed score

p_{ij} is the probability of a correct response on the j^{th} item by the i^{th} person based on the item parameters.

Ability Formula

The Save Ability Formula option from the Item Analysis red triangle menu saves the ability formula to a new column in the data table. This formula can be used to score additional subjects added to the data table or it can be copied to a new table to score a new set of subjects.

The function saved to the data table is called the `IRT Ability` function. The item parameter estimates are stored in a matrix in this function. For more information about this function, see Help > Scripting Index.

Chapter 12

Hierarchical Cluster

Group Observations Using a Tree of Clusters

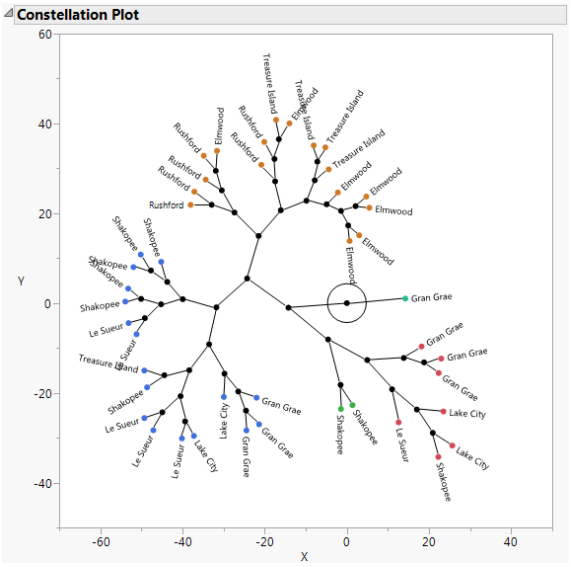
Clustering is a multivariate technique that groups together observations that share similar values across a number of variables. Use it to understand the clumping structure of your data.

Hierarchical clustering combines clusters successively. The method begins by treating each observation as its own cluster. Then, at each step, the two clusters that are closest in terms of distance are combined into a single cluster. The result is depicted as a tree, called a *dendrogram*.

Use hierarchical clustering for small data tables with no more than several tens of thousands of rows. The algorithm is time-intensive and can run slowly for larger data tables. For larger data tables, use K Means Cluster or Normal Mixtures.

Note: Hierarchical cluster supports character columns; K Means Cluster or Normal Mixtures require numeric columns.

Figure 12.1 Example of a Constellation Plot



Contents

Overview of the Hierarchical Clustering Platform 275
Overview of Platforms for Clustering Observations 275
Example of Clustering 277
Launch the Hierarchical Cluster Platform 280
Clustering Method 281
Method for Distance Calculation 281
Data Structure 282
Transformations to Y, Columns Variables 284
Hierarchical Cluster Report 285
Dendrogram Report 286
Illustration of Dendrogram and Distance Graph 286
Clustering History Report 288
Hierarchical Cluster Options 288
Additional Examples of the Hierarchical Clustering Platform 292
Example of a Distance Matrix 292
Example of Wafer Defect Classification Using Spatial Measures 294
Statistical Details for the Hierarchical Clustering Platform 296
Spatial Measures 297
Distance Method Formulas 299

Overview of the Hierarchical Clustering Platform

Hierarchical Clustering is one of four platforms that JMP provides for clustering observations. For a comparison of all four methods, see "Overview of Platforms for Clustering Observations".

The hierarchical clustering method starts with each observation forming its own cluster. At each step, the clustering process calculates the distance between all pairs of clusters and combines the two clusters that are closest together. This process continues until all the points are contained in one cluster. Hierarchical clustering is also called *agglomerative clustering* because of the combining approach that it uses.

The agglomerative process is portrayed as a tree, called a dendrogram. To help you decide on a number of clusters, JMP provides a distance graph. You can select a number of clusters by determining when the distances between clusters no longer appear to be of practical importance.

Hierarchical clustering also supports character columns. There are two ways to define the distances.

- If a column is ordinal, then the value used for clustering is the index of the ordered category, treated as if it were continuous data. These values are standardized as if they were continuous data.
- If a column is nominal, then the distance between two observations where the categories match is zero. If the categories differ, the distance is one.

Hierarchical clustering enables you to choose among five rules for defining distances between clusters: Average, Centroid, Ward, Single, and Complete. Each rule can generate a different sequence of clusters.

Tip: The hierarchical clustering process starts with $n(n + 1)/2$ distances for n observations, except when the Fast Ward method is used. For this reason, this method can take a long time to run when n is large. For large numbers of numeric observations, consider K Means Cluster or Normal Mixtures.

Overview of Platforms for Clustering Observations

Clustering is a multivariate technique that groups together observations that share similar values across a number of variables. Typically, observations are not scattered evenly through p-dimensional space, where p is the number of variables. Instead, the observations form clumps, or clusters. Identifying these clusters provides you with a deeper understanding of your data.

Note: JMP also provides a platform that enables you to cluster variables. See "Cluster Variables".

JMP provides four platforms that you can use to cluster observations:

- Hierarchical Cluster is useful for smaller tables with up to several tens of thousands of rows and allows character data. Hierarchical clustering combines rows in a hierarchical sequence that is portrayed as a tree. You can choose the number of clusters that is most appropriate for your data after the tree is built.
- K Means Cluster is appropriate for larger tables with up to millions of rows and allows only numerical data. You need to specify the number of clusters, *k*, in advance. The algorithm guesses at cluster seed points. It then conducts an iterative process of alternately assigning points to clusters and recalculating cluster centers.
- Normal Mixtures is appropriate when your data come from a mixture of multivariate normal distributions that might overlap and allows only numerical data. For situations where you have multivariate outliers, you can use an outlier cluster with an assumed uniform distribution.

  You need to specify the number of clusters in advance. Maximum likelihood is used to estimate the mixture proportions and the means, standard deviations, and correlations jointly. Each point is assigned a probability of being in each group. The EM algorithm is used to obtain estimates.
- Latent Class Analysis is appropriate when most of your variables are categorical. You need to specify the number of clusters in advance. The algorithm fits a model that assumes a multinomial mixture distribution. A maximum likelihood estimate of cluster membership is calculated for each observation. An observation is classified into the cluster for which its probability of membership is the largest.

Table 12.1 Summary of Clustering Methods

Method	Data Type or Modeling Type	Data Table Size	Specify Number of Clusters
Hierarchical Cluster	Any	With Fast Ward, up to 200,000 rows With other methods, up to 5,000 rows	No
K Means Cluster	Numeric	Up to millions of rows	Yes
Normal Mixtures	Numeric	Any size	Yes
Latent Class Analysis	Nominal or Ordinal	Any size	Yes

Some of the clustering platforms have options to handle outliers in the data. However, if your data has outliers, it is best to explore them first prior to analyzing. This can be done using the Explore Outliers Utility. For more information, see the Modeling Utilities chapter in *Predictive and Specialized Modeling*.

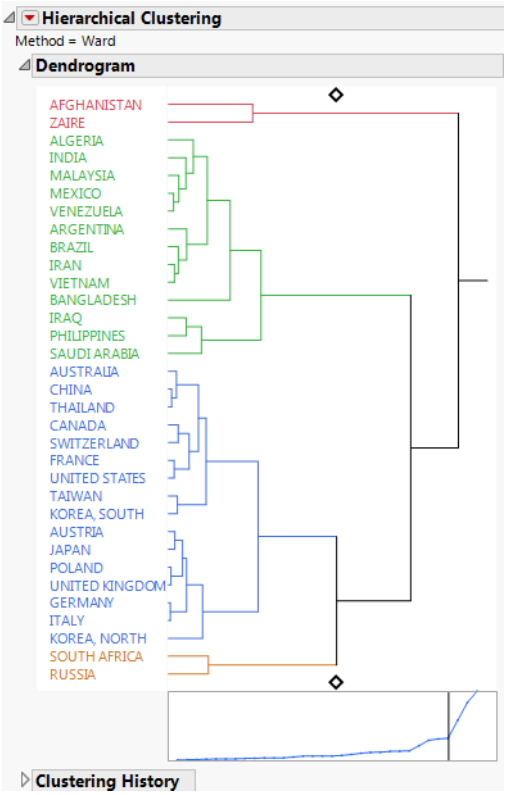
Example of Clustering

In this example, we group together countries by their 2009 crude birth and death rates per 1,000 people.

1. Select **Help > Sample Data Library** and open Birth Death Subset.jmp
2. Select **Analyze > Clustering > Hierarchical Cluster**.
3. Select birth and death and click **Y, Columns**.
4. Select country and click **Label**.

   This selection ensures that the country column, rather than the row number, is used to label the dendrogram that appears when you click OK.
5. Click **OK**.
6. Click the Hierarchical Clustering red triangle and select **Color Clusters**.

Figure 12.2 Hierarchical Clustering Report



The dendrogram shows how the clustering is conducted. The clustering process can be viewed by reading the dendrogram from left to right. Each step consists of combining the two *closest* clusters into a single cluster.

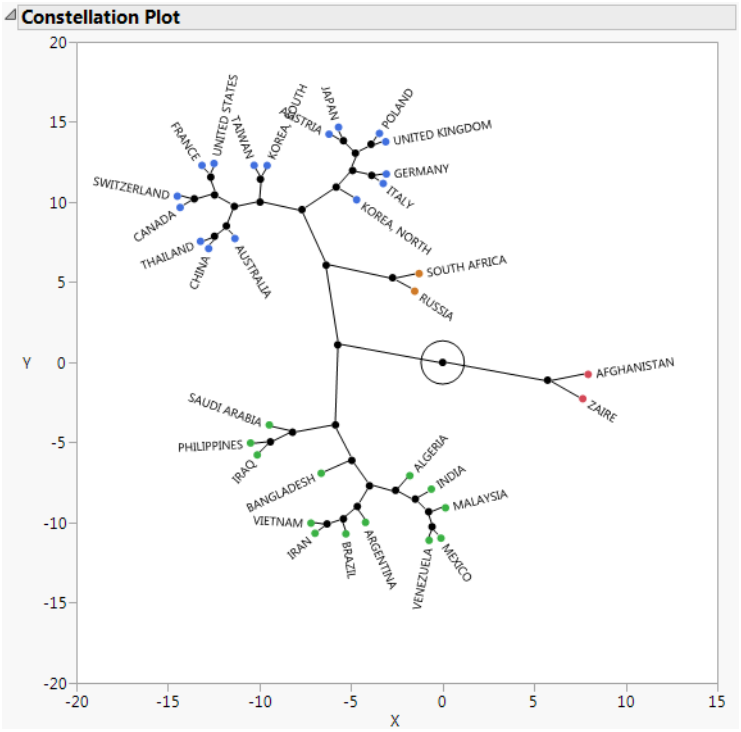
In the dendrogram, the relative distances between clusters are given by the horizontal distances between vertical lines that join the clusters. For example, Afghanistan and Zaire differ more than Malaysia differs from the cluster consisting of Mexico and Venezuela.

The plot that appears beneath the dendrogram has a point for each step where two clusters are joined into a single cluster. The horizontal coordinates represent the numbers of clusters and they decrease from left to right. The vertical coordinate of the point is the distance between the two clusters that are joined to form the specified number of clusters. You can click either diamond in the dendrogram and drag the line to choose the number of clusters that best represent the data. You can also use the Number of Clusters option in the Hierarchical Clustering red triangle menu to choose the number of clusters.

The distance graph has a noticeable change in slope at four clusters. The change in slope indicates that the differences in clusters that are joined up to the point where four clusters remain, are comparatively small. This suggests that four is a good choice for the number of clusters. Note that this is the number of clusters that was shown by default.

7. Click the Hierarchical Clustering red triangle and select **Constellation Plot**.

Figure 12.3 Constellation Plot

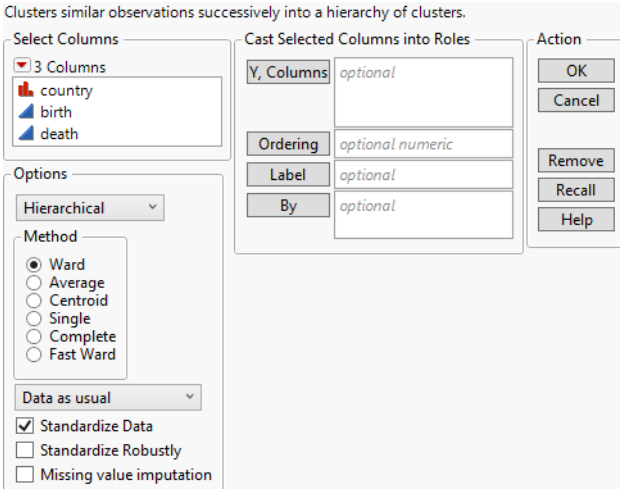


This constellation plot arranges the countries as endpoints and each cluster join as a new point. The lines represent membership in a cluster. The length of a line between cluster joins approximates the distance between the clusters that were joined. The constellation plot indicates that the cluster that contains Afghanistan and Zaire is about as distant from the cluster of remaining countries as are the two clusters that consist of the remaining countries in the upper half of the plot and those in the lower half of the plot.

Launch the Hierarchical Cluster Platform

Launch the Hierarchical Cluster platform by selecting **Analyze > Clustering > Hierarchical Cluster**.

Figure 12.4 Hierarchical Cluster Launch Dialog



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns The variables used for clustering observations.

Ordering Sorts clusters by their mean values based on the specified column.

> **Tip:** Use the first principal component obtained by conducting a principal components analysis as an Ordering column. The clusters are sorted by these values.

Attribute ID (Available only if **Data is stacked** is selected as the data structure.) Specifies the variables that are stacked.

Object ID (Available only if **Data are summarized** or **Data is stacked** is selected as the data structure.) A column or columns that provide a unique identifier for each unit for which measurements are stacked.

Label A column of values used to label the dendrogram in the report.

> **Note:** If the selected data structure is **Data is distance matrix**, the Label column must have a character data type.

By A column whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed. The results are presented in separate reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

The launch window has the following menus and options:

- "Clustering Method"
- "Method for Distance Calculation"
- "Data Structure"
- "Transformations to Y, Columns Variables"

Clustering Method

Hierarchical is the default clustering method, but the dialog enables you to switch to KMeans or Normal Mixtures. If you select KMeans or Normal Mixtures, when you click OK, a Control Panel appears where you can select any of the following as Method:

K-Means Clustering See "K Means Cluster".

Normal Mixtures See "Normal Mixtures".

Robust Normal Mixtures See "Normal Mixtures Report".

Self Organizing Map See "Self Organizing Map".

Method for Distance Calculation

Select a method used to calculate distances. For distance formulas, see "Distance Method Formulas".

Ward In Ward's minimum variance method, the distance between two clusters is the ANOVA sum of squares between the two clusters summed over all the variables. At each generation, the within-cluster sum of squares is minimized over all partitions obtainable by merging two clusters from the previous generation. The sums of squares are easier to interpret when they are divided by the total sum of squares to give the proportions of variance (squared semipartial correlations).

Ward's method joins clusters to maximize the likelihood at each level of the hierarchy under the assumptions of multivariate normal mixtures, spherical covariance matrices, and equal sampling probabilities.

Ward's method tends to join clusters with a small number of observations and is strongly biased toward producing clusters with approximately the same number of observations. It is also very sensitive to outliers. See Milligan (1980).

Average The distance between two clusters is the average distance between pairs of observations. Average linkage tends to join clusters with small variances and is slightly biased toward producing clusters with the same variance. See Sokal and Michener (1958).

Centroid The distance between two clusters is defined as the squared Euclidean distance between their means. The centroid method is more robust to outliers than most other hierarchical methods but in other respects might not perform as well as Ward's method or average linkage. See Milligan (1980).

Single The distance between two clusters is the minimum distance between an observation in one cluster and an observation in the other cluster. Single linkage has many desirable theoretical properties but has performed poorly in Monte Carlo studies. See Jardine and Sibson (1971), Fisher and Van Ness (1971), Hartigan (1981), and Milligan (1980). Single linkage was originated by Florek et al. (1951a, 1951b) and later reinvented by McQuitty (1957) and Sneath (1957).

By imposing no constraints on the shape of clusters, single linkage sacrifices performance in the recovery of compact clusters in return for the ability to detect elongated and irregular clusters. Single linkage tends to chop off the tails of distributions before separating the main clusters. See Hartigan (1981).

Complete The distance between two clusters is the maximum distance between an observation in one cluster and an observation in the other cluster. Complete linkage is strongly biased toward producing clusters with approximately equal diameters and can be severely distorted by moderate outliers. See Milligan (1980).

Fast Ward Applies an algorithm that computes Ward's method more quickly for large numbers of rows. The computation time is shorter because this algorithm does not require the calculation of a distance matrix. It is used automatically whenever there are more than 2,000 rows.

Data Structure

These options describe the form of the data that is used in calculating multivariate distances:

Data as usual Data that are rectangular with one row for each observation and one column for each variable.

Data as summarized Data that are summarized by the levels of one or more identifying columns. When you select this option, an Object ID text box appears in the launch window. Specify the identifying columns as the Object ID. The **Data as summarized** option calculates level means and treats these means as your input data.

Data is distance matrix Data that consist of distances between observations. For n observations, the distance table should have n rows and $n + 1$ columns. One column (usually the first) must contain a unique identifier for each of the n observations. The

remaining columns contain distances between that observation and the *n* observations. Note the following:

- The diagonal elements of the table should be zero or missing, because the distance between a point and itself is zero. Values that are not zero or missing are treated as zero, and a note appears in the report.
- The distance columns can be a symmetric square matrix, or they can be upper or lower triangular with missing entries in the lower or upper portion. If the distances are given as a square matrix, a warning appears in the report if the table is not symmetric.
- You can begin with a different data structure and then save a distance matrix. See "Save Distance Matrix".

  When you select the **Data is distance matrix** option, enter the distance columns as Y, Columns and the identifier column as Label. The Label column must have the Character data type. For an example, see "Example of a Distance Matrix".

Data is stacked Data that have a single response of interest and multiple rows for each object.

When you select the **Data is stacked** option, Attribute ID and Object ID text boxes appear in the launch window.

- Enter a *single* column as Y, Columns.
- Enter columns that describe groupings of the Y, Columns variable as Attribute ID. If only two columns are entered and if you select Add Spatial Measures, then you can add spatial components to be used in the cluster analysis. See "Add Spatial Measures".
- Enter the identifying columns for objects as Object ID.

  The analysis that is conducted is equivalent to splitting the Y, Column variable by the Attribute ID columns and then performing hierarchical clustering without standardizing the response columns.

Tip: Use this option together with the Add Spatial Measures option to perform two-dimensional spatial clustering. For example, wafer data are often recorded using one row for each die. Interest centers around clustering wafers. See "Example of Wafer Defect Classification Using Spatial Measures".

Caution: Because there is a single measurement column, the Standardize Data option is not appropriate for stacked data.

Not Enough Nonmissing Data Alert

The JMP alert **Not enough nonmissing data** can be difficult to understand when you are using the **Data as summarized** or **Data is stacked** data structures. The alert occurs in the following situations:

- For **Data as usua**l, when all rows or all but one row are missing at least one value for a Y, Columns variable.
- For **Data as summarized**, when your data are summarized across the Object ID columns, all rows or all but one row are missing at least one value of the summarized Y, Column variables. To see the data structure that the Cluster platform is analyzing, select Tables > Summary, enter the Object ID columns as Group and the Y, Columns variables as Statistics > Mean.
- For **Data is stacked**, when your data are split across the Attribute ID columns, all rows or all but one row are missing at least one value of the split Y, Column values. To see the data structure that the Cluster platform is analyzing, select **Tables > Split**, enter the Attribute ID columns as Split By, the Y, Columns variable as Split Columns, and the Object ID columns as Group.

Transformations to Y, Columns Variables

The following options specify the form of the Y, Columns variables to be used in the cluster analysis:

Standardize Data Addresses the issue of different measurement scales for continuous and ordinal columns. Except when the **Data is stacked** option is selected, the values in each column are standardized by subtracting the column mean and dividing by the column standard deviation. Deselect the Standardize Data check box if you do not want the cluster distances computed on standardized values.

Standardize Robustly Reduces the influence of outliers on estimates of the mean and standard deviation for continuous and ordinal columns. This option uses Huber M-estimates of the mean and standard deviation (Huber 1964; Huber 1973; Huber and Ronchetti 2009). For columns with outliers, this option gives the standardized values greater representation in determining multivariate distances.

Note: If both Standardize Data and Standardize Robustly are selected, each column is standardized by subtracting its robust column mean and dividing by its robust standard deviation. This option is useful when columns represent different measurement scales *or* when observations tend to be outliers in only specific dimensions.

Note: If Standardize Data is unchecked and Standardize Robustly is selected, the robust mean and robust standard deviation for the values in all columns combined are used to standardize each column. This option can be useful when columns all represent the same measurement scale *and* when observations tend to be outliers in all dimensions.

Missing value imputation Imputes missing values. If the number of variables is either 50 or less, or less than half the number of rows, multivariate normal imputation is used. Otherwise, multivariate SVD imputation is used.

Multivariate normal imputation calculates pairwise covariances to construct a covariance matrix for the response columns. Then each missing value is imputed by a method that is equivalent to regression prediction using all the predictors with no missing values for the given observation. If the constructed covariance matrix is not positive definite, missing values are imputed using their column means.

Multivariate SVD imputation avoids constructing a covariance matrix by using the singular value decomposition. See *Predictive and Specialized Modeling*.

Caution: Missing value imputation assumes that there are no clusters, that the data come from a single multivariate normal distribution, and that the values are missing completely at random. Because these assumptions are usually not reasonable in practice, use this feature with caution. However, the feature can produce more informative results than discarding most of your data.

Add Spatial Measures (Available only if **Data is stacked** is selected as the data structure.) Select the **Add Spatial Measures** option when your data are stacked and contain two attribute columns that correspond to spatial coordinates (horizontal and vertical coordinates, for example). This option opens a window in which you can select and weight spatial components to aid in clustering defect patterns. This is a specialty method and is applicable in only very specific settings. See "Spatial Measures" and "Example of Wafer Defect Classification Using Spatial Measures".

Hierarchical Cluster Report

The Hierarchical Cluster report displays the method used, a dendrogram, and the Clustering History table. If you assigned a column as a Label in the launch window, the column's values identify each observation in the dendrogram.

- "Dendrogram Report"
- "Illustration of Dendrogram and Distance Graph"
- "Clustering History Report"

Dendrogram Report

The dendrogram is a tree diagram that represents the agglomeration of observations into clusters. The dendrogram also gives information about the degree of dissimilarity of clusters.

The clustering process can be viewed by reading the dendrogram from left to right. Each step consists of combining the two *closest* clusters into a single cluster.

- The joining of clusters is indicated by horizontal lines that are connected by vertical lines.
- The horizontal position of the vertical line represents the distance between the two clusters that are most recently joined to form the specified number of clusters.

Note: When the number of observations is less than 256, the distances are proportional to the distances shown in the Distance Graph. Otherwise, Geometric Spacing is used. See "Dendrogram Scale".

You can perform the following tasks:

- Click and drag the diamond-shaped handle at either the top or bottom of the dendrogram to identify a given number of clusters.
- Click any cluster stem to select all the members of the cluster in the dendrogram and in the data table.

Distance Graph

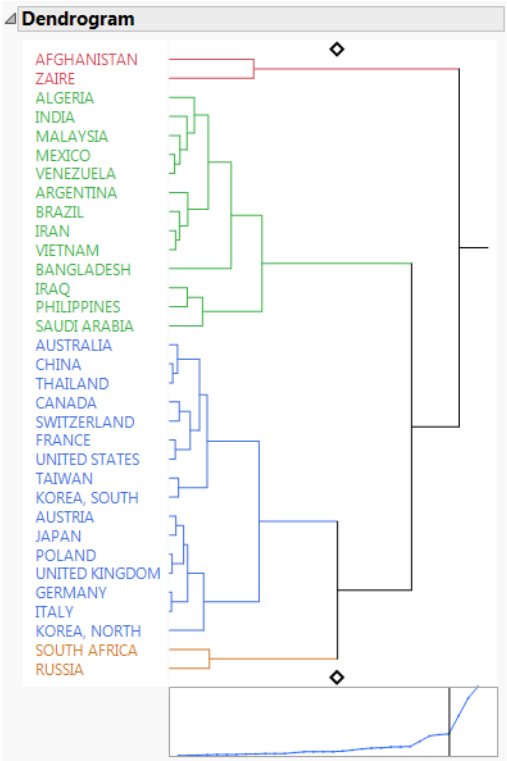
The Distance Graph is the plot that appears beneath the dendrogram. This graph has a point for each step where two clusters are joined into a single cluster. The horizontal coordinates represent the numbers of clusters, which decrease from left to right. The vertical coordinate of the point is the distance between the clusters that were joined at the given step.

You can click and drag either diamond-shaped handle in the dendrogram to control the chosen number of clusters. When you click and drag the diamond, a vertical line appears in the plot that moves to correspond to the number of clusters. Often there is a point where the slope of the distance graph levels off. Such a point suggests a natural break and helps you determine the number of clusters.

Illustration of Dendrogram and Distance Graph

Consider the dendrogram report for Birth Death Subset.jmp in "Example of Clustering".

Figure 12.5 Dendrogram Report for Birth Death Subset.jmp



The diamonds are set at four clusters. The two clusters that are most recently joined to form the four cluster model are the cluster consisting of Algeria to Bangladesh and the cluster consisting of Iraq to Saudi Arabia. The distance between these two clusters is the point on the distance plot indicated by the vertical line when the diamond is set to 4. The distance is given in the Clustering History report next to Number of Clusters equal to 4. There, it is shown that the distance is 1.618708760 and that clusters beginning with Algeria and Iraq are combined to yield four clusters.

The two clusters that are combined to yield five clusters are the cluster that consists of Australia to Korea, South, and the cluster that consists of Austria to Korea, North. The vertical join line in the dendrogram for these clusters is at about the same horizontal distance from the left as the vertical join line for the clusters that were joined to form the four-cluster model, Algeria to Bangladesh and Iraq to Saudi Arabia. It follows that there is not much difference in terms of distance between the clusters joined to form the four-cluster model and those joined to form the five-cluster model.

The fact that the distance plot levels off starting with the four-cluster model indicates that the cluster groupings up to that point do not account for much distance between clusters. However, the four-cluster model shows good separation between clusters.

Clustering History Report

The Clustering History table contains the clustering history.

Number of Clusters Lists the numbers of clusters that result *after* the joining indicated by the Leader and Joiner is performed. The number of clusters begins with the first join, when there are *n - 1* clusters, where *n* is the number of objects. The report lists the number of clusters in decreasing order until all objects are contained in one cluster. In this way, the Clustering History follows the order of the dendrogram from left to right.

Distance The distance between clusters, calculated according to the distance method that you select on the launch window. See "Method for Distance Calculation".

Leader A representative of the first cluster in the dendrogram being joined. The cluster order and the representative shown in the Leader column is a consequence of how the data are sorted and has no intrinsic meaning.

Joiner A representative of the second cluster in the dendrogram being joined. The cluster order and the representative shown in the Joiner column is a consequence of how the data are sorted and has no intrinsic meaning.

Hierarchical Cluster Options

The Hierarchical Clustering red triangle menu contains the following options:

Color Clusters Colors the labels for dendrogram and their associated join bars according to cluster membership. Also assigns the corresponding colors to the rows of the data table The colors update if you change the number of clusters. If you deselect this option, the colors are no longer updated based on the number of clusters.

Mark Clusters Assigns markers to the rows of the data table corresponding to the cluster to which the row belongs. The markers update if you change the number of clusters. If you deselect this option, the markers are no longer updated based on the number of clusters.

Number of Clusters Prompts you to enter a number of clusters and positions the dendrogram slider to that number.

Cluster Criterion Gives the Cubic Clustering Criterion (CCC) for the entire range of number of clusters. The CCC is used to estimate the number of clusters. It can be used with any distance-based clustering algorithm. Larger values of the CCC indicate better fit in terms of number of clusters. See SAS Institute Inc. (1983). (Not available when **Data is distance matrix** is selected.)

Show Dendrogram Shows or hides the Dendrogram report.

Dendrogram Scale Contains the following options for scaling the dendrogram:

Distance Scale Shows the horizontal distances between any two join points as the distances between the two clusters joined at that point, based on the distance method specified on the launch window. The distance scale is the same scale as used in the Distance Graph and is the default scale for the dendrogram.

Even Spacing Shows the horizontal distances between any two join points as equal.

Geometric Spacing Increases the horizontal distances between join points as the number of clusters increases. This option is useful when there are many objects and you want the smaller clusters to be more visible than the larger clusters.

Distance Graph Shows or hides the distance plot beneath the dendrogram.

Show NCluster Handle Shows or hides the handles on the dendrogram used to manually change the number of clusters.

Zoom to Selected Rows Selects and enlarges a particular cluster after you select the cluster in the dendrogram. Alternatively, you can double-click the cluster to zoom in on it. Use Release Zoom to return to the original view.

Release Zoom Returns the dendrogram to the original view after zooming.

Pivot on Selected Cluster Reverses the order of the two sub-clusters of the currently selected cluster.

Color Map Gives the option to add a color map, or heat map, showing each Y, Column variable colored by value. Several color theme choices are available in a submenu.

Two Way Clustering Clusters by the variables specified in Y, Columns as well as rows. A color map is added with a dendrogram for the Y, Column variables at its base. Typically, for two-way clustering, your variables are measured on the same scale and you do not select Standardize Data. (Not available when **Data is stacked** is selected.)

Positioning Provides options for changing the positions of labels and other parts of the dendrogram.

Legend Shows or hides a legend for the colors used in color maps. This option is available only if a color map is enabled.

More Color Map Columns Adds a color map for specified columns. (Not available when **Data as summarized**, **Data is distance matrix**, or **Data is stacked** is selected.)

Constellation Plot Shows or hides an alternative way to present the information in the hierarchical clustering dendrogram. Each observation (row) is represented by an endpoint and each cluster join is represented by a new point. The lines that are drawn represent

cluster membership. The lengths of the lines represent the distance between clusters. Longer lines represent greater distances between clusters.

You can hover over the lines in the constellation plot to see their length. However, the length values are meaningful only with respect to each other. The axis scaling, orientation of points, and angles of the lines are arbitrary. They are determined such that the ends of the nodes are spaced out and the plot does not appear cluttered, which is important with larger data sets.

To turn off the labels on the endpoints, right-click inside the Constellation Plot and deselect **Show Labels**.

Save Constellation Coordinates Saves the coordinates of the constellation plot to the data table. (Not available when **Data as summarized**, **Data is distance matrix**, or **Data is stacked** is selected.)

Save Clusters Creates a data table column that contains the cluster number. If Add Spatial Measures is selected on the launch window, the cluster numbers are also saved to the Hough Data Table.

Save Formula for Closest Cluster Creates a data table column that contains a formula for the closest cluster. This option calculates the squared Euclidean distance to each cluster's centroid and selects the cluster that is closest. Note that this formula does not always reproduce the cluster assignment given by Hierarchical Clustering since the clusters are determined differently. However, the cluster assignment is very similar. (Not available when **Data as summarized**, **Data is distance matrix**, or **Data is stacked** is selected.)

Save Display Order Creates a data table column that contains the order in which the row appears in the dendrogram.

Save Cluster Hierarchy Creates a data table that contains the information needed to write a script for a custom dendrogram. For each cluster join, there are three rows: the first for the joiner, the second for the leader, and the third for the result, giving the cluster centers, size, and other information.

Save Cluster Tree Creates a new data table that contains information needed to compare cluster trees between JMP and SAS. For each cluster join, there is one row for each new cluster, with the cluster's size and other information.

Save Distance Matrix Creates a new data table that contains the distances between the observations.

Save Cluster Means Creates a new data table that contains the number of rows and the means of each column in each cluster.

Cluster Summary (Not available when **Data is distance matrix** is selected.) Displays the following information:

Cluster Means A table that gives, for each cluster, the number of observations (or Object IDs, if the data are stacked) and means for each variable.

Cluster Standard Deviations A table that gives, for each cluster, the number of observations (or Object IDs, if the data are stacked) and standard deviations for each variable.

Cluster Means Plot Either a parallel plot or a two-dimensional heat map of the cluster means.

The plot is a parallel plot unless **Data is stacked** is selected and there are two Attribute ID variables. For the parallel plot, the axis for each variable is scaled.

– If Standardize Data were selected, the axis ranges from two standard deviations above and below the mean, where the standard deviation and mean are computed for the raw data. If a cluster mean falls beyond this range, the axis is extended to include it.

– If Standardize Data were not selected, there is a common vertical axis whose scaling is displayed. (The scaling is equivalent to the Scale Uniformly option in Graph Builder).

When **Data is stacked** is selected and there are two Attribute ID variables, two-dimensional plots of the mean of the Y variable at each location are shown for each cluster. These plots are colored using a Blue to Gray to Red color gradient.

Column Summary For each variable, gives the RSquare value that represents the proportion of variation explained by the clusters. This number is the RSquare value for a regression of the variable on the clusters. The option also gives a bar graph of RSquare values.

Scatterplot Matrix Creates a scatterplot matrix using all the variables. (Not available when **Data as summarized**, **Data is distance matrix**, or **Data is stacked** is selected.)

Parallel Coord Plots Creates a parallel coordinate plot for each cluster. (Not available when **Data as summarized**, **Data is distance matrix**, or **Data is stacked** is selected.) The axes are scaled as described for the Cluster Means Plot. See "Cluster Means Plot".

Cluster Treatment Comparisons (Available only if you hold Shift and click the Hierarchical Clustering red triangle.) Select a response column and a two-level treatment column. Creates a Hierarchically Clustered Differences report.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Examples of the Hierarchical Clustering Platform

- "Example of a Distance Matrix"
- "Example of Wafer Defect Classification Using Spatial Measures"

Example of a Distance Matrix

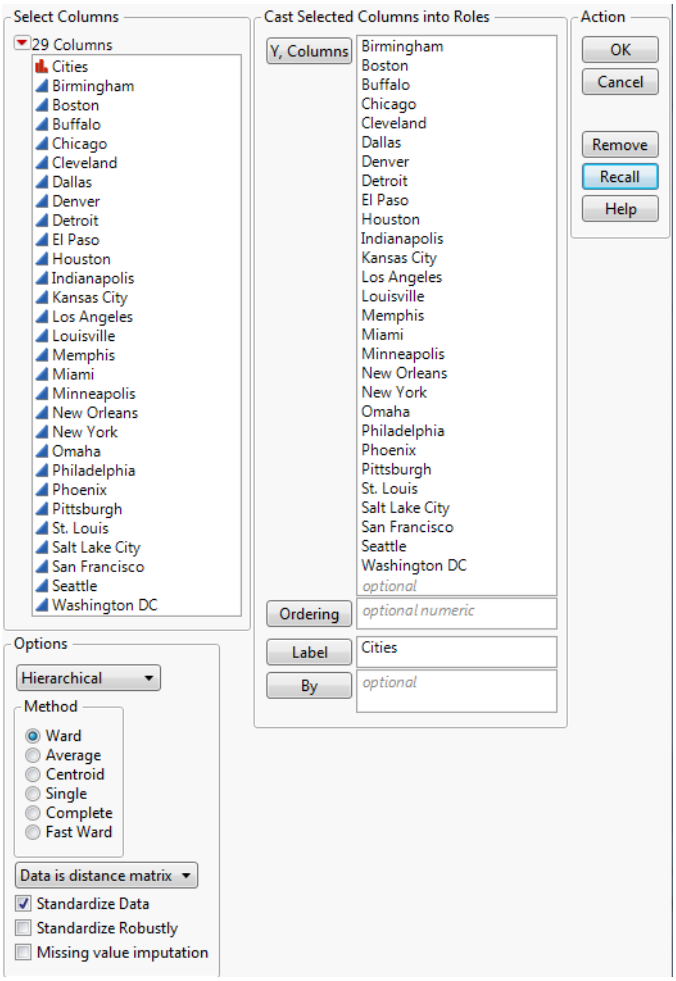
The proper data table structure for a distance matrix consists of the following:

- An identifier column (usually the first column) that has a Character data type.
- A set of *n* columns, where *n* is also the number of rows. These *n* columns define a symmetric matrix with zero or missing values on the diagonal.

Notice that the distance matrix in Flight Distances.jmp follows the preceding format.

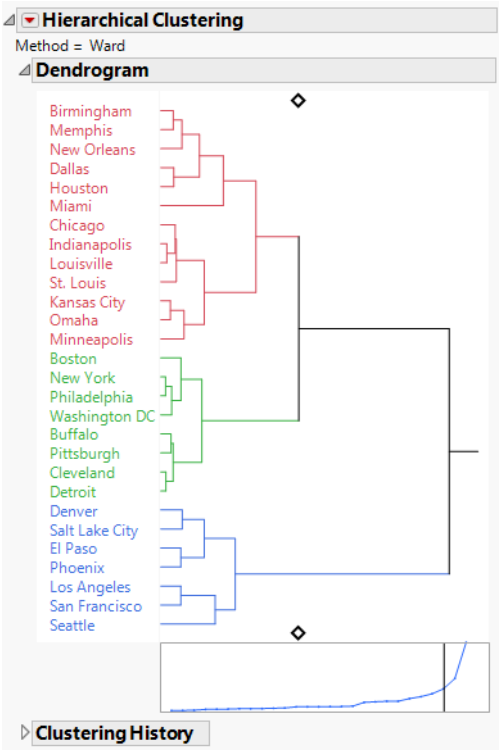
1. Select **Help > Sample Data Library** and open Flight Distances.jmp.
2. Select **Analyze > Clustering > Hierarchical Cluster**.
3. In the list at the bottom left corner of the launch window, change **Data as usual** to **Data is distance matrix**.
4. Select Cities and click **Label**.
5. Select all remaining columns and click **Y, Columns**.

Figure 12.6 Completed Distance Matrix Launch Window



6. Click **OK**.
7. Click the Hierarchical Clustering red triangle and select **Color Clusters**.

Figure 12.7 Dendrogram Report for Flight Distances



In the Dendrogram report for the flight distances, the placement of the diamonds indicates that the model has grouped the cities into three clusters. The clusters are color-coded on the dendrogram. For more information about how to interpret the report, see "Dendrogram Report".

Example of Wafer Defect Classification Using Spatial Measures

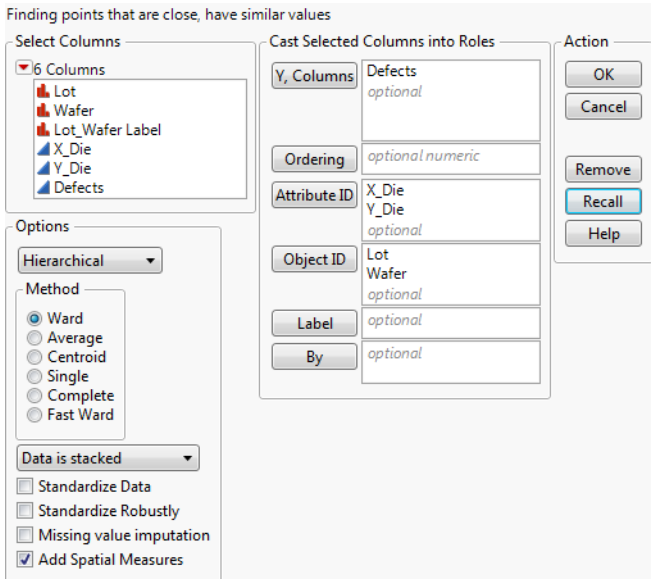
A specialty clustering option called Spatial Measures is available in the Hierarchical Cluster platform. In this example, you use this option to cluster wafers. For information about the option, see "Spatial Measures".

1. Select **Help > Sample Data Library** and open Wafer Stacked.jmp.
2. Select **Analyze > Clustering > Hierarchical Cluster**.
3. In the list in the lower left corner, change **Data as usual** to **Data is stacked**.

   Additional options for stacked data appear in the launch window.
4. Select Defects and click **Y, Columns**.
5. Select X_Die and Y_Die and click **Attribute ID**.

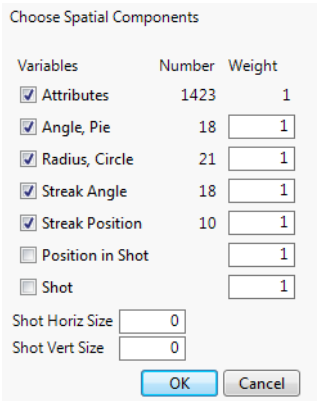
6. Select Lot and Wafer and click **Object ID**.
7. Select **Add Spatial Measures** from the list of options in the lower left corner.

Figure 12.8 Completed Clustering Launch Window



8. Click **OK**.

Figure 12.9 Spatial Components Window



Because Defects is measured at 1423 locations, there are 1423 Attributes variables.

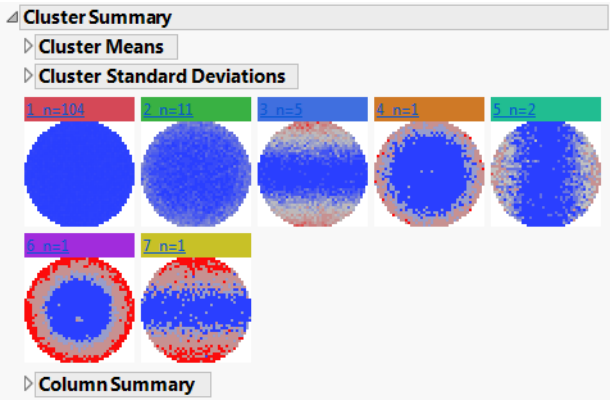
9. Click **OK** to accept the selections in the Spatial window.

   Two windows open: the Hierarchical Clustering report and the Wafer Stacked Defects Spatial data table.

10. In the Dendrogram plot, click and drag the diamond-shaped handle at the top to explore various numbers of clusters.

    As you drag the handle, the vertical line in the distance graph below the dendrogram moves to the corresponding number of clusters. The vertical coordinate gives the distance between the clusters that were joined at the given step. The graph seems to level off when the number of clusters is 7.
11. Click the Hierarchical Clustering red triangle and select **Number of Clusters**.
12. Enter 7 and click **OK**.
13. Click the Hierarchical Clustering red triangle and select **Cluster Summary**.

Figure 12.10 Cluster Summary Report



The wafer maps indicate the spatial nature of defects for each cluster. Cluster 1 contains 104 wafers with relatively few defects that are spread throughout the wafers. Cluster 3 has 5 wafers with defects concentrated at the extremes of the top and bottom hemispheres. You can view the maps for individual wafers and their Hough space maps in the data table produced by the cluster analysis. See "Spatial Measures".

Statistical Details for the Hierarchical Clustering Platform

- "Spatial Measures"
- "Distance Method Formulas"

Spatial Measures

To use the Add Spatial Measures option, your data must be stacked and contain two attribute columns that correspond to spatial coordinates. Some of the spatial measures are constructed using the Hough transform. See White et al. (2008) and Ballard (1981). See "Example of Wafer Defect Classification Using Spatial Measures".

Choose Spatial Components Window

The Choose Spatial Components window appears if you do the following in the launch window:

- Select the Data is stacked data structure
- Specify two columns as Attribute ID that correspond to spatial coordinates
- Specify an Object ID
- Select Add Spatial Measures

In the Choose Spatial Components window, you select and weight spatial components for your cluster analysis. These components are used to construct the variables used in the cluster analysis. A new table with a row for each object opens. This table contains the calculated spatial components for each object.

Variables The types of variables that are constructed and used in the cluster analysis. The variables are constructed using spatial components and the response, Y.

Attributes The value of the Y variable calculated at each location for each object, as defined by the two Attribute ID variables.

Angle, Pie Variables that reflect wedge shapes or hemispherical shapes.

Radius, Circle The variables that reflect circular shapes.

Streak Angle The variables that reflect streaks that have the same angle.

Streak Position The variables that reflect streaks with the same spatial position.

Position in Shot The variables that are based on the position of the die in the shot. Position in Shot variables are represented as ShotPos[vShotSize, hShotSize], where vShotSize and hShotSize are the defined vertical and horizontal shot sizes.

Shot The variables that identify which rectangle an object is in, where you specify the number of horizontal and vertical positions of objects in the rectangle. The term *shot* is used in semiconductor wafer data to identify which dies are imaged together across a wafer.

Enter values for Shot Horizontal Size and Shot Vertical Size. Specifying a horizontal shot size of 4 and a vertical shot size of 5 indicates that there are up to 20 dies in a shot. The total number of identifiers created is calculated as follows:

floor[(hSize+hShotSize-1)/hShotSize]* floor[(vSize+vShotSize-1)/vShotSize]

where hSize and vSize are the maximum numbers of horizontal and vertical positions, respectively, hShotSize = Shot Horizontal Size, and vShotSize = Shot Vertical Size.

Note: Shot variables are represented as Shot[vert, horiz], where vert and horiz represent the vertical and horizontal die locations, respectively.

Number The total number of variables of the given type that are constructed.

Weight A measure of importance for the given type of variable used in determining the clusters.

Spatial Measures Reports

When you click OK in the Choose Spatial Components window, two windows appear.

Hierarchical Clustering Report

When you conduct an analysis with stacked data and two Attribute IDs, the Cluster Summary report shows spatial maps of the Y variable. Each plot is a two-dimensional plot that displays the cluster mean for each location defined by the Attribute ID variables. The plot uses a Blue to Gray to Red color gradient with a Quantile scale. Using the quantile scale mitigates the effect of outliers.

Spatial Data Table

The data table for Spatial measures has a row for each unique Object ID. Columns are displayed using a Blue to Gray to Red default color gradient to show the Y variable. The table contains the following columns:

Object An expression column that shows a heat map of the Y variable at each spatial location defined by the two Attribute ID variables.

Hough An expression column that shows a heat map of the Hough space for each object. See White et al. (2008).

Spatial Measures A column for each spatial measure that shows the computed values for each object. Cells are colored by value.

Distance Method Formulas

This section provides the formulas used in calculating distances based on the Method that you select on the launch window. For a description of the methods, see "Method for Distance Calculation".

The formulas use the following notation, where lowercase symbols generally pertain to observations and uppercase symbols to clusters:

n is the number of observations

v is the number of variables

x_i is the ith observation

C_K is the Kth cluster, subset of {1, 2,..., n}

N_K is the number of observations in C_K

$\bar{x}$ is the sample mean vector

$\bar{x}_K$ is the mean vector for cluster C_K

$\|x\|$ is the square root of the sum of the squares of the elements of **x** (the Euclidean length of the vector **x**)

$d(x_i, x_j)$ is $\|x_i - x_j\|^2$

Average Linkage The distance for the average linkage cluster method is:

$$D_{KL} = \sum_{i \in C_K} \sum_{j \in C_L} \frac{d(x_i, x_j)}{N_K N_L}$$

Centroid Method The distance for the centroid method of clustering is:

$$D_{KL} = \left\| \bar{x}_K - \bar{x}_L \right\|^2$$

Ward's The distance for Ward's method is:

$$D_{KL} = \frac{\left\| \overline{x}_K - \overline{x}_L \right\|^2}{\frac{1}{N_K} + \frac{1}{N_L}}$$

Single Linkage The distance for the single linkage cluster method is:

$$D_{KL} = \min_{i \in C_K} \min_{j \in C_L} d(x_i, x_j)$$

Complete Linkage Distance for the Complete linkage cluster method is:

$$D_{KL} = \max_{i \in C_K} \max_{j \in C_L} d(x_i, x_j)$$

Chapter 13

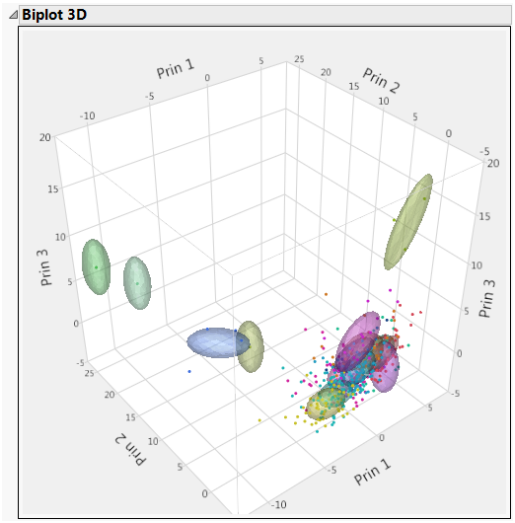
K Means Cluster

Group Observations Using Distances

Use the K Means Cluster platform to group observations that share similar values across a number of variables. Use the *k*-means method with larger data tables, ranging from approximately 200 to 100,000 observations.

The K Means Cluster platform constructs a specified number of clusters using an iterative algorithm that partitions the observations. The method, called *k-means*, partitions observations into clusters so as to minimize distances to cluster centroids. You must specify the number of clusters, *k*, in advance. However, you can compare the results of different values of *k* to select an optimal number of clusters for your data.

Figure 13.1 3D Biplot



Contents

Overview of the K Means Cluster Platform 303
Overview of Platforms for Clustering Observations 303
Example of K Means Cluster 305
Launch the K Means Cluster Platform 308
Iterative Clustering Report 309
Iterative Clustering Options 309
Iterative Clustering Control Panel 310
K Means Report 311
Cluster Comparison Report 311
K Means Report 311
K Means Report Options 312
Self Organizing Map 314
Self Organizing Map Control Panel 314
Self Organizing Map Report 315
Description of SOM Algorithm 315
Additional Example of K Means Cluster Platform 316
Example of a Self-Organizing Map 316

Overview of the K Means Cluster Platform

K Means Cluster is one of four platforms that JMP provides for clustering observations. For a comparison of all four methods, see "Overview of Platforms for Clustering Observations".

The K Means Cluster platform forms a specified number of clusters using an iterative fitting process. The *k*-means algorithm first selects a set of *k* points called *cluster seeds* as an initial guess for the means of the clusters. Each observation is assigned to the nearest cluster seed to form a set of temporary clusters. The seeds are then replaced by the cluster means, the points are reassigned, and the process continues until no further changes occur in the clusters.

The *k*-means algorithm is a special case of the *EM algorithm*, where *E* stands for Expectation, and *M* stands for maximization. In the case of the *k*-means algorithm, the calculation of temporary cluster means represents the Expectation step, and the assignment of points to the closest clusters represents the Maximization step.

K-Means clustering supports only numeric columns. *K*-Means clustering ignores modeling types (nominal and ordinal) and treats all numeric columns as continuous.

You must specify the number of clusters, *k*, or a range of values for *k*, in advance. However, you can compare the results of different values of *k* to select an optimal number of clusters for your data.

For background on *K*-Means clustering, see the FASTCLUS Procedure chapter in SAS Institute Inc. (2020d) and Hastie et al. (2009).

Overview of Platforms for Clustering Observations

Clustering is a multivariate technique that groups together observations that share similar values across a number of variables. Typically, observations are not scattered evenly through *p*-dimensional space, where *p* is the number of variables. Instead, the observations form clumps, or clusters. Identifying these clusters provides you with a deeper understanding of your data.

Note: JMP also provides a platform that enables you to cluster variables. See "Cluster Variables".

JMP provides four platforms that you can use to cluster observations:

- Hierarchical Cluster is useful for smaller tables with up to several tens of thousands of rows and allows character data. Hierarchical clustering combines rows in a hierarchical sequence that is portrayed as a tree. You can choose the number of clusters that is most appropriate for your data after the tree is built.

- K Means Cluster is appropriate for larger tables with up to millions of rows and allows only numerical data. You need to specify the number of clusters, *k*, in advance. The algorithm guesses at cluster seed points. It then conducts an iterative process of alternately assigning points to clusters and recalculating cluster centers.
- Normal Mixtures is appropriate when your data come from a mixture of multivariate normal distributions that might overlap and allows only numerical data. For situations where you have multivariate outliers, you can use an outlier cluster with an assumed uniform distribution.

  You need to specify the number of clusters in advance. Maximum likelihood is used to estimate the mixture proportions and the means, standard deviations, and correlations jointly. Each point is assigned a probability of being in each group. The EM algorithm is used to obtain estimates.
- Latent Class Analysis is appropriate when most of your variables are categorical. You need to specify the number of clusters in advance. The algorithm fits a model that assumes a multinomial mixture distribution. A maximum likelihood estimate of cluster membership is calculated for each observation. An observation is classified into the cluster for which its probability of membership is the largest.

Table 13.1 Summary of Clustering Methods

Method	Data Type or Modeling Type	Data Table Size	Specify Number of Clusters
Hierarchical Cluster	Any	With Fast Ward, up to 200,000 rows With other methods, up to 5,000 rows	No
K Means Cluster	Numeric	Up to millions of rows	Yes
Normal Mixtures	Numeric	Any size	Yes
Latent Class Analysis	Nominal or Ordinal	Any size	Yes

Some of the clustering platforms have options to handle outliers in the data. However, if your data has outliers, it is best to explore them first prior to analyzing. This can be done using the Explore Outliers Utility. For more information, see the Modeling Utilities chapter in *Predictive and Specialized Modeling*.

Example of K Means Cluster

In this example, you use the Cytometry.jmp sample data table to cluster observations using K Means Cluster. Cytometry is used to detect markers of the surface of cells and the readings from these markers help diagnose certain diseases. In this example, the observations are grouped based on readings of four markers in a cytometry analysis.

1. Select **Help > Sample Data Library** and open Cytometry.jmp
2. Select **Analyze > Clustering > K Means Cluster**.
3. Select CD3, CD8, CD4, and MCB and click **Y, Columns**.
4. Click **OK**.
5. Enter 3 next to **Number of Clusters.**
6. Enter 15 next to **Range of Clusters** (Optional).

   Because the Range of Clusters is set to 15, the platform provides fits for 3 to 15 clusters. You can then determine your preferred number of clusters.
7. Click **Go**.

Figure 13.2 Cluster Comparison Report

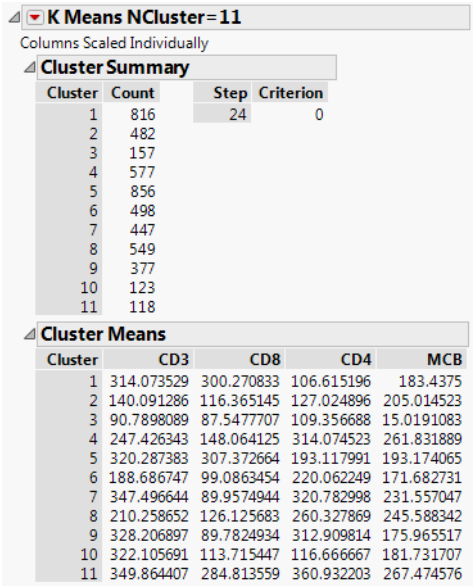
Cluster Comparison

Method	NCluster	CCC	Best
K-Means Clustering	3	23.1784	
K-Means Clustering	4	8.80709	
K-Means Clustering	5	29.5123	
K-Means Clustering	6	52.5517	
K-Means Clustering	7	49.5876	
K-Means Clustering	8	56.5308	
K-Means Clustering	9	54.053	
K-Means Clustering	10	69.8707	
K-Means Clustering	11	70.5239	Optimal CCC
K-Means Clustering	12	61.5326	
K-Means Clustering	13	68.1277	
K-Means Clustering	14	66.4044	
K-Means Clustering	15	69.9928	

The Cluster Comparison report appears at the top of the report window. The best fit is determined by the highest CCC value. In this case, the best fit occurs when you fit 11 clusters.

8. Scroll to the K Means NCluster=11 report.

Figure 13.3 K Means NCluster=11 Report



K Means NCluster=11

Columns Scaled Individually

Cluster Summary

Cluster	Count	Step	Criterion
1	816	24	0
2	482		
3	157		
4	577		
5	856		
6	498		
7	447		
8	549		
9	377		
10	123		
11	118		

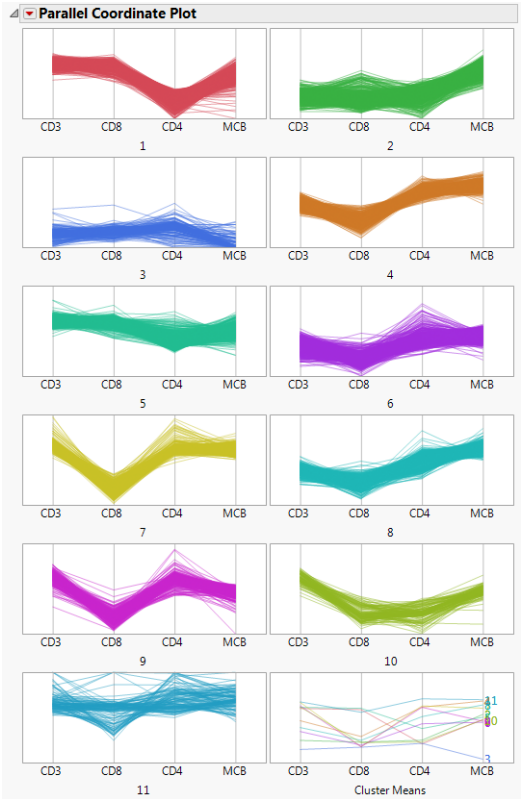
Cluster Means

Cluster	CD3	CD8	CD4	MCB
1	314.073529	300.270833	106.615196	183.4375
2	140.091286	116.365145	127.024896	205.014523
3	90.7898089	87.5477707	109.356688	15.0191083
4	247.426343	148.064125	314.074523	261.831889
5	320.287383	307.372664	193.117991	193.174065
6	188.686747	99.0863454	220.062249	171.682731
7	347.496644	89.9574944	320.782998	231.557047
8	210.258652	126.125683	260.327869	245.588342
9	328.206897	89.7824934	312.909814	175.965517
10	322.105691	113.715447	116.666667	181.731707
11	349.864407	284.813559	360.932203	267.474576

The Cluster Summary report shows the number of observations in each of the eleven clusters. The Cluster Means report shows the means of the four marker readings for each cluster.

9. Click the K Means NCluster=11 red triangle and select **Parallel Coord Plots**.

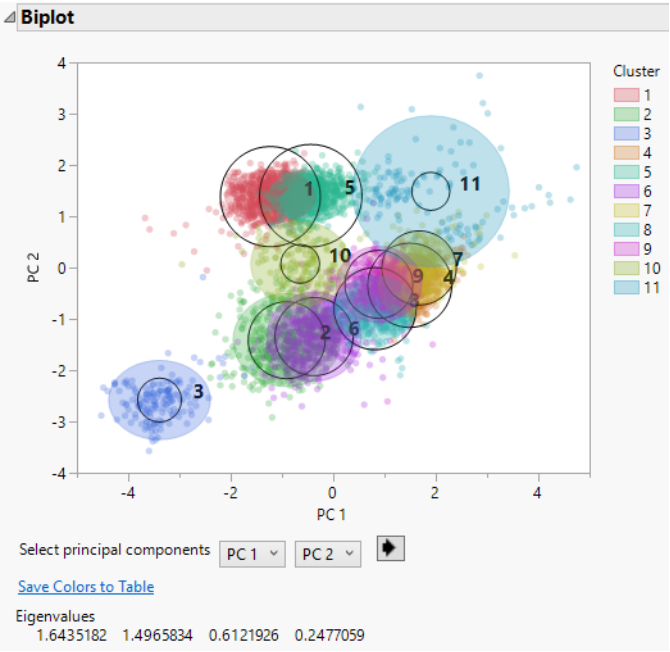
Figure 13.4 Parallel Coordinate Plots for Cytometry Data



The Parallel Coordinate Plots display the structure of the observations in each cluster. Use these plots to see how the clusters differ. Clusters 4, 6, 7, 8, and 9 tend to have comparatively low CD8 values and high CD4 values. Cluster 1, on the other hand, has higher CD8 values and lower CD4 values.

10. Click the K Means NCluster=11 red triangle and select **Biplot**.

Figure 13.5 Biplot for Cytometry Data

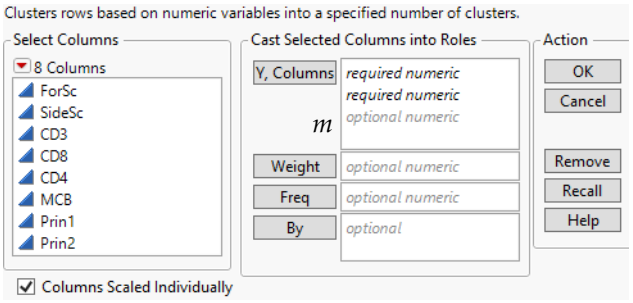


A legend that identifies the colors of the clusters is shown to the right of the plot. The clusters that appear to be most separated from the others based on their first two principal components are clusters 3, 10, and 11. This is supported by their parallel coordinate plots in Figure 13.4, which differ from the plots for the other clusters. Use the list below the plot to see the biplot for other combinations of principal components.

Launch the K Means Cluster Platform

Launch the K Means Cluster platform by selecting **Analyze > Clustering > K Means Cluster**.

Figure 13.6 K Means Cluster Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns The variables used for clustering observations.

> **Note:** *K*-Means clustering supports only numeric columns.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

By A column whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed. The results are presented in separate reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Launch Window Options

Columns Scaled Individually Scales each column independently of the other columns. Use when variables do not share a common measurement scale, and you do not want one variable to dominate the clustering process. For example, one variable might have values that are between 0 and 1000, and another variable might have values between 0 and 10. In this situation, you can use the option so that the clustering process is not dominated by the first variable.

When you click OK, a Control Panel appears. See "Iterative Clustering Control Panel".

Iterative Clustering Report

When you click OK in the launch window, the Iterative Clustering report window appears, showing a Control Panel for fitting models. See "Iterative Clustering Control Panel". As you fit models, additional reports are added to the window. See "K Means Report".

Iterative Clustering Options

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

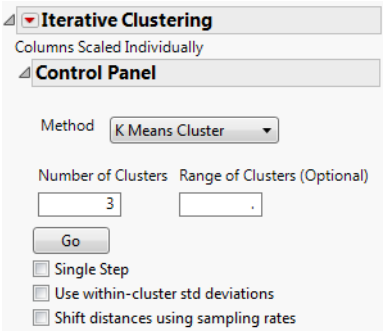
Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Iterative Clustering Control Panel

The Control Panel for the Cytometry.jmp data table is shown in Figure 13.7. You can iteratively fit different numbers of clusters or you can specify a range using the Range of Clusters option.

Figure 13.7 Iterative Clustering Control Panel



The Control Panel has the following options:

Method The following clustering methods are available:

KMeans Clustering Described in this chapter.

Self Organizing Map Described in "Self Organizing Map Control Panel".

Number of Clusters Designates the number of clusters to form.

Range of Clusters (Optional) Provides an upper bound for the number of clusters to form. If a number is entered here, the platform creates separate analyses for every integer between Number of Clusters and the value entered as Range of Clusters (Optional).

Go Unless Single Step is selected, fits the clusters automatically.

Single Step Enables you to step through the clustering process one iteration at a time. When you select Single Step and click Go, a K Means Cluster report appears with no cluster assignments but containing a Go and a Step button.

– Click the Step button to step through the iterations one at a time.

– Click the Go button to fit the clusters automatically.

Use within-cluster std deviations Scales distances using the estimated standard deviation of each variable for observations within each cluster. If you do not select this option, distances are scaled by an overall estimate of the standard deviation of each variable.

Shift distances using sampling rates Adjusts distances based on the sizes of clusters. If you have unequally sized clusters, an observation should have a higher probability of being assigned to larger clusters because there is a higher prior probability that the observation comes from a larger cluster.

K Means Report

When you click Go in the Control Panel, the following reports appear:

- A Cluster Comparison report. See "Cluster Comparison Report".
- One or more K Means reports. The reports are dynamically named K Means NCluster=<k> depending on, *k*, the number of clusters fit. A K Means NCluster=<k> report appears for each fit that you conduct.

The Cluster Comparison report and the KMeans NCluster=11 report for the Cytometry.jmp data table, with the variables CD3 through MCB as Y, Columns, are shown in Figure 13.2 and "K Means NCluster=11 Report".

Cluster Comparison Report

The Cluster Comparison report gives fit statistics to compare the various models. The fit statistic is the Cubic Clustering Criterion (CCC). Larger values of CCC indicate better fit. The best fit is indicated with the designation Optimal CCC in a column called Best. See SAS Institute Inc. (1983). Constant columns are not included in the CCC calculation.

K Means Report

Each K Means report gives the following summary statistics for each cluster:

- The Cluster Summary report gives the number of clusters and the observations in each cluster, as well as the number of iterations required.
- The Cluster Means report gives means for the observations in each cluster for each variable.
- The Cluster Standard Deviations report gives standard deviations for the observations in each cluster for each variable.

K Means Report Options

Each K Means report contains the following options:

Biplot Shows a plot of the points and clusters in the first two principal components of the data, along with a legend identifying the cluster colors. Circles are drawn around the cluster centers and the size of the circles is proportional to the count inside the cluster. The shaded area is the density contour around the mean. By default, this area indicates where 90% of the observations in that cluster would fall (Mardia et al. 1980). Use the list below the plot to change the plot axes to other principal components. Alternatively, use the arrow button to cycle through all possible axes combinations. An option to save the cluster colors to the data table is also located below the plot. See "Save Colors to Table". The eigenvalues are shown in decreasing order.

Note: If Columns Scaled Individually is checked in the launch window, the biplot uses a correlation matrix. If Columns Scaled Individually is not checked, the biplot uses a covariance matrix.

Biplot Options Contains the following options for controlling the appearance of the Biplot:

Show Biplot Rays Shows the biplot rays. The labeled rays show the directions of the covariates in the subspace defined by the principal components. They represent the degree of association of each variable with each principal component.

Biplot Ray Position Enables you to specify the position and radius scaling of the biplot rays. By default, the rays emanate from the point (0,0). In the plot, you can drag the rays or use this option to specify coordinates. You can also adjust the scaling of the rays to make them more visible with the radius scaling option.

Biplot Contour Density Enables you to specify the confidence level for the density contours. The default confidence level is 90%.

Mark Clusters Assigns markers that identify the clusters to the rows of the data table.

Biplot 3D Shows a three-dimensional biplot of the data. Available only when there are three or more variables.

Parallel Coord Plots Creates a parallel coordinate plot for each cluster. The plot report has options for showing and hiding the data and means. See *Essential Graphing*.

Scatterplot Matrix Creates a scatterplot matrix using all of the Y variables.

Save Colors to Table Assigns colors that identify the clusters to the rows of the data table. If there is a Biplot in the report window, the colors saved to the data table match the colors of the clusters in the Biplot. If the colors are changed in the Biplot and the

Save Colors To Table option is selected again, the colors in the table update to match those in the Biplot.

Note: When any of the Save options are selected, each saved column contains a Notes column property that specifies the number of clusters for that particular column's data. This enables you to save columns from more than one cluster fit and use the column property to identify which clustering fit the saved column is from.

Save Clusters Saves the following two columns to the data table:

- The Cluster column contains the number of the cluster to which the given row is assigned.
- (Not available for Self Organizing Maps.) The `Distance` column contains the squared Euclidean distance between the given observation and its cluster mean. For each variable, the difference between the observation's value and the cluster mean on that variable is divided by the overall standard deviation for the variable. These scaled differences are squared and summed across the variables.

Save Cluster Distance (Not available for Self Organizing Maps.) Saves a Distance column to the data table. This column is the same as the Distance column obtained from the **Save Clusters** option.

Save Cluster Formula Saves a formula column called Cluster Formula to the data table. This is the formula that identifies cluster membership for each.

Save Distance Formula (Not available for Self Organizing Maps.) Saves a formula column called Distance Formula to the data table. This is the formula that calculates the distance to the assigned cluster.

Save K Cluster Distances (Not available for Self Organizing Maps.) Saves *k* columns containing the squared Euclidean distances to each cluster center.

Save K Distance Formulas (Not available for Self Organizing Maps.) Saves *k* columns containing the formulas for the squared Euclidean distances to each cluster center.

Publish Cluster Formulas Publishes to the Formula Depot the same scoring code used in the Save Cluster Formula option.

Simulate Clusters Creates a new data table containing simulated cluster observations on the Y variables, using the cluster means and standard deviations.

Remove Removes the clustering report.

Self Organizing Map

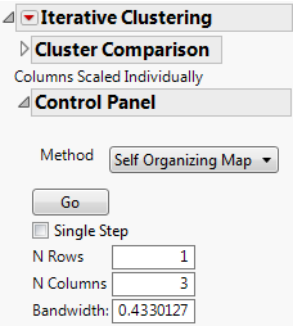
The *Self-Organizing Map* (SOM) technique was developed by Teuvo Kohonen (1989, 1990) and extended by other neural network enthusiasts and statisticians. The original SOM was cast as a learning process, like the original neural net algorithms, but the version implemented here is a variation on *k*-means clustering. In the SOM literature, this variation is called a *batch algorithm* using a *locally weighted linear smoother*.

The goal of a SOM is not only to form clusters in a particular layout on a cluster grid, such that points in clusters that are near each other in the SOM grid are also near each other in multivariate space. In classical *k*-means clustering, the structure of the clusters is arbitrary, but in SOMs the clusters have a grid structure. The grid structure helps interpret the clusters in two dimensions: clusters that are close are more similar than distant clusters. See "Description of SOM Algorithm".

Self Organizing Map Control Panel

Select the Self Organizing Map option from the Method list in the Iterative Clustering Control Panel.

Figure 13.8 Self Organizing Map Control Panel



Some of the options on the panel are described in "Iterative Clustering Control Panel". The remaining options are described below.

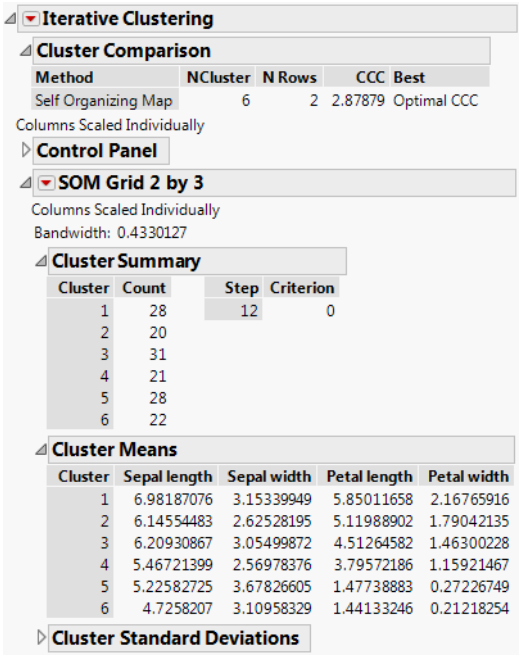
N Rows The number of rows in the cluster grid.

N Columns The number of columns in the cluster grid.

Bandwidth Specifies the effect of neighboring clusters for predicting centroids. A smaller bandwidth results in putting more weight on closer clusters.

Self Organizing Map Report

Figure 13.9 Self Organizing Map Report



Iterative Clustering

Cluster Comparison

Method	NCluster	N Rows	CCC	Best
Self Organizing Map	6	2	2.87879	Optimal CCC

Columns Scaled Individually

Control Panel

SOM Grid 2 by 3

Columns Scaled Individually

Bandwidth: 0.4330127

Cluster Summary

Cluster	Count	Step	Criterion
1	28	12	0
2	20		
3	31		
4	21		
5	28		
6	22		

Cluster Means

Cluster	Sepal length	Sepal width	Petal length	Petal width
1	6.98187076	3.15339949	5.85011658	2.16765916
2	6.14554483	2.62528195	5.11988902	1.79042135
3	6.20930867	3.05499872	4.51264582	1.46300228
4	5.46721399	2.56978376	3.79572186	1.15921467
5	5.22582725	3.67826605	1.47738883	0.27226749
6	4.7258207	3.10958329	1.44133246	0.21218254

Cluster Standard Deviations

The Cluster Comparison Report shows the total number of clusters and also the number of rows requested. The SOM report is named according to the Grid size requested. The Bandwidth is given at the top of the SOM Grid report. The report itself is analogous to the K Means NCluster report. See "K Means Report".

For more information about the red triangle options for Self Organizing Maps, see "K Means Report Options".

Description of SOM Algorithm

This section contains the steps of the SOM implementation in JMP.

- Initial cluster seeds are selected in a way that provides a good coverage of the multidimensional space. JMP uses principal components to determine the two directions that capture the most variation in the data.
- JMP then lays out a grid in this principal component space with its edges 2.5 standard deviations from the middle in each direction. The clusters seeds are determined by translating this grid back into the original space of the variables.

- The cluster assignment proceeds as with *k*-means. Each point is assigned to the cluster closest to it.
- The means are estimated for each cluster as in *k*-means. JMP then uses these means to set up a weighted regression with each variable as the response in the regression, and the SOM grid coordinates as the regressors. The weighting function uses a kernel function that gives large weight to the cluster whose center is being estimated. Smaller weights are given to clusters farther away from the cluster in the SOM grid. The new cluster means are the predicted values from this regression.
- These iterations proceed until the process has converged.

Additional Example of K Means Cluster Platform

Example of a Self-Organizing Map

This example uses the Iris.jmp sample data table, which includes measurements of sepal length, sepal width, petal length, and petal width for three species of irises.

1. Select **Help > Sample Data Library** and open Iris.jmp.
2. Select **Analyze > Clustering > K Means Cluster**.
3. Select Sepal length, Sepal width, Petal length, and Petal width and click **Y, Columns**.
4. Click **OK**.
5. Select **Self Organizing Map** from the Method menu on the Control Panel.
6. Set **N Rows** equal to 1 and **N Columns** equal to 2.
7. Click **Go**.
8. Open the Control Panel Report.
9. Set **N Rows** equal to 1 and **N Columns** equal to 3.
10. Click **Go**.
11. Open the Control Panel Report.
12. Set **N Rows** equal to 2 and **N Columns** equal to 2.
13. Click **Go**.

Figure 13.10 SOM Cluster Comparison

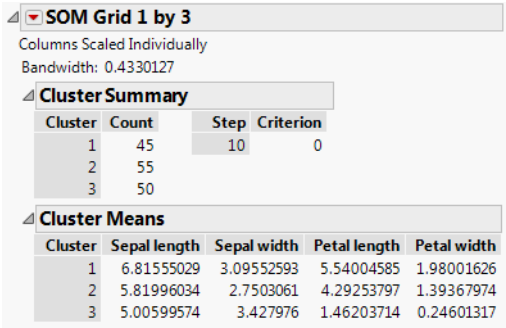
Cluster Comparison

Method	NCluster	N Rows	CCC	Best
Self Organizing Map	2	1	3.35952	
Self Organizing Map	3	1	4.95837	Optimal CCC
Self Organizing Map	4	2	3.64938	

The Cluster Comparison report appears at the top of the report window. The best fit is determined by the highest CCC value. Notice the number of clusters that gives the largest CCC is 3, which is the number of species.

14. Scroll to the SOM Grid 1 by 3 report. We can see the classification was not perfect; each cluster should represent each species, with 50 rows for each.

Figure 13.11 Self-Organizing Map Report for Iris.jmp



SOM Grid 1 by 3

Columns Scaled Individually

Bandwidth: 0.4330127

Cluster Summary

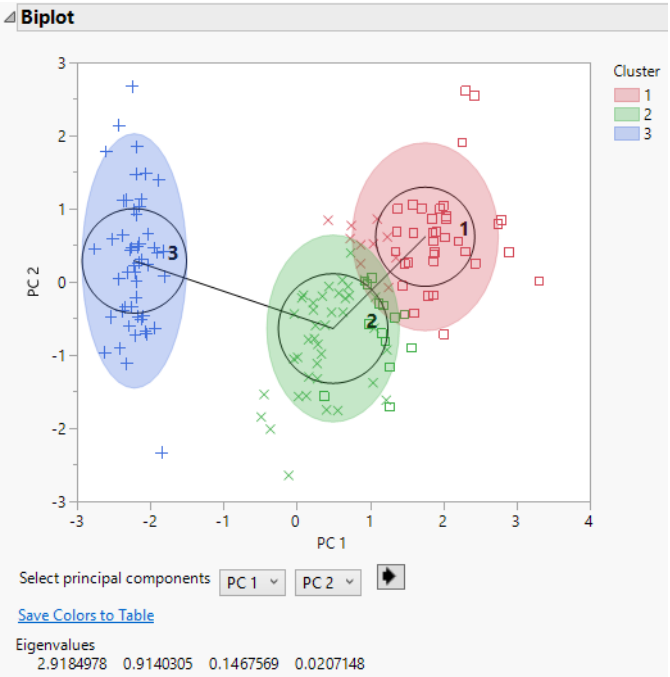
Cluster	Count	Step	Criterion
1	45	10	0
2	55		
3	50		

Cluster Means

Cluster	Sepal length	Sepal width	Petal length	Petal width
1	6.81555029	3.09552593	5.54004585	1.98001626
2	5.81996034	2.7503061	4.29253797	1.39367974
3	5.00599574	3.427976	1.46203714	0.24601317

15. In the data table, select the Species column and select **Rows > Color or Mark by Column**.
16. Select the **Classic** option under Markers.
17. Click **OK**.
18. Click the red triangle menu next to SOM Grid 1 by 3 and select **Biplot**.

Figure 13.12 SOM Biplot



We can see that all rows from Cluster 3 are correctly identified as the setosa species. The other two species, virginica and versicolor, overlap slightly and can be mistaken for each other.

Chapter 14

Normal Mixtures

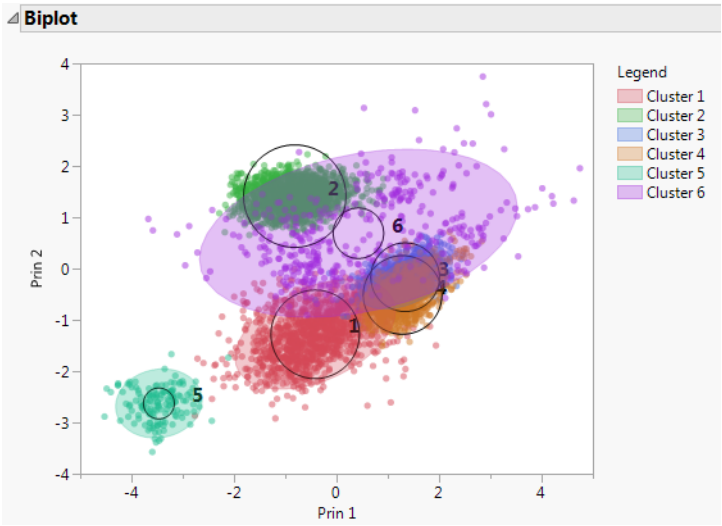
Group Observations Using Probabilities

Use Normal Mixtures for clustering when your data come from overlapping normal distributions. You need to specify the number of clusters in advance.

Normal mixtures is an iterative technique based on the assumption that the joint probability distribution of the observations is approximated using a mixture of multivariate normal distributions. These mixtures represent different clusters. The individual clusters have multivariate normal distributions.

When clusters are well separated, hierarchical and *k*-means clustering work well. But when clusters overlap, normal mixtures provides a better alternative, because it is based on cluster membership probabilities, rather than arbitrary cluster assignments based on borders.

Figure 14.1 Normal Mixtures Biplot



Contents

Overview of the Normal Mixtures Clustering Platform 321
Overview of Platforms for Clustering Observations 321
Example of Normal Mixtures Clustering 323
Launch the Normal Mixtures Clustering Platform 325
Model Based Clustering Report 326
Model Based Clustering Options 326
Model Based Clustering Control Panel 326
Normal Mixtures Report 327
Cluster Comparison Report 328
Normal Mixtures Report 328
Normal Mixtures Report Options 328
Statistical Details for the Normal Mixtures Clustering Platform 330

Overview of the Normal Mixtures Clustering Platform

Normal Mixtures is one of four platforms that JMP provides for clustering observations. For a comparison of all four methods, see "Overview of Platforms for Clustering Observations".

Normal mixtures is an iterative clustering technique for numerical variables. However, it also predicts the proportion of responses expected within each cluster. Normal mixtures assumes that the joint probability distribution of the measurement columns can be approximated using a mixture of multivariate normal distributions, which represent different clusters. Mean vectors and covariance matrices are estimated for each cluster. See McLachlan and Krishnan (1997) and Section 9.6 in Hand et al. (2001).

Note: The Normal Mixtures algorithm involves iterating through random guesses for the cluster centers. Because of this, results from different runs of the analysis might differ slightly.

If you suspect that you have multivariate outliers, you have two options. You can use an outlier cluster or the Explore Outliers Utility. The outlier cluster option assumes a uniform distribution and is less sensitive to outliers than the standard Normal Mixtures method. The Explore Outliers Utility enables you to explore and handle outliers prior to analysis. See "Outlier Cluster" and *Predictive and Specialized Modeling*.

Overview of Platforms for Clustering Observations

Clustering is a multivariate technique that groups together observations that share similar values across a number of variables. Typically, observations are not scattered evenly through *p*-dimensional space, where *p* is the number of variables. Instead, the observations form clumps, or clusters. Identifying these clusters provides you with a deeper understanding of your data.

Note: JMP also provides a platform that enables you to cluster variables. See "Cluster Variables".

JMP provides four platforms that you can use to cluster observations:

- Hierarchical Cluster is useful for smaller tables with up to several tens of thousands of rows and allows character data. Hierarchical clustering combines rows in a hierarchical sequence that is portrayed as a tree. You can choose the number of clusters that is most appropriate for your data after the tree is built.
- K Means Cluster is appropriate for larger tables with up to millions of rows and allows only numerical data. You need to specify the number of clusters, *k*, in advance. The algorithm guesses at cluster seed points. It then conducts an iterative process of alternately assigning points to clusters and recalculating cluster centers.

- Normal Mixtures is appropriate when your data come from a mixture of multivariate normal distributions that might overlap and allows only numerical data. For situations where you have multivariate outliers, you can use an outlier cluster with an assumed uniform distribution.

  You need to specify the number of clusters in advance. Maximum likelihood is used to estimate the mixture proportions and the means, standard deviations, and correlations jointly. Each point is assigned a probability of being in each group. The EM algorithm is used to obtain estimates.
- Latent Class Analysis is appropriate when most of your variables are categorical. You need to specify the number of clusters in advance. The algorithm fits a model that assumes a multinomial mixture distribution. A maximum likelihood estimate of cluster membership is calculated for each observation. An observation is classified into the cluster for which its probability of membership is the largest.

Table 14.1 Summary of Clustering Methods

Method	Data Type or Modeling Type	Data Table Size	Specify Number of Clusters
Hierarchical Cluster	Any	With Fast Ward, up to 200,000 rows With other methods, up to 5,000 rows	No
K Means Cluster	Numeric	Up to millions of rows	Yes
Normal Mixtures	Numeric	Any size	Yes
Latent Class Analysis	Nominal or Ordinal	Any size	Yes

Some of the clustering platforms have options to handle outliers in the data. However, if your data has outliers, it is best to explore them first prior to analyzing. This can be done using the Explore Outliers Utility. For more information, see the Modeling Utilities chapter in *Predictive and Specialized Modeling*.

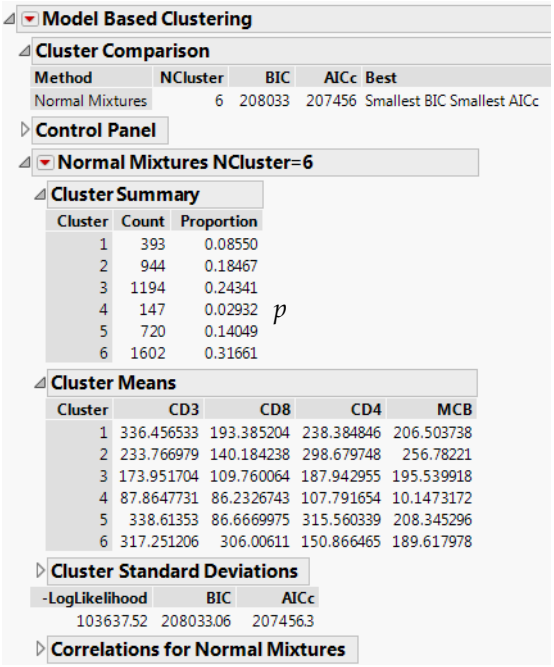
Example of Normal Mixtures Clustering

Cytometry is used to measure various characteristics of cells. Measurements of cell markers help diagnose certain diseases. In this example, you cluster observations based on readings of four markers in a cytometry analysis.

1. Select **Help > Sample Data Library** and open Cytometry.jmp
2. Select **Analyze > Clustering > Normal Mixtures**
3. Select CD3, CD8, CD4, and MCB and click **Y, Columns**.
4. Click **OK**.
5. Enter 6 next to **Number of Clusters**.
6. Click **Go**.

Note: Your results might differ because the algorithm has a random starting value.

Figure 14.2 Normal Mixtures NCluster=6 Report



Model Based Clustering

Cluster Comparison

Method	NCluster	BIC	AICc	Best
Normal Mixtures	6	208033	207456	Smallest BIC Smallest AICc

Control Panel

Normal Mixtures NCluster=6

Cluster Summary

Cluster	Count	Proportion
1	393	0.08550
2	944	0.18467
3	1194	0.24341
4	147	0.02932
5	720	0.14049
6	1602	0.31661

p

Cluster Means

Cluster	CD3	CD8	CD4	MCB
1	336.456533	193.385204	238.384846	206.503738
2	233.766979	140.184238	298.679748	256.78221
3	173.951704	109.760064	187.942955	195.539918
4	87.8647731	86.2326743	107.791654	10.1473172
5	338.61353	86.6669975	315.560339	208.345296
6	317.251206	306.00611	150.866465	189.617978

Cluster Standard Deviations

-LogLikelihood	BIC	AICc
103637.52	208033.06	207456.3

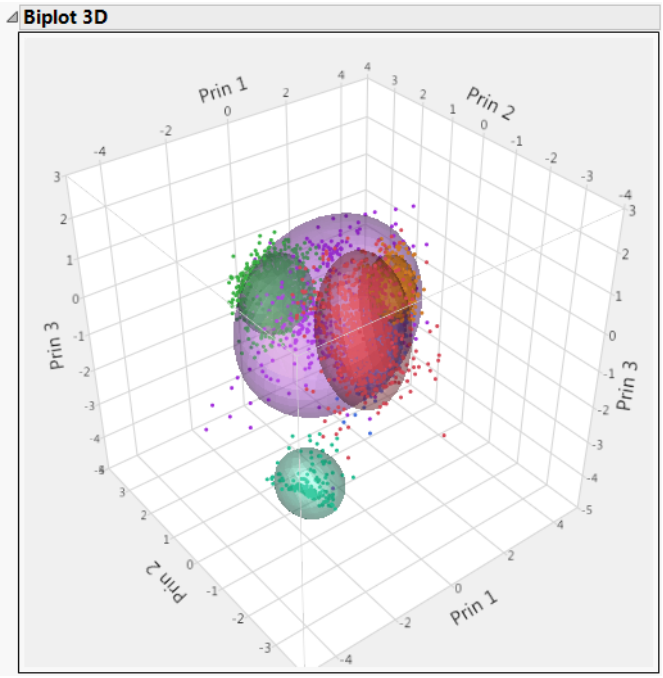
Correlations for Normal Mixtures

The Cluster Summary report shows the number of observations in each of the six clusters. The Cluster Means report shows the means of the four marker readings for each cluster.

7. Click the red triangle next to Normal Mixtures NCluster=6 and select **Biplot 3D**.

Note: Your biplot 3D might appear differently because the algorithm has a random starting value.

Figure 14.3 3D Biplot of Cytometry Data

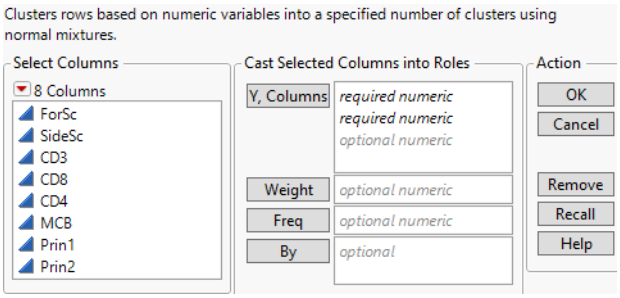


The plot shows contours for the normal densities that are fit to the clusters. Note that one cluster appears to be distinctly separated from the other clusters based on the first three principal components.

Launch the Normal Mixtures Clustering Platform

Launch the Normal Mixtures Clustering platform by selecting Analyze > Clustering > Normal Mixtures.

Figure 14.4 Normal Mixtures Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns The variables used for clustering observations.

> **Note:** Normal Mixtures clustering supports only numeric columns.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

By A column whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed. The results are presented in separate reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

When you click OK, a Control Panel appears. See "Model Based Clustering Control Panel".

Model Based Clustering Report

When you click OK in the launch window, the Model Based Clustering report window opens, showing a Control Panel for fitting models. See "Model Based Clustering Control Panel". As you fit models, additional reports are added to the window. See "Normal Mixtures Report".

Model Based Clustering Options

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

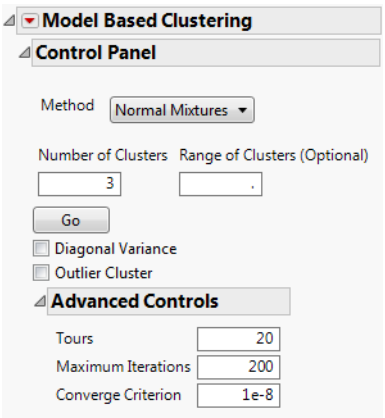
Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Model Based Clustering Control Panel

The Control Panel for the Cytometry.jmp data table, with the variables CD3 through MCB as **Y, Columns**, is shown in Figure 14.5. You can fit various numbers of clusters using the Control Panel iteratively or you can specify a range using the Range of Clusters option.

Figure 14.5 Control Panel for Normal Mixtures Method



The Model Based Clustering Control Panel has these options:

Number of Clusters Designates the number of clusters to form.

Range of Clusters (Optional) Provides an upper bound for the number of clusters to form. If a number is entered here, the platform creates separate analyses for every integer between **Number of clusters** and the value entered as **Range of Clusters (Optional)**.

Go Fits the clusters.

Diagonal Variance Constrains the off-diagonal elements of the covariance matrix to zero. The platform fits multivariate normal distributions that have no correlations between the variables.

Note: The Diagonal Variance option is sometimes necessary to avoid obtaining a singular covariance matrix when there are fewer observations than variables. It can also be used to avoid estimating very large covariance matrices for large numbers of variables.

Outlier Cluster Fits a cluster to catch outliers that do not fall into any of the normal clusters. If this cluster is created, it is designated Cluster 0, and the count of observations appears in the Cluster Summary report. The distribution of observations that fall in the outlier cluster is assumed to be uniform over the hypercube that encompasses the observations.

Advanced Controls The following advanced controls are available:

Tours The number of independent restarts of the estimation process. Each restart has a different starting value. Independent starts help guard against finding local solutions.

Maximum Iterations The maximum number of iterations of the convergence stage of the EM algorithm.

Converge Criterion The difference in the likelihood at which the EM iterations terminate.

Normal Mixtures Report

When you click Go in the Control Panel, the following reports appear:

- A Cluster Comparison report. See "Cluster Comparison Report"
- One or more Normal Mixtures reports. The reports are dynamically named Normal Mixtures NCluster=<k>, depending on *k,* the number of clusters fit. A Normal Mixtures NCluster=<k> report appears for each fit that you conduct.

The Cluster Comparison report and the Normal Mixtures NCluster=6 report for the Cytometry.jmp data table, with the variables CD3 through MCB as **Y, Columns**, are shown in Figure 14.2 and "Normal Mixtures NCluster=6 Report".

Cluster Comparison Report

The Cluster Comparison report gives fit statistics to compare the various models. The fit statistics are BIC and AICc. Smaller values of each indicate better fit. The best fit is indicated in a column called Best.

Normal Mixtures Report

The Normal Mixtures report gives summary statistics for each cluster:

- The Cluster Summary report gives the number of observations and proportion for each cluster.
- The Cluster Means report gives means for the observations in each cluster for each variable.
- The Cluster Standard Deviations report gives standard deviations for the observations in each cluster for each variable.
- The -LogLikelihood table gives the negative log-likelihood, BIC, and AICc. See *Fitting Linear Models*.
- The Correlations for Normal Mixtures report gives the estimated correlation matrix for each cluster

Normal Mixtures Report Options

Biplot Shows a plot of the points and clusters in the first two principal components of the data, along with a legend identifying the cluster colors. Circles are drawn around the cluster centers and the size of the circles is proportional to the count inside the cluster. The shaded area is the density contour around the mean. By default, this area indicates where 90% of the observations in that cluster would fall (Mardia et al. 1980). Use the list below the plot to change the plot axes to other principal components. Alternatively, use the arrow button to cycle through all possible axes combinations. An option to save the cluster colors to the data table is also located below the plot. See "Save Colors to Table". The eigenvalues are shown in decreasing order.

Note: The biplot always uses the correlation matrix to calculate the principal components.

Biplot Options Contains options for controlling the appearance of the Biplot.

Show Biplot Rays Shows the biplot rays. The labeled rays show the directions of the covariates in the subspace defined by the principal components. They represent the degree of association of each variable with each principal component.

Biplot Ray Position Enables you to specify the position and radius scaling of the biplot rays. By default, the rays emanate from the point (0,0). In the plot, you can drag the rays or use this option to specify coordinates. You can also adjust the scaling of the rays to make them more visible with the radius scaling option.

Biplot Contour Density Enables you to specify the confidence level for the density contours. The default confidence level is 90%.

Mark Clusters Assigns markers that identify the clusters to the rows of the data table.

Biplot 3D Shows a three-dimensional biplot of the data. Available only when there are three or more variables.

Parallel Coord Plots Creates a parallel coordinate plot for each cluster. The plot report has options for showing and hiding the data and means. See *Essential Graphing*.

Scatterplot Matrix Creates a scatterplot matrix using all of the Y variables.

Save Colors to Table Assigns colors that identify the clusters to the rows of the data table. If there is a Biplot in the report window, the colors saved to the data table match the colors of the clusters in the Biplot. If the colors are changed in the Biplot and the Save Colors To Table option is selected again, the colors in the table update to match those in the Biplot.

Note: When any of the Save options are selected, each saved column contains a Notes column property that specifies the number of clusters for that particular column's data. This enables you to save columns from more than one cluster fit and use the column property to identify which clustering fit the saved column is from.

Save Clusters Adds a column called Cluster that contains the number of the cluster to which the given row is assigned to the data table. For normal mixtures, this is the cluster that is most likely.

Save Cluster Formula Adds a formula column called Cluster Formula to the data table. This formula identifies which cluster the row belongs to.

Publish Cluster Formulas Publishes to the Formula Depot the same scoring code used in the Save Cluster Formula option. If Publish Cluster Formulas is selected and Run Script is chosen from the model within the Formula Depot, the columns saved to the data table should match those that are saved when Save Cluster Formula is selected.

Save Mixture Probabilities Adds a column called Prob Cluster <k> for each cluster that contains the probability an observation belongs to that cluster.

Save Mixture Formulas Adds columns to the data table that contain the formulas used to calculate the mixture probabilities. Use these formula columns to score probabilities for excluded data, or data that you add to the table.

Dist Formula <k> The estimated multivariate normal density function for Cluster <k> evaluated at the observation.

Dist Total The sum of the distance formula columns. The formula in this column is equivalent to the formula in the Mixture Density column created by the Save Density Formula option.

Prob Formula <k> The probability that the observation belongs to Cluster <k>. These columns contain the formulas that give the values in the Prob Cluster <k> columns created by the Save Mixture Probabilities option. The column formula for calculating the mixture probabilities is:

$$\text{Prob Formula <k>} = \frac{\text{Dist Formula <k>}}{\text{Dist Total}}$$

Save Density Formula Adds a column called Mixture Density that contains the estimated density function for the normal mixture to the data table.

Simulate Clusters Uses the mixture density to simulate predictor values. Saves these and the clusters into which they are classified in a new data table.

Remove Removes the clustering report.

Statistical Details for the Normal Mixtures Clustering Platform

Normal Mixtures uses the EM algorithm to do fitting because it is more stable than the Newton-Raphson algorithm. In addition, JMP uses a Bayesian regularized version of the EM algorithm, which allows smooth handling of cases where the covariance matrix is singular. Since the estimates are heavily dependent on initial guesses, the platform iterates through a number of tours. Each tour has randomly selected points for the initial center.

Doing multiple tours makes the estimation process somewhat expensive, so considerable patience is required for large problems. Controls enable you to specify the tour and iteration limits.

Chapter 15

Latent Class Analysis

Group Observations of Categorical Variables

Latent class analysis enables you to find clusters of observations for categorical response variables. A latent variable is an unobservable grouping variable. Each level of the latent variable is called a latent class. The Latent Class Analysis platform fits a latent class model and determines the most likely cluster or latent class for each observation. In most situations, a subject matter expert uses the results of a latent class analysis to create definitions for each latent class based on the characteristics of the class.

Figure 15.1 Example of Latent Class Analysis

Parameter Estimates

		sex		marital status		country			size			type		
Cluster	Overall	Female	Male	Married	Single	American	European	Japanese	Large	Medium	Small	Family	Sporty	Work
Cluster 1	0.28553	0.3763	0.6237	0.4160	0.5840	0.1555	0.2306	0.6139	0.0009	0.2839	0.7152	0.0190	0.9635	0.0175
Cluster 2	0.25943	0.4066	0.5934	0.6740	0.3260	0.0271	0.0562	0.9168	0.0029	0.2942	0.7029	0.6419	0.0368	0.3213
Cluster 3	0.20712	0.6793	0.3207	0.7526	0.2474	0.5618	0.1376	0.3005	0.0025	0.9914	0.0062	0.8013	0.1950	0.0036
Cluster 4	0.19697	0.4707	0.5293	0.9922	0.0078	0.8532	0.0826	0.0642	0.4965	0.2047	0.2989	0.7651	0.0020	0.2329
Cluster 5	0.05095	0.1837	0.8163	0.0324	0.9676	0.8477	0.1468	0.0054	0.7766	0.1180	0.1054	0.4366	0.0957	0.4677

Cluster	Overall	sex	marital status	country	size	type
Cluster 1	0.28553					
Cluster 2	0.25943					
Cluster 3	0.20712					
Cluster 4	0.19697					
Cluster 5	0.05095					

Contents

Overview of the Latent Class Analysis Platform 333
Example of Latent Class Analysis 333
Launch the Latent Class Analysis Platform 337
The Latent Class Analysis Report 338
Cluster Comparison Report 338
Latent Class Model Report 338
Latent Class Analysis Platform Options 341
Latent Class Analysis Options 341
Latent Class Model Options 341
Additional Example of the Latent Class Analysis Platform 342
Plot Probabilities of Cluster Membership 342
Statistical Details for the Latent Class Analysis Platform 344
Latent Class Model Fit 344
Maximum Number of Clusters 345

Overview of the Latent Class Analysis Platform

The Latent Class Analysis platform fits a latent class model to categorical response variables and determines the most likely cluster or latent class for each observation. A *latent variable* is an unobservable grouping variable. Each level of the latent variable is called a *latent class*. For example, latent classes could be clusters of survey respondents that are grouped by their preference for risk.

The model takes the form of a multinomial mixture model. There are two sets of parameters in the model: the γ parameters and the ρ parameters. The γ parameters represent the overall probabilities of cluster membership. The ρ parameters represent the probabilities of observing a given response conditional on cluster membership. A latent class is characterized by a pattern of these conditional probabilities.

In order for the analysis results to be meaningful, a subject matter expert must interpret the clusters that the platform generates. This subject matter expert examines characteristics of the latent classes and constructs a definition for each class based on those characteristics.

Note: Rows with missing values in any of the response columns are excluded from the analysis.

For more information about latent class models, see Collins and Lanza (2010) and Goodman (1974).

Example of Latent Class Analysis

This example uses the Latent Class Analysis platform to analyze responses to a 2005 survey of US high school students. The survey asked students a variety of multiple choice questions regarding health-risk behaviors.

In this example, you fit a latent class model to identify clusters of students based on their responses to 12 questions. The columns that you analyze were obtained from multiple choice survey questions by binning the responses into two classes (Yes/No).

1. Select **Help > Sample Data Library** and open Health Risk Survey.jmp.
2. In the Health Risk Survey data table, click the green triangle next to the Launch LCA Platform script.

   The script selects the 12 columns of interest, opens the Latent Class Analysis launch window, and enters the 12 columns of interest as Y.

Note: To launch the LCA Platform on your own, select Analysis > Clustering > Latent Class Analysis.

3. Type 5 in the box next to **Up to**.

   This option fits latent class models for 3 and up to 5 clusters.

4. Click **OK**.

Figure 15.2 Cluster Comparison Report

Cluster Comparison

NCluster	-LogLikelihood	BIC	AIC	Best
3	38713	77776.2	77501.9	
4	38207.1	76884.4	76516.3	
5	37964.8	76519.6	76057.6	Smallest BIC Smallest AIC

The Latent Class Analysis outline contains a Cluster Comparison report and three separate Latent Class Model reports. The Latent Class Model reports show the models for three, four, and five clusters. In the Cluster Comparison report, the model with five clusters has the smallest BIC and AIC, which indicates that it is the best fitting model out of the three. This is the model that you analyze.

5. In the Latent Class Model for 5 Clusters report, examine the bar charts under Parameter Estimates. Note the following:
   - Cluster 1 has mostly No answers to all of the risk behaviors.
   - Cluster 2 has high numbers of Yes answers for the four risk behaviors before the age of 13.
   - Cluster 3 has high numbers of Yes answers for driving when drinking and five or more drinks in the past 30 days.
   - Cluster 4 has high numbers of Yes answers for most of the risk behaviors except for the ones before the age of 13.
   - Cluster 5 has the highest number of Yes answers for most of the risk behaviors.

   Use this information to give the clusters meaningful names.

6. Click the red triangle next to Latent Class Model for 5 Clusters and select **Rename Clusters**:
   - Enter Low Risk for Cluster 1.
   - Enter Early Risk-Takers for Cluster 2.
   - Enter Drinkers for Cluster 3.
   - Enter Late High Risk for Cluster 4.
   - Enter High Risk for Cluster 5.

7. Click **OK**.

8. Click **OK** in the JMP Alert that appears.

Note: The new cluster names are not saved to scripts.

Figure 15.3 Partial Parameter Estimates Report

Parameter Estimates

Cluster	Overall	Drove 1+ times when drinking		Smoked cigarette before 13		Smoked daily for 30 days		Had first drink before 13		Five+ drinks 1+ past 30 days		Tried marijuana before 13		Used cocaine 1+ times in life		Sniffed glu 1+ times in
		No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No
Low Risk	0.67580	0.9872	0.0128	0.9623	0.0377	0.9740	0.0260	0.8647	0.1353	0.8973	0.1027	0.9943	0.0057	0.9930	0.0070	0.9439
Early Risk-Takers	0.13847	0.8958	0.1042	0.4456	0.5544	0.7406	0.2594	0.3413	0.6587	0.5958	0.4042	0.6535	0.3465	0.9516	0.0484	0.8207
Drinkers	0.09585	0.4297	0.5703	0.8666	0.1334	0.7618	0.2382	0.7613	0.2387	0.0358	0.9642	0.9666	0.0334	0.9093	0.0907	0.8689
Late High Risk	0.05158	0.6850	0.3150	0.8220	0.1780	0.4924	0.5076	0.7696	0.2304	0.3491	0.6509	0.9108	0.0892	0.2691	0.7309	0.5434
High Risk	0.03830	0.4939	0.5061	0.1329	0.8671	0.2874	0.7126	0.1187	0.8813	0.1266	0.8734	0.2134	0.7866	0.1640	0.8360	0.4705

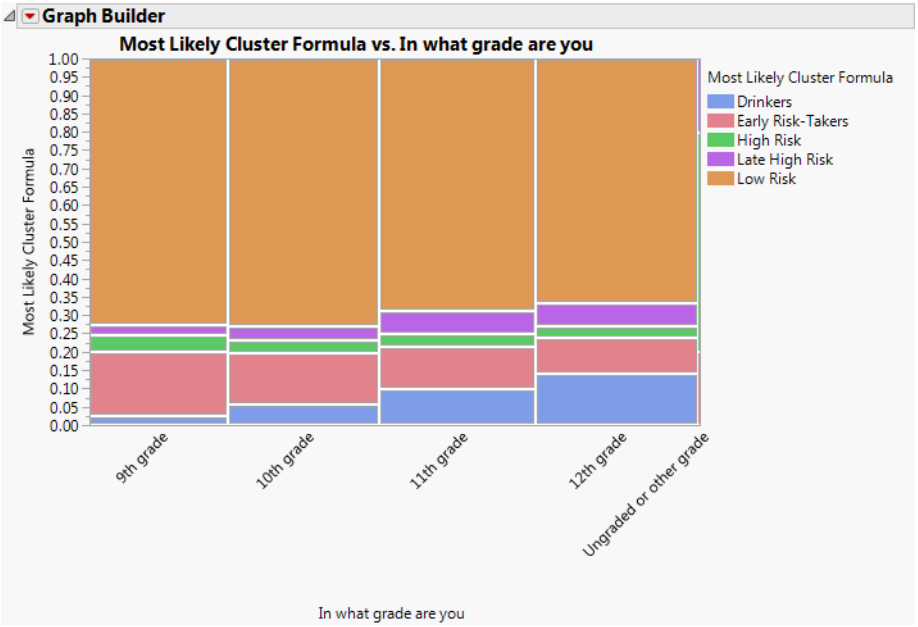
Cluster	Overall	Drove 1+ times when drinking	Smoked cigarette before 13	Smoked daily for 30 days	Had first drink before 13	Five+ drinks 1+ past 30 days	Tried marijuana before 13	Used cocaine 1+ times in life	Sniffed glue 1+ times in life
Low Risk	0.67580								
Early Risk-Takers	0.13847								
Drinkers	0.09585								
Late High Risk	0.05158								
High Risk	0.03830								

Figure 15.3 shows parameter estimates for the first eight variables in the analysis. The new cluster names appear in the report window.

Next, compare cluster membership to the demographic question “In what grade are you”.

9. Click the red triangle next to Latent Class Model for 5 Clusters and select **Save Mixture and Cluster Formulas**.
10. Select **Graph > Graph Builder**.
11. Enter In what grade are you as **X**.
12. Enter Most Likely Cluster Formula as **Y**.
13. Select the Mosaic element.
14. Click **Done**.

Figure 15.4 Mosaic Plot of Cluster Membership versus Grade Level

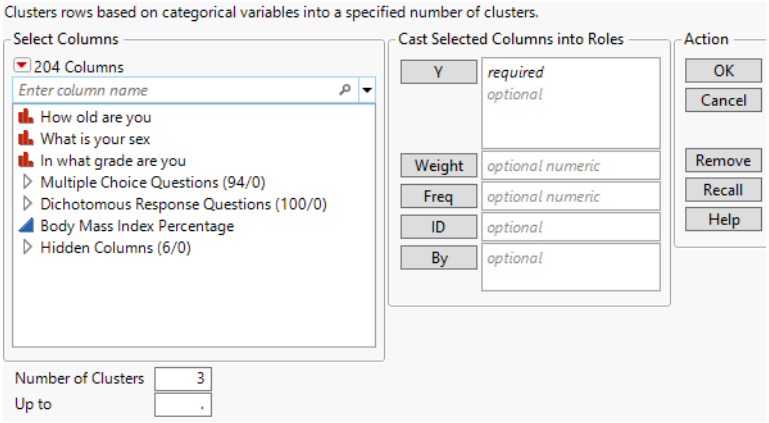


Observe that most of the respondents fall into the Low Risk cluster. The class labeled Drinkers includes more respondents as the grade level increases.

Launch the Latent Class Analysis Platform

Launch the Latent Class Analysis platform by selecting **Analyze > Clustering > Latent Class Analysis**.

Figure 15.5 Latent Class Analysis Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Latent Class Analysis platform launch window contains the following options:

Y The column or columns that you want to analyze. You can analyze columns with nominal, ordinal, or multiple response modeling types. To analyze nominal or ordinal responses, two or more columns are required. Only one column is required if it contains multiple responses and has a multiple response modeling type.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

ID A column used to identify separate respondents. This identification is used in some output tables.

By A column that creates a report consisting of separate analyses for each level of the variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

Number of Clusters The number of clusters to be computed in the analysis.

Up to Specifies a maximum number of clusters. If this number exceeds the value specified for Number of Clusters, a model report is produced with a number of clusters equal to

each integer value in the range between Number of Clusters and Up to. These reports appear as part of the Latent Class Analysis report outline.

Caution: There is a maximum number of clusters that an LCA model can adequately fit. If you request more clusters than the maximum, a warning message appears in the report window. The LCA Platform fits up to the maximum number of clusters supported by the columns. For more information about determining the maximum number of clusters, see "Maximum Number of Clusters".

The Latent Class Analysis Report

The initial Latent Class Analysis report contains a Cluster Comparison and Latent Class Model reports for each specified number of clusters.

Cluster Comparison Report

The Cluster Comparison report gives fit statistics to compare the various models. The fit statistics are the negative log-likelihood (-LogLikelihood), BIC, and AIC. Smaller values of each indicate better fit. The best fit is indicated in a column called Best. See *Fitting Linear Models*.

Latent Class Model Report

Each Latent Class Model Report is dynamically named Latent Class Model for <k> Clusters, depending on *k*, the number of clusters fit. The reports contains the following results and outlines:

- "Model Summary"
- "Parameter Estimates"
- "Transposed Parameter Estimates"
- "Effect Sizes"
- "MDS Plot"
- "Mixture Probabilities"

Model Summary

By default, a summary of the model for the specified number of clusters appears at the top of each Latent Class Model report. The model summary contains the -LogLikelihood, Number of Parameters, BIC, and AIC. These summary values can be used to determine how well the model fits the data. Lower values of -LogLikelihood, BIC, and AIC indicate better fits. See *Fitting Linear Models*. The Number of Parameters value gives the number of unique parameters in the latent class model. See "Statistical Details for the Latent Class Analysis Platform".

Parameter Estimates

The Parameter Estimates report contains tabular and graphical summaries of the parameter estimates and is displayed by default. Each summary contains rows corresponding to the model clusters.

The Overall column shows the probability of an observation belonging to each cluster. (These are the γ parameters. See "Statistical Details for the Latent Class Analysis Platform".)

The remaining columns in the displays are grouped with vertical dividers according to the Y columns specified in the Latent Class Analysis launch window:

- Each group of categorical response columns has a column for each level within the respective response. In each group, the value in a given row and column is the conditional probability of the response indicated by the column, given that the observation belongs to the cluster identified by the row. (These are the ρ parameters.)
- Each group of multiple response columns has a column for each category within the multiple response. In each group, the value in a given row and column is the conditional probability of a response at the lower level of the indicated category, given that the observation belongs to the cluster identified by the row. (These are the ρ parameters.)

The graphical display shows the conditional probability values as *share charts*. For each cluster and each Y, the conditional probabilities given cluster membership are plotted as a horizontal stacked bar chart. For a binary or nominal response column, the percentages in these charts sum to one for each response. For a multiple response column, the percentages are of the lower level of each of the categories and do not sum to one. The stacking of bars follows the order of appearance of the variables in the table of values. You can also hover over the bars to view the levels or categories of the variable.

Tip: You can select one or more rows in either table in the Parameter Estimates report to select the observations assigned to the corresponding clusters.

Transposed Parameter Estimates

The Transposed Parameter Estimates report contains a table that is the transpose of the Parameter Estimates report table. Here the clusters are shown as columns. The conditional probabilities for each cluster are shown for each response category of each Y column in the analysis.

Note: The estimates from the Overall column are not included in the transposed table.

Effect Sizes

The Effect Sizes table compares the Y columns across clusters and is displayed by default. The statistics in each row of this table are obtained from a contingency table analysis of expected counts for cluster membership by levels or categories of a Y column. The expected counts are obtained by multiplying the number of observations in each cluster by the conditional probabilities for each level or category of the Y column.

For each response, the Pearson chi-square statistic, χ^2, is calculated for the contingency table of expected counts for levels by clusters. Let n represent the number of observations. The value in the Effect Size column is defined as follows:

$$\text{Effect Size} = \sqrt{\frac{\chi^2}{n}}$$

Each value in the LR Logworth column shows -log10(p_{LR}) where p_{LR} is the likelihood ratio test p-value for the contingency table of expected counts. A Logworth value above 2 corresponds to significance at the 0.01 significance level.

Tip: You can select one or more rows in the Effect Sizes table to select the corresponding columns in the data table.

MDS Plot

The MDS Plot contains one point for each cluster and is displayed by default. It is a two-dimensional representation of cluster proximity. Clusters that are closer together are more similar. The plot is created from a dissimilarity matrix of the ρ parameters. For more information about MDS plots, see "Multidimensional Scaling".

Mixture Probabilities

The Mixture Probabilities table displays probabilities of cluster membership for each row. The Most Likely Cluster column indicates the cluster with the highest probability of membership for each row.

Note: Rows that contain a missing value for one or more of the Y columns are excluded from the analysis and do not appear in the Mixture Probabilities table.

Latent Class Analysis Platform Options

- "Latent Class Analysis Options"
- "Latent Class Model Options"

Latent Class Analysis Options

The Latent Class Analysis red triangle menu contains the following options:

New Number of Clusters Enables you to run another analysis using a different number of clusters. The new analysis report is appended to the current report.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Latent Class Model Options

The Latent Class Model for <k> Clusters red triangle menu contains the following options:

Model Reports Enables you to show or hide the available model reports. For more information about the model reports, see "The Latent Class Analysis Report".

Color by Cluster Colors each row in the data table according to its most likely cluster. For an example, see "Additional Example of the Latent Class Analysis Platform".

Save Mixture and Cluster Formulas Saves a formula column to the data table for each cluster as well as a formula column for the most likely cluster.

Save Cluster Formula Only Saves a column to the data table with a formula that determines the most likely cluster.

JMP PRO **Publish Probability Formulas** Creates probability formulas and saves them as formula column scripts in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Save Mixture Probabilities Saves the values in the Mixture Probabilities table to the corresponding rows in the data table.

Save Cluster Only Saves a new column to the data table that contains the most likely cluster for each row. This column does not contain a formula.

Rename Clusters Enables you to give meaningful names to the clusters in the report.

Note: The new cluster names are not saved to a script unless you have specified a random seed for the report. Setting a random seed is available only when you launch the report via a script.

Remove Fit Removes the specified clustering report from the report window.

Additional Example of the Latent Class Analysis Platform

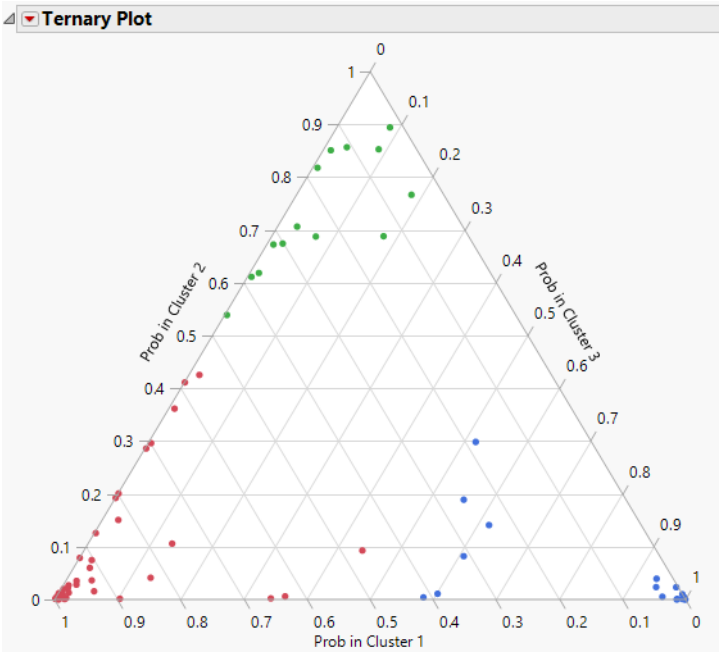
Plot Probabilities of Cluster Membership

This example uses the Car Poll.jmp sample data table, which contains survey data for car owners and car makes. You are interested in classifying the car owners into three clusters and producing a plot to visualize the probabilities of cluster membership. A ternary plot provides a good visualization when you have three clusters.

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Clustering > Latent Class Analysis**.
3. Select all of the columns except age and click **Y**.
4. Click **OK**.
5. Click the red triangle next to Latent Class Model for 3 Clusters and select **Color by Cluster**.
6. Click the red triangle next to Latent Class Model for 3 Clusters and select **Save Mixture Probabilities**.

7. In the Car Poll data table window, select the LCA Cluster Probabilities column group from the column list.
8. Select **Graph > Ternary Plot**.
9. Click **X, Plotting**.
10. Click **OK**.

Figure 15.6 Ternary Plot of Cluster Membership Probabilities



In the ternary plot of cluster probabilities for each observation, most of the cluster membership probabilities fall near the vertices. This indicates that they have high values for one cluster and lower values for the other two. However, there are some points in the middle of the plot, indicating that these observations do not have high probabilities of cluster membership for any of the clusters. These observations might warrant closer inspection or they might indicate that more clusters are needed to better represent the data.

Note: Your results might be different because a random seed was not specified.

Statistical Details for the Latent Class Analysis Platform

- "Latent Class Model Fit"
- "Maximum Number of Clusters"

Latent Class Model Fit

This section describes the latent class model that is fit in the Latent Class Analysis platform. For more information about latent class models, see Collins and Lanza (2010) and Agresti (2013).

Note: The LCA algorithm that is used in the Text Explorer platform takes advantage of the sparsity of the document term matrix. For this reason, the LCA results in the Text Explorer platform do not exactly match the results in the Latent Class Analysis platform.

Let $j = 1, \ldots, J$ represent the observed columns of responses. These are the Y columns in the Latent Class Analysis platform launch window. Denote the number of levels for column j by R_j.

A multidimensional contingency table of the J variables contains $W = R_1 * \ldots * R_J$ cells. Each of these cells is defined by its response pattern for the J variables. Therefore, each response pattern is a J-length vector of the form $\mathbf{y} = y_1, \ldots, y_j$. Define $\mathbf{Y}$ to be the W by J array of all the response patterns considered as row vectors. Each element, $\mathbf{y}_w$, in $\mathbf{Y}$ has a probability $\Pr(\mathbf{y}_w)$. These probabilities sum to 1:

$$\sum_{w=1}^{W} Pr(y_w) = 1$$

Consider the following notation:

- C is the number of clusters in the latent class model.
- γ_c is the probability of membership in cluster c. (The γ_c are the latent class prevalences.) These parameters sum to 1.
- $r_{j,k}$ is the k^{th} level of the j^{th} response.
- $\rho_{j,k|c}$ is the probability of observing response $r_{j,k}$ in column j conditional on membership in class c. (The $\rho_{j,k|c}$ are the item-response probabilities.) For a given cluster and response variable j, the sum of the $\rho_{j,k|c}$ is 1.

- $I(y_j = r_{j,k})$ is an indicator function that equals 1 when the y_j response is the k^{th} level of the j^{th} response, and 0 otherwise.

The probability of observing a specific vector of responses $\mathbf{y}_w = y_1, \ldots, y_j$ is the sum of the conditional probabilities of observing that vector of responses for each of the C latent classes:

$$\Pr(y) = \sum_{c=1}^{C} \gamma_c \prod_{j=1}^{J} \prod_{k=1}^{R_j} \rho_{j,k|c}^{I(y_j = r_{j,k})}$$

This equation is the denominator in the Prob Formula Cluster formulas that you can save to the data table by selecting the Save Mixture and Cluster Formulas option from the Latent Class Analysis red triangle menu. The formula in the Prob Formula Cluster column gives $\Pr(Cluster = c \mid \mathbf{y}_w)$, which equals $\Pr(\mathbf{y}_w, Cluster = c) / \Pr(\mathbf{y}_w)$.

The γ and ρ parameters for latent class models are estimated using the iterative Expectation-Maximization (EM) algorithm. The number of unique parameters in a latent class model is defined as follows:

$$(C-1) + C \sum_{j=1}^{J} (R_j - 1)$$

Maximum Number of Clusters

The maximum number of clusters that can be fit in an LCA model depends on the model degrees of freedom. The degrees of freedom in an LCA model are based on the size of the contingency table created by the columns. The size of the contingency table is the number of cells in the table that contain at least one observation and is denoted as K. If all cells contain at least one observation, K is the product of the number of levels of the response columns. The formula for degrees of freedom is defined as follows:

DF = K - {nCluster - 1 + nCluster(nTotalLevels - nCols)} - 1

where

nCluster = the number of clusters

nTotalLevels = the sum of the levels of the response columns

nCols = the number of response columns

In order for the LCA model to be adequately fit, the degrees of freedom must be positive. Therefore, to ensure DF > 0, the maximum number of clusters is defined as follows:

max(nCluster) < floor[K/(1 + nTotal Levels – nCols)]

Chapter 16

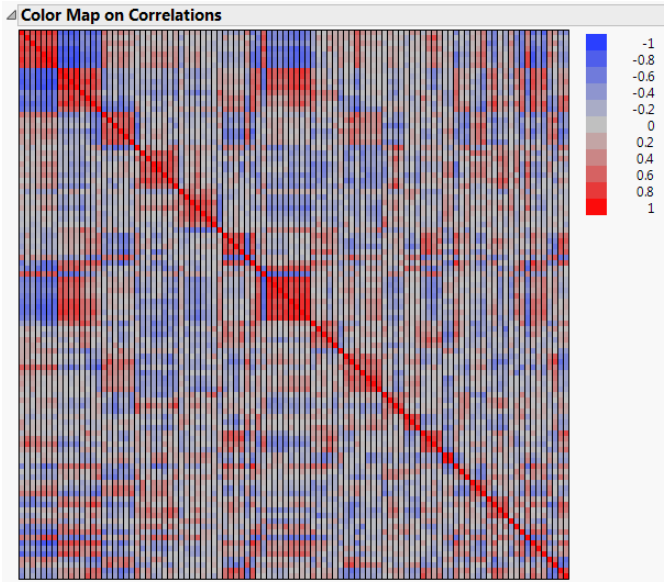
Cluster Variables

Group Similar Variables into Representative Groups

Variable clustering provides a method for grouping similar variables into representative groups. Each cluster can be represented by a single component or variable. The component is a linear combination of all variables in the cluster. Alternatively, the cluster can be represented by the variable identified to be the most representative member in the cluster.

You can use Cluster Variables as a dimension-reduction method. Instead of using a large set of variables in modeling, either the cluster components or the most representative variable in the cluster can be used to explain most of the variation in the data. In addition, dimension reduction using Cluster Variables is often more interpretable than dimension reduction using principal components.

Figure 16.1 Example of Correlation Map for Variables



Contents

Overview of the Cluster Variables Platform 349
Example of the Cluster Variables Platform 349
Launch the Cluster Variables Platform 351
The Cluster Variables Report 351
Color Map on Correlations 352
Cluster Summary 352
Cluster Members 353
Standardized Components 353
Cluster Variables Platform Options 353
Additional Examples of the Cluster Variables Platform 354
Example of Color Map on Correlations 354
Example of Cluster Variables Platform for Dimension Reduction 356
Statistical Details for the Cluster Variables Platform 359
Variable Clustering Algorithm 359

Overview of the Cluster Variables Platform

Principal components analysis constructs components that are linear combinations of all the variables in the analysis. In contrast, the Cluster Variables option constructs components that are linear combinations of variables in a cluster of similar variables. The entire set of variables is partitioned into clusters. For each cluster, a *cluster component* is constructed using the first principal component of the variables in that cluster. This cluster component is the linear combination that explains as much of the variation as possible among the variables in that cluster.

You can use the Cluster Variables option as a dimension-reduction method. A substantial part of the variation in a large set of variables can often be represented by cluster components or by the most representative variable in the cluster. These new variables can then be used in predictive or other modeling techniques. The new cluster-based variables are usually more interpretable than principal components based on all the variables.

Principal components constructed from a common set of variables are orthogonal. However, cluster components are not orthogonal because they are constructed from distinct sets of variables.

When you have a large set of variables, the Cluster Variables platform uses an algorithm based on the singular value decomposition to shorten computation time. For additional background, see "Wide Linear Methods and the Singular Value Decomposition".

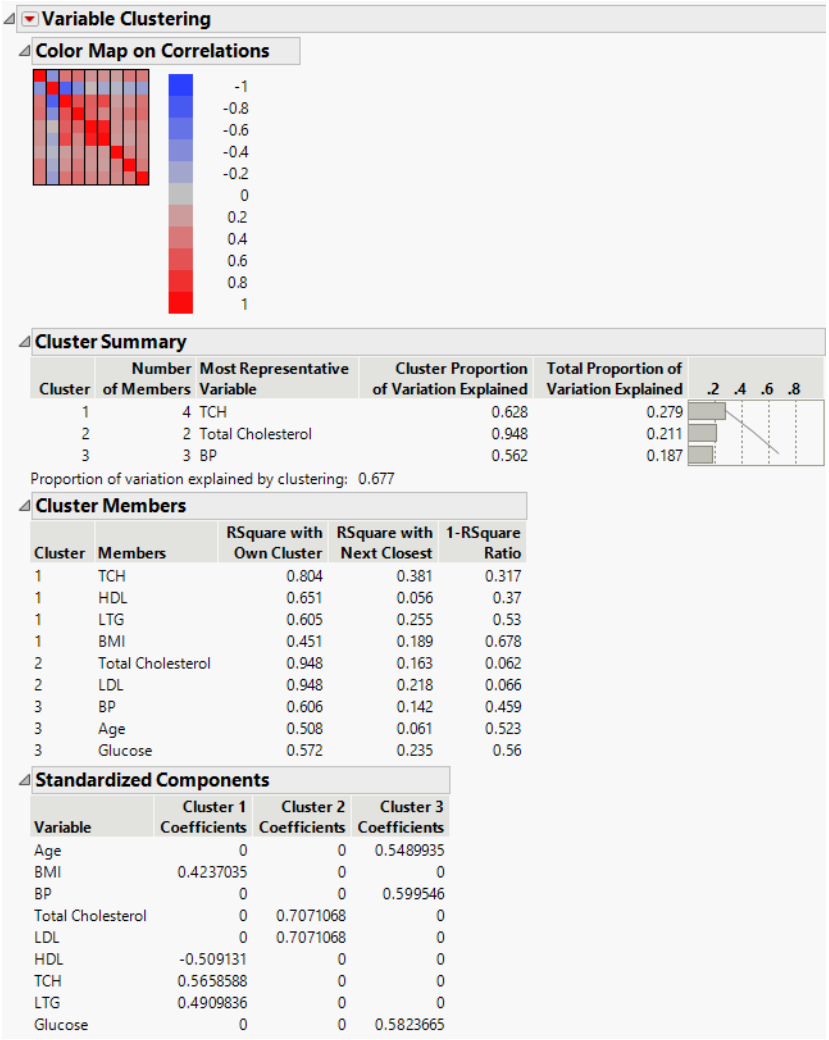
Example of the Cluster Variables Platform

The Diabetes.jmp sample data table contains ten baseline variables used in modeling disease progression. In this example, you cluster the continuous baseline variables.

1. Select **Help > Sample Data Library** and open Diabetes.jmp.
2. Select **Analyze > Clustering > Cluster Variables.**
3. Select the columns Age through Glucose except for Gender (Age, BMI, BP, Total Cholesterol, LDL, HDL, TCH, LTG, and Glucose) and click **Y, Columns**.

   The Gender column cannot be included because Cluster Variables requires numeric continuous variables.
4. Click **OK**.

Figure 16.2 Cluster Variables Report for Diabetes Data



Variable Clustering

Color Map on Correlations

-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Cluster Summary

Cluster	Number of Members	Most Representative Variable	Cluster Proportion of Variation Explained	Total Proportion of Variation Explained
1	4	TCH	0.628	0.279
2	2	Total Cholesterol	0.948	0.211
3	3	BP	0.562	0.187

.2 .4 .6 .8

Proportion of variation explained by clustering: 0.677

Cluster Members

Cluster	Members	RSquare with Own Cluster	RSquare with Next Closest	1-RSquare Ratio
1	TCH	0.804	0.381	0.317
1	HDL	0.651	0.056	0.37
1	LTG	0.605	0.255	0.53
1	BMI	0.451	0.189	0.678
2	Total Cholesterol	0.948	0.163	0.062
2	LDL	0.948	0.218	0.066
3	BP	0.606	0.142	0.459
3	Age	0.508	0.061	0.523
3	Glucose	0.572	0.235	0.56

Standardized Components

Variable	Cluster 1 Coefficients	Cluster 2 Coefficients	Cluster 3 Coefficients
Age	0	0	0.5489935
BMI	0.4237035	0	0
BP	0	0	0.599546
Total Cholesterol	0	0.7071068	0
LDL	0	0.7071068	0
HDL	-0.509131	0	0
TCH	0.5658588	0	0
LTG	0.4909836	0	0
Glucose	0	0	0.5823665

The Cluster Summary report shows that the variables were grouped into three clusters:

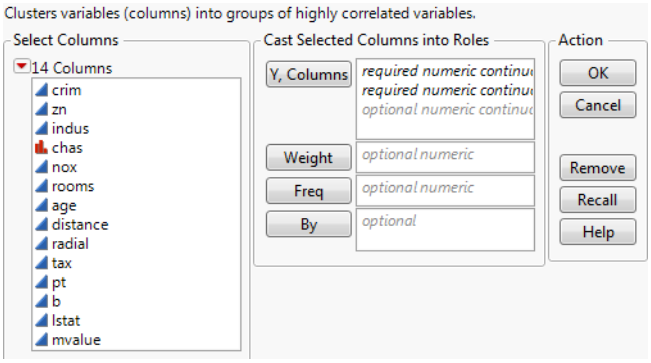
- Cluster 1 consists of TCH, HDL, LTG, and BMI, as shown in the Cluster Members report. The Cluster Summary report shows that TCH is the most representative variable for Cluster 1 and that for the variables in Cluster 1, 62.8% of the variation is explained by the first principal component.
- Cluster 2 consists of Total Cholesterol and LDL. The Cluster summary report shows that Total Cholesterol is the most representative variable for Cluster 2 and that for the variables in Cluster 2, 94.8% of the variation is explained by the first principal component.

- Cluster 3 consists of BP, Age, and Glucose. The Cluster Summary report shows that the most representative variable is BP and that for the variables in Cluster 3, 56.2% of the variation is explained by the first principal component.

Launch the Cluster Variables Platform

Launch the Cluster Variables platform by selecting **Analyze > Clustering > Cluster Variables**.

Figure 16.3 Cluster Variables Launch Dialog



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Columns The variables to be clustered. Variables must be numeric and continuous.

Weight A column whose numeric values assign a weight to each row in the analysis.

Freq A column whose numeric values assign a frequency to each row in the analysis.

By A column whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed. The results are presented in separate reports. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

The Cluster Variables Report

By default, the Cluster Variables report displays the following:

- "Color Map on Correlations"
- "Cluster Summary"

- "Cluster Members"
- "Standardized Components"

Tip: In any of the Cluster Variables reports, select rows in order to select the corresponding columns in the data table. Press Ctrl and click the row to deselect the column in the data table.

Color Map on Correlations

The Color Map on Correlations report displays a color map of the correlations between variables. The variables are arranged in the order in which they are listed in the Cluster Members report. This arrangement ensures that members of the same cluster are adjacent in the correlation plot. See "Example of Color Map on Correlations".

Tip: Hover over a square on the color map to see the variables involved in that square and their correlation.

Variables in the same cluster tend to have higher absolute correlations (deeper red or blue colors) than variables in different clusters. Therefore, the squares formed by the cells of the correlation map that correspond to the variables for a given component often stand out along the diagonal.

Correlations are computed using the row-wise method. This method excludes any observation with missing data on any of the variables from the correlation calculation. For more information about the row-wise estimation method, see "Estimation Methods".

Cluster Summary

The Cluster Summary report gives the following information:

Cluster The cluster identifier.

Number of Members The number of variables in the cluster.

Most Representative Variable The cluster variable that has the largest squared correlation with its cluster component.

Cluster Proportion of Variance Explained The cluster's proportion of variance explained by the first principal component among the variables in the cluster. If there is only one variable in the cluster, then this is 1. This statistic is based only on variables within the cluster rather than on all variables.

Total Proportion of Variation Explained The overall proportion of variance explained by the cluster component. This is equivalent to using only the variables within each cluster to calculate the first principal component.

A note beneath the table gives the total proportion of variation explained by all the cluster components.

Cluster Members

The Cluster Members report gives the following:

Cluster The cluster identifier.

Members The variables included in the cluster.

RSquare with Own Cluster The squared correlation of the variable with its cluster component.

RSquare with Next Closest The squared correlation of the variable with the cluster component for its next closest cluster. The next closest cluster is the cluster for which the squared correlation of the variable with the cluster component is the second highest.

1 - RSquare Ratio A measure of the relative closeness between the cluster to which a variable belongs and its next closest cluster. It is defined as follows:

(1 - RSquare with Own Cluster)/(1 - RSquare with Next Closest)

Standardized Components

The Standardized Components report gives the coefficients that define the cluster components. These coefficients are the eigenvectors of the first principal component within each cluster.

Cluster Variables Platform Options

The Variable Clustering red triangle menu contains the following options:

Color Map on Correlations Shows or hides the Color Map on Correlations plot. See "Color Map on Correlations".

Cluster Summary Shows or hides the Cluster Summary report. See "Cluster Summary".

Cluster Members Shows or hides the Cluster Members report. See "Cluster Members".

Cluster Components Shows or hides the Standardized Components report. See "Standardized Components".

Save Cluster Components Saves columns to the data table as a group called Cluster Components. Each column is named Cluster <i> Components and contains a formula that expresses the cluster component in terms of the uncentered and unscaled variables.

Launch Fit Model Opens a Model Specification window with the Most Representative Variables for each cluster entered in the Construct Model Effects list. Use this option to construct models based on the Most Representative variables.

Tip: To fit a model using the components, first select the **Save Cluster Components** option. Then replace the Most Representative variables for each cluster in the Construct Model Effects list with the desired Cluster Components columns.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Examples of the Cluster Variables Platform

- "Example of Color Map on Correlations"
- "Example of Cluster Variables Platform for Dimension Reduction"

Example of Color Map on Correlations

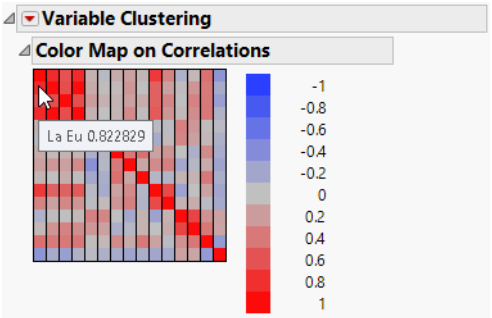
In this example, you construct and examine a Color Map on Correlations.

1. Select **Help > Sample Data Library** and open Cherts.jmp.
2. Select **Analyze > Clustering > Cluster Variables.**
3. Select all continuous variables and click **Y, Columns**.

4. Click **OK**.
5. Close the Cluster Summary and the Standardized Components reports.
6. Hover over the cell in the second row and first column of the Color Map.

   A tooltip appears, showing that the variables corresponding to this cell are La and Eu, and that their correlation is 0.822829.

Figure 16.4 Color Map on Correlations for Cherts.jmp



Cluster	Members	RSquare with Own Cluster	RSquare with Next Closest	1-RSquare Ratio
1	La	0.925	0.148	0.089
1	Sm	0.926	0.193	0.091
1	Eu	0.819	0.354	0.28
1	Sc	0.606	0.208	0.498
2	Na	0.873	0.072	0.137
2	Al	0.873	0.098	0.141
3	Mn	0.73	0.078	0.293
3	Co	0.499	0.112	0.564
3	Br	0.437	0.125	0.644
4	Hf	0.83	0.083	0.186
4	Ce	0.83	0.421	0.294
5	Cs	0.842	0.062	0.168
5	Cr	0.842	0.128	0.181
6	U	0.646	0.056	0.374
6	Fe	0.646	0.199	0.442

The Cluster Members report shows that there are four variables in Cluster 1. In the Color Map on Correlations, the four-by-four square of cells in the upper left corner that corresponds to these five variables shows a distinct pattern of positive correlations. The color map also shows patterns of positive correlations for the variables in Clusters 2, 4, and 5. The two-by-two square of cells in the lower right corner of the color map that corresponds to the two Cluster 6 variables shows that they are negatively correlated. See "Color Map on Correlations".

Example of Cluster Variables Platform for Dimension Reduction

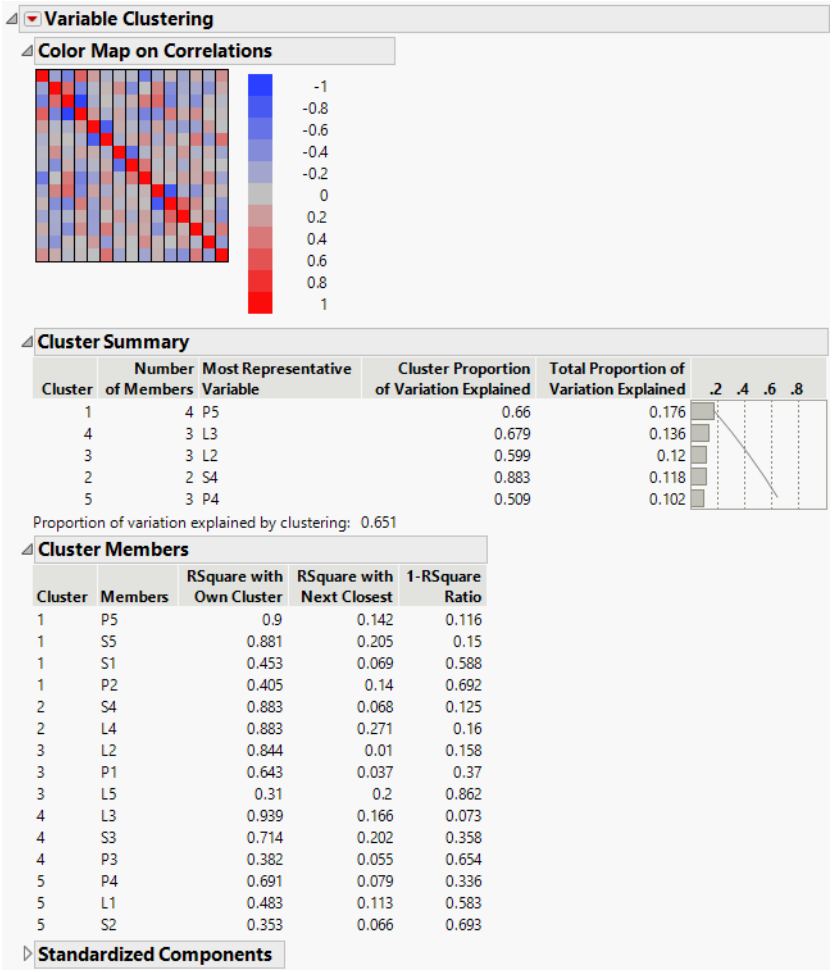
In this example, you use the Cluster Variables platform as a dimension-reduction tool for modeling. The Penta.jmp sample data table contains 15 variables used to predict the response variable, log RAI. Use Cluster Variables to reduce this number.

Cluster Variables

1. Select **Help > Sample Data Library** and open Penta.jmp.
2. Select **Analyze > Clustering > Cluster Variables**.
3. Select all of the continuous variables, except logRAI and click **Y, Columns**.
4. Click **OK**.
5. Click the Variable Clustering red triangle and select **Save Cluster Components**.

   Five grouped formula columns are added to the data table.

Figure 16.5 Cluster Variables Report for Penta.jmp



Variable Clustering

Color Map on Correlations

-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Cluster Summary

Cluster	Number of Members	Most Representative Variable	Cluster Proportion of Variation Explained	Total Proportion of Variation Explained
1	4	P5	0.66	0.176
4	3	L3	0.679	0.136
3	3	L2	0.599	0.12
2	2	S4	0.883	0.118
5	3	P4	0.509	0.102

.2 .4 .6 .8

Proportion of variation explained by clustering: 0.651

Cluster Members

Cluster	Members	RSquare with Own Cluster	RSquare with Next Closest	1-RSquare Ratio
1	P5	0.9	0.142	0.116
1	S5	0.881	0.205	0.15
1	S1	0.453	0.069	0.588
1	P2	0.405	0.14	0.692
2	S4	0.883	0.068	0.125
2	L4	0.883	0.271	0.16
3	L2	0.844	0.01	0.158
3	P1	0.643	0.037	0.37
3	L5	0.31	0.2	0.862
4	L3	0.939	0.166	0.073
4	S3	0.714	0.202	0.358
4	P3	0.382	0.055	0.654
5	P4	0.691	0.079	0.336
5	L1	0.483	0.113	0.583
5	S2	0.353	0.066	0.693

Standardized Components

The Cluster Summary and Cluster Members reports show that the variables are clustered into five groups, so there are five Cluster Component variables.

Fit Models

Next, fit and compare two models to predict logRAI:

- A model using all continuous variables as predictors.
- A model using the Cluster Components as predictors.

1. Click the Variable Clustering red triangle and select **Launch Fit Model**.
2. Select logRAI and click **Y**.

Notice that the Most Representative Variables the five clusters have been entered in the Construct Model Effects list. However, you want to enter all predictors.

3. Select all of the continuous variables from S1 to P5 and click **Add**.

   Be careful not to include Obs Name.

4. Select the box next to **Keep dialog open**.
5. Click **Run**.

Figure 16.6 Fit Least Squares Report for Model with All Continuous Predictors

Response log RAI

Effect Summary

Summary of Fit

RSquare	0.929316
RSquare Adj	0.853582
Root Mean Square Error	0.331225
Mean of Response	0.734333
Observations (or Sum Wgts)	30

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	15	20.193596	1.34624	12.2709
Error	14	1.535941	0.10971	Prob > F
C. Total	29	21.729537		<.0001*

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	-0.802632	0.924946	-0.87	0.4002
P5	2.0563803	1.651272	1.25	0.2335
S4	-0.062354	0.134935	-0.46	0.6511
L2	0.0860287	0.061206	1.41	0.1817
L3	0.3185383	0.080091	3.98	0.0014*
P4	0.4136598	0.394449	1.05	0.3121
S1	-0.09783	0.038948	-2.51	0.0249*
L1	0.032362	0.049732	0.65	0.5258
P1	-0.107951	0.085209	-1.27	0.2259
S2	0.086703	0.044276	1.96	0.0704
P2	0.0847235	0.086297	0.98	0.3429
S3	-0.037728	0.055602	-0.68	0.5085
P3	-0.027313	0.233655	-0.12	0.9086
L4	-0.029756	0.152012	-0.20	0.8476
S5	2.7123146	2.222039	1.22	0.2424
L5	-0.209128	0.270401	-0.77	0.4521

Effect Tests

Effect Details

6. In the Fit Model window, select all variables in the Construct Model Effects window and click **Remove**.
7. Select the Cluster Components group and click **Add**.
8. Click **Run**.

Figure 16.7 Fit Least Squares Report for Model with Cluster Components as Predictors

Response log RAI

Effect Summary

Summary of Fit

RSquare	0.8214
RSquare Adj	0.784191
Root Mean Square Error	0.402125
Mean of Response	0.734333
Observations (or Sum Wgts)	30

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	5	17.848635	3.56973	22.0757
Error	24	3.880902	0.16170	Prob > F
C. Total	29	21.729537		<.0001*

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	0.6651552	0.074349	8.95	<.0001*
Cluster 1 Components	-0.018483	0.056013	-0.33	0.7443
Cluster 2 Components	0.0035891	0.069032	0.05	0.9590
Cluster 3 Components	0.2043072	0.066714	3.06	0.0053*
Cluster 4 Components	0.5754553	0.065423	8.80	<.0001*
Cluster 5 Components	-0.046594	0.069786	-0.67	0.5107

Effect Tests

Effect Details

The model that includes the five Cluster Components as the only predictors explains a substantial amount of the variation in the response, with an adjusted RSquare of 0.784. The model that uses all fifteen predictors has only a slightly higher adjusted RSquare of 0.853 (Figure 16.6).

Statistical Details for the Cluster Variables Platform

Variable Clustering Algorithm

The clustering algorithm iteratively splits clusters of variables and reassigns variables to clusters until no more splits are possible. The initial cluster consists of all variables. The algorithm was developed by SAS and is implemented in PROC VARCLUS (SAS Institute Inc. 2020g).

Note: The algorithm uses only observations for which there are no missing values for any variable in the Y, Columns list.

These are the iterative steps in the algorithm:

1. For all clusters, do the following:
   a. Compute the principal components for the variables in each cluster.

   b. If the second eigenvalues for all of the clusters are less than one, then terminate the algorithm.

2. Partition the cluster whose second eigenvalue is the largest (and greater than 1) into two new clusters using the following steps:
   a. Rotate the principal components for the variables in the current cluster using an orthoblique rotation.
   b. Define one cluster to consist of the variables in the current cluster whose squared correlations to the first rotated principal component are higher than their squared correlations to the second principal component.
   c. Define the other cluster to consist of the remaining variables in the original cluster. These are the variables that are more highly correlated with the second principal component.
   d. Compute the principal components of the two new clusters.
3. Test to see whether any variable in the data set should be assigned to a different cluster. For each variable, do the following:
   a. Compute the variable's squared correlation with the first principal component for each cluster.
   b. Place the variable in the cluster for which its squared correlation is the largest.

Note: An orthoblique rotation is also known as a raw quartimax rotation. See Harris and Kaiser (1964).

Appendix A

Statistical Details

Multivariate Methods

This appendix discusses Wide Linear methods and the use of the singular value decomposition.

Contents

Wide Linear Methods and the Singular Value Decomposition 363
The Singular Value Decomposition 363
The SVD and the Covariance Matrix 364
The SVD and the Inverse Covariance Matrix 364
Calculating the SVD 365

Wide Linear Methods and the Singular Value Decomposition

Wide Linear methods in the Cluster, Principal Components, and Discriminant platforms enable you to analyze data sets with thousands (or even millions) of variables. Most multivariate techniques require the calculation or inversion of a covariance matrix. When your multivariate analysis involves a large number of variables, the covariance matrix can be prohibitively large so that calculating it or inverting it is problematic and computationally expensive.

Suppose that your data consist of n rows and p columns. The rank of the covariance matrix is at most the smaller of n and p. In wide data sets, p is often much larger than n. In these cases, the inverse of the covariance matrix has at most n nonzero eigenvalues. Wide Linear methods use this fact, together with the singular value decomposition, to provide efficient calculations. See "Calculating the SVD".

The Singular Value Decomposition

The *singular value decomposition* (SVD) enables you to express any linear transformation as a rotation, followed by a scaling, followed by another rotation. The SVD states that any n by p matrix $\mathbf{X}$ can be written as follows:

$$\mathbf{X} = \mathbf{U}Diag(\Lambda)\mathbf{V}'$$

Let r be the rank of $\mathbf{X}$. Denote the r by r identity matrix by $\mathbf{I}_r$.

The matrices $\mathbf{U}$, $Diag(\Lambda)$, and $\mathbf{V}$ have the following properties:

$\mathbf{U}$ is an n by r semi-orthogonal matrix with $\mathbf{U}'\mathbf{U} = \mathbf{I}_r$

$\mathbf{V}$ is a p by r semi-orthogonal matrix with $\mathbf{V}'\mathbf{V} = \mathbf{I}_r$

$Diag(\Lambda)$ is an r by r diagonal matrix with positive diagonal elements given by the column vector $\Lambda = (\lambda_1, \lambda_2, ..., \lambda_r)'$ where $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_r > 0$.

The λ_i are the nonzero *singular values* of $\mathbf{X}$.

The following statements relate the SVD to the spectral decomposition of a square matrix:

- The squares of the λ_i are the nonzero eigenvalues of $\mathbf{X}'\mathbf{X}$.
- The r columns of $\mathbf{V}$ are eigenvectors of $\mathbf{X}'\mathbf{X}$.

Note: There are various conventions in the literature regarding the dimensions of the matrices **U**, **V**, and the matrix containing the singular values. However, the differences have no practical impact on the decomposition up to the rank of **X**.

For more information about singular value decomposition, see Press et al. (1998, Section 2.6).

The SVD and the Covariance Matrix

This section describes how the eigenvectors and eigenvalues of a covariance matrix can be obtained using the SVD. When the matrix of interest has at least one large dimension, calculating the SVD is much more efficient than calculating its covariance matrix and its eigenvalue decomposition.

Let n be the number of observations and p the number of variables involved in the multivariate analysis of interest. Denote the n by p matrix of data values by **X**.

The SVD is usually applied to standardized data. To standardize a value, subtract its mean and divide by its standard deviation. Denote the n by p matrix of standardized data values by $\mathbf{X}_s$. Then the covariance matrix of the standardized data is the correlation matrix for **X** and is defined as follows:

$$Cov = \mathbf{X}_s'\mathbf{X}_s/(n-1)$$

The SVD can be applied to $\mathbf{X}_s$ to obtain the eigenvectors and eigenvalues of $\mathbf{X}_s'\mathbf{X}_s$. This allows efficient calculation of eigenvectors and eigenvalues when the matrix **X** is either extremely wide (many columns) or tall (many rows). This technique is the basis for Wide PCA. See "Principal Components Report".

The SVD and the Inverse Covariance Matrix

Some multivariate techniques require the calculation of inverse covariance matrices. This section describes how the SVD can be used to calculate the inverse of a covariance matrix.

Denote the standardized data matrix by $\mathbf{X}_s$ and define $\mathbf{S} = \mathbf{X}_s'\mathbf{X}_s$. The singular value decomposition allows you to write **S** as follows:

$$\mathbf{S} = (\mathbf{U}Diag(\Lambda)\mathbf{V}')'(\mathbf{U}Diag(\Lambda)\mathbf{V}') = \mathbf{V}Diag(\Lambda)^2\mathbf{V}'$$

If **S** is of full rank, then **V** is a p by p orthonormal matrix, and you can write $\mathbf{S}^{-1}$ as follows:

$$\mathbf{S}^{-1} = (\mathbf{V}Diag(\Lambda)^2\mathbf{V}')^{-1} = \mathbf{V}Diag(\Lambda)^{-2}\mathbf{V}'$$

If **S** is not of full rank, then $Diag(\Lambda)^{-1}$ can be replaced with a generalized inverse, $Diag(\Lambda)^{+}$, where the diagonal elements of $Diag(\Lambda)$ are replaced by their reciprocals. This defines a generalize inverse of **S** as follows:

$$\mathbf{S}^{-} = \mathbf{V}(Diag(\Lambda)^{+})^{2}\mathbf{V}'$$

This generalized inverse can be calculated using only the SVD.

For more information about the application of the SVD for wide linear discriminant analysis, see "Wide Linear Discriminant Method".

Calculating the SVD

In the Multivariate Methods platforms, JMP calculates the SVD of a matrix following the method suggested in Golub and Kahan (1965). Golub and Kahan's method involves a two-step procedure. The first step consists of reducing the matrix **M** to a bidiagonal matrix **J**. The second step consists of computing the singular values of **J**, which are the same as the singular values of the original matrix **M**. The columns of the matrix **M** are usually standardized in order to equalize the effect of the variables on the calculation. The Golub and Kahan method is computationally efficient.

Appendix B

References

Agresti, A. (2013). *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Baglama, J., and Reichel, L. (2005). "Augmented implicitly restarted Lanczos bidiagonalization methods." *SIAM Journal on Scientific Computing* 27:19–42.

Ballard, D. H. (1981). "Generalizing the Hough Transform to Detect Arbitrary Shapes." *Pattern Recognition* 13:111–122.

Bartlett, M. S. (1937). "Properties of sufficiency and statistical tests." *Proceedings of the Royal Society of London, Series A* 160:268–282.

Bartlett, M. S. (1954). "A Note on the Multiplying Factors for Various Chi Square Approximations." *Journal of the Royal Statistical Society, Series B* 16:296–298.

Benzécri, J. P. (1979). "Sur le calcul des taux d'inertie dans l'analyse d'un questionnaire, addendum et erratum à [BIN. MULT.]." *Cahiers de l'Analyse des Données* 4:377–378.

Bentler, P. M. (1990). "Comparative Fit Indexes in Structural Models." *Psychological Bulletin* 107:238.

Bentler, P. M., and Freeman, E. H. (1983). "Tests for Stability in Linear Structural Equation Systems." *Psychometrika* 48:143–145.

Bollen, K. A. (1989). *Structural Equations with Latent Variables*. New York: John Wiley & Sons.

Borg, I., and Groenen, P. J. F. (2005). *Modern Multidimensional Scaling: Theory and Applications.* 2nd ed. New York: Springer.

Boulesteix, A.-L., and Strimmer, K. (2007). "Partial Least Squares: A Versatile Tool for the Analysis of High-Dimensional Genomic Data." *Briefings in Bioinformatics* 8:32–44.

Browne, M. (2001). "An Overview of Analytic Rotation in Exploratory Factor Analysis." *Multivariate Behavioral Research* 36:111–150.

Browne, M. W., and Cudeck, R. (1993). "Alternative Ways of Assessing Model Fit." In *Testing Structural Equation Models*, edited by K. A. Bollen, and J. S. Long, 136–162. Newbury Park, CA: Sage Publications.

Chen, F. F. (2007). "Sensitivity of Goodness of Fit Indexes to Lack of Measurement Invariance." *Structural Equation Modeling: A Multidisciplinary Journal* 14:464–504.

Collins, L., and Lanza, S. (2010). *Latent Class and Latent Transition Analysis*. Hoboken NJ: John Wiley & Sons.

Cox, I., and Gaudard, M. (2013). *Discovering Partial Least Squares with JMP*. Cary, NC: SAS Institute Inc.

Cronbach, L. J. (1951). "Coefficient Alpha and the Internal Structure of Tests." *Psychometrika* 16:297–334.

Cudeck, R., and MacCallum, R. C., eds. (2007). Factor Analysis at 100, Historical Developments and Future Directions. Mahwah, NJ: Lawrence Erlbaum Associates.

de Ayala, R. J. (2009). *The Theory and Practice of Item Response Theory*. New York: Guilford Press.

De Jong, S. (1993). "SIMPLS: An Alternative Approach to Partial Least Squares Regression." *Chemometrics and Intelligent Laboratory Systems* 18:251–263.

Denham, M. C. (1997). "Prediction Intervals in Partial Least Squares." *Journal of Chemometrics* 11:39–52.

Eriksson, L., Johansson, E., Kettaneh-Wold, N., Trygg, J., Wikstrom, C., and Wold, S. (2006). *Multi- and Megavariate Data Analysis Basic Principles and Applications (Part I)*. Chapter 4. Umetrics.

Finkbeiner, C. (1979). "Estimation for the Multiple Factor Model when Data are Missing." *Psychometrika* 44:409–420.

Fisher, L., and Van Ness, J. W. (1971). "Admissible Clustering Procedures." *Biometrika* 58:91–104.

Florek, K., Lukaszewicz, J., Perkal, J., and Zubrzycki, S. (1951a). "Sur la liaison et la division des points d'un ensemble fini." *Colloquium Mathematicae* 2:282–285.

Florek, K., Lukaszewicz, J., Perkal, J., and Zubrzycki, S. (1951b). "Taksonomia Wroclawska." *Przeglad Antropologiczny* 17:193–211.

Frank, I. E., and Todeschini, T. (1994). *The Data Analysis Handbook*. New York: Elsevier.

Friedman, J. H. (1989). "Regularized Discriminant Analysis." *Journal of the American Statistical Association* 84:165–175.

Garthwaite, P. (1994). "An Interpretation of Partial Least Squares." *Journal of the American Statistical Association* 89:122–127.

Golub, G. H., and Kahan, W. (1965). "Calculating the singular values and pseudo-inverse of a matrix." *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis* 2:205–224.

Goodman, L. A. (1974). "Exploratory Latent Structure Analysis Using Both Identifiable and Unidentifiable Models." *Biometrika* 61:215–231.

Greenacre, M. J. (1984). *Theory and Applications of Correspondence Analysis*. London: Academic Press.

Hancock, G. R., and Mueller, R. O. (2001). "Rethinking Construct Reliability within Latent Variable Systems." In *Structural Equation Modeling: Present and Future – A Festschrift in Honor of Karl Jöreskog*, edited by R. Cudeck, S. du Toit, and D. Sörbom, 195–216. Lincolnwood, IL: Scientific Software International.

Hand, D., Mannila, H., and Smyth, P. (2001). *Principles of Data Mining*. Cambridge, MA: MIT Press.

Harris, C. W., and Kaiser, H. F. (1964). "Oblique Factor Analytic Solutions by Orthogonal Transformation." *Psychometrika* 32:363–379.

Hartigan, J. A. (1981). "Consistency of Single Linkage for High–Density Clusters." *Journal of the American Statistical Association* 76:388–394.

Hastie, T., Tibshirani, R., and Friedman, J. H.(2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer Verlag.

Hoskuldsson, A. (1988). "PLS Regression Methods." *Journal of Chemometrics* 2:211–228.

Hoeffding, W. (1948). "A Non-Parametric Test of Independence." *Annals of Mathematical Statistics* 19:546–557.

Hu, L.-T., and Bentler, P. M. (1999). "Cutoff Criteria for Fit Indices in Covariance Structure Analysis: Conventional Criteria versus New Alternatives." *Structural Equation Modeling* 6:1–55.

Huber, P. J. (1964). "Robust Estimation of a Location Parameter." *Annals of Mathematical Statistics* 35:73–101.

Huber, P. J. (1973). "Robust Regression: Asymptotics, Conjecture, and Monte Carlo." *Annals of Statistics* 1:799–821.

Huber, P. J., and Ronchetti, E. M. (2009). *Robust Statistics.* 2nd ed. Hoboken, NJ: John Wiley & Sons.

Jackson, J. E. (2003). *A User's Guide to Principal Components*. Hoboken, NJ: John Wiley & Sons.

Jardine, N., and Sibson, R. (1971). *Mathematical Taxonomy*. New York: John Wiley & Sons.

Jöreskog, K. G. (1977). "Factor Analysis by Least-Squares and Maximum Likelihood Methods." In *Statistical Methods for Digital Computers*, edited by K. Enslein, A. Ralston, and H. Wilf, 125 - 165. New York: John Wiley & Sons.

Kline, R. B. (2016). *Principles and Practice of Structural Equation Methodology*. 4th ed. New York: The Guilford Press.

Kohonen, T. (1989). *Self-Organization and Associative Memory*. 3rd ed. Vol. 8 of Springer Series in Information. Berlin: Springer-Verlag.

Kohonen, T. (1990). "The Self-Organizing Map." *Proceedings of the IEEE* 78:1464–1480.

Lindberg, W., Persson, J.-A., and Wold, S. (1983). "Partial Least-Squares Method for Spectrofluorimetric Analysis of Mixtures of Humic Acid and Ligninsulfonate." *Analytical Chemistry* 55:643–648.

LeRoux, B., and Rouanet, H. (2010). Multiple Correspondence Analysis. Vol.07–163 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Thousand Oaks, CA: Sage Publications.

Mardia, K., Kent, J., and Bibby, J. (1980). *Multivariate Analysis.* New York: Academic Press.

Mason, R. L., and Young, J. C. (2002). *Multivariate Statistical Process Control with Industrial Applications.* Philadelphia: SIAM.

Maydeu-Olivares, A., Shi, D., and Rosseel, Y. (2017). "Assessing Fit in Structural Equation Models: A Monte-Carlo Evaluation of RMSEA Versus SRMR Confidence Intervals and Tests of Close Fit." *Structural Equation Model: A Multidisciplinary Journal* 25:389–402.

McDonald, R. P. (1999). *Test Theory: A Unified Approach*. Mahwah, NJ: Erlbaum.

McLachlan, G. J., and Krishnan, T. (1997). *The EM Algorithm and Extensions*. New York: John Wiley & Sons.

McQuitty, L. L. (1957). "Elementary Linkage Analysis for Isolating Orthogonal and Oblique Types and Typal Relevancies." *Educational and Psychological Measurement* 17:207–229.

Milligan, G. W. (1980). "An Examination of the Effect of Six Types of Error Perturbation on Fifteen Clustering Algorithms." *Psychometrika* 45:325–342.

Nelson, P. R. C., Taylor, P. A., and MacGregor, J. F. (1996). "Missing Data Methods in PCA and PLS: Score calculations with incomplete observations." *Chemometrics and Intelligent Laboratory Systems* 35:45–65.

Nunnally, J. C. (1978). *Psychometric theory*. 2nd ed. New York: McGraw-Hill.

Penny, K. I. (1996). "Appropriate Critical Values When Testing for a Single Multivariate Outlier by Using the Mahalanobis Distance." *Journal of the Royal Statistical Society, Series C* 45:73–81.

Press, W. H, Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1998). *Numerical Recipes in C: The Art of Scientific Computing*. 2nd ed. Cambridge, England: Cambridge University Press.

Rasch, G. (1980). *Probabilistic Models for Some Intelligence and Attainment Tests*. Chicago: University of Chicago Press.

Rodriguez, A., Reise, S. P., and Haviland, M. G. (2016). "Evaluating Bifactor Models: Calculating and Interpreting Statistical Indices. *Psychological Methods* 21:137.

SAS Institute Inc. (1983). *SAS Technical Report A-108: Cubic Clustering Criterion*. Cary, NC: SAS Institute Inc. Accessed December 16, 2015. https://support.sas.com/kb/22/addl/fusion_22540_1_a108_5903.pdf.

SAS Institute Inc. (2020a). "The CALIS Procedure." *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/calis.pdf.

SAS Institute Inc. (2020b). "The CANDISC Procedure." *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/candisc.pdf.

SAS Institute Inc. (2020c). "The FACTOR Procedure." *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/factor.pdf.

SAS Institute Inc. (2020d). "The FASTCLUS Procedure." *SAS/STAT 15.2 User's Guide.* Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/fastclus.pdf.

SAS Institute Inc. (2020e). "The MIXED Procedure." *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/mixed.pdf.

SAS Institute Inc. (2020f). "The PLS Procedure." *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/pls.pdf.

SAS Institute Inc. (2020g). "The VARCLUS Procedure." *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/varclus.pdf.

Schafer, J., and Strimmer, K. (2005). "A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics." *Statistical Applications in Genetics and Molecular Biology* 4 Article 32.

Sneath, P. H. A. (1957). "The Application of Computers to Taxonomy." *Journal of General Microbiology* 17:201–226.

Sokal, R. R., and Michener, C. D. (1958). "A Statistical Method for Evaluating Systematic Relationships." *University of Kansas Science Bulletin* 38:1409–1438.

Steiger, J. H. (1989). *EzPATH: A Supplementary Module for SYSTAT and SYGRAPH*. Evanston, IL: Systat.

Steiger, J. H. (1990). "Structural Model Evaluation and Modification: An Interval Estimation Approach." *Multivariate Behavioral Research* 25:173–180.

Tobias, R. D. (1995). "An Introduction to Partial Least Squares Regression." In *Proceedings of the Twentieth Annual SAS Users Group International Conference*, 1250–1257. Cary, NC: SAS Institute Inc. http://www.sascommunity.org/sugi/SUGI95/Sugi-95-210%20Tobias.pdf.

Tracy, N. D., Young, J. C., and Mason, R. R. (1992). "Multivariate Control Charts for Individual Observations." *Journal of Quality Technology* 24:88–95.

Umetrics. (1995). *Multivariate Analysis (3-day course)*. Winchester, MA.

Waern, Y. (1972). "Structure in Similarity Matrices: A Graphic Approach." *Scandinavian Journal of Psychology* 13:5–16.

West, S. G., Taylor, A. B., and Wu, W. (2012). "Model Fit and Model Selection in Structural Equation Modeling." In *Handbook of Structural Equation Modeling*, edited by R. H. Hoyle, 209–231. New York: The Guilford Press.

White, K. P., Jr., Kundu, B., and Mastrangelo, C. M. (2008). "Classification of Defect Clusters on Semiconductor Wafers Via the Hough Transform." *IEEE Transactions on Semiconductor Manufacturing* 21:272–278.

Wold, S. (1994). "PLS for Multivariate Linear Modeling." In *QSAR: Chemometric Methods in Molecular Design. Methods and Principles in Medicinal Chemistry*, edited by H. van de Waterbeemd, pp. 195–218. Weinheim, Germany: Verlag-Chemie.

Wold, S., Sjostrom, M., and Eriksson, L. (2001). "PLS-Regression: A Basic Tool of Chemometrics." *Chemometrics and Intelligent Laboratory Systems* 58:109–130.

Appendix C

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

  - You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - You must cause any modified files to carry prominent notices stating that You changed the files; and
  - You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
  - You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Quality and Process Methods

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Quality and Process Methods*. Cary, NC: SAS Institute Inc.

JMP® 16 Quality and Process Methods

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Quality and Process Methods

1 Learn about JMP 13
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 15
JMP Help 16
JMP Documentation Library 16
Additional Resources for Learning JMP 22
JMP Tutorials 22
Sample Data Tables 22
Learn about Statistical and JSL Terms 23
Learn JMP Tips and Tricks 23
JMP Tooltips 23
JMP User Community 24
Free Online Statistical Thinking Course 24
JMP New User Welcome Kit 24
Statistics Knowledge Portal 24
JMP Training 24
JMP Books by Users 25
The JMP Starter Window 25
JMP Technical Support 25

2 Introduction to Quality and Process Methods 27
Tools for Process and Product Improvement

3 Control Chart Builder 29
Create Control Charts Interactively
Overview of Control Chart Builder 32
Example of Control Chart Builder 33
Control Chart Types 35
Shewhart Control Charts for Variables 35
Shewhart Control Charts for Attributes 36
Rare Event Control Charts 37
Control Chart Types 38
Launch Control Chart Builder 41

Control Chart Builder Interactive Workspace ... 42
Launch Windows for Specific Control Charts ... 44
Control Chart Builder Window ... 46
Control Chart Builder Options ... 47
Control Chart Builder Red Triangle Menu Options ... 48
Options Panel and Right-Click Chart Options ... 51
Control Chart Builder Right-Click Axis Options ... 61
Work with Control Limits ... 62
Example of Control Limits ... 62
Excluded and Hidden Samples in Control Chart Builder ... 70
Additional Examples of Control Chart Builder ... 71
Individual Measurement and Moving Range Charts Example ... 72
XBar and R Chart Phase Example ... 73
XBar and S Charts with Varying Subgroup Sizes Example ... 76
Run Chart Example ... 78
P chart Example ... 79
NP chart Example ... 80
C chart Example ... 82
U chart Example ... 83
G chart Example ... 84
T chart Example ... 85
Three Way Control Chart Example ... 87
Example of Multiple Control Charts ... 89
Statistical Details for Control Chart Builder ... 89
Control Limits for XBar and R Charts ... 90
Control Limits for XBar and S Charts ... 91
Control Limits for Individual Measurement and Moving Range Charts ... 92
Control Limits for P and NP Charts ... 93
Control Limits for U Charts and C Charts ... 94
Levey-Jennings Charts ... 95
Control Limits for G Charts ... 95
Control Limits for T Charts ... 96
Sigma Calculations for Three Way Control Charts ... 97

4 Measurement Systems Analysis ... 99
Evaluate a Continuous Measurement Process Using the EMP Method

Overview of Measurement Systems Analysis ... 101
Example of Measurement Systems Analysis ... 101
Launch the Measurement Systems Analysis Platform ... 105
Data Format ... 106

Measurement Systems Analysis Platform Options ... 107
Average Chart ... 109
Range Chart or Standard Deviation Chart ... 109
EMP Results ... 110
Effective Resolution ... 111
Shift Detection Profiler ... 112
Bias Comparison ... 117
Test-Retest Error Comparison ... 118
Additional Example of Measurement Systems Analysis ... 118
Statistical Details for Measurement Systems Analysis ... 124
Computation of Intraclass Correlation and Probable Error ... 124

5 Variability Gauge Charts ... 127
Evaluate a Continuous Measurement Process Using Gauge R&R

Overview of Variability Charts ... 129
Example of a Variability Chart ... 130
Launch the Variability/Attribute Gauge Chart Platform ... 131
Data Format ... 132
The Variability Gauge Chart ... 132
Variability Gauge Platform Options ... 134
Heterogeneity of Variance Tests ... 136
Variance Components ... 137
About the Gauge R&R Method ... 139
Gauge R&R Option ... 140
Discrimination Ratio ... 143
Misclassification Probabilities ... 143
Bias Report ... 144
Linearity Study ... 144
Additional Examples of Variability Charts ... 145
Example of the Heterogeneity of Variance Test ... 145
Example of the Bias Report Option ... 148
Statistical Details for Variability Charts ... 151
Statistical Details for Variance Components ... 151
Statistical Details for the Discrimination Ratio ... 152
Statistical Details for the Misclassification Probabilities ... 153

6 Attribute Gauge Charts ... 155
Evaluate a Categorical Measurement Process Using Agreement Measures

Overview of Attribute Gauge Charts ... 157
Example of an Attribute Gauge Chart ... 157
Launch the Variability/Attribute Gauge Chart Platform ... 159

The Attribute Gauge Chart and Reports . . . 160
Agreement Reports . . . 161
Effectiveness Report . . . 162
Attribute Gauge Platform Options . . . 164
Statistical Details for Attribute Gauge Charts . . . 165
Statistical Details for the Agreement Report . . . 167

7 Process Capability . . . 169
Measure the Variability of a Process over Time

Overview of the Process Capability Platform . . . 171
Example of the Process Capability Platform with Normal Variables . . . 173
Example of the Process Capability Platform with Nonnormal Variables . . . 175
Launch the Process Capability Platform . . . 181
Process Selection . . . 182
Process Subgrouping . . . 182
Moving Range Options . . . 183
Historical Information . . . 184
Distribution Options . . . 184
Other Specifications . . . 185
Entering Specification Limits . . . 186
Spec Limits Window . . . 186
Limits Data Table . . . 187
Spec Limits Column Property . . . 189
The Process Capability Report . . . 189
Goal Plot . . . 190
Capability Box Plots . . . 193
Capability Index Plot . . . 195
Process Capability Platform Options . . . 197
Individual Detail Reports . . . 200
Normalized Box Plots . . . 208
Process Performance Plot . . . 209
Summary Reports . . . 211
Make Goal Plot Summary Table . . . 212
Additional Examples of the Process Capability Platform . . . 213
Process Capability for a Stable Process . . . 213
Process Capability for an Unstable Process . . . 217
Simulation of Confidence Limits for a Nonnormal Process Ppk . . . 220
Statistical Details for the Process Capability Platform . . . 227
Variation Statistics . . . 227
Notation for Goal Plots and Capability Box Plots . . . 231

Goal Plot ... 231
Capability Box Plots for Processes with Missing Targets ... 233
Capability Indices for Normal Distributions ... 234
Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods ... 239
Parameterizations for Distributions ... 241

8 CUSUM Control Charts ... 245
Create Tabular CUSUM Control Charts with Decision Limits
Overview of the CUSUM Control Chart Platform ... 247
Example of CUSUM Control Chart ... 247
Launch the CUSUM Control Chart Platform ... 249
The CUSUM Control Chart Platform Report ... 250
Control Panel for CUSUM Control Chart ... 251
CUSUM Chart ... 252
CUSUM Control Chart Platform Options ... 252
Average Run Length (ARL) Report ... 255
Additional Examples of CUSUM Control Charts ... 255
Example of the Data Units Option in a CUSUM Control Chart ... 256
Example of CUSUM Chart with Subgroups ... 256
Statistical Details for the CUSUM Control Chart Platform ... 258
Statistical Details for CUSUM Control Chart Construction ... 258
Statistical Details for Shift Detection ... 259
Statistical Details for Average Run Length ... 259

9 EWMA Control Charts ... 261
Create Exponentially Weighted Control Charts
Overview of the EWMA Control Chart Platform ... 263
Example of the EWMA Control Chart Platform ... 263
Launch the EWMA Control Chart Platform ... 265
The EWMA Control Chart Platform Report ... 266
Control Panel for EWMA Control Chart ... 266
EWMA Chart Report ... 267
The EWMA Control Chart Platform Options ... 269
Average Run Length (ARL) Report for EWMA Control Charts ... 272
Additional Example of the EWMA Control Chart Platform ... 273
Statistical Details for the EWMA Control Chart Platform ... 276

10 Multivariate Control Charts ... 279
Monitor Multiple Process Characteristics Simultaneously
Overview of Multivariate Control Charts ... 281
Example of a Multivariate Control Chart ... 281

Launch the Multivariate Control Chart Platform ... 284
The Multivariate Control Chart ... 285
Multivariate Control Chart Platform Options ... 286
T Square Partitioned ... 287
Change Point Detection ... 288
Principal Components ... 289
Additional Examples of Multivariate Control Charts ... 289
Example of Monitoring a Process Using Sub-Grouped Data ... 289
Example of T Square Partitioned ... 292
Example of Change Point Detection ... 294
Statistical Details for Multivariate Control Charts ... 295
Statistical Details for Individual Observations ... 296
Statistical Details for Observations in Rational Subgroups ... 297
Statistical Details for Change Point Detection ... 300

11 Model Driven Multivariate Control Charts ... 305
Monitor and Diagnosis a Complex Process

Overview of Model Driven Multivariate Control Charts ... 307
Example of Model Driven Multivariate Control Charts ... 307
Launch the Model Driven Multivariate Control Chart Platform ... 310
Data Format ... 311
The Model Driven Multivariate Control Chart Report ... 312
Model Driven Multivariate Control Chart Platform Options ... 312
Plot Options ... 313
Score Plot ... 315
Additional Examples of the Model Driven Multivariate Control Chart Platform ... 315
Example of an MDMVCC with Historical Data ... 316
Example of an MDMVCC with a PLS Model ... 317
Statistical Details for the Model Driven Multivariate Control Chart Platform ... 318
Monitoring Statistics ... 318
Limits ... 320
Contributions ... 323
Score Plot Group Comparisons ... 325
Score Plot Loadings ... 325

12 Legacy Control Charts ... 327
Create Variable and Attribute Control Charts

Example of a Legacy Control Chart ... 329
Legacy Control Chart Types ... 330
Control Charts for Variables ... 330
Control Charts for Attributes ... 333

Launch a Legacy Control Chart Platform ... 334
Process Information ... 335
Limits Specifications ... 338
Specified Statistics ... 339
Legacy Control Chart Reports ... 340
V-Mask CUSUM Chart Reports ... 342
Interpret a Two-Sided V-Mask CUSUM Chart ... 342
Interpret a One-Sided CUSUM Chart ... 343
Legacy Control Chart Platform Options ... 344
Window Options for Legacy Control Charts ... 344
Chart Options for Legacy Control Charts ... 346
Chart Options for V-Mask CUSUM Control Charts ... 349
Saving and Retrieving Limits ... 349
Excluded, Hidden, and Deleted Samples ... 352
Additional Examples of the Control Chart Platform ... 354
Presummarize Chart Example ... 354
V-Mask CUSUM Chart Example ... 356
One-Sided CUSUM Chart Example ... 358
UWMA Chart Example ... 359
Statistical Details for the Control Chart Platform ... 360
Control Limits for Median Moving Range Charts ... 360
Statistical Details for Capability Analysis ... 360
Statistical Details for V-Mask CUSUM Control Charts ... 367
Statistical Details for Weighted Moving Average Charts ... 370

13 Pareto Plots ... 373
Focus Improvement Efforts on the Vital Few

Overview of the Pareto Plot Platform ... 375
Example of the Pareto Plot Platform ... 375
Launch the Pareto Plot Platform ... 378
The Pareto Plot Report ... 379
Pareto Plot Platform Options ... 380
Causes Options ... 382
Additional Examples of the Pareto Plot Platform ... 382
Threshold of Combined Causes Example ... 383
Using a Constant Size across Groups Example ... 384
Using a Non-Constant Sample Size across Groups Example ... 386
One-Way Comparative Pareto Plot Example ... 387
Two-Way Comparative Pareto Plot Example ... 389
Statistical Details for the Pareto Plot Platform ... 390

Likelihood Ratio Chi-Square Test 390

14 Cause-and-Effect Diagrams 393
Identify Root Causes

Overview of Cause-and-Effect Diagrams 395
Example of a Cause-and-Effect Diagram 395
Prepare the Data 396
Launch the Diagram Platform 396
The Cause-and-Effect Diagram 397
Right-Click Menus 397
Cause and Effect Diagram Menu Options 400
Save the Diagram 401
Save the Diagram as a Data Table 401
Save the Diagram as a Journal 401
Save the Diagram as a Script 402

15 Quality Utilities 403
Manage Specification Limits and Create OC Curves

Manage Spec Limits Utility 405
Example of the Manage Spec Limits Utility 405
Manage Spec Limits Options 408
Operating Characteristic Curves Utility 409
Launch the OC Curves Utility 409
OC Curves for Control Chart Options 410
OC Curves for Acceptance Sampling Options 410

A References 413

B Technology License Notices 417

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 15
JMP Help . . . 16
JMP Documentation Library . . . 16
Additional Resources for Learning JMP . . . 22
JMP Tutorials . . . 22
Sample Data Tables . . . 22
Learn about Statistical and JSL Terms . . . 23
Learn JMP Tips and Tricks . . . 23
JMP Tooltips . . . 23
JMP User Community . . . 24
Free Online Statistical Thinking Course . . . 24
JMP New User Welcome Kit . . . 24
Statistics Knowledge Portal . . . 24
JMP Training . . . 24
JMP Books by Users . . . 25
The JMP Starter Window . . . 25
JMP Technical Support . . . 25

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction to Quality and Process Methods

Tools for Process and Product Improvement

Quality and Process Methods describes a number of methods and tools that are available in JMP to help you evaluate and improve quality and process performance:

- Control charts provide feedback on key variables and show when a process is in, or out of, statistical control. "Control Chart Builder" describes the JMP approach to creating control charts using an interactive control chart platform called Control Chart Builder.
- The Measurement Systems Analysis platform assesses the precision, consistency, and bias of a system. Before you can study a process, you need to make sure that you can accurately and precisely measure the process. If variation comes from the measurement itself, then you are not reliably learning about the process. Use this analysis to find out how your system is performing. See "Measurement Systems Analysis".
- The Variability/Attribute Gauge Chart platform creates variability or attribute gauge charts. Variability charts analyze continuous measurements and reveal how your system is performing. Attribute charts analyze categorical measurements and show you measures of agreement across responses. You can also perform a gauge study to see measures of variation in your data. See "Variability Gauge Charts" and "Attribute Gauge Charts".
- The Process Capability platform measures the ability of a process to meet specification limits. You can compare process performance, summarized by process centering and variability, to specification limits. The platform calculates capability indices based on both long-term and short-term variation. The analysis helps identify the variation relative to the specifications; this enables you to achieve increasingly higher conformance values. See "Process Capability".
- CUSUM charts enable you to make decisions based on the cumulative sum. These charts can detect small shifts in a process. See "CUSUM Control Charts".
- Exponentially weighted moving average (EWMA) charts can also be used to detect small shifts in a process. See "EWMA Control Charts".
- When you need to monitor multiple process characteristics simultaneously, see "Multivariate Control Charts".
- The Model Driven Multivariate Control Chart (MDMVCC) platform enables you to build a control chart based on principal components or partial least squares models. See "Model Driven Multivariate Control Charts".
- "Legacy Control Charts" describes the older control chart platforms in JMP. Instead of using these platforms, you are encouraged to use the Control Chart Builder platform, as well as the new CUSUM and EWMA Control Chart platforms.

- The Pareto Plot platform shows the frequency of problems in a quality related process or operation. Pareto plots help you decide which problems to solve first by highlighting the frequency and severity of problems. See “Pareto Plots”.
- The Diagram platform constructs cause-and-effect diagrams, which organize the sources of a problem for brainstorming or as a preliminary analysis to identify variables for further experimentation. Once complete, further analysis can be done to identify the root cause of the problem. See “Cause-and-Effect Diagrams”.
- The Manage Spec Limits utility enables you to quickly add or edit many specification limits for several columns at once. See “Manage Spec Limits Utility”.
- The Operating Characteristic (OC) Curves utility enables you to construct OC curves for control charts and attribute acceptance sampling plans. See “Operating Characteristic Curves Utility”.

Chapter 3

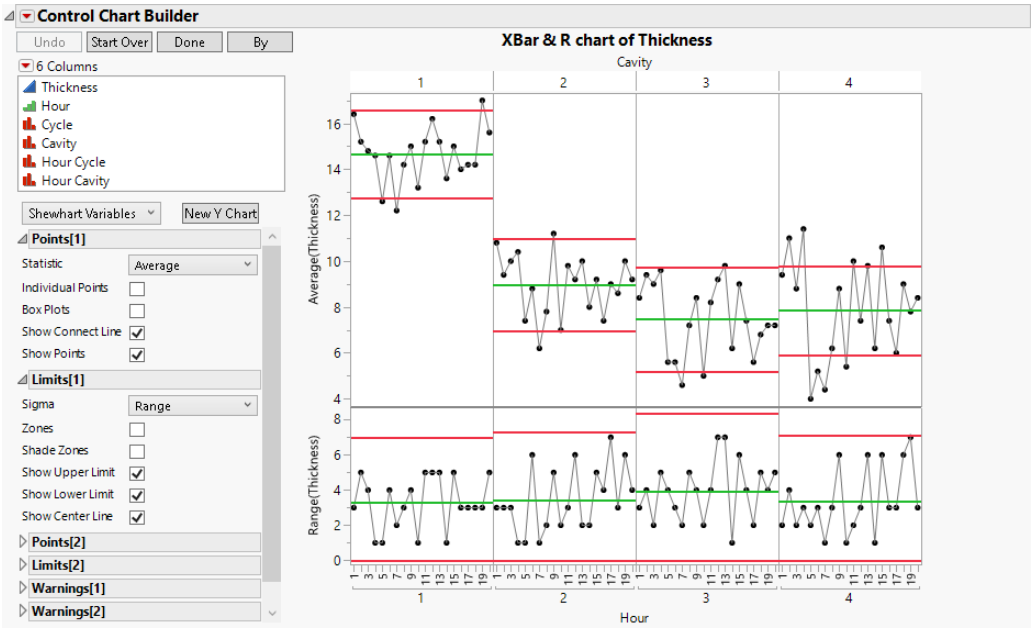
Control Chart Builder

Create Control Charts Interactively

A control chart is a graphical and analytic tool for monitoring process variation. The natural variation in a process can be quantified using a set of control limits. Control limits help distinguish common-cause variation from special-cause variation. Typically, action is taken to identify and eliminate special-cause variation. It is also important to quantify the common-cause variation in a process, as this determines the capability of a process.

Use Control Chart Builder to create control charts of your process data. Control Chart Builder can be launched as an interactive workspace or from specific control chart menu options. In the interactive workspace, you select the variables that you want to chart and drag them into zones. JMP automatically chooses an appropriate chart type based on the data. You can quickly create another type of chart, or change the current settings for an existing chart.

Figure 3.1 Control Chart Builder Example



Contents

Overview of Control Chart Builder . . . 32
Example of Control Chart Builder . . . 33
Control Chart Types . . . 35
Shewhart Control Charts for Variables . . . 35
Shewhart Control Charts for Attributes . . . 36
Rare Event Control Charts . . . 37
Control Chart Types . . . 38
Launch Control Chart Builder . . . 41
Control Chart Builder Interactive Workspace . . . 42
Launch Windows for Specific Control Charts . . . 44
Control Chart Builder Window . . . 46
Control Chart Builder Options . . . 47
Control Chart Builder Red Triangle Menu Options . . . 48
Options Panel and Right-Click Chart Options . . . 51
Control Chart Builder Right-Click Axis Options . . . 61
Work with Control Limits . . . 62
Example of Control Limits . . . 62
Excluded and Hidden Samples in Control Chart Builder . . . 70
Additional Examples of Control Chart Builder . . . 71
Individual Measurement and Moving Range Charts Example . . . 72
XBar and R Chart Phase Example . . . 73
XBar and S Charts with Varying Subgroup Sizes Example . . . 76
Run Chart Example . . . 78
P chart Example . . . 79
NP chart Example . . . 80
C chart Example . . . 82
U chart Example . . . 83
G chart Example . . . 84
T chart Example . . . 85
Three Way Control Chart Example . . . 87
Example of Multiple Control Charts . . . 89
Statistical Details for Control Chart Builder . . . 89
Control Limits for XBar and R Charts . . . 90
Control Limits for XBar and S Charts . . . 91
Control Limits for Individual Measurement and Moving Range Charts . . . 92
Control Limits for P and NP Charts . . . 94
Control Limits for U Charts and C Charts . . . 94
Levey-Jennings Charts . . . 95
Control Limits for G Charts . . . 95

Control Limits for T Charts 96
Sigma Calculations for Three Way Control Charts 97

Overview of Control Chart Builder

A control chart is a graphical and analytic tool for monitoring process variation and identifying special-cause variation in a process. Establishing control limits to filter out routine variation helps determine whether a process is stable and predictable. If the variation in a process is more than desired, the process can be adjusted to create higher quality output with potential cost savings.

All processes exhibit measurement variation as the process is monitored over time. There are two types of variation in process measurements:

- *Routine* or *common-cause* variation. Measurements from a stable process still exhibit random variation. When process measurements exhibit only common-cause variation, the measurements stay within expected limits.
- *Abnormal* or *special-cause* variation. Special-cause variation is indicated by patterns observed on the control chart. Examples include a shift in the process mean, points above or below the control limits, or measurements that trend up or down. These shifts in the process measurements can be caused by factors such as a broken tool or machine, equipment degradation, or changes to raw materials. A change or defect in the process is often identifiable by abnormal variation in the process measurements.

Control Chart Builder enables you to create several types of control charts including Shewhart and Rare Event control charts. Shewhart control charts are broadly classified into control charts for variables and control charts for attributes. Rare event charts are designed for events that occur infrequently. JMP provides a flexible, user-defined approach to building control charts. You can construct control charts in the following ways:

- Use the interactive Control Chart Builder workspace. When you drag a data column to the workspace, Control Chart Builder creates an appropriate chart based on the data type and sample size.
- Use the control chart menu options to build a specific control chart using a launch window.

Once an initial chart is created through either method above, use the menus and other options to change the type of chart, change the statistic on the chart, reformat the chart, or add additional charts.

Example of Control Chart Builder

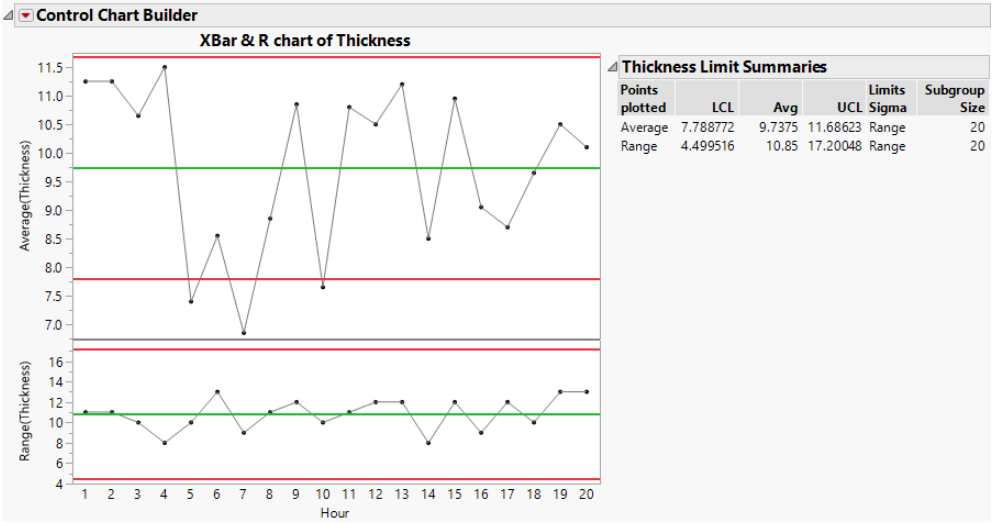
The Socket Thickness.jmp sample data table contains measurements for the thickness of sockets. There has been an increase in the number of defects during production and you want to investigate why this is occurring. This example illustrates how to perform this investigation in Control Chart Builder using either the interactive approach or the launch window approach. The second approach is convenient if you know which type of control chart you want to build.

Control Chart Builder Interactive Method

Use the interactive Control Chart Builder workspace to investigate the variability in the process data.

1. Select **Help > Sample Data Library** and open Quality Control/Socket Thickness.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Thickness to the **Y** zone.
4. Drag Hour to the **Subgroup** zone (at bottom).

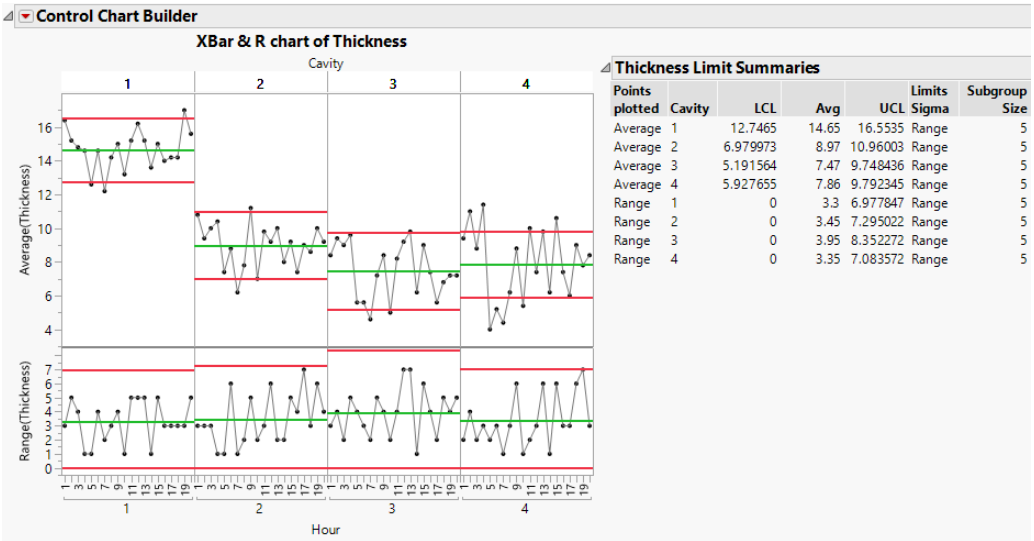
Figure 3.2 Control Charts for Socket Thickness



Looking at the Average chart, you can see that there are several points below the lower control limit of 7.788772. You want to see whether another variable might be contributing to the problem.

5. Drag Cavity into the **Phase** zone.
6. Click **Done**.

Figure 3.3 Control Charts for Each Cavity



From the Average chart, you can conclude the following:

- There are differences between the cavities, indicating the need for separate control limits for each cavity.
- Cavity 1 is producing sockets with an average thickness above that of the other cavities. This indicates that further investigation of the differences between cavities is warranted.
- All of the cavities have points that are outside the control limits. Therefore, you should investigate the lack of control in the process for each cavity.

The Range chart for each cavity shows that the within-subgroup measurements are in control and are similar across cavities.

Control Chart Builder Launch Window Method

Use the XBar Control Chart launch window to obtain the same chart as Figure 3.3.

1. Select **Help > Sample Data Library** and open Quality Control/Socket Thickness.jmp.
2. Select **Analyze > Quality and Process > Control Chart > XBar Control Chart**.
3. Select Thickness and click **Y**.
4. Select Hour and click **Subgroup**.
5. Select Cavity and click **Phase**.
6. Click **OK**.

   You should see the same control chart that appears in Figure 3.3.

Control Chart Types

Control Chart Builder enables you to create several types of control charts, including Shewhart Variable, Shewhart Attribute, and Rare Event charts.

- "Shewhart Control Charts for Variables"
- "Shewhart Control Charts for Attributes"
- "Rare Event Control Charts"
- "Control Chart Types"

Shewhart Control Charts for Variables

Control charts for variables are classified according to the subgroup summary statistic plotted on the chart.

- XBar charts are a type of location chart that display subgroup means (averages).
- R charts are a type of dispersion chart that display subgroup ranges (maximum – minimum).
- S charts are a type of dispersion chart that display subgroup standard deviations.
- Individual Measurement charts are a type of location chart that display individual measurements.
- Run charts are a type of location chart that display individual measurements as a connected series of points.
- Presummarize charts display both subgroup means and standard deviations.
- Moving Range charts are a type of dispersion chart that display moving ranges of two successive measurements.

Note: If you remove a dispersion chart or turn off the preference Show Two Shewhart Charts in File > Preferences > Platforms > Control Chart Builder, you will see only the location chart. Any associated scripts will contain the JSL option Show Two Shewhart Charts set to off (0).

XBar, R, and S Charts

For quality characteristics measured on a continuous scale, a typical analysis shows both the process mean and its variability with a mean chart aligned above its corresponding R or S chart.

Individual Measurement Charts

Individual Measurement charts displays individual measurements. Individual Measurement charts are appropriate when only one measurement is available for each sampling time point. If you are charting individual measurements, the individual measurement chart shows above its corresponding moving range chart. Moving Range charts displays moving ranges of two successive measurements.

Presummarize Charts

If your data consist of repeated measurements of the same process unit, you can combine these into one measurement for the unit. Pre-summarizing is not recommended unless the data contain repeated measurements on each process or measurement unit.

Presummarize summarizes the process column into sample means and/or standard deviations, based either on the sample size or sample label chosen. Then it charts the summarized data based on the options chosen in the window.

Levey-Jennings Charts

Levey-Jennings charts show a process mean with control limits based on a long-term sigma. The control limits are placed at 3s distance from the center line. The standard deviation, s, for the Levey-Jennings chart is calculated the same way standard deviation is in the Distribution platform.

Shewhart Control Charts for Attributes

Attributes charts are applicable for count data. Attribute charts are based on binomial and Poisson models. Because the counts are measured per subgroup, it is important when comparing multiple charts to determine whether you have a similar number of items in the subgroups between the charts. Attribute charts, like variables charts, are classified according to the subgroup sample statistic plotted on the chart.

Table 3.1 Attribute Chart Determination

Distribution Used to Calculate Sigma	Statistic Type: Proportion	Statistic Type: Count
Binomial	P chart	NP chart
Poisson	U chart	C chart

Control Chart Builder makes some decisions for you based on the variable selected. Once the basic chart is created, you can use the menus and other options to change the type, the statistic, and the format of the chart.

- P charts display the proportion of nonconforming (defective) items in subgroup samples, which can vary in size. Because each subgroup for a P chart consists of N_i items, and an item is judged as either conforming or nonconforming, the maximum number of nonconforming items in a subgroup is N_i.
- NP charts display the number of nonconforming (defective) items in subgroup samples. Because each subgroup for an NP chart consists of N_i items, and an item is judged as either conforming or nonconforming, the maximum number of nonconforming items in subgroup i is N_i.
- C charts display the number of nonconformities (defects) in a subgroup sample that usually, but does not necessarily, consists of one inspection unit.
- U charts display the number of nonconformities (defects) per unit in subgroup samples that can have a varying number of inspection units.

Rare Event Control Charts

A Rare Event chart is a control chart that provides information about a process where the data comes from rarely occurring events. Tracking processes that occur infrequently on a traditional control chart tend to be ineffective. Rare event charts were developed in response to the limitations of control charts in rare event scenarios. Control Chart Builder provides two types of rare event charts (G charts and T charts). The difference between a G chart and a T chart is the quantity used to measure distance between rare events. The G chart measures counts of events between incidents, whereas the T chart measures time intervals between incidents.

Table 3.2 Rare Event Chart Determination Based on Sigma

Distribution Used to Calculate Sigma	Chart Type
Negative Binomial	G chart
Weibull	T chart

G charts

A G chart measures the number of events between rarely occurring errors or nonconforming incidents, and creates a chart of a process over time. Each point on the chart represents the number of units between occurrences of a relatively rare event. For example, in a production setting, where an item is produced daily, an unexpected line shutdown can occur. You can use a G chart to look at the number of units produced between line shutdowns.

When reading a G chart, the points above the upper control limit indicate that the number of events between errors has increased. If the number of events between rarely occurring errors or nonconforming incidents has increased, that is good. Therefore, a point flagged as out of control above the limits is generally considered a desirable effect when working with G charts.

T charts

A T chart measures the time intervals elapsed since the last event. Each point on the chart represents a number of time intervals that have passed since a prior occurrence of a rare event. A T chart can be used for numeric, nonnegative data, date/time data, and time-between data. Since a traditional plot of these data might contain many points at zero and an occasional point at one, using a T chart avoids flagging numerous points as out of control. The data points for a T chart in Control Chart Builder are restricted to integer values.

When reading a T chart, the points above the upper control limit indicate that the amount of time between events has increased. This means that the rate of adverse events has decreased. Therefore, a point flagged as out of control above the limits is generally considered a desirable effect when working with T charts.

Control Chart Types

The most common control charts are available in Control Chart Builder and in the platforms in the Analyze > Quality and Process > Control Chart menu. Use Control Chart Builder as your first choice to easily and quickly generate charts. JMP automatically chooses the appropriate chart type based on the data. Table 3.3 through Table 3.7 summarize the different control chart types.

Table 3.3 Variable Charts Without Grouping (X) Variable or Nonsummarized Data

Chart Types	Control Chart Builder Options	
	Points > Statistic	Limits > Sigma
Individual	Individual	Moving Range
Moving Range on Individual	Moving Range	Moving Range

Table 3.3 Variable Charts Without Grouping (X) Variable or Nonsummarized

Chart Types	Control Chart Builder Options	
	Points > Statistic	**Limits > Sigma**
Individual (limits computed on median moving range)	Individual	Median Moving Range
Median Moving Range on Individual	Moving Range	Median Moving Range
Levey Jennings	Individual	Levey Jennings

Table 3.4 Variable Charts with Grouping (X) Variables or Summarized Data

Chart Types	Control Chart Builder Options	
	Points > Statistic	**Limits > Sigma**
XBar (limits computed on range)	Average	Range
XBar (limits computed on standard deviation)	Average	Standard Deviation
R	Range	Range
S	Standard Deviation	Standard Deviation
Levey Jennings	Average	Levey Jennings or overall Standard Deviation

Table 3.5 Presummarize Charts

Chart Types	Control Chart Builder Options	
	Points > Statistic	**Limits > Sigma**
Individual on Group Means	Average	Moving Range
Individual on Group Means (limits computed on median moving range)	Average	Median Moving Range
Individual on Group Std Devs	Standard Deviation	Moving Range

Table 3.5 Presummarize Charts *(Continued)*

Chart Types	Control Chart Builder Options	
	Points > Statistic	**Limits > Sigma**
Individual on Group Std Devs (limits computed on median moving range)	Standard Deviation	Median Moving Range
Moving Range on Group Means	Moving Range on Means	Moving Range
Median Moving Range on Group Means	Moving Range on Mean	Median Moving Range
Moving Range on Group Std Devs	Moving Range on Std Dev	Moving Range
Median Moving Range on Group Std Devs	Moving Range on Std Dev	Median Moving Range

Table 3.6 Attribute Charts

Chart Types	Control Chart Builder Options	
	Points > Statistic	**Limits > Sigma**
P chart	Proportion	Binomial
NP chart	Count	Binomial
C chart	Count	Poisson
U chart	Proportion	Poisson

Table 3.7 Rare Event Charts

Chart Types	Control Chart Builder Options	
	Points > Statistic	**Limits > Sigma**
G chart	Count	Negative Binomial
T chart	Count	Weibull

Launch Control Chart Builder

You can launch Control Chart Builder in the following two ways:

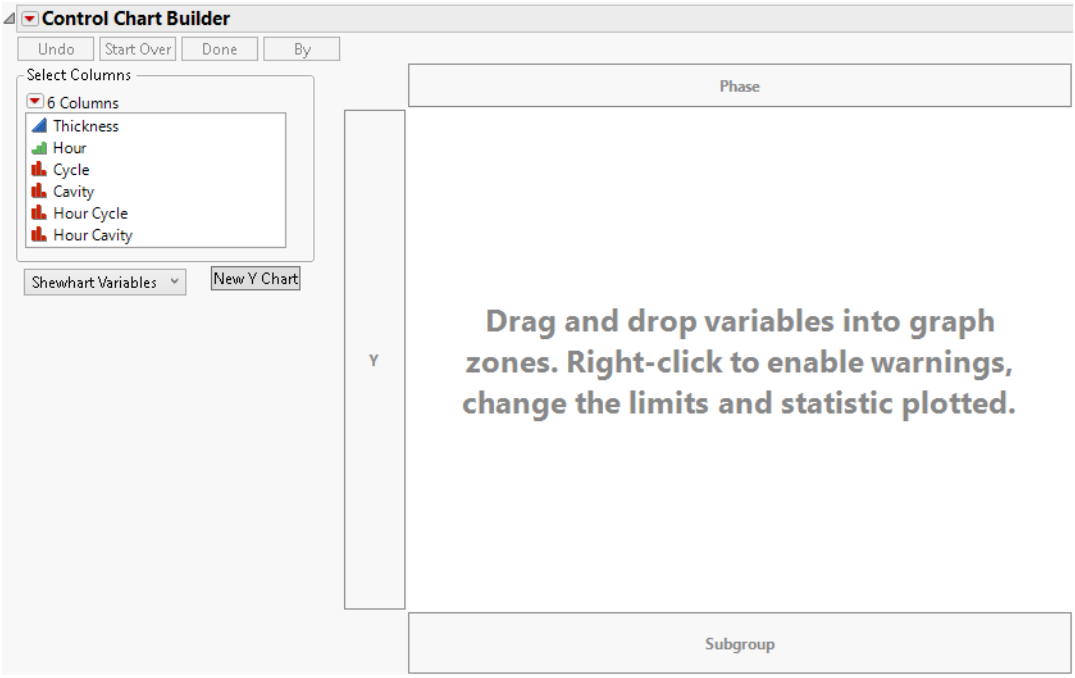
- If you are not sure what type of control chart is appropriate for your data, select **Analyze > Quality and Process > Control Chart Builder**. This method enables you to drag data columns to the workspace and Control Chart Builder creates an appropriate chart based on the data type and sample size. See "Control Chart Builder Interactive Workspace".
- If you know which type of control chart is appropriate for your data, select the appropriate chart from the **Analyze > Quality and Process > Control Chart** submenu. This displays a control chart launch window. See "Launch Windows for Specific Control Charts". There are control chart builder launch windows for the following control charts:
  - I/MR Control Chart
  - XBar Control Chart
  - Run Chart
  - P Control Chart
  - NP Control Chart
  - C Control Chart
  - U Control Chart
  - Levey Jennings Control Chart
  - I/MR on Means Control Chart
  - Three Way Control Chart

  Note: The CUSUM Control Chart, EWMA, and Multivariate Control Charts launch in their own platforms instead of launching in Control Chart Builder, and are documented separately. See "CUSUM Control Charts", "Multivariate Control Charts", and "EWMA Control Charts".

Once you click OK in a launch window, the Control Chart Builder window appears with the Control Panel hidden by default. All other options and features are the same.

Control Chart Builder Interactive Workspace

Figure 3.4 Interactive Control Chart Builder Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

To begin creating a control chart, drag variables from the **Select Columns** box into the zones. If you drop variables in the center, JMP guesses where to put them based on whether the variables are continuous or categorical. The Control Chart Builder workspace contains the following zones:

Y Assigns the process variable.

Subgroup Assigns subgroup variables. To define subgroup levels as a combination of multiple columns, add multiple variables to the **Subgroup** zone. When a subgroup variable is assigned, each point on the control chart corresponds to a summary statistic for all of the points in the subgroup.

Phase Assigns phase variables. When a Phase variable is assigned, separate control limits are computed for each phase. See also "XBar and R Chart Phase Example".

The initial Control Chart Builder window contains the following buttons:

Undo Reverses the last change made to the window.

Start Over Returns the window to the default condition, removing all data, and clearing all zones.

Done Hides the buttons and the Select Columns box and removes all drop zone outlines. In this presentation-friendly format, you can copy the graph to other programs. To restore the window to the interactive mode, click the Control Chart Builder red triangle and select **Show Control Panel**.

By Identifies the variable and produces a separate analysis for each value that appears in the column.

Shewhart Variables/Shewhart Attribute/Rare Event Enables you to select Shewhart Variables, Shewhart Attribute, or Rare Event control chart types. If you select an Attribute chart type, an n Trials box and zone appear on the chart.

n Trials (Available for Attribute charts.) Assigns a lot size for an attribute control chart.

New Y Chart Produces a copy of the current chart for every column selected in the Select Columns box. The new charts use the selected columns in the Y role.

Once you drag variables to the chart, other buttons and options appear at left that enable you to show, hide or switch items on the chart (Figure 3.7). Many of these functions (Points, Limits, Warnings, etc.) are the same as the functions available when you right-click the chart. See "Options Panel and Right-Click Chart Options". For information about warnings and rules, see "Tests" and "Westgard Rules".

Three Way Control Chart Enables you to produce a three way control chart for variable chart types. The subgroup size must be greater than one. The plotting statistic is based on subgroup averages, within-subgroup variation, or between-subgroup variation. The default set of three includes a presummarized chart of the averages using Moving Range limits, a Moving Range chart and a Range chart.

Event Chooser Allows the chart to respond in real time to selection changes. There are several standard groups of responses that are recognized and pre-scored (for example, pass/fail, yes/no, Likert Scales, conforming/non-conforming, and defective/non-defective). If you are analyzing results from a survey and want to focus solely on a specific sector of the results for one or more questions, you can make the selection on the screen. When you make the selection, the chart is scored again and replotted immediately. The levels selected in the Event Chooser are counted as events, and all other levels are counted as non-events.

The Event Chooser is available for attribute charts with response columns that have a modeling type of nominal or ordinal. If you want the Event Chooser to work on a numeric integer-valued nominal or ordinal response column, you must select the Use Event Chooser option from the Control Chart Builder red triangle menu. The Event Chooser does not appear for response columns with a modeling type of continuous.

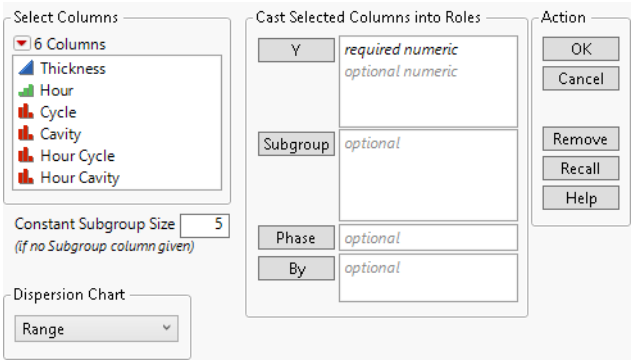
Launch Windows for Specific Control Charts

The options that you see in the launch windows vary depending on whether you are launching variable control charts or attribute control charts.

Launch Windows for Variable Control Charts

This section contains information about the launch windows for I/MR, XBar, Run, Levey Jennings, I/MR on Means, and Three Way Control Charts.

Figure 3.5 Launch Window for Variable Control Charts



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y Assigns the process variables.

Subgroup (Available only for XBar, Levey Jennings, I/MR on Means, and Three Way Control Charts.) Assigns the subgroup variables. When a subgroup variable is assigned, each point on the control chart corresponds to a summary statistic for all of the points in the subgroup.

Sample Label (Available only for I/MR Charts.) Assigns a sample label variable to uniquely label individual measurements.

Phase (Not available for Run Charts.) Assigns the phase variable. When a Phase variable is assigned, separate control limits are computed for each phase.

By Identifies a variable to produce a separate analysis for each value that appears in the column.

Sort by Sample Label Sorts all variables by the sample label.

Constant Subgroup Size (Available only for XBar charts.) Specifies a number of observations that make up each subgroup. If the number of observations per subgroup is not constant, you should specify a column in the Subgroup role.

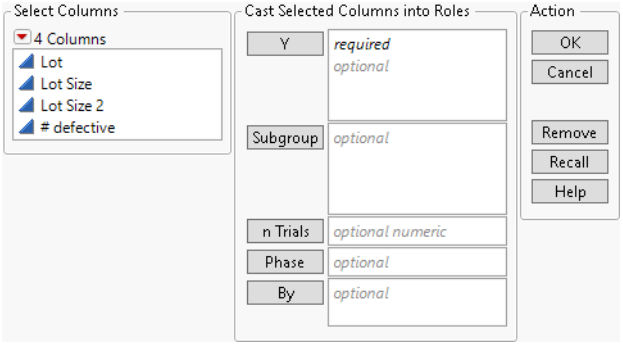
Note: If you specify a Subgroup variable, the platform ignores the setting for the Constant Subgroup Size option.

Dispersion Chart (Available only for XBar charts.) Specifies the type of dispersion chart that appears below the XBar chart. You can choose between no dispersion chart, a range chart (XBar & R), or a standard deviation chart (XBar & S).

Launch Windows for Attribute Control Charts

This section contains information about the launch windows for NP, P, C, and U Control Charts.

Figure 3.6 Launch Window for Attribute Control Charts



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y Assigns the process variables.

Subgroup Assigns the subgroup variables. When a subgroup variable is assigned, each point on the control chart corresponds to a summary statistic for all of the points in the subgroup.

n Trials Assigns the subgroup sample size.

Phase Assigns the phase variable. When a Phase variable is assigned, separate control limits are computed for each phase.

By Identifies a variable to produce a separate analysis for each value that appears in the column.

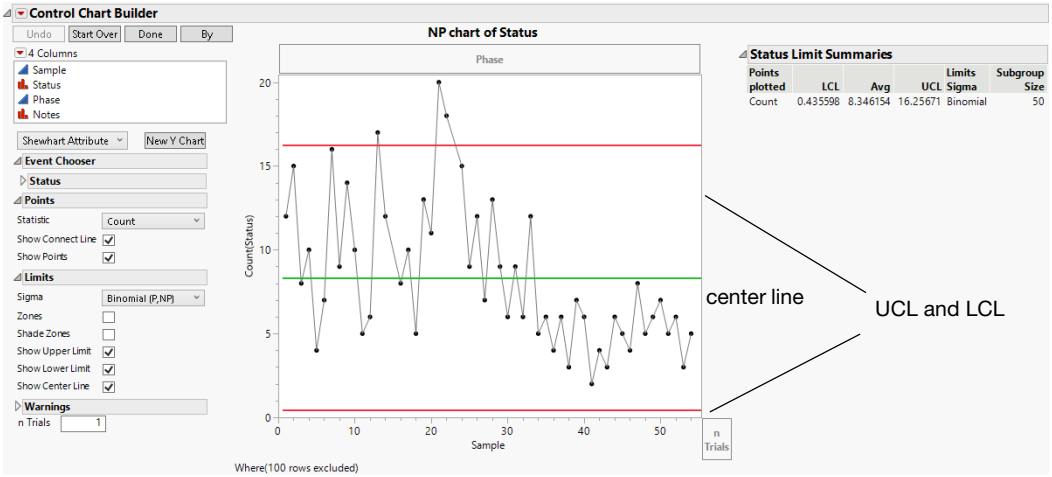
Control Chart Builder Window

Use Control Chart Builder to construct control charts for process data. The analysis produces a chart that can be used to evaluate whether a process is in a state of statistical control. The report varies depending on which type of chart you select. Control charts update dynamically as data is added or changed in the data table. Figure 3.7 displays the Control Chart Builder window for the Bottle Tops.jmp sample data table.

To create the chart:

1. Select **Help > Sample Data Library** and open Quality Control/Bottle Tops.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Status to the **Y** zone.
4. Drag Sample to the **Subgroup** zone (at bottom).

Figure 3.7 Control Chart Builder Window



You can drag other variables into the various zones to augment the analysis and use the "Control Chart Builder Options" to further examine the data. Some of the right-click chart options (for example, show or hide points, limits, warnings, and zones; select statistic and sigma options) also appear on the left hand side of the chart for easy access.

Control charts have the following characteristics:

- Each point plotted on the chart represents an individual process measurement or summary statistic. Subgroups should be chosen *rationally*, that is, they should be chosen to maximize the probability of seeing a true process signal *between* subgroups. Often, this

requires knowledge of the process to determine the most effective grouping strategy. See Wheeler (2004); Woodall and Adams (1998).

- The vertical axis of a control chart is scaled in the same units as the summary statistic.
- The horizontal axis of a control chart identifies the subgroup samples and is time ordered. Observing the process over time is important in assessing if the process is changing.

The green line is the center line, or the average of the data. The center line indicates the average (expected) value of the summary statistic when the process is in statistical control. Measurements should appear equally on both sides of the center line. If not, this is possible evidence that the process average is changing.

- The two red lines are the upper and lower control limits, labeled UCL and LCL. These limits give the range of variation to be expected in the summary statistic when the process is in statistical control. If the process is exhibiting only routine variation, then all the points should fall randomly in that range.

  Note: To hide the lower control limits on dispersion, attribute, and rare event charts, deselect the Show Lower Limit option in the options panel. To change the default to always hide the lower control limits, deselect the Show Lower Control Limit preference in File > Preferences > Platforms > Control Chart Builder.

- A point outside the control limits signals the presence of a special cause of variation.

Options in the Control Chart Builder window create control charts that can be updated dynamically as samples are received and recorded or added to the data table. When a control chart signals abnormal variation, action should be taken to return the process to a state of statistical control if the process degraded. If the abnormal variation indicates an improvement in the process, the causes of the variation should be studied and implemented.

When you double-click the axes, the appropriate Axis Specification window appears for you to specify the format, axis values, number of ticks, gridline, reference lines, and other options to display.

Control Chart Builder Options

Control Chart Builder options appear in the red triangle menu or by right-clicking on a chart or axis. Some of the right-click chart options also appear on the bottom left hand side of the chart for easy access. You can also set preferences for many of the options in Control Chart Builder at File > Preferences > Platforms > Control Chart Builder.

- "Control Chart Builder Red Triangle Menu Options"
- "Options Panel and Right-Click Chart Options"
- "Control Chart Builder Right-Click Axis Options"

Control Chart Builder Red Triangle Menu Options

The Control Chart Builder red triangle menu contains the following options:

Show Control Panel Shows or hides the following elements:

- buttons
- the Select Columns box
- the drop zone borders
- check boxes and drop-down menus

Show Limit Summaries Shows or hides the Limit Summaries report. This report shows the control limits (LCL and UCL), the center line (Avg), the Points and Limits plotted, and the Sample Size for the chart. Sample size is not shown for rare event charts.

Show Capability (Available only for Shewhart Variables charts that have specification limits.) Shows or hides the Process Capability Analysis report. Since the report is part of the Limit Summaries report, the Process Capability report appears only when the Show Limit Summaries option is selected. For more information, see "The Process Capability Report". You can set preferences for many of the options in the Process Capability report in Control Chart Builder at File > Preferences > Platforms > Process Capability.

Note: Show Capability is not available if the response variable has no variation.

Show Alarm Report Shows or hides a report that contains information about out of control samples. The report reflects failures for currently enabled tests in each chart and updates automatically as different tests are enabled and disabled and as data and row states change. A second table lists the currently enabled tests for each chart. The first table contains the following columns:

Position Indicates the numerical position of the chart, starting from the top of the report.

Total Samples Out of Control Counts the number of samples that failed at least one of the selected tests.

Alarm Rate The total number of samples out of control divided by the total number of nonmissing samples. This is also known as the Proportion Out of Control.

Note: The counts that contribute to the calculation of the alarm rate include excluded samples only if the Test Excluded Subgroups and the Show Excluded Region options are both selected.

Show Limit Labels Shows or hides labels for the limits in each chart. The limits are shown inside the right frame of the chart.

Show Sigma Report (Available only for Shewhart Variables charts.) Shows or hides the Process Sigma Report, which is a table of sigma values. The Process Sigma Report contains the overall sample size, number of subgroups, sample mean, overall sigma, within sigma, and stability index. For three way control charts, the between-sigma and between-and-within sigma values are also shown. If a phase variable is specified, a set of values is given for each phase.

Note: The Process Sigma Report appears only if the Limit Summaries report is turned on.

Get Limits Retrieves the control limits that are stored in an open or saved data table.

Show Excluded Region Shows (on) or removes (off) the regions of the chart where samples have been excluded. When entirely excluded subgroups are shown on the location chart, they appear as dimmed points to indicate that they are excluded.

Caution: The Show Excluded Region option impacts the chart. Excluded samples are removed from the calculation of control limits, whether this option is on or off. Excluded samples are included in alarm rate calculations only if the Test Excluded Subgroups and the Show Excluded Region options are both selected.

Set Subgroup Size (Not available if a subgroup variable is specified.) Sets a subgroup size. Missing values are taken into account when computing limits and sigma.

Note: If the Set Subgroup Size option is used, the Show Excluded Region option is turned on automatically.

Save Limits Saves the control limits in one of the following ways:

in Column Saves control limits as a column property in the existing data table for the response variable. If the limits are constant, LCL, Avg, and UCL values for each chart type in the report are saved. This option is not available with phase charts. In addition, the option has no effect if the sample sizes are not constant for each chart.

in New Table Saves the standard deviation and mean for each chart into a new data table. If the limits are constant, the LCL, Avg, UCL, and Sample Size for each chart are saved as well. If there are phases, a new set of values is saved for each phase. There is a row for each statistic and a column for each Y variable.

in New Tall Table (Not available for Rare Event, Attribute, or Phase charts.) Saves the standard deviation, mean, and Sigma for each chart into a new data table. If the limits are constant, the LCL, Avg, UCL, and Sample Size for each chart are saved as well. There is a row for each Y variable and a column for each statistic. A column for Sigma that can be used in the Process Screening platform is also saved.

Save Summaries Creates a new data table containing such information as the sample label, sample sizes, statistic being plotted, center line, control limits, and any tests, warnings and failures. The specific statistics included in the table depend on the type of chart.

Graph Spacing Sets the amount of space between the graphs.

Include Missing Categories Enables the graph to collect rows with missing values in a categorical column, and displays the missing values on the graph as a separate category. If this option is disabled, all rows with a missing X value are removed from the calculations, in addition to being hidden from the graph.

This option is not available for continuous X variables or categorical Y variables because there is no compelling way to display the collected missing values on the relevant axes. By default, this option is enabled.

Note: If Include Missing Categories is enabled, capability analysis results in Control Chart Builder do not match those in the Process Capability platform if a categorical X variable has missing values.

Use Event Chooser (Available only for attribute charts with numeric non-continuous Y variables.) Categorizes ordinal numeric data and offers individual numeric-level modeling selections.

Alarm Script Enables you to write and run a script that indicates when the data fail special causes tests. See "Tests". Results can be written to a file, written to the log or sent in an email. There is an option to include an explanation of why the test failed.

As an Alarm Script is invoked, the following variables are available, both in the issued script and in subsequent JSL scripts:

`qc_col` is the name of the column

`qc_test` is the test that failed

`qc_sample` is the sample number

`qc_phase` is the label of the phase during which the failure occurred

See the *Scripting Guide* for more information about writing custom Alarm Scripts.

Note: Alarm scripts are not available in reports that use the Local Data Filter.

Sort by Row Order Sorts all subgroup and phase variables in the order in which the levels appear in the data table. This applies to all combinations of nested subgroup and phase variables.

Test Excluded Subgroups (Available only if the Show Excluded Region option is selected.) Includes (on) or excludes (off) entirely excluded subgroups in the computation of tests.

When excluded subgroups are shown and the Text Excluded Subgroups option is not selected, the excluded subgroups are treated as missing values.

Note: For any test that relies on consecutive points (runs tests), an entirely excluded subgroup is treated as missing and counts of consecutive points are restarted.

Control Chart Dialog (Available only if the control chart is launched through a Control Chart launch window.) Opens the Control Chart launch window with the original settings that were used to create the control chart.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

- Column Switcher is available only for a single Y variable having two or fewer associated charts. Based on the selected chart type, only columns that are appropriate for the Y role are included in the Column Switcher column list.
- In Control Chart Builder, the Automatic Recalc option is turned on by default and cannot be turned off.
- When using the local data filter, you can deselect the Show Excluded Region option for more focused exploration.

Options Panel and Right-Click Chart Options

The following options appear on the left hand side of the Control Chart Builder report or when you right-click a chart. Some options appear in both locations.

Event Chooser (Available only for attribute charts with response columns that have a modeling type of nominal or ordinal.) Allows the chart to respond in real time to selection changes. See "Event Chooser".

Points Provides the following options:

Statistic Changes the statistic plotted on the chart. See "Statistic".

Individual Points Show or hides individual observations in a subgroup. Available only with a subgroup variable or Set Sample Size. This option is not available for Attribute chart types or Rare Event charts.

Box Plots (Available only for Shewhart Variables charts.) Shows or hides box plots on the chart.

Show Connect Line Shows connecting lines between the points.

Show Points Shows or hides the points on the chart.

Limits Provides the following options:

Sigma Specifies the method of computing sigma. See "Sigma".

Zones (Available only for Variables and Attribute chart types.) Shows or hides the zones on the chart. There are three equal-width zones on either side of the mean. Zones are not drawn below the LCL or above the UCL. If the limits for a Variables chart are not centered around the mean, min(Avg-LCL,UCL-Avg)/3 is used as the width of each zone. The zones for an Attributes chart use a width of (UCL-Avg)/3.

Shade Zones Shows or hides shading zones by ranges. Zone C is shaded green, zones A and B are shaded yellow, and beyond zone A is shaded red.

Spec Limits (Available only if the data table has a Spec Limits column property or if you specify Spec Limits using the Add Spec Limits option.) Shows or hides the specification limits on the chart. By default, the spec limits are shown if the Spec Limits column property has the Show as Graph Reference Lines option selected. See *Using JMP* for information about adding a Spec Limits column property.

Add Spec Limits (Available in the right-click menu.) Enables you to enter specification limits.

Set Control Limits (Available in the right-click menu.) Enables you to enter control limits for tests. After you click OK in the Set Control Limits window, the specified control limits are set uniformly across groups. Select this option again to remove the specified control limits.

Add Limits (Available in the right-click menu.) Specifies additional control limits to be plotted on the chart. These limits are not used in tests.

Show Upper Limit Shows or hides the upper control limit on the chart. If you hide the upper control limit on a chart, the Test Beyond Limits and Test 1 options do not flag points associated with the hidden upper control limit.

Show Lower Limit Shows or hides the lower control limit on the chart. If you hide the lower control limit on a chart, the Test Beyond Limits and Test 1 options do not flag points associated with the hidden lower control limit.

Show Center Line Shows or hides the center line on the chart.

Add Dispersion Chart (Available in the right-click menu.) Adds a dispersion chart to the chart area. Change the chart type with the Points options. A dispersion chart illustrates the variation in the data by plotting one of many forms of dispersion, including the range, standard deviation, or moving range. Available only for Variables chart types.

Note: You can customize the default dispersion chart type using the Dispersion Chart and Summarized Dispersion Chart preferences in File > Preferences > Platforms > Control Chart Builder.

Set Subgroup Size Sets a subgroup size. Missing values are taken into account when computing limits and sigma.

Warnings Provides the following options:

Customize Tests (Available in the right-click menu.) Enables you to design custom tests and select or deselect multiple tests at once. After the option is selected, the Customize Tests window appears for designing the tests. Select a test description, and enter the desired number (n) and label. You can save the settings to preferences and also restore the default settings. Available only for Variables and Attribute chart types.

Tests Enables you to select which statistical control tests to enable. For more information about tests, see "Tests". Available only for Variables and Attribute chart types.

Note: Hover over a flagged point on the chart to see a description of the test that failed.

Westgard Rules Specifies the set of Westgard statistical control tests that are enabled. Because Westgard rules are based on sigma and not the zones, they can be computed without regard to constant sample size. For more information about tests, see "Westgard Rules". Available only for Variables and Attribute chart types.

Test Beyond Limits (Called Test 15 in JMP) Enables the test for any points beyond the control limits. These points are identified on the chart. This test works on all charts with limits, regardless of the sample size being equal.

Note: If you hide the upper or lower control limits, the Test Beyond Limits option does not flag points that are beyond limits that are not shown on the control chart.

Remove Graph (Available in the right-click menu.) Removes the control chart.

Remove Location Chart (Available only if you right-click a location chart.) Removes the location chart from the report.

Remove Dispersion Chart (Available only if you right-click a dispersion chart.) Removes the dispersion chart from the report.

Note: For a description of the Rows, Graph, Customize, and Edit menus, see *Using JMP*.

Three Way Control Chart Enables you to produce a three way control chart for variable chart types. See "Three Way Control Chart".

Statistic

You can change the statistic represented by the points on the chart. The options available depend on the chart type selected.

For Variables chart types, you can change the statistic represented by the points on the chart using the following options:

Individual Creates a chart where each point represents an individual value in the data table.

Average Creates a chart where each point represents the average of the values in a subgroup.

Range Creates a chart where each point represents the range of the values in a subgroup.

Standard Deviation Creates a chart where each point represents the standard deviation of the values in a subgroup.

Moving Range on Means Computes the difference in the range between two consecutive subgroup means.

Moving Range on Std Dev Computes the difference in the range between two consecutive subgroup standard deviations.

Moving Range Creates a chart where each point is the difference between two consecutive observations.

Note: The Average, Range, Standard Deviation, Moving Range on Means, and Moving Range on Std Dev methods appear only if a subgroup variable with a sample size greater than one is specified or a sample size is set.

For Attribute chart types, you can change the statistic represented by the points on the chart using the following options:

Proportion Creates a chart where each point represents the proportion of items in subgroup samples.

Count Creates a chart where each point represents the number of items in subgroup samples.

For Rare Event chart types, the statistic represented by the points on the chart uses the Count option.

Sigma

You can change the method for computing sigma for the chart. The options available depend on the chart type selected.

For Variables chart types, you can use the following options:

Range Uses the range of the data in a subgroup to estimate sigma.

Standard Deviation Uses the standard deviation of the data in a subgroup to estimate sigma.

Moving Range Uses the moving ranges to estimate sigma. The moving range is the difference between two consecutive points.

Median Moving Range Uses the median moving range to estimate sigma, rather than the average moving range.

Levey-Jennings Uses the standard deviation of all the observations to estimate sigma. If your chart has phases, sigma is calculated for each phase separately.

For Attribute chart types, you can use the following options:

Binomial Uses the binomial distribution model to estimate sigma. The model indicates the number of successes in a sequence of experiments, where each experiment yields success with some probability. Selecting Binomial yields either a P or NP chart.

Poisson Uses the Poisson distribution model to estimate sigma. The model indicates the number of events and the time at which these events occur in a given time interval. Selecting Poisson yields either a C or U chart.

For Rare Event chart types, you can use the following options:

Negative Binomial Uses the negative binomial distribution model to estimate sigma. The model indicates the number of successes in a sequence of trials before a specified number of failures occur. Selecting Negative Binomial yields a G chart.

Weibull Uses the Weibull distribution model to estimate sigma. The model indicates the mean time between failures. Selecting Weibull yields a T chart.

Tests

The Warnings option in the pop-up menu or on the left hand side of the window displays a submenu for Tests selection. You can select one or more tests for special causes (Western Electric rules) from the menu. Nelson (1984) developed the numbering notation used to identify special tests on control charts. The tests work with both equal and unequal sample sizes.

If a selected test is positive for a particular sample, that point is labeled with the test number. When you select several tests for display and more than one test signals at a particular point, the label of the numerically lowest test specified appears beside the point. You can hover over a flagged point on the chart to see a description of the test that failed.

Tip: To add or remove several tests at once, select or deselect the tests in the Control Panel under **Warnings > Tests**.

Table 3.8 lists and interprets the eight tests, and Figure 3.9 illustrates the tests. The following rules apply to each test:

- The area between the upper and lower limits is divided into six zones, each with a width of one standard deviation.
- The zones are labeled A, B, C, C, B, A with zones C nearest the center line.
- A point lies in Zone B or beyond if it lies beyond the line separating zones C and B. That is, if it is more than one standard deviation from the center line.
- Any point lying on a line separating two zones lines is considered belonging to the innermost zone. So, if a point lies on the line between Zone A and Zone B, the point is considered to be in Zone B.
- When a Phase variable is specified, the counts for each test are reset at the start of each phase.

Notes:

- Tests 1 through 8 apply to all Shewhart Variables chart types.
- Tests 1, 2, 5, and 6 apply to the upper and lower halves of the chart separately.
- Tests 3, 4, 7, and 8 apply to the whole chart.
- Once a runs test (one that is based on consecutive observations) is triggered, the counts do not reset to 0 when moving to the next sample.
- Runs tests handle excluded rows based on the setting of the Show Excluded Region and Test Excluded Subgroups options.
  - By default, both options are selected, and the excluded rows are included in the runs tests calculations.
  - If the Show Excluded Region option is selected and the Test Excluded Subgroups option is not selected, the excluded rows are treated as missing and the counts for the runs tests reset to 0 when moving to the next sample.
  - If the Show Excluded Region option is not selected, the excluded rows are treated as if they are deleted.
- Tests 5 through 8 are not available for attribute charts.

See Nelson (1984, 1985) for further recommendations on how to use these tests.

Figure 3.8 Zones for Western Electric Rules

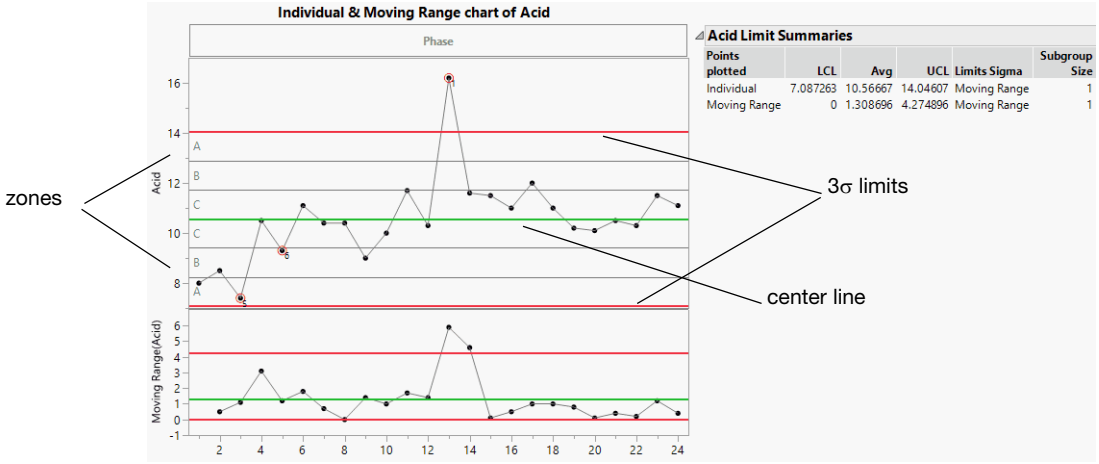


Table 3.8 Description and Interpretation of Tests for Special Causes[a]

Test 1	One point beyond Zone A (upper or lower)	Detects a shift in the mean, an increase in the standard deviation, or a single aberration in the process. For interpreting Test 1, any dispersion chart (R, S, or MR) can be used to rule out increases in variation. Note that if you hide the upper or lower control limits, the Test 1 option does not flag points that are associated with limits that are not shown on the control chart.
Test 2	Nine points in a row in a single (upper or lower) side of Zone C or beyond	Detects a shift in the process mean.
Test 3	Six points in a row steadily increasing or decreasing (anywhere on the chart)	Detects a trend or drift in the process mean.
Test 4	Fourteen points in a row alternating up and down (anywhere on the chart)	Detects systematic effects such as two alternately used machines, vendors, or operators.

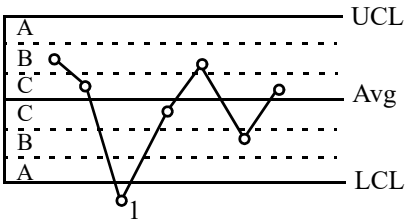
Table 3.8 Description and Interpretation of Tests for Special Causes[a] *(Continued)*

Test 5	Two out of three points in a row in or beyond Zone A and the point itself is in or beyond Zone A; the two points must be on the same side (upper or lower)	Detects a shift in the process average or increase in the standard deviation. Any two out of three points provide a positive test.
Test 6	Four out of five points in a row in or beyond Zone B and the point itself is in or beyond Zone B; the four points must be on the same side (upper or lower)	Detects a shift in the process mean. Any four out of five points provide a positive test.
Test 7	Fifteen points in a row in Zone C, above and below the center line	Detects stratification of subgroups when the observations in a single subgroup come from various sources with different means. Also detects a reduction in variation.
Test 8	Eight points in a row on both sides of the center line with none in Zones C	Detects stratification of subgroups when the observations in one subgroup come from a single source, but subgroups come from different sources with different means.

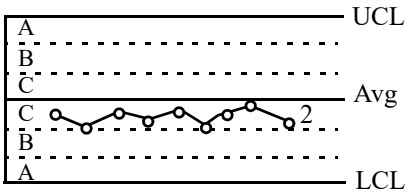
a. Nelson (1984, 1985)

Figure 3.9 Illustration of Special Causes Tests[1]

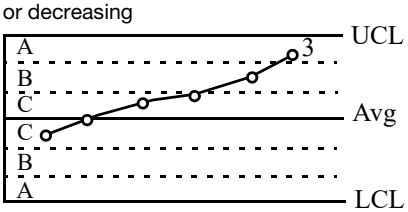
Test 1: One point beyond Zone A



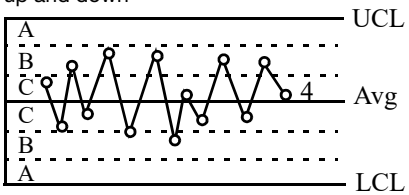
Test 2: Nine points in a row in a single (upper or lower) side of Zone C or beyond



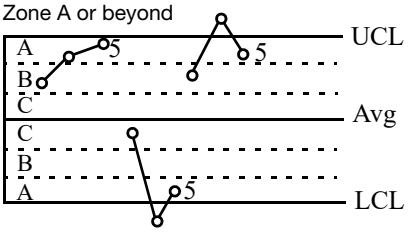
Test 3: Six points in a row steadily increasing or decreasing



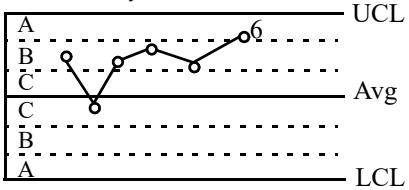
Test 4: Fourteen points in a row alternating up and down



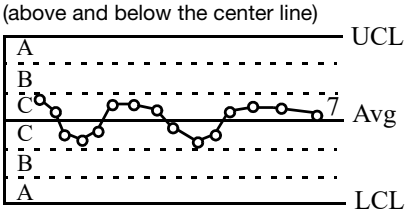
Test 5: Two out of three points in a row in Zone A or beyond



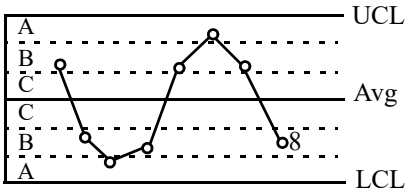
Test 6: Four out of five points in a row in Zone B or beyond



Test 7: Fifteen points in a row in Zone C (above and below the center line)



Test 8: Eight points in a row on both sides of the center line with none in Zone C



Westgard Rules

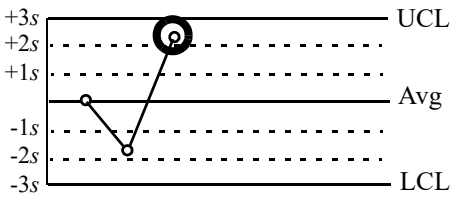
Westgard rules are implemented under the Westgard Rules submenu of the Warnings option when you right-click a chart or on the left hand side of the window. The different tests are abbreviated with the decision rule for the particular test. For example, **1 2s** refers to a test where one point is two standard deviations away from the mean.

1. Nelson 1984, 1985)

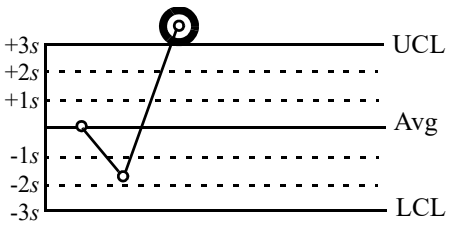
Notes:

- Once a runs test (one that is based on consecutive observations) is triggered, the counts do not reset to 0 when moving to the next sample.
- Runs tests handle excluded rows based on the setting of the Show Excluded Region and Test Excluded Subgroups options.
  - By default, both options are selected, and the excluded rows are included in the runs tests calculations.
  - If the Show Excluded Region option is selected and the Test Excluded Subgroups option is not selected, the excluded rows are treated as missing and the counts for the runs tests reset to 0 when moving to the next sample.
  - If the Show Excluded Region option is not selected, the excluded rows are treated as if they are deleted.

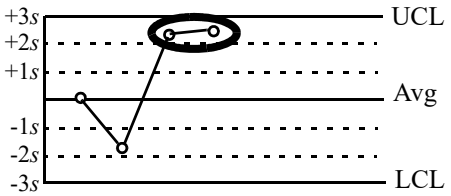
Rule 1 2S (called Test 9 in JMP) is commonly used with Levey-Jennings charts, where control limits are set 2 standard deviations away from the mean. The rule is triggered when any one point goes beyond these limits.



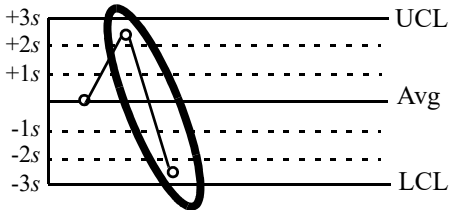
Rule 1 3S (called Test 10 in JMP) refers to a rule common to Levey-Jennings charts where the control limits are set 3 standard deviations away from the mean. The rule is triggered when any one point goes beyond these limits.



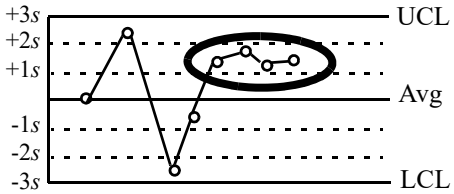
Rule 2 2S (called Test 11 in JMP) is triggered when two consecutive control measurements are farther than two standard deviations from the mean.



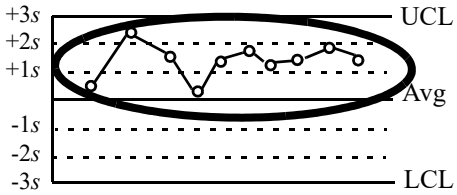
Rule R 4S (called Test 12 in JMP) is triggered when one measurement is greater than two standard deviations from the mean and the previous measurement is greater than two standard deviations from the mean in the opposite direction such that the difference is greater than 4 standard deviations.



Rule 4 1S (called Test 13 in JMP) is triggered when four consecutive measurements are more than one standard deviation from the mean.



Rule 10 X (called Test 14 in JMP) is triggered when ten consecutive points are on one side of the mean.



Control Chart Builder Right-Click Axis Options

Remove Graph Removes the entire graph.

Remove Removes a variable.

Note: If there is more than one chart type on the graph, a submenu listing the different charts is displayed. You can select which chart to remove.

For more information about the Axis Settings, Revert Axis, Add or Remove Axis Label, Save to Column Property, and Edit options, see *Using JMP*.

Work with Control Limits

Control limits are based on the performance of your process and tell you about the variability in your process. Upper control limits (UCLs), center lines, and lower control limits (LCLs) are calculated from the data when a control chart is created. You can use these calculated control limits to indicate when your process has changed.

It is important to note that control limits are different from specification limits, which are often used in capability analysis.

Table 3.9 Control Limits versus Specification Limits

Control Limits	Specification Limits
Calculated from data	Defined by the customer or design
Based on variability	Based on system requirements
The voice of the process	The voice of the customer

Example of Control Limits

In this example, consider a company's printing process. Variations can cause distortion in the line, including skew, thickness, and length problems. In this example, we will consider the length of the line. A line is considered good if it has a printed length of 16 cm +/- 0.2 cm. Any longer and the sentence might run off of the page. Any shorter and there would be a lot of wasted space on the page. For every print run, the first and last books are taken for measurement. The line lengths are measured on a specified page in the middle of each book.

You want to know: Is this process in control (stable)? Are we getting consistent print quality? What happens when we make improvements to the printing process? Does quality improve? To answer these questions, we need to create control charts and use control limits.

This example is in three parts. In most cases, you would start with "Create the Baseline Control Chart", where you let JMP calculate the control limits for you. Then, to apply these control limits to new data, you would either "Specify Control Limits" or "Specify Multiple Sets of Control Limits" (for phase data).

Create the Baseline Control Chart

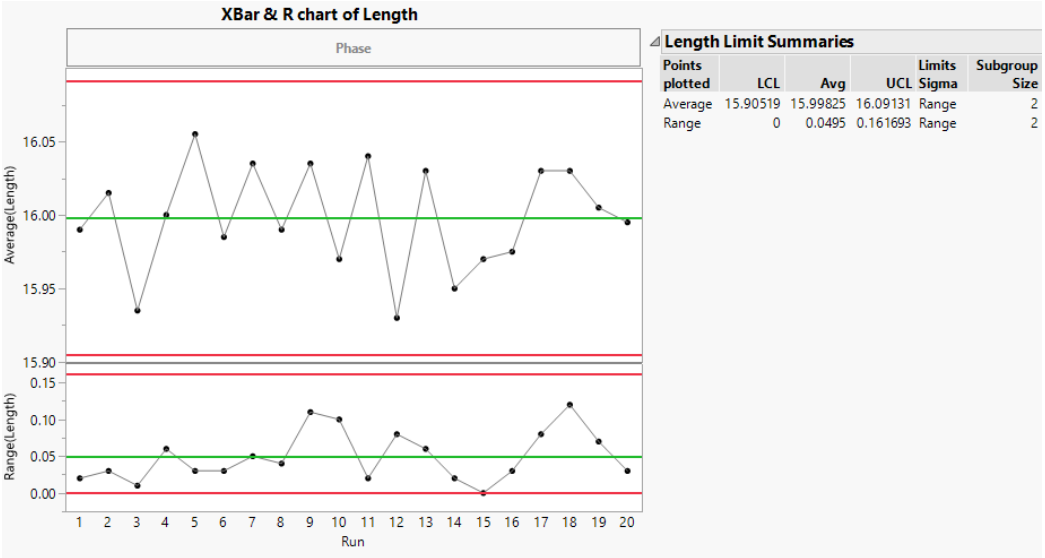
First, examine whether the existing process is in control. If it is, we can use the control limits created by JMP as our baseline or historical limits.

1. Select **Help > Sample Data Library** and open Quality Control/Line Length.jmp.

2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Length to the **Y** zone.

   An Individual and Moving Range chart of Length appears. This chart is appropriate if you have no natural subgrouping in your data. However, in this example, there is a natural subgrouping, which is each print run.
4. Drag Run to the **Subgroup** zone (at bottom).

Figure 3.10 XBar and R Chart of Line Length by Print Run



Three lines are drawn horizontally across the XBar and R charts. These are the calculated LCL (lower control limit), Avg (average) and UCL (upper control limit).

Ideally, we would like for all of our points to fall within the control limits, and we would like for the points to fall randomly within these limits. Looking at the graph, we see that no points fall outside of the control limits, and there does not appear to be a pattern to the points. To investigate further, perform Western Electric tests to check for patterns and trends that would cause these tests to fail. (The Western Electric tests are also referred to as Nelson tests.)

5. In the XBar chart, right-click and select **Warnings > Tests > All Tests**.

   Notice that no points were circled or flagged. This means that our process is in control or stable.

   If we had determined that our process was not in control, we would investigate out of control points or work to alter our process so that it is in control. For this example, since the process is already in control or stable, we can skip that step. Now, you can use these

control limits with new data. Proceed to "Specify Control Limits", or "Specify Multiple Sets of Control Limits" (to see an example with phase data).

Specify Control Limits

Since we established that the process is in control, we can use these historical limits with new data to see how the new data compares to the existing process. To use historical limits, we need to specify control limits instead of having JMP calculate them.

There are several ways to specify control limits in JMP:

- "Set Control Limits Option"
- "Add a Column Property"
- "Use the Get Limits Option"
- "Exclude Rows"

Set Control Limits Option

One simple way to specify control limits is to use the Set Control Limits option in Control Chart Builder.

1. Select **Help > Sample Data Library** and open Quality Control/New Length Data.jmp.

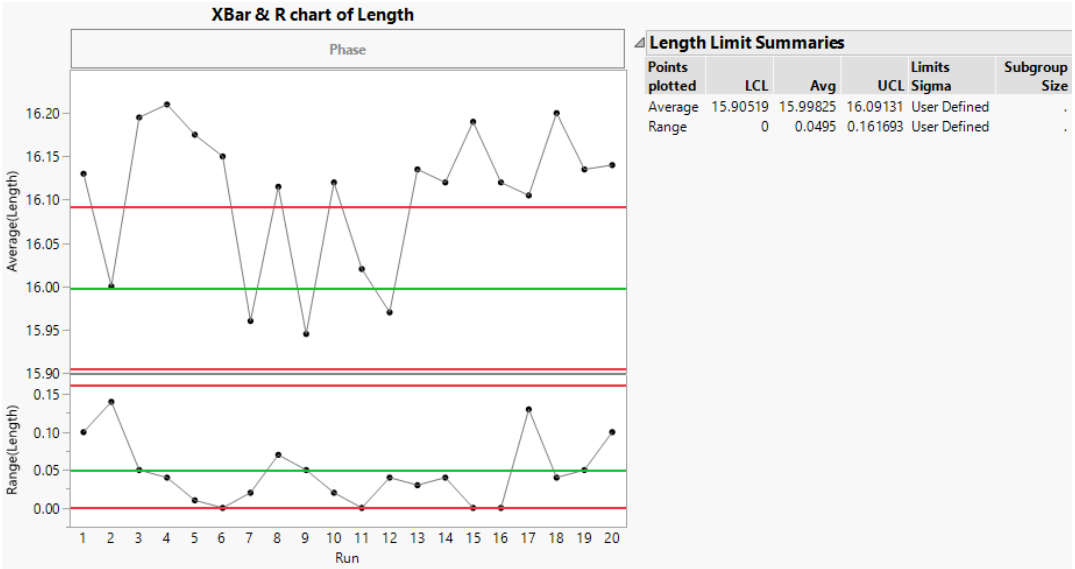
   This is the table that contains your new data.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Length to the **Y** zone.
4. Drag Run to the **Subgroup** zone (at bottom).
5. Right-click in the Average (XBar) chart and select **Limits > Set Control Limits**.
6. Enter these limits:
   - LCL - 15.90519
   - Avg - 15.99825
   - UCL - 16.09131

   These are the historical limits from the Average (XBar) chart in Figure 3.10.
7. Click **OK**.
8. Right-click in the Range (R) chart and select **Limits > Set Control Limits**.
9. Enter these limits:
   - LCL - 0
   - Avg - 0.0495
   - UCL - 0.161693

   These are the historical limits from the Range (R) chart in Figure 3.10.

10. Click **OK**.

Figure 3.11 XBar and R Chart of Line Length with Historical Limits



Rather than calculating limits from the data, JMP used the historical control limits that you defined. In the Length Limit Summaries table, notice that the Limits Sigma now says User Defined. Many points now fall outside of the limits. Also, the averages are higher than those of the baseline process. This process appears different from the original process that we used to calculate the baseline control limits.

Add a Column Property

Another way to specify control limits is to add the Control Limits column property to a column in your new data table.

1. Select **Help > Sample Data Library** and open Quality Control/New Length Data.jmp.

   This is the table that contains your new data.
2. Select the Length column and click **Cols > Column Info**.
3. Click **Column Properties > Control Limits**.
4. XBar is selected, so enter these fixed limits for the Average (XBar) chart:
   - Avg - 15.99825
   - LCL - 15.90519
   - UCL - 16.09131

   These are the historical limits from the Average (XBar) chart in Figure 3.10.

Leave the value for Subgroup Size as missing. This value is not used in the Control Chart Builder platform.

5. Click **XBar > R**. Enter these fixed limits for the Range (R) chart:
   - Avg - 0.0495
   - LCL - 0
   - UCL - 0.161693

   These are the historical limits from the Range (R) chart in Figure 3.10.

   Leave the value for Subgroup Size as missing. This value is not used in the Control Chart Builder platform.
6. Click **OK**.

   You have entered control limits for XBar and R charts for the Length column. Now you can create a control chart.
7. Select **Analyze > Quality and Process > Control Chart Builder**.
8. Drag Length to the **Y** zone.
9. Drag Run to the **Subgroup** zone (at bottom).

The control chart is identical to Figure 3.11.

Use the Get Limits Option

The Get Limits method of specifying control limits is the most flexible. You should use this method in the following cases:

- If you have control limits for many different processes
- If you have different control limits for each phase (see "Specify Multiple Sets of Control Limits")

To use the Get Limits method, you need a data table that defines your historical limits. For more information about how to create a limits table, see "Saving and Retrieving Limits".

Note: When no subgroup variable is specified, the Get Limits option uses the subgroup size (_Sample Size) from the limits table. Also, when the limits are missing in the file, JMP also looks for the sigma (_Std Dev). When no LCL or UCL are specified in the limits file (if both the average and sigma are found, and the subgroup size is constant), the option sets the limits based on the average, subgroup size, and sigma.

In this example, a limits data table has already been created.

1. Select **Help > Sample Data Library** and open Quality Control/New Length Data.jmp.

   This is the table that contains your new data.
2. Select **Analyze > Quality and Process > Control Chart Builder**.

3. Drag Length to the **Y** zone.
4. Drag Run to the **Subgroup** zone (at bottom).
5. Click the Control Chart Builder red triangle and select **Get Limits**.
6. Select **Other** and click **OK**.
7. Navigate and open the limits data table for this example, called Length Limits.jmp. By default, the file is located here:
   - On Windows: C:\Program Files\SAS\JMP\16\Samples\Data\Quality Control
   - On macOS: \Library\Application Support\JMP\16\Samples\Data\Quality Control

The control chart is identical to Figure 3.11.

Exclude Rows

Another way to specify control limits is to exclude rows in a data table. One advantage to this method is that you can see both the historical data and new data in the same graph. This can help to visualize and investigate differences when they occur between the data collection periods.

To use this method, you must meet the following criteria:

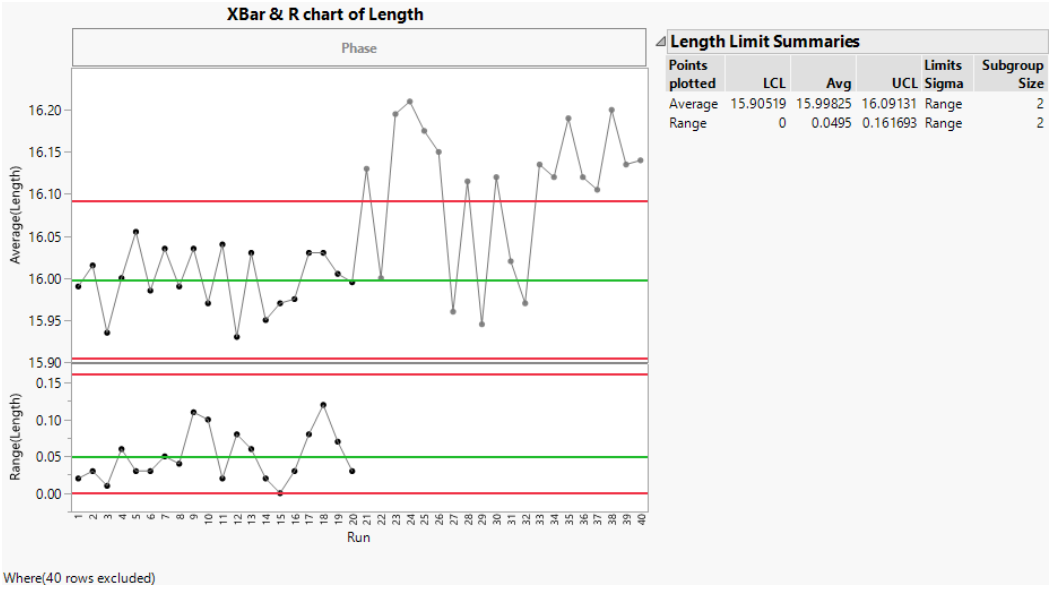
- New and old data must reside in the same data table.
- Historical data and new data must all have equal subgroup sizes.
- All new data must be excluded in the data table (using Rows > Exclude/Unexclude).

In this example, new data have already been excluded.

1. Select **Help > Sample Data Library** and open Quality Control/Combined.jmp.

   This table contains old and new data, and the rows corresponding to the new data are excluded.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Length to the **Y** zone.
4. Drag Run to the **Subgroup** zone (at bottom).

Figure 3.12 XBar and R Chart of Line Length with Excluded Data



JMP uses only the unexcluded rows (historical data) to create the control limits. The new data (excluded data) are plotted on the graph (dimmed), but these data were not used in any of the calculations.

Specify Multiple Sets of Control Limits

In this example, you want to set different control limits for different phases of a process. The column property, set control limits, and excluded row state methods will not work in this situation because these methods are limited to only one set of control limits for the entire chart. For a control chart with phases, you need to use the get limits method.

In the printing company, the goal is to reduce the variability of the force needed to break the bond between paper and the book spine for three different sites. Each site has different machines, different operators, and is also located in different countries; therefore, each site has a unique set of historical limits. For all three sites, the company does the following:

1. Creates a baseline control chart based on the existing process data.
2. Changes the process, based on a designed experiment.
3. Gathers data from the new process.
4. Creates a new control chart based on the new process data.

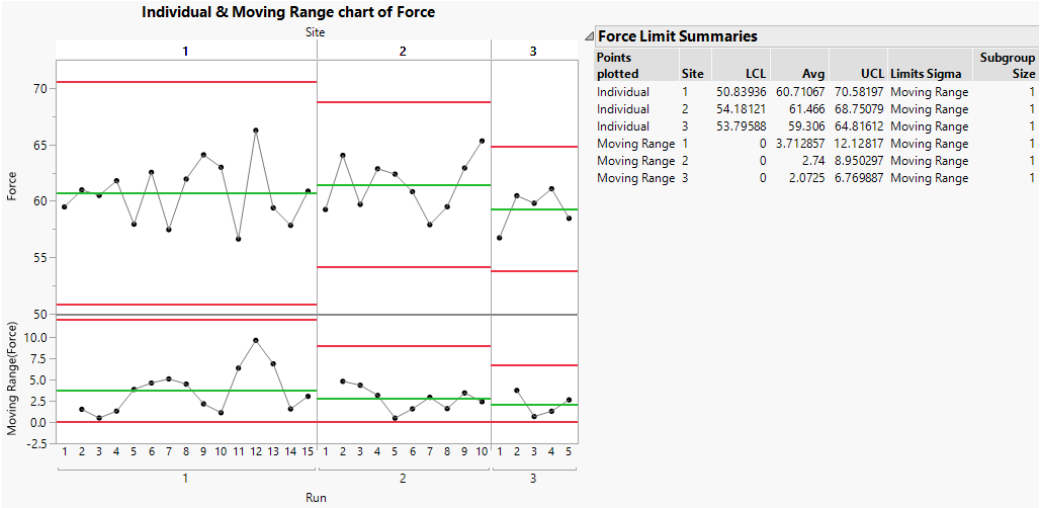
The goal is to plot the new data on a control chart using historical limits from the old process. In this way, the printing company can compare the new process to the old process limits.

Create a Control Chart Based on Existing Process

1. Select **Help > Sample Data Library** and open Quality Control/Phase Historical Data.jmp.

   This table contains the existing process data for all three sites.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Force to the **Y** zone.
4. Drag Run to the **Subgroup** zone (at bottom).
5. Drag Site to the **Phase** zone.

Figure 3.13 Baseline Control Chart for Existing Data



Force Limit Summaries

Points plotted	Site	LCL	Avg	UCL	Limits Sigma	Subgroup Size
Individual	1	50.83936	60.71067	70.58197	Moving Range	1
Individual	2	54.18121	61.466	68.75079	Moving Range	1
Individual	3	53.79588	59.306	64.81612	Moving Range	1
Moving Range	1	0	3.712857	12.12817	Moving Range	1
Moving Range	2	0	2.74	8.950297	Moving Range	1
Moving Range	3	0	2.0725	6.769887	Moving Range	1

Create a Control Chart Based on Updated Process

1. From the report in Figure 3.13, click the Control Chart Builder red triangle and select **Save Limits > in New Table**.

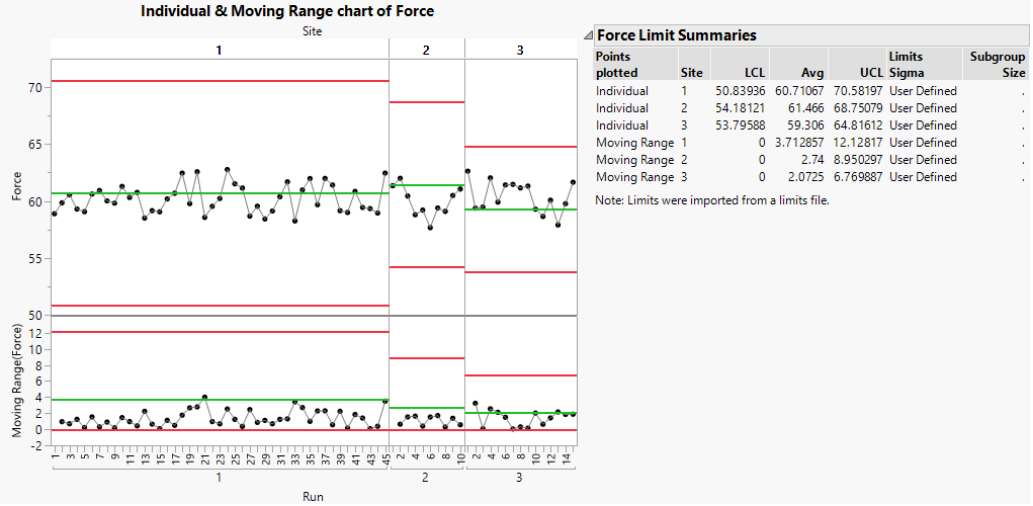
   This creates a limits table.
2. Save this new limits table to any location, so you can access it later.
3. Select **Help > Sample Data Library** and open Quality Control/Phase New Data.jmp.

   This data was collected from all three sites after the change was made to the process.
4. Select **Analyze > Quality and Process > Control Chart Builder**.
5. Drag Force to the **Y** zone.
6. Drag Run to the **Subgroup** zone (at bottom).
7. Drag Site to the **Phase** zone.

8. Click the Control Chart Builder red triangle and select **Get Limits**. Open the limits table that you saved in step 2.

   This applies the historical limits to the new data in the Control Chart Builder report.

Figure 3.14 Control Chart for New Data Based on Historical Limits



Now you can see how the new data (after the process change) compare with the historical process limits (before the process change). None of the points fall outside of the control limits for either the location or dispersion chart. The goal was to reduce variability. Looking at the moving range chart, you can see that most points fall below the average line. For sites 1 and 2, it is clear that the variability of force needed to break the bond between pages and the book spine has been decreased. The decrease at Site 3 is not as strong as at sites 1 and 2. The improvements to the printing process appear to have succeeded in reducing the variability.

Excluded and Hidden Samples in Control Chart Builder

The following bullets summarize the use of excluded and hidden samples in control chart builder:

- Excluded subgroups are not used in the calculations of control limits, and appear on the chart as dimmed points by default. If the Show Excluded Region option is not selected, the points for the excluded subgroups do not appear in the chart, are treated as missing in Tests for Special Causes, and are not included in the count of points for Tests for Special Causes.

- Hidden observations are used in the calculations of control limits, but do not appear in the chart.
- Rows that are both hidden and excluded are included in the count of points for Tests for Special Causes when the Test Excluded Subgroups option is selected. An excluded row can be labeled with a special cause flag. A hidden point cannot be labeled. If the flag for a Tests for Special Causes test is on a hidden point, it will not appear in the chart.
- For partially excluded subgroups, if one or more observations within a subgroup is excluded, and at least one observation within the subgroup is included, the excluded observation is not included in the calculations of either the point statistic or the limits.
- Checks for negative and non-integer data happen on the entire data (even excluded values).
- Tests apply to all excluded subgroups only when the Test Excluded Subgroups option is selected.

Additional Examples of Control Chart Builder

Note: In this section, some examples show the Control Panel while others do not. To show or hide the Control Panel, select Show Control Panel from the Control Chart Builder red triangle menu.

- "Individual Measurement and Moving Range Charts Example"
- "XBar and R Chart Phase Example"
- "XBar and S Charts with Varying Subgroup Sizes Example"
- "Run Chart Example"
- "P chart Example"
- "NP chart Example"
- "C chart Example"
- "U chart Example"
- "G chart Example"
- "T chart Example"
- "Three Way Control Chart Example"
- "Example of Multiple Control Charts"

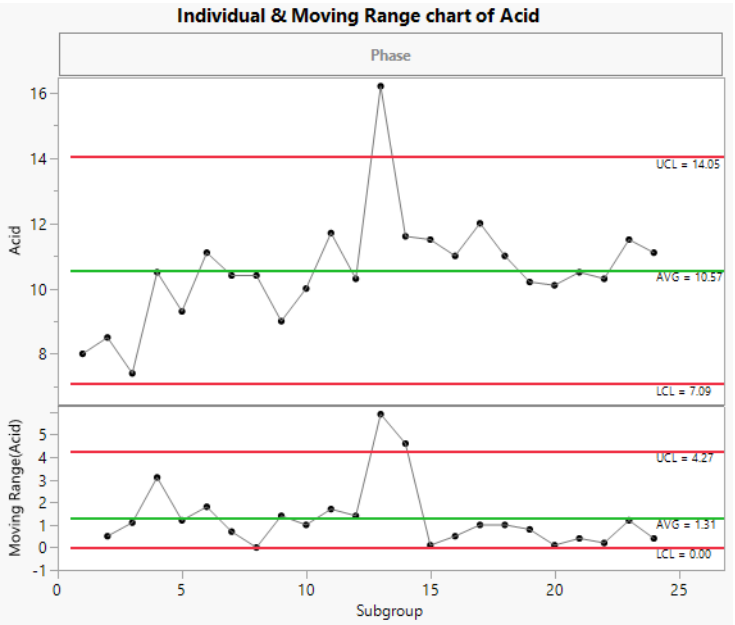
Individual Measurement and Moving Range Charts Example

The Pickles.jmp data in the Quality Control sample data folder contains the acid content for vats of pickles. Because the pickles are sensitive to acidity and produced in large vats, high acidity ruins an entire pickle vat. The acidity in four vats is measured each day at 1, 2, and 3 PM. The data table records day, time, and acidity measurements.

1. Select **Help > Sample Data Library** and open Quality Control/Pickles.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Acid to the **Y** role.
4. Click the **Control Chart Builder** red triangle and select **Show Limit Labels**.

   This option labels the control limits and averages in both charts.

Figure 3.15 Individual Measurement and Moving Range Charts for Acid

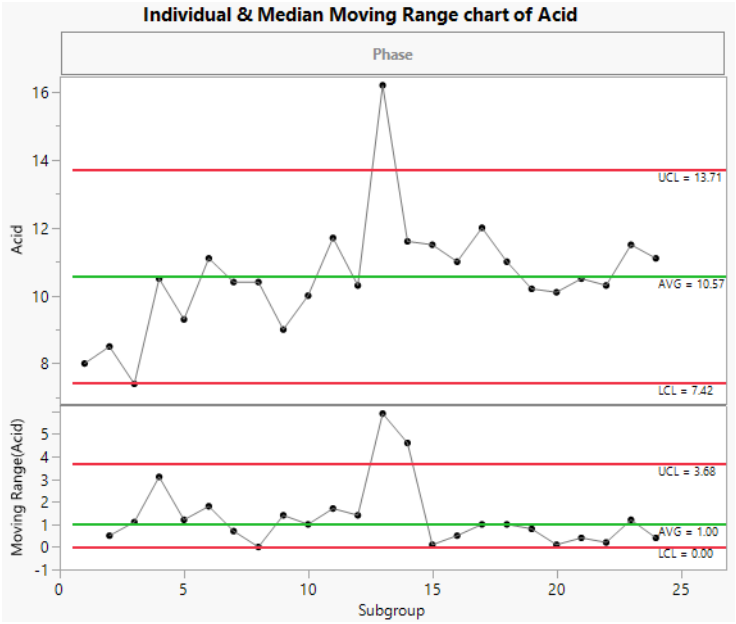


   The individual measurement and moving range charts monitor the acidity in each vat produced (subgroup of size 1). Vat 13 has an acidity above the upper control limit of 14.05.

You can also view a Median Moving Range chart. Continue with the following steps to change the charts to use median moving ranges.

5. In the Limits[1] outline, change the **Sigma** setting to **Median Moving Range**.
6. In the Limits[2] outline, change the **Sigma** setting to **Median Moving Range**.

Figure 3.16 Individual Measurement and Median Moving Range Charts for Acid



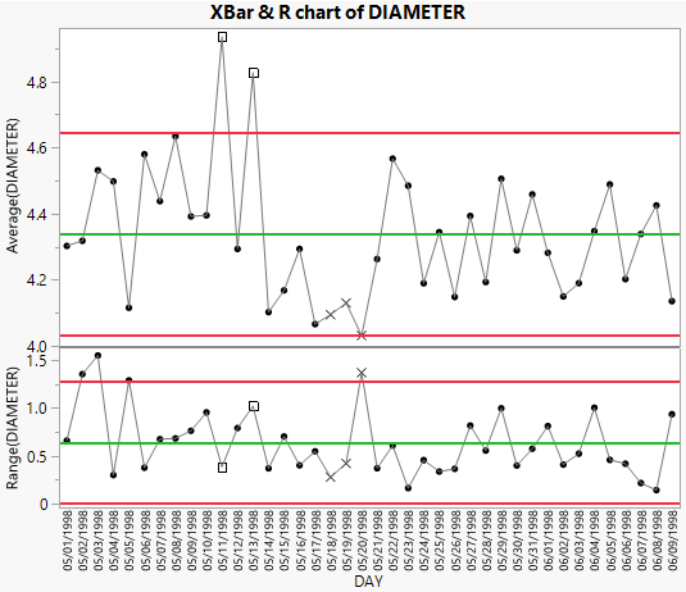
The limits in the individual measurement and median moving range charts use the median moving range as the sigma, rather than the average moving range. This results in slightly narrower control limits for Acid.

XBar and R Chart Phase Example

A manufacturer of medical tubing collected tube diameter data for a new prototype. The data was collected over the past 40 days of production. After the first 20 days (phase 1), some adjustments were made to the manufacturing equipment. Analyze the data to determine whether the past 20 days (phase 2) of production are in a state of control.

1. Select **Help > Sample Data Library** and open Quality Control/Diameter.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag DIAMETER to the **Y** role.
4. Drag DAY to the **Subgroup** role (at bottom).

Figure 3.17 Control Charts for Diameter

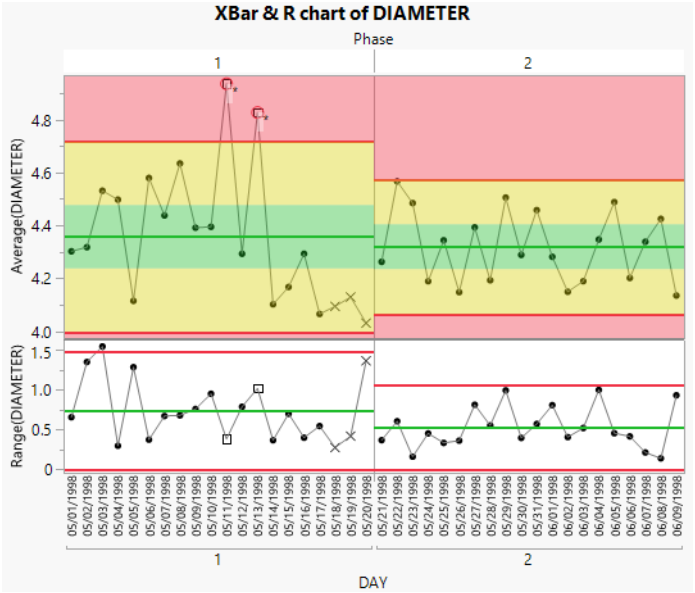


The first 20 days appear to have high variability, and in the Average chart, there are three observations that are outside of the control limits. An adjustment was made to the manufacturing equipment and new control limits were incorporated.

To compute separate control limits for each phase:

5. Drag Phase to the **Phase** role.
6. In the Average chart, right-click and select **Warnings > Test Beyond Limits**.
7. In the Limits[1] outline, select **Shade Zones**.

Figure 3.18 Control Charts for each Phase



Including the Phase variable means that the control limits for phase 2 are based only on the data for phase 2. None of the phase 2 observations are outside the control limits. This is highlighted by including the zone shading on the chart. Therefore, you can conclude that the process is in control after the adjustments were made.

Filter the Control Chart by Another Variable

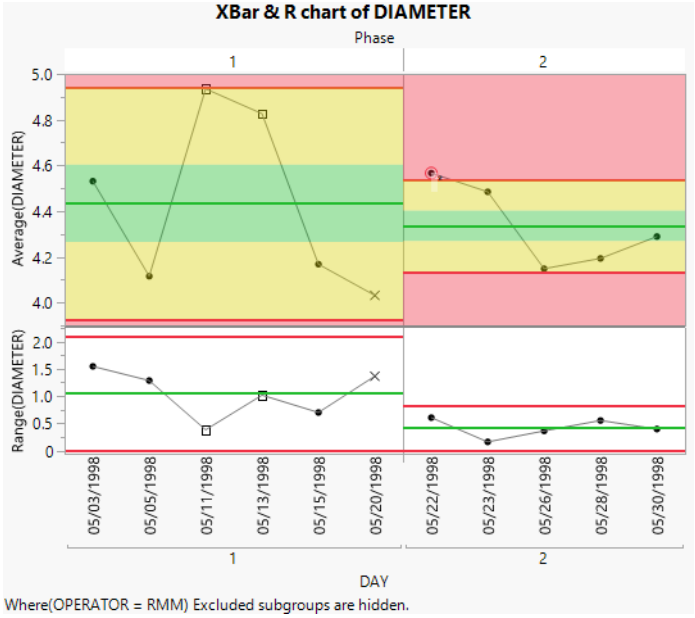
This data table, Diameter.jmp, contains a column for the operator of the machine for each sample. You can use the Local Data Filter with Control Chart Builder to show the data for a subset of operators.

8. Click the **Control Chart Builder** red triangle and deselect **Show Excluded Region**.

   Turning off the Show Excluded Region option indicates that the subgroups that are excluded by settings in the Local Data Filter no longer appear on the horizontal axis of the control chart as you make selections in the Local Data Filter. As a result, you see only the portion of the data that are of interest.

9. Click the **Control Chart Builder** red triangle and select **Local Data Filter**.
10. In the Local Data Filter, click on OPERATOR and click the + button.
11. In the Local Data Filter, select the bar labeled RMM.

Figure 3.19 XBar and R Chart of Diameter for Operator RMM



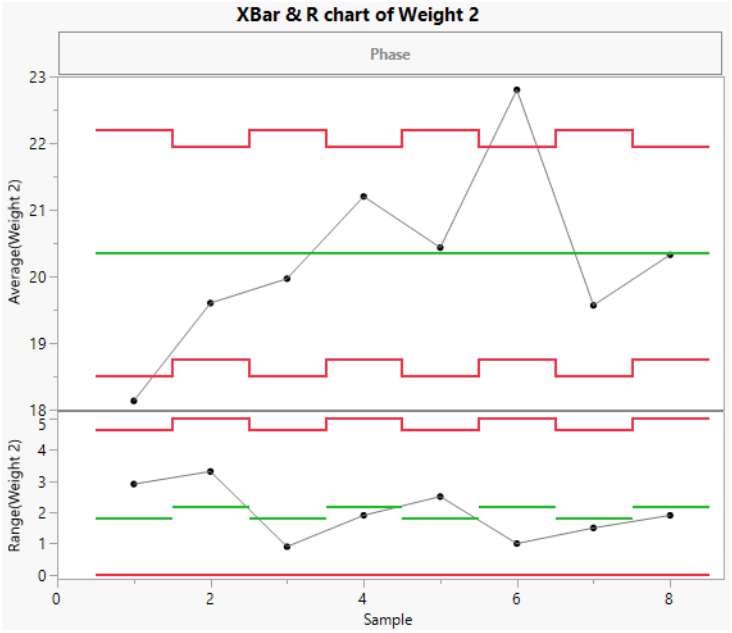
The XBar and R charts now show only the points for the RMM operator, as denoted by the Where() statement below the charts. The limits for both phases have been adjusted to reflect that the observations for the other three operators have been excluded.

XBar and S Charts with Varying Subgroup Sizes Example

This example uses the Coating.jmp data table. This quality characteristic of interest is the Weight 2 column.

1. Select **Help > Sample Data Library** and open Quality Control/Coating.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Weight 2 to the **Y** role.
4. Drag Sample to the **Subgroup** role (at bottom).

Figure 3.20 XBar and S Charts for Varying Subgroup Sizes

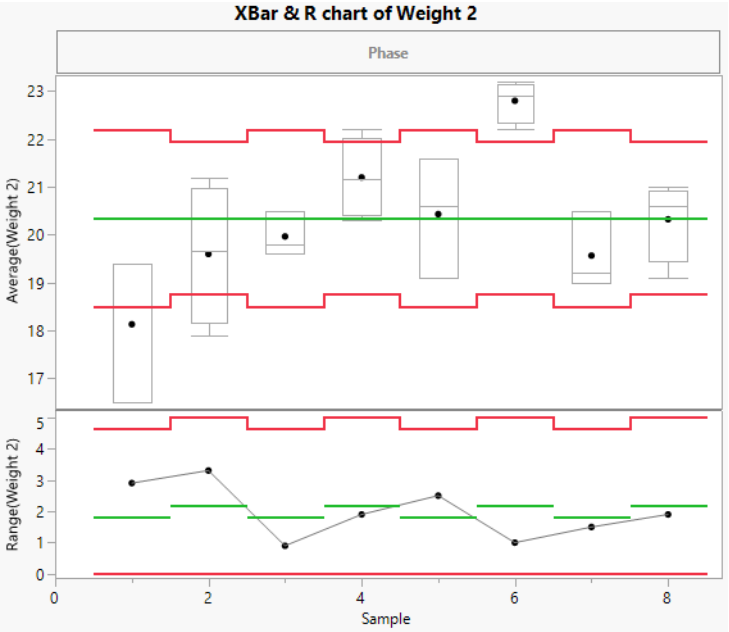


Weight 2 has several missing values in the data, so the chart has uneven limits. Although each sample has the same number of observations, samples 1, 3, 5, and 7 each have a missing value.

Instead of viewing a line connecting the averages of each sample, you can switch to viewing box plots at each sample.

5. In the **Points[1]** outline, deselect the **Show Connected Line** option.
6. In the **Points[1]** outline, select the **Box Plots** option.

Figure 3.21 XBar and S Chart with Box Plots

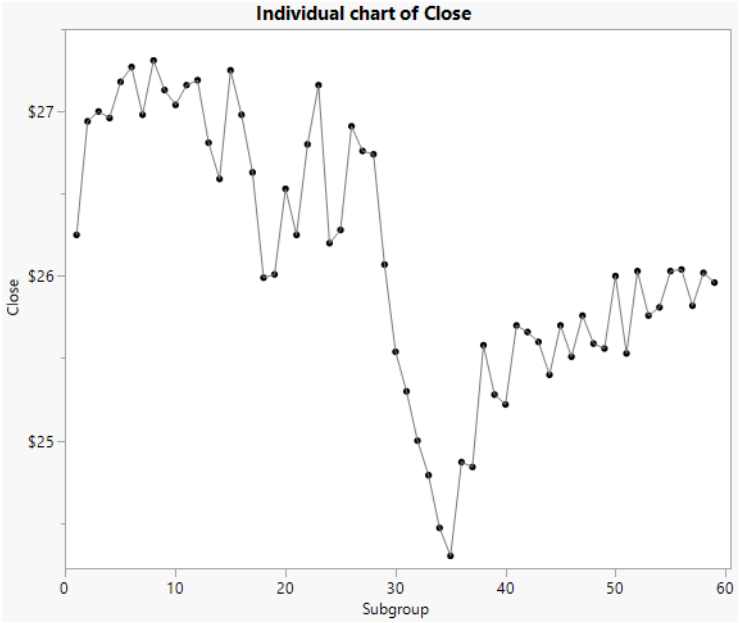


Run Chart Example

Run charts display a column of data as a connected series of points. This example is a Run chart for the Close variable from Stock Averages.jmp.

1. Select **Help > Sample Data Library** and open Stock Averages.jmp.
2. Select **Analyze > Quality and Process > Control Chart > Run Chart**.
3. Select Close and click **Y**.
4. Click **OK**.

Figure 3.22 Run Chart for Stock Averages Closing Price



P chart Example

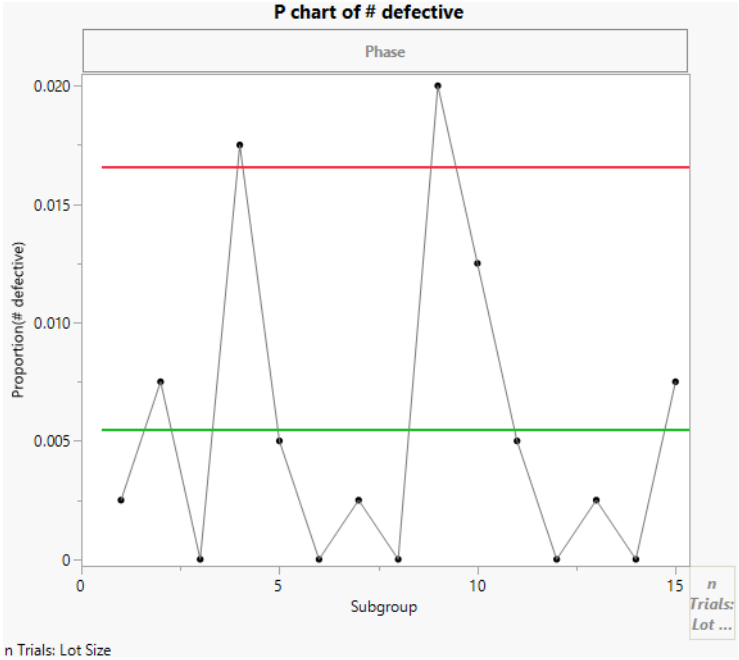
The Washers.jmp sample data contains defect data for two different lot sizes from the *ASTM Manual on Presentation of Data and Control Chart Analysis*, American Society for Testing and Materials (1976). To view the differences between constant and variable sample sizes, you can compare charts for Lot Size and Lot Size 2.

The Washers.jmp data in the Quality Control sample data folder contains defect counts of 15 lots of 400 galvanized washers. The washers were inspected for finish defects such as rough galvanization and exposed steel. If a washer contained a finish defect, it was deemed nonconforming or defective. Thus, the defect count represents how many washers were defective for each lot of size 400. Using the Washers.jmp data table, specify a sample size variable, which would allow for varying sample sizes. This data table contains all constant sample sizes.

1. Select **Help > Sample Data Library** and open Quality Control/Washers.jmp.
2. Select **Analyze > Quality and Process > Control Chart > P Control Chart**.
3. Select # defective and click **Y**.
4. Select Lot Size and click **n Trials**.
5. Click **OK**.
6. In the Limits outline, deselect the **Show Lower Limit** option.

This hides the lower limit, which is not of interest in this situation.

Figure 3.23 P chart of # defective with sample size



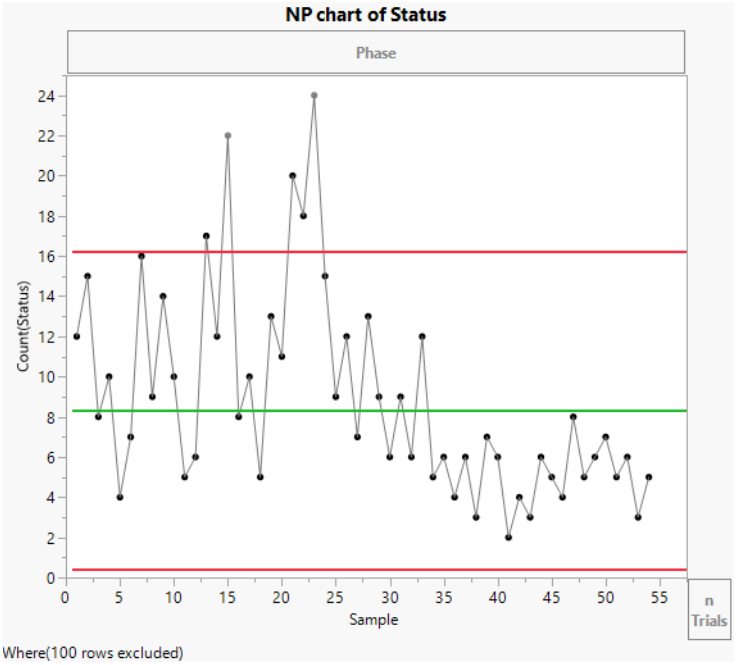
To view the differences between constant and variable sample sizes, you can compare charts for Lot Size and Lot Size 2 by simply dragging the variables to the nTrials zone.

NP chart Example

The Bottle Tops.jmp sample data contains simulated data from a bottle top manufacturing process. Sample is the sample ID number for each bottle. Status indicates whether the bottle top conformed to the design standards. In the Phase column, the first phase represents the time before the process adjustment. The second phase represents the time after the process adjustment. Notes on changes in the process are also included.

1. Select **Help > Sample Data Library** and open Quality Control/Bottle Tops.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Sample to the **Subgroup** role.
4. Drag Status to the **Y** role.

Figure 3.24 NP chart of Status (Nonconforming)

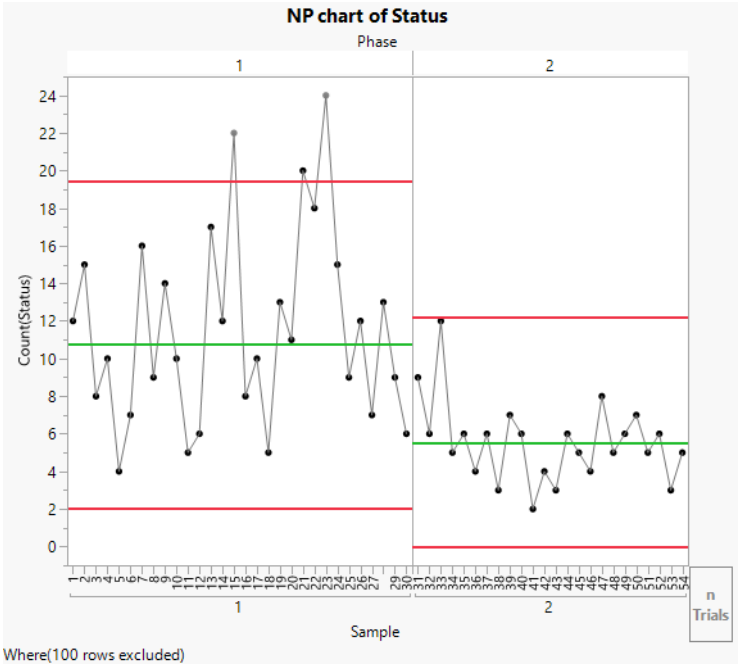


The original observations appear to have high variability and there are five observations (Samples 13, 15, 21, 22 and 23) that are outside of the upper control limit. Samples 15 and 23 note that new material and a new operator were introduced into the process, respectively. At the end of the phase, an adjustment was made to the manufacturing equipment. Therefore, the control limits for the entire series should not be used to assess the control during phase 2.

To compute separate control limits for each phase:

5. Drag Phase to the **Phase** zone.

Figure 3.25 NP chart by Phase



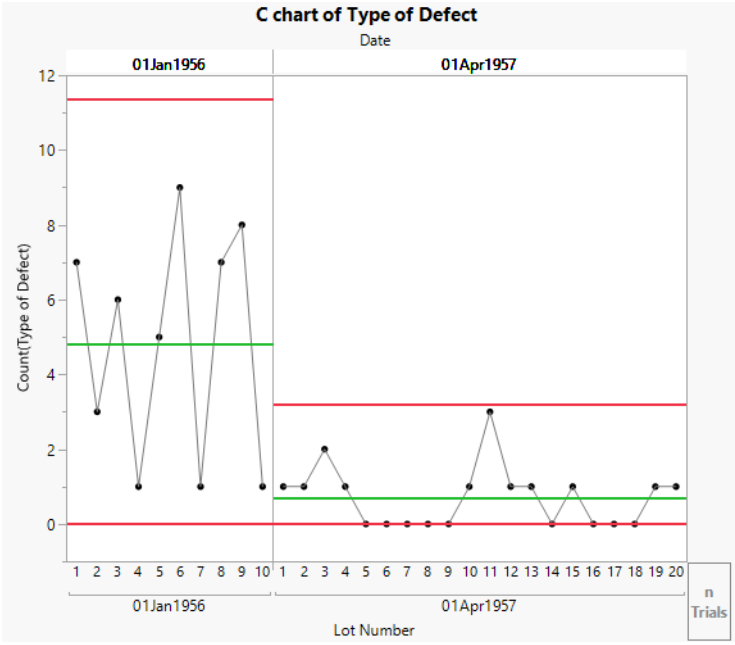
Including the Phase variable means that the control limits for phase 2 are based only on the data for phase 2. None of the phase 2 observations are outside the control limits. Therefore, you can conclude that the process is in control after the adjustment.

C chart Example

The Cabinet Defects.jmp sample data table contains data concerning the various defects discovered while manufacturing cabinets over two time periods.

1. Select **Help > Sample Data Library** and open Quality Control/Cabinet Defects.jmp.
2. Select **Analyze > Quality and Process > Control Chart > C Control Chart**.
3. Select Type of Defect and click **Y**.
4. Select Lot Number and click **Subgroup**.
5. Select Date and click **Phase**.
6. Click **OK**.

Figure 3.26 C chart of Type of Defect with Phases



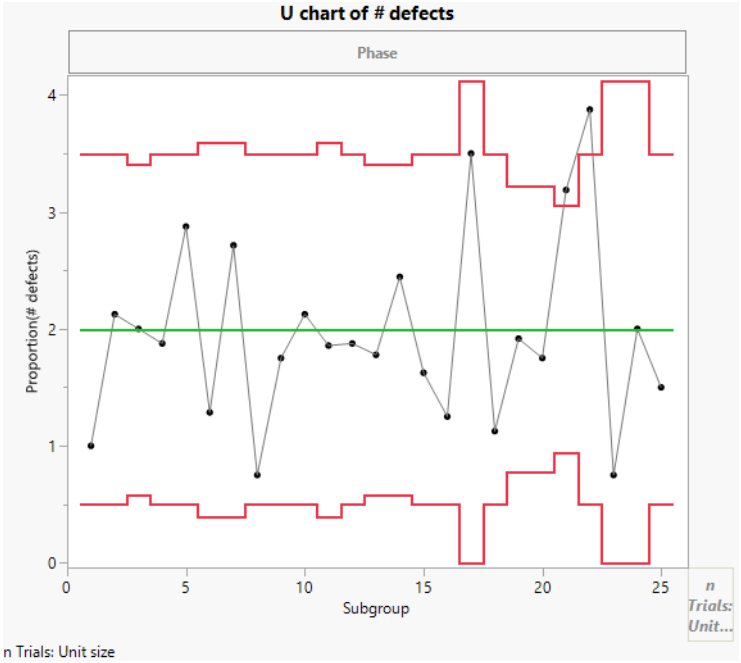
You can now view the results on the two different days. Both appear to be within limits. To examine other defect type behavior, select another defect type under the **Event Chooser > Type of Defect** and view the results as the limits are updated.

U chart Example

The Braces.jmp data in the Quality Control sample data folder records the defect count in boxes of automobile support braces. A box of braces is one inspection unit. The number of boxes inspected (per day) is the subgroup sample size, which can vary. The U chart in Figure 3.27 is monitoring the number of brace defects per subgroup sample size. The upper and lower bounds vary according to the number of units inspected.

1. Select **Help > Sample Data Library** and open Quality Control/Braces.jmp.
2. Select **Analyze > Quality and Process > Control Chart > U Control Chart**.
3. Select # defects and click **Y**.
4. Select Unit size and click **n Trials**.
5. Click **OK**.

Figure 3.27 U chart of # Defects



Because the sample sizes are not equal across subgroups, the limits are uneven. Two of the last five samples are not within the control limits.

G chart Example

A G chart is an effective way to understand whether rare events are occurring more frequently than expected and warrant an intervention. See "Rare Event Control Charts".

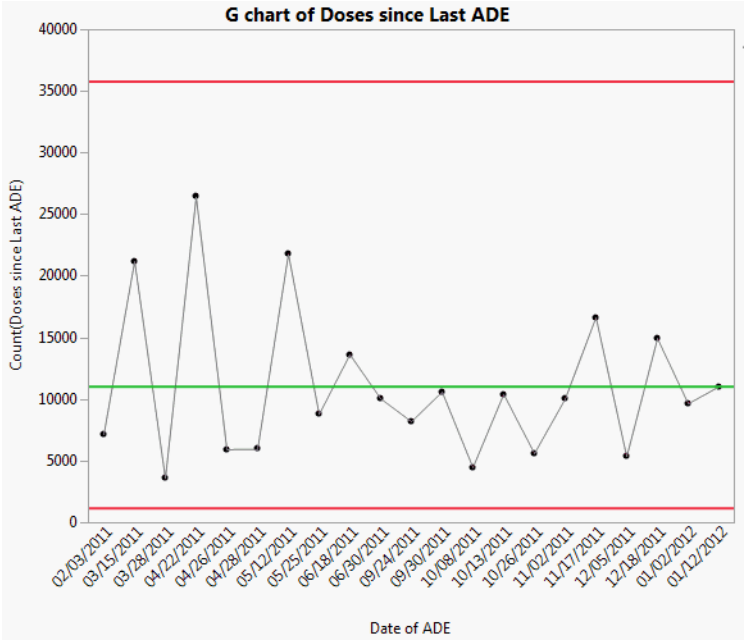
The Adverse Reactions.jmp sample data table contains simulated data about adverse drug events (ADEs) reported by a group of hospital patients. An ADE is any type of injury or reaction the patient suffered after taking the drug. The date of the reaction and the number of days since the last reaction were recorded.

1. Select **Help > Sample Data Library** and open Quality Control/Adverse Reactions.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Doses since Last ADE to the **Y** role.
4. Drag Date of ADE to the **Subgroup** role.

   An Individual & Moving Range chart of Doses since Last ADE appears.
5. In the drop-down list, select **Rare Event** instead of **Shewhart Variables**.

A G chart of Doses since Last ADE appears, showing that the number of doses given since the last event.

Figure 3.28 G chart of Doses since Last ADE



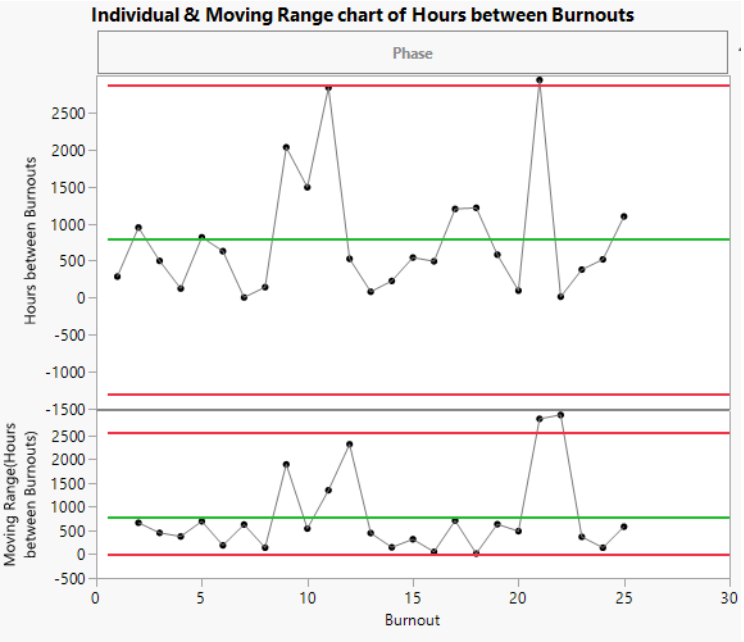
T chart Example

T charts are used to measure the time that has elapsed since the last event. See "Rare Event Control Charts".

The Fan Burnout.jmp sample data table contains simulated data for a fan manufacturing process. The first column identifies each fan that burned out. The second column identifies the number of hours between each burnout.

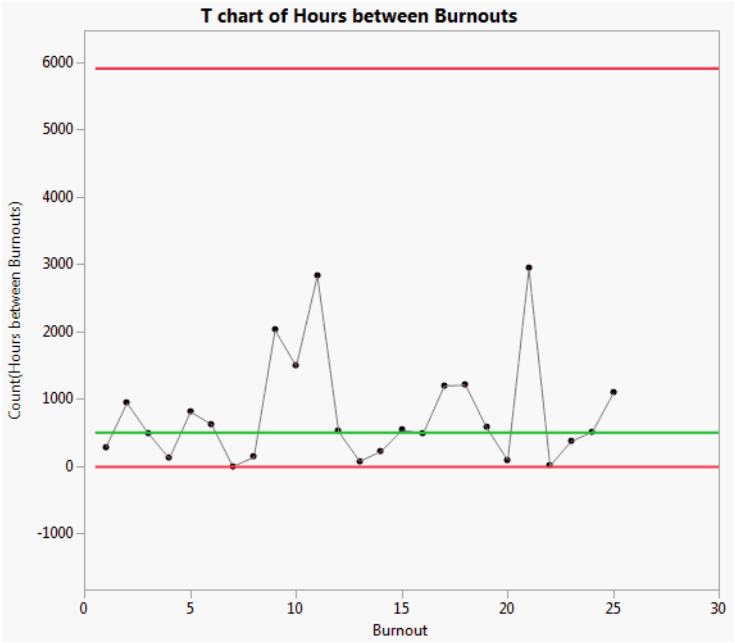
1. Select **Help > Sample Data Library** and open Quality Control/Fan Burnout.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Hours between Burnouts to the **Y** role.
4. Drag Burnout to the **Subgroup** role.

Figure 3.29 Individual and Moving Range Chart of Hours Between Burnouts



5. In the drop-down list, select **Rare Event** instead of **Shewhart Variables**.
6. Under Limits, change the **Sigma** from **Negative Binomial** to **Weibull**.

Figure 3.30 T chart of Hours Between Burnouts



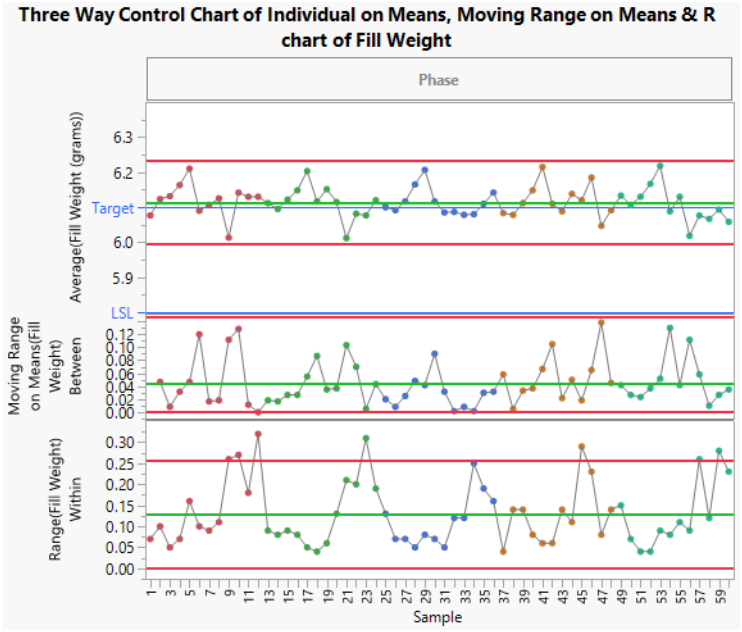
In the T chart, all points appear to be within the control limits. It is clear that the Individual & Moving Range chart was inappropriate for the analysis, as the limits were too narrow.

Three Way Control Chart Example

Three way control charts are useful when there is variation between batches and variation within batches.

1. Select **Help > Sample Data Library** and open Quality Control/Vial Fill Weights.jmp.
2. Select **Analyze > Quality and Process > Control Chart > Three Way Control Chart**.
3. Select Fill Weight and click **Y**.
4. Select Sample and click **Subgroup**.
5. Click **OK**.

Figure 3.31 Three Way Control Chart for Fill Weight

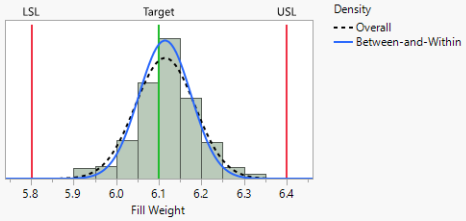


Fill Weight Limit Summaries

Points plotted	LCL	Avg	UCL	Limits Sigma	Subgroup Size
Average	5.995722	6.114611	6.233501	Moving Range	6
Moving Range on Means	0	0.044718	0.146071	Moving Range	6
Range	0	0.129	0.258494	Range	6

Process Capability Analysis

Histogram



Process Summary

LSL	5.8
Target	6.1
USL	6.4
N	360
N Subgroups	60
Sample Mean	6.114611
Between Sigma	0.033745
Within Sigma	0.050899
Between-and-Within Sigma	0.061069
Overall Sigma	0.069929
Stability Index	1.145077

Within sigma estimated by average of ranges.
Between sigma estimated by average moving range of subgroup means.

Between-and-Within Sigma Capability

Index	Estimate
Cpk	1.558
Cpl	1.717
Cpu	1.558
Cp	1.637
Cpm	1.593

Overall Sigma Capability

Index	Estimate	Lower 95%	Upper 95%
Ppk	1.360	1.255	1.466
Ppl	1.500	1.385	1.614
Ppu	1.360	1.255	1.465
Pp	1.430	1.325	1.535
Cpm	1.400	1.299	1.504

Nonconformance

Portion	Observed %	Expected Between-and-Within %	Expected Overall %
Below LSL	0.0000	0.0000	0.0003
Above USL	0.0000	0.0001	0.0022
Total Outside	0.0000	0.0002	0.0026

A Moving Range chart appears between the Range and Average charts. The limits on the Average (XBar) chart are now calculated using the moving range between each sample.

Example of Multiple Control Charts

You can create multiple control charts in the same window using the New Y Chart option in the interactive workspace or by specifying multiple processes in a specific control chart launch window. In this example, use the I/MR control chart launch window and then save the control limits to a new data table.

1. Select **Help > Sample Data Library** and open Semiconductor Capability.jmp.
2. Select **Analyze > Quality and Process > Control Chart > IMR Control Chart**.
3. Click the triangle next to Processes to show all of the processes.
4. Select the first five processes and click **Y**.
5. Click **OK**.

   If you scroll through the report window, there are Individual and Moving Range control charts for all five selected processes. Since the processes have a Spec Limits column property, there are also Process Capability Analysis reports for each process.
6. Click the Control Chart Builder red triangle and select **Save Limits > in New Tall Table**.

Figure 3.32 New Tall Table

	Process	_Std Dev	_Sample Size	_Sigma	_Mean	_LCL	_UCL	_AvgR	_LCLR	_UCLR
1	NPN1	2.6350722541	1	2.6350722541	114.79276556	106.8875488	122.69798232	2.9733606353	0	9.7125774226
2	PNP1	59.430128559	1	59.430128559	313.06971647	134.77933079	491.36010214	67.059718964	0	219.05271249
3	PNP2	79.27035928	1	79.27035928	456.61569285	218.80461501	694.42677069	89.44702198	0	292.18155238
4	NPN2	2.1316521676	1	2.1316521676	115.74213802	109.34718152	122.13709452	2.4053118974	0	7.8570280886
5	PNP3	6.1609122855	1	6.1609122855	137.6145741	119.13183725	156.09731096	6.9518450733	0	22.708423829

The control limits are saved to a new data table, with a row for each Y variable and a column for each statistic.

Statistical Details for Control Chart Builder

Notes:

- Control limits are calculated using a sigma multiplier, K, which is set to 3 by default. To change the default value of K, use the KSigma preference in File > Preferences > Platforms > Control Chart Builder.
- For sample sizes up to $n = 50$, JMP uses control chart constants $d_2(n)$ and $d_3(n)$ that are defined in Table 2 of Harter (1960). For samples with a sample size greater than 50, JMP

uses the control chart constant values for sample size 50 in both the sigma and control limit calculations.

- "Control Limits for XBar and R Charts"
- "Control Limits for XBar and S Charts"
- "Control Limits for Individual Measurement and Moving Range Charts"
- "Control Limits for P and NP Charts"
- "Control Limits for U Charts and C Charts"
- "Levey-Jennings Charts"
- "Control Limits for G Charts"
- "Control Limits for T Charts"
- "Sigma Calculations for Three Way Control Charts"

Control Limits for XBar and R Charts

JMP generates control limits for XBar and R charts using the following formulas:

LCL for XBar chart = $\bar{\bar{X}}_w - \frac{K\hat{\sigma}}{\sqrt{n_i}}$

UCL for XBar chart = $\bar{\bar{X}}_w + \frac{K\hat{\sigma}}{\sqrt{n_i}}$

LCL for R chart = $\max\left(d_2(n_i)\hat{\sigma} - Kd_3(n_i)\hat{\sigma}, 0\right)$

UCL for R chart = $d_2(n_i)\hat{\sigma} + Kd_3(n_i)\hat{\sigma}$

Center line for R chart: By default, the center line for the i^{th} subgroup indicates an estimate of the expected value of R_i. This value is computed as $d_2(n_i)\hat{\sigma}$, where $\hat{\sigma}$ is an estimate of σ.

The standard deviation for XBar and R charts is estimated using the following formula:

$$\hat{\sigma} = \frac{\frac{R_1}{d_2(n_1)} + \ldots + \frac{R_N}{d_2(n_N)}}{N}$$

where:

$\bar{X}_w$ = weighted average of subgroup means

K = the sigma multiplier and is set to 3 by default

σ = process standard deviation

n_i = sample size of i^{th} subgroup

$d_2(n)$ is the expected value of the range of n independent normally distributed variables with unit standard deviation

$d_3(n)$ is the standard deviation of the range of n independent normally distributed variables with unit standard deviation

R_i is the range of i^{th} subgroup

N is the number of subgroups for which $n_i \geq 2$.

Control Limits for XBar and S Charts

JMP generates control limits for XBar and S charts using the following formulas:

LCL for XBar chart = $\bar{\bar{X}}_w - \frac{K\hat{\sigma}}{\sqrt{n_i}}$

UCL for XBar chart = $\bar{\bar{X}}_w + \frac{K\hat{\sigma}}{\sqrt{n_i}}$

LCL for S chart = $\max\left(c_4(n_i)\hat{\sigma} - Kc_5(n_i)\hat{\sigma}, 0\right)$

UCL for S chart = $c_4(n_i)\hat{\sigma} + Kc_5(n_i)\hat{\sigma}$

Center line for S chart: By default, the center line for the i^{th} subgroup indicates an estimate of the expected value of s_i. This value is computed as $c_4(n_i)\hat{\sigma}$, where $\hat{\sigma}$ is an estimate of σ.

The estimate for the standard deviation for XBar and S charts is:

$$\hat{\sigma} = \frac{\frac{s_1}{c_4(n_1)} + \ldots + \frac{s_N}{c_4(n_N)}}{N}$$

where:

$\overline{X}_w$ = weighted average of subgroup means

K = the sigma multiplier and is set to 3 by default

σ = process standard deviation

n_i = sample size of i^{th} subgroup

$c_4(n)$ is the expected value of the standard deviation of n independent normally distributed variables with unit standard deviation

$c_5(n)$ is the standard error of the standard deviation of n independent normally distributed variables with unit standard deviation

N is the number of subgroups for which $n_i \geq 2$

s_i is the sample standard deviation of the i^{th} subgroup

Control Limits for Individual Measurement and Moving Range Charts

Control limits for Individual Measurement charts are computed as follows:

LCL for Individual Measurement Chart = $\overline{X} - K\hat{\sigma}$

UCL for Individual Measurement Chart = $\overline{X} + K\hat{\sigma}$

Control limits for Individual Measurement charts with sigma estimated by the median moving range are computed as follows:

LCL for Individual Measurement Chart = $\overline{X} - K\hat{\sigma}_{MMR}$

UCL for Individual Measurement Chart = $\overline{X} + K\hat{\sigma}_{MMR}$

Control limits for Moving Range charts are computed as follows:

LCL for Moving Range Chart = $\max\{d_2(n)\hat{\sigma} - Kd_3(n)\hat{\sigma}, 0\}$

UCL for Moving Range Chart = $d_2(n)\hat{\sigma} + Kd_3(n)\hat{\sigma}$

Control limits for Median Moving Range charts are computed as follows:

$LCL_{MMR} = \max(0, MMR - Kd_3(n)\ \hat{\sigma}_{MMR}\ )$

$UCL_{MMR} = MMR + Kd_3(n)\ \hat{\sigma}_{MMR}$

The standard deviation for Individual Measurement and Moving Range charts is estimated as follows:

$$\hat{\sigma} = \frac{\overline{MR}}{d_2(n)}$$

The standard deviation for Individual Measurement and Moving Range charts when using the median is estimated as follows:

$\hat{\sigma}_{MMR}$ = MMR/0.954

where:

$\overline{X}$ = the mean of the individual measurements

K = the sigma multiplier and is set to 3 by default

$\overline{MR}$ = the mean of the nonmissing moving ranges computed as

$(MR_2+MR_3+...+MR_N)/(N-1)$ where $MR_i = |x_i - x_{i-1}|$.

MMR = the median of the nonmissing moving ranges

σ = the process standard deviation

$d_2(n)$ = expected value of the range of n independent normally distributed variables with unit standard deviation.

$d_3(n)$ = standard deviation of the range of n independent normally distributed variables with unit standard deviation

Note: Moving Range charts in Control Chart Builder use a range span of $n = 2$.

Control Limits for P and NP Charts

The lower and upper control limits, LCL, and UCL, respectively, are computed using the following formulas:

P chart LCL = $\max(\bar{p} - K\sqrt{\bar{p}(1-\bar{p})/n_i}, 0)$

P chart UCL = $\min(\bar{p} + K\sqrt{\bar{p}(1-\bar{p})/n_i}, 1)$

NP chart LCL = $\max(n_i\bar{p} - K\sqrt{n_i\bar{p}(1-\bar{p})}, 0)$

NP chart UCL = $\min(n_i\bar{p} + K\sqrt{n_i\bar{p}(1-\bar{p})}, n_i)$

where:

$\bar{p}$ is the average proportion of nonconforming items taken across subgroups

$$\bar{p} = \frac{n_1 p_1 + \ldots + n_N p_N}{n_1 + \ldots + n_n} = \frac{X_1 + \ldots + X_N}{n_1 + \ldots + n_N}$$

n_i is the number of items in the i^{th} subgroup

K is the sigma multiplier and is set to 3 by default

Control Limits for U Charts and C Charts

The lower and upper control limits, LCL, and UCL, are computed using the following formulas:

U chart LCL = $\max(\bar{u} - K\sqrt{\bar{u}/n_i}, 0)$

U chart UCL = $\bar{u} + K\sqrt{\bar{u}/n_i}$

C chart LCL = $\max(n_i\bar{u} - K\sqrt{n_i\bar{u}}, 0)$

C chart UCL = $n_i\bar{u} + K\sqrt{n_i\bar{u}}$

The limits vary with n_i.

u_i is the number of nonconformities per unit in the i^{th} subgroup. In general, $u_i = c_i/n_i$.

K is the sigma multiplier and is set to 3 by default

c_i is the total number of nonconformities in the i^{th} subgroup

n_i is the number of inspection units in the i^{th} subgroup

$\bar{u}$ is the average number of nonconformities per unit taken across subgroups. The quantity $\bar{u}$ is computed as a weighted average

$$\bar{u} = \frac{n_1 u_1 + \ldots + n_N u_N}{n_1 + \ldots + n_N} = \frac{c_1 + \ldots + c_N}{n_1 + \ldots + n_N}$$

N is the number of subgroups

Levey-Jennings Charts

Levey-Jennings charts show a process mean with control limits based on a long-term sigma. The control limits are placed at K*s distance from the center line, where K = 3 by default.

The standard deviation, s, for the Levey-Jennings chart is calculated the same way standard deviation is in the Distribution platform.

$$s = \sqrt{\sum_{i=1}^{N} \frac{(y_i - \bar{y})^2}{N-1}}$$

See Levey and Jennings (1950); Westgard (2002).

Control Limits for G Charts

The negative binomial distribution is an extension of the geometric (Poisson) distribution and allows for over-dispersion relative to the Poisson. The negative binomial distribution can be used to construct both exact and approximate control limits for count data. Approximate control limits can be obtained based on a chi-square approximation to the negative binomial. All data is used as individual observations regardless of subgroup size.

Let X have a negative binomial distribution with parameters (μ, k). Then:

$$P(X \le r) \sim P\left(\chi^2_v \le \frac{2r+1}{1+\mu k}\right)$$

where:

χ^2_v is a chi-square variate with $v = 2\mu/(1+\mu k)$ degrees of freedom.

Based on this approximation, approximate upper and lower control limits can be determined. For a nominal level α Type 1 error probability in one direction, an approximate upper control limit is a limit UCL such that the following equation is true:

$$P(X > \text{UCL}) = 1 - P\left(\chi^2_v \le \frac{2\text{UCL}+1}{1+\mu k}\right) = \alpha$$

Likewise, an approximate lower control limit, LCL, is a limit such that the following equation is true:

$$P(X < \text{LCL}) = 1 - P\left(\chi^2_v \ge \frac{2\text{LCL}+1}{1+\mu k}\right) = \alpha$$

Thus, an approximate level lower and upper control limits, LCL and UCL, respectively, are computed using the following formulas:

$$\text{UCL} = \frac{\chi^2_{v,\,1-\alpha}(1+\mu k)-1}{2}$$

$$\text{LCL} = \max\left\{0, \frac{\chi^2_{v,\,\alpha}(1+\mu k)-1}{2}\right\}$$

where:

$\chi^2_{v,\,1-\alpha}(\chi^2_{v,\,\alpha})$ is the upper (lower) percentile of the chi-square distribution with $v = 2\mu/(1+\mu k)$ degrees of freedom. Negative lower control limits can be set to zero.

For more information about the negative binomial control limits, see Hoffman (2003).

Control Limits for T Charts

The estimates of the shape and scale parameters are calculated from the data and used to obtain the percentiles of the Weibull distribution.

Note: Subgroups with a response value of zero are given a weight of zero when estimating the Weibull distribution parameters.

Define the following quantities:

p1 = normalDist(-*K*) for Normal (0,1)

p2 = normalDist(0) for Normal (0,1)

p3 = normalDist(*K*) for Normal (0,1)

Then the limits are calculated using the following formulas:

LCL = Weibull Quantile (p1, β, α)

CL = Weibull Quantile (p2, β, α)

UCL = Weibull Quantile (p3, β, α)

where:

β is the shape parameter and α is the scale parameter for the Weibull Quantile function

K is the sigma multiplier and is set to 3 by default

For more information about the Weibull Quantile function, see **Help > Scripting Index**.

Sigma Calculations for Three Way Control Charts

Within Sigma Based on Average of Ranges

The within sigma estimate for three way control charts that is estimated using the average of ranges can be used for the Individual on Means, Moving Range on Means and R chart.

$$\hat{\sigma}_{\text{within}} = \frac{\frac{R_1}{d_2(n_1)} + \ldots + \frac{R_N}{d_2(n_N)}}{N}$$

The formula uses the following notation:

R_i = range of i^{th} subgroup

n_i = sample size of i^{th} subgroup

$d_2(n_i)$ = expected value of the range of n_i independent normally distributed variables with unit standard deviation

N = number of subgroups for which $n_i \geq 2$

Within Sigma Based on Average of Unbiased Standard Deviations

The within sigma estimate for three way control charts that is estimated using the average of unbiased standard deviations can be used for the Individual on Means, Moving Range on Means, and S chart.

$$\hat{\sigma}_{\text{within}} = \frac{\frac{s_1}{c_4(n_1)} + \ldots + \frac{s_N}{c_4(n_N)}}{N}$$

The formula uses the following notation:

s_i = sample standard deviation of the i^{th} subgroup

n_i = sample size of i^{th} subgroup

$c_4(n_i)$ = expected value of the standard deviation of n_i independent normally distributed variables with unit standard deviation

N = number of subgroups for which $n_i \geq 2$

Between Sigma

The between sigma estimate for three way control charts is estimated using the moving range of subgroup means.

$$\hat{\sigma}_{\text{between}} = \sqrt{\left(\frac{\overline{MR}}{d_2(2)}\right)^2 - \frac{\hat{\sigma}^2_{\text{within}}}{H}}$$

The formula uses the following notation:

$\overline{MR}$ = the mean of the nonmissing moving ranges computed as $(MR_2+MR_3+...+MR_N)/(N-1)$ where $MR_i = |y_i - y_{i-1}|$.

$d_2(2)$ = expected value of the range of two independent normally distributed variables with unit standard deviation.

$H = \dfrac{N}{\frac{1}{n_1} + \frac{1}{n_2} + \ldots + \frac{1}{n_N}}$, the harmonic mean of subgroup sample sizes.

Note: If between Sigma is estimated as a negative value, it is set to 0.

Between-and-Within Sigma

The between-and-within sigma estimate for three way control charts is estimated using a combination of the within sigma and between sigma estimates.

$$\hat{\sigma}_{\text{between-and-within}} = \sqrt{\hat{\sigma}^2_{\text{within}} + \hat{\sigma}^2_{\text{between}}}$$

Chapter 4

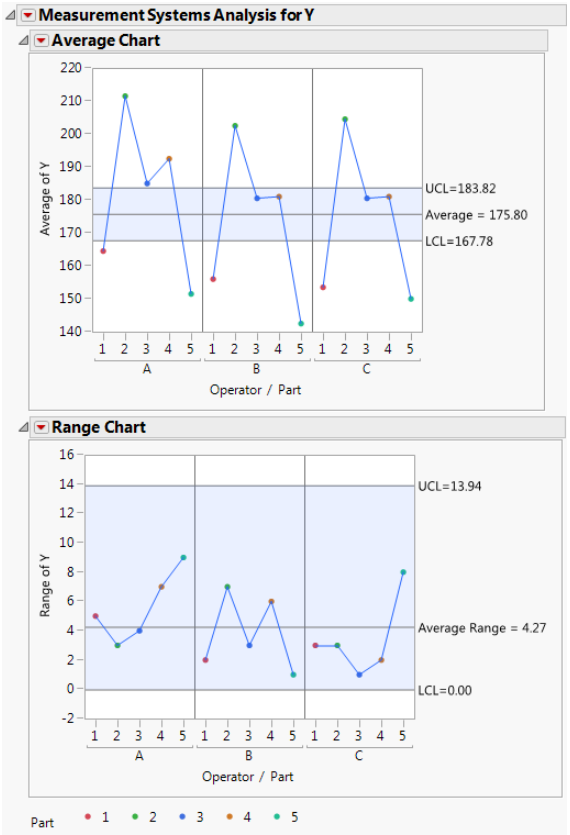
Measurement Systems Analysis

Evaluate a Continuous Measurement Process Using the EMP Method

The Measurement Systems Analysis (MSA) platform assesses the precision, consistency, and bias of a measurement system. Before you can study the process itself, you need to make sure that you can accurately and precisely measure the process. If most of the variation that you see comes from the measuring process itself, then you are not reliably learning about the process. Use MSA to find out how your measurement system is performing.

This chapter covers the EMP method. The Gauge R&R method is described in "Variability Gauge Charts".

Figure 4.1 Example of a Measurement System Analysis



Contents

Overview of Measurement Systems Analysis . . . 101
Example of Measurement Systems Analysis . . . 101
Launch the Measurement Systems Analysis Platform . . . 105
Data Format . . . 106
Measurement Systems Analysis Platform Options . . . 107
Average Chart . . . 108
Range Chart or Standard Deviation Chart . . . 109
EMP Results . . . 110
Effective Resolution . . . 111
Shift Detection Profiler . . . 112
Bias Comparison . . . 117
Test-Retest Error Comparison . . . 118
Additional Example of Measurement Systems Analysis . . . 118
Statistical Details for Measurement Systems Analysis . . . 124
Computation of Intraclass Correlation and Probable Error . . . 124

Overview of Measurement Systems Analysis

The EMP (Evaluating the Measurement Process) method in the Measurement Systems Analysis platform is largely based on the methods presented in Donald J. Wheeler's book *EMP III Using Imperfect Data* (2006). The EMP method provides visual information and results that are easy to interpret and helps you improve your measurement system to its full potential.

The Gauge R&R method analyzes how much of the variability is due to operator variation (reproducibility) and measurement variation (repeatability). Gauge R&R is available for many combinations of crossed and nested models, regardless of whether the model is balanced. See "Variability Gauge Charts".

Within the Six Sigma DMAIC methodology, MSA (Measurement System Analysis) addresses the Measure phase and process behavior charts (or control charts) address the Control phase. MSA helps you predict and characterize future outcomes. You can use the information gleaned from MSA to help you interpret and configure your process behavior charts.

For more information about Control Charts, see "Control Chart Builder".

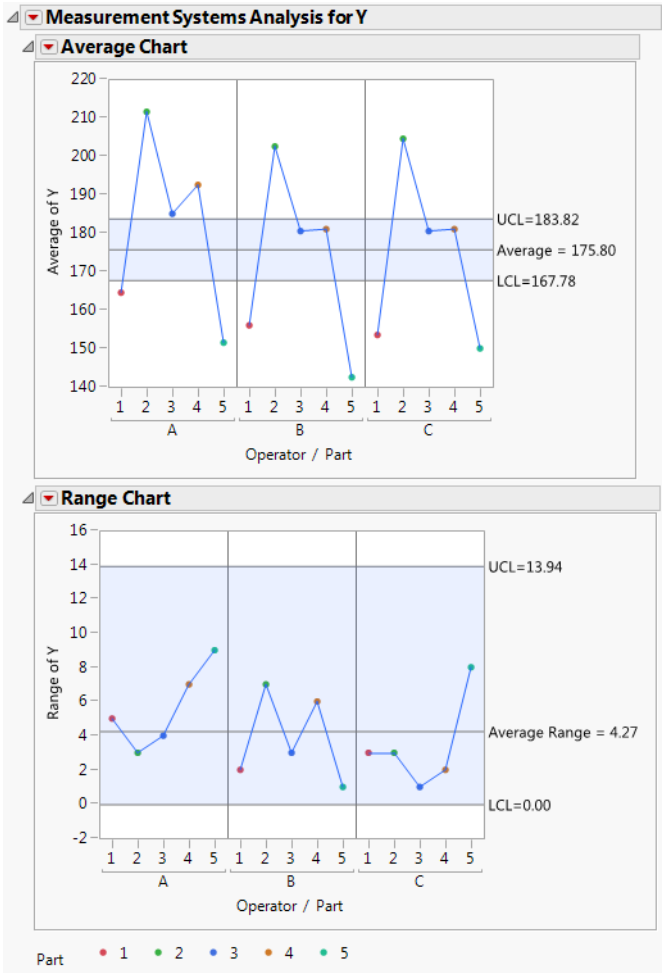
Example of Measurement Systems Analysis

In this example, three operators measured the same five parts. See how the measurement system is performing, based on how much variation is found in the measurements.

1. Select **Help > Sample Data Library** and open Variability Data/Gasket.jmp.
2. Select **Analyze > Quality and Process > Measurement Systems Analysis**.
3. Assign Y to the **Y, Response** role.
4. Assign Part to the **Part, Sample ID** role.
5. Assign Operator to the **X, Grouping** role.

   Notice that the **MSA Method** is set to **EMP**, the **Chart Dispersion Type** is set to **Range**, and the **Model Type** is set to **Crossed**.
6. Click **OK**.

Figure 4.2 MSA Initial Report



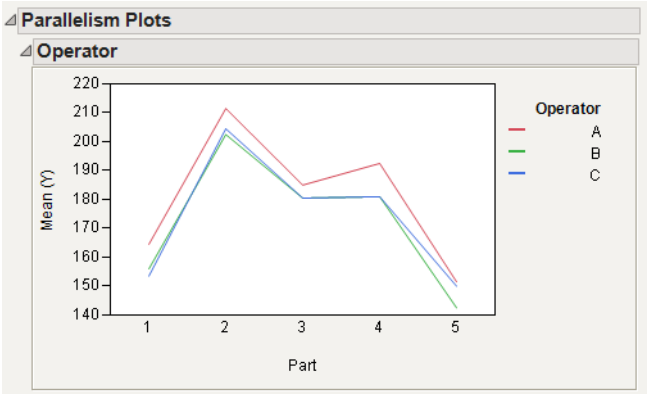
The Average Chart shows the average measurements for each operator and part combination. In this example, the means of the part measurements are generally beyond the control limits. This is a desirable outcome, because it indicates that you can detect part-to-part variation.

The Range Chart shows the variability for each operator and part combination. In this example, the ranges are within the control limits. This is a desirable outcome, because it indicates that the operators are measuring parts in the same way and with similar variation.

The color coding for each part is shown in the legend below the charts.

7. Click the red triangle next to Measurement Systems Analysis for Y and select **Parallelism Plots**.

Figure 4.3 Parallelism Plot for Operator and Part



The Parallelism Plots chart shows the average measurements for each part by operator. Because the lines are generally parallel and there is no major crossing, you conclude that there is no interaction between operators and parts.

Tip: Interactions indicate a serious issue that requires further investigation.

8. Click the red triangle next to Measurement Systems Analysis for Y and select **EMP Results**.

Figure 4.4 EMP Results Report

EMP Results

EMP Test	Results	Description
Test-Retest Error	3.7812	Within Error
Degrees of Freedom	13.39	Amount of information used to estimate within error
Probable Error	2.5504	Median error for a single measurement
Intraclass Correlation (no bias)	0.9745	Proportion of variation attributed to part variation without including bias factors
Intraclass Correlation (with bias)	0.9437	Proportion of variation attributed to part variation with bias factors
Bias Impact	0.0308	Amount by which the bias factors reduce the intraclass correlation

System	Classification
Current (with bias)	First Class
Potential (no bias)	First Class

Monitor Classification Legend

Classification	Intraclass Correlation	Attenuation of Process Signal	Probability of Warning, Test 1 Only*	Probability of Warning, Tests 1-4*
First Class	0.80 - 1.00	Less than 11%	0.99 - 1.00	1.00
Second Class	0.50 - 0.80	11% - 29%	0.88 - 0.99	1.00
Third Class	0.20 - 0.50	29% - 55%	0.40 - 0.88	0.92 - 1.00
Fourth Class	0.00 - 0.20	More than 55%	0.03 - 0.40	0.08 - 0.92

*** Probability of warning for a 3 standard error shift within 10 subgroups using Wheeler's tests, which correspond to Nelson's tests 1, 2, 5, and 6.**

The EMP Results report computes several statistics to help you assess and classify your measurement system. The Intraclass Correlation indicates the proportion of the total variation that you can attribute to the part.

From the EMP Results report, you can conclude the following:

- The Intraclass Correlation values are close to 1, indicating that most of the variation is coming from the part instead of the measurement system.
- The classification is First Class, meaning that the strength of the process signal is weakened by less than 11%.
- There is at least a 99% chance of detecting a warning using Test 1 only.
- There is 100% chance of detecting a warning using Tests 1-4.

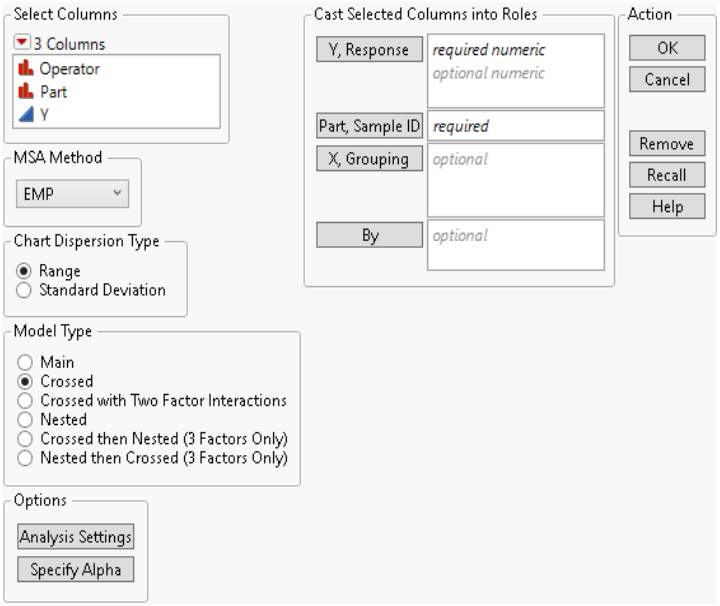
Note: For more information about tests and detecting process shifts, see "Shift Detection Profiler".

There is no interaction between operators and parts, and there is very little variation in your measurements (the classification is First Class). Therefore, you conclude that the measurement system is performing quite well.

Launch the Measurement Systems Analysis Platform

Launch the Measurement Systems Analysis platform by selecting **Analyze > Quality and Process > Measurement Systems Analysis**.

Figure 4.5 The Measurement Systems Analysis Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Measurement Systems Analysis window contains the following features:

Select Columns Lists all of the variables in your current data table. Move a selected column into a role.

MSA Method Select the method to use: **EMP** (Evaluating the Measurement Process) or **Gauge R&R**. This chapter covers the **EMP** method. For more information about the **Gauge R&R** method, see "Variability Gauge Charts".

Chart Dispersion Type Designates the type of chart for showing variation. Select the **Range** option or the **Standard Deviation** option.

Note: For the **EMP** method, the chart dispersion type determines how the statistics in the EMP Results report are calculated. If the **Range** option is selected, and you have a one factor or a two factor, balanced, crossed model, the statistics in this report are based on ranges. Otherwise, the statistics in this report are based on standard deviations.

Model Type Designates the model type:

Main Variables with nominal or ordinal modeling types are treated as main effects.

Crossed The model is crossed when every level of every factor occurs with every level of every other factor.

Crossed with Two Factor Interactions The model is crossed when each level of two factors occurs with every level of the other factor.

Nested The model is nested when all levels of a factor appear within only a single level of any other factor.

Cross then Nested (3 Factors Only) The factors are crossed and then nested for 3 factors.

Nested then Crossed (3 Factors Only) The factors are nested and then crossed for 3 factors.

Options Contains the following options:

Analysis Settings Sets the REML maximum iterations and convergence.

Specify Alpha Specifies the 1-alpha confidence level.

Y, Response The column of measurements.

Part, Sample, ID The column designating the part or unit.

X, Grouping The column(s) representing grouping variables.

By Identifies a column that creates a report consisting of separate analyses for each level of the variable.

Data Format

To use the Measurement Systems Analysis platform, all response measurements must be in a single response column. Sometimes, responses are recorded in multiple columns, where each row is a level of a design factor and each column is a level of a different design factor. Data that are in this format must be stacked before running the Measurement Systems Analysis platform. See *Using JMP*.

Measurement Systems Analysis Platform Options

Platform options appear within the red triangle menu next to Measurement Systems Analysis. Selecting an option creates the respective graph or report in the MSA report window. Deselecting an option removes the graph or report. Choose from the following options:

Average Chart A plot of the average measurement values for each combination of the part and *X* variables. The Average Chart helps you detect product variation despite measurement variation. In an Average Chart, out of control data is desirable because it detects part-to-part variation. See "Average Chart".

Range Chart A plot of the variability statistic for each combination of the part and *X* variables. Appears only if you selected **Range** as the Chart Dispersion Type in the launch window. The Range Chart helps you check for consistency within subgroups. In a Range Chart, data within limits is desirable, indicating homogeneity in your error. See "Range Chart or Standard Deviation Chart".

Std Dev Chart A plot of the standard deviation statistic for each combination of the part and *X* variables. Appears only if you selected **Standard Deviation** as the Chart Dispersion Type in the launch window. The Standard Deviation Chart helps you check for consistency within subgroups. In a Standard Deviation Chart, data within limits is desirable, indicating homogeneity in your error. See "Range Chart or Standard Deviation Chart".

Parallelism Plots An overlay plot that reflects the average measurement values for each part. If the lines are relatively not parallel or crossing, there might be an interaction between the part and *X* variables.

Tip: Interactions indicate a serious issue that requires further investigation. For example, interactions between parts and operators mean that operators are measuring different parts differently, on average. Therefore, measurement variability is not predictable. This issue requires further investigation to find out why the operators do not have the same pattern or profile over the parts.

EMP Results A report that computes several statistics to help you assess and classify your measurement system. See "EMP Results".

Effective Resolution A report containing results for the resolution of a measurement system. See "Effective Resolution".

Bias Comparison An Analysis of Means chart for testing if the *X* variables have different averages. See "Bias Comparison".

Test-Retest Error Comparison An Analysis of Means for Variances or Analysis of Means Ranges chart for testing if any of the groups have different test-retest error levels. See "Test-Retest Error Comparison".

Shift Detection Profiler An interactive set of charts that you can adjust to see the probabilities of getting warnings on your process behavior chart. See "Shift Detection Profiler".

Variance Components A report containing the estimates of the variance components for the given model. The calculations in this report are based on variances, not ranges. Balanced data uses the EMS method. Unbalanced data uses the REML method.

Note: This report is similar to the Variance Components report in the Variability Chart platform, except that it does not compute Bayesian variance component estimates. See "Variance Components".

EMP Gauge R&R Results A report that partitions the variability in the measurements into part variation and measurement system variation. The calculations in this report are based on variances, not ranges. Because negative variance components are set to zero, values of zero could indicate outliers in your results.

Note: This report is similar to the Gauge R&R report in the Variability Chart platform. However, by default, the calculation for Reproducibility does not include interactions. To specify that interactions be included in the Reproducibility calculation, select the Include Interactions in Reproducibility platform preference. This preference is located in File > Preferences > Platforms > EMP Measurement Systems Analysis. For more information about Gauge R&R studies, see "About the Gauge R&R Method".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Average Chart

The red triangle menu next to Average Chart contains the following options:

Show Grand Mean Draws the overall mean of the *Y* variable on the chart.

Show Connected Means Draws lines connecting all of the average measurement values.

Show Control Limits Draws lines representing the Upper Control Limit (UCL) and the Lower Control Limit (LCL) and labels those values. The control limits for the Average Chart use the same calculations as an XBar control chart. See "Control Limits for XBar and R Charts".

Show Control Limits Shading Adds shading between the UCL and LCL.

Show Separators Draws vertical lines to delineate between the *X* variables.

Show Data Adds the data points to the chart.

Note: You can replace variables in the Average Chart in one of two ways: swap existing variables by dragging and dropping a variable from one axis to the other axis; or, click a variable in the Columns panel of the associated data table and drag it onto an axis.

Range Chart or Standard Deviation Chart

The red triangle menu next to Range Chart or Standard Deviation Chart contains the following options:

Show Average Dispersion Draws the average range or standard deviation on the chart.

Show Connected Points Draws lines connecting all of the ranges or standard deviations.

Show Control Limits Draws lines representing the Upper Control Limit (UCL) and the Lower Control Limit (LCL) and labels those values. For more information about the calculations of the limits used in the Range Chart, see "Control Limits for XBar and R Charts". For more information about the calculations of the limits used in the Standard Deviation Chart, see "Control Limits for XBar and S Charts".

Show Control Limits Shading Adds shading between the UCL and LCL.

Show Separators Draws vertical lines to delineate between the *X* variables.

Note: You can replace variables in the Range or Standard Deviation Charts in one of two ways: swap existing variables by dragging and dropping a variable from one axis to the other axis; or, click a variable in the Columns panel of the associated data table and drag it onto an axis.

EMP Results

Note: The statistics in this report are based on ranges in the following instances: if you selected **EMP** as the **MSA Method** and **Range** as the **Chart Dispersion Type**, and you have a one factor or a two factor, balanced, crossed model. Otherwise, the statistics in this report are based on variances.

The EMP Results report computes several statistics to help you assess and classify your measurement system. Using this report, you can determine the following:

- How your process chart is affected.
- Which tests to set.
- How much the process signal is attenuated.
- How much the bias factors are affecting your system and reducing your potential intraclass correlation coefficient.

The EMP Results report contains the following calculations:

Test-Retest Error Indicates measurement variation or repeatability (also known as within error or pure error).

Degrees of Freedom Indicates the amount of information used to estimate the within error.

Probable Error The median error for a single measurement. Indicates the resolution quality of your measurement and helps you decide how many digits to use when recording measurements. See "Effective Resolution".

Intraclass Correlation Indicates the proportion of the total variation that you can attribute to the part. If you have very little measurement variation, this number is closer to 1.

Intraclass Correlation (no bias) Does not take bias or interaction factors into account when calculating the results.

Intraclass Correlation (with bias) Takes the bias factors (such as operator, instrument, and so on) into account when calculating the results.

Intraclass Correlation (with bias and interaction) Takes the bias and interaction factors into account when calculating the results. This calculation appears only if the model is crossed and uses standard deviation instead of range.

Bias Impact The amount by which the bias factors reduce the Intraclass Correlation.

Bias and Interaction Impact The amount by which the bias and interaction factors reduce the Intraclass Correlation. This calculation appears only if the model is crossed and uses standard deviation instead of range.

Classes of Process Monitors

In order to understand the System and Classification parameters, you must first understand the Monitor Classification Legend.

Figure 4.6 Monitor Classification Legend

Monitor Classification Legend

Classification	Intraclass Correlation	Attenuation of Process Signal	Probability of Warning, Test 1 Only*	Probability of Warning, Tests 1-4*
First Class	0.80 - 1.00	Less than 11%	0.99 - 1.00	1.00
Second Class	0.50 - 0.80	11% - 29%	0.88 - 0.99	1.00
Third Class	0.20 - 0.50	29% - 55%	0.40 - 0.88	0.92 - 1.00
Fourth Class	0.00 - 0.20	More than 55%	0.03 - 0.40	0.08 - 0.92

*** Probability of warning for a 3 standard error shift within 10 subgroups using Wheeler's tests, which correspond to Nelson's tests 1, 2, 5, and 6.**

This legend describes the following classifications: First, Second, Third, and Fourth Class. Each classification indicates the following:

- the corresponding Intraclass Correlation values
- the amount of process signal attenuation (decrease)
- the chance of detecting a 3 standard error shift within 10 subgroups, using Wheeler's test one or all four tests

Wheeler (2006) identifies four detection tests known as the Western Electric Zone Tests. Within the Shift Detection Profiler, there are eight tests that you can select from. The tests that correspond to the Wheeler tests are the first, second, fifth, and sixth tests.

Tip: To prevent the legend from appearing, deselect **Show Monitor Classification Legend** in the EMP Measurement Systems Analysis platform preferences.

Effective Resolution

The Effective Resolution report helps you determine how well your measurement increments are working. You might find that you need to add or drop digits when recording your measurements, or your current increments might be effective as is. Note the following:

- The Probable Error calculates the median error of a measurement.
- The Current Measurement Increment reflects how many digits you are currently rounding to and is taken from the data as the nearest power of ten. This number is compared to the Smallest Effective Increment, Lower Bound Increment, and Largest Effective Increment. Based on that comparison, a recommendation is made.

- Large measurement increments have less uncertainty in the last digit, but large median errors. Small measurement increments have small median errors, but more uncertainty in the last digit.

Shift Detection Profiler

Use the Shift Detection Profiler to assess the sensitivity of the control chart that you use to monitor your process. The Shift Detection Profiler estimates the probability of detecting shifts in the product mean or product standard deviation. The control chart limits include sources of measurement error variation. Based on these limits, the Shift Detection Profiler estimates the Probability of Warning. This is the probability that a control chart monitoring the process mean signals a warning over the next *k* subgroups.

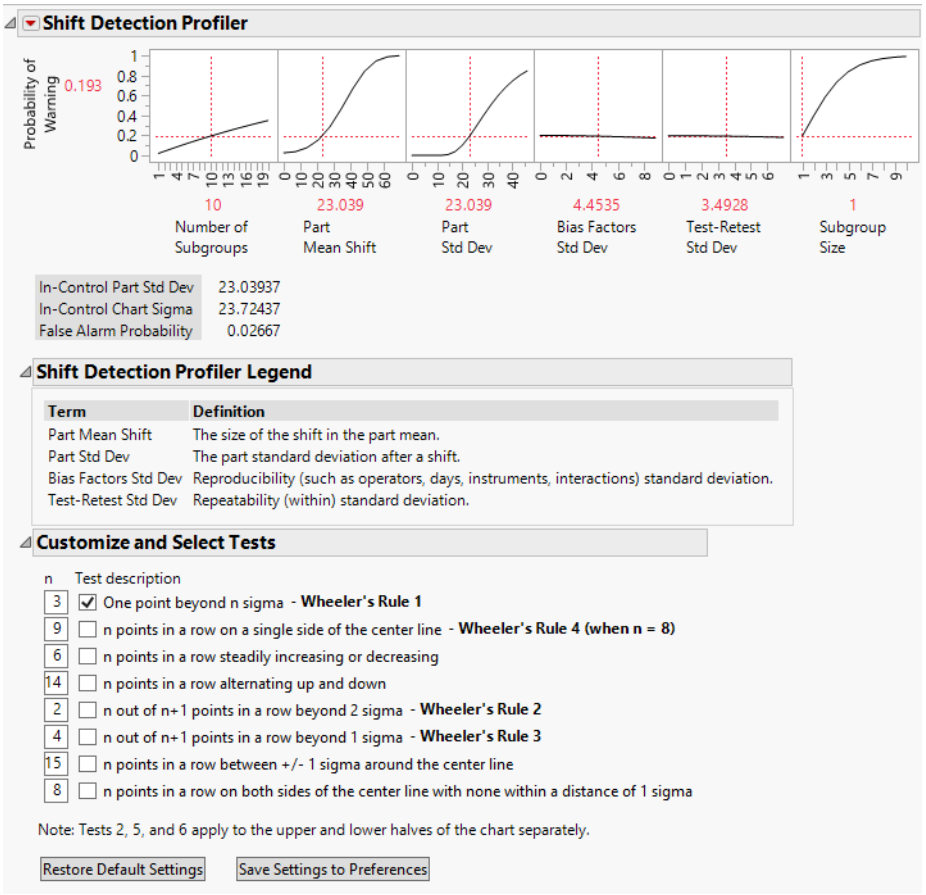
You can set the subgroup size that you want to use for your control chart. Note the following:

- If the Subgroup Size equals one, the control chart is an Individual Measurement chart.
- If the Subgroup Size exceeds one, the control chart is an *X*Bar-chart.

You can explore the effect of Subgroup Size on the control chart's sensitivity. You can also explore the benefits of reducing bias and test-retest error.

Figure 4.7 shows the Shift Detection Profiler report for the Gasket.jmp sample data table, found in the Variability Data folder.

Figure 4.7 Shift Detection Profiler for Gasket.jmp



Probability of Warning

The Probability of Warning is the probability of detecting a change in the process. A change is defined by the Part Mean Shift and the Part Std Dev settings in the Shift Detection Profiler. The probability calculation assumes that the tests selected in the Customize and Select Tests outline are applied to the Number of Subgroups specified in the Profiler.

The control limits for the Individual Measurement chart (Subgroup Size = 1) and the XBar-chart (Subgroup Size > 1) are based on the In-Control Chart Sigma. The In-Control Sigma takes into account the bias factor (reproducibility) variation and the test-retest (repeatability) variation. These are initially set to the values obtained from your MSA study. The In-Control Chart Sigma also incorporates the In-Control Part Std Dev. Both of these values appear beneath the profiler, along with the False Alarm Probability, which is based on the In-Control Chart Sigma.

In-Control Part Std Dev The standard deviation for the true part values, exclusive of measurement errors, for the stable process. The default value for In-Control Part Std Dev is the standard deviation of the part component estimated by the MSA analysis and found in the Variance Components report.

Often, parts for an MSA study are chosen to have specific properties and do not necessarily reflect the part-to-part variation seen in production. For this reason, you can specify the in-control part standard deviation by selecting **Change In-Control Part Std Dev** from the Shift Detection Profiler red triangle menu.

In-Control Chart Sigma The value of sigma used to compute control limits. This value is computed using the In-Control Part Std Dev, the Bias Factors Std Dev, and Test-Retest Std Dev specified in the Shift Detection Profiler, and the Subgroup Size. The reproducibility factors are assumed to be constant within a subgroup.

For a subgroup of size n, control limits are set at the following values:

$$\pm 3(\text{In-Control Chart Sigma})/(\sqrt{n})$$

It follows that the In-Control Chart Sigma is the square root of the sum of the squares of the following terms:

- In-Control Part Std Dev
- Bias Factors Std Dev, as specified in the Shift Detection Profiler, multiplied by $\sqrt{n}$
- Test-Retest Std Dev, as specified in the Shift Detection Profiler

The Bias Factors Std Dev is multiplied by $\sqrt{n}$ to account for the assumption that the reproducibility factors are constant within a subgroup.

JMP updates the In-Control Chart Sigma when you change the In-Control Part Std Dev, the Bias Factors Std Dev, the Test-Retest Std Dev, or the Subgroup Size.

False Alarm Probability The probability that the control chart tests signal a warning when no change in the part mean or standard deviation has occurred. JMP updates the False Alarm Probability when you change the Number of Subgroups or the tests in Customize and Select Tests.

For more information about the Variance Components report, see "Variance Components".

Shift Detection Profiler Settings

Number of Subgroups The number of subgroups over which the probability of a warning is computed. If the number of subgroups is set to k, the profiler gives the probability that the control chart signals at least one warning based on these k subgroups. The Number of Subgroups is set to 10 by default. Drag the vertical line in the plot to change the Number of Subgroups.

Part Mean Shift The shift in the part mean. By default, the profiler is set to detect a 1 sigma shift. The initial value is the standard deviation of the part component estimated by the MSA analysis and found in the Variance Components report. Drag the vertical line in the plot or click the value beneath the plot to change the Part Mean Shift.

Part Std Dev The standard deviation for the true part values, exclusive of measurement errors. The initial value for Part Std Dev is the standard deviation of the part component estimated by the MSA analysis and is found in the Variance Components report. Drag the vertical line in the plot or click the value beneath the plot to change the Part Std Dev.

Bias Factors Std Dev The standard deviation of factors related to reproducibility. Bias factors include operator and instrument. The bias factor variation does not include part and repeatability (within) variation. The initial value is derived using the reproducibility and interaction variance components estimated by the MSA analysis and is found in the Variance Components report. Drag the vertical line in the plot or click the value beneath the plot to change the Bias Factors Std Dev.

Test-Retest Std Dev The standard deviation of the test-retest, or repeatability, variation in the model. The initial value is the standard deviation of the Within component estimated by the MSA analysis and is found in the Variance Components report. Drag the vertical line in the plot or click the value beneath the plot to change the Test-Retest Std Dev.

Subgroup Size The sample size used for each subgroup. This is set to 1 by default. You can increase the sample size to investigate improvement in control chart performance. Increasing the sample size from 1 demonstrates what happens when you move from an Individual Measurement chart to an *X*Bar-chart. Drag the vertical line in the plot to change the Subgroup Size.

Shift Detection Profiler Options

The red triangle menu for the Shift Detection Profiler provides several options. Only one option is described here.

Change In-Control Part Std Dev Specify a value for the part standard deviation for the stable process. The in-control part standard deviation should reflect the variation of the true part values, exclusive of measurement errors. Enter a new value and click OK.

The In-Control Part Std Dev is originally set to the standard deviation of the part component estimated by the MSA analysis, found in the Variance Components report.

This option is useful if the parts chosen for the EMP study were not a random sample from the process.

Reset Factor Grid Displays a window for each factor allowing you to enter a specific value for the factor's current setting, to lock that setting, and to control aspects of the grid. See *Profilers*.

Factor Settings Submenu that consists of the following options:

Remember Settings Adds an outline node to the report that accumulates the values of the current settings each time the Remember Settings command is invoked. Each remembered setting is preceded by a radio button that is used to reset to those settings. There are options to remove selected settings or all settings in the Remember Settings red triangle menu.

Copy Settings Script Copies the current Profiler's settings to the clipboard.

Paste Settings Script Pastes the Profiler settings from the clipboard to a Profiler in another report.

Set Script Sets a script that is called each time a factor changes. The set script receives a list of arguments of the form:

```
{factor1 = n1, factor2 = n2, ...}
```

For example, to write this list to the log, first define a function:

```
ProfileCallbackLog = Function({arg},show(arg));
```

Then enter `ProfileCallbackLog` in the Set Script dialog.

Similar functions convert the factor values to global values:

```
ProfileCallbackAssign = Function({arg},evalList(arg));
```

Or access the values one at a time:

```
 ProfileCallbackAccess =
Function({arg},f1=arg["factor1"];f2=arg["factor2"]);
```

Shift Detection Profiler Legend

This panel gives a brief description of four of the Shift Detection Profiler settings. See "Shift Detection Profiler Settings".

Tip: To prevent the legend from appearing, deselect **Show Shift Detection Profiler Legend** in the EMP Measurement Systems Analysis platform preferences.

Customize and Select Tests

In the Customize and Select Tests panel, select and customize the tests that you want to apply to the *k* subgroups in your control chart. The eight tests are based on Nelson (1984). For more information about the tests, see "Tests".

The Shift Detection Profiler calculations take these tests into account. The Probability of Warning and False Alarm Probability values increase as you add more tests. Because the calculations are based on a quasi-random simulation, there might be a slight delay as the profiler is updated.

The Customize and Select Tests panel has the following options:

Restore Default Settings If no settings have been saved to preferences, this option resets the selected tests to the first test only. The values of *n* are also reset to the values described in "Tests". If settings have been saved to preferences, this option resets the selected tests and the values of *n* to those specified in the preferences.

Note: You can access preferences for control chart tests by selecting **File** > **Preferences** > **Platforms** > **Control Chart Builder**. Custom Tests 1 through 8 correspond to the eight tests shown in Customize and Select Tests.

Save Settings to Preferences Saves the selected tests and the values of *n* for use in future analyses. These preferences are added to the Control Chart Builder platform preferences.

Bias Comparison

The **Bias Comparison** option creates an Analysis of Means chart. This chart shows the mean values for each level of the grouping variables and compares them with the overall mean. You can use this chart to see whether an operator is measuring parts too high or too low, on average.

The red triangle menu next to Analysis of Means contains the following options:

Set Alpha Level Select an option from the most common alpha levels or specify any level using the **Other** selection. Changing the alpha level modifies the upper and lower decision limits.

Show Summary Report Shows a report containing group means and decision limits, and reports if the group mean is above the upper decision limit or below the lower decision limit.

Display Options Include the following options:

Show Decision Limits Draws lines representing the Upper Decision Limit (UDL) and the Lower Decision Limit (LDL) and defines those values.

Show Decision Limit Shading Adds shading between the UDL and the LDL.

Show Center Line Draws the center line statistic that represents the average.

Point Options Changes the chart display to needles, connected points, or points.

Test-Retest Error Comparison

The **Test-Retest Error Comparison** option creates a type of Analysis of Means for Variances or Analysis of Means Ranges chart. This chart shows if there are differences in the test-retest error between operators. For example, you can use this chart to see whether there is an inconsistency in how each operator is measuring. The Analysis of Mean Ranges chart is displayed when ranges are used for variance components.

- For information about the options in the red triangle menu next to Operator Variance Test, see "Bias Comparison".
- For more information about Analysis of Means for Variances charts, see "Variance Components".

Additional Example of Measurement Systems Analysis

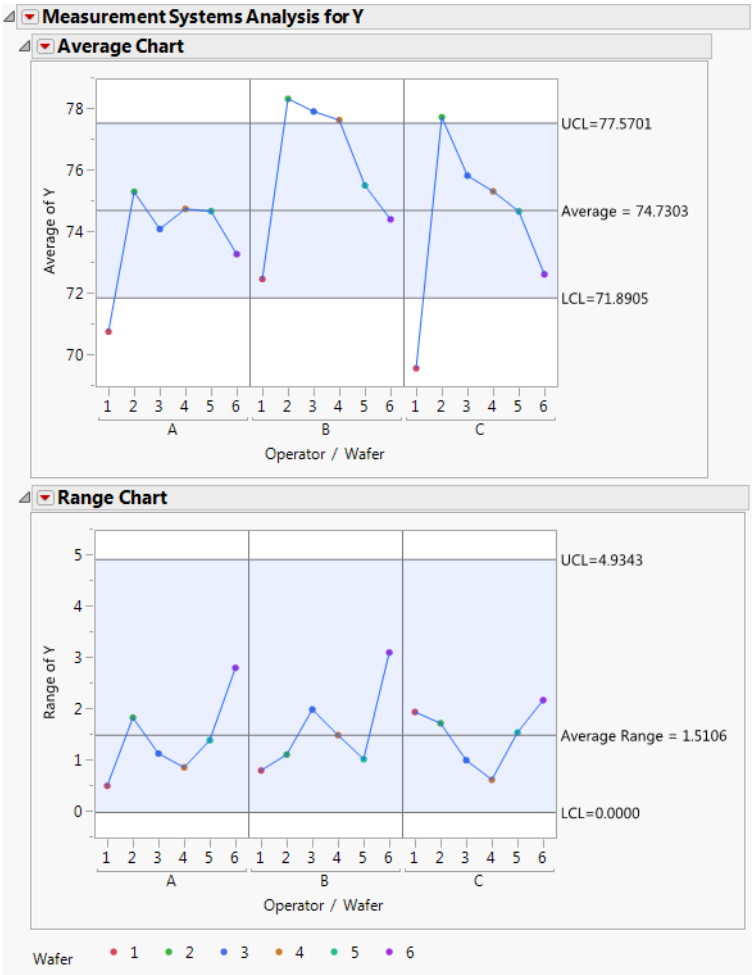
In this example, three operators have measured a single characteristic twice on each of six wafers. Perform a detailed analysis to find out how well the measurement system is performing.

Perform the Initial Analysis

1. Select **Help > Sample Data Library** and open Variability Data/Wafer.jmp.
2. Select **Analyze > Quality and Process > Measurement Systems Analysis**.
3. Assign Y to the **Y, Response** role.
4. Assign Wafer to the **Part, Sample ID** role.
5. Assign Operator to the **X, Grouping** role.

   Notice that the **MSA Method** is set to **EMP**, the **Chart Dispersion Type** is set to **Range**, and the **Model Type** is set to **Crossed**.
6. Click **OK**.

Figure 4.8 Average and Range Charts



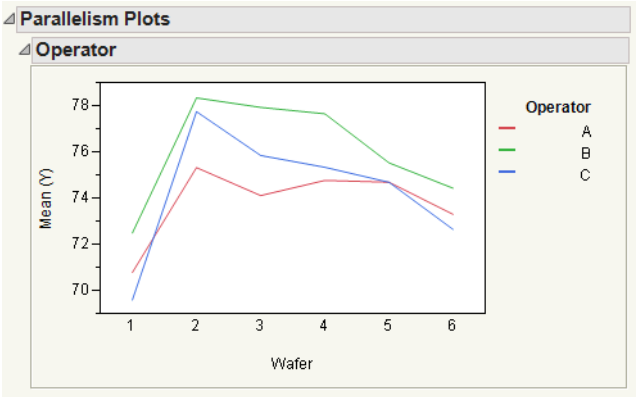
The Average Chart shows that some of the average part measurements fall beyond the control limits. This is desirable, indicating measurable part-to-part variation.

The Range Chart shows no points that fall beyond the control limits. This is desirable, indicating that the operator measurements are consistent within part.

Examine Interactions

Take a closer look for interactions between operators and parts. Click the red triangle next to Measurement Systems Analysis for Y and select **Parallelism Plots**.

Figure 4.9 Parallelism Plot

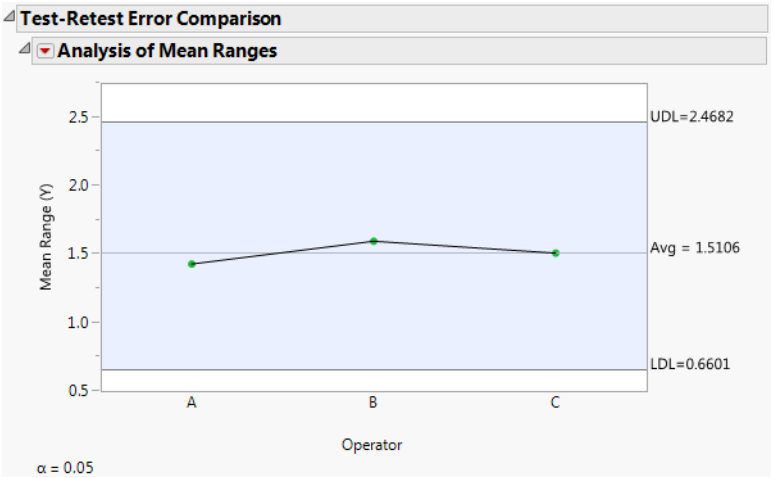


Looking at the parallelism plot by operator, you can see that the lines are relatively parallel and that there is only some minor crossing.

Examine Operator Consistency

Take a closer look at the variance between operators. Click the red triangle next to Measurement Systems Analysis for Y and select **Test-Retest Error Comparison**.

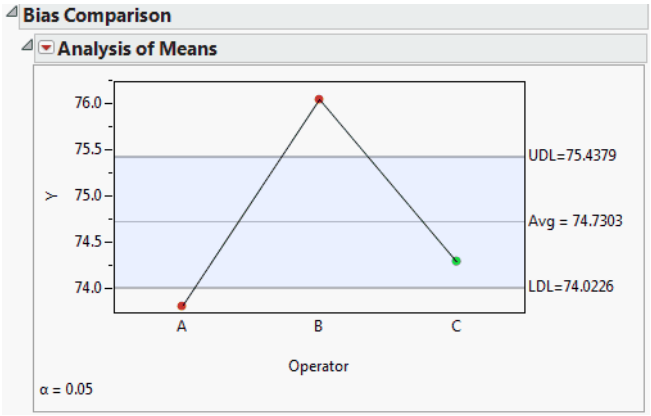
Figure 4.10 Test-Retest Error Comparison



Looking at the Test-Retest Error Comparison, you can see that none of the operators have a test-retest error that is significantly different from the overall test-retest error. The operators appear to be measuring consistently.

Just to be sure, you decide to look at the Bias Comparison chart, which indicates whether an operator is measuring parts too high or too low. Click the red triangle next to Measurement Systems Analysis for Y and select **Bias Comparison**.

Figure 4.11 Bias Comparison



Looking at the Bias Comparison chart, you make the following observations:

- Operator A and Operator B have detectable measurement bias, as they are significantly different from the overall average.
- Operator A is significantly biased low.
- Operator B is significantly biased high.
- Operator C is not significantly different from the overall average.

Classify Your Measurement System

Examine the EMP Results report to classify your measurement system and look for opportunities for improvement. Click the red triangle next to Measurement Systems Analysis for Y and select **EMP Results**.

Figure 4.12 EMP Results

EMP Results

EMP Test	Results	Description
Test-Retest Error	1.3387	Within Error
Degrees of Freedom	16.017	Amount of information used to estimate within error
Probable Error	0.9029	Median error for a single measurement
Intraclass Correlation (no bias)	0.7385	Proportion of variation attributed to part variation without including bias factors
Intraclass Correlation (with bias)	0.6272	Proportion of variation attributed to part variation with bias factors
Bias Impact	0.1113	Amount by which the bias factors reduce the intraclass correlation

System	Classification
Current (with bias)	Second Class
Potential (no bias)	Second Class

Monitor Classification Legend

Classification	Intraclass Correlation	Attenuation of Process Signal	Probability of Warning, Test 1 Only*	Probability of Warning, Tests 1-4*
First Class	0.80 - 1.00	Less than 11%	0.99 - 1.00	1.00
Second Class	0.50 - 0.80	11% - 29%	0.88 - 0.99	1.00
Third Class	0.20 - 0.50	29% - 55%	0.40 - 0.88	0.92 - 1.00
Fourth Class	0.00 - 0.20	More than 55%	0.03 - 0.40	0.08 - 0.92

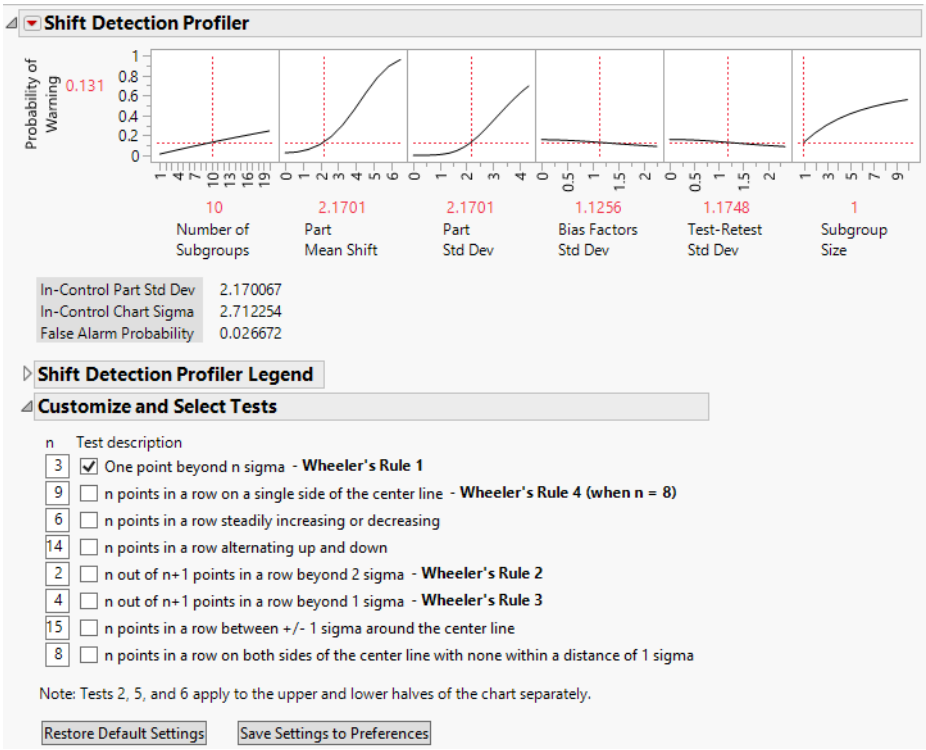
*** Probability of warning for a 3 standard error shift within 10 subgroups using Wheeler's tests, which correspond to Nelson's tests 1, 2, 5, and 6.**

The classification is Second Class, which means that there is a better than 88% chance of detecting a three standard error shift within ten subgroups, using Test one only. You notice that the bias factors have an 11% impact on the Intraclass Correlation. In other words, if you could eliminate the bias factors, your Intraclass Correlation coefficient would improve by 11%.

Explore the Ability of a Control Chart to Detect Process Changes

Use the Shift Detection Profiler to explore the probability that a control chart will be able to detect a change in your process. Click the red triangle next to Measurement Systems Analysis for Y and select **Shift Detection Profiler**.

Figure 4.13 Shift Detection Profiler



By default, the only test selected is for a point beyond the 3 sigma limits. Also note that the default Subgroup Size is 1, indicating that you are using an Individual Measurement chart.

Explore your ability to detect a shift in the mean of two part standard deviations in the 10 subgroups following the shift. Click the **Part Mean Shift** value of 2.1701 and change it to 4.34 (2.17 multiplied by 2). The probability of detecting a shift of twice the part standard deviation is 56.9%.

Next, see how eliminating bias affects your ability to detect the shift of two part standard deviations. Change the **Bias Factors Std Dev** value from 1.1256 to 0. The probability of detecting the shift increases to 67.8%.

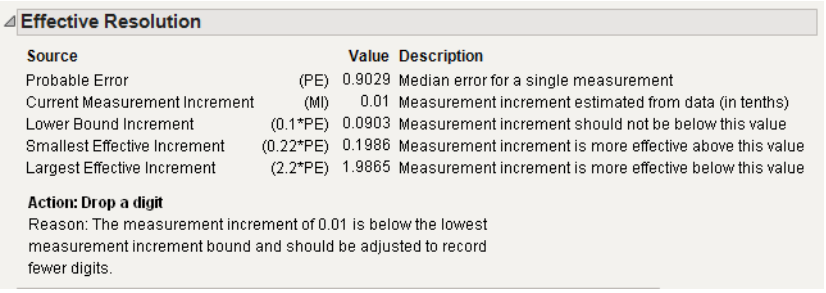
Finally, add more tests to see how your ability to detect the two part standard deviation shift changes. In addition to the first test, select the second, fifth, and sixth tests (Wheeler's Rules 4, 2, and 3). With these four tests and no bias variation, your probability of detecting the shift is 99.9%.

You can also explore the effect of using a control chart based on larger subgroup sizes. For subgroup sizes of two or more, the control chart is an *X*Bar-chart. Change the **Bias Factors Std Dev** value back to 1.1256 and deselect all but the first test. Set the **Subgroup Size** in the profiler to 4. The probability of detecting the two part standard deviation shift is 98.5%.

Examine Measurement Increments

Finally, see how well your measurement increments are working. Click the red triangle next to Measurement Systems Analysis for Y and select **Effective Resolution**.

Figure 4.14 Effective Resolution



Effective Resolution

Source		Value	Description
Probable Error	(PE)	0.9029	Median error for a single measurement
Current Measurement Increment	(MI)	0.01	Measurement increment estimated from data (in tenths)
Lower Bound Increment	(0.1*PE)	0.0903	Measurement increment should not be below this value
Smallest Effective Increment	(0.22*PE)	0.1986	Measurement increment is more effective above this value
Largest Effective Increment	(2.2*PE)	1.9865	Measurement increment is more effective below this value

Action: Drop a digit
Reason: The measurement increment of 0.01 is below the lowest measurement increment bound and should be adjusted to record fewer digits.

The Current Measurement Increment of 0.01 is below the Lower Bound Increment of 0.09, indicating that you should adjust your future measurements to record one less digit.

Statistical Details for Measurement Systems Analysis

For more information about the calculations of the limits used in the Range Chart, see "Control Limits for XBar and R Charts". For more information about the calculations of the limits used in the Standard Deviation Chart, see "Control Limits for XBar and S Charts".

Computation of Intraclass Correlation and Probable Error

Intraclass Correlation without bias is computed as follows:

$$r_{pe} = \frac{\hat{\sigma}_p^2}{\hat{\sigma}_p^2 + \hat{\sigma}_{pe}^2}$$

Intraclass Correlation with bias is computed as follows:

$$r_b = \frac{\hat{\sigma}_p^2}{\hat{\sigma}_p^2 + \hat{\sigma}_b^2 + \hat{\sigma}_{pe}^2}$$

Intraclass Correlation with bias and interaction factors is computed as follows:

$$r_{int} = \frac{\hat{\sigma}_p^2}{\hat{\sigma}_p^2 + \hat{\sigma}_b^2 + \hat{\sigma}_{int}^2 + \hat{\sigma}_{pe}^2}$$

Probable Error is computed as follows:

$$Z_{0.75} \times \hat{\sigma}_{pe}$$

Note the following:

$\hat{\sigma}_{pe}^2$ = variance estimate for pure error

$\hat{\sigma}_p^2$ = variance estimate for product

$\hat{\sigma}_b^2$ = variance estimate for bias factors

$\hat{\sigma}_{int}^2$ = variance estimate for interaction factors

$Z_{0.75}$ = the 75% quantile of standard normal distribution

Chapter 5

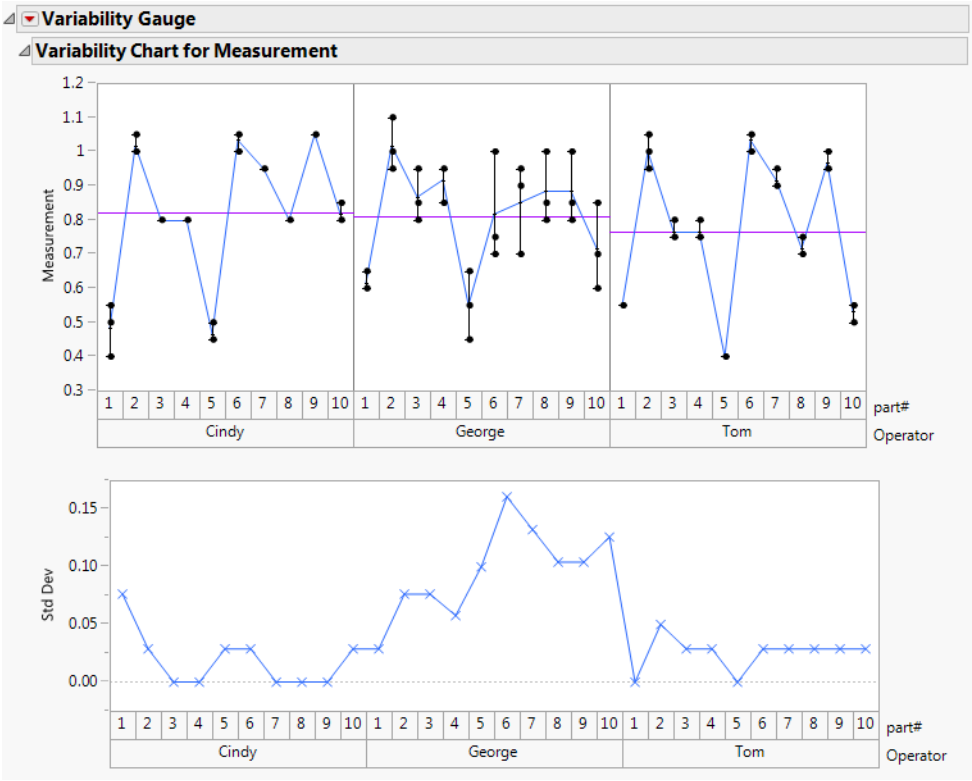
Variability Gauge Charts

Evaluate a Continuous Measurement Process Using Gauge R&R

Variability gauge charts analyze continuous measurements and can reveal how your measurement system is performing. You can also perform a gauge study to see measures of variation in your data.

Tip: This chapter covers only variability charts. For more information about attribute charts, see "Attribute Gauge Charts".

Figure 5.1 Example of a Variability Chart



Contents

Overview of Variability Charts 129
Example of a Variability Chart 130
Launch the Variability/Attribute Gauge Chart Platform 131
Data Format 132
The Variability Gauge Chart 132
Variability Gauge Platform Options 134
Heterogeneity of Variance Tests 136
Variance Components 137
About the Gauge R&R Method 139
Gauge R&R Option 140
Discrimination Ratio 142
Misclassification Probabilities 142
Bias Report 143
Linearity Study 144
Additional Examples of Variability Charts 144
Example of the Heterogeneity of Variance Test 145
Example of the Bias Report Option 147
Statistical Details for Variability Charts 150
Statistical Details for Variance Components 150
Statistical Details for the Discrimination Ratio 151
Statistical Details for the Misclassification Probabilities 152

Overview of Variability Charts

Tip: The traditional name for a variability chart is a *multi vari* chart, but because that name is not well known, the more generic term variability chart is used instead.

Just as a control chart shows variation across time in a process, a variability chart shows the same type of variation across categories such as parts, operators, repetitions, and instruments. A variability chart plots the data and means for each level of grouping factors, with all plots side by side. Along with the data, you can view the mean, range, and standard deviation of the data in each category, seeing how they change across the categories. The report options are based on the assumption that the primary interest is how the mean and variance change across the categories.

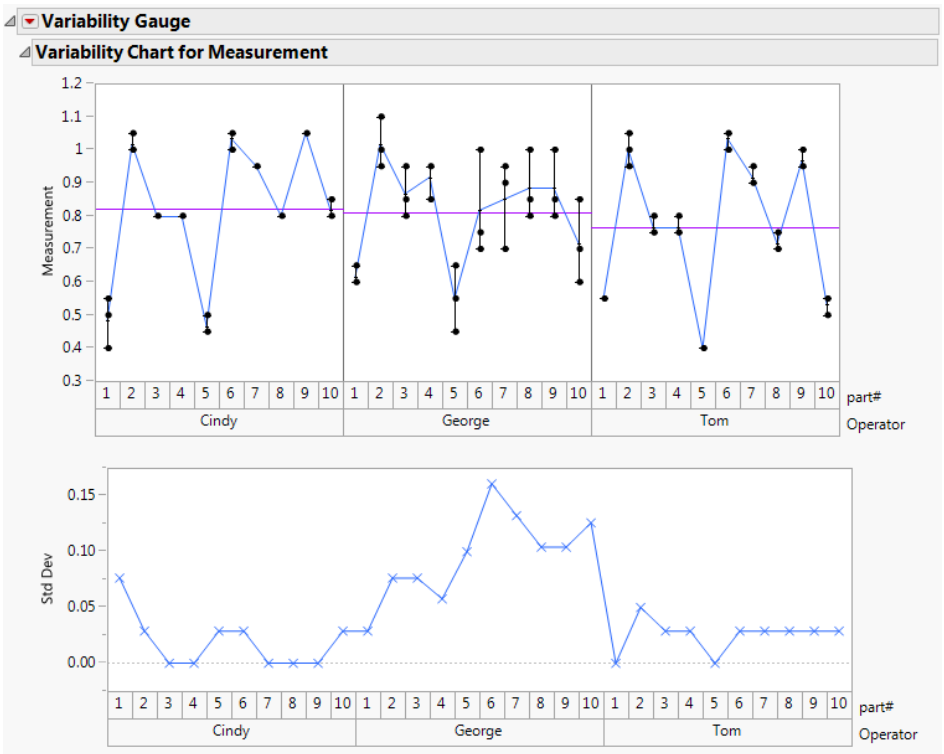
Variability charts are commonly used for measurement systems analysis such as Gauge R&R. This analysis examines how much of the variability is due to operator variation (reproducibility) and measurement variation (repeatability). Gauge R&R is available for many combinations of crossed and nested models, regardless of whether the model is balanced.

Example of a Variability Chart

Suppose that you have data containing part measurements. Three operators, Cindy, George, and Tom, each took measurements of 10 parts. They measured each part three times, making a total of 90 observations. You want to identify the variation between operators.

1. Select **Help > Sample Data Library** and open Variability Data/2 Factors Crossed.jmp.
2. Select **Analyze > Quality and Process > Variability / Attribute Gauge Chart.**
3. For **Chart Type**, select **Variability**.
4. Select Measurement and click **Y, Response**.
5. Select Operator and click **X, Grouping**.
6. Select part# and click **Part, Sample ID**.
7. Click **OK**.
8. Click the Variability Gauge red triangle and select **Show Group Means** and **Connect Cell Means**.

Figure 5.2 Example of a Variability Chart

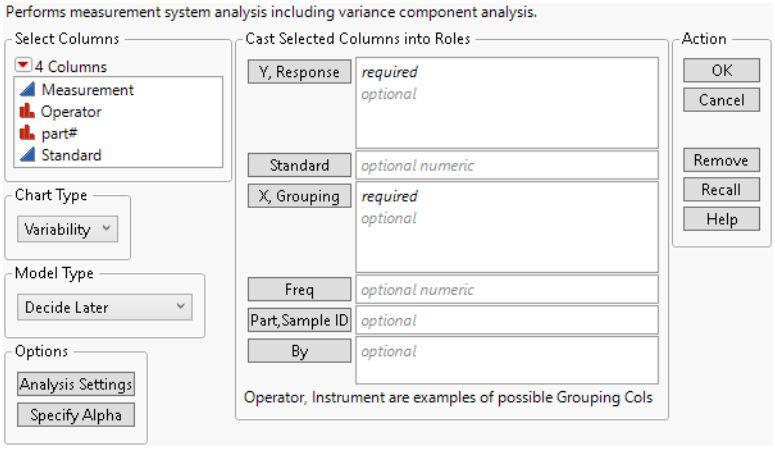


Looking at the Std Dev chart, you can see that Cindy and George have more variation in their measurements than Tom, who appears to be measuring parts the most consistently. George seems to have the most variation in his measurements, so he might be measuring parts the most inconsistently.

Launch the Variability/Attribute Gauge Chart Platform

Launch the Variability/Attribute Gauge Chart platform by selecting **Analyze > Quality and Process > Variability/Attribute Gauge Chart**. Set the **Chart Type** to **Variability**.

Figure 5.3 The Variability/Attribute Gauge Chart Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Chart Type Choose between a variability gauge analysis (for a continuous response) or an attribute gauge analysis (for a categorical response, usually “pass” or “fail”).

> **Note:** The content in this chapter covers only the **Variability** chart type. For more information about the **Attribute** chart type, see “Attribute Gauge Charts”.

Model Type Choose the model type (**Main Effect**, **Crossed**, **Nested**, and so on). See “Statistical Details for Variance Components”.

Analysis Settings Specify the method for computing variance components. See “Analysis Settings”.

Specify Alpha Specify the alpha level used by the platform.

Y, Response Specify the measurement column. Specifying more than one *Y* column produces a separate variability chart for each response.

Standard Specify a standard or reference column that contains the "true" or known values for the measured part. Including this column enables the **Bias** and **Linearity Study** options. These options perform analysis on the differences between the observed measurement and the reference or standard value. See "Bias Report" and "Linearity Study".

X, Grouping Specify the classification columns that group the measurements. If the factors form a nested hierarchy, specify the higher terms first. If you are doing a gauge study, specify the operator first and then the part.

Freq Identifies the data table column whose values assign a frequency to each row. Can be useful when you have summarized data.

Part, Sample ID Identifies the part or sample that is being measured.

By Identifies a column that creates a report consisting of separate analyses for each level of the variable.

Data Format

To use the Variability Chart platform, all response measurements must be in a single response column. Sometimes, responses are recorded in multiple columns, where each row is a level of a design factor and each column is a level of a different design factor. Data that are in this format must be stacked before running the Variability Chart platform. See *Using JMP*.

The Variability Gauge Chart

The variability chart and the standard deviation chart show patterns of variation. You can use these charts to identify possible groups of variation (within subgroups, between subgroups, over time). If you notice that any of these sources of variation are large, you can then work to reduce the variation for that source.

Follow the instructions in "Example of a Variability Chart" to produce the results shown in Figure 5.4.

Figure 5.4 Variability Gauge Chart

The charts show the response on the *y*-axis and a multilevel, categorized *x*-axis.

In Figure 5.4, the Measurement chart shows the range of measurements for each operator by part. Each measurement appears on the chart. Maximum and minimum bars indicate the range of values for each cell, and a cell means bar indicates the median value for each combination of values. The Std Dev chart plots the standard deviation of the measurements taken on each part by operator.

You can add features to the charts, as illustrated in Figure 5.4. See "Variability Gauge Platform Options".

To replace variables in charts, do one of the following:

- Swap existing variables by dragging a variable from one axis label to the other axis label. When you drag a variable over a chart or click an axis label, the axis labels are highlighted. This indicates where to drop the variable.
- Click a variable in the Columns panel of the associated data table and drag it onto an axis label.

In other platforms, rows that are excluded in the associated data table still appear on the charts or plots. However, in variability charts, excluded rows do not appear on the charts.

Variability Gauge Platform Options

Use the red triangle options to modify the appearance of the chart, perform Gauge R&R analysis, and compute variance components.

Note: Figure 5.4 illustrates some of these options.

Tip: To set the default behavior of these options, select **File > Preferences > Platforms > Variability Chart**.

Vertical Charts Changes the layout to horizontal or vertical.

Variability Chart Shows or hides the variability chart.

Show Points Shows or hides the points for individual rows.

Show Range Bars Shows or hides the bars indicating the minimum and the maximum value of each cell.

Show Cell Means Shows or hides the mean mark for each cell.

Connect Cell Means Connects or disconnects cell means within a group of cells.

Show Separators Shows or hides the separator lines between levels of the **X, Grouping** variables.

Show Group Means (Available only if you have two or more X, Grouping variables or one X, Grouping variable and one Part, Sample ID variable.) Shows or hides the mean for groups of cells, represented by a horizontal solid line. A window appears, prompting you to select one of the grouping variables.

Show Grand Mean Shows or hides the overall mean, represented by a gray dotted line across the entire graph.

Show Grand Median Shows or hides the overall median, represented by a blue dotted line across the entire graph.

Show Box Plots Shows or hides box plots.

Mean Diamonds Shows or hides mean diamonds. The confidence intervals use the within-group standard deviation for each cell.

XBar Control Limits Shows or hides lines at the UCL and LCL on the variability chart. For more information about the calculations of these limits, see "Statistical Details for Control Chart Builder".

Points Jittered Adds some random noise to the plotted points so that coincident points do not plot on top of one another.

Show Standard Mean (Available only if you have specified a **Standard** variable.) Shows or hides the mean of the standard column.

Variability Summary Report Shows or hides a report that gives the mean, standard deviation, coefficient of variation (CV), standard error of the mean, lower and upper confidence intervals, and the minimum, maximum, and number of observations.

Std Dev Chart Shows or hides a separate graph that shows cell standard deviations across category cells.

Mean of Std Dev Shows or hides a line at the mean standard deviation on the Std Dev chart.

S Control Limits Shows or hides lines showing the LCL and UCL in the Std Dev chart. For more information about the calculations of these limits, see "Statistical Details for Control Chart Builder".

Group Means of Std Dev Shows or hides the mean lines on the Std Dev chart.

Heterogeneity of Variance Tests Performs a test for comparing variances across groups. See "Heterogeneity of Variance Tests".

Variance Components Estimates the variance components for a specific model. Variance components are computed for these models: main effects, crossed, nested, crossed then nested (three factors only), and nested then crossed (three factors only). See "Variance Components".

Gauge Studies Contains the following options:

Gauge R&R Interprets the first factors as grouping columns and the last factor as Part, and creates a gauge R&R report using the estimated variance components. (Note that there is also a Part field in the launch window). See "Gauge R&R Option".

Discrimination Ratio Characterizes the relative usefulness of a given measurement for a specific product. It compares the total variance of the measurement with the variance of the measurement error. See "Discrimination Ratio".

Misclassification Probabilities Shows probabilities for rejecting good parts and accepting bad parts. See "Misclassification Probabilities".

Bias Report Shows the average difference between the observed values and the standard. A graph of the average biases and a summary table appears. This option is available only when you specify a Standard variable in the launch window. See "Bias Report".

Linearity Study Performs a regression using the standard values as the X variable and the bias as the Y variable. This analysis examines the relationship between bias and the size of the part. Ideally, you want the slope to equal 0. A nonzero slope indicates that your gauge performs differently with different sized parts. This option is available only when you specify a Standard variable in the launch window. See "Linearity Study".

Gauge R&R Plots Shows or hides Mean Plots (the mean response by each main effect in the model) and Std Dev plots. If the model is purely nested, the graphs appear with a nesting structure. If the model is purely crossed, interaction graphs appear. Otherwise, the graphs plot independently at each effect. For the standard deviation plots, the red lines connect $\sqrt{\text{mean weighted variance}}$ for each effect.

AIAG Labels Enables you to specify that quality statistics should be labeled in accordance with the AIAG standard, which is used extensively in automotive analyses.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Heterogeneity of Variance Tests

Note: See "Example of the Heterogeneity of Variance Test".

The **Heterogeneity of Variance Tests** option performs a test for comparing variances across groups. The test is an Analysis of Means for Variances (ANOMV) based method. This method indicates whether any of the group standard deviations are different from the square root of the average group variance.

To be robust against non-normal data, the method uses a permutation simulation to compute decision limits. For more information about this method, see Wludyka and Sa (2004). Because the method uses simulations, the decision limits can be slightly different each time. To obtain the same results each time, press Ctrl+Shift, select the option, and then specify the same random seed.

The red triangle menus for the test reports contain the following options:

Set Alpha Level Sets the alpha level for the test.

Show Summary Report Shows or hides a summary report for the test. The values in the report are the same values that are shown in the plot.

Display Options Shows or hides the decision limits, shading, center line, and needles.

Variance Components

The **Variance Components** option models the variation from measurement to measurement. The response is assumed to be a constant mean plus random effects associated with various levels of the classification.

Note: Once you select the **Variance Components** option, if you did not select the **Model Type** in the launch window (if you selected **Decide Later**), you are prompted to select the model type. For more information about model types, see "Launch the Variability/Attribute Gauge Chart Platform".

Figure 5.5 Example of the Variance Components Report

Analysis of Variance

Source	DF	SS	Mean Square	F Ratio	Prob > F
Operator	2	0.054889	0.02744	1.3150	0.2931
part#	9	2.633583	0.29262	14.0209	<.0001*
Operator*part#	18	0.375667	0.02087	5.0425	<.0001*
Within	60	0.248333	0.00414		
Total	89	3.312472	0.03722		

Variance Components

Component	Var Component	% of Total	20 40 60 80	Sqrt(Var Comp)
Operator	0.00021914	0.5461		0.01480
part#	0.03019444	75.2		0.17377
Operator*part#	0.00557716	13.9		0.07468
Within	0.00413889	10.3		0.06433
Total	0.04012963	100.0		0.20032

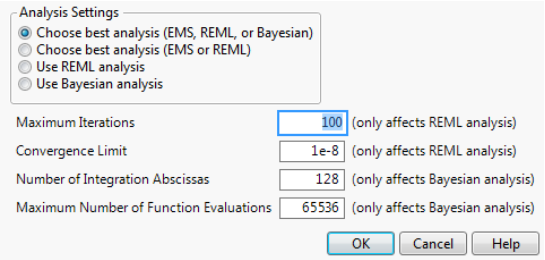
The Analysis of Variance report appears only if the EMS method of variance component estimation is used. This report shows the significance of each effect in the model.

The Variance Components report shows the estimates themselves. See "Statistical Details for Variance Components".

Analysis Settings

From the launch window, click **Analysis Settings** to choose the method for computing variance components.

Figure 5.6 Analysis Settings Window



Choose best analysis (EMS, REML, or Bayesian) Chooses the best analysis from EMS, REML, or Bayesian, using the following logic:

- If the data are balanced, and if no variance components are negative, the EMS (expected mean squares) method is used to estimate the variance components.
- If the data are unbalanced, the REML (restricted maximum likelihood) method is used, unless a variance component is estimated to be negative, then the Bayesian method is used.
- If any variance component is estimated to be negative using the EMS method, the Bayesian method is used.
- If there is confounding in the variance components, then the bounded REML method is used, and any negative variance component estimates are set to zero.

Choose best analysis (EMS or REML) Chooses the best analysis from EMS or REML, using the same logic as the **Choose best analysis (EMS, REML, or Bayesian)** option. However, this option never uses the Bayesian method, even for negative variance components. The bounded REML method is used and any negative variance component is forced to be 0.

Use REML analysis Uses the bounded REML method, even if the data are balanced. The bounded REML method can handle unbalanced data and forces any negative variance component to be 0.

Use Bayesian analysis Uses the Bayesian method. The Bayesian method can handle unbalanced data and forces all variances components to be positive and nonzero. If there is confounding in the variance components, then the bounded REML method is used, and any negative variance component estimates are set to zero. The method implemented in JMP computes the posterior means using a modified version of Jeffreys' prior. See Portnoy (1971) and Sahai (1974).

Maximum Iterations (Applicable only for the REML method.) For difficult problems, you might want to increase the number of iterations. Increasing this value means that JMP will try more times to find a solution in the optimization phase.

Convergence Limit (Applicable only for the REML method.) For problems where you want greater precision, you might want to change the convergence limit to be smaller.

Decreasing this value means that JMP will find the solution to a higher level of accuracy in the optimization phase. However, this can increase the time taken to find a solution. Providing a larger convergence value returns quicker results, but is less precise.

Number of Iteration Abscissas (Applicable only for the Bayesian method.) For greater accuracy, you might want to increase the number of iteration abscissas. However, this can increase the time taken to find a solution. Providing a smaller number returns quicker results, but is less precise.

Maximum Number of Function Evaluations (Applicable only for the Bayesian method.) For greater accuracy, you might want to increase the maximum number of function evaluations. However, this can increase the time taken to find a solution. Providing a smaller number returns quicker results, but is less precise.

About the Gauge R&R Method

The Gauge R&R method analyzes how much of the variability in your measurement system is due to operator variation (reproducibility) and measurement variation (repeatability). Gauge R&R studies are available for many combinations of crossed and nested models, regardless of whether the model is balanced.

Tip: Alternatively, you can use the EMP method to assess your measurement system. See "Measurement Systems Analysis".

Before performing a Gauge R&R study, you collect a random sample of parts over the entire range of part sizes from your process. Select several operators at random to measure each part several times. The variation is then attributed to the following sources:

- The *process variation*, from one part to another. This is the ultimate variation that you want to be studying if your measurements are reliable.
- The variability inherent in making multiple measurements, that is, *repeatability*. In Table 5.1, this is called the *within variation*.
- The variability due to having different operators measure parts—that is, *reproducibility*.

A Gauge R&R analysis then reports the variation in terms of repeatability and reproducibility.

Table 5.1 Definition of Terms and Sums in Gauge R&R Analysis

Variances Sums	Term and Abbreviation	Alternate Term
V(Within)	Repeatability (EV)	Equipment Variation

Table 5.1 Definition of Terms and Sums in Gauge R&R Analysis *(Continued)*

Variances Sums	Term and Abbreviation	Alternate Term
V(Operator)+V(Operator*Part)	Reproducibility (AV)	Appraiser Variation
V(Operator*Part)	Interaction (IV)	Interaction Variation
V(Within)+V(Operator)+V(Operator*Part)	Gauge R&R (RR)	Measurement Variation
V(Part)	Part Variation (PV)	Part Variation
V(Within)+V(Operator)+V(Operator*Part)+V(Part)	Total Variation (TV)	Total Variation

A Shewhart control chart can identify processes that are going out of control over time. A variability chart can also help identify operators, instruments, or part sources that are systematically different in mean or variance.

Gauge R&R Option

The **Gauge R&R** option shows measures of variation interpreted for a gauge study of operators and parts.

Once you select the **Gauge R&R** option, if you have not already selected the model type, you are prompted to do so. Then, modify the Gauge R&R specifications.

Note: The Platform preferences for Variability include the Gauge R&R Specification Dialog option. The preference is selected by default. Deselect the preference to use the spec limits that are defined in the data table.

Enter/Verify Gauge R&R Specifications

The Enter/Verify Gauge R&R Specifications window contains these options:

Choose tolerance entry method Choose how to enter the tolerance:

Select **Tolerance Interval** to enter the tolerance directly, where tolerance = USL – LSL.

Select **LSL and/or USL** to enter the specification limits and then have JMP calculate the tolerance.

K, Sigma Multiplier **K** is a constant value that you choose to multiply with sigma. For example, you might type 6 so that you are looking at 6*sigma or a 6 sigma process.

Tip: Modify the default value of **K** by selecting **File > Preferences > Platforms > Variability Chart**.

Tolerance Interval, USL-LSL Enter the tolerance for the process, which is the difference between the upper specification limits and the lower specification limits.

Spec Limits Enter upper and lower specification limits. See *Using JMP*.

Historical Mean Computes the tolerance range for one-sided specification limits, either USL-Historical Mean or Historical Mean-LSL. If you do not enter a historical mean, the grand mean is used.

Historical Sigma Enter a value that describes the variation (you might have this value from history or past experience).

The Gauge R&R Report

Figure 5.7 Example of the Gauge R&R Report

Gauge R&R

Measurement Source		Variation (6*StdDev)		which is 6*sqrt of
Repeatability	(EV)	0.3860052	Equipment Variation	V(Within)
Reproducibility	(AV)	0.4568005	Appraiser Variation	V(Operator) + V(Operator*part#)
Operator		0.0888194		V(Operator)
Operator*part#		0.4480823		V(Operator*part#)
Gauge R&R	(RR)	0.5980524	Measurement Variation	V(Within) + V(Operator) + V(Operator*part#)
Part Variation	(PV)	1.0425929	Part Variation	V(part#)
Total Variation	(TV)	1.2019429	Total Variation	V(Within) + V(Operator) + V(Operator*part#) + V(part#)

6 k
49.7571 % Gauge R&R = 100*(RR/TV)
0.57362 Precision to Part Variation = RR/PV
2 Number of Distinct Categories = 1.41*(PV/RR)
Using last column 'part#' for Part.

Full Gauge R&R Report

Gauge R&R

Measurement		Variation (6*StdDev)		which is 6*sqrt of
Operator*part#	(IV)	0.4480823	Interaction Variation	V(Operator*part#)
Repeatability	(EV)	0.3860052	Equipment Variation	V(Within)
Reproducibility	(AV)	0.4568005	Appraiser Variation	V(Operator)+V(Operator*part#)
Gauge R&R	(RR)	0.5980524	Measurement Variation	V(Within)+V(Operator)+V(Operator*part#)
Part Variation	(PV)	1.0425929	Part Variation	V(part#)
Total Variation	(TV)	1.2019429	Total Variation	V(Within)+V(Operator)+V(Operator*part#)+V(part#)

6 k
49.7571 % Gauge R&R = 100*(RR/TV)
0.57362 Precision to Part Variation = RR/PV
2 Number of Distinct Categories = 1.41*(PV/RR)

Reduced Gauge R&R Report

Note: To generate the reduced Gauge R&R report, select **File > Preferences > Platforms > Variability Chart** > **Reduced Gauge R&R Report**.

In this example, the values in the Variation column are the square roots of sums of variance components scaled by the value of *k* (6 in this example).

Table 5.2 shows guidelines for measurement variation, as suggested by Barrentine (1991).

Table 5.2 Acceptable Percent Measurement Variation

< 10%	excellent
11% to 20%	adequate
21% to 30%	marginally acceptable
> 30%	unacceptable

Note the following:

- If you have provided a **Tolerance Interval** in the Enter/Verify Gauge R&R Specifications window, a % of Tolerance column appears in the Gauge R&R report. This column is computed as 100*(Variation/Tolerance). Also, a Precision-to-Tolerance ratio appears at the bottom of the report. This ratio represents the proportion of the tolerance or capability interval that is lost due to gauge variability.
- If you have provided a **Historical Sigma** in the Enter/Verify Gauge R&R Specifications window, a % Process column appears in the Gauge R&R report. This column is defined as 100*(Variation/(K*Historical Sigma)).
- The Number of Distinct Categories (NDC) is defined as (1.41*(PV/RR)), rounded down to the nearest integer.

Discrimination Ratio

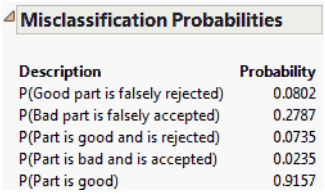
The discrimination ratio characterizes the relative usefulness of a given measurement for a specific product. Generally, when the discrimination ratio is less than 2, the measurement cannot detect product variation, implying that the measurement process needs improvement. A discrimination ratio greater than 4 adequately detects unacceptable product variation, implying that the production process needs improvement.

See “Statistical Details for the Discrimination Ratio”.

Misclassification Probabilities

Due to measurement variation, good parts can be rejected and bad parts can be accepted. This is called misclassification. Misclassification rates decrease as measurement variability decreases. When you select the **Misclassification Probabilities** option, you are prompted to select the model type and enter specification limits if you have not already done so.

Figure 5.8 Example of the Misclassification Probabilities Report



Misclassification Probabilities

Description	Probability
P(Good part is falsely rejected)	0.0802
P(Bad part is falsely accepted)	0.2787
P(Part is good and is rejected)	0.0735
P(Part is bad and is accepted)	0.0235
P(Part is good)	0.9157

The misclassification probabilities are based on the joint probability function of *Y*, the measured value of the part, and *X*, the true value of the part. The joint probability density function used is a bivariate normal distribution. To understand the descriptions, define the following probabilities:

$\delta = P[(LSL \leq X \leq USL) \text{ and } (Y < LSL \text{ or } Y > USL)]$

$\beta = P[(X < LSL \text{ or } X > USL) \text{ and } (LSL \leq Y \leq USL)]$

$\pi = P(LSL \leq X \leq USL)$

Descriptions

P(Good part is falsely rejected) The conditional probability that a part is rejected given that it is a good part, or δ/π.

P(Bad part is falsely accepted) The conditional probability that a part is accepted given that it is a bad part, or $\beta/(1-\pi)$.

P(Part is good and is rejected) The joint probability that a part is good and that it is rejected, or δ.

P(Part is bad and is accepted) The joint probability that a part is bad and that it is accepted, or β.

P(Part is good) The probability that a part is good, or π.

For more information, see "Statistical Details for the Misclassification Probabilities" as well as Burdick et al. (2005).

Bias Report

The **Bias Report** shows a graph for Overall Measurement Bias with a summary table and a graph for Measurement Bias by Standard with a summary table. The average bias, or the differences between the observed values and the standard values, appears for each level of the X variable. A *t* test for the bias is also given.

The **Bias Report** option is available only when a Standard variable is provided in the launch window.

The Measurement Bias Report red triangle menu contains the following options:

Confidence Intervals Calculates confidence intervals for the average bias for each part and places marks on the Measurement Bias Report by Standard plot.

Measurement Error Graphs Produces a graph of Measurement Error versus all grouping columns together. There are also graphs of Measurement Error by each grouping column separately.

Linearity Study

The **Linearity Study** performs a regression analysis using the standard variable as the X variable and the bias as the Y variable. This analysis examines the relationship between bias and the size of the part. Ideally, you want to find a slope of zero. If the slope is significantly different from zero, you can conclude that there is a significant relationship between the size of the part or variable measured as a standard and the ability to measure.

The **Linearity Study** option is available only when a Standard variable is provided in the launch window.

The report shows the following information:

- Bias summary statistics for each standard.
- An ANOVA table that tests if the slope of the line is equal to zero.
- The line parameters, including tests for the slope (linearity) and intercept (bias). The test for the intercept is useful only if the test on the slope fails to reject the hypothesis of slope = 0.
- The equation of the line appears directly beneath the graph.

The Linearity Study red triangle menu contains the following options:

Set Alpha Level Changes the alpha level that is used in the bias confidence intervals.

Linearity by Groups Produces separate linearity plots for each level of the **X, Grouping** variables that you specified in the launch window.

Additional Examples of Variability Charts

- "Example of the Heterogeneity of Variance Test"
- "Example of the Bias Report Option"

Example of the Heterogeneity of Variance Test

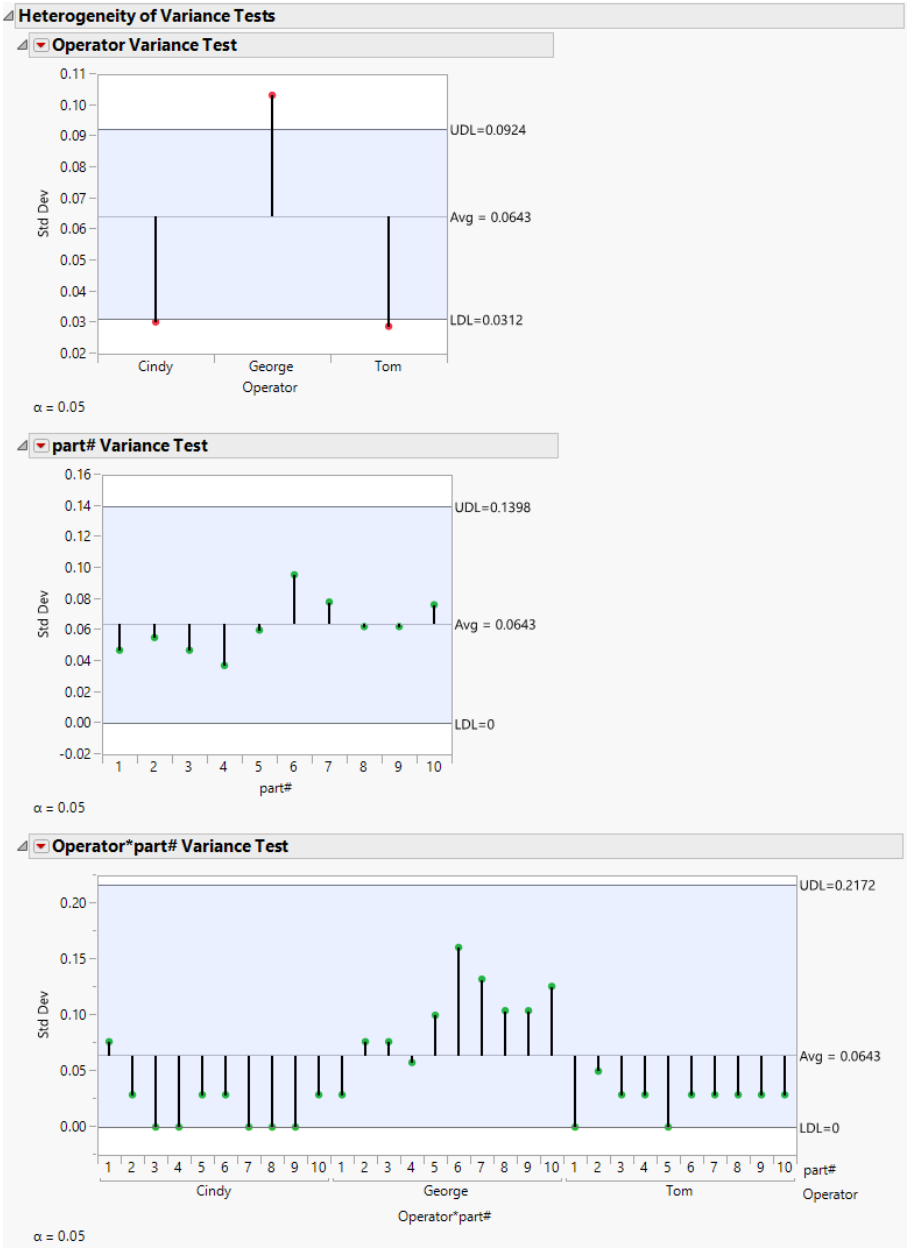
Suppose that you have data containing part measurements. Three operators (Cindy, George, and Tom) each took measurements of 10 parts. They measured each part three times, making a total of 90 observations. You want to examine the following:

- whether the variance of measurements for each operator are the same or different
- whether the variance for each part is the same or different
- whether the variance for each Operator*part combination is the same or different

Ideally, you want all of the variances for each of the groups to be considered the same statistically.

1. Select **Help > Sample Data Library** and open Variability Data/2 Factors Crossed.jmp.
2. Select **Analyze > Quality and Process > Variability / Attribute Gauge Chart**.
3. Select Measurement and click **Y, Response**.
4. Select Operator and click **X, Grouping**.
5. Select part# and click **Part, Sample ID**.
6. Click **OK**.
7. Click the Variability Gauge red triangle and select **Heterogeneity of Variance Tests**.
8. Select **Crossed**.
9. Click **OK**.

Figure 5.9 Heterogeneity of Variances Tests Report



Note: Because the method uses simulations, the decision limits can be slightly different each time.

In the Operator Variance test, all three levels exceed the upper and lower decision limits. From this, you conclude that each operator has a different variability from the square root of the average group variance. You might want to examine why the variation between each operator is different.

For the part# Variance test and the interaction (Operator*part#) Variance test, none of the levels exceed the decision limits. From this, you conclude that the variances are not statistically different from the square root of the average group variance. Each part has a similar variance to the other parts, and each Operator*part# combination has similar variance to the other Operator*part# combinations.

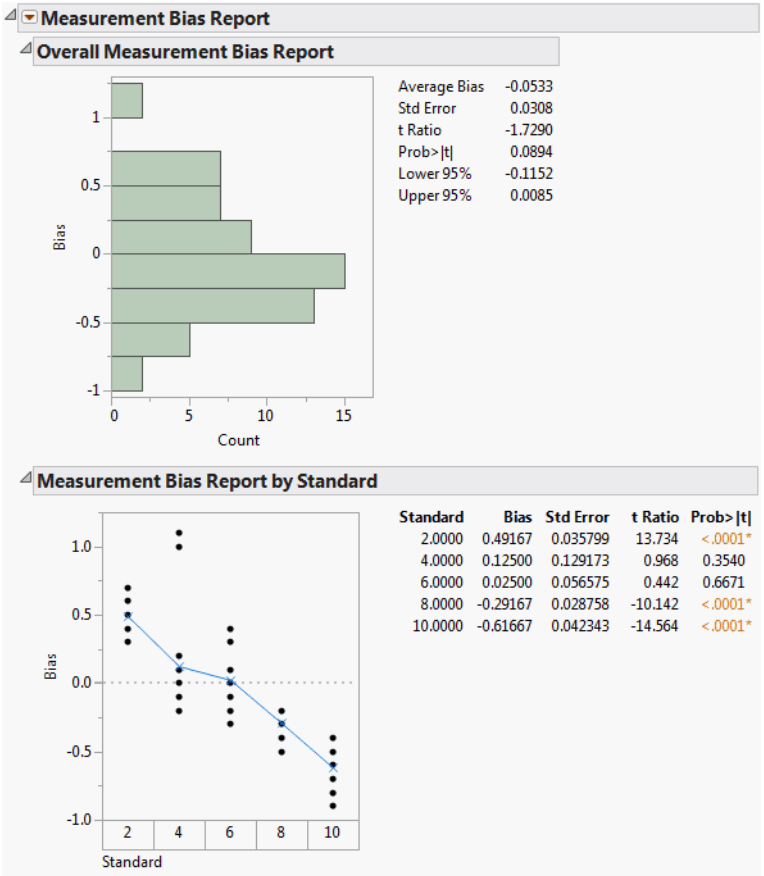
Example of the Bias Report Option

Note: These data come from the Automotive Industry Action Group (2002).

Assume that as a plant supervisor, you are introducing a new measurement system into your process. As part of the Production Part Approval Process (PPAP), the bias and linearity of the measurement system needs to be evaluated. Five parts were chosen throughout the operating range of the measurement system, based on documented process variation. Each part was measured by layout inspection to determine its reference value. Each part was then measured twelve times by the lead operator. The parts were selected at random during the day. In this example, you want to examine the overall bias and the individual measurement bias (by standard).

1. Select **Help > Sample Data Library** and open Variability Data/MSALinearity.jmp.
2. Select **Analyze > Quality and Process > Variability / Attribute Gauge Chart**.
3. Select Response and click **Y, Response**.
4. Select Standard and click **Standard**.
5. Select Part and click **X, Grouping**.
6. Click **OK**.
7. Click the Variability Gauge red triangle and select **Gauge Studies > Bias Report**.

Figure 5.10 Measurement Bias Report



The bias (Response minus Standard) is calculated for every measurement. The Overall Measurement Bias Report shows a histogram of the bias and a *t* test to see whether the average bias is equal to 0. You can see that the Average Bias is not zero, it is -0.0533. However, zero is contained within the confidence interval (-0.1152,0.0085), which means that the Average Bias is not significantly different from 0. Using a significance level of 0.05, you can see that the *p*-value is greater than 0.05, which also shows that the Average Bias is not significantly different from 0.

The Measurement Bias Report by Standard shows average bias values for each part. The bias averages are plotted on the graph along with the actual bias values for every part, so that you can see the spread. In this example, part number 1 (with a standard value of 2) is biased high and parts 4 and 5 (with standard values of 8 and 10) are biased low.

Tip: To see confidence intervals for the bias, right-click in the table and select **Columns > Lower 95% and Upper 95%**.

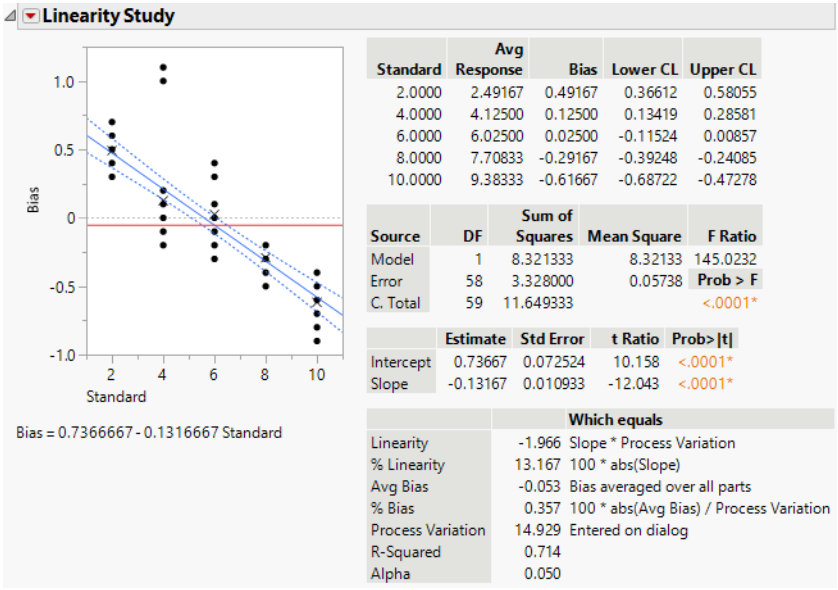
Example of a Linearity Study

Using the same data and scenario as the **Bias Report** option, you can now examine the linearity to determine whether there is a significant relationship between the size of the parts and the operator's ability to measure them.

1. Select **Help > Sample Data Library** and open Variability Data/MSALinearity.jmp.
2. Select **Analyze > Quality and Process > Variability / Attribute Gauge Chart**.
3. Select Response and click **Y, Response**.
4. Select Standard and click **Standard**.
5. Select Part and click **X, Grouping**.
6. Click **OK**.
7. Click the Variability Gauge red triangle and select **Gauge Studies > Linearity Study**.
8. In the window that prompts you to Specify Process Variation, type 14.9286.

   The value 14.9286 is 6 times the standard deviation of the response, 2.488105.

Figure 5.11 Linearity Study



Standard	Avg Response	Bias	Lower CL	Upper CL
2.0000	2.49167	0.49167	0.36612	0.58055
4.0000	4.12500	0.12500	0.13419	0.28581
6.0000	6.02500	0.02500	-0.11524	0.00857
8.0000	7.70833	-0.29167	-0.39248	-0.24085
10.0000	9.38333	-0.61667	-0.68722	-0.47278

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	1	8.321333	8.32133	145.0232
Error	58	3.328000	0.05738	Prob > F
C. Total	59	11.649333		<.0001*

	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	0.73667	0.072524	10.158	<.0001*
Slope	-0.13167	0.010933	-12.043	<.0001*

		Which equals
Linearity	-1.966	Slope * Process Variation
% Linearity	13.167	100 * abs(Slope)
Avg Bias	-0.053	Bias averaged over all parts
% Bias	0.357	100 * abs(Avg Bias) / Process Variation
Process Variation	14.929	Entered on dialog
R-Squared	0.714	
Alpha	0.050	

Note the following:

- The slope is -0.131667. This value appears as part of the equation below the graph, and also in the third table.
- The *p*-value associated with the test on the slope is quite small, <.0001. The *t* test for the slope is testing whether the bias changes with the standard value.

Because the *p*-value is small, you can conclude that there is a significant linear relationship between the size of the parts and the operator's ability to measure them. You can also see this in the graph. If the part or standard value is small, the bias is high, and vice versa.

Statistical Details for Variability Charts

- "Statistical Details for Variance Components"
- "Statistical Details for the Discrimination Ratio"
- "Statistical Details for the Misclassification Probabilities"

Statistical Details for Variance Components

The exact model type that you choose depends on how the data was collected. For example, are the operators measuring the same parts (in which case you have a crossed design) or are they measuring different parts (in which case you have a nested design)? To illustrate, in a model where *B* is nested within *A*, multiple measurements are nested within both *B* and *A*, and there are *na*•*nb*•*nw* measurements, the following statements hold:

- *na* random effects are due to *A*
- *na*•*nb* random effects due to each *nb B* levels within *A*
- *na*•*nb*•*nw* random effects due to each *nw* levels within *B* within *A*:

$$y_{ijk} = u + Za_i + Zb_{ij} + Zw_{ijk} \quad .$$

The *Z*s are the random effects for each level of the classification. Each *Z* is assumed to have a mean of zero and to be independent from all other random terms. The variance of the response *y* is the sum of the variances due to each *z* component:

$$\mathrm{Var}(y_{ijk}) = \mathrm{Var}(Za_i) + \mathrm{Var}(Zb_{ij}) + \mathrm{Var}(Zw_{ijk}) \quad .$$

Table 5.3 shows the supported models and what the effects in the model would be.

Table 5.3 Models Supported by the Variability Charts Platform

Model	Factors	Effects in the Model
Main Effects	1	A
	2	A, B
	unlimited	and so on, for more factors

Table 5.3 Models Supported by the Variability Charts Platform *(Continued)*

Model	Factors	Effects in the Model
Crossed	1	A
	2	A, B, A*B
	3	A, B, A*B, C, A*C, B*C, A*B*C
	4	A, B, A*B, C, A*C, B*C, A*B*C, D, A*D, B*D, A*B*D, C*D, A*C*D, B*C*D, A*B*C*D,
	unlimited	and so on, for more factors
Nested	1	A
	2	A, B(A)
	3	A, B(A), C(A,B)
	4	A, B(A), C(A,B), D(A,B,C)
	unlimited	and so on, for more factors
Crossed then Nested	3	A, B, A*B, C(A,B)
Nested then Crossed	3	A, B(A), C, A*C, C*B(A)

Statistical Details for the Discrimination Ratio

The discrimination ratio compares the total variance of the measurement, *M*, with the variance of the measurement error, *E*. The discrimination ratio is computed for all main effects, including nested main effects. The discrimination ratio, *D*, is computed as follows:

$$D = \sqrt{2\left(\frac{P}{T-P}\right)+1}$$

where:

P = estimated variance for a factor

T = estimated total variance

Statistical Details for the Misclassification Probabilities

This section describes the computations for the probabilities in the Misclassification Probabilities report. The misclassification probabilities are based on the joint probability function of Y, the measured value of the part, and X, the true value of the part. The joint probability distribution function $F_{YX}(y, x)$ uses a bivariate normal distribution with mean vector [μ, μ] and the following covariance matrix:

$$\begin{bmatrix} \gamma_P + \gamma_M & \gamma_P \\ \gamma_P & \gamma_P \end{bmatrix}$$

where γ_P is the part-to-part variation, γ_M is the measurement variation, and μ is the grand mean. These quantities can be found or derived from quantities in the report window. Specifically, $\gamma_P + \gamma_M$ is equal to the square of Total Variation (TV) divided by 6: $(TV/6)^2$ and γ_P is equal to the square of Part Variation (PV) divided by 6: $(PV/6)^2$. The correlation ρ_{YX} between Y and X is defined as the square root of $\gamma_P/(\gamma_P + \gamma_M)$.

Next, define the following probabilities:

$\delta = P[(LSL \le X \le USL)$ and $(Y < LSL$ or $Y > USL)]$

$\beta = P[(X < LSL$ or $X > USL)$ and $(LSL \le Y \le USL)]$

$\pi = P(LSL \le X \le USL)$

These probabilities can be expressed in terms of the joint probability distribution function $F_{YX}(y, x)$ and the marginal probability distribution functions for Y and X: $F_Y(y)$ and $F_X(x)$:

$\delta = F_{YX}(LSL, USL) - F_{YX}(LSL, LSL) - F_{YX}(USL, USL) + F_{YX}(USL, LSL) + F_X(USL) - F_X(LSL)$

$\beta = F_{YX}(USL, LSL) - F_{YX}(LSL, LSL) - F_{YX}(USL, USL) + F_{YX}(LSL, USL) + F_Y(USL) - F_Y(LSL)$

$\pi = F_X(USL) - F_X(LSL)$

P(Good part is falsely rejected) = δ/π

P(Bad part is falsely accepted) = $\beta/(1-\pi)$

P(Part is good and is rejected) = δ

P(Part is bad and is accepted) = β

P(Part is good) = π

Chapter 6

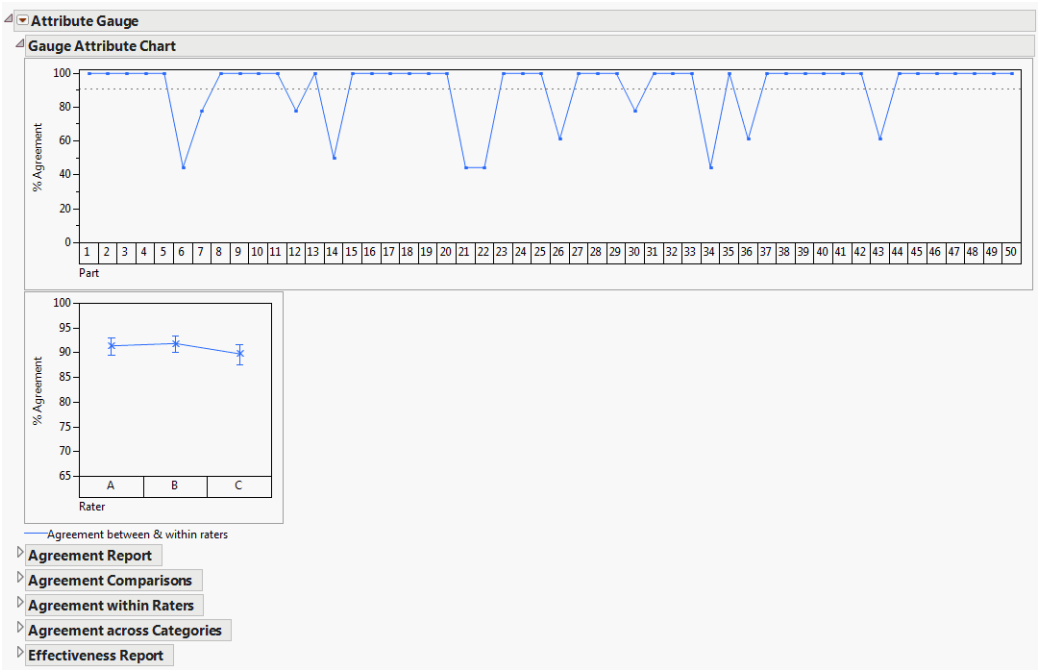
Attribute Gauge Charts

Evaluate a Categorical Measurement Process Using Agreement Measures

Attribute charts analyze categorical measurements and can help show you measures of agreement across responses, such as raters. In *attribute data*, the variable of interest has a finite number of categories. Typically, data has only two possible results, such as pass or fail. You can examine aspects such as how effective raters were at classifying a part, how much they agreed with each other, and how much they agreed with themselves over the course of several ratings.

Tip: This chapter covers only attribute charts. For more information about variability charts, see "Variability Gauge Charts".

Figure 6.1 Example of an Attribute Chart



Contents

Overview of Attribute Gauge Charts 157
Example of an Attribute Gauge Chart 157
Launch the Variability/Attribute Gauge Chart Platform 159
The Attribute Gauge Chart and Reports 160
Agreement Reports 161
Effectiveness Report 162
Attribute Gauge Platform Options 164
Statistical Details for Attribute Gauge Charts 165
Statistical Details for the Agreement Report 167

Overview of Attribute Gauge Charts

Before you create an attribute gauge chart, your data should be formatted using the following guidelines:

- In order to compare agreement among raters, each rater in the data table must be in a separate column. These columns are then assigned to the **Y, Response** role in the launch window. In Figure 6.2, each rater (A, B, and C) is in a separate column.
- Responses in the different columns can be character (pass or fail) or numeric (0 or 1). In Figure 6.2, rater responses are numeric (0 for pass, 1 for fail). All response columns must have the same data type.
- Any other variables of interest that you might want to use as **X, Grouping** variables should appear stacked in one column each (see the Part column in Figure 6.2). You can also define a Standard column, which produces reports that compare raters with the standard. The Standard column and response columns must have the same data type.

Figure 6.2 Attribute Gauge Data

Attribute Gauge
Locked File C:\Progra
Attribute Chart

Columns (7/0)
Part
Standard
Code
A
B
C
RefValue

	Part	Standard	Code	A	B	C	RefValue
1	1	1	+	1	1	1	0.476901
2	1	1	+	1	1	1	0.476901
3	1	1	+	1	1	1	0.476901
4	2	1	+	1	1	1	0.509015
5	2	1	+	1	1	1	0.509015
6	2	1	+	1	1	1	0.509015
7	3	0	-	0	0	0	0.576459
8	3	0	-	0	0	0	0.576459
9	3	0	-	0	0	0	0.576459
10	4	0	-	0	0	0	0.566152
11	4	0	-	0	0	0	0.566152

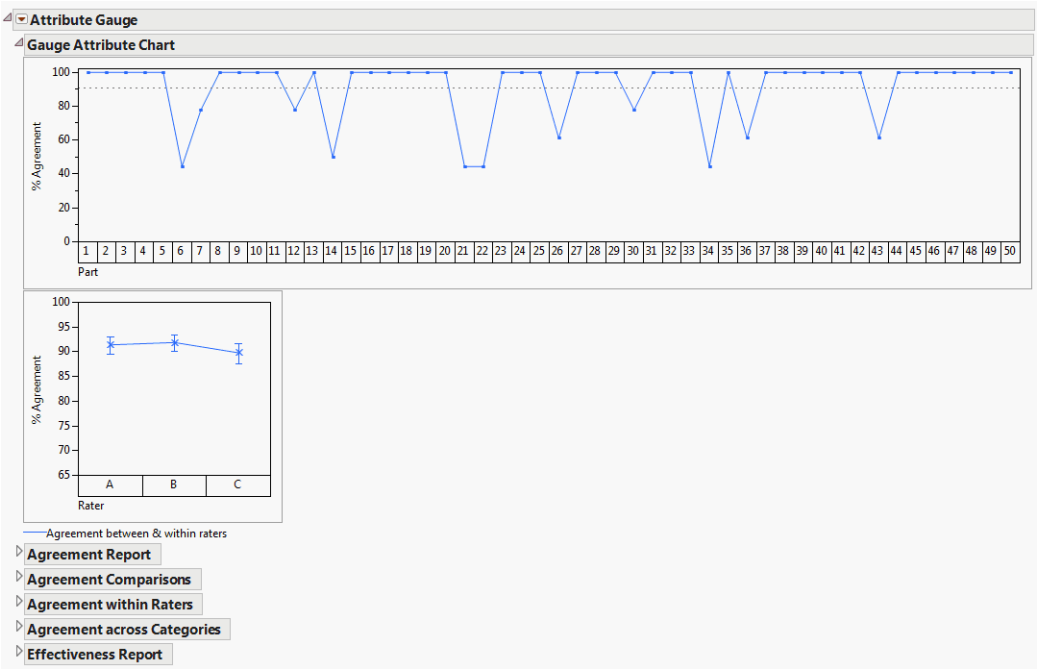
Example of an Attribute Gauge Chart

Suppose that you have data containing pass or fail ratings for parts. Three raters, identified as A, B, and C, each noted a 0 (pass) or a 1 (fail) for 50 parts, three times each. You want to examine how effective the raters were in correctly classifying the parts, and how well the raters agreed with each other and with themselves over the course of the ratings.

1. Select **Help > Sample Data Library** and open Attribute Gauge.jmp.
2. Select **Analyze > Quality and Process > Variability / Attribute Gauge Chart.**
3. For **Chart Type**, select **Attribute**.

4. Select A, B, and C and click **Y, Response**.
5. Select Standard and click **Standard**.
6. Select Part and click **X, Grouping**.
7. Click **OK**.

Figure 6.3 Example of an Attribute Chart



The first chart (Part) shows how well the raters agreed with each other for each part. For example, here you can see that the percent agreement dropped for part 6, 12, 14, 21, 22, and so on. These parts might have been more difficult to categorize.

The second chart (Rater) shows each rater's agreement with him or herself and the other raters for a given part, summed up over all of the parts. In this example, it looks like the performance of the raters is relatively similar. Rater C had the lowest agreement, but the difference is not major (about 89% instead of 91%).

8. Open the Effectiveness Report and scroll down to the Conformance Report.

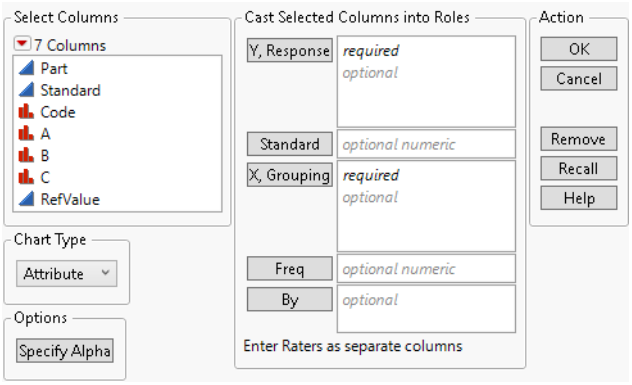
   You can see that 0 = non-conform (fail) and a 1 = conform (pass). However in this data, it is exactly the opposite: 0 is a pass and 1 is a fail. Reverse this setting.

9. Click the Conformance Report red triangle and select **Change Conforming Category**.

Launch the Variability/Attribute Gauge Chart Platform

Launch the Variability/Attribute Gauge Chart platform by selecting **Analyze > Quality and Process > Variability/Attribute Gauge Chart**. Set the **Chart Type** to **Attribute**.

Figure 6.4 The Variability/Attribute Gauge Chart Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Chart Type Choose between a variability gauge analysis (for a continuous response) or an attribute gauge analysis (for a categorical response, usually "pass" or "fail").

> **Note:** The content in this chapter covers only the **Attribute** chart type. For more information about the **Variability** chart type, see "Variability Gauge Charts".

Specify Alpha Specify the alpha level used by the platform.

Y, Response Specify the columns of ratings given by each rater. You must specify more than one rating column.

Standard Specify a standard or reference column that contains the "true" or known values for the part. In the report window, an Effectiveness Report and an additional section in the Agreement Comparisons report appear, which compare the raters with the standard.

X, Grouping Specify the classification columns that group the measurements. If the factors form a nested hierarchy, specify the higher terms first.

Freq Identifies the data table column whose values assign a frequency to each row. Can be useful when you have summarized data.

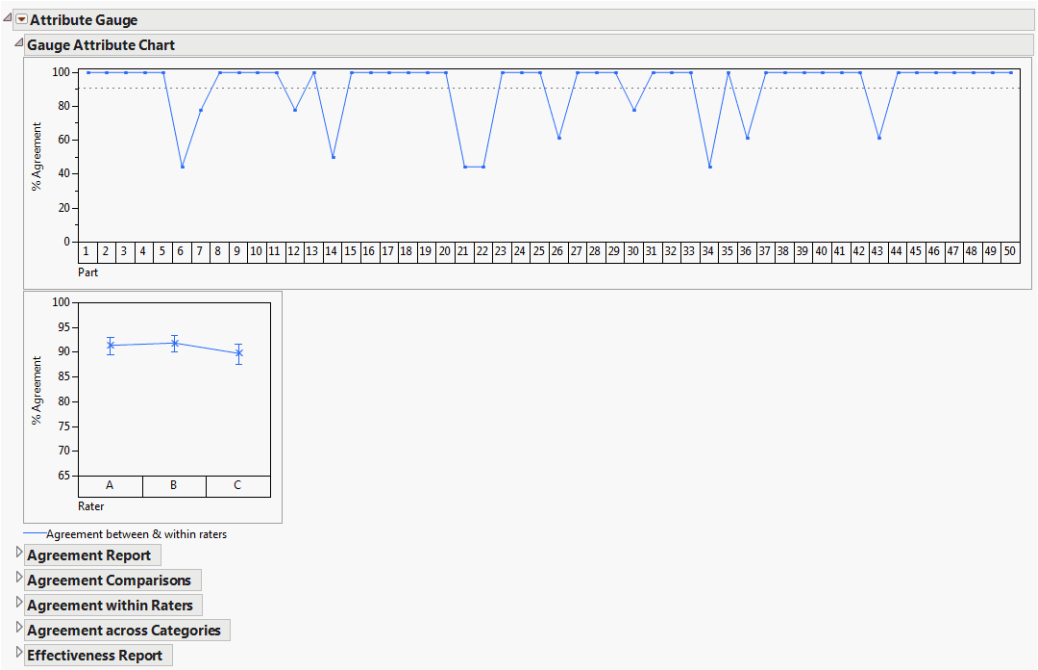
By Identifies a column that creates a report consisting of separate analyses for each level of the variable.

The Attribute Gauge Chart and Reports

Attribute gauge chart plots the percent Agreement, which is a measurement of rater agreement for every part in the study. The agreement for each part is calculated by comparing the ratings for every pair of raters for all ratings of that part. See "Statistical Details for Attribute Gauge Charts".

Follow the instructions in "Example of an Attribute Gauge Chart" to produce the results shown in Figure 6.5.

Figure 6.5 Attribute Gauge Chart



The first chart in Figure 6.5 uses all X grouping variables (in this case, the Part) on the *x*-axis. The second chart uses all Y variables on the *x*-axis (typically, and in this case, the Rater).

- In the first graph, you can look for parts with a low percent Agreement value, and investigate to determine why raters do not agree about the measurement of that particular part.

- In the second graph, you can look for raters with a low percent Agreement value, and investigate to determine why they do not agree with the other raters or with themselves.

For information about additional options, see "Attribute Gauge Platform Options".

Agreement Reports

Note: The Kappa value is a statistic that expresses agreement. The closer the Kappa value is to 1, the more agreement there is. A Kappa value closer to 0 indicates less agreement.

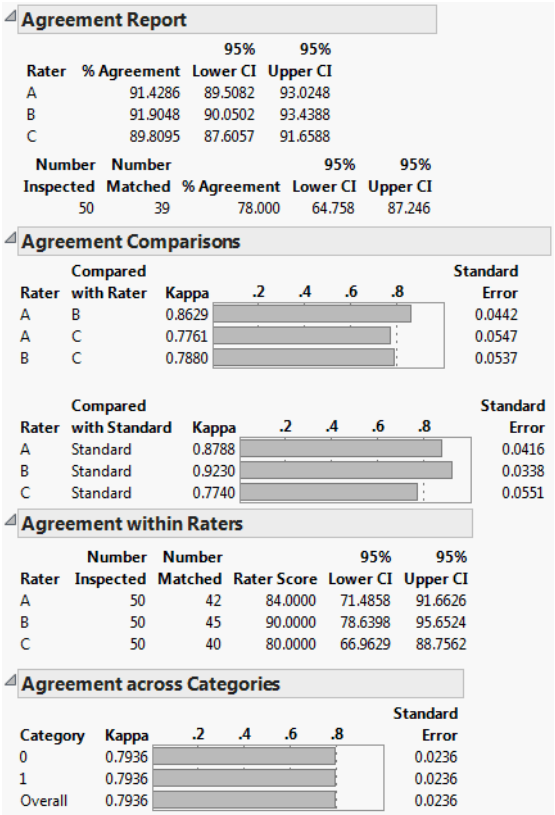
The Agreement Report shows agreement summarized for each rater and overall agreement. This report is a numeric form of the data presented in the second chart in the Attribute Gauge Chart report (Figure 6.5).

The Agreement Comparisons report shows each rater compared with all other raters, using Kappa statistics. The rater is compared with the standard only if you have specified a Standard variable in the launch window.

The Agreement within Raters report shows the number of items that were inspected. The confidence intervals are score confidence intervals, as suggested by Agresti and Coull (1998). The Number Matched is the sum of the number of items inspected, where the rater agreed with him or herself on each inspection of an individual item. The Rater Score is the Number Matched divided by the Number Inspected.

The Agreement across Categories report shows the agreement in classification over that which would be expected by chance. It assesses the agreement between a fixed number of raters when classifying items.

Figure 6.6 Agreement Reports



Agreement Report

Rater	% Agreement	95% Lower CI	95% Upper CI
A	91.4286	89.5082	93.0248
B	91.9048	90.0502	93.4388
C	89.8095	87.6057	91.6588

Number Inspected	Number Matched	% Agreement	95% Lower CI	95% Upper CI
50	39	78.000	64.758	87.246

Agreement Comparisons

Rater	Compared with Rater	Kappa	Standard Error
A	B	0.8629	0.0442
A	C	0.7761	0.0547
B	C	0.7880	0.0537

Rater	Compared with Standard	Kappa	Standard Error
A	Standard	0.8788	0.0416
B	Standard	0.9230	0.0338
C	Standard	0.7740	0.0551

Agreement within Raters

Rater	Number Inspected	Number Matched	Rater Score	95% Lower CI	95% Upper CI
A	50	42	84.0000	71.4858	91.6626
B	50	45	90.0000	78.6398	95.6524
C	50	40	80.0000	66.9629	88.7562

Agreement across Categories

Category	Kappa	Standard Error
0	0.7936	0.0236
1	0.7936	0.0236
Overall	0.7936	0.0236

Effectiveness Report

The Effectiveness Report appears only if you have specified a Standard variable in the launch window. For a description of a Standard variable, see "Launch the Variability/Attribute Gauge Chart Platform". This report compares every rater with the standard.

Figure 6.7 Effectiveness Report

Effectiveness Report

Agreement Counts

Rater	Correct(0)	Correct(1)	Total Correct	Incorrect(0)	Incorrect(1)	Grand Total
A	45	97	142	3	5	150
B	45	100	145	3	2	150
C	42	93	135	6	9	150

Effectiveness

Rater	Effectiveness	95% Lower CI	95% Upper CI	Error rate
A	94.6667	89.8296	97.2730	0.0533
B	96.6667	92.4348	98.5680	0.0333
C	90.0000	84.1565	93.8459	0.1000
Overall	93.7778	91.1542	95.6603	0.0622

Misclassifications

Standard Level	0	1
0	.	16
1	12	.
Other	0	0

Conformance Report

Rater	P(False Alarms)	P(Misses)
A	0.0490	0.0625
B	0.0196	0.0625
C	0.0882	0.1250

Assumptions
NonConform = 0
Conform = 1

The Agreement Counts table shows cell counts on the number correct and incorrect for every level of the standard. In Figure 6.7, the standard variable has two levels, 0 and 1. Rater A had 45 correct responses and 3 incorrect responses for level 0, and 97 correct responses and 5 incorrect responses for level 1.

Effectiveness is defined as the number of correct decisions divided by the total number of opportunities for a decision. For example, say that rater A sampled every part three times. On the sixth part, one of the decisions did not agree (for example, pass, pass, fail). The other two decisions would still be counted as correct decisions. This definition of effectiveness is different from the MSA 3rd edition. According to MSA, all three opportunities for rater A on part six would be counted as incorrect. Including all of the inspections separately gives you more information about the overall inspection process.

In the Effectiveness table, 95% confidence intervals are given about the effectiveness. These are score confidence intervals. It has been demonstrated that score confidence intervals provide increased coverage probability, particularly where observations lie near the boundaries. See Agresti and Coull (1998).

The Misclassifications table shows the incorrect labeling. The rows represent the levels of the standard or accepted reference value. The columns contain the levels given by the raters.

Conformance Report

The Conformance Report shows the probability of false alarms and the probability of misses. The Conformance Report appears only when the rating has two levels (such as pass or fail, or 0 or 1).

The following descriptions apply:

False Alarm The part is determined to be non-conforming, when it actually is conforming.

Miss The part is determined to be conforming, when it actually is not conforming.

P(False Alarms) The number of parts that have been incorrectly judged to be nonconforming divided by the total number of parts that are judged to be conforming.

P(Miss) The number of parts that have been incorrectly judged to be conforming divided by the total number of parts that are actually nonconforming.

The Conformance Report red triangle menu contains the following options:

Change Conforming Category Reverses the response category that is considered conforming.

Calculate Escape Rate Calculates the Escape Rate, which is the probability that a non-conforming part is produced and not detected. The Escape Rate is calculated as the probability that the process will produce a non-conforming part times the probability of a miss. You specify the probability that the process will produce a non-conforming part, also called the Probability of Nonconformance.

Note: Missing values are treated as a separate category in this platform. To avoid this separate category, exclude rows of missing values in the data table.

Attribute Gauge Platform Options

The Attribute Gauge red triangle menu contains the following options:

Attribute Gauge Charts Shows or hides the gauge attribute chart and the efficiency chart.

Show Agreement Points Shows or hides the agreement points on the charts.

Connect Agreement Points Connects the agreement points in the charts.

Agreement by Rater Confid Intervals Shows or hides the agreement by rater confidence intervals on the efficiency chart.

Show Agreement Group Means Shows or hides the agreement group means on the gauge attribute chart. This option is available when you specify more than one X, Grouping variable.

Show Agreement Grand Mean Shows or hides the overall agreement mean on the gauge attribute chart.

Show Effectiveness Points Shows or hides the effectiveness points on the charts.

Connect Effectiveness Points Draws lines between the effectiveness points in the charts.

Effectiveness by Rater Confid Intervals Shows or hides confidence intervals on the second chart in the Attribute Gauge Chart report (Figure 6.5).

Effectiveness Report Shows or hides the Effectiveness report. This report compares every rater with the standard, using the Kappa statistic.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Statistical Details for Attribute Gauge Charts

For the first chart in Figure 6.5 that plots all **X, Grouping** variables on the *x*-axis, the percent Agreement is calculated as follows:

$$\%\ \text{Agreement for part } i = \frac{\sum_{l=1}^{k} \binom{\text{number of responses for level } l}{2}}{\binom{N_i}{2}}$$

For the second chart in Figure 6.5 that plots all **Y, Response** variables on the x-axis, the percent Agreement is calculated as follows:

$$\%\ \text{Agreement for rater } k = \frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{r_i} \begin{matrix}\text{number of uncounted matching levels} \\ \text{for this rater k within part i for rep j}\end{matrix}\right)}{\sum_{i=1}^{n}\left(\sum_{j=1}^{r_i} N_i - j\right)}$$

Note the following:

- n = number of parts (grouping variables)
- r_i = number of reps for part i (i = 1,...,n)
- m = number of raters
- k = number of levels
- $N_i = m \times r_i$. Number of ratings on part i (i = 1,...,n). This includes responses for all raters, and repeat ratings on a part. For example, if part i is measured 3 times by each of 3 raters, then N_i is 3 x 3 = 9.

For example, consider the following table of data for three raters, each having three replicates for one part.

Table 6.1 Three Replicates for Raters A, B, and C

	A	B	C
1	1	1	1
2	1	1	0
3	0	0	0

Using this table, you can make these calculations:

$$\%\ \text{Agreement} = \frac{\binom{4}{2} + \binom{5}{2}}{\binom{9}{2}} = \frac{16}{36} = 0.444$$

$$\%\ \text{Agreement [rater A]} = \text{percent Agreement [rater B]} = \frac{4+3+3}{8+7+6} = \frac{10}{21} = 0.476 \quad \text{and}$$

$$\% \text{ Agreement [rater C]} = \frac{4+3+2}{8+7+6} = \frac{9}{21} = 0.4286$$

Statistical Details for the Agreement Report

The simple Kappa coefficient is a measure of inter-rater agreement.

$$\hat{\kappa} = \frac{P_0 - P_e}{1 - P_e}$$

where:

$$P_0 = \sum_i p_{ii}$$

and:

$$P_e = \sum_i p_{i.} p_{.i}$$

If you view the two response variables as two independent ratings of the n parts, the Kappa coefficient equals +1 when there is complete agreement of the raters. When the observed agreement exceeds chance agreement, the Kappa coefficient is positive, and its magnitude reflects the strength of agreement. Although unusual in practice, Kappa is negative when the observed agreement is less than the chance agreement. The minimum value of Kappa is between -1 and 0, depending on the marginal proportions.

Estimate the asymptotic variance of the simple Kappa coefficient with the following equation:

$$\text{var} = \frac{A + B - C}{(1 - P_e)^2 n}$$

where:

$$A = \sum_i p_{ii} \left[1 - (p_{i.} + p_{.i})(1 - \hat{\kappa}) \right]$$

$$B = (1 - \hat{\kappa})^2 \sum_{i \neq j} \sum p_{ij} (p_{.i} + p_{j.})^2$$

and:

$$C = \left[\hat{\kappa} - P_e(1 - \hat{\kappa})\right]^2$$

The Kappas are plotted and the standard errors are also given.

Note: The Kappa statistics in the Attribute Chart platform are shown even when the levels of the variables are unbalanced.

Categorical Kappa statistics (Fleiss 1981) are found in the Agreement Across Categories report.

Given the following assumptions:

- n = number of parts (grouping variables)
- m = number of raters
- k = number of levels
- r_i = number of reps for part i (i = 1,...,n)
- $N_i = m \times r_i$. Number of ratings on part i (i = 1, 2,...,n). This includes responses for all raters, and repeat ratings on a part. For example, if part i is measured 3 times by each of 2 raters, then N_i is 3 x 2 = 6.
- x_{ij} = number of ratings on part i (i = 1, 2,...,n) into level j (j = 1, 2,...,k)

The individual category Kappa is defined as follows:

$$\hat{\kappa}_j = 1 - \frac{\sum_{i=1}^{n} x_{ij}(N_i - x_{ij})}{(\bar{p}_j\bar{q}_j)\sum_{i=1}^{n} N_i(N_i - 1)} \quad \text{where} \quad \bar{p}_j = \frac{\sum_{i=1}^{n} x_{ij}}{\sum_{i=1}^{n} N_i} \quad \bar{q}_j = 1 - \bar{p}_j$$

The overall Kappa is defined as follows:

$$\bar{\hat{\kappa}} = \frac{\sum_{j=1}^{k} \bar{q}_j\bar{p}_j\hat{\kappa}_j}{\sum_{j=1}^{k} \bar{p}_j\bar{q}_j}$$

The variance of $\hat{\kappa}_j$ and $\bar{\hat{\kappa}}$ are calculated as follows:

$$\text{var}(\hat{\kappa}_j) = \frac{2}{nN(N-1)}$$

$$\text{var}(\bar{\hat{\kappa}}) = \frac{2}{\left(\sum_{j=1}^{k} \bar{p}_j \bar{q}_j\right)^2 nN(N-1)} \times \left[\left(\sum_{j=1}^{k} \bar{p}_j \bar{q}_j\right)^2 - \sum_{j=1}^{k} \bar{p}_j \bar{q}_j (\bar{q}_j - \bar{p}_j)\right]$$

The standard errors of $\hat{\kappa}_j$ and $\bar{\hat{\kappa}}$ are shown only when there are an equal number of ratings per part (for example, $N_i = N$ for all $i = 1,\ldots,n$).

Chapter 7

Process Capability

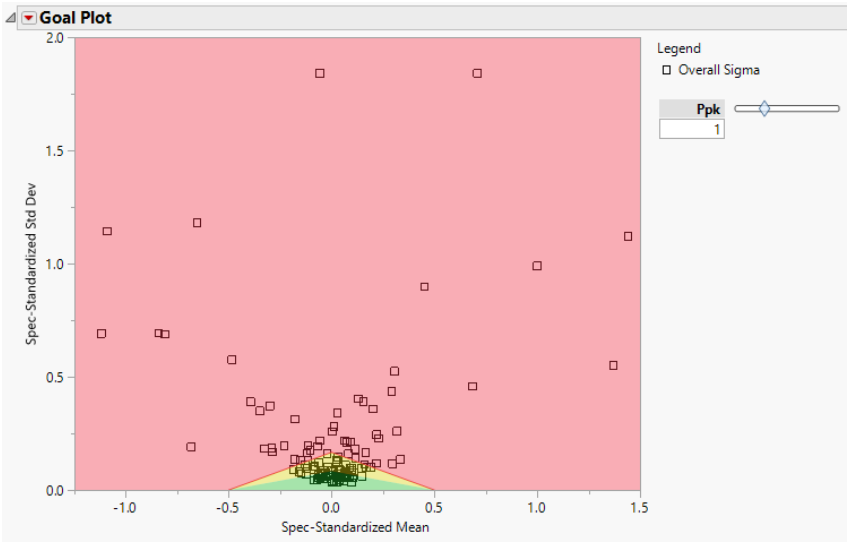
Measure the Variability of a Process over Time

Process capability analysis, used in process control, measures how well a process is performing compared to given specification limits. A good process is one that is stable and consistently produces product that is well within specification limits. A capability index is a measure that relates process performance, summarized by process centering and variability, to specification limits.

Graphical tools such as a goal plot and box plots give you quick visual ways of identifying which process or product characteristics are within specifications. Individual detail reports display a capability report for each variable in the analysis. The analysis enables you to identify variation relative to the specifications or requirements; this enables you to achieve increasingly higher conformance values.

You can specify subgroups to compare the overall variation of the process to the within subgroup variation. You can compute capability indices for processes that produce measurements that follow various distributions. For data that follow none of the specified distributions, you can compute nonparametric capability indices.

Figure 7.1 Example of the Process Capability Platform



Contents

Overview of the Process Capability Platform . . . 171
Example of the Process Capability Platform with Normal Variables . . . 173
Example of the Process Capability Platform with Nonnormal Variables . . . 175
Launch the Process Capability Platform . . . 181
Process Selection . . . 182
Process Subgrouping . . . 182
Moving Range Options . . . 183
Historical Information . . . 184
Distribution Options . . . 184
Other Specifications . . . 185
Entering Specification Limits . . . 186
Spec Limits Window . . . 186
Limits Data Table . . . 187
Spec Limits Column Property . . . 189
The Process Capability Report . . . 189
Goal Plot . . . 190
Capability Box Plots . . . 193
Capability Index Plot . . . 195
Process Capability Platform Options . . . 197
Individual Detail Reports . . . 200
Normalized Box Plots . . . 208
Process Performance Plot . . . 209
Summary Reports . . . 211
Make Goal Plot Summary Table . . . 212
Additional Examples of the Process Capability Platform . . . 213
Process Capability for a Stable Process . . . 213
Process Capability for an Unstable Process . . . 217
Simulation of Confidence Limits for a Nonnormal Process Ppk . . . 220
Statistical Details for the Process Capability Platform . . . 227
Variation Statistics . . . 227
Notation for Goal Plots and Capability Box Plots . . . 231
Goal Plot . . . 231
Capability Box Plots for Processes with Missing Targets . . . 233
Capability Indices for Normal Distributions . . . 234
Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods . . . 239
Parameterizations for Distributions . . . 241

Overview of the Process Capability Platform

The Process Capability platform provides the tools needed to measure the compliance of a process to given specifications. By default, JMP shows a Goal Plot, Capability Box Plots, and a Capability Index Plot for the variables that you fit with normal distributions. Capability indices for nonnormal variables are plotted on the Capability Index Plot. You can add normalized box plots, summary reports, and individual detail reports for the variables in your analysis.

You can supply specification limits in several ways:

- in the data table, using a column property
- by requesting the Spec Limits Dialog in the launch window
- by loading the limits from a specification limits data table
- using the Manage Spec Limits utility (Analyze > Quality and Process > Manage Spec Limits)

You can specify two-sided, one-sided, or asymmetric specification limits.

Note: The Process Capability platform expands significantly on the Capability analyses that are available through Analyze > Distribution and through Analyze > Quality and Process > Control Chart.

Capability Indices

A capability index is a ratio that relates the ability of a process to produce product that meets specification limits. The index relates estimates of the mean and standard deviation of the quality characteristic to the specification limits. Within estimates of capability are based on an estimate of the standard deviation constructed from within-subgroup variation. Overall estimates of capability use an estimate of standard deviation constructed from all of the process data. See "Capability Indices for Normal Distributions" and "Variation Statistics".

Estimates of the mean or standard deviation are well-defined only if the processes related to centering or spread are *stable*. Therefore, interpretation of within capability indices requires that process spread is stable. Interpretation of overall capability indices requires that both process centering and spread are stable.

Capability indices constructed from small samples can be highly variable. The Process Capability platform provides confidence intervals for most capability indices. Use these to determine the range of potential values for your quality characteristic's actual capability.

Note: JMP PRO When confidence intervals are not provided (for example, for nonnormal distributions) you can use the Simulate feature to construct confidence intervals. For an example, see "Simulation of Confidence Limits for a Nonnormal Process Ppk".

Guidelines for values of capability indices can be found in Montgomery (2013). The minimum recommended value is 1.33. Six Sigma initiatives aim for much higher capability levels that correspond to extremely low rates of defective parts per million.

Capability Indices for Nonnormal Processes

The Process Capability platform constructs capability indices for process measurements with the following distributions: Normal, Beta, Exponential, Gamma, Johnson, Lognormal, Mixture of 2 Normals, Mixture of 3 Normals, SHASH, and Weibull. A Best Fit option determines the best fit among these distributions and provides capability indices for this fit. The platform also provides a Nonparametric fit option that gives nonparametric estimates of capability.

For the nonnormal methods, estimates are constructed using two approaches: the ISO/Quantile method (Percentiles) and the Bothe/Z-scores method (Z-Score). For more information about these methods, see "Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods".

Note: Process Capability analysis for individual responses is accessible through Analyze > Quality and Process > Control Chart Builder. However, nonnormal distributions are available only in the Process Capability platform.

Overall and Within Estimates of Sigma

Most capability indices in the Process Capability platform can be computed based on estimates of the *overall* (long-term) variation and the *within*-subgroup (short-term) variation. If the process is stable, these two measures of variation should yield similar results since the overall and within subgroup variation should be similar. The normalized box plots and summary tables can be calculated using either the overall or the within-subgroup variation. See "Additional Examples of the Process Capability Platform" for examples of capability indices computed for stable and unstable processes.

You can specify subgroups for estimating within-subgroup variation in the launch window. You can specify a column that defines subgroups or you can select a constant subgroup size. For each of these methods, you can choose to estimate the process variation using the average of the unbiased standard deviations or using the average of the ranges. If you do not specify subgroups, the Process Capability platform constructs a within-subgroup estimate of the process variation using a moving range of subgroups of size two. Finally, you can specify a historical sigma to be used as an estimate of the process standard deviation.

Capability Index Notation

The Process Capability platform provides two sets of capability indices. See "Capability Indices for Normal Distributions" for more information about the calculation of the capability indices.

- Cpk, Cpl, Cpu, Cp, and Cpm. These indices are based on a within-subgroup (short-term) estimate of the process standard deviation.
- Ppk, Ppl, Ppu, Pp, and Cpm. These indices are based on an overall (long-term) estimate of the process standard deviation. Note that the process standard deviation does not exist if the process is not stable. See Montgomery (2013).

The Process Capability platform uses the appropriate AIAG notation for capability indices: Ppk labeling denotes an index constructed from an overall variation estimate and Cpk denotes an index constructed from a within-subgroup variation estimate.

Note: The AIAG (Ppk) Labeling platform preference is selected by default. You can change the reporting to use Cp notation only by deselecting this preference under Process Capability.

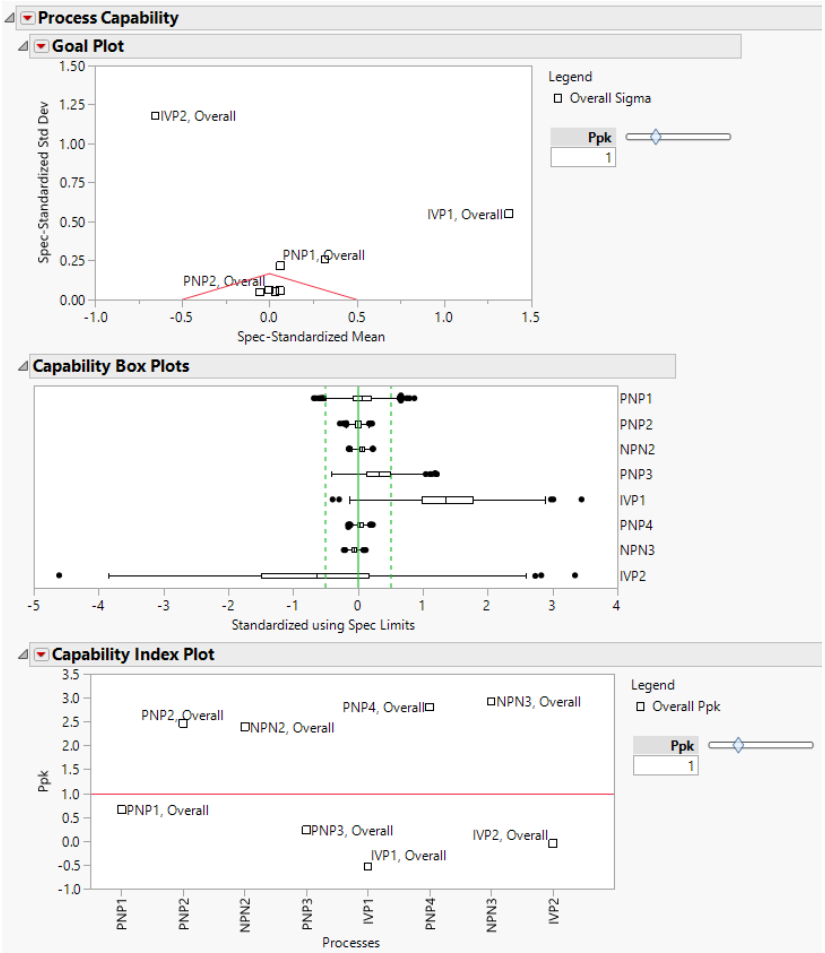
For more information about process capability analysis, see Montgomery (2013) and Wheeler (2004).

Example of the Process Capability Platform with Normal Variables

This example uses the Semiconductor Capability.jmp sample data table. The variables represent standard measurements that a semiconductor manufacturer might make on a wafer as it is being processed. Specification limits for the variables have been entered in the data table through the Column Properties > Spec Limits property.

1. Select **Help > Sample Data Library** and open Semiconductor Capability.jmp.
2. Select **Analyze > Quality and Process > Process Capability**.
3. Click the white triangle next to Processes to view all of the continuous variables.
4. Select PNP1, PNP2, NPN2, PNP3, IVP1, PNP4, NPN3, and IVP2, and click **Y, Process.**
5. Click **OK**.
6. Click the Goal Plot red triangle and select **Label Overall Sigma Points**.
7. Click the Capability Index Plot red triangle and select **Label Overall Sigma Points**.

Figure 7.2 Example Results for Semiconductor Capability.jmp



The Goal Plot shows the spec-normalized mean shift on the x-axis and the spec-normalized standard deviation on the y-axis for each variable. The triangular region defined by the red lines in the bottom center of the plot is the goal triangle. It defines a region of capability index values. You can adjust the goal triangle using the Ppk slider to the right of the plot. When the slider is set to 1, note that PNP1, PNP3, IVP1, and IVP2 are outside of the goal triangle and possibly out of specification.

The Capability Box Plots report shows a box plot for each variable in the analysis. The values for each column are centered by their target value and scaled by the specification range. In this example, all process variables have both upper and lower specification limits, and these are symmetric about the target value. It follows that the solid green line shows where the target should be and the dashed lines represent the specification limits.

It appears that the majority of points for IVP1 are above its upper specification limit (USL), and the majority of points for IVP2 are less than its target. PNP2 seems to be on target with all data values inside the specification limits.

The Capability Index Plot plots the Ppk values for each variable. Four variables come from very capable processes, with Ppk values of 2 or more. Four variables have Ppk values below 1.

Example of the Process Capability Platform with Nonnormal Variables

The Process Measurements.jmp data table contains measurements made on seven different processes used to construct a product. For each process, specification limits are saved as column properties. You begin by examining the distributions of your process data. You see that the distributions are not normal. Then you use the nonnormal capability features of the Process Capability platform to compute capability indices.

View the Distributions

1. Select **Help > Sample Data Library** and open Process Measurements.jmp.
2. Select **Analyze > Distribution**.
3. Select all seven columns from the **Select Columns** list and click **Y, Columns**.
4. Check the box next to **Histograms Only**.
5. Click **OK**.

   For most processes, the histograms show evidence that the theoretical distribution of measurements is skewed and does not follow a normal distribution. Therefore, for each process, you find the best fitting distributions among all of the available parametric distributions.

Perform a Capability Analysis

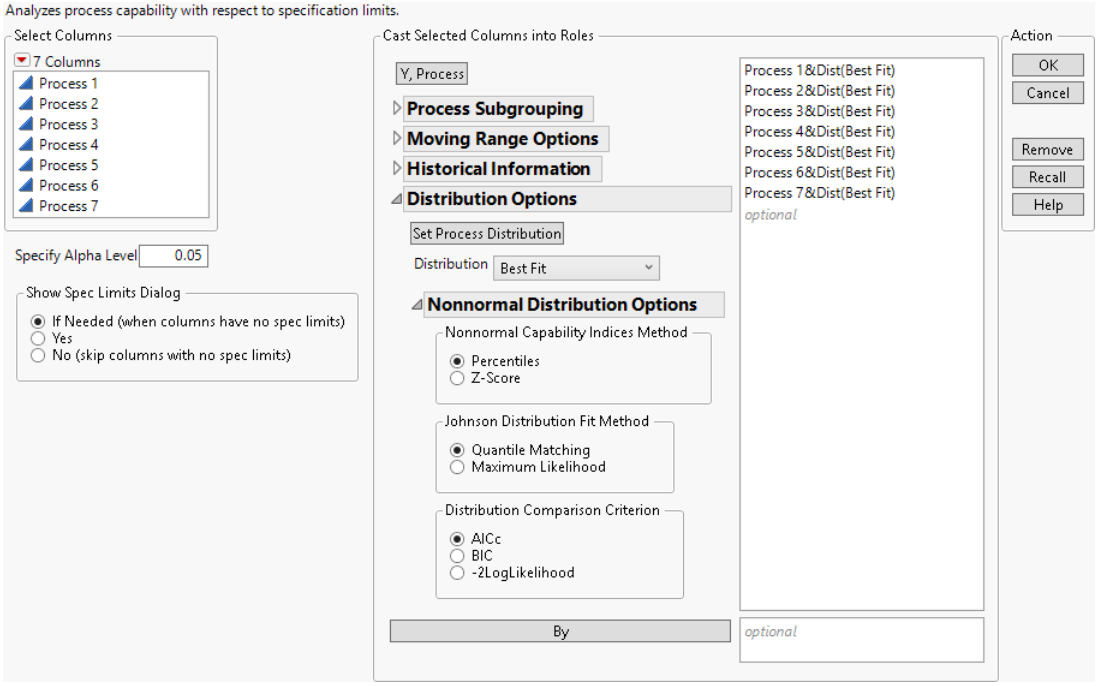
1. Select **Analyze > Quality and Process > Process Capability**.
2. Select all seven columns from the **Columns** list and click **Y, Process**.
3. Select all seven columns in the **Y, Process** list.
4. Open the **Distribution Options** panel and select **Best Fit** from the **Distribution** list.
5. Click **Set Process Distribution**.

   The suffix **&Dist(Best Fit)** is added to each variable name in the Y, Process list. The Best Fit option specifies that the best-fitting parametric distribution should be fit to each variable. The available parametric distributions are Normal, Beta, Exponential, Gamma, Johnson,

Lognormal, Mixture of 2 Normals, Mixture of 3 Normals, SHASH, and Weibull (Figure 7.3).

6. Open the **Nonnormal Distribution Options** outline. Note that the Nonnormal Capability Indices Method is set to **Percentiles**, the Johnson Distribution Fitting Method is set to **Quantile Matching**, and the Distribution Comparison Criterion is set to **AICc**.

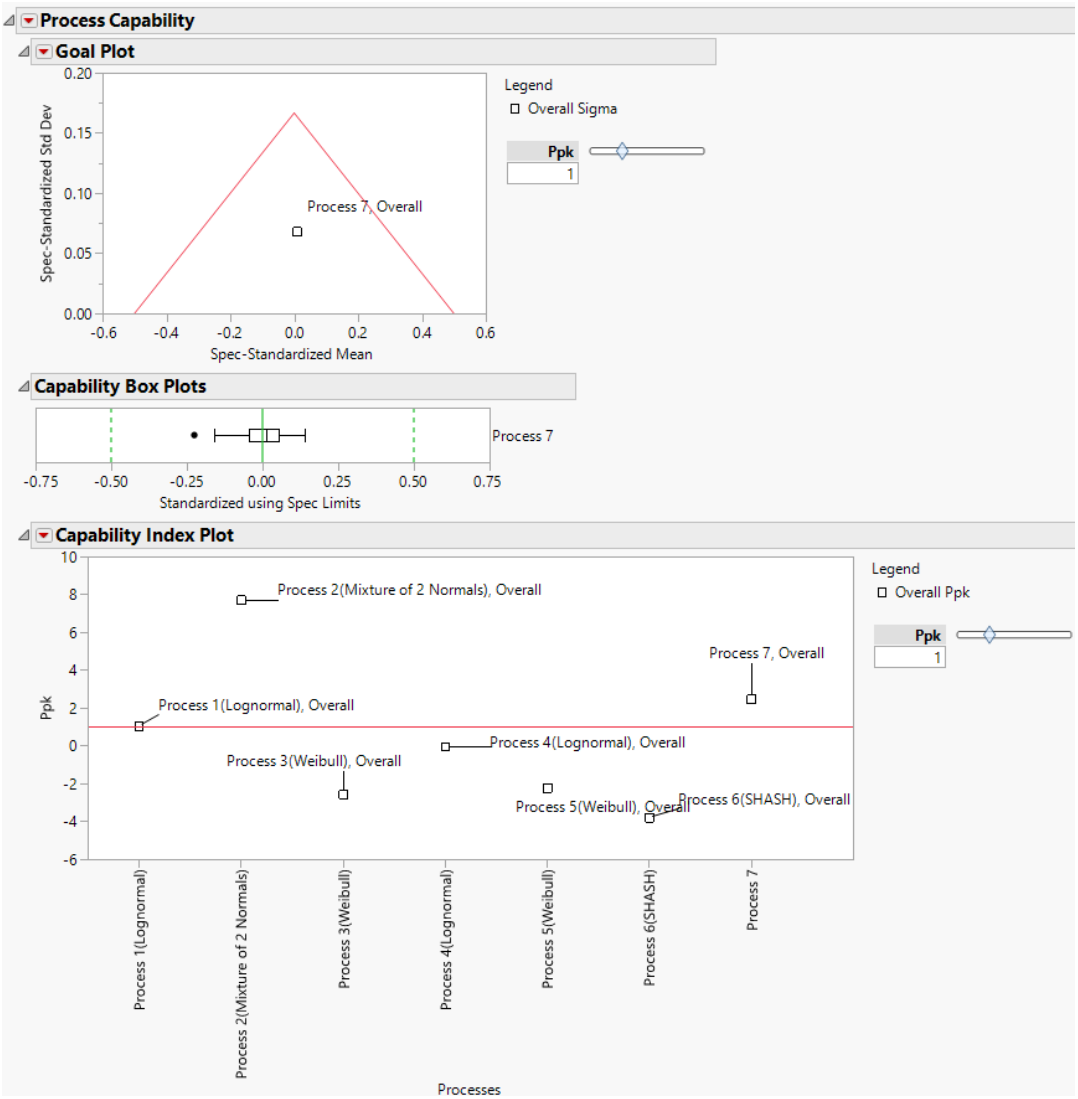
Figure 7.3 Completed Launch Window



The Quantile Matching method is the default method used for fitting Johnson distributions because of its stability and speed as compared to Maximum Likelihood. Note that Maximum Likelihood is used in the Distribution platform.

7. Click **OK**.
8. Click the Goal Plot red triangle and select **Label Overall Sigma Points**.
9. Click the Capability Index Plot red triangle and select **Label Overall Sigma Points**.

Figure 7.4 Initial Report with Variables Labeled



Note: Click a label in the plot and drag it to make the plot more interpretable. Click the right side frame of the Capability Index Plot and drag it to the right to make the labels easier to distinguish.

The Goal Plot shows only one point and it corresponds to Process 7. The Capability Box Plots report shows a single box plot for Process 7. This is because the best fit for Process 7 is a normal distribution.

10. To the right of the Capability Index Plot, set the Ppk value to 2.

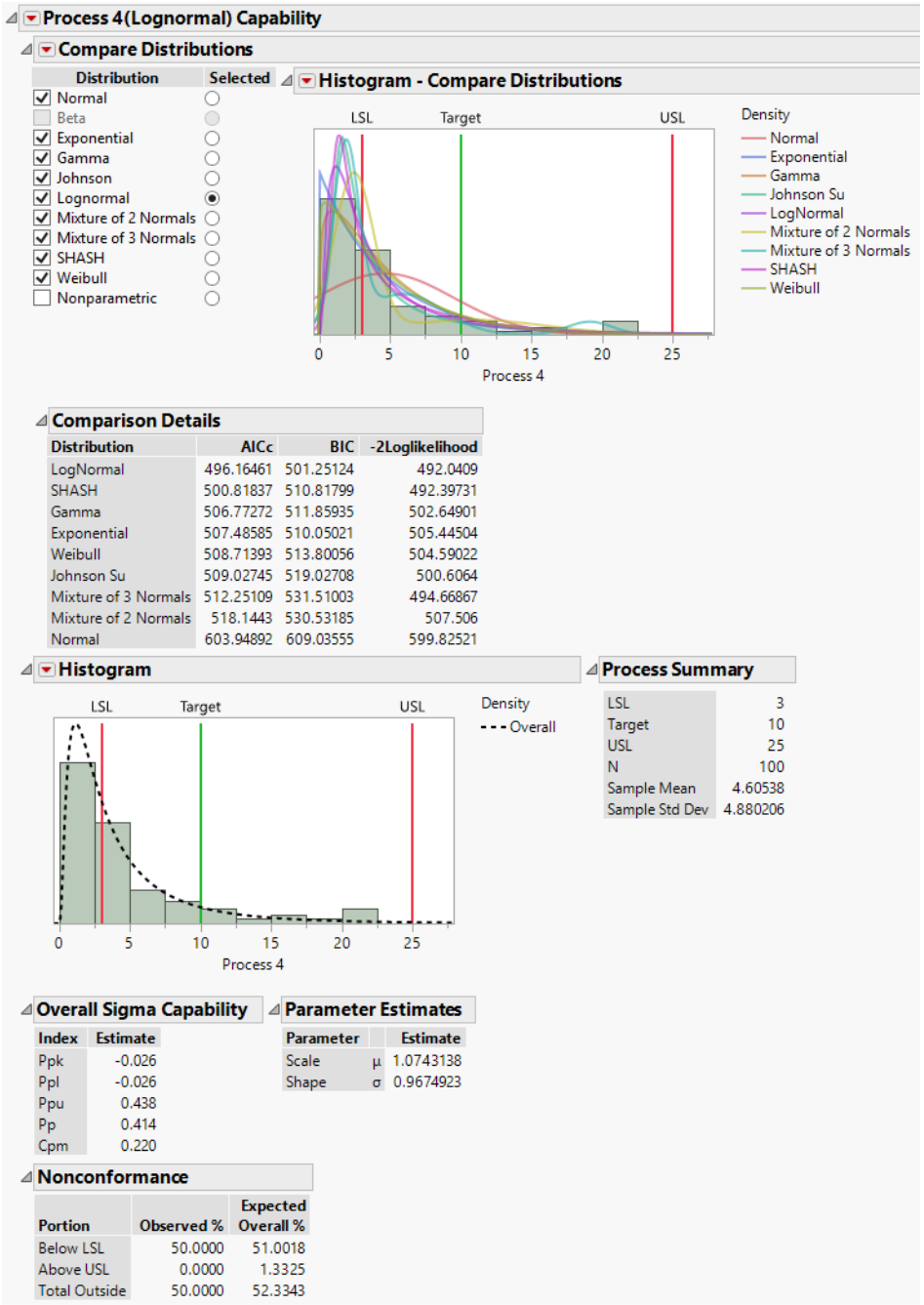
The Capability Index Plot shows Ppk values for all seven processes. Only two processes, Process 2 and Process 7, have capability values that exceed 2. Note that the best fitting nonnormal distributions are shown in parentheses to the right of the variable names in the Capability Index Plot. The best fitting distribution for Process 7 is not shown because it is a normal distribution.

11. Click the Process Capability red triangle and select **Individual Detail Reports**.

    Because you requested Best Fit in the launch window, the Compare Distributions option has been selected from each distribution's red triangle menu.

12. Scroll to the report entitled **Process 4(Lognormal) Capability**.

Figure 7.5 Individual Detail Report for Process 4



The title of the report for Process 4 indicates that the capability calculations are based on a lognormal fit. All of the check boxes in the Compare Distributions report, except the boxes

for Nonparametric and Beta, are checked, indicating that these nine distributions are fit. (This is because you requested a Best Fit in the launch window.) The button that is selected in the Selected column indicates that the Lognormal distribution is the distribution that is used in the remainder of the Process 4(Lognormal) Capability report to estimate capability and nonconformance.

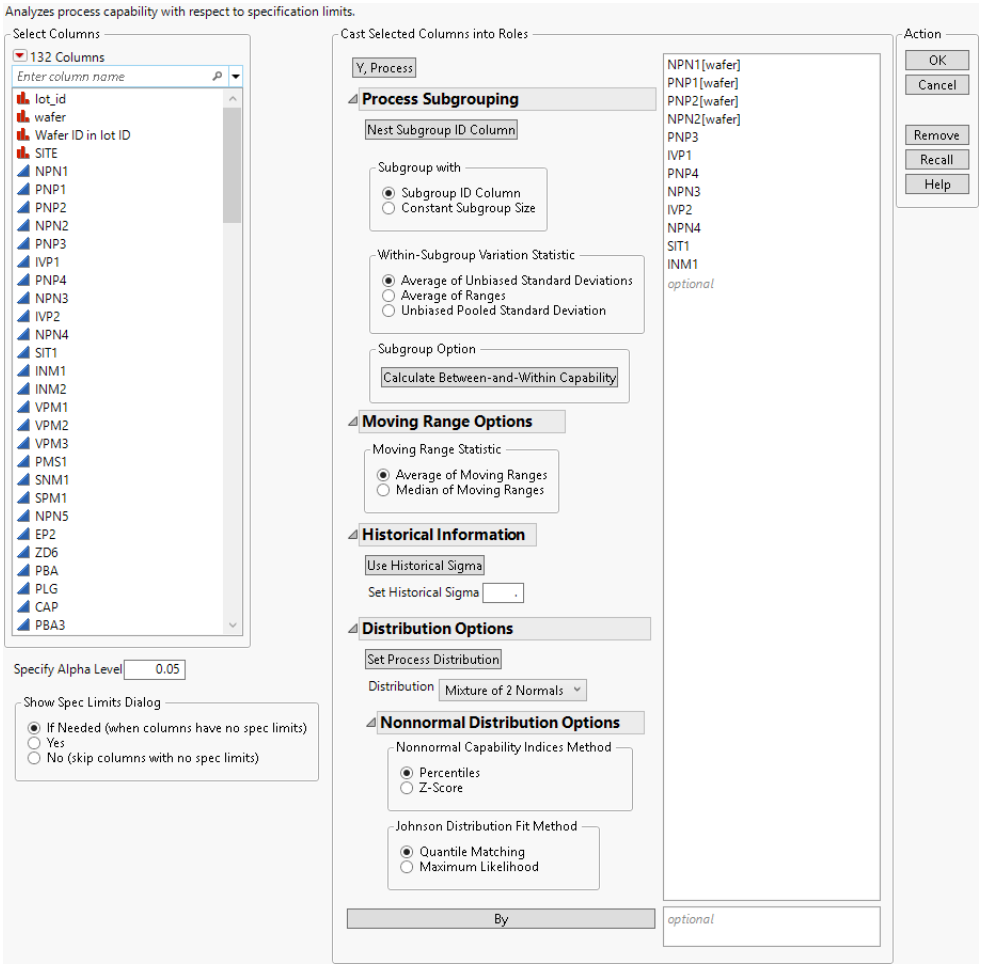
The Compare Distributions report enables you to compare the nine distributional fits. The Histogram - Compare Distributions report gives a visual assessment of the fit and the Comparison Details report shows fit statistics for the selected distributions. Both the plot and the fit statistics indicate that the lognormal distribution gives the best fit among the selected distributions.

The Individual Detail Report information that is shown by default includes a histogram showing the estimated best-fit distribution, a summary of the process information, capability indices based on an overall estimate of sigma, parameter estimates for the fitted lognormal distribution, and observed and expected nonconformance levels.

Launch the Process Capability Platform

Launch the Process Capability Platform by selecting Analyze > Quality and Process > Process Capability. In Figure 7.6, which uses the Semiconductor Capability.jmp data table, all outlines and panels have been opened.

Figure 7.6 Process Capability Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Process Capability launch window contains the following outlines and options:

- "Process Selection"
- "Process Subgrouping"

- "Moving Range Options"
- "Historical Information"
- "Distribution Options"
- "Other Specifications"

After you click OK in the launch window, the Spec Limits window appears unless one of the following occurs:

- All of the columns contain specification limits.
- You selected **No (skip columns with no spec limits)** on the launch window.

The Spec Limits window also appears if you select Yes on the launch window. Otherwise, the Process Capability report window appears.

Process Selection

Select the process variables to include in the capability analysis.

Y, Process Assigns the variables that you want to analyze.

Notes:

- The Transform menu is not available for the Select Column list in the Process Capability launch window. Right-click a column heading in the data table and select **New Formula Column** to create a transform column for use in Process Capability. See *Using JMP* for more information about creating new formula columns.
- Reference columns for virtually joined tables are not available in the Process Capability platform.

Process Subgrouping

This group of options enables you to assign each variable in the Y, Process list a subgroup ID column or a constant subgroup size.

Create Subgroups Using an ID Column

1. Select a variable or variables in the Y, Process list.
2. Select **Subgroup ID Column** from the **Subgroup with** options.
3. Select a subgroup ID column in the Select Columns list.
4. Click **Nest Subgroup ID Column**.

The subgroup ID column appears in brackets to the right of the variable names in the Y, Process list.

Create Subgroups Using a Constant Subgroup Size

1. Select a variable or variables in the Y, Process list.
2. Select **Constant Subgroup Size** from the **Subgroup with** options.
3. Enter the subgroup size next to **Set Constant Subgroup Size**.
4. Click **Subgroup by Size**.

The subgroup size appears in brackets to the right of the variable names in the Y, Process list.

Nest Subgroup ID Column (Available when you select Subgroup ID Column.) Assigns a column that you select from the Select Columns list to define the subgroups for the selected Y, Process columns.

Subgroup by Size (Available when you select Constant Subgroup Size.) Assigns the subgroup size that you specify in the Set Constant Subgroup Size box to define the subgroups for the selected Y, Process columns.

Set Constant Subgroup Size (Available when you select Constant Subgroup Size.) Specify the constant subgroup size for the selected Y, Process columns. You need to assign this value using Subgroup by Size.

Within-Subgroup Variation Statistic (Available when Process Subgrouping is used.) Specifies if the within-subgroup estimate of standard deviation is calculated using standard deviations or ranges.

Calculate Between-and-Within Capability (Available when Process Subgrouping is used.) Specifies that the between-and-within subgroup estimate of the standard deviation should be used in the capability analysis.

Moving Range Options

Use this outline to specify which moving range statistic is used in the within sigma estimate when subgrouping is not used.

Note: When you specify subgrouping and click **Calculate Between-and-Within Capability**, use the Moving Range Options outline to specify which moving range statistic is used in the between sigma estimate.

Average of Moving Range Uses the mean of the moving ranges to estimate sigma. The moving range is the difference between two consecutive points.

Median of Moving Range Uses the median of the moving ranges to estimate sigma.

Historical Information

Use this outline to assign historically accepted values of the standard deviation to variables in the Y, Process list.

1. Select a variable or variables in the Y, Process list.
2. Enter a value next to Set Historical Sigma.
3. Select Use Historical Sigma to assign that value to the selected variables.

The specified value appears in parentheses in the expression "&Sigma()" to the right of the variable names in the Y, Process list.

Note: If you set a historical sigma, then subgroup assignments for the selected process variable are no longer relevant and are removed.

Distribution Options

Unless otherwise specified, all Y, Process variables are analyzed using the assumption that they follow a normal distribution. Use the Distribution Options outline to assign other distributions or calculation methods to variables in the Y, Process list and to specify options related to nonnormal calculations.

- The available distributions are the Normal, Beta, Exponential, Gamma, Johnson, Lognormal, Mixture of 2 Normals, Mixture of 3 Normals, SHASH, and Weibull distributions. Except for Johnson distributions, maximum likelihood estimation is used to fit distributions. See "Johnson Distribution Fit Method".
- The Best Fit option determines the best fit among the available distributions and applies this fit.
- The Nonparametric option fits a distribution using kernel density estimation.

For more options related to nonnormal fits, see "Nonnormal Distribution Options".

Specify a Distribution

1. Select a variable or variables in the Y, Process list.
2. Select a distribution from the Distribution list.
3. Select Set Process Distribution to assign that distribution to the selected variables.

The specified distribution appears in parentheses in the expression "&Dist()" to the right of the variable names in the Y, Process list.

Note: If you select a distribution other than Normal, you cannot assign a Subgroup ID column or a Historical Sigma. These selections are not supported by the methods used to calculate nonnormal capability indices. See "Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods".

Nonnormal Distribution Options

Nonnormal Capability Indices Method Specifies the method used to compute capability indices for nonnormal distributions. See "Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods".

Johnson Distribution Fit Method Specifies the method used to find the best-fitting Johnson distribution. Before estimating the parameters, the best-fitting family of distributions is determined from among the Johnson Su, Sb, and Sl families. The procedure described in Slifker and Shapiro (1980) is used to find the best-fitting family.

Quantile Matching The default method. It is more stable and faster than Maximum Likelihood. Quantile Matching Parameter estimates, assuming the best-fitting family, are obtained using a quantile-matching approach. See Slifker and Shapiro (1980).

Maximum Likelihood Parameters for the best-fitting family are determined using maximum likelihood.

Distribution Comparison Criterion (Available when a Best Fit Distribution is selected.) Specify the criterion that you want to use in determining a Best Fit. This criterion also determines the ordering of distributions in the Comparison Details report. See "Order by Comparison Criterion".

Other Specifications

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Specify Alpha Level Specifies the significance level for confidence limits.

Show Spec Limits Dialog Specifies how to handle columns that do not have specification limits.

Note: It is good practice to ensure that specification limits for all process variables are specified as Spec Limits column properties or to load specification limits from a Limits Data table (see "Limits Data Table"). Otherwise, you can specify limits interactively in the Spec Limits window that appears after you click OK in the launch window (unless you select **No (skip columns with no spec limits)** on the launch window).

Entering Specification Limits

The lower specification limit (LSL), upper specification limit (USL), and target define the lower bound, upper bound, and target value for a quality process.

There are several ways to enter specification limits:

- Enter limits in the Spec Limits window after selecting columns in the launch window. See "Spec Limits Window".
- Import limits from a JMP data table (known as a Limits Table). See "Limits Data Table".
- Enter limits as Spec Limits column properties in the data table. See "Spec Limits Column Property".
- If you are creating a Process Capability report by running a JSL script, enter limits in the script. See "The Process Capability Report".

Only one specification limit is required for a selected column. If only the USL is specified, the box plots and Goal Plot point are colored blue. If only the LSL is specified, the box plots and Goal Plot point are colored red.

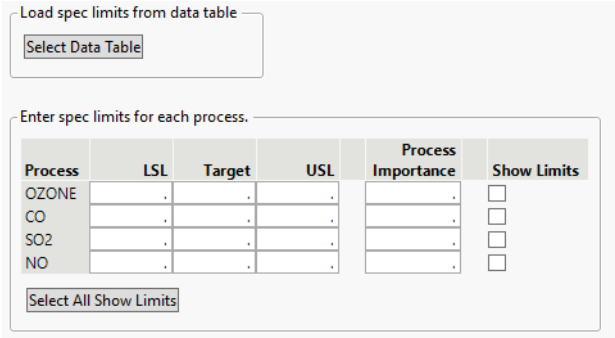
Spec Limits Window

After you click OK on the launch window, the Spec Limits window appears if any of the columns do not contain limits and you did not select **No (skip columns with no spec limits)** on the launch window. The Spec Limits window also appears if you select Yes on the launch window. Figure 7.7 shows the Spec Limits window for the Cities.jmp sample data table after selecting OZONE, CO, SO2, and NO as process variables in the launch window. Enter the known specification limits and click OK to view the Process Capability report.

You can specify process importance values for each column. Process importance values provide a mechanism to sort processes in the order that you prefer. Process importance values are used to size markers in many of the graphs in the Process Capability report.

If you select the Show Limits option for a process and then save the specification limits to a column property, the Show as Graph Reference Lines option is selected in the saved Spec Limits column property. If you select the Show Limits option for a process and then save the specification limits to a new table, the Show Limits column in the new table contains a 1 for the process. The **Select All Show Limits** button selects the Show Limits option for all processes.

Figure 7.7 Spec Limits Window for Cities.jmp



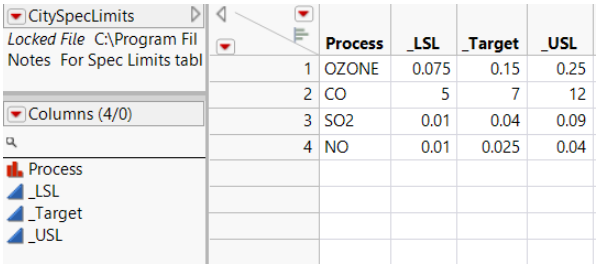
Limits Data Table

You can also specify a limits data table with the **Load spec limits from data table** option from the Spec Limits window. Click the **Select Data Table** button and then select the appropriate data table that contains the specification limits for the analysis. After you select the appropriate limits table, the values populate the window. Click **OK** to view the Process Capability report.

A limits data table can be in two different formats: *tall* or *wide*. A tall limits data table has one column for the responses and the limits key words are the other columns. A wide limits data table has a column for each response with one column to label the limits keys. Either of these formats can be read using the **Load spec limits from data table** option.

- A tall table contains four or five columns and has one row for each process. The first column has a character data type and contains the names of the columns analyzed in the Process Capability platform. The next three columns need to be named LSL, Target, and USL. These column names can also be preceded by an underscore character. The optional final column named Show Limits specifies if the specification limits are shown as reference lines in select analysis plots.

Figure 7.8 Example of a Tall Specification Limits Table


CitySpecLimits
Locked File C:\Program Fil
Notes For Spec Limits tabl
Columns (4/0)
Process
_LSL
_Target
_USL

	Process	_LSL	_Target	_USL
1	OZONE	0.075	0.15	0.25
2	CO	5	7	12
3	SO2	0.01	0.04	0.09
4	NO	0.01	0.025	0.04

- A wide table contains three rows and one column for each column analyzed in the Process Capability platform plus a _LimitsKey column. In the _LimitsKey column, the three rows need to contain the identifiers _LSL, _Target, and _USL.

Figure 7.9 Example of a Wide Specification Limits Table

	_LimitsKey	OZONE	CO	SO2	NO
1	_LSL	0	5	0	0
2	_Target	0.05	10	0.03	0.025
3	_USL	0.1	20	0.08	0.6

The easiest way to create a limits data table is to save results computed by the Process Capability platform. The Save options in the Process Capability red triangle menu enable you to save limits from the sample values. After entering or loading the specification limits, you can do the following:

- Select **Save > Save Spec Limits as Column Properties** to save the limits as Spec Limits column properties to the columns in the data table.
- Select **Save > Save Distributions as Column Properties** to save the distributions used in calculating capability as Process Capability Distribution column properties to the columns in the data table.
- Select **Save > Save Spec Limits to New Table** to save the limits to a new tall specification limits data table. If you have selected at least one nonnormal distribution, a column called Distribution that contains the specified distributions is also added to the limits data table.

See "Process Capability Platform Options".

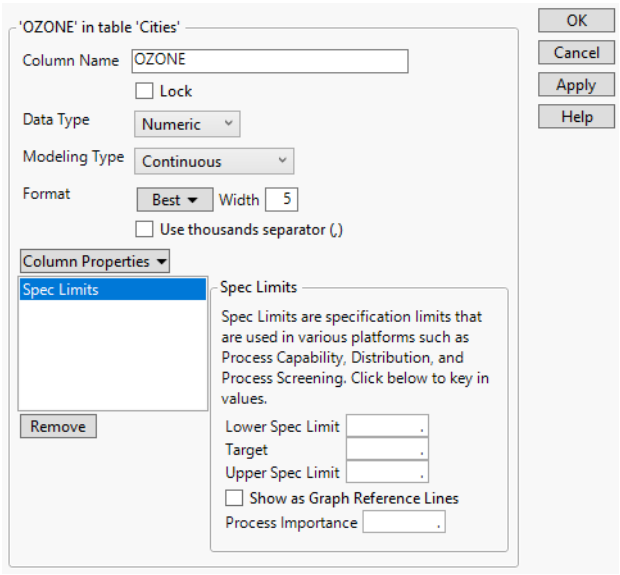
Spec Limits Column Property

When you perform a capability analysis, you can use Column Properties > Spec Limits to save specification limits as a column property. The Spec Limits property applies only to numeric columns.

Some processes have one-sided specifications. Some have no target. You can enter any of these that apply: a lower specification limit, an upper specification limit, a target value, or a process importance value.

Figure 7.10 displays the Spec Limits section of the Column Properties window for OZONE in the sample data table Cities.jmp.

Figure 7.10 Spec Limits Section of the Column Properties Window



Tip: Saving specification limits as a column property ensures consistency when you repeat an analysis.

The Process Capability Report

By default, the Process Capability platform provides the following reports:

- "Goal Plot" (provided only if at least one variable is fit with a normal distribution and shows only points for variables fit with normal distributions)

- "Capability Box Plots" (provided only if at least one variable is fit with a normal distribution and shows only box plots for variables fit with normal distributions)
- "Capability Index Plot"

Figure 7.2 shows an example of a default Process Capability report.

Using the Process Capability red triangle menu, you can add individual detail reports, normalized box plots, and summary reports. The red triangle menu also has options for identifying out-of-spec values in your data table, creating a summary data table, changing the display order of analyzed columns, and saving out spec limits. These options are described in "Process Capability Platform Options".

You can change the default report at File > Preferences > Platforms > Process Capability. You can also make changes to the appearance of reports produced by options by selecting the relevant Process Capability topic at File > Preferences > Platforms.

Goal Plot

The Goal Plot shows, for each variable, the spec-normalized mean shift on the *x*-axis, and the spec-normalized standard deviation on the *y*-axis. It is useful for getting a quick, summary view of how the variables are conforming to specification limits. By default, the Goal Plot shows only those points for each column that are calculated using the overall sigma. Hover over each point to view the variable name and the sigma method used to calculate the point. See "Goal Plot" for more information about the calculation of the coordinates for the Goal Plot.

Note: Process variables with distributions other than Normal are not plotted on the Goal Plot.

Goal Plot Points

Points on the Goal Plot correspond to columns, not rows. Selecting a point in the Goal Plot selects the corresponding column in the data table. If process importance values are specified, the goal plot points are sized by importance.

Hover over a point in the Goal Plot to view a control chart for that process. Click the control chart to launch Control Chart Builder with the corresponding control chart and capability report.

Note: A control chart is not available for a process if the unbiased pooled standard deviation is chosen as the within-group variation statistic for that process.

The points on the Goal Plot are also linked to the rows of the Goal Plot Summary Table, where each row corresponds to a column. You can select a point in the Goal Plot, right-click, and apply row states. These row states are applied to the rows of the Goal Plot Summary Table. Row states that you apply in the Goal Plot Summary Table are reflected in the Goal Plot. To see this table, select Make Goal Plot Summary Table from the Process Capability red triangle menu. See "Make Goal Plot Summary Table".

Tip: If you hide a point in the Goal Plot, you can show the point again by changing the corresponding row state in the Goal Plot Summary Table.

Goal Plot Triangle

The goal plot triangle appears in the center of the bottom of the Goal Plot. The slider to the right of the plot enables you to adjust the size of goal triangle in the plot.

By default, the Ppk slider and the value beneath it are set to Ppk = 1. This approximates a non-conformance rate of 0.0027, if the distribution is normal. The goal triangle represents the Ppk shown in the box. To change the Ppk value, move the slider or enter a number in the box.

JMP gives the Goal Plot in terms of Ppk values by default. You can change this preference at File > Preferences > Platforms > Process Capability. When the AIAG (Ppk) Labeling preference is unchecked, all of the Ppk labeling is changed to Cpk labeling, including the label of the slider to the right of the goal plot.

Goal Plot Options

The Goal Plot red triangle menu has the following options:

Show Within Sigma Points Shows or hides the points calculated using the within sigma estimate.

Show Within or Between-and-Within Sigma Points (Available only when Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides the points calculated using the within sigma estimate or, if specified, the between-and-within sigma estimate.

Show Overall Sigma Points Shows or hides the points calculated using the overall sigma estimate.

Shade Levels Shows or hides the Ppk level shading (Figure 7.11). When you select Shade Levels, shaded areas appear in the plot. The shaded areas depend on the relationship between p and Ppk, with p representing the value shown in the box beneath Ppk:

- Points in the red area have Ppk < p.
- Points in the yellow area have p < Ppk < 2p.

– Points in the green area have $2p < \text{Ppk}$.

Label Within Sigma Points Shows or hides labels for points calculated using the within sigma estimate.

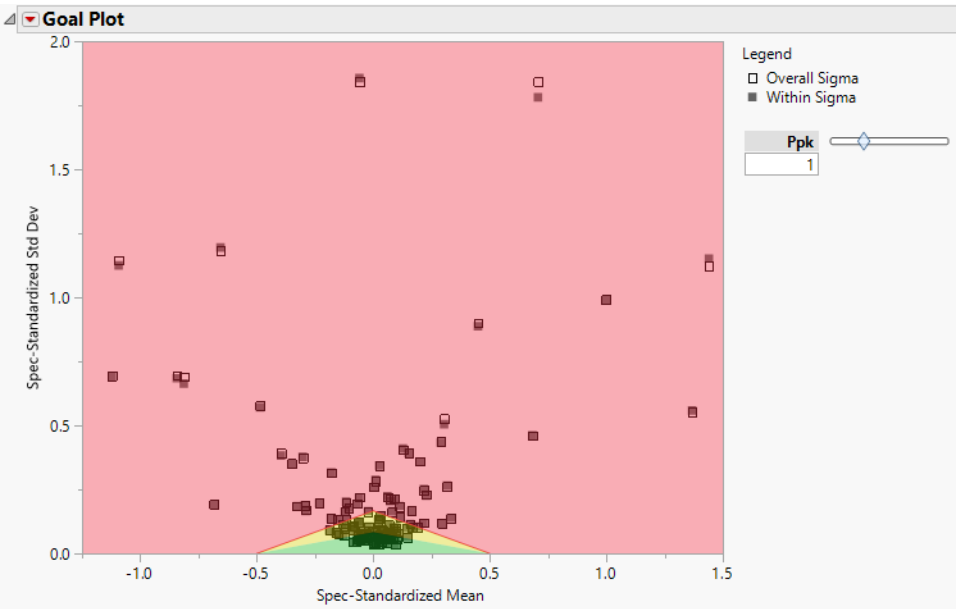
Label Within or Between-and-Within Sigma Points (Available only when Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides labels for points calculated using the within sigma estimate or, if specified, the between-and-within sigma estimate.

Label Overall Sigma Points Shows or hides labels for points calculated using the overall sigma estimate.

Defect Rate Contour Shows or hides a contour representing a specified defect rate.

Figure 7.11 shows the Goal Plot for the entire data set for the Semiconductor Capability.jmp sample data table after selecting Shade Levels and Show Within Sigma Points from the Goal Plot red triangle menu.

Figure 7.11 Goal Plot



One-Sided or Missing Specification Limits

When there is only one specification limit for a column, markers and colors are used in the following ways:

- If only the upper specification limit (USL) is specified, the point on the Goal Plot is represented by a right-pointing triangle and is colored blue.

- If only the lower specification limit (LSL) is specified, the point on the Goal Plot is represented by a left-pointing triangle and is colored red.
- If at least one process has only an upper specification limit, the right half of the goal triangle is blue.
- If at least one process has only a lower specification limit, the left half of the goal triangle is red.

Processes with only an upper specification limit are represented by blue and should be compared to the blue (right) side of the goal triangle. Processes with only a lower specification limit are represented by red and should be compared to the red (left) side of the goal triangle. For more information about how the coordinates of points are calculated, see "Goal Plot".

Capability Box Plots

The Capability Box Plots show a box plot for each variable selected in the analysis. The values for each column are centered by their target value and scaled by the difference between the specification limits. If the target is not centered between the specification limits, the values are scaled by twice the minimum difference between the target and specification limits. For each process column Y_j (see "Notation for Goal Plots and Capability Box Plots" for a description of the notation):

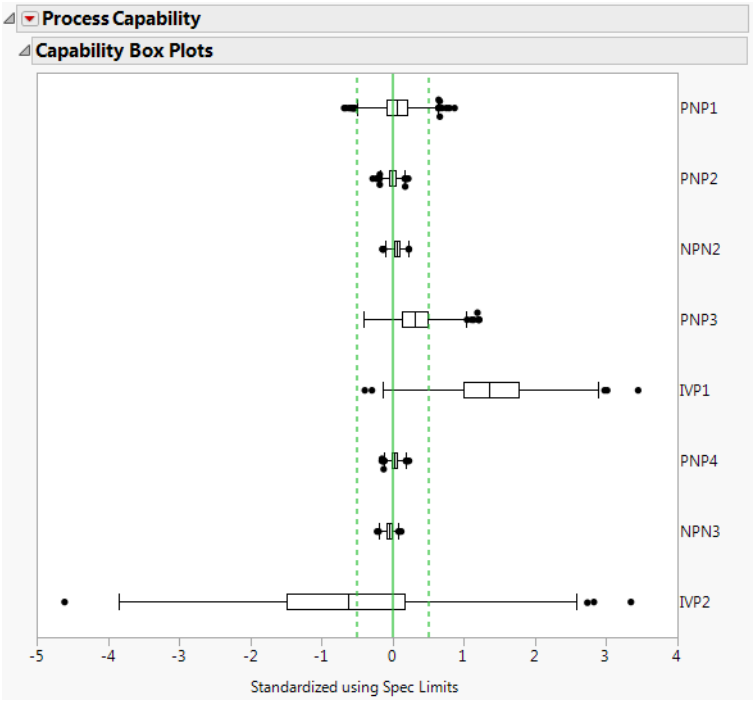
$$Z_{ij} = \frac{Y_{ij} - T_j}{2 \times \min(T_j - \mathrm{LSL}_j, \mathrm{USL}_j - T_j)}$$

For a process with a one-sided specification, see "One-Sided or Missing Specification Limits". For the situation where no target is specified, see "Capability Box Plots for Processes with Missing Targets".

Note: Process variables with distributions other than Normal are not plotted on the Capability Box Plot.

Figure 7.11 shows a Capability Box Plots report for eight variables in the Semiconductor Capability.jmp sample data table.

Figure 7.12 Capability Box Plot



The plot displays dotted green lines drawn at ±0.5.

- For a process with a target that is centered between its specification limits, the dotted green lines represent the standardized specification limits.
- For a process with a target that is not centered between its specification limits, one of the dotted green lines represents the standardized specification limit for the limit closer to the target. The other dotted green line represents the same distance in the opposite direction.

This plot is useful for comparing variables with respect to their specification limits. For example, in Figure 7.12, the majority of points for IVP1 are above its USL, and the majority of its points for IVP2 are less than its target. PNP2 seems to be on target with all data points in the specification limits.

One-Sided or Missing Specification Limits

When there is only one specification limit for a column, colors are used in the following ways:

- If only the upper specification limit (USL) is specified, the box plot is colored blue.
- If only the lower specification limit (LSL) is specified, the box plot is colored red.
- If at least one process has only an upper specification limit, the dotted line at 0.5 is blue.
- If at least one process has only a lower specification limit, the dotted line at -0.5 is red.

Suppose that only the lower specification limit is specified and that the process target is specified. The capability box plot is based on the following values for the transformed observations. See "Notation for Goal Plots and Capability Box Plots" for a description of the notation:

$$Z_{ij} = \frac{Y_{ij} - T_j}{2(T_j - \mathrm{LSL}_j)}$$

Suppose that only the upper specification limit is specified and that the process target is specified. The capability box plot is based on the following values for the transformed observations:

$$Z_{ij} = \frac{Y_{ij} - T_j}{2(\mathrm{USL}_j - T_j)}$$

For more information about how missing targets are handled with one-sided specification limits, see "Single Specification Limit and No Target".

Capability Index Plot

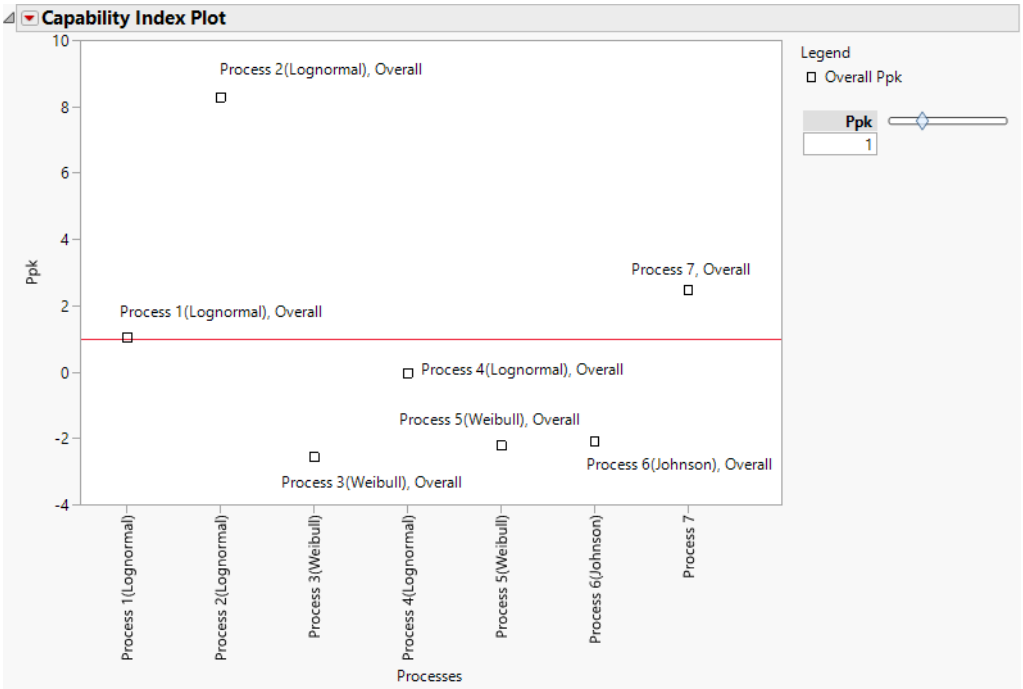
The Capability Index Plot shows Ppk values for all variables that you entered as Y, Process. Each variable name appears on the horizontal axis and the Ppk values appear on the vertical axis. If you fit a nonnormal distribution, the fitted distribution name appears in the plot as a parenthetical suffix to the variable name. If process importance values are specified, the points on the capability index plot are sized by importance. A horizontal line is placed at the Ppk value that is specified by the slider to the right of the plot.

Hover over a point in the Capability Index Plot to view a control chart for that process. Click the control chart to launch Control Chart Builder with the corresponding control chart and capability report.

Note: A control chart is not available for a process if the unbiased pooled standard deviation is chosen as the within-group variation statistic for that process.

Figure 7.13 shows a Capability Index Plot report for the Process Measurements.jmp sample data table. Six of the variables are fit with nonnormal distributions. Process 7 is fit with a normal distribution. Points have been labeled using the Label Overall Sigma Points option that is available in the Capability Index Plot red triangle menu.

Figure 7.13 Capability Index Plot with Nonnormal Distributions



Capability Index Plot Options

The Capability Index Plot red triangle menu has the following options:

Show Within Sigma Points Shows or hides the points calculated using the within sigma estimate.

Show Within or Between-and-Within Sigma Points (Available only when Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides the points calculated using the within sigma estimate or, if specified, the between-and-within sigma estimate.

Show Overall Sigma Points Shows or hides the points calculated using the overall sigma estimate.

Shade Levels Shows or hides the Ppk level shading. When you select Shade Levels, shaded areas appear in the plot. The shaded areas depend on the relationship between p and Ppk, with p representing the value shown in the box beneath Ppk:

- Points in the red area have Ppk < p.
- Points in the yellow area have p < Ppk < 2p.
- Points in the green area have 2p < Ppk.

Label Within Sigma Points Shows or hides labels for points calculated using the within sigma estimate.

Label Within or Between-and-Within Sigma Points (Available only when Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides labels for points calculated using the within sigma estimate or, if specified, the between-and-within sigma estimate.

Label Overall Sigma Points Shows or hides labels for points calculated using the overall sigma estimate.

Process Capability Platform Options

The Process Capability red triangle menu contains the following options:

Individual Detail Reports Shows or hides individual detail reports for each variable in the analysis. See "Individual Detail Reports".

Goal Plot Shows or hides a goal plot for the data. The Goal Plot shows the spec-normalized mean shift on the x-axis and the spec-normalized standard deviation on the y-axis for each variable. See "Goal Plot". (Only variables for which you specify normal distributions are shown on the plot.)

Capability Box Plots Shows or hides a capability box plot for each variable in the analysis. The values for each column are centered by their target value and scaled by twice the minimum difference between the target value and the specification limits. See "Capability Box Plots". (Box plots are shown only for variables for which you specify normal distributions.)

Normalized Box Plots Provides two options for plots that show normalized box plots for each process variable. Each column is standardized by subtracting its mean and dividing by an estimate of the column's standard deviation. The box plot is constructed using quantiles for the standardized values. See "Normalized Box Plots". (Normalized box plots are shown only for variables for which you specify normal distributions.)

Within Sigma Normalized Box Plots Shows or hides a plot called Within Sigma Normalized Box Plots. The box plots are constructed using the within-subgroup estimate of standard deviation.

Within or Between-and-Within Sigma Normalized Box Plots (Available only when Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides a plot called Within or Between-and-Within Normalized Box Plots. The box plots are constructed using the within group estimate of the standard deviation or, if specified, the between-and-within estimate.

Overall Sigma Normalized Box Plots Shows or hides a plot called Overall Sigma Normalized Box Plots. The box plots are constructed using the overall estimate of standard deviation.

Capability Index Plot Shows overall Ppk values for all variables that you entered as Y, Process. See "Capability Index Plot".

Process Performance Plot Shows or hides a four-quadrant plot of capability versus stability. Each process that has at least one specification limit is represented by a point. See "Process Performance Plot".

Summary Reports Provides two options for summary reports of capability indices. See "Summary Reports".

Within Sigma Summary Report Shows or hides a summary report of capability indices calculated using the within-subgroup estimate of standard deviation. (Results are available only for variables with specified normal distributions.)

Within or Between-and-Within Sigma Summary Report (Available only when Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides a summary report of capability indices calculated using the within group estimate of the standard deviation or, if specified, the between-and-within group estimate.

Overall Sigma Summary Report Shows or hides a summary report of capability indices calculated using the overall estimate of standard deviation.

Action Options

The following red triangle menu options perform actions:

Out of Spec Values Provides options for the cells in the data table containing values that are out of spec.

Select Out of Spec Values Selects all rows and columns in the data table that contain at least one value that does not fall within the specification limits.

Color Out of Spec Values Colors the cells in the data table that correspond to values that are out of spec. The cell is colored blue if the value is above the USL and red if the value is below the LSL.

Tip: To remove colors in specific cells, select all cells of interest. Right-click in one of the cells and select Clear Color. To remove colors in all cells, deselect Color Out of Spec Values.

Make Goal Plot Summary Table Creates a summary table for the points plotted in the Goal Plot. This table includes the variable's name, its spec-normalized mean shift, and its

spec-normalized standard deviation. Each variable has two rows in this table: one for each sigma type (within and overall). See "Make Goal Plot Summary Table".

Order By Reorders the box plots, summary reports, and individual detail reports. You can reorder by Initial Order, Reverse Initial Order, Within Sigma Cpk Ascending, Within or Between-and-Within Sigma Cpk Ascending, Within Sigma Cpk Descending, Within or Between-and-Within Sigma Cpk Descending, Overall Sigma Ppk Ascending, or Overall Sigma Ppk Descending. The options that order by Within Sigma reorder plot elements only for variables with specified normal distributions.

Note: The options to order by Within or Between-and-Within Sigma are available only if Calculate Between-and-Within Capability is selected for at least one process in the launch window.

Save Provides options for saving specification limits and distributions.

Save Spec Limits as Column Properties Saves the specification limits to a column property for each variable in the analysis. If no Spec Limits column property is present, the column property is created. If a Spec Limits column property is present, the values in the column property are overwritten. See "Spec Limits Column Property".

Save Distributions as Column Properties Saves the distribution used in calculating capability as a Process Capability Distribution column property. See *Using JMP*.

If a column contains the Distribution property specifying a nonnormal distribution and no Process Capability Distribution property, then the Process Capability platform applies a nonnormal fit. The Process Capability platform uses the distribution specified in the Distribution column property, or a Johnson fit if that distribution is not supported in Process Capability. If a column contains the Process Capability Distribution property, then the Process Capability platform uses the distribution specified in the Process Capability Distribution column property.

Note: If you want to use a specific distribution in the Process Capability platform, save it as a Process Capability Distribution column property.

Save Spec Limits to New Table Saves the specification limits and the setting for Show Limits for each process to a limits data table in tall format. See "Limits Data Table".

Relaunch Dialog Opens the platform launch window and recalls the settings used to create the report.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

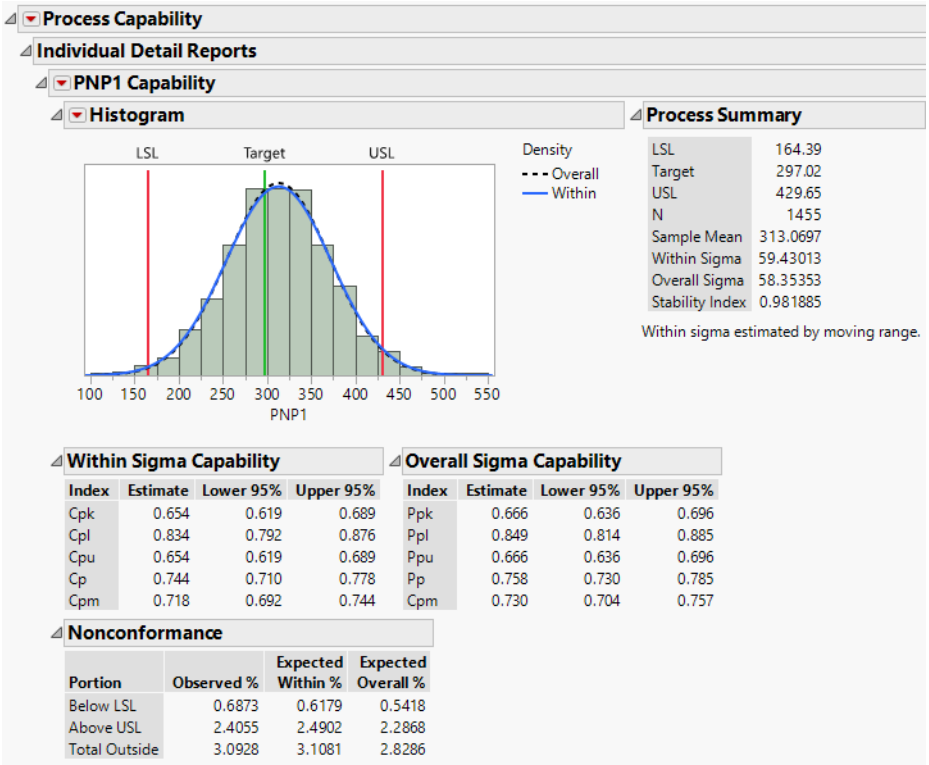
Individual Detail Reports

The Individual Detail Reports option displays a capability report for each variable in the analysis.

Normal Distributions

Figure 7.14 shows the Individual Detail Report for PNP1 from the Semiconductor Capability.jmp sample data table as described in "Example of the Process Capability Platform with Normal Variables".

Figure 7.14 Individual Detail Report



The Individual Details report for a variable with a normal distribution shows a histogram, process summary details, and capability and nonconformance statistics. The histogram shows the distribution of the values, the lower and upper specification limits and the process target (if they are specified), and one or two curves showing the assumed distribution. The histogram in Figure 7.14 shows two normal curves, one based on the overall estimate of standard deviation and the other based on the within-subgroup estimate.

When you fit your process with a normal distribution, the Process Summary includes the *Stability Index*, which is a measure of stability of the process. The stability index is defined as follows:

(Overall Sigma/Within Sigma)

If Calculate Between-and-Within Capability is specified for a process in the launch window, the stability index for that process is defined as follows:

(Overall Sigma/Between-and-Within Sigma)

A stable process has stability index near one. Higher values indicate less stability.

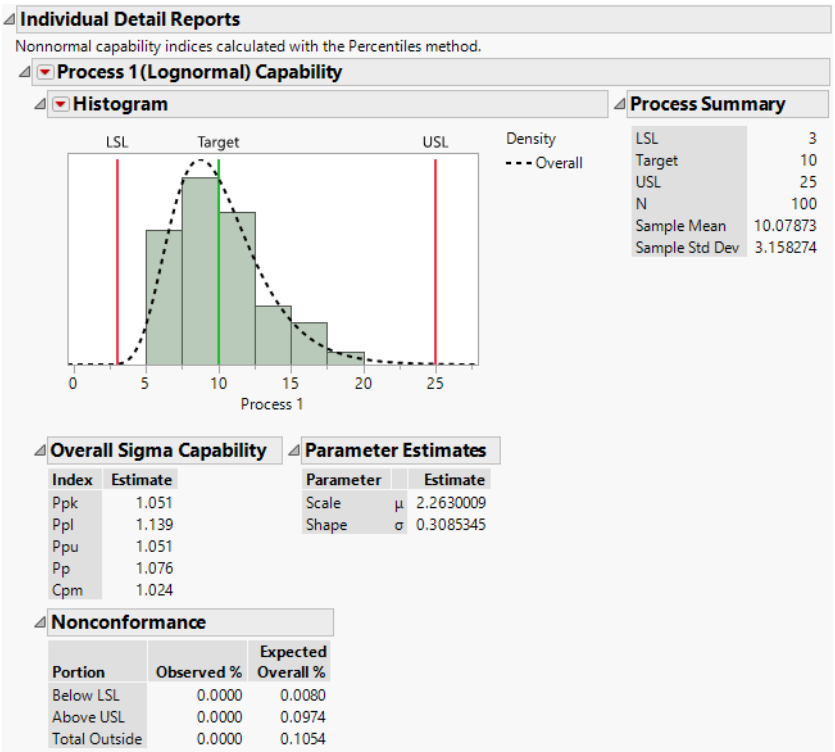
Note: You can change the preferences for stability assessment type in File > Preferences > Platforms > Process Capability. This changes the stability assessment type used through the Process Capability platform.

Nonnormal Distributions

Note: Capability indices based on within-subgroup variation and stability indices are not available for processes for which you have specified nonnormal distributions.

Figure 7.15 shows the Individual Detail Report for Process 1 from the Process Measurements.jmp sample data table as described in "Example of the Process Capability Platform with Nonnormal Variables".

Figure 7.15 Individual Detail Report for Process 1



The report opens with a note summarizing the Nonnormal Distribution Options that you selected in the launch window.

The Individual Details report for a variable with a nonnormal distribution shows a histogram, process summary details, and capability and nonconformance statistics. The histogram shows the distribution of the values, the lower and upper specification limits and the process target (if they are specified). A curve showing the fitted distribution is superimposed on the histogram. If you selected a Nonparametric distribution, the curve shown in the histogram is the nonparametric density.

The report also shows a Parameter Estimates report if you selected a nonnormal parametric distribution or a Nonparametric Density report if you selected a Nonparametric fit. See "Parameter Estimates" and "Nonparametric Density".

Individual Detail Report Options

The outline title for each variable in the Individual Detail Reports section is of the form <Variable Name> Capability. However, if you request nonnormal capability, the relevant distribution name is shown parenthetically in the outline title.

Each Capability report has a red triangle menu with the following options:

Compare Distributions Shows or hides the control panel for comparing distributions for the process. See "Compare Distributions".

Process Summary Shows or hides the summary statistics for the variable, including the overall sigma estimate, and, if you have specified a normal distribution, the within sigma estimate and the stability index. If you have specified Calculate Between-and-Within Capability for at least one process in the launch window, estimates for the between sigma and the between-and-within sigma are also included.

Histogram Shows or hides the histogram of the values of the variable. The histogram report includes a red triangle menu that controls the following features of the histogram:

Show Spec Limits Shows or hides vertical red lines on the histogram at the specification limits for the process.

Show Target Shows or hides a vertical green line on the histogram at the process target.

Show Within Sigma Density Shows or hides an approximating normal density function on the histogram with mean given by the sample mean and standard deviation given by the within estimate of sigma.

Show Between-and-Within Sigma Density (Available only when Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides an approximating normal density function on the histogram with mean given by the sample mean and standard deviation given by the between-and-within estimate of sigma.

Show Overall Sigma Density Shows or hides an approximating normal density function on the histogram with mean given by the sample mean and standard deviation given by the overall estimate of sigma.

Show Count Axis Shows or hides an additional axis to the right of the histogram plot showing the count of observations.

Show Density Axis Shows or hides an additional axis to the right of the histogram plot showing the density.

Capability Indices Controls display of the following capability index reports:

Within Sigma Capability (Available when distribution is Normal.) Shows or hides capability indices (and confidence intervals) based on the within (short-term) sigma.

Between-and-Within Sigma Capability (Available only when distribution is Normal and Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides capability indices based on the between-and-within sigma.

Within Sigma Z Benchmark (Available when distribution is Normal.) Shows or hides Z benchmark indices based on the within (short-term) sigma.

Between-and-Within Sigma Z Benchmark (Available only when distribution is Normal and Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides Z benchmark indices based on the between-and-within sigma.

Within Sigma Target Index (Available when distribution is Normal.) Shows or hides an estimate of the target index that is based on the within (short-term) sigma.

Between-and-Within Sigma Target Index (Available only when distribution is Normal and Calculate Between-and-Within Capability is selected for at least one process in the launch window.) Shows or hides an estimate of the target index that is based on the between-and-within sigma.

Overall Sigma Capability Shows or hides capability indices (and confidence intervals) based on the overall (long-term) sigma.

Overall Sigma Z Benchmark (Available when distribution is Normal.) Shows or hides Z benchmark indices based on the overall (long-term) sigma.

Note: By default, the confidence intervals for the capability indices are constructed based on $\alpha = 0.05$. To change the default confidence level, select File > Preferences > Platforms > Process Capability.

Nonconformance Shows or hides the observed and expected percentages of observations below the LSL, above the USL, and outside of the specification limits. The Nonconformance table contains hidden columns for observed and expected PPM and counts.

Interactive Capability Plot Shows or hides the Interactive Capability Plot. The Interactive Capability Plot enables you to change the value of one or more summary statistics and see how the changes affect the capability analysis. There are Original and New reports that show the original and new summary statistics, capability indices, and expected PPM. Use the slider controls or text boxes to change the spec limits, mean, and overall sigma from the original values. You can also use the Mean Shift box to shift the mean by a factor of the original sigma. The Interactive Capability Plot report has the following red triangle menu options:

Capability Shows or hides the capability indices in the Original and New reports.

Expected PPM Shows or hides the expected PPM values in the Original and New reports.

Revert to Original Values Reverts the interactive capability plot and the summary values in the New report back to the original values.

Save New Spec Limits as a Column Property Saves the new specification limits as a Spec Limits column property to the column in the original data table.

Note: The analysis is not rerun with the new specification limits unless the Auto Recalc option is turned on.

Parameter Estimates (Available when a distribution other than Normal or Nonparametric is selected.) Shows or hides the Parameter Estimates report, which gives estimates for the parameters of the selected distribution.

The estimates for all except the Johnson family distributions are obtained using maximum likelihood. For more information about Johnson family fits, see "Johnson Distribution Fit Method".

The parameters and probability density functions for the normal, beta, exponential, gamma, Johnson, lognormal, and Weibull distributions are described in "Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods". These are the same parameterizations used in the Distribution platform, with the exception that Process Capability does not support threshold parameters. See *Basic Analysis*.

Fix Parameters (Available when a distribution other than Normal or Nonparametric is selected.) Displays a window that enables you to fix one or more parameter values in a nonnormal distribution. Enter a value in the User-defined Value column for the parameters that you would like to fix. Once you click OK, the omitted parameter values

are re-estimated given the fixed parameter values. The re-estimated parameter values appear in the Parameter Estimates report, along with a column indicating which parameters are fixed.

Nonparametric Density (Available when Nonparametric is selected as the distribution.) Shows or hides the Nonparametric Density report, which gives the *kernel bandwidth* used in fitting the nonparametric distribution. The kernel bandwidth is given by the following, where n is the number of observations and S is the uncorrected sample standard deviation:

$$\text{bandwidth} = \frac{0.9S}{n^{1/5}}$$

Compare Distributions

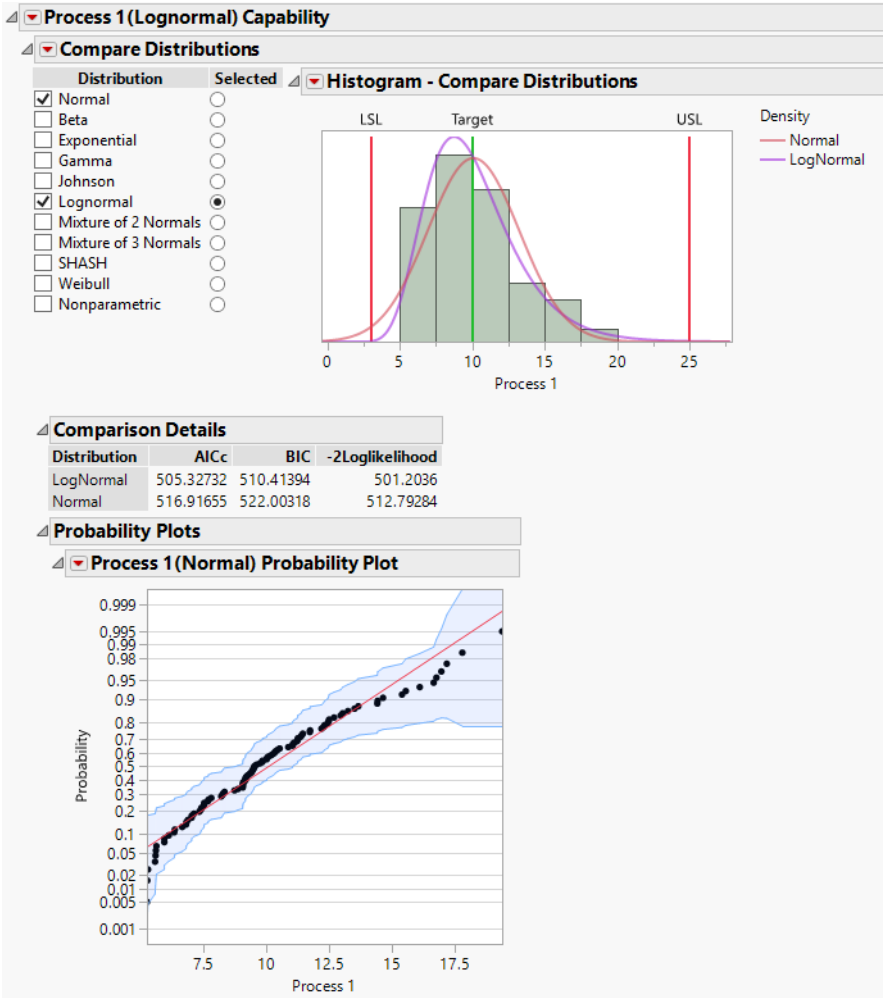
The Compare Distributions report enables you to compare and apply various distributional fits. Note the following:

- Your selected distribution is indicated in the Selected column.
- The report initially shows fit statistics for your Selected distribution and other fitted distributions in the Comparison Details report. If you selected Best Fit, the Comparison Details report initially shows statistics for all parametric fits.
- Check the distributions in the Distribution list that you want to compare.
  - The probability density function for the best fitting distribution in each family that you select is superimposed on the histogram in the Histogram - Compare Distributions report.
  - If the distribution is parametric, a row for that family containing fit results is added to the Comparison Details report.
  - If Nonparametric is checked in the Distribution list, the Nonparametric Density report, showing the automatically selected kernel bandwidth, is added to the Compare Distributions report. See "Nonparametric Density".
  - You can change your selected distribution by selecting its radio button under Selected. The capability report is updated to show results for the selected distribution.

Figure 7.16 shows the Compare Distributions report for Process 1 in the Process Measurements.jmp sample data table. The Selected distribution, which is Lognormal, is being compared to a Normal distribution. The Comparison Details report shows fit statistics for both distributions.

To obtain probability plots, click the Compare Distributions red triangle and select Probability Plots. The points in the probability plot for the normal distribution in Figure 7.16 do not follow the line closely. This indicates a poor fit.

Figure 7.16 Compare Distributions with Probability Plot for Normal



Compare Distributions Options

The Compare Distributions red triangle menu contains the following options:

Comparison Details For each distribution, gives AICc, BIC, and -2Loglikelihood values. See *Fitting Linear Models*. (Not available for a Nonparametric fit.)

Comparison Histogram Shows or hides the Histogram report.

Probability Plots Shows or hides a report that displays probability plots for each parametric distribution that you fit (Figure 7.16). An observation's horizontal coordinate is its observed data value. An observation's vertical coordinate is the value of the quantile of the

fitted distribution for the observation's rank. For the normal distribution, the overall estimate of sigma is used in determining the fitted distribution.

The red triangle menus associated with each Probability Plot contain the following options.

Simultaneous Empirical Confidence Limits Shows or hides confidence limits that have a simultaneous 95% confidence level of containing the true probability function, given that the data come from the selected parametric family. These limits have the same estimated precision at all points. Use them to determine whether the selected parametric distribution fits the data well. See Nair (1984) and Meeker and Escobar (1998).

Simultaneous Empirical Confidence Limits Shading Shows or hides shading of the region between the Simultaneous Empirical Confidence Limits.

Parametric Fit Line Shows or hides the line that shows the predicted probabilities for the observations based on the fitted distribution.

Parametric Fit Confidence Limits Shading Shows or hides shading of the region between parametric fit confidence intervals. The parametric fit confidence limits have confidence level (1 - Alpha), where Alpha is the value that you specify in the launch window. (Available only when the parametric fit confidence limits are meaningful and when it is possible to calculate them.)

When possible, the intervals are computed by expressing the parametric distribution F as a location-scale family, so that $F(y) = G(z)$, where $z = (y - \mu)/\sigma$. The approximate standard error of the fitted location-scale component at a point is computed using the delta method. Using the standard error estimate, a Wald confidence interval for z is computed for each point. The confidence interval for the cumulative distribution function F is obtained by transforming the Wald interval using G. Note that, in some cases, special accommodations are required to provide appropriate intervals near the endpoints of the interval of process measurements.

Order by Comparison Criterion Orders the distributions in the Comparison Details report according to the criterion that you select. The default ordering is by AICc, unless you selected another criterion in the Distribution Comparison Criterion panel in the launch window.

Normalized Box Plots

The Normalized Box Plots options show or hide box plots that have been normalized using the specified sigma in the title. When drawing normalized box plots, JMP standardizes each column by subtracting the mean and dividing by the standard deviation. The box plots are formed for each column using these standardized values.

Figure 7.17 Within Sigma Normalized Box Plot

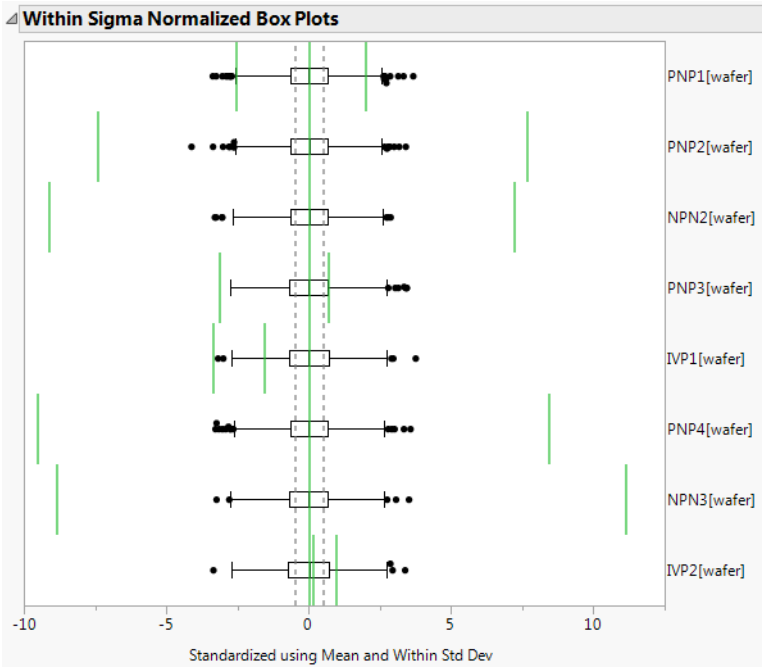


Figure 7.17 shows the Within Sigma Normalized Box Plot for a selection of the process variables in the Semiconductor Capability.jmp sample data table using wafer as a subgroup variable.

The green vertical lines represent the specification limits for each variable normalized by the mean and standard deviation of each variable. The gray dotted vertical lines are drawn at ±0.5, since the data is standardized to a standard deviation of 1.

Process Performance Plot

The Process Performance Plot option shows or hides a four-quadrant plot of capability versus stability. Each process that has specification limits is represented by a point. If process importance values are specified, the points are sized by importance. The horizontal coordinate of each point equals the stability index of the process and the vertical coordinate of each point equals the overall Ppk capability of the process. The plot is divided into four shaded quadrants based on the following default boundaries:

- A stability index that exceeds 1.25 indicates that the process is unstable.
- A Ppk that is smaller than 1.0 indicates that the process is not capable.

Additionally, there is a red line on the graph that indicates where the Cpk value is 1. The boundaries that define the four quadrants can be adjusted using the Ppk and Stability Index slider controls to the right of the plot. You can also set preferences for your desired Capability and Stability boundaries, as well as stability assessment type in
File > Preferences > Platforms > Process Performance Plot and
File > Preferences > Platforms > Process Capability.

The legend contains descriptions of the shaded regions. If any of the processes are missing a lower or upper specification limit, the legend also shows the markers used for those processes. If the markers do not appear in the legend, then all of the processes in the plot contain both lower and upper specification limits. See "One-Sided or Missing Specification Limits".

Hover over a point in the Process Performance Plot to view a control chart for that process. Click the control chart to launch Control Chart Builder with the corresponding control chart and capability report.

Note: A control chart is not available for a process if the unbiased pooled standard deviation is selected as the within-group variation statistic for that process.

The Process Performance Plot red triangle menu contains the following option:

Label Points Shows or hides labels for each point in the Process Performance Plot.

Show Within Cpk Curve Shows or hides the within Cpk curve in the Process Performance Plot.

Figure 7.18 Process Performance Plot

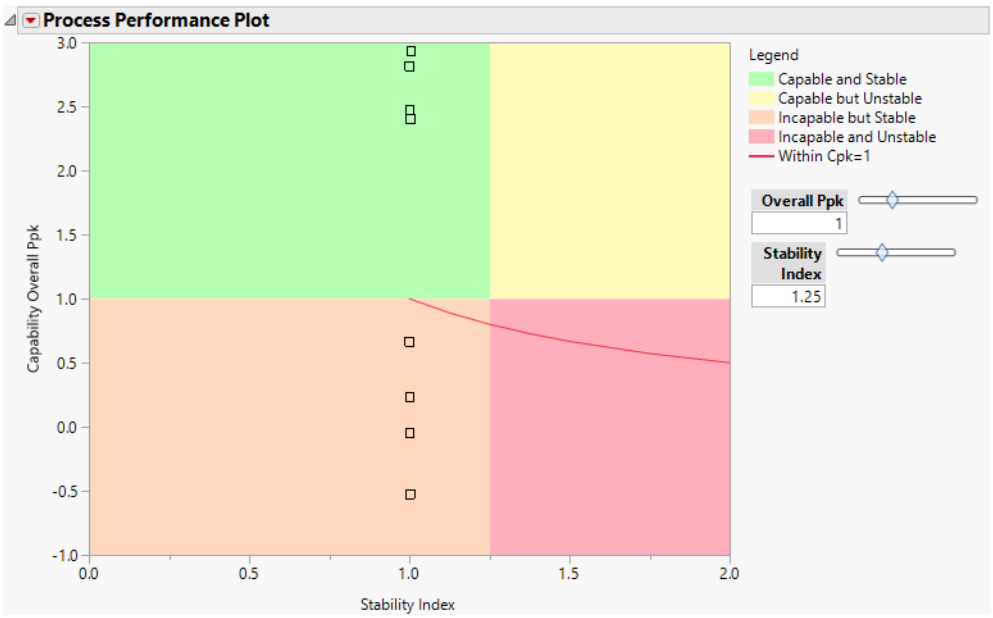


Figure 7.18 shows the Process Performance Plot for a selection of the process variables in the Semiconductor Capability.jmp sample data table using wafer as a subgroup variable.

Summary Reports

The Summary Report options show or hide a table that contains the following statistics for each variable: LSL, Target, USL, Sample Mean, various Sigma estimates, Stability Index, Cpk, Cpl, Cpu, Cp, Cpm, and Nonconformance statistics. If there is at least one nonmissing process importance value, an Importance column is also included in the Summary Report. These statistics are calculated using the sigma estimate specified in the report title. The columns for Stability Index, Cpk, Cpl, Cpu, and Cp are colored as green, yellow, and red to indicate adequate, marginal, and poor stability or capability. This color coding scheme matches what you would see in the Process Performance graph.

Note: You can change the preferences for stability assessment type in File > Preferences > Platforms > Process Capability. This changes the stability assessment type used through the Process Capability platform.

Figure 7.19 shows a subset of columns for both summary reports as described in "Example of the Process Capability Platform with Normal Variables". The following optional columns are available for this report:

- Confidence intervals for Cpk, Cpl, Cpu, CP, and Cpm
- Expected and observed PPM statistics (outside, below LSL, above USL)

  Note: The expected PPM statistics are the percentages you would expect to see based on the distribution chosen. By default, the distribution is normal. The observed PPM statistics are the percentages based on the actual data.

- Sample standard deviation
- The sample size (N), the minimum, and the maximum.
- Target Index

  Note: Target Index is only available in the Within Sigma Capability Summary report.

To reveal these optional columns, right-click the report and select the column names from the Columns submenu.

Note that the report (based on overall sigma) shows the overall capability indices Ppk, Ppl, Ppu, and Pp instead of the within capability indices Cpk, Cpl, Cpu, and Cp. The labeling of the overall capability indices depends on the setting of the AIAG (Ppk) Labeling preference.

Figure 7.19 Within Sigma and Overall Sigma Capability Summary Reports

Process Capability

Within Sigma Capability Summary Report

Process	LSL	Target	USL	Sample Mean	Within Sigma	Stability Index	Cpk	Cpl	Cpu	Cp	Cpm
PNP1	164.39	297.02	429.65	313.0697	59.43013	0.981885	0.654	0.834	0.654	0.744	0.718
PNP2	-136.12	465.44	1067.01	456.6157	79.27036	1.007008	2.492	2.492	2.567	2.530	2.514
NPN2	96.59	113.75	130.9	115.7421	2.131652	0.98552	2.370	2.995	2.370	2.683	1.959
PNP3	118.68	130.29	141.9	137.6146	6.160912	0.983744	0.232	1.024	0.232	0.628	0.404
IVP1	59.62	63.41	67.2	73.78072	4.238298	0.990097	-0.518	1.114	-0.518	0.298	0.113
PNP4	-54.43	238.74	531.91	256.3756	33.22573	0.981389	2.764	3.118	2.764	2.941	2.598
NPN3	97.32	120.8	144.29	118.1352	2.362847	1.000809	2.936	2.936	3.690	3.313	2.198
IVP2	139.2	142.31	145.41	138.2432	7.406516	0.989286	-0.043	-0.043	0.323	0.140	0.122

Overall Sigma Capability Summary Report

Process	LSL	Target	USL	Sample Mean	Overall Sigma	Stability Index	Ppk	Ppl	Ppu	Pp	Cpm
PNP1	164.39	297.02	429.65	313.0697	58.35353	0.981885	0.666	0.849	0.666	0.758	0.730
PNP2	-136.12	465.44	1067.01	456.6157	79.82589	1.007008	2.475	2.475	2.549	2.512	2.497
NPN2	96.59	113.75	130.9	115.7421	2.100786	0.98552	2.405	3.039	2.405	2.722	1.975
PNP3	118.68	130.29	141.9	137.6146	6.060762	0.983744	0.236	1.041	0.236	0.639	0.407
IVP1	59.62	63.41	67.2	73.78072	4.196326	0.990097	-0.523	1.125	-0.523	0.301	0.113
PNP4	-54.43	238.74	531.91	256.3756	32.60738	0.981389	2.817	3.177	2.817	2.997	2.636
NPN3	97.32	120.8	144.29	118.1352	2.364757	1.000809	2.934	2.934	3.687	3.310	2.197
IVP2	139.2	142.31	145.41	138.2432	7.327164	0.989286	-0.044	-0.044	0.326	0.141	0.123

Make Goal Plot Summary Table

The Make Goal Plot Summary Table option produces a summary data table that includes each variable's name, its spec-normalized mean shift (Spec-Standardized Mean), and its spec-normalized standard deviation (Spec-Standardized Std Dev). For each variable, there is a row for each of the sigma types.

Note: If a variable is fit with a distribution other than normal, the name of the fitted distribution is appended parenthetically to the variable name. The Spec-Standardized Mean and Spec-Standardized Std Dev values are not provided for nonnormal variables.

The points in the Goal Plot are linked to the rows in the Goal Plot Summary Table. If you apply row states to a point in the Goal Plot, you can change the corresponding row states in the Goal Plot Summary Table. Conversely, if you apply row states in the Goal Plot Summary Table, they are reflected on the Goal Plot.

Figure 7.20 shows the Goal Plot Summary Table for the Semiconductor Capability.jmp sample data table as described in "Example of the Process Capability Platform with Normal Variables".

Figure 7.20 Summary Table

	Process	Sigma Type	Spec-Standardized Mean	Spec-Standardized Std Dev
1	PNP1	Within	0.0605056038	0.2240448185
2	PNP2	Within	-0.00733452	0.0658873257
3	NPN2	Within	0.0580798256	0.0621472935
4	PNP3	Within	0.3154424678	0.2653278331
5	IVP1	Within	1.3681693325	0.5591421926
6	PNP4	Within	0.0300774244	0.0566663241
7	NPN3	Within	-0.05674621	0.0503161575
8	IVP2	Within	-0.65593547	1.1945993146
9	PNP1	Overall	0.0605056038	0.2199861461
10	PNP2	Overall	-0.00733452	0.0663490674
11	NPN2	Overall	0.0580798256	0.0612473972
12	PNP3	Overall	0.3154424678	0.2610147096
13	IVP1	Overall	1.3681693325	0.5536050447
14	PNP4	Overall	0.0300774244	0.0556117297

Additional Examples of the Process Capability Platform

- "Process Capability for a Stable Process"
- "Process Capability for an Unstable Process"
- "Simulation of Confidence Limits for a Nonnormal Process Ppk"

Process Capability for a Stable Process

In this example, you verify the assumptions that enable you to estimate PPM defective rates based on a capability analysis. You access Process Capability through Control Chart Builder and then directly. The data consist of 22 subgroups of size five. There are six missing readings, with three in each of two consecutive subgroups.

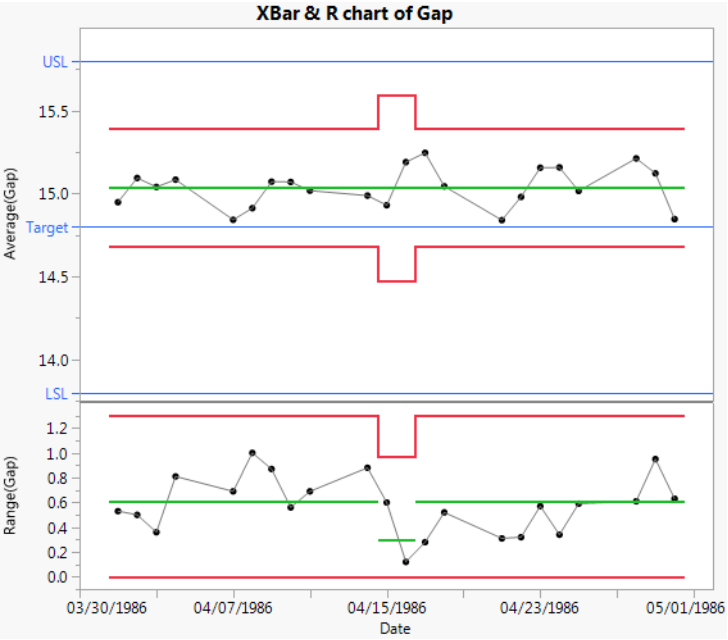
Process Capability through Control Chart Builder

You can use Control Chart Builder to check process stability and the normality assumption for your process characteristic. You can also obtain Process Capability information directly within Control Chart Builder.

1. Select **Help > Sample Data Library** and open Quality Control/Clips2.jmp.
2. Select **Analyze > Quality and Process > Control Chart Builder**.
3. Drag Date to the **Subgroup** zone.

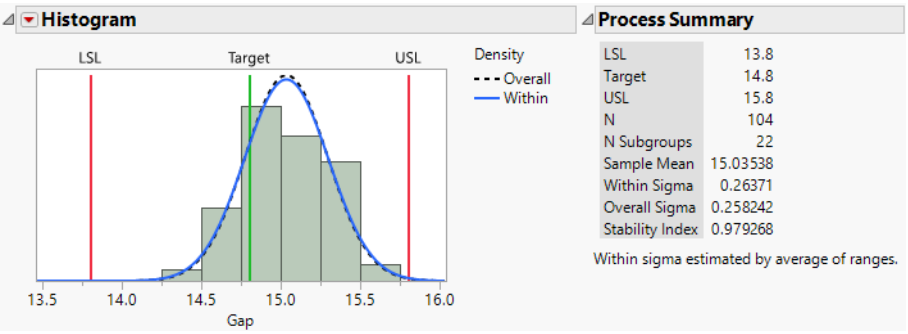
4. Drag Gap to the **Y** zone.

Figure 7.21 XBar and R Control Chart for Gap



The control chart indicates that Gap is stable over time. Because Gap has the Spec Limits column property, a Process Capability Analysis report appears to the right of the control chart.

Figure 7.22 Histogram in Process Capability Analysis for Gap

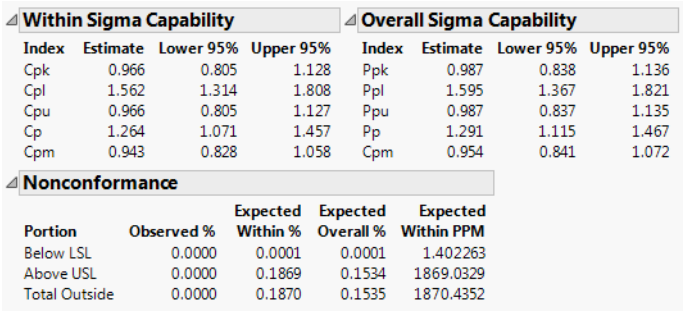


The histogram and fitted normal blue curve suggest that the distribution of Gap is approximately normal. Although the process is stable, the distribution of Gap is shifted to the right of the specification range.

The Process Summary report shows the specification limits that are saved to the Spec Limits column property. It also shows that the estimate of sigma calculated from within-subgroup variation (Within Sigma) does not differ greatly from the overall estimate given by the sample standard deviation (Overall Sigma). Consequently, the Stability Index is near one (0.979268). This is expected because the process is stable.

5. Right-click in the body of the Nonconformance report and select **Expected Within PPM** from the Columns submenu.

Figure 7.23 Capability Indices and Nonconformance Report



Within Sigma Capability

Index	Estimate	Lower 95%	Upper 95%
Cpk	0.966	0.805	1.128
Cpl	1.562	1.314	1.808
Cpu	0.966	0.805	1.127
Cp	1.264	1.071	1.457
Cpm	0.943	0.828	1.058

Overall Sigma Capability

Index	Estimate	Lower 95%	Upper 95%
Ppk	0.987	0.838	1.136
Ppl	1.595	1.367	1.821
Ppu	0.987	0.837	1.135
Pp	1.291	1.115	1.467
Cpm	0.954	0.841	1.072

Nonconformance

Portion	Observed %	Expected Within %	Expected Overall %	Expected Within PPM
Below LSL	0.0000	0.0001	0.0001	1.402263
Above USL	0.0000	0.1869	0.1534	1869.0329
Total Outside	0.0000	0.1870	0.1535	1870.4352

The Cpk value calculated using subgroup variation is 0.966, indicating that the process is not very capable. The Cpl value suggests good performance, but this is because the process is shifted away from the lower specification limit. Defective parts generally result from large values of Gap.

Note that the confidence interval for Cpk is wide; it ranges from 0.805 to 1.128. This occurs even though there are 104 observations. Capability indices are surprisingly variable, due to the fact that they are ratios. It is easy to reach incorrect conclusions based on the point estimate of a capability index.

The estimates of out-of-specification product given in the Nonconformance report provide a direct measure of process performance. The PPM values in the Nonconformance report indicate that Gap hardly ever falls below the lower specification limit (1.4 parts per million). However, the number of parts for which Gap falls above the upper specification limit is 1869.0 parts per million.

For an uncentered process, the Cp value indicates potential capability if the process were adjusted to be centered. If this process were adjusted to be centered at the target value of 14.8, then its capability would be 1.264, with a confidence interval from 1.071 to 1.457.

Process Capability Platform

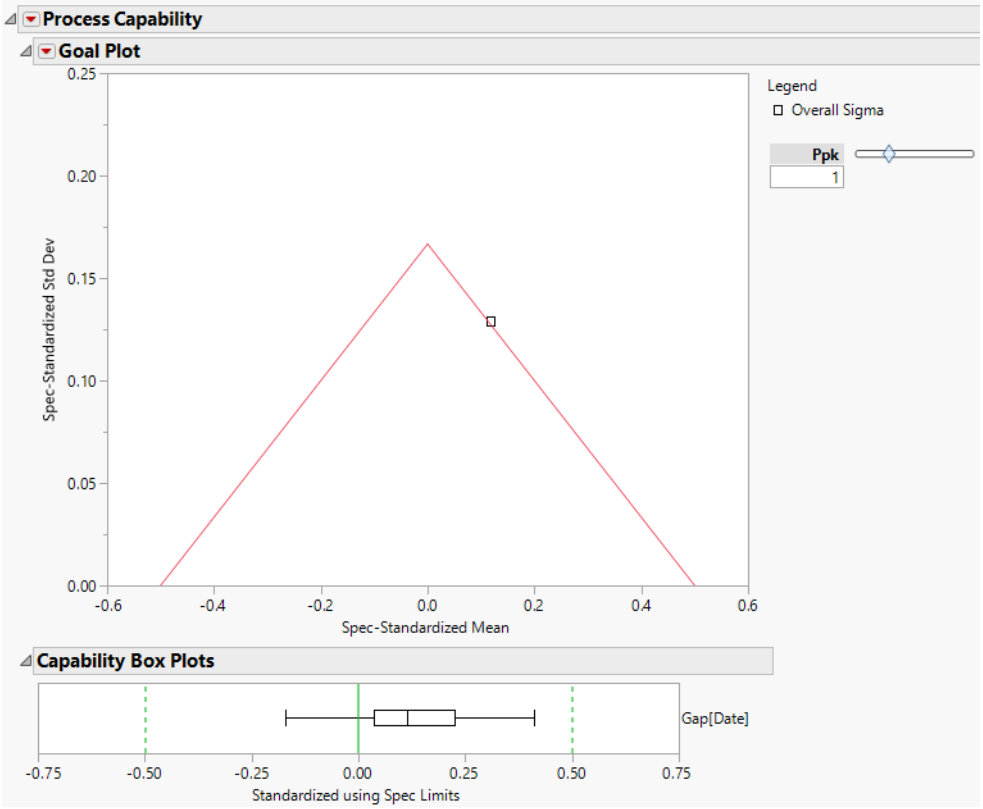
Now that you have verified stability and normality for Gap, you can obtain additional information in the Process Capability platform.

1. Select **Analyze > Quality and Process > Process Capability**.

2. Select Gap and click **Y, Process**.
3. Open the **Process Subgrouping** outline.
4. Select Date in the Select Columns list and Gap in the Roles list.
5. Click **Nest Subgroup ID Column**.

   By default, the Within-Subgroup Variation Statistic selection is set to Average of Unbiased Standard Deviations. In the Control Chart Builder example ("Process Capability through Control Chart Builder"), subgroup ranges were used.
6. Click **OK**.

Figure 7.24 Goal Plot and Box Plot for Gap



The Goal Plot shows the Ppk index for Gap as being essentially equal to 1. The box plot shows that most values fall within specifications, but the preponderance of data values are shifted to the right within the specification range.

7. Click the Process Capability red triangle and select Individual Detail Reports.

   The report is the one obtained using Control Chart Builder, except that the Within Sigma is based on average standard deviations rather than average ranges. See "Histogram in

Process Capability Analysis for Gap" and "Capability Indices and Nonconformance Report".

Process Capability for an Unstable Process

This example shows a case where the overall variation differs from the within variation because the process is not stable. It uses the Coating.jmp data table from the Quality Control folder of Sample Data (taken from the *ASTM Manual on Presentation of Data and Control Chart Analysis)*. The process variable of interest is the Weight column, grouped into subgroups by the Sample column.

1. Select **Help > Sample Data Library** and open Quality Control/Coating.jmp.
2. Select **Analyze > Quality and Process > Process Capability**.
3. Select Weight and click **Y, Process**.
4. Open the **Process Subgrouping** outline.
5. Select Sample in the **Select Columns** list on the left.
6. Select Weight in the **Cast Selected Columns into Roles** list on the right.
7. Click **Nest Subgroup ID Column**.
8. Click **OK**.
9. Enter 16 for **LSL**, 20 for **Target**, and 24 for **USL** in the **Spec Limits** window.
10. Click **OK**.
11. Click the Goal Plot red triangle and select **Show Within Sigma Points**.
12. Click the Process Capability red triangle and select **Individual Detail Reports**.

Figure 7.25 Process Capability Report for Coating.jmp Data

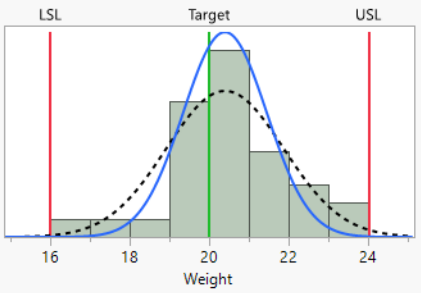
Process Capability

Individual Detail Reports

Weight[Sample] Capability

Grouped by subgroup ID: Sample

Histogram



Process Summary

LSL	16
Target	20
USL	24
N	32
N Subgroups	8
Sample Mean	20.39688
Within Sigma	1.051753
Overall Sigma	1.475058
Stability Index	1.402476

Within sigma estimated by average of unbiased standard deviations.

Within Sigma Capability

Index	Estimate	Lower 95%	Upper 95%
Cpk	1.142	0.789	1.495
Cpl	1.394	0.971	1.813
Cpu	1.142	0.789	1.491
Cp	1.268	0.899	1.635
Cpm	1.186	0.899	1.473

Overall Sigma Capability

Index	Estimate	Lower 95%	Upper 95%
Ppk	0.814	0.581	1.048
Ppl	0.994	0.720	1.264
Ppu	0.814	0.580	1.045
Pp	0.904	0.680	1.127
Cpm	0.873	0.670	1.101

Nonconformance

Portion	Observed %	Expected Within %	Expected Overall %
Below LSL	0.0000	0.0015	0.1437
Above USL	0.0000	0.0306	0.7289
Total Outside	0.0000	0.0321	0.8726

Goal Plot

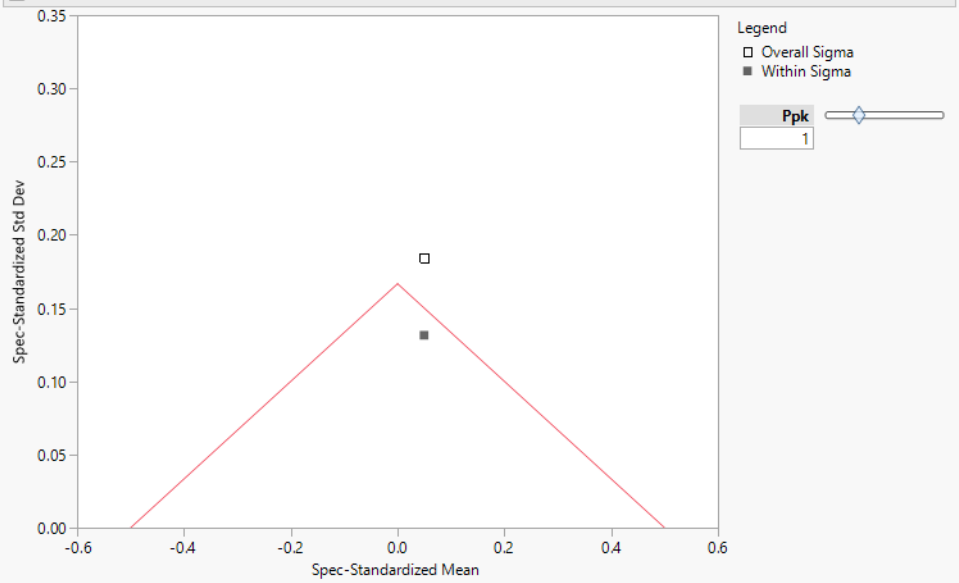


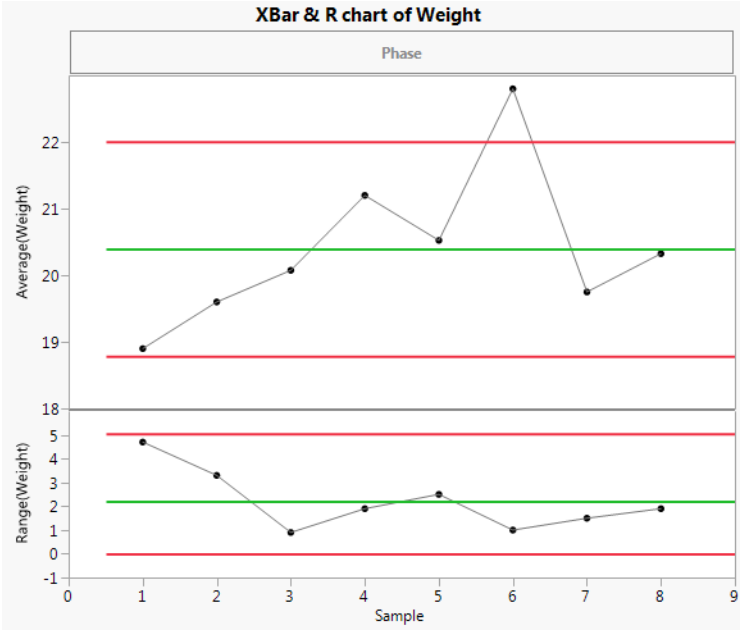
Figure 7.25 shows the resulting Process Capability report. The Goal Plot shows two points that represent the mean shift and standard deviation standardized to the specification limits. There is a legend next to the Goal Plot that identifies the two points. The Overall Sigma point is calculated using the overall sample standard deviation. The Within Sigma point is calculated using a within-subgroup estimate of the standard deviation.

The point calculated using Overall Sigma is outside the goal triangle corresponding to a Ppk of 1. This indicates that the variable Weight results in non-conforming product.

However, the point calculated using Within Sigma is inside the goal triangle. This indicates that, if the process were stable, Weight values would have a high probability of falling within the specification limits.

13. Hover cursor over one of the points in the Goal Plot.
14. Click the control chart.

Figure 7.26 XBar and R Chart for Weight



The control chart indicates that the Weight measurements are unstable. The process is affected by special causes and is unpredictable. This makes the interpretation of capability indices and nonconformance estimates highly questionable. Even estimates based on Overall Sigma are questionable, because the process is not predictable.

The histogram in Figure 7.25 shows the distribution of the Weight values with normal density curves using both sigma estimates superimposed over the histogram. The normal curve that uses the Overall Sigma estimate is flatter and wider than the normal curve that uses the Within Sigma estimate. This normal curve is more dispersed because the estimate of Overall Sigma is inflated by the special causes that make the process unstable. If the process were stable, the narrower normal curve would reflect process behavior.

You can also compare the Cpk estimate (1.142) to the Ppk estimate (0.814). The fact that Ppk is much smaller than Cpk is additional evidence that this is an unpredictable process. The Cpk estimate is a forecast of the capability that you would achieve by bringing the process to a stable state.

Note: The Individual Detail Reports Cutoff preference determines whether the Individual Reports appear by default. If the preference is enabled, the Individual Reports appear by default if the number of process variables is less than or equal to the number specified in the preference. You can change this preference in Preferences > Platforms > Process Capability.

Simulation of Confidence Limits for a Nonnormal Process Ppk

In this example, you first perform a capability analysis for the three nonnormal variables in Tablet Measurements.jmp. You then use Simulate to find confidence limits for the nonconformance percentage for the variable Purity.

Nonnormal Capability Analysis

If you prefer not to follow the steps below, you can obtain the results in this section by running the **Process Capability** table script in Tablet Measurements.jmp.

1. Select **Help > Sample Data Library** and open Tablet Measurements.jmp.
2. Select **Analyze > Quality and Process > Process Capability**.
3. Select Weight, Thickness, and Purity and click **Y, Process**.
4. Select Weight, Thickness, and Purity in the **Cast Selected Columns into Roles** list on the right.
5. Open the **Distribution Options** outline.
6. From the Distribution list, select **Best Fit**.
7. Click **Set Process Distribution**.

   The **&Dist(Best Fit)** suffix is added to each column name in the list on the right.
8. Click **OK**.

   A Capability Index Plot appears, showing the Ppk values. Note that only the Thickness variable appears above the line that denotes Ppk = 1. Purity is nearly on the line. Although

the number of measurements, 250, seems large, the estimated Ppk value is still quite variable. For this reason, you construct a confidence interval for the true Purity Ppk value.

Note: Because a Goal Plot is not shown, you can conclude that a normal distribution fit was not the best fit for any of the three variables.

9. Click the Process Capability red triangle and select **Individual Detail Reports**.

   The best fits are different for each process.

   – Weight: Lognormal
   – Thickness: SHASH
   – Purity: Weibull

Construct the Simulation Column

To use the Simulate utility to estimate Ppk confidence limits, you need to construct a simulation formula that reflects the fitted Weibull distribution. If you prefer not to follow the steps below, you can obtain the results in this section by running the **Add Simulation Column** table script.

1. Scroll to the Purity (Weibull) Capability report and find the Parameter Estimates report.

Figure 7.27 Weibull Parameter Estimates for Purity

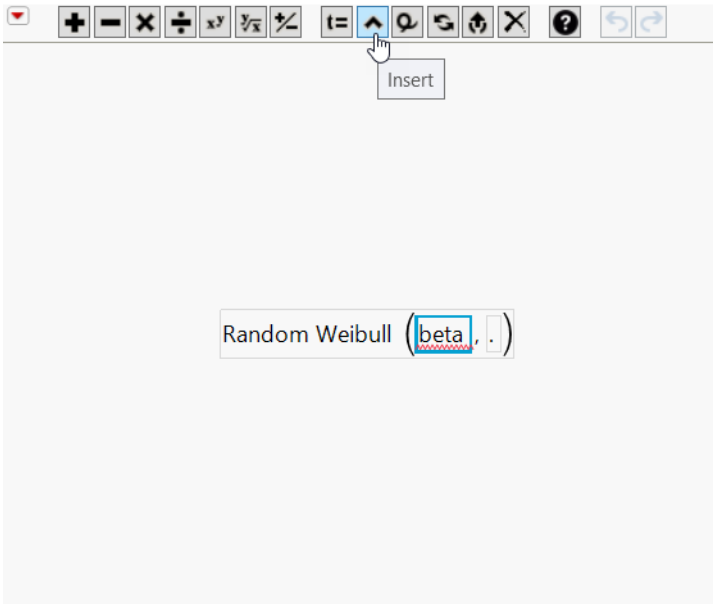
Parameter Estimates

Parameter		Estimate
Scale	α	99.918709
Shape	β	1589.7168

   These are the parameter estimates for the best fitting distribution, which is Weibull.

2. In the Tablet Measurements.jmp sample data table, select **Cols > New Columns**.
3. Next to **Column Name**, enter Simulated Purity.
4. From the **Column Properties** list, select **Formula**.
5. In the formula editor, select **Random > Random Weibull**.
6. The placeholder for **beta** is selected. Click the insertion element (**^**).

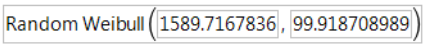
Figure 7.28 Formula Editor for Simulated Purity Column



This adds a placeholder for the parameter **alpha**.

7. In the Process Capability report, under Purity (Weibull) Capability, right-click in the Parameter Estimates report table and select **Make into Data Table**.
8. Copy the entry in Row 2 in the Estimate column (1589.7167836).
9. In the formula editor window, select the placeholder for **beta** in the **Random Weibull** formula and paste 1589.7167836 into the placeholder for **beta**.
10. In the data table that you created from the Parameter Estimates report, copy the entry in Row 1 in the Estimate column (99.918708989).
11. In the formula editor window, select the placeholder for **alpha** in the **Random Weibull** formula and paste 99.918708989 into the placeholder for **alpha**.

Figure 7.29 Completed Formula Window



12. Click **OK** in the formula editor window.
13. In the Tablet Measurements.jmp data table, right-click the Simulated Purity column and select **Column Properties > Spec Limits**.
14. Next to **Lower Spec Limit**, type 99.5.
15. Click **OK** in the New Column window.

The Simulated Purity column contains a formula that simulates values from the best-fitting distribution.

Simulate Confidence Intervals for Purity Ppk and Expected Percent Nonconforming

When you use Simulate, the entire analysis is run the number of times that you specify. To shorten the computing time, you can minimize the computational burden by running only the required analysis. In this example, because you are interested only in Purity with a fitted Weibull distribution, you perform only this analysis before running Simulate.

Note: If you do not care about computing time, you can use the same report you created in the previous section and start with step 7.

1. In the Process Capability report, click the Process Capability red triangle and select **Relaunch Dialog**.
2. (Optional) Close the Process Capability report.
3. In the launch window, from the **Cast Selected Columns into Roles** list, select Weight&Dist(Lognormal) and Thickness&Dist(SHASH).
4. Click **Remove**.
5. Click **OK**.
6. Click the Process Capability red triangle and select **Individual Detail Reports**.

   Both Ppk and Ppl values are provided, but they are identical because Purity has only a lower specification limit.
7. In the Overall Sigma Capability report, right-click the **Estimate** column and select **Simulate**.

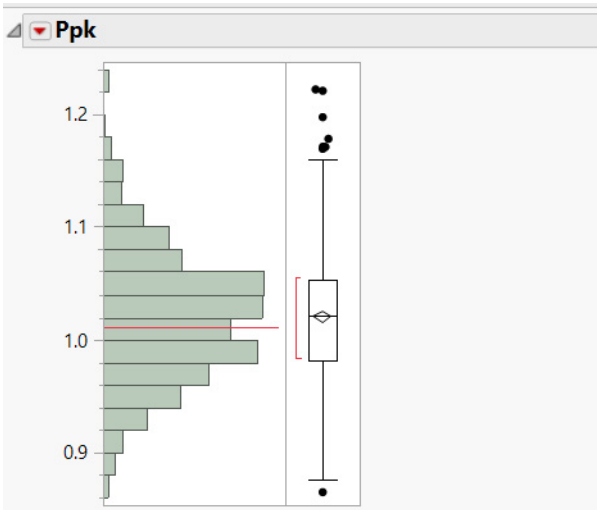
   In the **Column to Switch Out** list, make sure Purity is selected. In the **Column to Switch In** list, make sure Simulated Purity is selected.
8. Next to **Number of Samples**, type **500**.

   Note: The next step is not required. However, it ensures that you obtain exactly the simulated values shown in this example.
9. (Optional) Next to **Random Seed**, type **12345**.
10. Click **OK**.

    The calculation might take several seconds. A data table entitled Process Capability Simulate Results (Estimate) appears. The Ppk and Ppl columns in this table each contain 500 values calculated based on the Simulated Purity formula. The first row, which is excluded, contains the values for Purity obtained in your original analysis. Because Purity has only a lower specification limit, the Ppk values are identical to the Ppl values.

11. In the Process Capability Simulate Results (Estimate) data table, click the green triangle next to the **Distribution** script.

Figure 7.30 Distribution of Simulated Ppk Values for Purity



Quantiles

100.0%	maximum	1.2221546009
99.5%		1.2091396868
97.5%		1.1446471769
90.0%		1.094807522
75.0%	quartile	1.0536248674
50.0%	median	1.0205019662
25.0%	quartile	0.9810111196
10.0%		0.9500946687
2.5%		0.9091919384
0.5%		0.8809542448
0.0%	minimum	0.8646511006

Summary Statistics

Mean	1.0205936
Std Dev	0.0573743
Std Err Mean	0.0025659
Upper 95% Mean	1.0256348
Lower 95% Mean	1.0155524
N	500

Simulation Results

Y_0 = 1.01099 (Original Estimate)

Confidence Intervals

Alpha	Percentile Lower CI	Percentile Upper CI
0.05	0.90919	1.14465
0.10	0.93059	1.11828
0.20	0.95009	1.09481
0.50	0.98101	1.05362

Empirical p-Values

Test	p-Value
$Y \geq \lvert Y_0 \rvert$	0.5600
$Y \leq Y_0$	0.4400
$Y \geq Y_0$	0.5600

Note: Your values may vary slightly from what is shown, depending on the decimal precision of the parameters in the Simulated Purity column formula.

Two Distribution reports are shown, one for Ppk and one for Ppl. But Purity has only a lower specification limit, so that the Ppk and Ppl values are identical. For this reason, the Distribution reports are identical.

The Simulation Results report shows that a 95% confidence interval for Ppk is 0.909 to 1.145. This indicate that the true Ppk value could be above 1.0, which would place Purity above the Ppk = 1 line in the Capability Index Plot you constructed in "Nonnormal Capability Analysis".

12. In the Process Capability report, right-click the **Expected Overall %** column in the Nonconformance report and select **Simulate**.

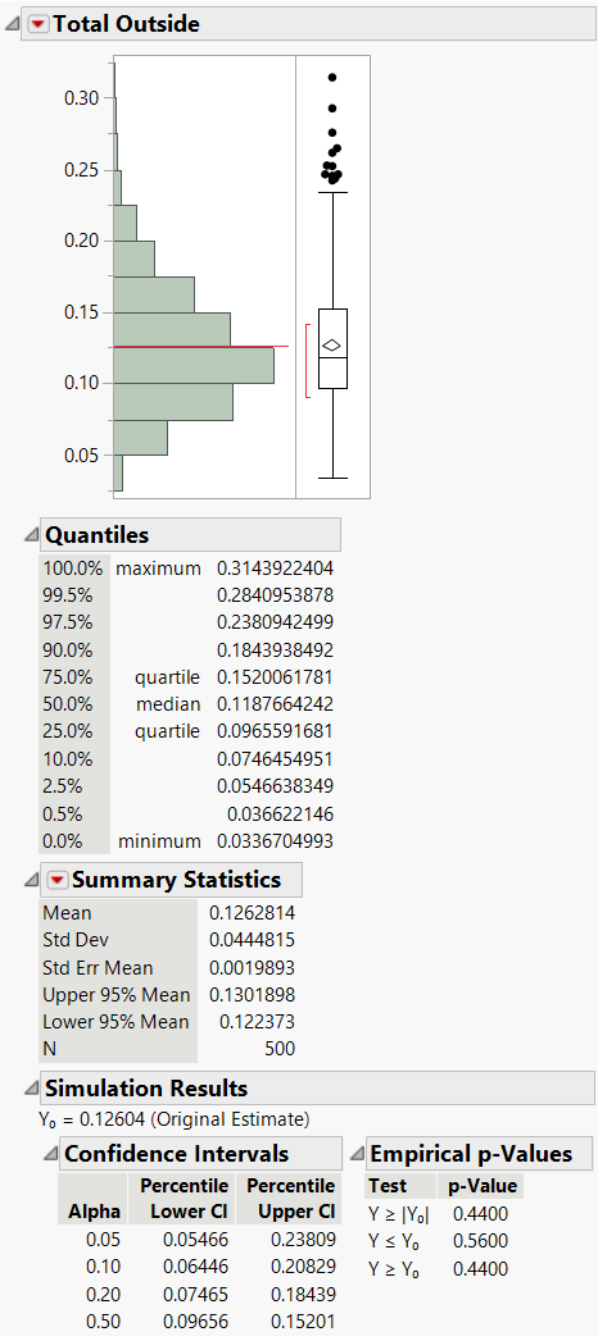
    In the **Column to Switch Out** list, make sure Purity is selected. In the **Column to Switch In** list, make sure Simulated Purity is selected.

13. Next to **Number of Samples**, enter **500**.
14. (Optional) Next to **Random Seed**, enter **12345**.
15. Click **OK**.

    The calculation might take several seconds. A data table entitled Process Capability Simulate Results (Expected Overall %) appears. Because Purity has only a lower specification limit, the Below LSL values are identical to the Total Outside values.

16. In the Process Capability Simulate Results (Expected Overall %) data table, click the green triangle next to the **Distribution** script.

Figure 7.31 Distribution of Simulated Total Outside Values for Purity



Note: Your values may vary slightly from what is shown, depending on the decimal precision of the parameters in the Simulated Purity column formula.

Again, two identical Distribution reports appear. The Simulation Results report shows that a 95% confidence interval for the Expected Overall % nonconforming is 0.055 to 0.238.

Statistical Details for the Process Capability Platform

- "Variation Statistics"
- "Notation for Goal Plots and Capability Box Plots"
- "Goal Plot"
- "Capability Box Plots for Processes with Missing Targets"
- "Capability Indices for Normal Distributions"
- "Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods"
- "Parameterizations for Distributions"

Variation Statistics

Denote the standard deviation of the process by σ. The Process Capability platform provides two types of capability indices. The Ppk indices are based on an estimate of σ that uses all of the data in a way that does not depend on subgroups. This overall estimate can reflect special cause as well as common cause variation. The Cpk indices are based on an estimate that attempts to capture only common cause variation. The Cpk indices are constructed using within-subgroup or between-and-within-subgroup estimates of σ. In this way, they attempt to reflect the true process standard deviation. When a process is not stable, the different estimates of σ can differ markedly.

Overall Sigma

The overall sigma does not depend on subgroups. The overall estimate of σ is calculated as follows:

$$\hat{\sigma} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

The formula uses the following notation:

N = number of nonmissing values in the entire data set

y_i = value of the i^{th} observation

$\bar{y}$ = mean of nonmissing values in the entire data set

Caution: When the process is stable, the Overall Sigma estimates the process standard deviation. If the process is not stable, the overall estimate of σ is of questionable value, since the process standard deviation is unknown.

Estimates of Sigma Based on Within-Subgroup Variation

An estimate of σ that is based on within-subgroup variation can be constructed in one of the following ways:

- Within sigma estimated by average of ranges
- Within sigma estimated by average of unbiased standard deviations
- Within sigma estimated by moving range
- Within sigma estimated by unbiased pooled standard deviation

If you specify a subgroup ID column or a constant subgroup size on the launch window, you can specify your preferred within-subgroup variation statistic. See "Launch the Process Capability Platform". If you do not specify a subgroup ID column, a constant subgroup size, or a historical sigma, JMP estimates the within sigma using the third method (moving range of subgroups of size two).

Within Sigma Based on Average of Ranges

Within sigma estimated by the *average of ranges* is the same as the estimate of the standard deviation of an *X*Bar and *R* chart:

$$\hat{\sigma} = \frac{\frac{R_1}{d_2(n_1)} + \ldots + \frac{R_N}{d_2(n_N)}}{N}$$

The formula uses the following notation:

R_i = range of i^{th} subgroup

n_i = sample size of i^{th} subgroup

$d_2(n_i)$ = expected value of the range of n_i independent normally distributed variables with unit standard deviation

N = number of subgroups for which $n_i \geq 2$

Within Sigma Based on Average of Unbiased Standard Deviations

Within sigma estimated by the *average of unbiased standard deviations* is the same as the estimate for the standard deviation in an *X*Bar and *S* chart:

$$\hat{\sigma} = \frac{\frac{s_1}{c_4(n_1)} + \ldots + \frac{s_N}{c_4(n_N)}}{N}$$

The formula uses the following notation:

n_i = sample size of i^{th} subgroup

$c_4(n_i)$ = expected value of the standard deviation of n_i independent normally distributed variables with unit standard deviation

N = number of subgroups for which $n_i \geq 2$

s_i = sample standard deviation of the i^{th} subgroup

Within Sigma Based on Average Moving Range

Within sigma estimated by *average moving range* is the same as the estimate for the standard deviation for Individual Measurement and Moving Range charts:

$$\hat{\sigma} = \frac{\overline{MR}}{d_2(2)}$$

The formula uses the following notation:

$\overline{MR}$ = the mean of the nonmissing moving ranges computed as $(MR_2+MR_3+...+MR_N)/(N\text{-}1)$ where $MR_i = |y_i - y_{i\text{-}1}|$.

$d_2(2)$ = expected value of the range of two independent normally distributed variables with unit standard deviation.

Within Sigma Based on Median Moving Range

Within sigma estimated by median moving range:

$$\hat{\sigma} = \frac{MMR}{0.954}$$

The formula uses the following notation:

MMR = the median of the nonmissing moving ranges computed as Median(MR_2, MR_3,..., MR_N) where $MR_i = |y_i - y_{i\text{-}1}|$.

Within Sigma Based on Unbiased Pooled Standard Deviation

Within sigma estimated by the *unbiased pooled standard deviation*:

$$\hat{\sigma} = \frac{\sqrt{(n_1 - 1)s_1^2 + \ldots + (n_N - 1)s_N^2}}{c_4(n)\sqrt{n_1 + \ldots + n_2 - N}}$$

The formula uses the following notation:

n_i = sample size of i^{th} subgroup

$n = n_1 + \ldots + n_N$, the total sample size

$c_4(n)$ = expected value of the standard deviation of n independent normally distributed variables with unit standard deviation

N = number of subgroups for which $n_i \geq 2$

s_i = sample standard deviation of the i^{th} subgroup

Estimate of Sigma Based on Between Group Variation

Between Sigma Based on Moving Range

The estimate of σ that is based on between-subgroup variation is estimated by the moving range of subgroup means:

$$\hat{\sigma} = \sqrt{\left(\frac{\overline{MR}}{d_2(2)}\right)^2 - \frac{\hat{\sigma}^2_{\text{Within}}}{H}}$$

The formula uses the following notation:

$\overline{MR}$ = the mean of the nonmissing moving ranges computed as $(MR_2+MR_3+...+MR_N)/(N-1)$ where $MR_i = |y_i - y_{i-1}|$.

$d_2(2)$ = expected value of the range of two independent normally distributed variables with unit standard deviation.

σ^2_{within} = the specified within sigma estimate.

$H = \dfrac{N}{\frac{1}{n_1} + \frac{1}{n_2} + \ldots + \frac{1}{n_N}}$, the harmonic mean of subgroup sample sizes.

Estimate of Sigma Based on Between and Within Group Variation

Between-and-Within Sigma

The estimate of sigma that is based on the combined between and within group variation is defined as follows:

$$\hat{\sigma} = \sqrt{\hat{\sigma}^2_{\text{within}} + \hat{\sigma}^2_{\text{between}}}$$

Notation for Goal Plots and Capability Box Plots

The formulas for the Goal Plot and Capability Box Plots use the following notation:

Y_{ij} = i^{th} observation for process j

$\bar{Y}_j$ = mean of the observations on process j

SD(Y_j) = standard deviation of the observations on process j

T_j = target value for process j

LSL_j = lower specification limit for process j

USL_j = upper specification limit for process j

Goal Plot

This section provides details about the calculation of the mean shift and standard deviation standardized to specification quantities plotted in the Goal Plot. This section uses the notation defined in "Notation for Goal Plots and Capability Box Plots".

The mean shift and the standard deviation standardized to the specification limits for the j^{th} column are defined as follows:

$$\text{Spec-Standardized Mean} = \frac{\bar{Y}_j - T_j}{2 \times \min(T_j - \text{LSL}_j, \text{USL}_j - T_j)}$$

$$\text{Spec-Standardized Std Dev} = \frac{SD(Y_j)}{2 \times \min(T_j - \text{LSL}_j, \text{USL}_j - T_j)}$$

Note: If either LSL_j or USL_j is missing, twice the distance from the target to the nonmissing specification limit is used in the denominators of the Goal Plot coordinates.

Goal Plot Points for Processes with Missing Targets

Suppose that the process has both a lower and an upper specification limit but no target. Then the formulas given in "Goal Plot" are used, replacing T_j with the midpoint of the two specification limits.

Suppose that the process has only one specification limit and no target. To obtain (x,y) coordinates for a point on the Goal Plot, the capability indices of the process are used. (See "Capability Indices for Normal Distributions" for definitions in terms of the theoretical mean and standard deviation.) For sample observations, the following relationships hold:

$$C_{pu} = \frac{\mathrm{USL}_j - \overline{Y}_j}{3SD(Y_j)}$$

$$C_{pl} = \frac{\overline{Y}_j - \mathrm{LSL}_j}{3SD(Y_j)}$$

If a process has two specification limits and a target at the midpoint of the limits, then the (x,y) coordinates for the point on the Goal Plot satisfy these relationships:

$$C_{pu} = (0.5 - x)/3y$$

$$C_{pl} = (0.5 + x)/3y$$

To obtain coordinates when there is only one specification limit and no target, these relationships are used. To identify a unique point requires an assumption about the slope of the line from the origin on which the points fall. A slope of 0.5 is assumed for the case of an upper specification limit and of -0.5 for a lower specification limit. When capability values are equal to one and the Ppk slider for the goal plot triangle is set to 1, these slopes place the points on the goal plot triangle lines.

Consider the case of only an upper specification limit and no target. Using the assumption that the (x,y) coordinates fall on a line from the origin with slope 0.5, solving for x and y gives the following coordinates:

$$x = 1/(3C_{pu} + 2)$$
$$y = 1/(6C_{pu} + 4)$$

Consider the case of only a lower specification limit and no target. Using the assumption that the (x,y) coordinates fall on a line from the origin with slope -0.5, solving for x and y gives the following coordinates:

$$x = -1/(3C_{pl} + 2)$$
$$y = 1/(6C_{pl} + 4)$$

Note: If either C_{pu} or C_{pl} is less than -0.6, then it is set to -0.6 in the formulas above. At the value -2/3, the denominator for x assumes the value 0. Bounding the capability values at -0.6 prevents the denominator from assuming the value 0 or switching signs.

Capability Box Plots for Processes with Missing Targets

A column with no target can have both upper and lower specification limits, or only a single specification limit. This section uses the notation defined in "Notation for Goal Plots and Capability Box Plots".

Two Specification Limits and No Target

When no target is specified for the j^{th} column, the capability box plot is based on the following values for the transformed observations:

$$Z_{ij} = \frac{Y_{ij} - (\mathrm{LSL}_j + \mathrm{USL}_j)/2}{\mathrm{USL}_j - \mathrm{LSL}_j}$$

Single Specification Limit and No Target

Suppose that only the lower specification limit is specified. (The case where only the upper specification limit is specified in a similar way.)

When no target is specified for the j^{th} column, the capability box plot is based on the following values for the transformed observations:

$$Z_{ij} = \frac{Y_{ij} - \bar{Y}_j}{2(\bar{Y}_j - \mathrm{LSL}_j)}$$

Note: When a column has only one specification limit and no target value, and the sample mean falls outside the specification interval, no capability box plot for that column is plotted.

Capability Indices for Normal Distributions

This section provides details about the calculation of capability indices for normal data.

For a process characteristic with mean μ and standard deviation σ, the population-based capability indices are defined as follows. For sample observations, the parameters are replaced by their estimates:

$$Cp = \frac{USL - LSL}{6\sigma}$$

$$Cpl = \frac{\mu - LSL}{3\sigma}$$

$$Cpu = \frac{USL - \mu}{3\sigma}$$

$$Cpk = \min(Cpl, Cpu)$$

$$Cpm = \frac{\min(T - LSL, USL - T)}{3\sigma\sqrt{1 + \left(\frac{T - \mu}{\sigma}\right)^2}}$$

$$\text{Target Index} = 3(Cp - Cpk)$$

The formulas use the following notation:

LSL = Lower specification limit

USL = Upper specification limit

T = Target value

For estimates of Within Sigma capability, σ is estimated using the subgrouping method that you specified. For estimates of Overall Sigma capability, σ is estimated using the sample standard deviation. If either of the specification limits is missing, the capability indices containing the missing specification limit are reported as missing.

Note: With the default AIAG (Ppk) Labeling, the indices based on Overall Sigma are denoted by Pp, Ppl, Ppu, and Ppk. The labeling for the index Cpm does not change when Overall Sigma is used. The formulas in this section are defined using Cp labels.

Confidence Intervals for Capability Indices

Confidence intervals for capability indices are available only for processes with normal distributions. Confidence intervals are calculated for both Within and Overall Sigma capability and are shown in the Individuals Detail Reports.

Cp

The 100(1 - α)% confidence interval for Cp is calculated as follows:

$$\left(\hat{Cp} \sqrt{\frac{\chi^2_{\alpha/2,\, df}}{df}}, \hat{Cp} \sqrt{\frac{\chi^2_{1-\alpha/2,\, df}}{df}} \right)$$

where

$\hat{Cp}$ is the estimated value for Cp

$\chi^2_{\alpha/2,\, df}$ is the $(\alpha/2)^{th}$ quantile of a chi-square distribution with *df* degrees of freedom

df is the degrees of freedom

N is the number of observations

m is the number of subgroups

For Overall Sigma capability, the degrees of freedom is equal to *N* - 1.

For Within Sigma capability, the degrees of freedom depends on the subgrouping and the within sigma estimation method.

- For Within Sigma capability with unbalanced subgroups, the degrees of freedom calculation is the same regardless of the within sigma estimation method. The degrees of freedom is equal to *N* - *m*.
- For Within Sigma capability with balanced subgroups of size $n = N/m$, the degrees of freedom calculation depends on the within sigma estimation method.
  - When Within Sigma is estimated by the average of the unbiased standard deviations, the degrees of freedom is equal to *f* * (*N* - *m*). The scale factor *f*, which varies between 0.875 and 1, is defined as follows:

$$f = \frac{1}{2(n-1)\left(\frac{(n-1)}{2} \left(\frac{\Gamma\left(\frac{n-1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)} \right)^2 - 1 \right)}$$

    where $\Gamma(n)$ is the gamma function evaluated at *n*.

    For more information, see Bissell (1990).
  - When Within Sigma is estimated by the average of ranges, the degrees of freedom is calculated as $df = 1/A - (3/16) * A + (3/64) * A^2 + 0.25$. *A* is defined as follows:

$$A = \frac{2d_3(n)^2}{m \cdot d_2(n)^2}$$

$d_2(n)$ is the expected value of the range of n independent normally distributed variables with unit standard deviation

$d_3(n)$ is the standard deviation of the range of n independent normally distributed variables with unit standard deviation

For more information, see David (1951).

– When Within Sigma is estimated by the unbiased pooled standard deviations, the degrees of freedom is equal to $N - m$.

- For Within Sigma capability with no subgroups, the degrees of freedom calculation depends on the within sigma estimation method.
  – When Within Sigma is estimated by the average moving ranges, the degrees of freedom is calculated as 0.62 * (N - 1).
  – When Within Sigma is estimated by the median moving ranges, the degrees of freedom is calculated as 0.32 * (N - 1).

  For more information, see Wheeler (2004, p. 82).

Cpk

The 100(1 - α)% confidence interval for Cpk is calculated as follows:

$$\left(\hat{C}_{pk}\left[1 - \Phi^{-1}{}_{1-\alpha/2}\sqrt{\frac{1}{9N\hat{C}_{pk}^2} + \frac{1}{2df}}\right], \hat{C}_{pk}\left[1 + \Phi^{-1}{}_{1-\alpha/2}\sqrt{\frac{1}{9N\hat{C}_{pk}^2} + \frac{1}{2df}}\right]\right)$$

where

$\hat{C}_{pk}$ is the estimated value for Cpk

$\Phi^{-1}{}_{1-\alpha/2}$ is the $(1 - \alpha/2)^{th}$ quantile of a standard normal distribution

df is the degrees of freedom

N is the number of observations

m is the number of subgroups

For Overall Sigma capability, the degrees of freedom is equal to N - 1.

For Within Sigma capability, the degrees of freedom depends on the subgrouping and the within sigma estimation method.

- For Within Sigma capability with unbalanced subgroups, the degrees of freedom calculation is the same regardless of the within sigma estimation method. The degrees of freedom is equal to $N - m$.
- For Within Sigma capability with balanced subgroups of size $n = N/m$, the degrees of freedom calculation depends on the within sigma estimation method.
  - When Within Sigma is estimated by the average of the unbiased standard deviations, the degrees of freedom is equal to $f * (N - m)$. The scale factor f, which varies between 0.875 and 1, is defined as follows:

$$f = \frac{1}{2(n-1)\left(\frac{(n-1)}{2}\left(\frac{\Gamma\left(\frac{n-1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}\right)^2 - 1\right)}$$

    where $\Gamma(n)$ is the gamma function evaluated at n.

    For more information, see Bissell (1990).
  - When Within Sigma is estimated by the average of ranges, the degrees of freedom is calculated as $df = 1/A - (3/16) * A + (3/64) * A^2 + 0.25$. A is defined as follows:

$$A = \frac{2d_3(n)^2}{m \cdot d_2(n)^2}$$

    $d_2(n)$ is the expected value of the range of n independent normally distributed variables with unit standard deviation

    $d_3(n)$ is the standard deviation of the range of n independent normally distributed variables with unit standard deviation

    For more information, see David (1951).
  - When Within Sigma is estimated by the unbiased pooled standard deviations, the degrees of freedom is equal to $N - m$.
- For Within Sigma capability with no subgroups, the degrees of freedom calculation depends on the within sigma estimation method.
  - When Within Sigma is estimated by the average moving ranges, the degrees of freedom is calculated as $0.62 * (N - 1)$.
  - When Within Sigma is estimated by the median moving ranges, the degrees of freedom is calculated as $0.32 * (N - 1)$.

  For more information, see Wheeler (2004, p. 82).

Cpm

Note: The confidence interval for Cpm is computed only when the target value is centered between the lower and upper specification limits.

The 100(1 - α)% confidence interval for Cpm is calculated as follows:

$$\left(\hat{\text{Cpm}} \sqrt{\frac{\chi^2_{\alpha/2,\gamma}}{\gamma}}, \hat{\text{Cpm}} \sqrt{\frac{\chi^2_{1-\alpha/2,\gamma}}{\gamma}} \right)$$

where

$\hat{\text{Cpm}}$ is the estimated value for Cpm

$\chi^2_{\alpha/2,\gamma}$ is the $(\alpha/2)^{\text{th}}$ quantile of a chi-square distribution with γ degrees of freedom

$$\gamma = \frac{N\left(1 + \left(\frac{\bar{x} - T}{s}\right)^2\right)^2}{1 + 2\left(\frac{\bar{x} - T}{s}\right)^2}$$

N is the number of observations

$\bar{x}$ is the mean of the observations

T is the target value

s is the sigma estimate

For Overall Sigma capability, s is the Overall Sigma estimate. For Within Sigma capability, s is replaced by the Within Sigma estimate.

Tip: For more information on confidence intervals for Cp, Cpk, and Cpm, see Pearn and Kotz (2006).

Cpl and Cpu

Lower and upper confidence limits for Cpl and Cpu are computed using the method of Chou et al. (1990).

The 100(1 - α)% confidence limits for Cpl (denoted by CPL_L and CPL_U) satisfy the following equations:

$$\Pr[t_{n-1}(\delta_L) \geq 3\hat{C}_{pl}\sqrt{n}] = \alpha/2 \quad \text{where} \quad \delta_L = 3CPL_L\sqrt{n}$$

$$\Pr[t_{n-1}(\delta_U) \leq 3\hat{C}_{pl}\sqrt{n}] = \alpha/2 \quad \text{where} \quad \delta_U = 3CPL_U\sqrt{n}$$

where

$t_{n\text{-}1}(\delta)$ has a non-central *t*-distribution with *n* - 1 degrees of freedom and noncentrality parameter δ

$\hat{C}_{pl}$ is the estimated value for Cpl

The 100(1 - α)% confidence limits for Cpu (denoted by CPU_L and CPU_U) satisfy the following equations:

$$\Pr[t_{n-1}(\delta_L) \geq 3\hat{C}_{pu}\sqrt{n}] = \alpha/2 \quad \text{where} \quad \delta_L = 3CPU_L\sqrt{n}$$

$$\Pr[t_{n-1}(\delta_U) \leq 3\hat{C}_{pu}\sqrt{n}] = \alpha/2 \quad \text{where} \quad \delta_U = 3CPU_U\sqrt{n}$$

where

$t_{n\text{-}1}(\delta)$ has a non-central *t*-distribution with *n* - 1 degrees of freedom and noncentrality parameter δ

$\hat{C}_{pu}$ is the estimated value for Cpu

Capability Indices for Nonnormal Distributions: Percentile and Z-Score Methods

This section describes how capability indices are calculated for nonnormal distributions. Two methods are described: the Percentile (also known as ISO/Quantile) method and the Z-Score (also known as Bothe/Z-scores) method. When you select a distribution for a nonnormal process variable, you can fit a parametric distribution or a nonparametric distribution. You can use either the Percentile or the Z-Score methods to calculate capability indices for the process variable of interest. However, unless you have a very large amount of data, a nonparametric fit might not accurately reflect behavior in the tails of the distribution.

Note: For both the Percentile and the Z-Score methods, if the data are normally distributed, the capability formulas reduce to the formulas for normality-based capability indices.

The descriptions of the two methods use the following notation:

LSL = Lower specification limit

USL = Upper specification limit

T = Target value

Percentile (ISO/Quantile) Method

The percentile method replaces the mean in the standard capability formulas with the median of the fitted distribution and the 6σ range of values with the corresponding percentile range. The method is described in AIAG (2005).

Denote the α*100th percentile of the fitted distribution by P_α. Then Percentile method capability indices are defined as follows:

$$P_{pk} = \min\left(\frac{P_{0.5} - LSL}{P_{0.5} - P_{0.00135}}, \frac{USL - P_{0.5}}{P_{0.99865} - P_{0.5}}\right)$$

$$P_{pl} = \frac{P_{0.5} - LSL}{P_{0.5} - P_{0.00135}}$$

$$P_{pu} = \frac{USL - P_{0.5}}{P_{0.99865} - P_{0.5}}$$

$$P_p = \frac{USL - LSL}{P_{0.99865} - P_{0.00135}}$$

$$C_{pm} = \frac{\min\left(\frac{T - LSL}{P_{0.5} - P_{0.00135}}, \frac{USL - T}{P_{0.99865} - P_{0.5}}\right)}{\sqrt{1 + \left(\frac{\mu - T}{\sigma}\right)^2}}$$

Z-Score (Bothe/Z-Scores) Method

The Z-Score method transforms the specification limits to values that have the same probabilities on a standard normal scale. It computes capability measures that correspond to a normal distribution with the same risk levels as the fitted nonnormal distribution.

Let F denote the fitted distribution for a process variable with lower and upper specification limits given by LSL and USL. Equivalent standard normal specification limits are defined as follows:

$$\begin{bmatrix} LSL_F = \Phi^{-1}(F(LSL)) \\ USL_F = \Phi^{-1}(F(USL)) \end{bmatrix}$$

Then the Z-Score method capability indices are defined as follows:

$$P_{pk} = \min(-LSL_F/3, USL_F/3)$$

$$P_{pl} = -LSL_F/3$$

$$P_{pu} = USL_F/3$$

$$P_p = (USL_F - LSL_F)/6$$

Note: Because Cpm is a target-based measure, it cannot be calculated using the Z-Scores method.

Note: For very capable data, *F(LSL)* or *F(USL)* can be so close to zero or one, respectively, that LSL_F or USL_F cannot be computed. In these cases, JMP automatically switches from the Z-Score method to the Percentile method by default. This gives more meaningful capability indices. To turn off this default setting, select File > Preferences > Platforms > Process Capability.

Parameterizations for Distributions

This section gives the density functions f for the distributions used in the Process Capability platform. It also gives expected values and variances for all but the Johnson and SHASH distributions.

Normal

$$f(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{1}{2\sigma^2}(x-\mu)^2\right] \quad , \quad -\infty < x < \infty \quad , \quad -\infty < \mu < \infty \quad , \quad \sigma > 0$$

$E(X) = \mu$

$Var(X) = \sigma^2$

Beta

$$f(x|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1} \quad , \quad 0 \le x \le 1 \quad , \quad \alpha > 0 \quad , \quad \beta > 0$$

$$E(X) = \frac{\alpha}{\alpha + \beta}$$

$$Var(X) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

where $B(\cdot)$ is the Beta function.

Exponential

$$f(x|\sigma) = \frac{1}{\sigma}\exp(-x/\sigma) \quad , \quad x > 0 \quad , \quad \sigma > 0$$

$E(X) = \sigma$

$Var(X) = \sigma^2$

Gamma

$$f(x|\alpha, \sigma) = \frac{1}{\Gamma(\alpha)\sigma^{\alpha}} x^{\alpha-1}\exp(-x/\sigma) \quad , \quad x > 0 \quad , \quad \alpha > 0 \quad , \quad \sigma > 0$$

$E(X) = \alpha\sigma$

$Var(X) = \alpha\sigma^2$

where $\Gamma(\cdot)$ is the gamma function.

Johnson

Johnson Su

$$f(x|\gamma, \delta, \sigma, \theta) = \frac{\delta}{\sigma}\left[1 + \left(\frac{x-\theta}{\sigma}\right)^2\right]^{-1/2} \phi\left[\gamma + \delta \sinh^{-1}\left(\frac{x-\theta}{\sigma}\right)\right] \quad , \quad -\infty < x, \theta, \gamma < \infty \quad , \quad \theta > 0 \quad ,$$

$\delta > 0$

Johnson Sb

$$f(x|\gamma, \delta, \sigma, \theta) = \phi\left[\gamma + \delta \ln\left(\frac{x-\theta}{\sigma-(x-\theta)}\right)\right]\left(\frac{\delta\sigma}{(x-\theta)(\sigma-(x-\theta))}\right), \quad \theta < x < \theta + \sigma, \quad \sigma > 0$$

Johnson SI

$$f(x|\gamma, \delta, \sigma, \theta) = \frac{\delta}{|x-\theta|}\phi\left[\gamma + \delta \ln\left(\frac{x-\theta}{\sigma}\right)\right], \quad x > 0 \text{ if } \sigma = 1, \quad x < 0 \text{ if } \sigma = -1$$

where $\phi(\cdot)$ is the standard normal probability density function.

Lognormal

$$f(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \frac{\exp\left[\frac{-(\log(x)-\mu)^2}{2\sigma^2}\right]}{x}, \quad x > 0, \quad -\infty < \mu < \infty, \quad \sigma > 0$$

$$E(X) = \exp(\mu + \sigma^2/2)$$

$$\text{Var}(X) = \exp(2(\mu + \sigma^2)) - \exp(2\mu + \sigma^2)$$

Mixture of Normals

The Mixture of 2 Normals and Mixture of 3 Normals options for Distribution share the following parameterization:

$$f(x|\mu_i, \sigma_i, \pi_i) = \sum_{i=1}^{k} \frac{\pi_i}{\sigma_i} \phi\left(\frac{x-\mu_i}{\sigma_i}\right)$$

$$E(X) = \sum_{i=1}^{k} \pi_i \mu_i$$

$$\text{Var}(X) = \sum_{i=1}^{k} \pi_i(\mu_i^2 + \sigma_i^2) - \left(\sum_{i=1}^{k} \pi_i \mu_i\right)^2$$

where μ_i, σ_i, and π_i are the respective mean, standard deviation, and proportion for the i^{th} group, and $\phi(\cdot)$ is the standard normal probability density function. For the Mixture of 2 Normals, k is equal to 2. For the Mixture of 3 Normals distribution, k is equal to 3. A separate mean, standard deviation, and proportion of the whole is estimated for each group in the mixture.

SHASH

$$f(x|\gamma, \delta, \sigma, \theta) = \frac{\delta \cosh(w)}{\sqrt{\sigma^2 + (x-\theta)^2}} \phi[\sinh(w)], \quad -\infty < \gamma, x, \theta < \infty, \; 0 < \delta, \; 0 < \sigma$$

where

$\phi(\cdot)$ is the standard normal pdf

$$w = \gamma + \delta \sinh^{-1}\left(\frac{x-\theta}{\sigma}\right)$$

Note: When $\gamma = 0$ and $\delta = 1$, the SHASH distribution is equivalent to the normal distribution with location θ and scale σ.

Weibull

$$f(x|\alpha, \beta) = \frac{\beta}{\alpha^\beta} x^{\beta-1} \exp\left[-\left(\frac{x}{\alpha}\right)^\beta\right], \quad \alpha > 0, \quad \beta > 0$$

$$E(X) = \alpha \Gamma\left(1 + \frac{1}{\beta}\right)$$

$$Var(X) = \alpha^2 \left\{ \Gamma\left(1 + \frac{2}{\beta}\right) - \Gamma^2\left(1 + \frac{1}{\beta}\right) \right\}$$

where $\Gamma(\cdot)$ is the gamma function.

Chapter 8

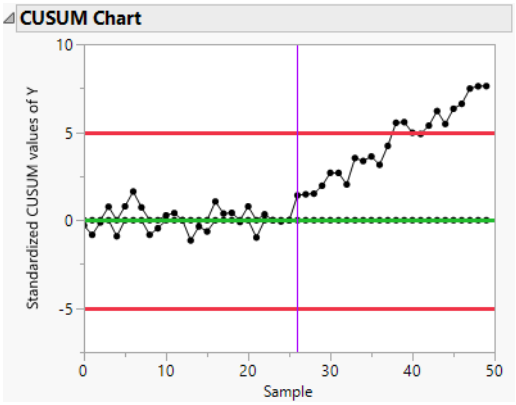
CUSUM Control Charts

Create Tabular CUSUM Control Charts with Decision Limits

Cumulative Sum (CUSUM) control charts enable you to detect small shifts in a process. They are useful in detecting shifts that occur over time, such as a gradual drift, and that are not necessarily accompanied by a sudden shift. The CUSUM Control Chart platform creates a CUSUM chart with decision limits, similar to a Shewhart chart. This chart is also called a *tabular CUSUM chart*. To create a V-mask cumulative sum control chart, see "V-Mask CUSUM Control Charts".

The CUSUM Control Chart platform also provides information about average run length (ARL). The *average run length* is the average number of samples or observations that can be expected to occur before an out-of-control signal occurs. You can use the average run length to assess the performance of a CUSUM chart, given specific parameters and assuming constant variance.

Figure 8.1 CUSUM Control Chart



Contents

Overview of the CUSUM Control Chart Platform 247
Example of CUSUM Control Chart 247
Launch the CUSUM Control Chart Platform 249
The CUSUM Control Chart Platform Report 250
Control Panel for CUSUM Control Chart 251
CUSUM Chart 252
CUSUM Control Chart Platform Options 252
Average Run Length (ARL) Report 254
Additional Examples of CUSUM Control Charts 255
Example of the Data Units Option in a CUSUM Control Chart 255
Example of CUSUM Chart with Subgroups 256
Statistical Details for the CUSUM Control Chart Platform 258
Statistical Details for CUSUM Control Chart Construction 258
Statistical Details for Shift Detection 259
Statistical Details for Average Run Length 259

Overview of the CUSUM Control Chart Platform

A tabular CUSUM chart consists of two one-sided decision limits charts superimposed on one chart. The chart contains decision limits that signal when the process is out of control and places a shift line on the chart where the shift is suspected to have occurred. To use the CUSUM Control Chart platform, you must determine the smallest change in the mean that you consider important. You can view the CUSUM control chart in standard deviation units or in data units. For more information about tabular CUSUM charts, see Woodall and Adams (1998) and Montgomery (2013).

Another form of a cumulative sum control chart is the V-mask chart. To create a V-mask CUSUM chart, see "V-Mask CUSUM Control Charts".

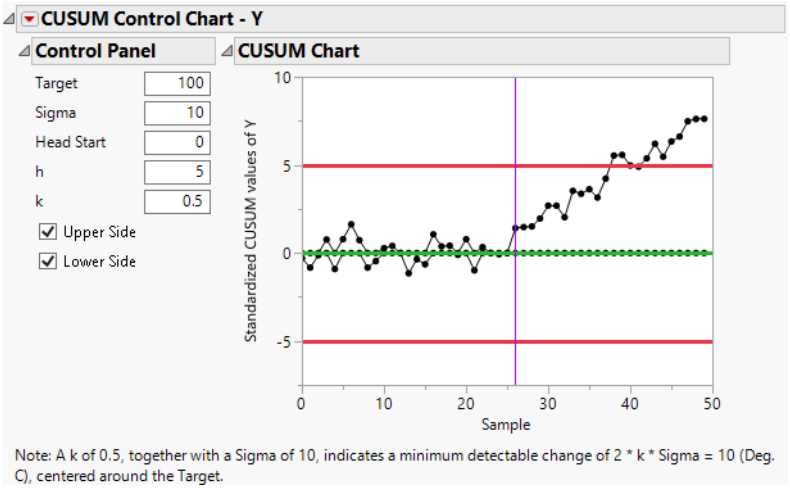
Note: The summary results in the CUSUM Control Chart platform do not always match the summary results in the V-mask CUSUM platform. Specifically, the summary results for a two-sided V-mask CUSUM chart do not match those from a CUSUM Control Chart with both Upper Side and Lower Side options selected. However, the one-sided summary reports from the CUSUM Control Chart platform and the V-mask CUSUM platform do match.

Example of CUSUM Control Chart

You want to detect small shifts in the temperature of an engine. The data table contains temperature measurements from the engine thermostat.

1. Select **Help > Sample Data Library** and open Quality Control/Engine Temperature Sensor.jmp.
2. Select **Analyze > Quality and Process > Control Chart > CUSUM Control Chart**.
3. Select Y and click **Y**.
4. Click **OK**.
5. In the Target box, type 100.
6. In the Sigma box, type 10.

Figure 8.2 CUSUM Control Chart Report



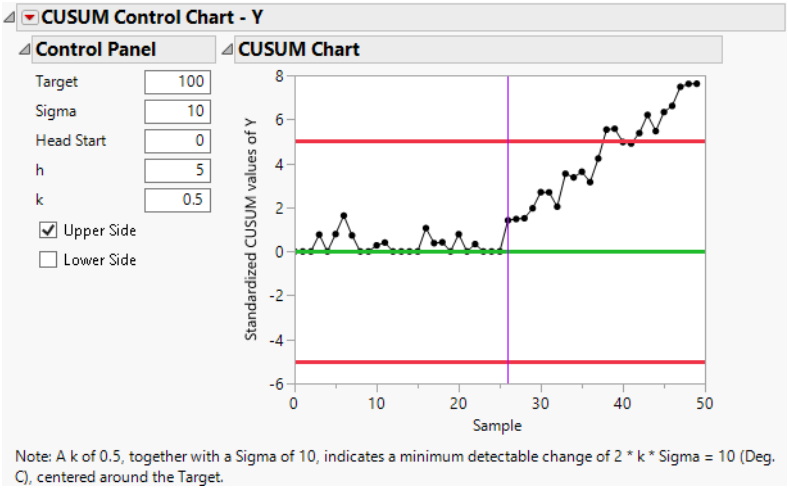
The vertical line on the CUSUM Chart indicates that a shift in the temperature measurements started around sample 26.

Note: You can compare this result to the Individual Moving Range control chart by running the IMR Chart table script in Engine Temperature Sensor.jmp. The IMR chart does not trigger any of the Nelson tests.

Example of a One-Sided CUSUM Control Chart

Continuing the previous example, suppose that you care only about increasing temperature changes. To change the CUSUM control chart in Figure 8.2 to a one-sided chart, deselect the Lower Side check box. When you do that, the points for the negative cumulative sums are removed from the chart. You are left with a CUSUM control chart that contains only the positive cumulative sum points.

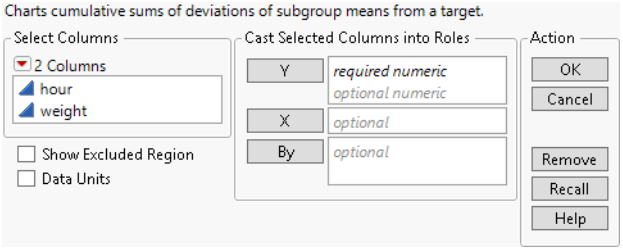
Figure 8.3 One-Sided CUSUM Control Chart Report



Launch the CUSUM Control Chart Platform

Launch the CUSUM Control Chart platform by selecting **Analyze > Quality and Process > Control Chart > CUSUM Control Chart**.

Figure 8.4 The CUSUM Control Chart Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The CUSUM Control Chart platform launch window contains the following options:

Y Identifies the variables that you want to chart.

Note: The rows of the data table must be sorted in the order in which the observations were collected.

X Identifies a subgroup variable. The horizontal axis of the CUSUM chart is labeled by the subgroup variable. If a value of this column is present more than once, the average response at each X value is plotted on the CUSUM chart.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Show Excluded Region (Applicable only when an X variable is specified.) Specifies that subgroups that are entirely excluded are shown on the horizontal axis in the CUSUM control chart.

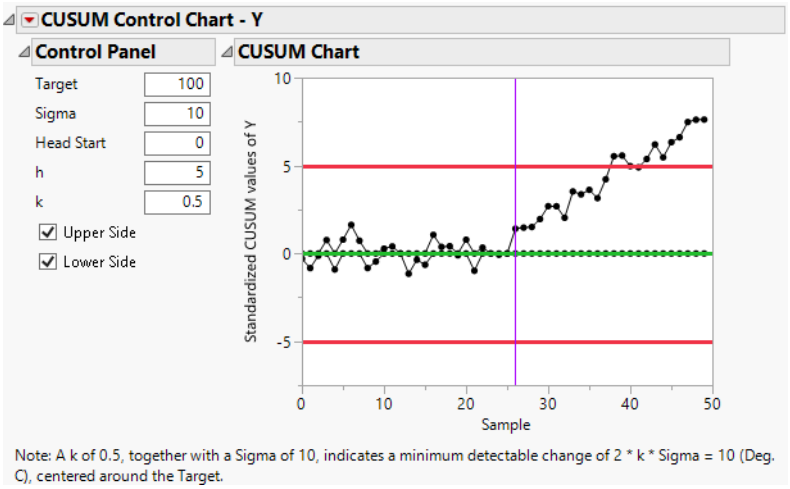
Data Units Specifies that data units be used in the report rather than standard deviation units. By default, the chart and parameters are shown in standard deviation units. However, if you select the Data Units option in the launch window, the chart and parameters are shown in the units of the data column that is being analyzed.

When you use standard deviation units, values for the *h* and *k* parameters do not depend on the process standard deviation. This can be an advantage.

The CUSUM Control Chart Platform Report

By default, the CUSUM Control Chart platform produces a report that contains a parameter control panel and a CUSUM chart.

Figure 8.5 CUSUM Control Chart Report



Control Panel for CUSUM Control Chart

The Control Panel report contains the current values for the chart parameters. The current values are in boxes that enable you to update the parameter values. There are also boxes for Upper Side and Lower Side. If you have specified the Data Units option in the launch window, this setting is denoted in the Control Panel below the check boxes.

The following options appear in the Control Panel report:

Target The known value of the mean. This is the value of the center line in the chart. By default, this parameter is set to the Target value in the Spec Limits column property for the Y column. If the Y column does not have a Target value in the Spec Limits column property, this parameter is set to the overall average of the Y column.

Note: To use the overall average of the Y column as the value of the center line even if the Y column has a Target value in the Spec Limits column property, select the Use Process Mean for Center Line platform preference. This preference is located in File > Preferences > Platforms > CUSUM Control Chart.

Sigma The known value of the standard deviation. By default, this parameter is set to the average moving range of the Y column. If there is an X variable, the Sigma parameter is set to the average moving range of the summary data.

Head Start The value of the cumulative sums before the first sample. Starting the cumulative sums at a nonzero value increases the sensitivity of the CUSUM chart near the beginning of the samples. This parameter is also known as the fast initial response (FIR) value. By default, this parameter is set to 0.

h or H The value of the parameter that defines the limits. If the Data Units option was not selected in the launch window, this is the h parameter. If the Data Units option was selected in the launch window, this is the H parameter. Note that H is equal to h times Sigma. By default, h is equal to 5 and H is equal to 5 times Sigma.

k or K The value of the parameter that defines the smallest change in the mean that is valuable to detect. If the Data Units option was not selected in the launch window, this is the k parameter. If the Data Units option was selected in the launch window, this is the K parameter. Note that K is equal to k times Sigma. By default, k is equal to 0.5 and K is equal to one half of Sigma.

Upper Side Shows or hides the positive values for the cumulative sum on the chart. These values are the C^+ values.

Lower Side Shows or hides the negative values for the cumulative sum on the chart. These values are the C^- values.

Using Data Units The presence of this text indicates that the Data Units option was selected in the launch window and that the values in the CUSUM chart are centered but not standardized.

CUSUM Chart

The CUSUM Chart report contains the cumulative sum control chart with decision limits that are determined by the current values of the chart parameters. The samples (or subgroups if you specified an X variable) are denoted on the horizontal axis. The vertical axis denotes centered values of the positive and negative values for the cumulative sum. If the Data Units option was not selected in the launch window, the vertical axis denotes cumulative sums for standardized response values. If the Data Units option was selected in the launch window, the vertical axis denotes cumulative sums for unstandardized response values.

CUSUM Control Chart Platform Options

The CUSUM Control Chart red triangle menu contains the following options:

Show Limits Shows or hides the upper and lower decision limits in the CUSUM Chart.

Show Center Line Shows or hides the center line in the CUSUM Chart.

Show Shift Lines (Available only when there is a shift detected in the data.) Shows or hides the vertical lines in the CUSUM Chart that designate shifts. Shift lines are drawn at the start of a shift.

- A positive shift occurs when the value of C^+ exceeds the upper limit on the chart. The start of the shift is defined as the first point after the most recent zero value for C^+.
- A negative shift occurs when the value of C^- falls below the lower limit on the chart. The start of the shift is defined as the first point after the most recent zero value for C^-.

Show ARL Shows or hides the Average Run Length (ARL) report. See "Average Run Length (ARL) Report".

ARL Profiler Shows or hides a profiler of average run length versus the parameters h and k. If you have specified the Data Units option in the launch window, the profiler plots average run length versus the parameters H and K.

The average run length (ARL) for a specified shift is the average number of runs expected before an out-of-control signal occurs. For example, the ARL at 0 represents the average

number of runs expected before a false-alarm signal occurs when the process is in control. When the process is in control, the shift size is 0.

The ARL Profiler enables you to explore how various settings of the parameters affect the performance of the corresponding CUSUM chart. As the parameters in the Control Panel report are updated, the ARL Profiler is updated as well. An ideal CUSUM chart has a high ARL(0) value and a low ARL(Δ) value, where Δ is the size shift that is of interest.

The ARL Profiler depends on the settings of the Upper Side and Lower Side options in the Control Panel report:

- If both the Upper Side and Lower Side options are selected, the profiler represents the average run length for crossing either the upper or lower decision limits on the CUSUM chart.
- If only the Upper Side option is selected, the profiler represents the average run length for the upper decision limit on the CUSUM chart.
- If only the Lower Side option is selected, the profiler represents the average run length for the lower decision limit on the CUSUM chart.

For more information about the options in the red triangle menu next to ARL Profiler, see *Profilers*.

Control Panel Shows or hides a report of the current values of the parameters. This report enables you to change the parameter values as well as the sidedness of the CUSUM chart.

Parameters Report Shows or hides a report of the current values of the parameters.

Test Beyond Limits Shows or hides a red circle around any point that is above the upper limit or below the lower limit in the CUSUM chart.

Save Summaries Creates a new data table that contains statistics for each subgroup in the CUSUM chart. The following statistics are saved to the new data table: the subgroup number and size, the subgroup sample mean, an indicator of shift starts, a value that indicates each interval between shift starts, the upper and lower cumulative sums and corresponding consecutive run counts, and the LCL and UCL values.

Tune Chart Shows or hides a control that enables you to set an acceptable range for the Y variable. Adjust the minimum and maximum values of the acceptable range and click **Done**. At this point, the CUSUM chart updates based on a new value of the k parameter.

- If you specified the Data Units option in the launch window, the imputed k parameter is the average of the minimum and maximum values.

– If the Data Units option is not specified, the imputed K parameter is the average of the minimum and maximum values divided by the Sigma parameter.

Setting the acceptable range for the Y variable enables you to set the practical significance of the CUSUM chart. This is particularly helpful when the testing interval is more frequent, which can result in a much shorter practical average run length.

Reset to Defaults Resets all parameters back to their default values.

Alarm Script Enables you to write and run a script that indicates when the data fail special causes tests. Results can be written to a file, written to the log or sent in an email. There is an option to include an explanation of why the test failed.

As an Alarm Script is invoked, the following variables are available, both in the issued script and in subsequent JSL scripts:

`qc_col` is the name of the column

`qc_test` is the test that failed

`qc_sample` is the sample number

Tip: After an alarm script is specified, the alarm script is invoked when the Test Beyond Limits option is turned on.

See the *Scripting Guide* for more information about writing custom Alarm Scripts.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Average Run Length (ARL) Report

The Average Run Length (ARL) report contains a table and a graph of ARL values. The average run length (ARL) for a specified shift is the average number of runs expected before an out-of-control signal occurs. For example, the ARL at 0 represents the average number of runs expected before seeing a false-alarm signal when the process is in control. When the process is in control, the shift size is 0.

The table and graph in the ARL report enable you to explore how various settings of the parameters affect the performance of the corresponding CUSUM chart. As the *h* and *k* parameters in the Control Panel report are updated, the ARL report is updated as well. An ideal CUSUM chart has a high ARL(0) value and a low ARL(Δ) value, where Δ is the size shift that is of interest.

The Average Run Length (ARL) report depends on the settings of the Upper Side and Lower Side options in the Control Panel report. If only one option is selected, the ARL report uses calculations for the corresponding one-sided CUSUM chart. If both options are selected, the ARL report uses calculations for the two-sided CUSUM chart. Note that the two-sided ARL values are related to the positive and negative one-sided ARL values by the following equation:

$$\frac{1}{\text{ARL}} = \frac{1}{\text{Positive ARL}} + \frac{1}{\text{Negative ARL}}$$

ARL Table

The ARL Table shows the average run length for shifts (Δ) between zero and three at 0.25 increments. If the Data Units option is specified, the shift is represented by $2*K/Sigma^2$. If the Data Units option is not specified, the shift is represented by 2*k/Sigma.

ARL Graph

The ARL Graph shows the average run length for shifts (Δ) between 0 and 3. This graph contains the same data points as the ARL Table to the left of the ARL Graph.

Additional Examples of CUSUM Control Charts

- "Example of the Data Units Option in a CUSUM Control Chart"
- "Example of CUSUM Chart with Subgroups"

Example of the Data Units Option in a CUSUM Control Chart

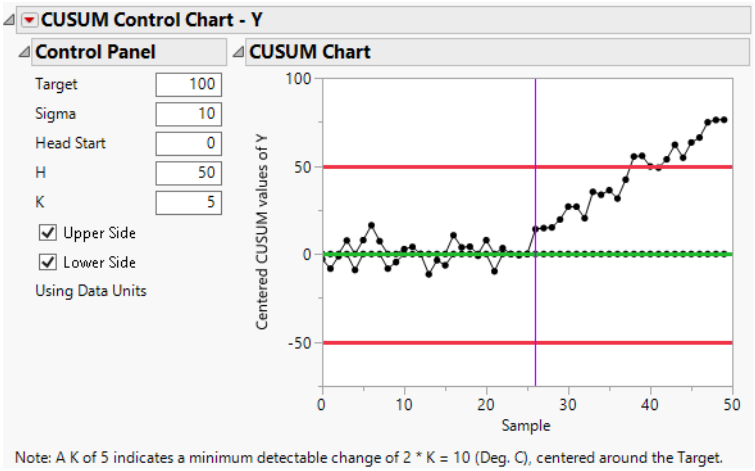
This example uses the Data Units option and reproduces the analysis in "Example of CUSUM Control Chart". You want to detect small shifts in the temperature of an engine. The data table contains temperature measurements from the engine thermostat.

1. Select **Help > Sample Data Library** and open Quality Control/Engine Temperature Sensor.jmp.
2. Select **Analyze > Quality and Process > Control Chart > CUSUM Control Chart**.

3. Select Y and click **Y**.
4. Select the box next to Data Units.
5. Click **OK**.
6. In the Target box, type 100.
7. In the Sigma box, type 10.

   Note that the options below Head Start are H and K, instead of h and k. These parameters are now specified in units of the data column, rather than in standard deviation units.

Figure 8.6 CUSUM Control Chart Report



Like in the example using sigma units, the vertical line on the CUSUM Chart indicates that a shift in the temperature measurements started around sample 26.

Example of CUSUM Chart with Subgroups

A machine fills 8-ounce cans of two-cycle engine oil additive. The filling process is believed to be in statistical control. The process is set so that the average weight of a filled can (μ_0) is 8.10 ounces. Previous analysis shows that the standard deviation of fill weights (σ_0) is 0.05 ounces.

Subgroup samples of four cans are selected and weighed every hour for twelve hours. Each observation in the Oil1 Cusum.jmp data table contains one value of weight and its associated value of hour. The observations are sorted so that the values of hour are in increasing order. You want to be able to detect a 2σ shift in the process.

1. Select **Help > Sample Data Library** and open Quality Control/Oil1 Cusum.jmp.
2. Select **Analyze > Quality and Process > Control Chart > CUSUM Control Chart**.
3. Select weight and click **Y**.

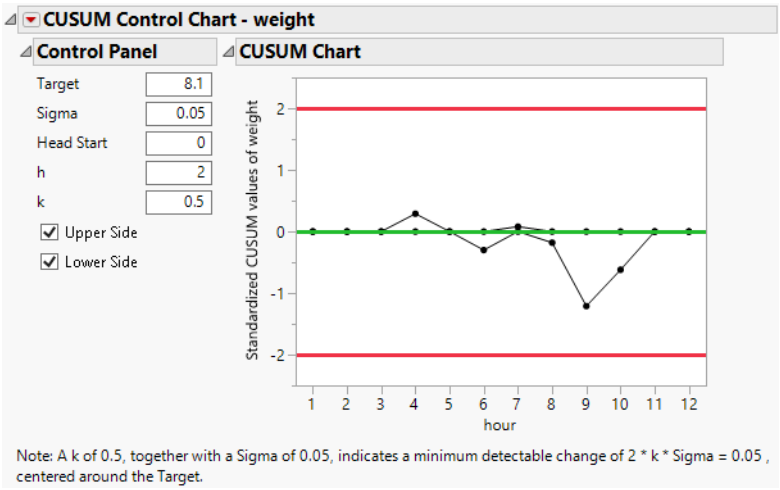
4. Select hour and click **X**.
5. Click **OK**.
6. In the Target box, type 8.1.

   This is the target mean for the process.
7. In the Sigma box, type 0.05.

   This is the known standard deviation for the process.
8. In the h box, type 2.

   This defines the decision limits to be 2 standard deviations in each direction.

Figure 8.7 CUSUM Control Chart with Subgroups



The CUSUM Chart does not show any points outside of the upper or lower decision limits. There is no evidence that a shift in the process has occurred.

Note: Montgomery (2013) states that "only if there is some significant economy of scale or some other valid reason for taking samples of size greater than one should subgroups of size greater than one be used with the CUSUM." The use of rational subgroups in the tabular CUSUM chart does not always improve the performance of the chart.

Statistical Details for the CUSUM Control Chart Platform

- "Statistical Details for CUSUM Control Chart Construction"
- "Statistical Details for Shift Detection"
- "Statistical Details for Average Run Length"

Statistical Details for CUSUM Control Chart Construction

This section defines the statistics that are used in the construction of the CUSUM Chart. Some of these statistics are also saved in the data table that is created by the Save Summaries command.

One-Sided Upper and Lower Cumulative Sums

The definitions of C^+ and C^- depend on the setting of the Data Units option.

Note: In the Save Summaries data table, C^+ and C^- are labeled Upper Cumulative Sum and Lower Cumulative Sum, respectively.

Cumulative Sums in Standardized Units

If the Data Units option is not selected, C^+ and C^- for each step are defined as follows:

$$C_i^+ = \max\left(0, \frac{x_i - T}{\sigma} - k + C_{i-1}^+\right)$$

$$C_i^- = \min\left(0, \frac{x_i - T}{\sigma} + k + C_{i-1}^-\right)$$

where:

x_i is the value of the response at the i^{th} step

T is the target of the process

σ is the standard deviation of the process

k is the reference value, in units of standard deviations

If a value is specified for Head Start, that value is used as the initial C^+ value and the negative of that value is used as the initial C^- value. Otherwise, the initial values of C^+ and C^- are zero.

Cumulative Sums in Data Units

If the Data Units option is selected, C^+ and C^- for each step are defined as follows:

$$C_i^+ = \max(0, (x_i - T) - K + C_{i-1}^+)$$

$$C_i^- = \min(0, (x_i - T) + K + C_{i-1}^-)$$

where:

x_i is the value of the response at the i^{th} step

T is the target of the process

σ is the standard deviation of the process

K is the reference value, in units of the data

If a value is specified for Head Start, that value is used as the initial C^+ value and the negative of that value is used as the initial C^- value. Otherwise, the initial values of C^+ and C^- are zero.

Counters for Positive and Negative Runs

N^+ at each step is the number of steps since the most recent zero value for C^+. N^- at each step is the number of steps since the most recent zero value for C^-.

Note: In the Save Summaries data table, N^+ and N^- are labeled Positive Runs and Negative Runs, respectively.

Statistical Details for Shift Detection

A positive shift occurs when the value of C^+ exceeds the upper limit on the chart. The start of the shift is defined as the first point after the most recent zero value for C^+.

A negative shift occurs when the value of C^- exceeds the lower limit on the chart. The start of the shift is defined as the first point after the most recent zero value for C^-.

Statistical Details for Average Run Length

The one-sided average run length (ARL) values are calculated using the integral equation method (with 24 Gaussian points) described by Goel and Wu (1971). If the Head Start value is greater than 0, the values are calculated according to the method in Appendix A.1 of Lucas and Crosier (1982).

Note that the two-sided ARL values are related to the positive and negative one-sided ARL values by the following equation:

$$\frac{1}{\text{ARL}} = \frac{1}{\text{Positive ARL}} + \frac{1}{\text{Negative ARL}}$$

Lucas and Crosier (1982) describe the properties of a Head Start value for CUSUM charts in which the initial CUSUM S_0 is set to a nonzero value. This is sometimes referred to as a fast initial response (FIR) feature. Average run length calculations given by them show that the FIR feature has little effect when the process is in control and that it leads to a faster response to an initial out-of-control condition than a standard CUSUM chart.

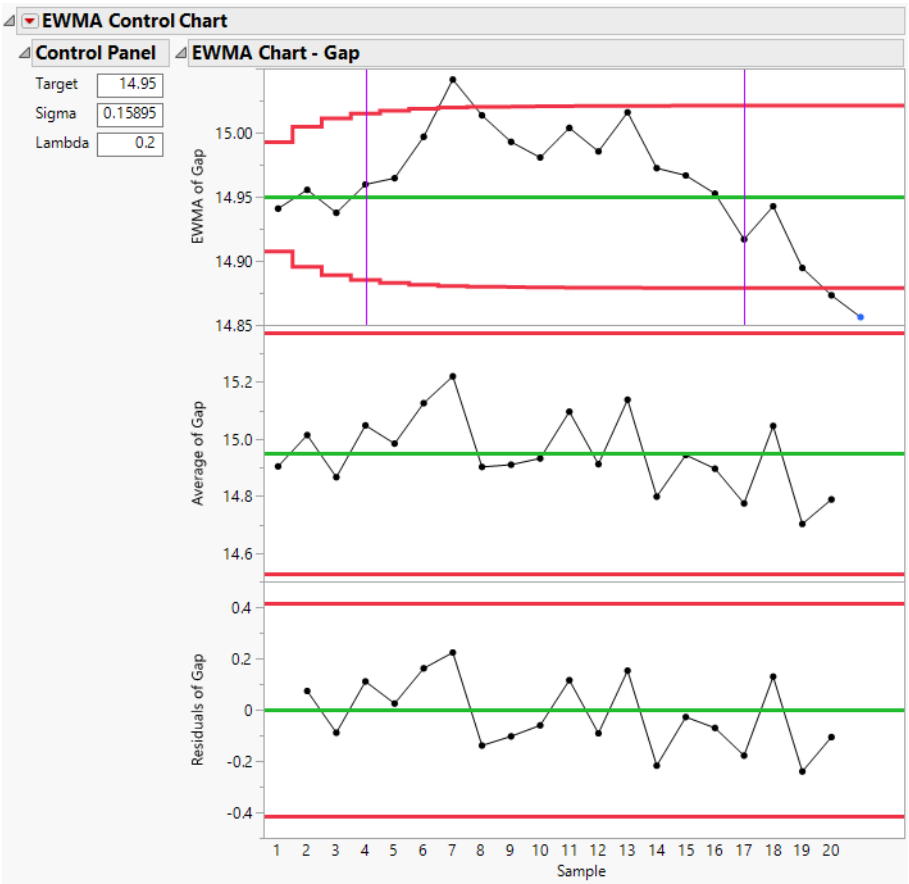
Chapter 9

EWMA Control Charts

Create Exponentially Weighted Control Charts

Exponentially weighted moving average (EWMA) charts can be used to detect small shifts in a process. Each point on an EWMA chart is the weighted average of all the previous subgroup means, including the mean of the present subgroup sample. The weights decrease exponentially going backward in time.

Figure 9.1 EWMA Control Chart Report



Contents

Overview of the EWMA Control Chart Platform 263
Example of the EWMA Control Chart Platform 263
Launch the EWMA Control Chart Platform 265
The EWMA Control Chart Platform Report 266
Control Panel for EWMA Control Chart 266
EWMA Chart Report 267
The EWMA Control Chart Platform Options 269
Average Run Length (ARL) Report for EWMA Control Charts 272
Additional Example of the EWMA Control Chart Platform 273
Statistical Details for the EWMA Control Chart Platform 276

Overview of the EWMA Control Chart Platform

Exponentially weighted moving average (EWMA) charts can be used to detect small shifts in a process. Each point on an EWMA chart is the weighted average of all the previous subgroup means, including the mean of the present subgroup sample. The weights decrease exponentially going backward in time. For more information about exponentially weighted moving average charts, see Box et al. (2009) and Montgomery (2013).

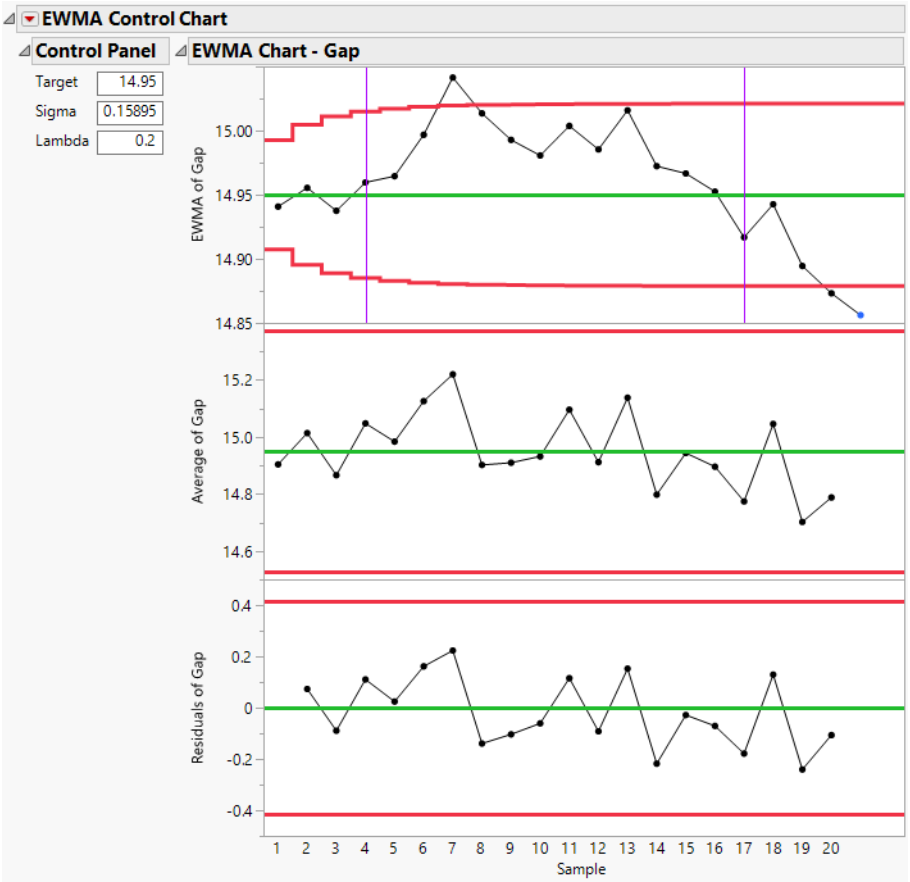
The EWMA Control Chart platform pairs an EWMA chart with an X chart and a residual chart. If you do not specify a Subgroup variable, the X chart is an individual measurements chart. If you specify a Subgroup variable and at least one subgroup size is greater than 1, the X chart is an XBar chart.

Example of the EWMA Control Chart Platform

In the sample data table Clips1.jmp, the measure of interest is the gap between the ends of manufactured metal clips. To monitor the process for a change in the average gap, subgroup samples of five clips have been measured daily.

1. Select **Help > Sample Data Library** and open Quality Control/Clips1.jmp.
2. Select **Analyze > Quality and Process > Control Chart > EWMA Control Chart**.
3. Select Gap and click **Y**.
4. Select Sample and click **Subgroup**.
5. Click **OK**.

Figure 9.2 EWMA Control Chart

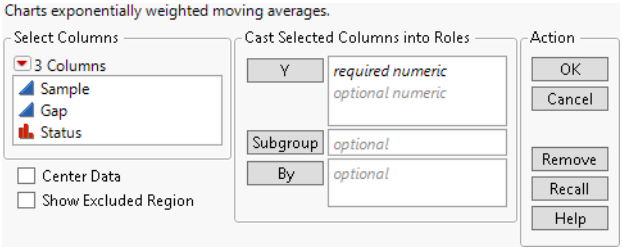


Purple vertical lines in the EWMA chart denote shifts. Shift starts are detected at samples 4 and 17.

Launch the EWMA Control Chart Platform

Launch the EWMA Control Chart platform by selecting **Analyze > Quality and Process > Control Chart > EWMA Control Chart**.

Figure 9.3 EWMA Control Chart Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The EWMA Control Chart platform launch window contains the following options:

Y Identifies the variables that you want to chart.

> **Note:** If you do not specify a Subgroup variable, the rows of the data table must be sorted in the order in which the observations were collected.

Subgroup Identifies a subgroup variable. The horizontal axis of the EWMA chart is labeled by the subgroup variable.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

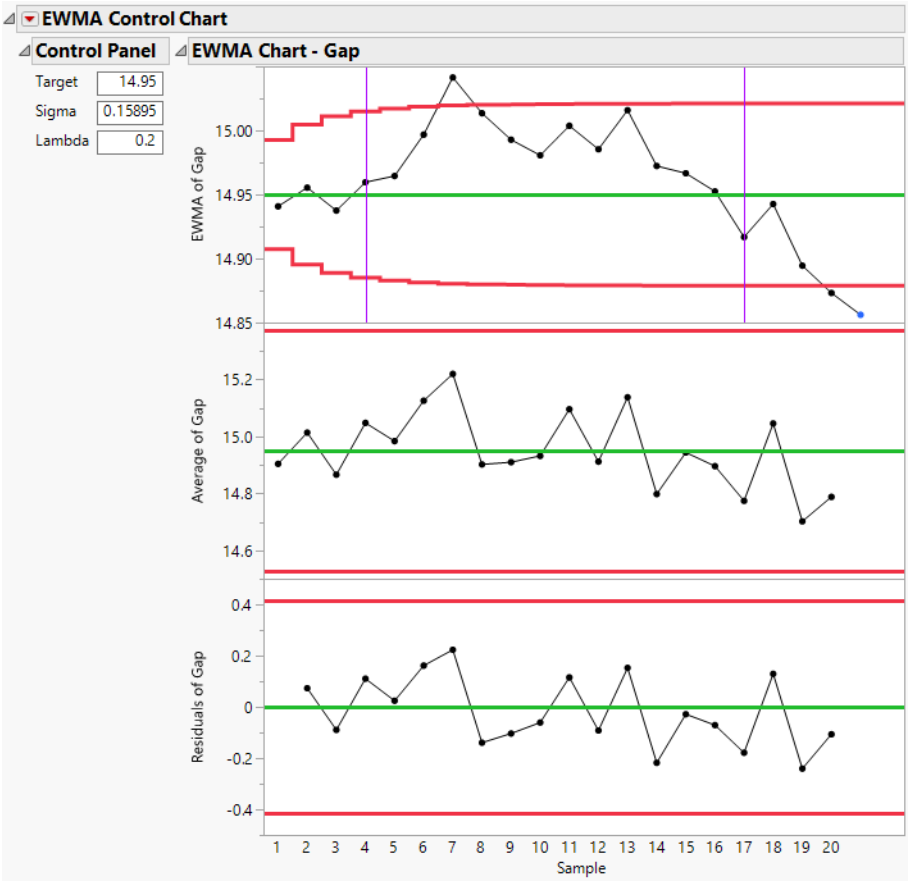
Center Data Specifies that the data are centered by subtracting the target from each observation.

Show Excluded Region (Applicable only when a Subgroup variable is specified.) Specifies that subgroups that are entirely excluded in the data table are shown in the EWMA control chart.

The EWMA Control Chart Platform Report

By default, the EWMA Control Chart platform produces a report that contains a parameter control panel, an EWMA chart, an X chart, and a residuals chart.

Figure 9.4 EWMA Control Chart Report



Control Panel for EWMA Control Chart

The Control Panel contains the current values for the chart parameters. The current values are in boxes that enable you to update the parameter values.

The following options appear in the Control Panel:

Target The known value of the mean. This is the value of the center line in the chart. By default, this parameter is set to the Target value in the Spec Limits column property for the Y column. If the Y column does not have a Target value in the Spec Limits column property, this parameter is set to the overall average of the Y column.

The Target value cannot be edited in these situations:

- If you specified the Center Data option in the launch window.
- If the *Y* variable has a Spec Limits column property that contains a value for Target. However, you can select the Use Overall Mean for Target option in the red triangle menu to switch between the Target value in the Spec Limits column property and the overall mean of the *Y* variable.

Sigma The known value of the standard deviation. By default, this parameter is set to the average moving range of the Y column. If there is a Subgroup variable, the Sigma parameter is set to the average of the moving ranges of the subgroup means.

Note: The default Sigma value calculation does not include excluded rows.

Lambda The value of the smoothing constant for weighting prior samples. By default, this parameter is set to 0.2.

***Using Centered Data** The presence of this text indicates that the Centered Data option was selected in the launch window and that the values in the EWMA chart and X chart are centered around the Target value.

EWMA Chart Report

The EWMA Chart report contains three charts: an EWMA chart, an X chart, and a residuals chart. For more information about the interpretation of these charts, see Box et al. (2009).

EWMA Chart

The EWMA chart is an exponentially weighted moving average (EWMA) chart with decision limits that are determined by the current values of the chart parameters:

- If you specified a Subgroup variable and at least one subgroup size is greater than 1, the horizontal axis denotes the subgroups.
- Otherwise, the horizontal axis denotes the samples.

In both cases, the vertical axis denotes the exponentially weighted moving average. If you specified the Center Data option in the launch window, the vertical axis denotes the exponentially weighted moving average with the target value subtracted from it. Each sample or subgroup has a single point on the chart. There is one additional point that is a forecast point, which is shown in blue.

Note: If the last sample (or subgroup) is both hidden and excluded, the line connecting the last sample (or subgroup) to the forecast point is not drawn.

X Chart

The X chart is a Shewhart control chart of the observations:

- If you specified a Subgroup variable and at least one subgroup size is greater than 1, the X chart is an XBar chart of the mean values. The horizontal axis denotes the subgroups. The vertical axis denotes the subgroup means. If you selected the Center Data option in the launch window, the vertical axis denotes the subgroup means with the target value subtracted from them. Each subgroup has a single point on the chart.
- Otherwise, the X chart is an individual measurements chart of the values. The horizontal axis denotes the samples. The vertical axis denotes the measurements. If you selected the Center Data option in the launch window, the vertical axis denotes the measurements with the target value subtracted from them. Each sample has a single point on the chart.

For more information about the limits on the X chart, see "Statistical Details for Control Chart Builder".

Residuals Chart

The residuals chart enables you to visually check for autocorrelation:

- If you specified a Subgroup variable and at least one subgroup size is greater than 1, the residuals chart is a chart of the differences between each subgroup mean and the EWMA value for the previous subgroup. The i^{th} residual is calculated as $r_i = X_i - EWMA_{i-1}$ where X_i denotes the i^{th} subgroup mean and $EWMA_{i-1}$ denotes the $(i-1)^{th}$ EWMA value.
- Otherwise, the residuals chart is a chart of the differences between each sample value and the EWMA value for the previous sample. The i^{th} residual is calculated as $r_i = X_i - EWMA_{i-1}$ where X_i denotes the i^{th} sample value and $EWMA_{i-1}$ denotes the $(i-1)^{th}$ EWMA value.

The limits on the residuals chart are ±3**ResidSigma*, where *ResidSigma* is the standard deviation of the residuals.

The EWMA Control Chart Platform Options

The EWMA Control Chart red triangle menu contains the following options:

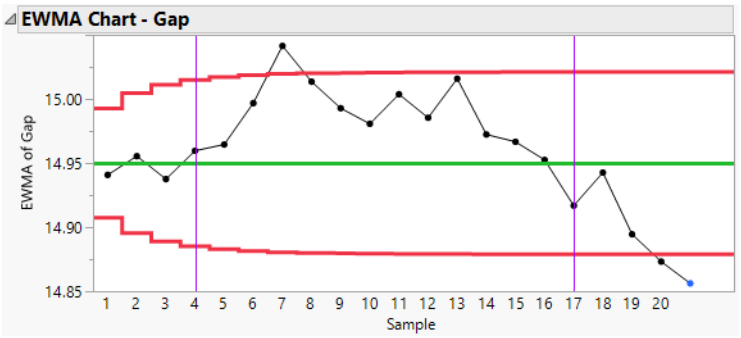
Show Limits Shows or hides the upper and lower decision limits in the EWMA chart, X chart, and residuals chart.

Show Center Line Shows or hides the center line in the EWMA chart, X chart, and residuals chart.

Show Shift Lines Shows or hides the vertical lines in the EWMA chart that designate shifts. Shift lines are drawn at the start of a shift. A shift start is defined as the first point after the EWMA value crosses the center line in a particular direction.

- A positive shift occurs when the EWMA value exceeds the upper limit on the chart. The start of the shift is defined as the first point after the most recent EWMA value below the Target line.
- A negative shift occurs when the EWMA value falls below the lower limit on the chart. The start of the shift is defined as the first point after the most recent EWMA value above the Target line.

Figure 9.5 Shift Lines Example



In this example, the point for sample 7 is above the upper limit. Looking back from sample 7, the most recent point below the Target line is sample 3. Therefore, sample 4 is the first point above the Target line since the most recent point below the Target line and is denoted as the start of the positive shift.

Similarly, the point for sample 20 is below the lower limit and the most recent point above the Target line is sample 16. Therefore, sample 17 is denoted as the start of the negative shift.

Test Beyond Limits Shows or hides a red circle around any point that is above the upper limit or below the lower limit in the EWMA and X charts.

Show ARL Shows or hides the Average Run Length (ARL) report. See "Average Run Length (ARL) Report for EWMA Control Charts".

Control Panel Shows or hides a report of the current values of the parameters. This report enables you to change the parameter values in the EWMA chart.

Parameters Report Shows or hides a report of the current values of the parameters.

Constant Limits Specifies that the EWMA chart limits are calculated using an asymptotic expression so that the limits on the EWMA chart are constant.

Caution: The Constant Limits option has no effect when the sample sizes are not equal across subgroups.

Save Summaries Creates a new data table that contains statistics for each subgroup in the EWMA chart. The following statistics are saved to the new data table: the subgroup number, the subgroup label, the subgroup size, the subgroup mean, an indicator of shift starts, a value that indicates each interval between shift starts, the exponentially weighted moving average of each subgroup, the number of positive and negative consecutive run counts, the LCL and UCL values, and test failure indicators. The forecast value is saved in the last row of the summary table.

Reset to Defaults Resets all parameters to their default values. If a Lambda value has been specified in the Lambda platform preference, the Lambda value is reset to the value that is specified in the platform preference.

Note: When the *Y* variable has a Spec Limits column property that contains a value for Target, the Reset to Defaults option sets the Target to the Target value in the Spec Limits column property.

Restart EWMA After Empty Subgroup Specifies how calculations for the moving average and limits are handled when there are empty subgroups. A subgroup can be empty if all the observations for the subgroup are missing values or are in excluded rows. If this option is selected, the calculations restart in the first nonmissing subgroup that follows an empty subgroup. The restart of the calculations resets the moving average to the overall mean. If this option is not selected, the EWMA calculations continue with the most recent nonmissing subgroup moving average.

Use Overall Mean for Target (Available only when the *Y* variable has a Spec Limits column property that contains a value for Target.) Sets the Target in the EWMA chart to the overall mean of the *Y* variable. If this option is not selected, the Target in the EWMA chart is set to the Target value in the Spec Limits column property.

Overlay Charts Specifies whether the individual location values are overlaid on the EWMA chart. When this option is selected, the Location chart is no longer shown. Instead, the

points from the Location chart appear on the EWMA chart as unconnected gray Xs. The limits from the Location chart do not appear on the EWMA chart, unless the Show X Limits on Overlay Charts option is selected.

Show X Limits on Overlay Charts (Available only when the Overlay Charts option is selected.) Shows or hides the limits from the Location chart on the EWMA chart when the Overlay Charts option is selected. When the Location chart limits are shown on the EWMA chart, they appear as dashed lines.

Note: The Test Beyond Limits option is applied to the Location chart values in the Overlay Chart only when the Show X Limits on Overlay Charts option is selected.

Lambda Slider Shows or hides a slider control that enables you to change the value of the Lambda parameter interactively.

Show X Chart Shows or hides the Location chart below the EWMA chart.

Note: When the Overlay Charts option is selected, the Show X Chart option shows or hides the individual location values and limits that are overlaid on the EWMA chart.

Show Residuals Chart Shows or hides a chart of residuals.

Alarm Script Enables you to write and run a script that indicates when the data fail special causes tests. Results can be written to a file, written to the log or sent in an email. There is an option to include an explanation of why the test failed.

As an Alarm Script is invoked, the following variables are available, both in the issued script and in subsequent JSL scripts:

`qc_col` is the name of the column

`qc_test` is the test that failed

`qc_sample` is the sample number

Tip: After an alarm script is specified, the alarm script is invoked when the Test Beyond Limits option is turned on.

See the *Scripting Guide* for more information about writing custom Alarm Scripts.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Average Run Length (ARL) Report for EWMA Control Charts

The Average Run Length (ARL) report contains a table and a graph of ARL values. The average run length (ARL) for a specified shift is the average number of runs expected before an out-of-control signal occurs. For example, the ARL at 0 represents the average number of runs expected before seeing a false-alarm signal when the process is in control. When the process is in control, the shift size is 0.

The table and graph in the ARL report enable you to explore how various settings of the Lambda parameter affect the performance of the corresponding EWMA chart. The table and graph also enable you to compare the performance of the EWMA chart with a Shewhart chart, such as the X chart in the EWMA Chart report. The Shewhart ARL column is equivalent to the EWMA ARL column when Lambda is set to 1.

The value of EWMA ARL at 0 depends on the setting of the Constant Limits option:

- If the Constant Limits option is selected, the process is assumed to have been in control long enough that the effect of the starting value is negligible. In this case, also referred to as a *steady state* of the EWMA chart, the value of EWMA ARL(0) is calculated using the method described in Crowder (1987).
- If the Constant Limits option is not selected, the value of EWMA ARL(0) is calculated using the method described in Knoth (2004). This situation is also referred to as a *zero state* of the EWMA chart.

ARL Report

The ARL Report shows the average run length for standardized shifts (Δ) between 0 and 3 at 0.25 increments. The shift is represented by sqrt(n)*RawShift/Sigma, where n is the constant sample size in each subgroup. For the multiplier used in calculating control limits, K = 3 is assumed. This table contains ARL values for the EWMA chart as well as a Shewhart chart.

ARL Graph

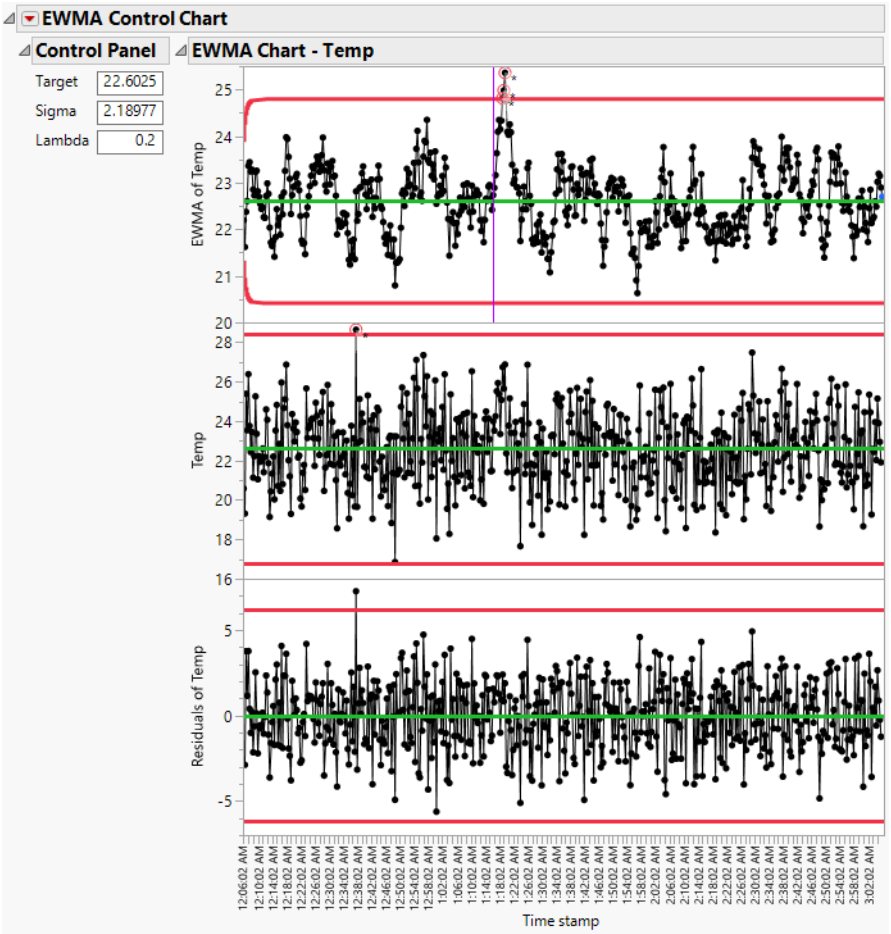
The ARL Graph shows the average run length for standardized shifts (Δ) between 0 and 3. This graph contains the same data points as the ARL Table to the left of the ARL Graph. The solid line corresponds to the EWMA ARL values, and the dashed line corresponds to the Shewhart ARL values.

Additional Example of the EWMA Control Chart Platform

In this example, you create a chart that overlays the subgroup points and limits on the EWMA control chart. This enables you to see the EWMA points in the context of the subgroup points. The Quench.jmp sample data table contains simulated observations of temperature at the time of magnet quench. Magnet quench occurs when part of the superconducting coil enters a resistive state and increases in temperature. When this happens, containment of the fusion reactor is at risk. Immediately shutting down power to the magnet is required to prevent a breach.

1. Select **Help > Sample Data Library** and open Quality Control/Quench.jmp.
2. Select **Analyze > Quality and Process > Control Chart > EWMA Control Chart**.
3. Select Temp and click **Y**.
4. Select Time stamp and click **Subgroup**.
5. Click **OK**.
6. Click the EWMA Control Chart red triangle and select **Test Beyond Limits**.

Figure 9.6 EWMA Chart with Points Beyond Limits Highlighted



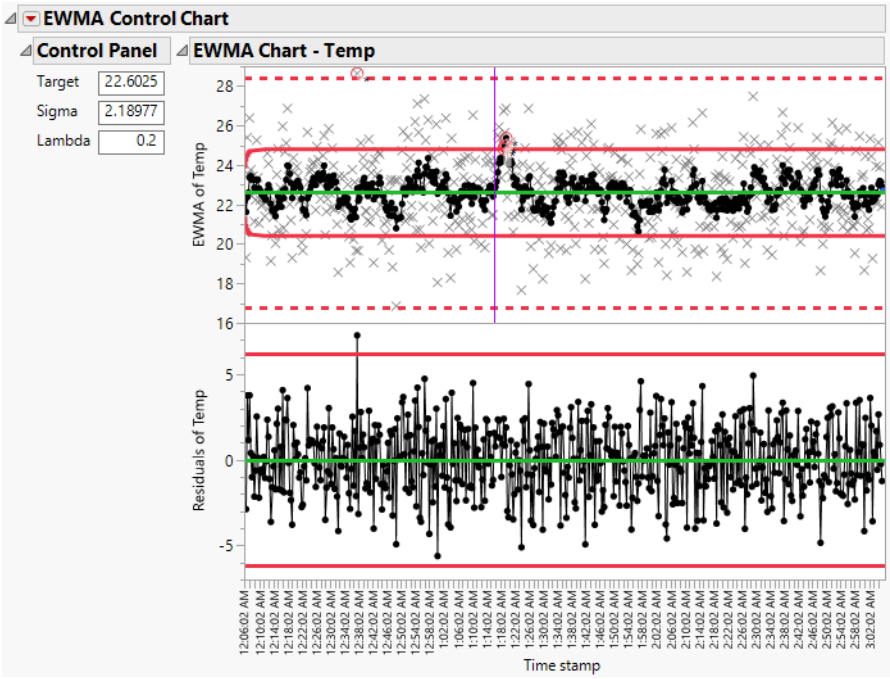
Note that three observations are outside the limits in the EWMA chart and one observation is outside the limits in the X chart. The observations outside of the limits are indicated by a red circle. The purple vertical line in the EWMA chart denotes a shift upward around 1:16 AM.

7. Click the EWMA Control Chart red triangle and select **Overlay Charts**.

   The X chart limits are not shown on the overlay chart by default, so the observation outside the X limits is not circled until you turn on the X limits for the overlay chart.

8. Click the EWMA Control Chart red triangle and select **Show X Limits on Overlay Charts**.

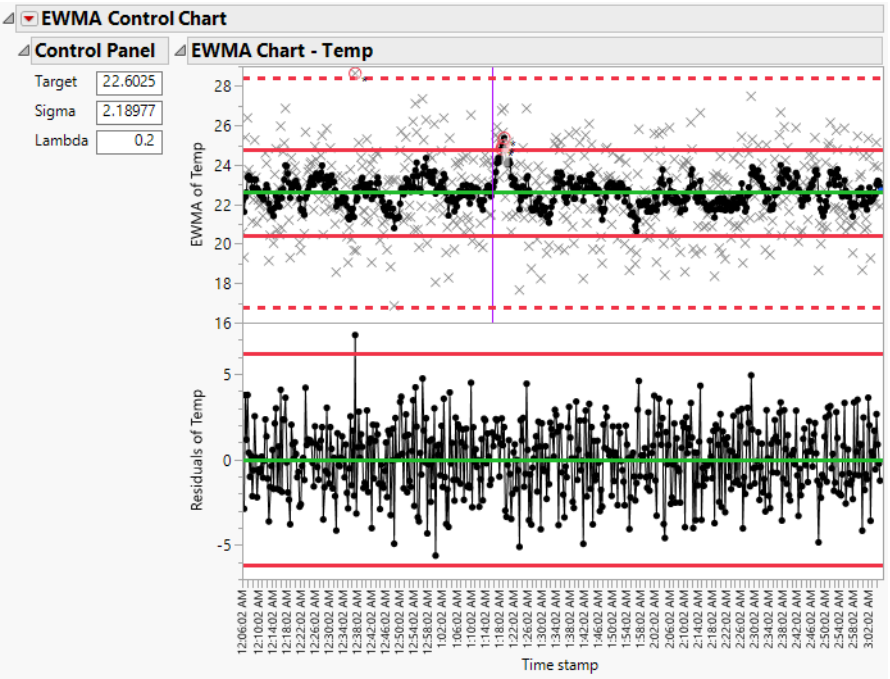
Figure 9.7 EWMA Chart with Overlaid X Chart



The solid lines and circles in the top chart represent the limits and points for the EWMA chart. The dashed lines and x markers represent the limits and points for the X chart. Note that the observations outside the EWMA limits and the X limits are still circled.

9. Click the EWMA Control Chart red triangle and select **Constant Limits**.

Figure 9.8 EWMA Chart with Constant Limits



Because the time span of the data is so long, the asymptotic limits differ from the original EWMA limits only for the first few subgroups. In this case, you might decide that you can use constant (asymptotic) limits instead of the EWMA limits.

Statistical Details for the EWMA Control Chart Platform

This section defines the statistics that are used in the construction of the EWMA chart. Some of these statistics are also saved in the data table that is created by the Save Summaries command.

The i^{th} point on the EWMA chart is calculated as follows:

$$EWMA_i = \lambda X_i + (1 - \lambda) EWMA_{i-1}$$

where:

λ = Lambda parameter

$X_i = i^{th}$ sample value (or subgroup mean)

$EWMA_{i\text{-}1}$ = $(i\text{-}1)^{\text{th}}$ EWMA value

When $i = 1$, define $EWMA_0$ as the Target value.

Note: When the Restart EWMA after Empty Subgroup option is selected, the $EWMA_{i\text{-}1}$ value following an empty subgroup is the Target value. When the Restart EWMA after Empty Subgroups option is not selected, the $EWMA_{i\text{-}1}$ value following an empty subgroup is the *EWMA* value for the most recent non-empty subgroup.

The computation of the control limits on the EWMA chart is determined by the setting of the Constant Limits option.

EWMA Limits

When the Constant Limits option is not selected and the subgroup sizes are not equal, the EWMA control limits are computed as follows:

$$\text{LCL} = T - K\sigma\lambda\sqrt{\sum_{j=1}^{i}\frac{(1-\lambda)^{2(i-j)}}{n_j}}$$

$$\text{UCL} = T + K\sigma\lambda\sqrt{\sum_{j=1}^{i}\frac{(1-\lambda)^{2(i-j)}}{n_j}}$$

where:

T = Target value

K = the sigma multiplier and is set to 3 by default

σ = Sigma value

i = the number of the sample (or subgroup)

n_i = the size of subgroup i

When the Constant Limits option is not selected and the subgroup sizes are equal, the formulas for the EWMA control limits simplify as follows:

$$\text{LCL} = T - K\sigma\sqrt{\frac{\lambda}{n(2-\lambda)}[1-(1-\lambda)^{2i}]}$$

$$\text{UCL} = T + K\sigma\sqrt{\frac{\lambda}{n(2-\lambda)}[1-(1-\lambda)^{2i}]}$$

where:

T = Target value

K = the sigma multiplier and is set to 3 by default

σ = Sigma value

i = the number of the sample (or subgroup)

n = the size of each subgroup (or 1 if no subgroup is specified)

Constant Limits

When the Constant Limits option is selected, the EWMA control limits are computed as follows:

$$\text{LCL} = T - K\sigma\sqrt{\frac{\lambda}{n(2-\lambda)}}$$

$$\text{UCL} = T + K\sigma\sqrt{\frac{\lambda}{n(2-\lambda)}}$$

where:

T = Target value

K = the sigma multiplier and is set to 3 by default

σ = Sigma value

n = the subgroup size (or 1 if no subgroup is specified)

For more information about constructing exponentially weighted moving average charts, see Montgomery (2013).

Chapter 10

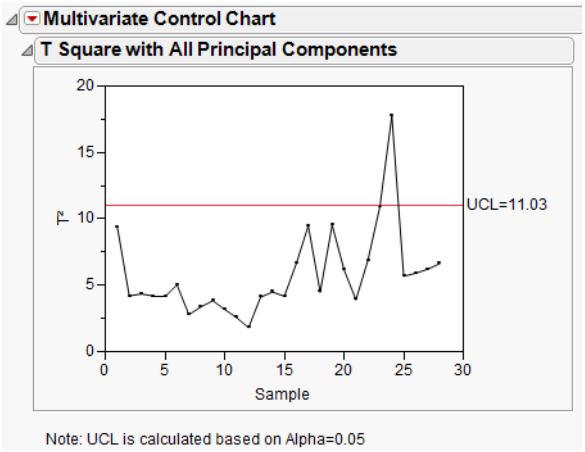
Multivariate Control Charts

Monitor Multiple Process Characteristics Simultaneously

Multivariate control charts are used to monitor two or more interrelated process variables. Where univariate control charts are used to monitor a single independent process characteristic, multivariate control charts are necessary when process variables are correlated. The Multivariate Control Chart platform enables you to build Hotelling T^2 charts. You can use the platform to determine whether a process is stable as well as to monitor a process as new data are collected.

For monitoring and diagnosing complex processes, see "Model Driven Multivariate Control Charts".

Figure 10.1 Example of a Multivariate Control Chart



Contents

Overview of Multivariate Control Charts 281
Example of a Multivariate Control Chart 281
Launch the Multivariate Control Chart Platform 284
The Multivariate Control Chart 285
Multivariate Control Chart Platform Options 286
T Square Partitioned 287
Change Point Detection 288
Principal Components 289
Additional Examples of Multivariate Control Charts 289
Example of Monitoring a Process Using Sub-Grouped Data 289
Example of T Square Partitioned 292
Example of Change Point Detection 294
Statistical Details for Multivariate Control Charts 295
Statistical Details for Individual Observations 296
Statistical Details for Observations in Rational Subgroups 297
Statistical Details for Change Point Detection 300

Overview of Multivariate Control Charts

Multivariate control charts are used to monitor two or more interrelated process variables. Where univariate control charts are used to monitor a single independent process characteristic, multivariate control charts are necessary when process variables are correlated. A Hotelling T^2 chart, or just T^2 chart for short, is one type of multivariate control chart. A T^2 chart can detect shifts in the mean or the relationship between several interrelated variables. The observations can either be individual observations of the process variables or they can be grouped into rational subgroups.

You can construct a multivariate control chart using current or historical data. The control chart is said to be a Phase I chart if it is constructed using current data; the control chart is said to be a Phase II chart if it is constructed using target statistics from a historical data set. In Phase I, you check that the process is stable and establish a historical data set from which to calculate target statistics for the process. In Phase II, the multivariate control chart uses the target statistics from Phase I in order to monitor new process observations.

To construct a Phase II multivariate control chart, first identify a period of time during which the process is stable and capable.

1. Develop a Phase I control chart to verify that the process is stable over this period.

   The data used in Phase I provides a historical data set.
2. Save the target statistics for this historical data set.
3. Monitor the on-going process using a Phase II control chart based on the target statistics that were saved in step 2.

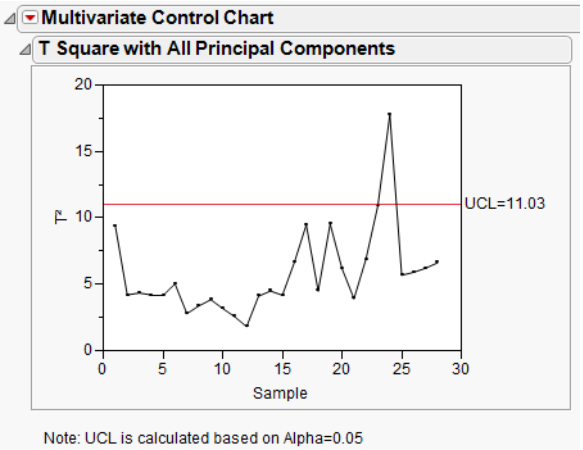
Example of a Multivariate Control Chart

This example illustrates constructing a control chart for data that are not sub-grouped. The data are measurements on a steam turbine engine. For an example that uses sub-grouped data, "Example of Monitoring a Process Using Sub-Grouped Data".

Step 1: Determine Whether the Process Is Stable

1. Select **Help > Sample Data Library** and open Quality Control/Steam Turbine Historical.jmp.
2. Select **Analyze > Quality and Process > Control Chart > Multivariate Control Chart**.
3. Select all of the columns and click **Y, Columns**.
4. Click **OK**.

Figure 10.2 Initial Multivariate Control Chart



The process seems to be in reasonable statistical control, because there is only one out-of-control point. Therefore, it is appropriate to create targets based on this data.

Step 2: Save Target Statistics

1. Click the red triangle next to Multivariate Control Chart and select **Save Target Statistics**.

   This creates a new data table containing target statistics for the process.

Figure 10.3 Target Statistics for Steam Turbine Data

	Ref_Stats	Fuel	Steam Flow	Steam Temp	MW	Cool Temp	Pressure
1	_SampleSize	28	28	28	28	28	28
2	_NumSample	1	1	1	1	1	1
3	_Mean	237595.78571	179015.78571	846.39285714	20.647142857	53.871428571	29.139285714
4	_Std	7247.6859825	4374.3063819	2.9481857034	0.5341650261	0.2088010623	0.0497347461
5	_Corr_Fuel	1	0.8714382899	-0.549875041	0.8558570808	-0.270049819	-0.469928462
6	_Corr_Steam Flow	0.8714382899	1	-0.629023927	0.9852529223	-0.223127002	-0.533056185
7	_Corr_Steam Temp	-0.549875041	-0.629023927	1	-0.595214609	0.2475387217	0.2192147319
8	_Corr_MW	0.8558570808	0.9852529223	-0.595214609	1	-0.207305813	-0.50447312
9	_Corr_Cool Temp	-0.270049819	-0.223127002	0.2475387217	-0.207305813	1	0.3617461646
10	_Corr_Pressure	-0.469928462	-0.533056185	0.2192147319	-0.50447312	0.3617461646	1

2. Save the new data table as Steam Turbine Targets.jmp.

Now that target statistics have been established, create the multivariate control chart that monitors the process.

Step 3: Monitor the Process

1. Select **Help > Sample Data Library** and open Quality Control/Steam Turbine Current.jmp.

This sample data table contains recent observations from the process.

2. Select **Analyze > Quality and Process > Control Chart > Multivariate Control Chart**.
3. Select all of the columns and click **Y, Columns**.
4. Click **Get Targets**.
5. Open the Steam Turbine Targets.jmp table that you saved.
6. Click **OK**.

The default alpha level is set to 0.05. Change it to 0.001.

7. Click the red triangle next to Multivariate Control Chart and select **Set Alpha Level > Other**.
8. Type 0.001 and click **OK**.

Figure 10.4 Steam Turbine Control Chart

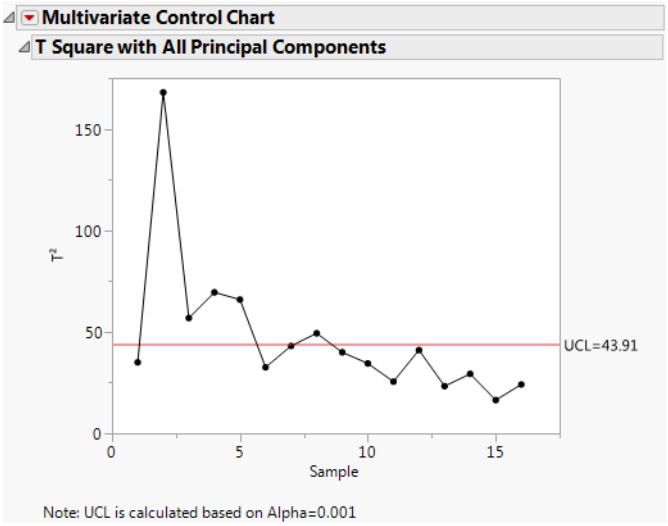
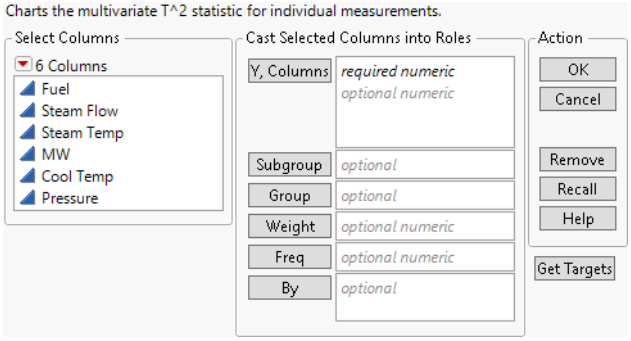


Figure 10.4 shows out-of-control conditions occurring at observations 2, 3, 4, 5, and 8. This result implies that these observations do not conform to the historical data from Steam Turbine Historical.jmp, and that the process should be further investigated. To find an assignable cause, you might want to examine individual univariate control charts or perform another univariate procedure.

Launch the Multivariate Control Chart Platform

Launch the Multivariate Control Chart platform by selecting **Analyze > Quality And Process > Control Chart > Multivariate Control Chart**.

Figure 10.5 The Multivariate Control Chart Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Multivariate Control Chart platform launch window contains the following options:

Y, Columns Specify the columns to be analyzed.

Subgroup Enter a column with sub-grouped data. Hierarchically, this group is nested within **Group**.

Group Enter a column that specifies group membership at the highest hierarchical level.

Weight Identifies the data table column whose variables assign weight (such as importance or influence) to the data.

Freq Identifies the data table column whose values assign a frequency to each row. Can be useful when your data table contains summarized data.

By Identifies a column that creates a report consisting of separate analyses for each level of the variable.

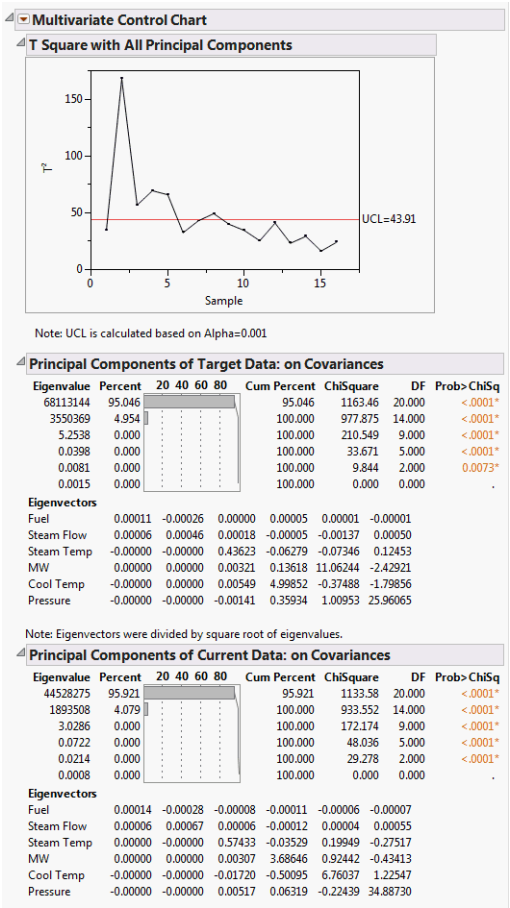
Get Targets Click to select a JMP table that contains historical targets for the process.

The Multivariate Control Chart

Use the multivariate control chart to quickly identify shifts in your process and to monitor your process for special cause indications.

Follow the instructions in "Example of a Multivariate Control Chart" to produce the results shown in Figure 10.6.

Figure 10.6 Multivariate Control Chart



Tip: For information about additional options, see "Multivariate Control Chart Platform Options".

The multivariate control chart plots Hotelling's T^2 statistic. The calculation for the control limit differs based on whether targets have been specified. To understand how the T^2 statistic and the UCL (Upper Control Limit) are calculated, see "Statistical Details for Multivariate Control Charts". For more information about control limits, see Tracy et al. (1992).

In this example, the Principal Components reports for both data sets indicate that the first eigenvalue, corresponding to the first principal component, explains about 95% of the total variation in the variables. The values in both Eigenvectors tables indicate that the first principal component is driven primarily by the variables Fuel and Steam Flow. You can use this information to construct a potentially more sensitive control chart based only on this first component. For more information about the Principal Components reports, see "Principal Components".

Multivariate Control Chart Platform Options

The Multivariate Control Chart red triangle menu contains the following options:

T Square Chart Shows the T^2 chart. Hotelling's T^2 chart is a multivariate extension of the *X*Bar chart that takes correlation into account.

T Square Partitioned Constructs multivariate control charts based on the principal components of Y. Specify the number of major principal components for T^2. See "T Square Partitioned".

Set Alpha Level Set the α level used to calculate the control limit. The default is α=0.05.

Show Covariance Shows the Covariance report. Covariance is a measure of the linear relationship between two variables. When a subgroup variable is specified, the Pooled Covariance report is shown.

Show Correlation Shows the Correlation report. When a subgroup variable is specified, the Pooled Correlation report is shown.

Show Inverse Covariance Shows the Inverse Covariance report. If the inverse covariance is singular, a generalized inverse of the covariance matrix is reported. When a subgroup variable is specified, the Pooled Inverse Covariance report is shown.

Show Inverse Correlation Shows the Inverse Correlation report. If the inverse correlation is singular, a generalized inverse of the correlation matrix is reported. When a subgroup variable is specified, the Pooled Inverse Correlation report is shown.

Show Means Shows the Group Means report, which contains the means for each group.

Save T Square Creates a new column in the data table containing T^2 values.

Save T Square Formula Creates a new column in the data table. Stores a formula in the column that calculates the T^2 values.

Save Target Statistics Creates a new data table containing target statistics for the process. Target statistics include: sample size, the number of samples, mean, standard deviation, and any correlations.

Change Point Detection (Not available for sub-grouped data.) Shows a Change Point Detection plot of test statistics by row number and indicates the row number where the change point appears. See "Change Point Detection".

Principal Components Shows reports showing eigenvalues and their corresponding eigenvectors. Principal components help you understand which of the many variables you might be monitoring are primarily responsible for the variation in your process. See "Principal Components".

Save Principal Components Creates new columns in the data table that contain the scaled principal components.

T Square Partitioned

If you are monitoring a large number of correlated process characteristics, you can use the T Square Partitioned option to construct a control chart based on principal components. If a small number of principal components explains a large portion of the variation in your measurements, then a multivariate control chart based on these big components might be more sensitive than a chart based on your original, higher-dimensional data.

The T Square Partitioned option is also useful when your covariance matrix is ill-conditioned. When this is the case, components with small eigenvalues, explaining very little variation, can have a large, and misleading, impact on T^2. It is useful to separate out these less important components when studying process behavior.

Once you select the T Square Partitioned option, you need to decide how many major principal components to use.

The option creates two multivariate control charts: T Square with Big Principal Components and T Square with Small Principal Components. Suppose that you enter *r* as the number of major components when you first select the option. The chart with Big Principal Components is based on the *r* principal components corresponding to the *r* largest eigenvalues. These are the *r* components that explain the largest amount of variation, as shown in the Percent and Cum Percent columns in the Principal Components: on Covariances reports. The chart with Small Principal Components is based on the remaining principal components.

For a given subgroup, its T^2 value in the Big Principal Components chart and its T^2 value in the Small Principal Components chart sum to its overall T^2 statistic presented in the T^2 with All Principal Components report. For more information about how the partitioned T^2 values are calculated, see Kourti and MacGregor (1996).

Tip: Consider using the Model Driven Multivariate Control Chart platform for decomposition of the T^2 statistic. See "Model Driven Multivariate Control Charts".

Change Point Detection

When the data set consists of multivariate individual observations, a control chart can be developed to detect a shift in the mean vector, the covariance matrix, or both. This method partitions the data and calculates likelihood ratio test statistics for a shift. The statistic that is plotted on the control chart is an observation's likelihood ratio test statistic divided by the product of the following:

- Its approximate expected value assuming no shift.
- An approximate value for an upper control limit.

Division by the approximate upper control limit allows the points to be plotted against an effective upper control limit of 1. A Change Point Detection plot readily shows the change point for a shift occurring at the maximized value of the control chart statistic. The Change Point Detection implementation in JMP is based on Sullivan and Woodall (2000) and is described in "Statistical Details for Change Point Detection".

Note: The Change Point Detection method is designed to show a single shift in the data. Detect multiple shifts by recursive application of this method.

Note the following about the Change Point Detection plot:

- Values above 1.0 indicate a possible shift in the data.
- Control chart statistics for the Change Point Detection plot are obtained by dividing the likelihood ratio statistic of interest (either a mean vector or a covariance matrix) by a normalizing factor.
- The change point of the data occurs for the observation having the maximum test statistic value for the Change Point Detection plot.

Note the following about the scatterplot matrix:

- This plot shows the shift in the sample mean vector.
- In the "Example of Change Point Detection", data are divided into two groups. The first 24 observations are classified as the first group. The remaining observations are classified as the second group.

Principal Components

The Principal Components report contain the following information:

Eigenvalue Eigenvalues for the covariance matrix.

Percent Percent variation explained by the corresponding eigenvector. Also shows an accompanying bar chart.

Cum Percent Cumulative percent variation explained by eigenvectors corresponding to the eigenvalues.

ChiSquare Provides a test of whether the correlation remaining in the data is of a random nature. This is a Bartlett test of sphericity. When this test rejects the null hypothesis, this implies that there is structure remaining in the data that is associated with this eigenvalue.

DF Degrees of freedom associated with the Chi-square test.

Prob > ChiSq *p*-value for the test.

Eigenvectors Table of eigenvectors corresponding to the eigenvalues. Note that each eigenvector is divided by the square root of its corresponding eigenvalue.

For more information about principal components, see *Multivariate Methods*.

Additional Examples of Multivariate Control Charts

- "Example of Monitoring a Process Using Sub-Grouped Data"
- "Example of T Square Partitioned"
- "Example of Change Point Detection"

Example of Monitoring a Process Using Sub-Grouped Data

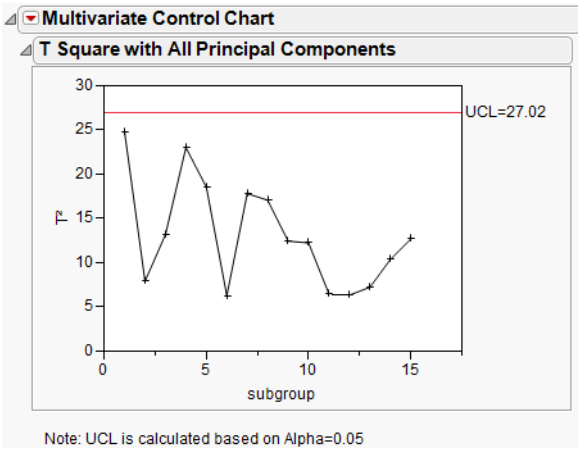
The workflow for monitoring a multivariate process with sub-grouped data is similar to the one for individual data. See "Example of a Multivariate Control Chart". You create an initial control chart to save target statistics and then use these targets to monitor the process.

Step 1: Determine Whether the Process Is Stable

1. Select **Help > Sample Data Library** and open Quality Control/Aluminum Pins Historical.jmp.
2. Select **Analyze > Quality and Process > Control Chart > Multivariate Control Chart**.
3. Select all of the Diameter and Length columns and click **Y, Columns**.

4. Select subgroup and click **Subgroup**.
5. Click **OK**.

Figure 10.7 Multivariate Control Chart for Sub-Grouped Data, Step 1



The process appears to be in statistical control, making it appropriate to create targets using this data.

Step 2: Save Target Statistics

1. Click the red triangle next to Multivariate Control Chart and select **Save Target Statistics.**

   This creates a new data table containing target statistics for the process.
2. Save the new data table as Aluminum Pins Targets.jmp.

   Now that target statistics have been established, create the multivariate control chart for process monitoring.

Step 3: Monitor the Process

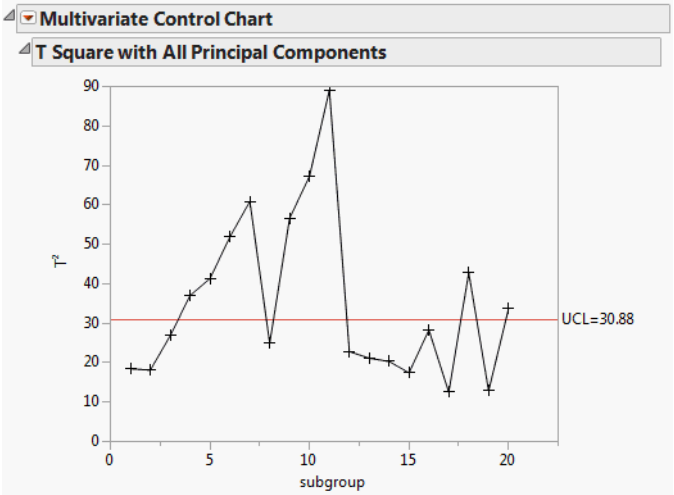
1. Select **Help > Sample Data Library** and open Quality Control/Aluminum Pins Current.jmp.

   This sample data table contains recent observations from the process.
2. Select **Analyze > Quality and Process > Control Chart > Multivariate Control Chart**.
3. Select all of the Diameter and Length columns and click **Y, Columns**.
4. Select subgroup and click **Subgroup**.
5. Click **Get Targets**.
6. Open the Aluminum Pins Targets.jmp table that you saved.
7. Click **OK**.

8. Click the red triangle next to Multivariate Control Chart and select **Show Means**.

   The Show Means option gives the means for each subgroup. You can then observe which groups are most dissimilar from each other.

Figure 10.8 Multivariate Control Chart for Sub-Grouped Data, Step 3



Principal Components of Target Data: on Covariances

Eigenvalue	Percent	20 40 60 80	Cum Percent	ChiSquare	DF	Prob>ChiSq
0.0020	77.438		77.438	194.981	20.000	<.0001*
0.0004	14.112		91.549	95.046	14.000	<.0001*
0.0002	6.543		98.092	57.140	9.000	<.0001*
0.0000	1.110		99.201	7.721	5.000	0.1723
0.0000	0.442		99.643	0.227	2.000	0.8925
0.0000	0.357		100.000	0.000	0.000	.

Eigenvectors

Diameter1	-1.74765	25.69209	11.26651	-2.09837	-141.425	234.5830
Diameter2	-1.94301	21.04343	4.33901	83.66331	225.8678	72.80281
Diameter3	-2.05350	24.72990	1.72384	96.30263	-115.819	-195.706
Diameter4	-1.81865	30.15028	4.14986	-136.824	58.89822	-97.0216
Length1	17.11878	2.03343	49.34954	4.23294	6.82763	-19.3736
Length2	13.94341	11.23070	-57.7762	2.53526	-4.00887	21.85602

Note: Eigenvectors were divided by square root of eigenvalues.

Principal Components of Current Data: on Covariances

Group Means

subgroup	Count	Diameter1	Diameter2	Diameter3	Diameter4	Length1	Length2
16	2	10.0000	9.9900	9.9900	14.9950	49.9250	60.0300
17	2	10.0000	9.9950	9.9900	14.9900	49.9150	60.0200
18	2	10.0000	9.9950	9.9900	14.9950	49.9300	60.0250
19	2	10.0000	9.9950	9.9900	14.9950	49.9100	59.9950
20	2	10.0000	10.0000	9.9900	14.9950	49.9000	59.9900

Figure 10.8 shows indications of instability at subgroups 4-7, 9-11, 18, and 20. This result implies that these observations do not conform to the historical data from Aluminum Pins Historical.jmp, and that the process should be further investigated. To determine why the process was out of control at these points, you might want to examine individual univariate control charts or perform another univariate procedure.

An alternative method to monitoring this process is based on the big principal components. In this example, for the historical data, the first three principal components account for about 98% of the variation. Based on this, you might construct a chart for the first three principal components. Then you would monitor current data using those three components. The control limits for the chart used in monitoring the process should be based on the corresponding chart for the historical data.

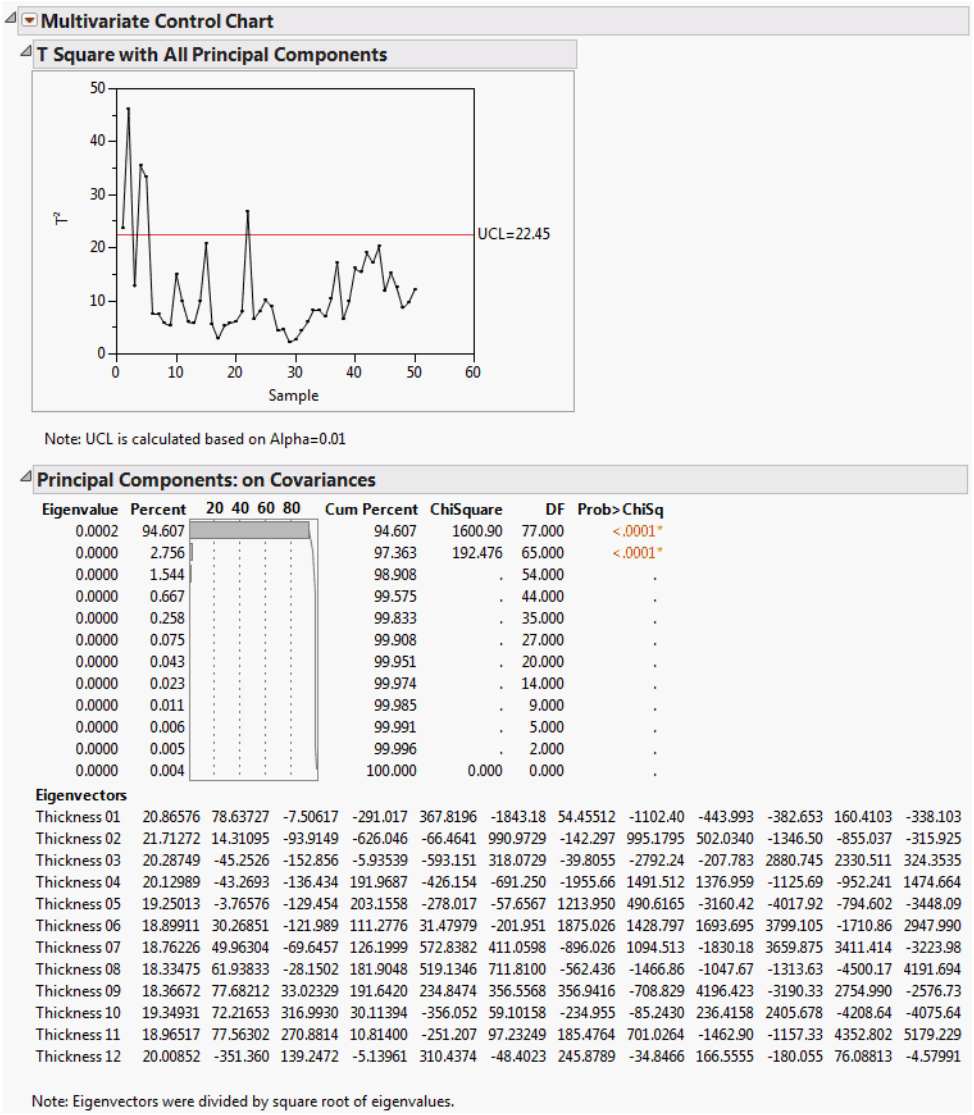
Example of T Square Partitioned

Use T Square Partitioned to separate out the more important components from the less important components when studying process behavior. In this example, the coating on each of 50 bars was measured at 12 uniformly spaced locations across the bar. You want to examine the variation in the measurements and determine whether the causes of variation need to be investigated further.

1. Select **Help > Sample Data Library** and open Quality Control/Thickness.jmp.
2. Select **Analyze > Quality and Process > Control Chart > Multivariate Control Chart**.
3. Select all of the Thickness columns and click **Y, Columns**.
4. Click **OK**.

   The current alpha level is set to 0.05, which corresponds to a 5% false alarm rate. You want to set the false alarm rate to 1%.
5. To change the alpha level, click the red triangle next to Multivariate Control Chart, select **Set Alpha Level**, and choose **0.01**.

Figure 10.9 Initial Multivariate Control Chart for Thickness.jmp



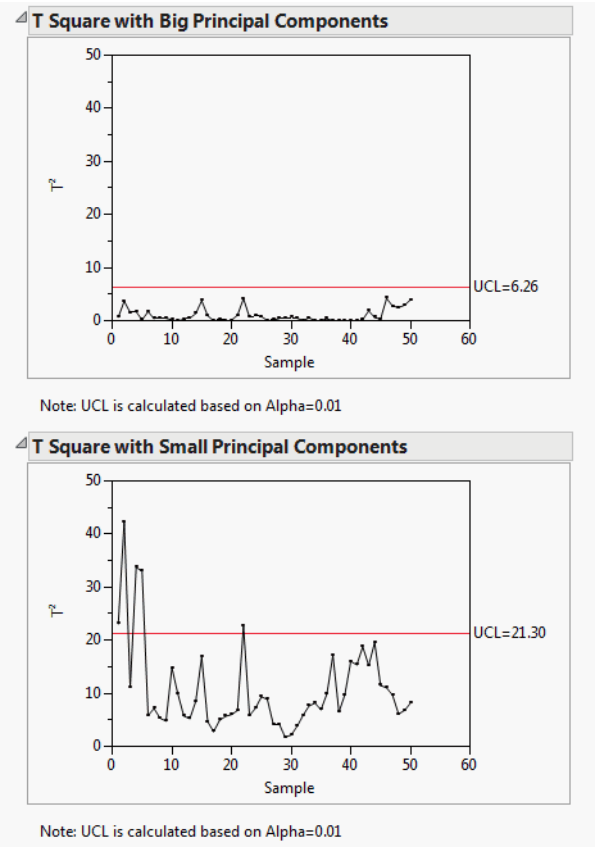
Eigenvalue	Percent	Cum Percent	ChiSquare	DF	Prob>ChiSq
0.0002	94.607	94.607	1600.90	77.000	<.0001*
0.0000	2.756	97.363	192.476	65.000	<.0001*
0.0000	1.544	98.908	.	54.000	.
0.0000	0.667	99.575	.	44.000	.
0.0000	0.258	99.833	.	35.000	.
0.0000	0.075	99.908	.	27.000	.
0.0000	0.043	99.951	.	20.000	.
0.0000	0.023	99.974	.	14.000	.
0.0000	0.011	99.985	.	9.000	.
0.0000	0.006	99.991	.	5.000	.
0.0000	0.005	99.996	.	2.000	.
0.0000	0.004	100.000	0.000	0.000	.

Eigenvectors												
Thickness 01	20.86576	78.63727	-7.50617	-291.017	367.8196	-1843.18	54.45512	-1102.40	-443.993	-382.653	160.4103	-338.103
Thickness 02	21.71272	14.31095	-93.9149	-626.046	-66.4641	990.9729	-142.297	995.1795	502.0340	-1346.50	-855.037	-315.925
Thickness 03	20.28749	-45.2526	-152.856	-5.93539	-593.151	318.0729	-39.8055	-2792.24	-207.783	2880.745	2330.511	324.3535
Thickness 04	20.12989	-43.2693	-136.434	191.9687	-426.154	-691.250	-1955.66	1491.512	1376.959	-1125.69	-952.241	1474.664
Thickness 05	19.25013	-3.76576	-129.454	203.1558	-278.017	-57.6567	1213.950	490.6165	-3160.42	-4017.92	-794.602	-3448.09
Thickness 06	18.89911	30.26851	-121.989	111.2776	31.47979	-201.951	1875.026	1428.797	1693.695	3799.105	-1710.86	2947.990
Thickness 07	18.76226	49.96304	-69.6457	126.1999	572.8382	411.0598	-896.026	1094.513	-1830.18	3659.875	3411.414	-3223.98
Thickness 08	18.33475	61.93833	-28.1502	181.9048	519.1346	711.8100	-562.436	-1466.86	-1047.67	-1313.63	-4500.17	4191.694
Thickness 09	18.36672	77.68212	33.02329	191.6420	234.8474	356.5568	356.9416	-708.829	4196.423	-3190.33	2754.990	-2576.73
Thickness 10	19.34931	72.21653	316.9930	30.11394	-356.052	59.10158	-234.955	-85.2430	236.4158	2405.678	-4208.64	-4075.64
Thickness 11	18.96517	77.56302	270.8814	10.81400	-251.207	97.23249	185.4764	701.0264	-1462.90	-1157.33	4352.802	5179.229
Thickness 12	20.00852	-351.360	139.2472	-5.13961	310.4374	-48.4023	245.8789	-34.8466	166.5555	-180.055	76.08813	-4.57991

The overall control chart in Figure 10.9 suggests that special causes affected bars 1, 2, 4, 5, and 22. Looking at the Principal Components report, you can see that almost 95% of the variation in the 12 thickness measurements is explained by the first principal component. You want to study the variation associated with this principal component further.

6. Click the red triangle Multivariate Control Chart and select **T Square Partitioned**.
7. Accept the default value of 1 principal component by clicking **OK**.

Figure 10.10 T Square Partitioned Control Charts



In contrast to the Principal Components report, the T Square with Big Principal Components chart, which reflects variation for only the first component, shows no evidence of special causes. The T Square with Small Principal Components chart shows that the special cause indications reside in the remaining smaller components. These smaller components do not explain much variation, and likely represent random noise. Therefore, you might conclude that the variation in the thickness measurements is not a major cause for concern.

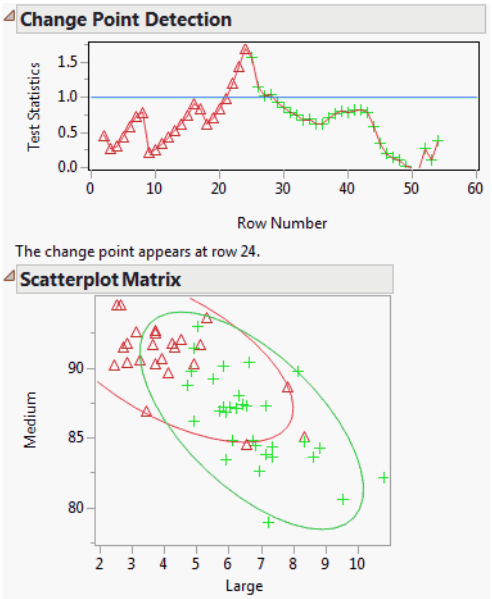
Example of Change Point Detection

Use change point detection to find the point at which a shift occurs in your data.

1. Select **Help > Sample Data Library** and open Quality Control/Gravel.jmp.
2. Select **Analyze > Quality and Process > Control Chart > Multivariate Control Chart**.
3. Select Large and Medium and click **Y, Columns**.
4. Click **OK**.

5. Click the red triangle next to Multivariate Control Chart and select **Change Point Detection**.

Figure 10.11 Change Point Detection for Gravel.jmp



Tip: You might need to drag the axes to see the density ellipses for the two groups, depending on your data.

In the Change Point Detection plot, values above 1.0 indicate a possible shift in the data. At least one shift is apparent; the change point occurs at observation 24 and the shift occurs immediately after observation 24. The 95% prediction regions for the two groups have approximately the same size, shape, and orientation, visually indicating that the sample covariance matrices are similar.

Statistical Details for Multivariate Control Charts

- "Statistical Details for Individual Observations"
- "Statistical Details for Observations in Rational Subgroups"
- "Statistical Details for Change Point Detection"

Statistical Details for Individual Observations

Consider measurements that are not sub-grouped, that is, where the natural subgroup size is $n = 1$. Denote the number of observations by m and the number of variables measured by p. A T^2 statistic is calculated and plotted for each observation. The calculation of the T^2 statistic and upper control limit (UCL) depends on the source of the target statistics. In a Phase I chart, the limits are based on the same data that is being plotted on the control chart. In a Phase II chart, the limits are based on target statistics that were calculated from a historical data set. For more information about T^2 statistic and control limit calculations for Hotelling T^2 control charts, see Montgomery (2013).

Calculations for Phase I Control Charts

In Phase I control charts, the T^2 statistic for the i^{th} observation is defined as follows:

$$T_i^2 = (Y_i - \bar{Y})' S^{-1} (Y_i - \bar{Y})$$

where:

Y_i is the column vector of p measurements for the i^{th} observation

$\bar{Y}$ is the column vector of sample means of the p variables

S^{-1} is the inverse of the sample covariance matrix

The T_i^2 value for each of the i observations are the points plotted on the multivariate control chart.

When computing Phase I control limits, the UCL is based on the beta distribution. Specifically, the upper control limit (UCL) is defined as follows:

$$UCL = \frac{(m-1)^2}{m} \beta_{\left[1-\alpha;\frac{p}{2};\frac{m-p-1}{2}\right]}$$

where:

p = number of variables

m = number of observations

$\beta_{\left[1-\alpha;\frac{p}{2};\frac{m-p-1}{2}\right]}$ = $(1-\alpha)^{th}$ quantile of a Beta $\left(\frac{p}{2}, \frac{m-p-1}{2}\right)$ distribution

Calculations for Phase II Control Charts

In Phase II control charts, define the historical data set as X. Then the T^2 statistic for the i^{th} observation is defined as follows:

$$T_i^2 = (Y_i - \overline{X})' S_X^{-1} (Y_i - \overline{X})$$

where:

Y_i is the column vector of p measurements for the i^{th} observation

$\overline{X}$ is the column vector of sample means of the p variables, calculated from the historical data set

S_X^{-1} is the inverse of the sample covariance matrix, calculated from the historical data set

The T_i^2 value for each of the i observations are the points plotted on the multivariate control chart.

When computing Phase II control limits, new observations are independent of the historical data set. In this case, the upper control limit (UCL) is a function of the F distribution and partially depends on the number of observations in the historical data set from which the targets are calculated. The UCL is defined as follows:

$$UCL = \begin{cases} \dfrac{p(m+1)(m-1)}{m(m-p)} F_{[1-\alpha, p, m-p]} & \text{if } m \le 100 \\ \dfrac{p(m-1)}{m-p} F_{[1-\alpha, p, m-p]} & \text{if } m > 100 \end{cases}$$

where:

p = number of variables

m = number of observations in the historical data set

$F_{[1-\alpha, p, m-p]}$ = $(1-\alpha)^{th}$ quantile of an $F_{(p, m-p)}$ distribution

Statistical Details for Observations in Rational Subgroups

Consider the case where p variables are monitored and m subgroups of size $n > 1$ are obtained. A T^2 statistic is calculated and plotted for each subgroup. The calculation of the T^2 statistic and upper control limit (UCL) depends on the source of the target statistics. In a Phase I chart, the limits are based on the same data that is being plotted on the control chart. In a Phase II chart, the limits are based on target statistics that were calculated from a historical data set. For more information about T^2 statistic and control limit calculations for Hotelling T^2 control charts, see Montgomery (2013).

Calculations for Phase I Control Charts

For Phase I control charts, the T^2 statistic for the j^{th} subgroup is defined as follows:

$$T_j^2 = (\bar{Y}_j - \bar{Y})' S_p^{-1} (\bar{Y}_j - \bar{Y})$$

where:

$\bar{Y}_j$ is the mean of the n column vectors of p measurements for the j^{th} subgroup

$\bar{Y} = \frac{1}{m} \sum_{j=1}^{m} \bar{Y}_j$ is the mean of the subgroup means

S_j is the sample covariance matrix for the n observations in the j^{th} subgroup

$S_p = \frac{1}{m} \sum_{j=1}^{m} S_j$ is the pooled covariance matrix, calculated as the mean of the within-subgroup covariance matrices

The Phase I upper control limit (UCL) is defined as follows:

$$UCL = \frac{p(m-1)(n-1)}{mn - m - p + 1} F_{[1-\alpha, p, mn-m-p+1]}$$

where:

p = number of variables

n = sample size for each subgroup

m = number of subgroups

$F_{[1-\alpha, p, mn-m-p+1]}$ = $(1-\alpha)^{th}$ quantile of an $F\ (p, mn-m-p+1)$ distribution

Calculations for Phase II Control Charts

In Phase II control charts, define the historical data set from which the target statistics are calculated as X. Then the T^2 statistic for the j^{th} subgroup is defined as follows:

$$T_j^2 = (\bar{Y}_j - \bar{X})' S_p^{-1} (\bar{Y}_j - \bar{X})$$

where:

$\bar{Y}_j$ is the mean of the n column vectors of p measurements for the j^{th} subgroup

$\bar{X}_k$ is the mean of the n column vectors of p measurements for the k^{th} subgroup from the historical data set

$\bar{X} = \frac{1}{m} \sum_{k=1}^{m} \bar{X}_k$ is the overall mean of the observations

S_k is the sample covariance matrix for the n observations in the k^{th} subgroup from the historical data set

$S_p = \frac{1}{m} \sum_{k=1}^{m} S_k$ is the pooled covariance matrix, calculated as the mean of the within-subgroup covariance matrices

The Phase II upper control limit (UCL) is defined as follows:

$$UCL = \frac{p(m+1)(n-1)}{mn-m-p+1} F_{[1-\alpha, p, mn-m-p+1]}$$

where:

p = number of variables

n = subgroup sample size

m = number of subgroups in the historical data set

$F_{[1-\alpha, p, mn-m-p+1]}$ = $(1-\alpha)^{th}$ quantile of an $F\ (p, mn-m-p+1)$ distribution

Additivity of Test Statistics for Observations in Rational Subgroups

When a sample of mn independent normal observations is grouped into m rational subgroups each of size n, define T^2_M as the distance between the mean $\overline{Y}_j$ of the jth subgroup and the target value. (T^2_M is equivalent to T^2 in the previous sections for observations in rational subgroups.) You can also calculate T^2 statistics related to the internal variability in each subgroup and the overall variability around the target value. The components of the T^2 statistic are additive, much like sums of squares. Specifically, the following relationship is true for each of the m subgroups:

$$T^2_{A_j} = T^2_{M_j} + T^2_{D_j}$$

In all of the following definitions, S_p is defined as it is in the previous sections, depending on whether the control chart is a Phase I or a Phase II control chart. Also, define μ as $\overline{Y}$ for Phase I control charts and as $\overline{X}$ for Phase II control charts.

The distance from the target value for the j^{th} subgroup is defined as follows:

$$T^2_{M_j} = n(\overline{Y}_j - \mu)' S_P^{-1} (\overline{Y}_j - \mu)$$

The internal variability for the j^{th} subgroup is defined as follows:

$$T^2_{D_j} = \sum_{i=1}^{n} (Y_{ji} - \overline{Y}_j)' S_P^{-1} (Y_{ji} - \overline{Y}_j)$$

where Y_{ji} is the i^{th} column vector of p measurements for the j^{th} subgroup.

The overall variability for the j^{th} subgroup is defined as follows:

$$T^2_{A_j} = \sum_{i=1}^{n} (Y_{ji} - \mu)' S_P^{-1} (Y_{ji} - \mu)$$

where Y_{ji} is the i^{th} column vector of p measurements for the j^{th} subgroup.

Note: When you select the **Save T Square** or **Save T Square Formula** options from the Multivariate Control Chart red triangle menu, the three values saved in each row correspond to one value of i in the three definitions above.

Statistical Details for Change Point Detection

This discussion follows the development in Sullivan and Woodall (2000).

Assumptions

Denote a multivariate distribution of dimension p with mean vector μ_i and covariance matrix Σ_i by $N_p(\mu_i,\Sigma_i)$. Suppose that the x_i are m (where $m > p$) independent observations from such a distribution:

$$x_i \sim N_p(\mu_i,\Sigma_i), \quad i = 1, \ldots, m$$

If the process is stable, the means μ_i and the covariance matrices Σ_i equal a common value so that the x_i have a $N_p(\mu, \Sigma)$ distribution.

Suppose that a single change occurs in either the mean vector or the covariance matrix, or both, between the m_1 and m_1+1 observations. Then the following conditions hold:

- Observations 1 through m_1 have the same mean vector and the same covariance matrix (μ_a,Σ_a).
- Observations $m_1 + 1$ to m have the same mean vector and covariance matrix (μ_b,Σ_b).
- One of the following occurs:
  - If the change affects the mean, $\mu_a \neq \mu_b$.
  - If the change affects the covariance matrix, $\Sigma_a \neq \Sigma_b$.
  - If the change affects both the mean and the covariance matrix, $\mu_a \neq \mu_b$ and $\Sigma_a \neq \Sigma_b$.

Overview

A likelihood ratio test approach is used to identify changes in one or both of the mean vector and covariance matrix. The likelihood ratio test statistic is used to compute a control chart statistic that has an approximate upper control limit of 1. The control chart statistic is plotted for all possible m_1 values. If any observation's control chart statistic exceeds the upper control limit of 1, this is an indication that a shift occurred. Assuming that exactly one shift occurs, that shift is considered to begin immediately after the observation with the maximum control chart statistic value.

Likelihood Ratio Test Statistic

The maximum value of twice the log-likelihood function for the first m_1 observations is defined as follows:

$$l_1 = -m_1 k_1 \log[2\pi] - m_1 \log\left[\left|S_1\right|_{k_1}\right] - m_1 k_1$$

The equation for l_1 uses the following notation:

- S_1 is the maximum likelihood estimate of the covariance matrix for the first m_1 observations.
- $k_1 = \text{Min}[p, m_1\text{-}1]$ is the rank of the p x p matrix S_1.
- The notation $|S_1|_{k_1}$ denotes the generalized determinant of the matrix S_1, which is defined as the product of its k_1 positive eigenvalues λ_j:

$$|S_1|_{k_1} = \prod_{j=1}^{k_1} \lambda_j$$

The generalized determinant is equal to the ordinary determinant when S_1 has full rank.

Denote the maximum of twice the log-likelihood function for the subsequent $m_2 = m - m_1$ observations by l_2, and the maximum of twice the log-likelihood function for all m observations by l_0. Both l_2 and l_0 are given by expressions similar to that given for l_1.

The likelihood ratio test statistic compares the sum $l_1 + l_2$ to l_0. The sum $l_1 + l_2$ is twice the log-likelihood that assumes a possible shift at m_1. The value l_0 is twice the log-likelihood that assumes no shift. If l_0 is substantially smaller than $l_1 + l_2$, the process is assumed to be unstable.

The likelihood ratio test statistic for a test of whether a change begins at observation $m_1 + 1$ is defined as follows:

$$\begin{aligned} \text{lrt}[m_1] &= (l_1 + l_2 - l_0) \\ &= (m_1(p-k_1) + m_2(p-k_2))(1 + \log(2\pi)) \\ &\quad + m\log[|S|] - m_1\log\left[|S_1|_{k_1}\right] - m_2\log\left[|S_2|_{k_2}\right] \end{aligned}$$

The distribution of the likelihood ratio test statistic is asymptotically chi-square distributed with $p(p + 3)/2$ degrees of freedom. Large log-likelihood ratio values indicate that the process is unstable.

The Control Chart Statistic

Simulations indicate that the expected value of $\text{lrt}[m_1]$ varies based on the observation's location in the series, and, in particular, depends on p and m. See Sullivan and Woodall (2000).

Approximating formulas for the expected value of $\text{lrt}[m_1]$ are derived by simulation. To reduce the dependence of the expected value on p, $\text{lrt}[m_1]$ is divided by its asymptotic expected value, $p(p + 3)/2$.

The formulas for the approximated expected value of lrt[m_1] divided by $p(p+3)/2$ are defined as follows:

$$\text{ev}[m,p,m_1] = \begin{cases} a_p + m_1 b_p, & \text{if } m_1 < p+1 \\ a_p + (m-m_1)b_p, & \text{if } m - m_1 < p+1 \\ 1 + \dfrac{m-2p-1}{(m_1-p)(m-p-m_1)}, & \text{otherwise} \end{cases}$$

where

$$a_p = -\frac{0.08684(p-14.69)(p-2.036)}{(p-2)}$$

and

$$b_p = \frac{0.1228(p-1.839)}{(p-2)}$$

For $p = 2$, the value of ev[m,p,m_1] when m_1 or $m_2 = 2$ is 1.3505.

Note: The formulas above are not accurate for $p > 12$ or $m < (2p + 4)$. In such cases, simulation should be used to obtain approximate expected values.

An approximate upper control limit that yields a false out-of-control signal with probability approximately 0.05, assuming that the process is stable, is calculated as follows:

$$\text{UCL}[m,p] \cong (3.338 - 2.115\log[p] + 0.8819(\log[p])^2 - 0.1382(\log[p])^3)$$

$$+ \quad (0.6389 - 0.3518\log[p] + 0.01784(\log[p])^3)\log[m].$$

Note that this formula depends on m and p.

The control chart statistic is defined to be twice the log of the likelihood ratio test statistic divided by $p(p + 3)$, divided by its approximate expected value, and also divided by the approximate value of the control limit. Because of the division by the approximate value of the UCL, the control chart statistic can be plotted against an upper control limit of 1. The approximate control chart statistic is calculated as follows:

$$\hat{y}[m_1] = \frac{2\text{lrt}[m_1]}{p(p+3)(\text{ev}[m,p,m_1]\text{UCL}[m,p])}$$

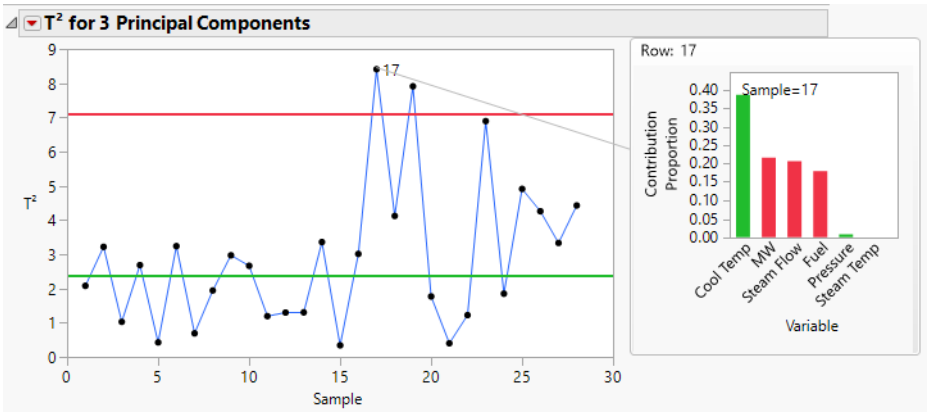
Chapter 11

Model Driven Multivariate Control Charts

Monitor and Diagnosis a Complex Process

Model-driven multivariate control charts are used to monitor parameters for multiple processes in a single control chart. The Model Driven Multivariate Control Chart (MDMVCC) platform enables you to build a control chart based on principal components or partial least squares models. For a set of continuous variables, the MDMVCC platform uses principal components to build the control chart. For saved principal components or partial least squares score functions, the MDMVCC platform builds a control chart based on the provided models. Use the MDMVCC platform to interactively explore and understand the underlying components that lead to out-of-control signals.

Figure 11.1 Model-driven Multiple Control Chart



Contents

Overview of Model Driven Multivariate Control Charts 307
Example of Model Driven Multivariate Control Charts 307
Launch the Model Driven Multivariate Control Chart Platform 310
Data Format 311
The Model Driven Multivariate Control Chart Report 312
Model Driven Multivariate Control Chart Platform Options 312
Plot Options 313
Score Plot 315
Additional Examples of the Model Driven Multivariate Control Chart Platform 315
Example of an MDMVCC with Historical Data 316
Example of an MDMVCC with a PLS Model 317
Statistical Details for the Model Driven Multivariate Control Chart Platform 318
Monitoring Statistics 318
Limits 320
Contributions 323
Score Plot Group Comparisons 325
Score Plot Loadings 325

Overview of Model Driven Multivariate Control Charts

The Model Driven Multivariate Control Chart (MDMVCC) platform has two primary functions: monitoring and diagnosing.

- Use multivariate control charts to monitor a multivariate process.
- You can interactively drill down to investigate the contributions of individual variables to the overall signal to diagnosis the process.

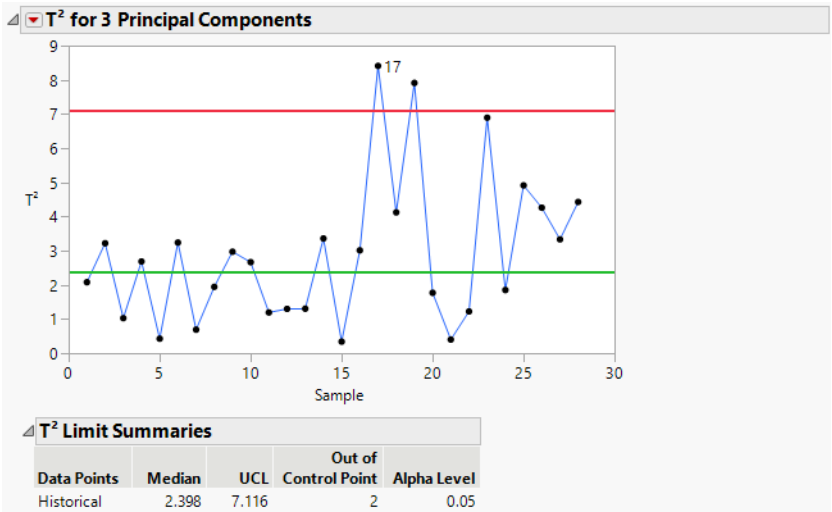
For more information about multivariate control charts, see Kourti and MacGregor (1996).

Example of Model Driven Multivariate Control Charts

This example uses the Steam Turbine Historical.jmp sample data table that contains process variables from a steam turbine system. You want to build a control chart for the six monitored variables.

1. Select **Help > Sample Data Library** and open Quality Control/Steam Turbine Historical.jmp.
2. Select **Analyze > Quality and Process > Model Driven Multivariate Control Chart**.
3. Select all six columns, click **Process,** and click **OK**.

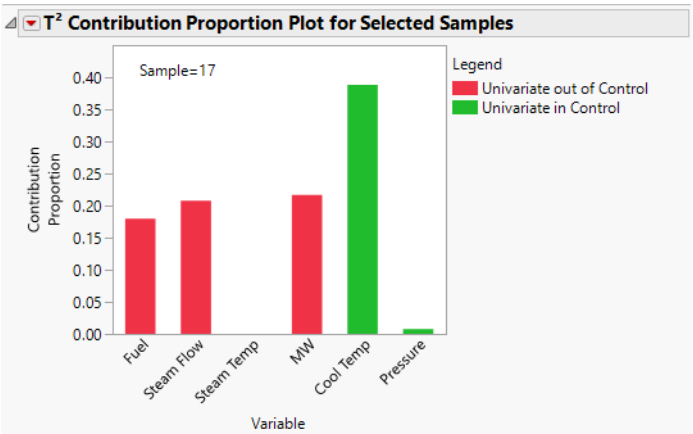
Figure 11.2 Steam Turbine Report



Note that the process shifts after sample 16.

4. Select the sample 17 data point. Right-click and select **Rows > Row Label**.
5. Hover over the sample 17 data point to view the T^2 contribution proportion plot for that point. Click on the plot to open the plot in the report window.

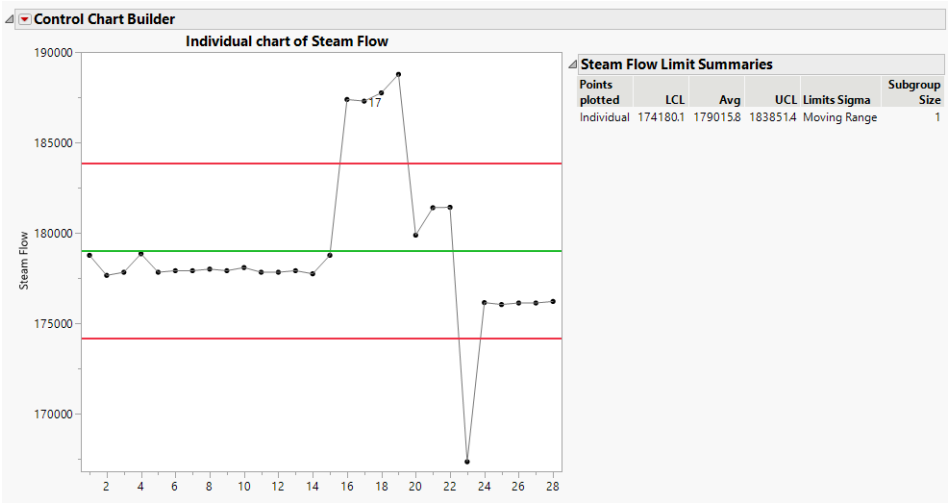
Figure 11.3 Contribution Proportion Plot for Sample 17



Note that Cool Temp contributes 40% of the T^2 value. The Cool Temp bar is green indicating that sample 17 is within the univariate control limits for Cool Temp. Steam Flow and MW each contribute about 20% of the T^2 value. They are both red, which indicates that sample 17 is outside of the univariate control limits for each variable. Steam Temp has a zero contribution to the T^2 value. In this example, you found variables where the multivariate out-of-control sample could be traced to an out-of-control univariate variable. However, that is not always the case. In multivariate process control you may observe an out-of-control point on the T^2 chart but find that the sample is in-control at the univariate level for all variables.

6. Hover over the Steam Flow bar in the contribution proportion plot to see a univariate control chart for Steam Flow. Click on the chart to open in a new report window.

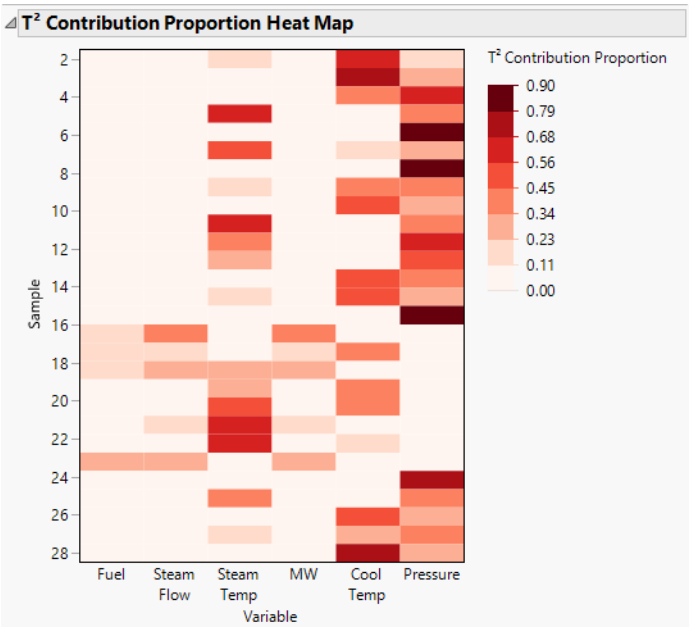
Figure 11.4 Individual Chart for Steam Flow



The individual chart indicates that the steam flow might have experienced an upset around sample 17. When control limits are specified in the Controls Limits column property, the individual chart respects those limits.

7. In the PCA Model Driven Multivariate Control Chart report window, Click the T^2 for 3 Principal Components red triangle and select **Contribution Proportion Heat Map**.

Figure 11.5 Contribution Proportion Heat Map

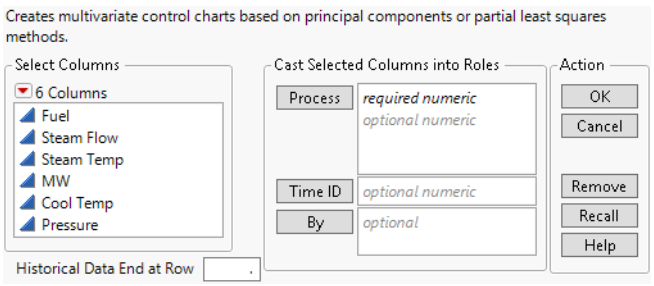


The contribution proportion heat map shows that there is a shift in the contribution proportions for rows 16, 17, and 18 and again at row 23 as compared to other rows. Generally, Steam Temp, Cool Temp, and Pressure contribute the most to the T^2 value for each row.

Launch the Model Driven Multivariate Control Chart Platform

Launch the Model Driven Multivariate Control Chart (MDMVCC) platform by selecting **Analyze > Quality and Process > Model Driven Multivariate Control Chart**.

Figure 11.6 The Model Driven Multivariate Control Chart Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Model Driven Multivariate Control Chart platform launch window contains the following options:

Process Assigns the process columns. See "Data Format".

Time ID Assigns a column that is used to identify samples. If no Time ID is assigned, the row number identifies the observations. If the Time ID column is a time, the time identifies each sample. Otherwise, the numeric value of the Time ID identifies each sample.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the By variables.

Historical Data End at Row Specifies a row number to indicate where historical data end. This enables you to calculate chart limits based on historical data. Both historical and current data are plotted on the charts. Historical data are also known as Phase I data, and current data are also known as Phase II data.

Data Format

The MDMVCC platform accepts data in the following three allowable data formats:

Raw Data Use continuous process data to build a control chart that is based on the principal components of the data. The default dimension of the control chart is based on the number of principal components that account for 85% of the process variation. This number is based on the cumulative percent of the principal component eigenvalues.

Principal Components Use principal component columns that were previously saved from a principal component analysis (PCA). The default dimension of the control chart is the number of components specified as process variables.

Partial Least Squares Score Data Use score columns that were previously saved from a partial least squares (PLS) analysis to build a control chart that is based on the score columns. The default dimension of the control chart is the number of scores specified as process variables.

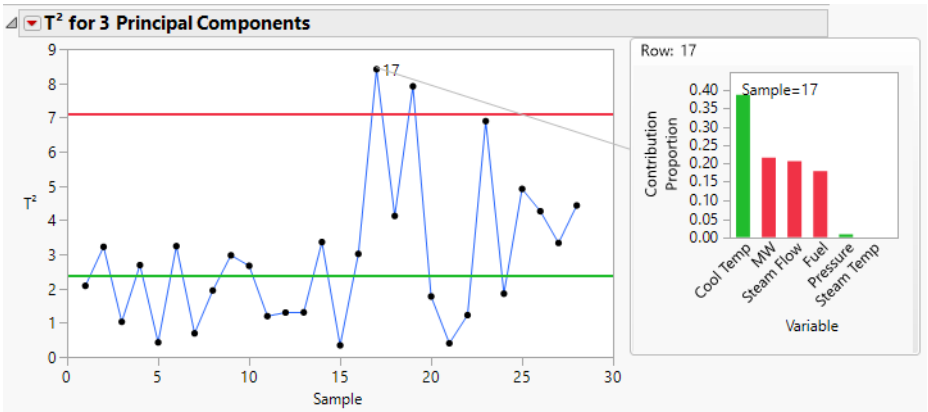
Notes:

- PCA or PLS models built with a frequency or weight column are not supported.
- PCA or PLS models built with historical data must use the same number of historical data rows as specified in the MDMVCC launch window.
- PCA models built from within the Multivariate platform are not supported.

The Model Driven Multivariate Control Chart Report

The initial Model Driven Multivariate Control Chart Report shows a T^2 control chart. The hover labels on the chart are themselves charts. Click the hover label charts to open larger versions of the charts. Depending on the chart, they open in a separate report window or in the Diagnosis the Process section of the MDMVCC report. You can use the graphlets to interactively drill down into the data.

Figure 11.7 MDMVCC Report with a Hover Graphlet



Model Driven Multivariate Control Chart Platform Options

Show History Summary Statistics Shows or hides summary statistics for rows designated as historical data or all rows if historical data rows are not specified. Summary statistics include univariate means and standard deviations for process variables. For PCA-based charts, the eigenvalues and eigenvectors are displayed. For charts based on PLS scores, the standard deviation of scores and the score loadings are displayed.

Monitor the Process

Show Monitoring Plots Shows or hides the selected process monitoring plots.

Set Component Enables you to set the number of components for the T^2, DModX, or SPE plots. The number of components can range from one up to the number of valid eigenvectors for PCA driven analysis or from one up to the number of PLS model factors for PLS driven analysis.

Set α Level Enables you to adjust the alpha level that is used for all control chart limits.

T^2 Plot Shows or hides a T^2 plot. The T^2 statistic is a summary of multivariate variation that measures how far away an observation is from the center of a PCA or PLS model.

Normalized DModX Plot Shows or hides a plot of the normalized DModX values. DModX measures the distance of each observation to the PCA or PLS model. A high DModX value indicates an observation that deviates from the underlying correlation structure of the data.

Squared Prediction Error Plot Shows or hides the squared prediction error (SPE) plot. SPE measures the sum of squared of the residuals from the PCA or PLS model. A high SPE value indicates an observation that deviates from the underlying correlation structure of the data.

Score Plot Shows or hides a score plot of principal components or partial least squares factors. See "Score Plot".

Diagnose the Process (Available when at least one diagnostic plot is active.) Shows or hides diagnostic plots.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Plot Options

The following options apply to the T^2, Normalized DModX, Squared Prediction Error, and Score Plots. The plots, when selected, appear in the Diagnosis the Process section of the report window.

Show Limit Summaries (Not available for the Score Plot.) Shows or hides control chart limits and summary data in a report table below the chart.

Contribution Heat Map Shows or hides a heat map that is colored by the variable contributions of each observation.

Contribution Proportion Heat Map Shows or hides a heat map that is colored by the variable contributions expressed as proportions of the overall value of each observation.

Contribution Plot for Selected Samples (Available only when one or more points are selected.) Shows or hides a bar chart of the individual component contributions to the overall value for each selected sample.

Contribution Proportion Plot for Selected Samples (Available only when one or more points are selected.) Shows or hides a bar chart of the individual component contributions expressed as proportions of contribution of the overall value for each selected sample.

Mean Contribution Proportion Plot for Selected Samples (Available only when two or more points are selected.) Shows or hides a bar chart of the average of the individual component contributions to the overall value for each selected sample.

Note: Bars are green if the value or mean value is within 3 sigma of the mean. Bars are red if the value or mean value is beyond 3 sigma of the mean. When control limits are specified in the Controls Limits column property, the bar coloring respects those limits.

The following options are available for contribution plots:

Sort Bars Enables you to sort the bars from largest to smallest contribution or from largest to smallest average contribution for multiple plots.

Label Bars Enables you to label the bars by the value, the column name, or to remove labels (No Label).

Control Charts for Selected Items Shows a control chart for the selected columns. When control limits are specified in the Controls Limits column property, the individual chart respects those limits.

Scatterplot Matrix (Available when two or more bars are selected.) Shows a scatterplot matrix for the selected variables.

Remove Plot Removes the plot from the report window.

Normalized Score Plot for Selected Samples (Available only for a Score Plot when one or more points are selected.) Shows or hides a bar chart of normalized scores for each sample selected.

Score Ellipse Coverage (Available only for Score Plots with two components.) Adds an ellipse with the specified coverage to the score plot. The ellipse is based on historical data. When both Phase I and Phase II data are present, there is an ellipse for each phase and the Phase II ellipse is dashed.

Connect Points (Available only for Score Plots.) Connects data points in the score plot.

Show Loadings (Available only for Score Plots.) Shows the PCA loadings on the score plot using biplot arrows.

Save Columns For each plot there are three options to save values to the data table:

Save Values Saves values (T^2, Normalized DModX, SPE, or scores) to a new data table column.

Save Contributions Saves contributions to new data table columns.

Save Contribution Proportions Saves contribution proportions to new data table columns.

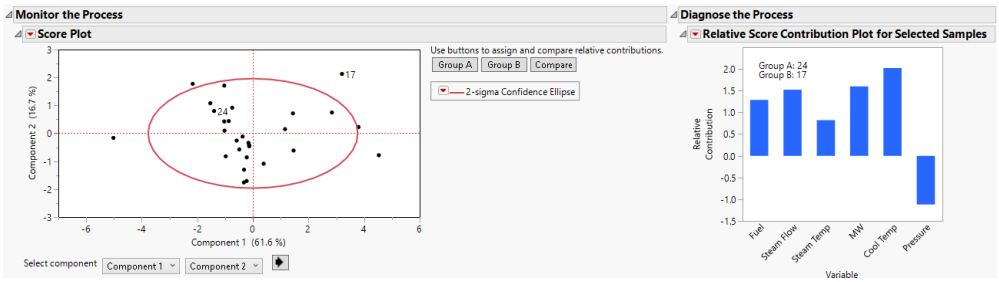
Score Plot

The Score Plot displays a plot of principal components or partial least squares factors. Use the controls below the plot to change the components shown in the Score Plot.

Use the buttons to the right of the plot to assign and compare the relative contributions between two groups of points. Relative contributions show how two or more samples differ from each other. Relative contributions show what changes in the underlying process variables contribute to differences in groups of samples. One use is to investigate the differences between an in-control sample and an out-of-control sample.

Group A is the reference group and Group B the comparator group. Each group can consist of one or more points. To assign the reference group, select one or more points and then click Group A. To assign the comparator group, select one or more points and then click Group B. To display the Relative Score Contribution Plot, click Compare.

Figure 11.8 Score Plot with Relative Contribution Plot for Row 17 Relative to Row 24



Additional Examples of the Model Driven Multivariate Control Chart Platform

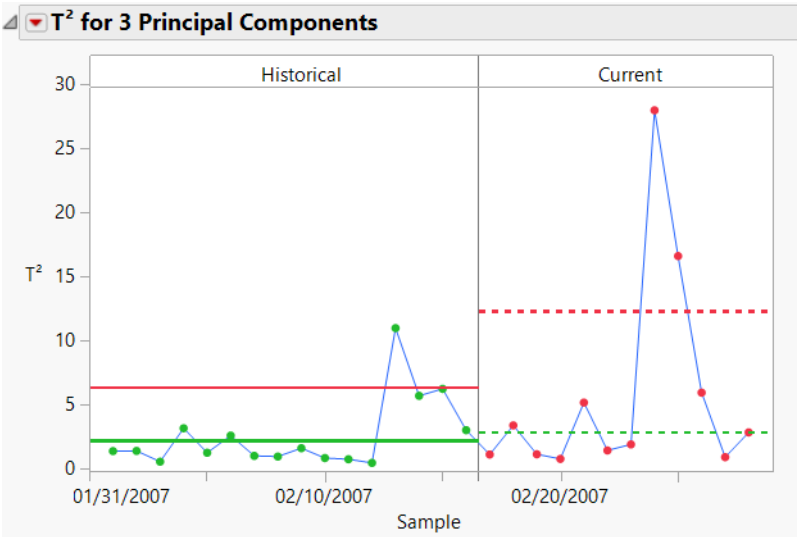
- "Example of an MDMVCC with Historical Data"
- "Example of an MDMVCC with a PLS Model"

Example of an MDMVCC with Historical Data

This example demonstrates the use of historical data to set the monitoring limits for current data.

1. Select **Help > Sample Data Library** and open Quality Control/Flight Delays.jmp.
2. Select **Analyze > Quality and Process > Model Driven Multivariate Control Chart**.
3. Select the AA through WN and click **Process**.
4. Select Flight Date and click **Time ID**.
5. Enter 16 for **Historical Data End at Row**.
6. Click **OK**.

Figure 11.9 T^2 Chart for Historical and Current Data



T² Limit Summaries

Data Points	Median	UCL	Out of Control Point	Alpha Level
Historical	2.42932	6.55278	1	0.05
Current	3.05848	12.5435	2	

Note that there are two sets of limits. One set applies to the historical data. A second set of limits applies to the current data. For more information about how the historical data is used to calculate the two sets of limits, see "Limits".

Tip: Turn on Automatic Recalc to enable the chart to automatically update as you add additional observations to the data table. The Automatic Recalc option is under redo when you click the PCA Model Driven Multivariate Control Chart red triangle.

Example of an MDMVCC with a PLS Model

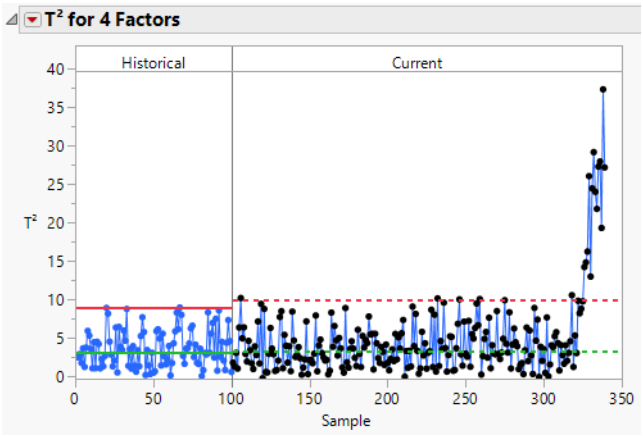
This example demonstrates the use of a PLS model for monitoring a multivariate process. Consider a process with 14 inputs and 5 quality variables. You have a PLS model that explains the process and you want to use this model to monitor the process for deviations.

1. Select **Help > Sample Data Library** and open Polyethylene Process.jmp.

   This data table contains 14 process variables and 5 quality or output variables. The first 100 rows are historical data used to build a PLS model to describe the process. These rows are colored blue. The remaining 239 rows are data collected since the model was built.

   The partial least squares model finds 4 score functions that describe the process. These functions are saved to the data table in columns X Score 1 Formula to X Score 4 Formula. To build the PLS model, use the table script *Set current data as excluded* to exclude the 239 rows of data collected after the historical data. Then use the *Fit Partial Least Squares* table script to build the PLS model to relate the quality variable to the process variables.
2. Select **Analyze > Quality and Process > Model Driven Multivariate Control Chart**.
3. Select the X Score 1 Formula through X Score 4 Formula and click **Process**.
4. Set the **Historical Data End at Row** to 100.
5. Click **OK**.

Figure 11.10 T^2 Chart

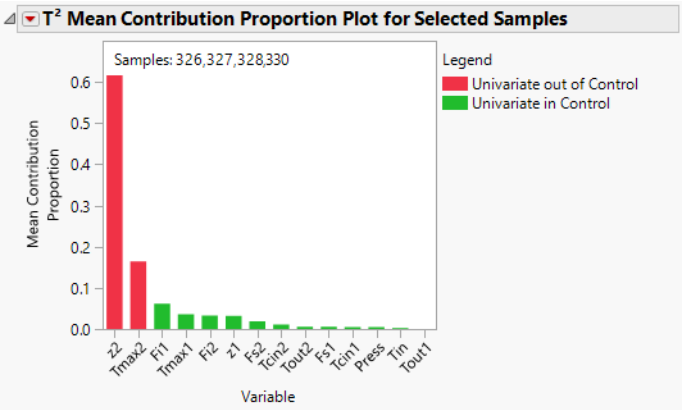


The blue data points represent the historical data. The black data points are data points collected after the control chart was established. The process experienced an upset that begins around sample number 326.

6. Hover over the sample data points that are above the upper control limit to view contribution plots.

7. Select the first cluster of data points above the upper control limit. Click the T^2 for 4 Factors red triangle and select **Mean Contribution Proportion Plot for Selected Samples**.
8. Click the red triangle next to T2 Mean Contribution Proportion Plot for Selected Samples and select **Sort Bars**.

Figure 11.11 Mean Contribution Proportion Plot



Notice that the contributions plot is in terms of the PLS model input variables. It appears that z2 and Tmax2 are causing the process upset. Tmax2 and z2 are related. Tmax2 is a reactor temperature and z2 is the location of the Tmax2 temperature.

Note: The descriptions of the factors are recorded in the Notes column property.

Statistical Details for the Model Driven Multivariate Control Chart Platform

Monitoring Statistics

T^2

The T_i^2 value for each of the i observations is plotted on the T^2 control chart. For historical and current data, the T^2 values for a PCA or PLS model with k components are defined as:

$$T_i^2 = t_i^T S_k^{-1} t_i$$

where:

t_i = the vector of k scores for the i^{th} observation

$\mathbf{S}_k$ = the diagonal sample covariance matrix of the k scores for historical observations

For PCA models, $\mathbf{S}_k$ is the diagonal eigenvalue matrix.

The mean of each of the k historical score vectors is 0 when the data is centered during the data preprocessing step. This step occurs in PCA on correlations or covariances and in PLS with centering. For preprocessing options where **X** is not centered, the data is assumed to have been centered by the user, so the mean of each of the k score vectors is 0. For more information about Hotelling's T^2, see Montgomery (2013).

SPE

For both PCA and PLS models, the preprocessed **X** matrix can be decomposed as:

$$X = T_k P_k^T + E$$

where $T_k = (t_1,...,t_k)$ is the k dimensional score matrix and $P_k = (p_1,...,p_k)$ is a matrix with the first k eigenvectors for PCA models or the loading matrix for PLS models. The squared prediction error of this PCA or PLS model is used for the SPE control chart.

The SPE_i value for each of the i observations is plotted on the SPE control chart. The squared prediction error is defined as:

$$SPE_i = e_i^T e_i = \sum_{j=1}^{p} e_{ij}^2$$

where

e_i = the residual vector for observation i

p = number of variables

DModX

The $DModX_i$ value for each of the i observations is plotted on the DModX control chart. The normalized distance to model (DModX) is defined as:

$$DModX_i = \frac{SPE_i/(df_1)}{\left(\sum_{i=1}^{n} SPE_i\right)/(df_2)} = \frac{\sum_{j=1}^{p} e_{ij}^2/(df_1)}{\sum_{i=1}^{n}\sum_{j=1}^{p} e_{ij}^2/(df_2)} = \frac{\sum_{j=1}^{p} e_{ij}^2}{d}$$

where

e_{ij} = the residual for observation i and variable j

$df_1 = p-k$

$df_2 = (n-k-1)(p-k)$ if the data is centered and $(n-k)(p-k)$ if the data is not centered

n = number of historical data observations

k = number of PCA/PLS components

p = number of variables

Note: $DModX_i$ is equal to SPE_i scaled by $1/d$.

Limits

All data are treated as historical data when the number of historical rows is not specified in the launch window. See "Launch the Model Driven Multivariate Control Chart Platform".

T^2

The upper control limit (UCL) for historical data is based on the Beta distribution and defined as:

$$UCL = \frac{(n-1)^2}{n}\beta\left[1-\alpha;\frac{k}{2};\frac{n-k-1}{2}\right]$$

where:

n = number of historical data observations

k = number of PCA or PLS components

$\beta\left[1-\alpha;\frac{k}{2};\frac{n-k-1}{2}\right]$ = $(1-\alpha)^{th}$ quantile of a Beta $\left[\frac{k}{2};\frac{n-k-1}{2}\right]$ distribution.

The UCL for current data is based on the F distribution and defined as:

$$UCL = \frac{k(n+1)(n-1)}{n(n-k)} F[1-\alpha;k;(n-k)]$$

where:

n = number of historical data observations

k = number of PCA or PLS components

$F(1-\alpha; k; n-k) = (1-\alpha)^{th}$ quantile of an $F(k; n-k)$ distribution.

DModX

For PCA and PLS models, the UCL is based on the F distribution. The DModX UCL for PCA models is defined as:

$$UCL = F[1-\alpha;df_1;df_2]$$

where:

$df_1 = p-k$

$df_2 = (n-k-1)(p-k)$ if the data is centered and $(n-k)(p-k)$ if the data is not centered

n = number of historical data observations

k = number of PCA components

p = number of variables

$F(1-\alpha; n-p-1; p-k) = (1-\alpha)^{th}$ quantile of a $F(n-p-1; p-k)$ distribution.

The DModX UCL for PLS models is defined as:

$$UCL = F[1-\alpha;h;nh]$$

where:

$$h = (2\hat{\mu}^2_{SPE})/(\hat{\sigma}^2_{SPE})$$

μ_{SPE} = historical sample mean of SPE

$\hat{\sigma}^2_{SPE}$ = historical sample variance of SPE

n = number of historical data observations

$F(1-\alpha; h; nh) = (1-\alpha)^{th}$ quantile of an $F(h; nh)$ distribution.

SPE

The SPE UCL for PCA models is defined as:

$$UCL = \theta_1 \left[1 - \frac{\theta_2 h_0 (1 - h_0)}{\theta_1^2} + \frac{z_{1-\alpha} (2\theta_2 h_0^2)^{1/2}}{\theta_1} \right]^{1/h_0}$$

where:

$$h_0 = 1 - 2\theta_1\theta_3 / (3\theta_2^2)$$

$$\theta_1 = \sum_{a=1}^{k} \lambda_a$$

$$\theta_2 = \sum_{a=1}^{k} \lambda_a^2$$

$$\theta_3 = \sum_{a=1}^{k} \lambda_a^3$$

λ_a = the a^{th} eigenvalue

k = number of PCA components

$z_{1-\alpha}$ = $(1-\alpha)^{\text{th}}$ quantile of the standard normal distribution

For more information about SPE control limits for PCA models, see Jackson and Mudholkar (1979).

For PLS models, the UCL is based on the chi-square distribution and defined as:

$$UCL = g\chi^2_{[1-\alpha;h]}$$

where

$$g = (\hat{\sigma}^2_{SPE}) / (2\hat{\mu}_{SPE})$$

$$h = (2\hat{\mu}^2_{SPE}) / (\hat{\sigma}^2_{SPE})$$

μ_{SPE} = historical sample mean of SPE

$\hat{\sigma}^2_{SPE}$ = historical sample variance of SPE

$\chi^2(1-\alpha; h) = (1-\alpha)^{th}$ quantile of an $\chi^2(h)$ distribution

The g and h parameters are estimated by the method of moments. For more information about SPE control limits for PLS models, see Nomikos (1995).

Contributions

T^2

The T^2 contributions for a PCA or PLS model with p variables and k components are calculated as:

$$\begin{aligned} T_i^2 &= t_i^T S_k^{-1} t_i \\ &= \sum_{a=1}^{k} \frac{t_{ia}^2}{s_a} \\ &= \sum_{a=1}^{k} \frac{t_{ia}}{s_a} \sum_{j=1}^{p} r_{ja} x_{ij} \\ &= \sum_{j=1}^{p} \left(\sum_{a=1}^{k} \frac{t_{ia}}{s_a} r_{ja} x_{ij} \right) \end{aligned}$$

where:

t_i = the vector of k scores for the i^{th} observation

$\mathbf{S}_k$ = the diagonal sample covariance matrix of the k scores for historical observations. For PCA models, $\mathbf{S}_k$ is the diagonal eigenvalue matrix.

s_a = the a^{th} diagonal element of $\mathbf{S}_k$

r_{ja} = the j^{th} element of the a^{th} eigenvector for PCA models and the a^{th} column of the $\mathbf{R}_k$ loading matrix for PLS models. $\mathbf{R}_k$ is the matrix used to relate the score matrix, $\mathbf{T}_k$ to the $\mathbf{X}$ matrix, such that $\mathbf{T}_k=\mathbf{X}\mathbf{R}_k$.

x_{ij} = the value of the j^{th} variable for the i^{th} observation.

Note: The p terms in the last sum are the variable contributions.

The contribution of each variable is the sum of its contribution to each score, weighted by the normalized score value. A variable is considered to have a large contribution to T_i^2 if there is a large normalized score value, and the variable contribution is large.

$$\sum_{j=1}^{p} Cont(T_i^2)_j = T_i^2$$

the contribution proportion of variable j is defined as:

$$ContProp(T_i^2)_j = \frac{Cont(T_i^2)_j}{\sum_{j=1}^{p} Cont(T_i^2)_j}$$

Note: When computing T^2 contribution proportions, JMP zeros out negative contributions. Negative contributions are possible due to the interaction of variables during the projection of **X** in PCA and PLS models. The negative contributions are zeroed in order to identify the variable contributions that represent a large proportion of the total positive contributions.

For more information about PCA contributions and negative contributions, see Kourti and MacGregor (1996). For more information about PLS contributions, see Li et al. (2009).

DModX

For PCA and PLS models, the contribution of variable j to $DModX_i$ is defined as:

$$Cont(DModX_i)_j = \frac{e_{ij}}{\sqrt{d}}$$

Note that since

$$DModX_i = \sum_{j=1}^{p} \frac{e_{ij}^2}{d}$$

the contribution proportion of variable j is defined as:

$$ContProp(DModX_i)_j = \frac{e_{ij}^2/d}{DModX_i}$$

SPE

For PCA and PLS models, the contribution of variable j to SPE_i is defined as:

$$Cont(SPE_i)_j = e_{ij}$$

Note that since

$$SPE_i = \sum_{j=1}^{p} e_{ij}^2$$

the contribution proportion of variable j is defined as:

$$ContProp(SPE_i)_j = \frac{e_{ij}^2}{SPE_i}$$

Score Contributions

The score contribution computation is the same as T^2 contributions but are computed only for the dimensions selected in the score plot.

Score Plot Group Comparisons

For the score plot group comparisons, the relative score contribution for variable j is the difference between the average contribution in group B and the average contribution in group A:

$$\sum_{i \in B} \frac{Cont(T_i^*)_j}{n_b} - \sum_{i \in A} \frac{Cont(T_i^*)_j}{n_a}$$

where

T^*_i = the i^{th} row of the score matrix with columns corresponding to the dimensions shown in the score plot.

A = the set of observations in group A

B = the set of observations in group B

n_a = the number of observations in group A

n_b = the number of observations in group B.

Score Plot Loadings

The loadings shown on the score plot are based on PCA eigenvectors or PLS X score loadings (**R** matrix). These loadings are scaled by the maximum absolute value of scores. The scaling is performed in order to graph the loadings on the score plot. The loadings illustrate each variable's approximate influence on each score.

Chapter 12

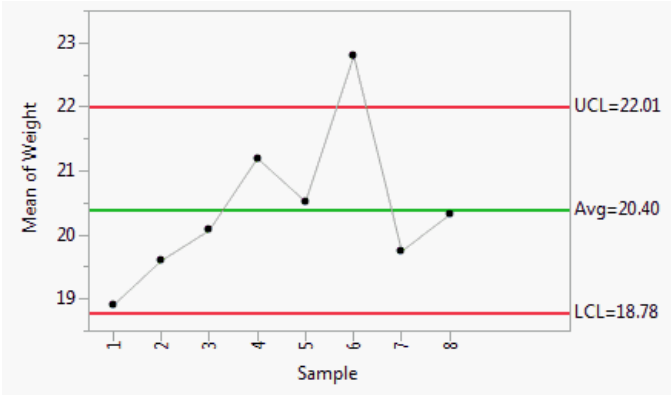
Legacy Control Charts

Create Variable and Attribute Control Charts

A control chart is a graphical and analytic tool for monitoring process variation. The natural variation in a process can be quantified using a set of control limits. Control limits help distinguish common-cause variation from special-cause variation. Typically, action is taken to identify and eliminate special-cause variation. It is also important to quantify the common-cause variation in a process, as this determines the capability of a process.

The legacy control chart platforms in JMP provide a variety of control charts, as well as runs charts, V-Mask CUSUM charts, and weighted moving average charts. To support process improvement initiatives, most of the control chart options display separate control charts for different phases of a project on the same chart.

Figure 12.1 Control Chart Example



Contents

Example of a Legacy Control Chart 329
Legacy Control Chart Types 330
Control Charts for Variables 330
Control Charts for Attributes 333
Launch a Legacy Control Chart Platform 334
Process Information 335
Limits Specifications 338
Specified Statistics 339
Legacy Control Chart Reports 340
V-Mask CUSUM Chart Reports 342
Interpret a Two-Sided V-Mask CUSUM Chart 342
Interpret a One-Sided CUSUM Chart 343
Legacy Control Chart Platform Options 344
Window Options for Legacy Control Charts 344
Chart Options for Legacy Control Charts 346
Chart Options for V-Mask CUSUM Control Charts 349
Saving and Retrieving Limits 349
Excluded, Hidden, and Deleted Samples 352
Additional Examples of the Control Chart Platform 354
Presummarize Chart Example 354
V-Mask CUSUM Chart Example 356
One-Sided CUSUM Chart Example 358
UWMA Chart Example 359
Statistical Details for the Control Chart Platform 360
Control Limits for Median Moving Range Charts 360
Statistical Details for Capability Analysis 360
Statistical Details for V-Mask CUSUM Control Charts 367
Statistical Details for Weighted Moving Average Charts 370

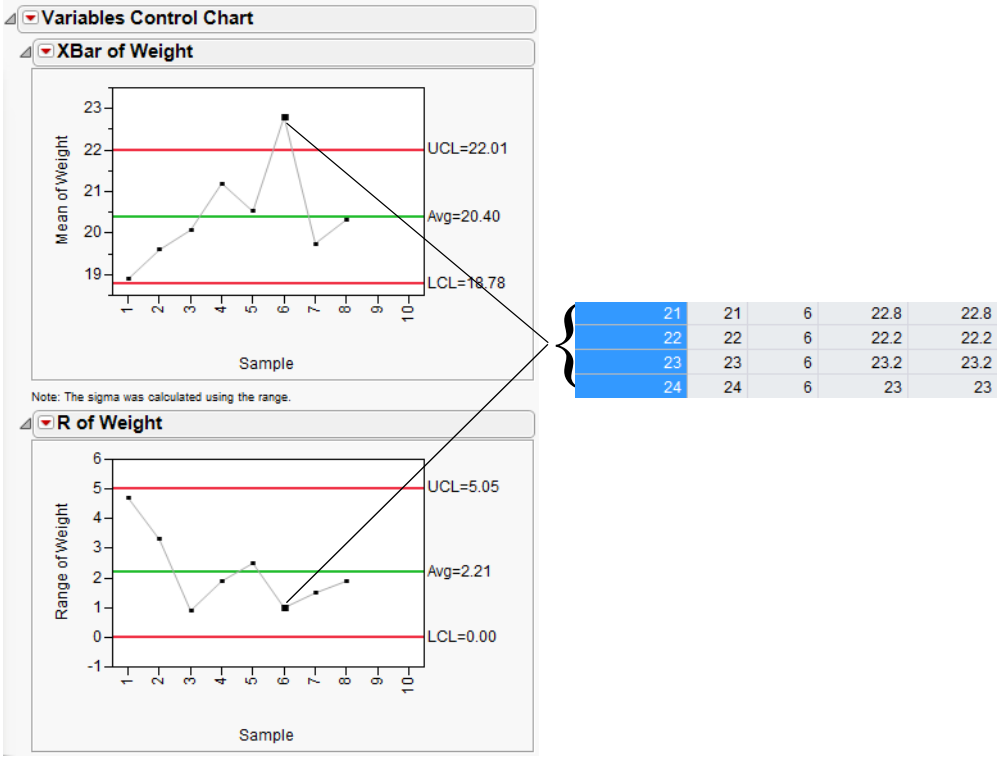
Example of a Legacy Control Chart

This example uses the Coating.jmp sample data table in the Quality Control sample data folder (taken from the *ASTM Manual on Presentation of Data and Control Chart Analysis*). The quality characteristic of interest is the Weight column. A subgroup sample of four is chosen.

1. Select **Help > Sample Data Library** and open Quality Control/Coating.jmp.
2. Select **Analyze > Quality And Process > Legacy Control Charts > XBar**.

   Note the selected chart types of **XBar** and **R**.
3. Select Weight and click **Process**.
4. Select Sample and click **Sample Label**.
5. Click **OK**.

Figure 12.2 Variables Charts for Coating Data



An XBar chart and an R chart for the process are shown in Figure 12.2. Sample six indicates that the process is not in statistical control. To check the sample values, click the sample six summary point on either control chart. The corresponding rows highlight in the data table.

Note: If an S chart is chosen with the XBar chart, then the limits for the XBar chart are based on the standard deviation. Otherwise, the limits for the XBar chart are based on the range.

Legacy Control Chart Types

Legacy control charts are broadly classified into two categories:

- "Control Charts for Variables" — IR, XBar, Runs Chart, Levey-Jennings, Presummarize, CUSUM, UWMA, and EWMA.
- "Control Charts for Attributes" — P, NP, C, and U.

Control Charts for Variables

Control charts for variables are classified according to the subgroup summary statistic plotted on the chart:

- The **IR** selection provides additional chart types:
  - Individual Measurement charts display individual measurements. These charts are appropriate when only one measurement is available for each subgroup sample.
  - Moving Range charts display moving ranges of two or more successive measurements. See "Moving Range (Average) Charts".
- **XBar** charts display subgroup means (averages). This selection provides additional chart types:
  - **R** charts display subgroup ranges (maximum – minimum).
  - **S** charts display subgroup standard deviations.

  For quality characteristics measured on a continuous scale, a typical analysis shows both the process mean and its variability with a mean chart aligned above its corresponding R or S chart.
- **Runs Chart** displays data as a connected series of points. Runs charts can also plot the group means when the **Sample Label** role is used, either on the window or through a script.
- **Levey-Jennings** charts show a process mean with control limits based on a long-term sigma. The control limits are placed at 3*s* distance from the center line. The standard deviation, *s*, for the Levey-Jennings chart is calculated the same way standard deviation is in the Distribution platform.
- **Presummarize** charts display subgroup means and standard deviations. See "Presummarize Charts".

- **CUSUM** charts show cumulative sums of subgroup or individual measurements from a target value. See "V-Mask CUSUM Control Charts".
- UWMA charts show a uniformly weighted moving average of a specified number of measurements. See "Uniformly Weighted Moving Average Charts".
- EWMA charts show an exponentially weighted moving average of all measurements with a specified weight. See "Exponentially Weighted Moving Average Charts".

Moving Range (Average) Charts

In a Moving Average chart, the quantities that are averaged can be individual observations instead of subgroup means. However, a Moving Average chart for individual measurements is not the same as a control chart for individual measurements or moving ranges with individual measurements plotted.

Moving Range (Average) charts display moving ranges of two or more successive measurements. Moving ranges are computed for the number of consecutive measurements that you enter in the **Range Span** box. The default range span is 2. Because moving ranges are correlated, these charts should be interpreted with care.

A Median Moving Range chart is also available. If you choose a Median Moving Range chart and an Individual Measurement chart, the limits on the Individual Measurement chart use the Median Moving Range as the sigma, rather than the Average Moving Range.

Presummarize Charts

If your data consist of repeated measurements of the same process unit, you can combine these into one measurement for the unit. Pre-summarizing is not recommended unless the data have repeated measurements on each process or measurement unit.

Presummarize summarizes the process column into sample means or standard deviations, based either on the sample size or sample label chosen. Then it charts the summarized data based on the options chosen in the launch window. You can also append a capability analysis by checking the appropriate box in the launch window.

The **Presummarize** launch window has the following options for chart types:

- Individual on Group Means
- Individual on Group Std Devs
- Moving Range on Group Means
- Moving Range on Group Std Devs
- Median Moving Range on Group Means
- Median Moving Range on Group Std Devs

There is also an option for setting the range span that is used for the moving range chart types.

V-Mask CUSUM Control Charts

V-Mask Cumulative Sum (CUSUM) control charts show cumulative sums of subgroup or individual measurements from a target value. V-Mask CUSUM charts can help you decide whether a process is in a state of statistical control by detecting small, sustained shifts in the process mean. In comparison, standard Shewhart control charts can detect sudden and large changes in measurement, such as a two or three sigma shift, but they are less effective at spotting smaller changes, such as a one sigma shift.

The **CUSUM** menu selection has options for V-mask cumulative sum charts. In addition to KSigma, you also specify:

- The vertical distance h between the origin for the V-mask and the upper or lower arm of the V-mask for a two-sided chart. For a one-sided chart, H is the decision interval. Choose H as a multiple of the standard error.
- The reference value k, where k is greater than zero.

Another form of a cumulative sum control chart is the tabular CUSUM chart. To create a tabular CUSUM chart, see "CUSUM Control Charts". The tabular CUSUM chart is recommended over the V-mask chart for a variety of reasons, including the following:

- The V-mask must be moved with each observation, not simply placed on the last observation.
- The cumulative sums in the V-mask procedure can end up a long way from the center of the graph, even for an on-target process.

Caution: Montgomery (2013) strongly "advises against using the V-mask procedure."

Uniformly Weighted Moving Average Charts

Each point on a Uniformly Weighted Moving Average (UWMA) chart is the average of the w most recent subgroup means, including the present subgroup mean. When you obtain a new subgroup sample, the next moving average is computed by dropping the oldest of the previous w subgroup means and including the newest subgroup mean. The constant, w, is called the *span* of the moving average.

In addition to KSigma and Alpha, in the UWMA launch window you also specify:

- The Moving Average Span, or w, which indicates how many subgroups to include to form the moving average. The larger the Moving Average Span (w), the smoother the UWMA line, and the less it reflects the magnitude of shifts. This means that larger values of w guard against smaller shifts. See "Control Limits for UWMA Charts".

Exponentially Weighted Moving Average Charts

Each point on an Exponentially Weighted Moving Average (EWMA) chart is the weighted average of all the previous subgroup means, including the mean of the present subgroup sample. The weights decrease exponentially going backward in time.

Note: An Exponentially Weighted Moving Average (EWMA) chart can also be called a Geometric Moving Average (GMA) chart.

In addition to KSigma and Alpha, in the EWMA launch window you also specify:

- A Weight parameter, which is the weight ($0 < \text{weight} \leq 1$) assigned to the present subgroup sample mean. Small values of Weight are used to guard against small shifts. See "Control Limits for EWMA Charts".

Tip: See "EWMA Control Charts" for the newer EWMA Control Charts platform.

Control Charts for Attributes

In the previous types of charts, measurement data was the process variable. This type of data is often continuous, and the charts are based on theory for continuous data. Another type of data is count data, where the variable of interest is a discrete count of the number of defects or blemishes per subgroup. For discrete count data, attribute charts are applicable, as they are based on binomial and Poisson models. Because the counts are measured per subgroup, it is important when comparing charts to determine whether you have a similar number of items in the subgroups between the charts. Attribute charts, like variables charts, are classified according to the subgroup sample statistic plotted on the chart.

Determining Which Attribute Chart to Use

Each item is judged as either conforming or non-conforming:

p-chart Shows the *proportion* of defective items.

np-chart Shows the *number* of defective items.

The number of defects is counted for each item:

c-chart Shows the *number* of defects.

u-chart Shows the *proportion* of defects.

For attribute charts, specify the column containing the defect count or defective proportion as the Process variable. The data are interpreted as counts, unless the column contains non-integer values between 0 and 1.

- P charts display the proportion of nonconforming (defective) items in subgroup samples, which can vary in size. Since each subgroup for a P chart consists of N_i items, and an item is judged as either conforming or nonconforming, the maximum number of nonconforming items in a subgroup is N_i.
- NP charts display the number of nonconforming (defective) items in subgroup samples. Because each subgroup for an NP chart consists of N_i items, and an item is judged as either conforming or nonconforming, the maximum number of nonconforming items in subgroup i is N_i.

  Note: To use the Sigma column property for P or NP charts, the value needs to be equal to the proportion. JMP calculates the sigma as a function of the proportion and the sample sizes.

- C charts display the number of nonconformities (defects) in a subgroup sample that usually, but does not necessarily, consists of one inspection unit.

  Caution: For a C chart, if you do not specify a Sample Size or Constant Size, then the Sample Label is used as the sample size.

- U charts display the proportion of nonconformities (defects) in each subgroup sample that can have a varying number of inspection units.

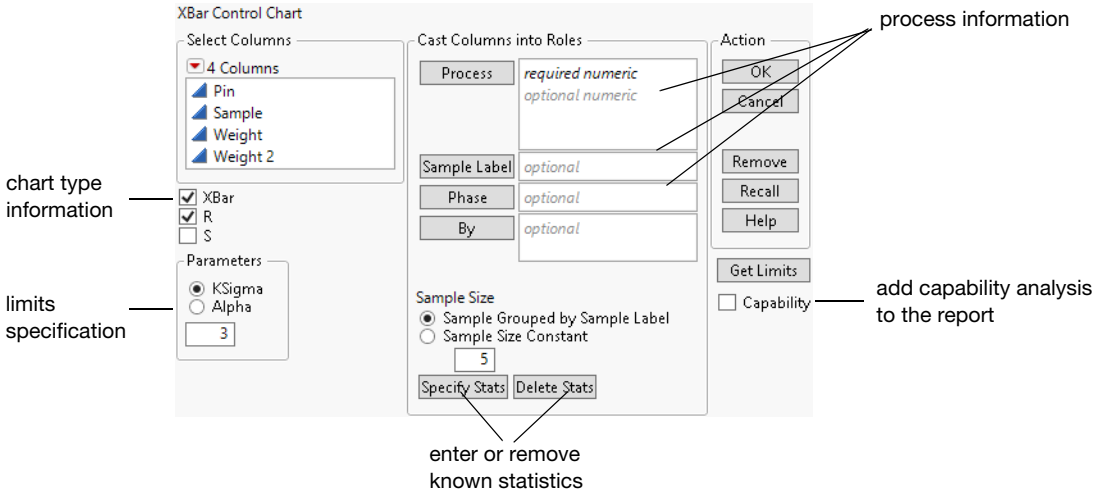
  Caution: For a U chart, if you do not specify a Unit Size or Constant Size, then the Sample Label is used as the unit size.

Launch a Legacy Control Chart Platform

When you launch a legacy control chart platform by selecting **Analyze > Quality And Process > Legacy Control Charts**, a launch window similar to Figure 12.3 appears. The specific controls vary depending on which type of chart you select. Initially, the window shows the following types of information:

- "Process Information" for measurement variable selection
- Chart type information (for more information, see "Legacy Control Chart Types")
- "Limits Specifications"
- "Specified Statistics"

Figure 12.3 XBar Control Chart Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Process Information

The launch window displays a list of columns in the current data table. Here, you specify the variables to be analyzed and the subgroup sample size.

Process

The **Process** role selects variables for charting:

- For variables charts, specify measurements as the process.
- For attribute charts, specify the defect count or defective proportion as the process. The data are interpreted as counts, unless it contains non-integer values between 0 and 1.

Note: The rows of the data table must be sorted in the order in which the observations were collected. Even if there is a **Sample Label** variable specified, you still must sort the observations accordingly.

Sample Label

The **Sample Label** role enables you to specify a variable whose values label the horizontal axis and can also identify unequal subgroup sizes. If you do not specify a sample label variable, the samples are identified by their subgroup sample number.

- If the sample subgroups are the same size, select the **Sample Size Constant** option and enter the size in the text box. If you entered a Sample Label variable, its values are used to label the horizontal axis. The sample size is used in the calculation of the limits regardless of whether the samples have missing values.
- If the sample subgroups have an unequal number of rows or have missing values and you have a column identifying each sample, select the **Sample Grouped by Sample Label** option and enter the sample identifying column as the sample label.

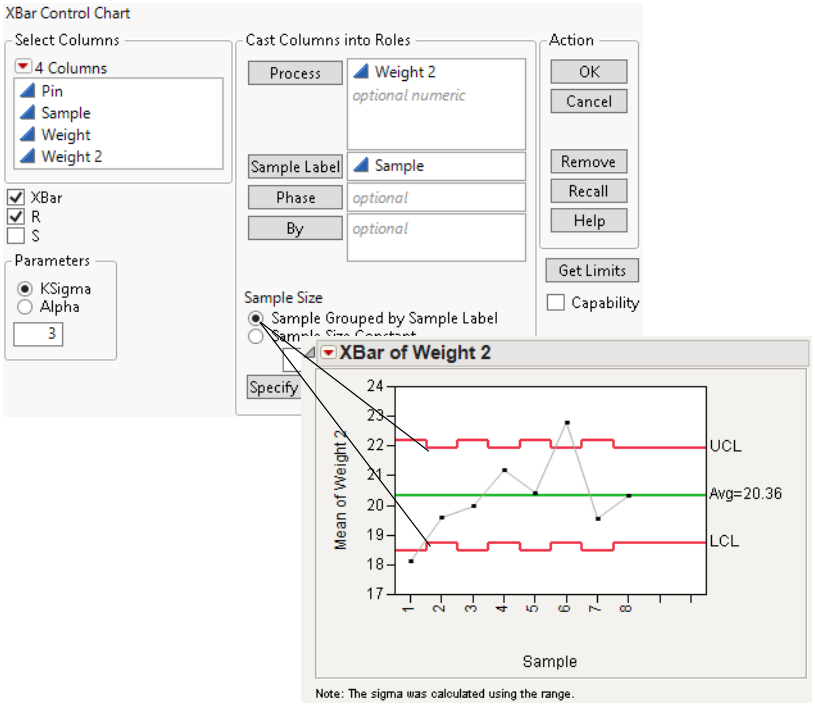
For attribute charts (P, NP, C, and U charts), this variable is the subgroup sample size. Additional options appear on the launch window, including **Sample Size**, **Constant Size**, or **Unit Size**, depending on your selection. In variables charts, it identifies the sample. When the chart type is **IR**, a **Range Span** text box appears. The *range span* specifies the number of consecutive measurements from which the moving ranges are computed.

Notes:

- The rows of the data table must be sorted in the order in which the observations were collected. Even if there is a **Sample Label** variable specified, you still must sort the observations accordingly.
- The non-integer part of the value for **Constant Size** is truncated. If you have a constant non-integer subgroup sample size, you must specify a column of constant values.

The illustration in Figure 12.4 shows an XBar chart for a process with unequal subgroup sample sizes, using the Coating.jmp sample data from the Quality Control sample data folder.

Figure 12.4 Variables Charts with Unequal Subgroup Sample Sizes



Phase

The **Phase** role enables you to specify a column identifying different phases, or sections. A *phase* is a group of consecutive observations in the data table. For example, phases might correspond to time periods during which a new process is brought into production and then put through successive changes. Phases generate, for each level of the specified Phase variable, a new sigma, set of limits, zones, and resulting tests.

On the window for XBar, R, S, IR, P, NP, C, U, Presummarize, and Levey-Jennings charts, a **Phase** variable button appears. If a phase variable is specified, the phase variable is examined, row by row, to identify to which phase each row belongs. Saving to a limits file reveals the sigma and specific limits calculated for each phase.

By

The **By** role identifies a variable to produce a separate analysis for each value that appears in the column.

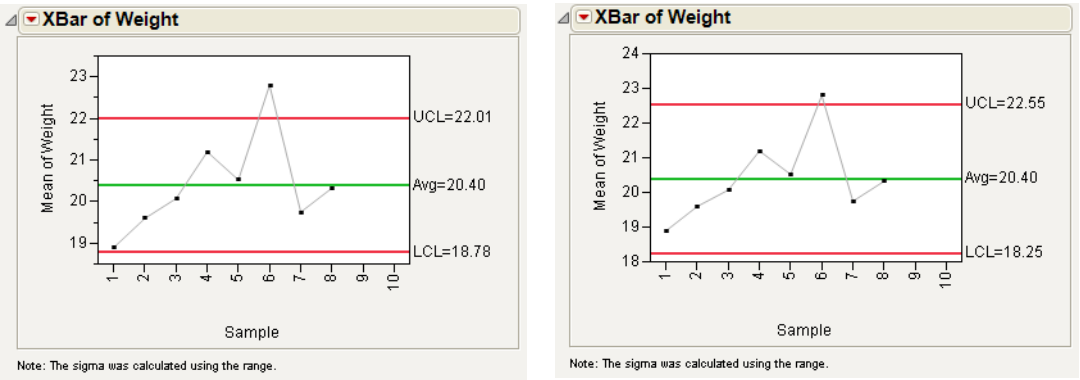
Limits Specifications

You can specify computations for control limits by entering a value for *k* (**K Sigma**), or by entering a probability for α (**Alpha**), or by retrieving a limits value from the process columns' properties or a previously created Limits Table. Limits Tables and the **Get Limits** button are discussed in the section "Saving and Retrieving Limits". There must be a specification of either **K Sigma** or **Alpha**. The window default for **K Sigma** is 3.

KSigma

The **KSigma** parameter option enables specification of control limits in terms of a multiple of the sample standard error. **KSigma** specifies control limits at *k* sample standard errors above and below the expected value, which shows as the center line. To specify *k*, the number of sigmas, click the radio button for **KSigma** and enter a positive *k* value into the text box. The usual choice for *k* is 3, which is three standard deviations. The examples shown in Figure 12.5 compare the XBar chart for the Coating.jmp data with control lines drawn with **KSigma** = 3 and **KSigma** = 4.

Figure 12.5 K Sigma =3 (left) and K Sigma=4 (right) Control Limits



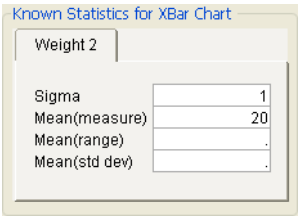
Alpha

The **Alpha** parameter option specifies control limits (also called *probability limits*) in terms of the probability α that a single subgroup statistic exceeds its control limits, assuming that the process is in control. To specify alpha, click the **Alpha** radio button and enter the desired probability. Reasonable choices for α are 0.01 or 0.001. For XBar charts under the assumption of normality and known in-control parameters, the **Alpha** value equivalent to a **KSigma** of 3 is 0.0027.

Specified Statistics

After specifying a process variable, if you click the **Specify Stats** (when available) button on a Control Chart launch window, a tab with editable fields is appended to the bottom of the window. This lets you enter historical statistics (that is, statistics obtained from historical data) for the process variable. The Control Chart platform uses those entries to construct control charts. The example here shows 1 as the standard deviation of the process variable and 20 as the mean measurement.

Figure 12.6 Example of Specify Stats



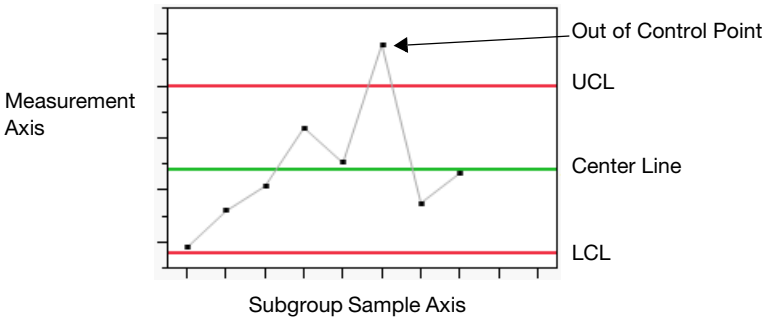
Note: When the mean is user-specified, it is labeled in the plot as μ0.

If you check the **Capability** option on a Control Chart launch window (Figure 12.3), a window appears as the platform is launched asking for specification limits. The standard deviation for the control chart selected is sent to the window and appears as a Specified Sigma value, which is the default option. After entering the specification limits and clicking OK, capability output appears in the same window next to the control chart. For information about how the capability indices are computed, see "Capability Indices for Normal Distributions".

Legacy Control Chart Reports

The analysis produces a chart that can be used to determine whether a process is in a state of statistical control. The report varies depending on the type of chart that you select. Figure 12.7 displays the parts of a simple control chart. Control charts update dynamically as data is added or changed in the data table.

Figure 12.7 Example of a Control Chart



Note: Any rows that are excluded in the data table are also hidden in Runs charts, P charts, U charts, and C charts.

Control charts have the following characteristics:

- Each point plotted on the chart represents an individual process measurement or summary statistic. In Figure 12.7, the points represent the average for a sample of measurements.

  Subgroups should be chosen *rationally*, that is, they should be chosen to maximize the probability of seeing a true process change *between* subgroups. Often, this requires knowledge of the process to determine the most effective grouping strategy. See Wheeler (2004); Woodall and Adams (1998).
- The vertical axis of a control chart is scaled in the same units as the summary statistic.
- The horizontal axis of a control chart identifies the subgroup samples and is time ordered. Observing the process over time is important in assessing if the process is changing.
- The green line is the center line, or the average of the data. The center line indicates the average (expected) value of the summary statistic when the process is in statistical control. Measurements should appear equally on both sides of the center line. If not, this is possible evidence that the process average is changing.
- The two red lines are the upper and lower control limits, labeled UCL and LCL. These limits give the range of variation to be expected in the summary statistic when the process is in statistical control. If the process is exhibiting only routine variation, then all the points

should fall randomly in that range. In Figure 12.7, one measurement is above the upper control limit. This is evidence that the measurement could have been influenced by a special cause, or is possibly a defect.

- A point outside the control limits (or the V-mask of a CUSUM chart) signals the presence of a special cause of variation.

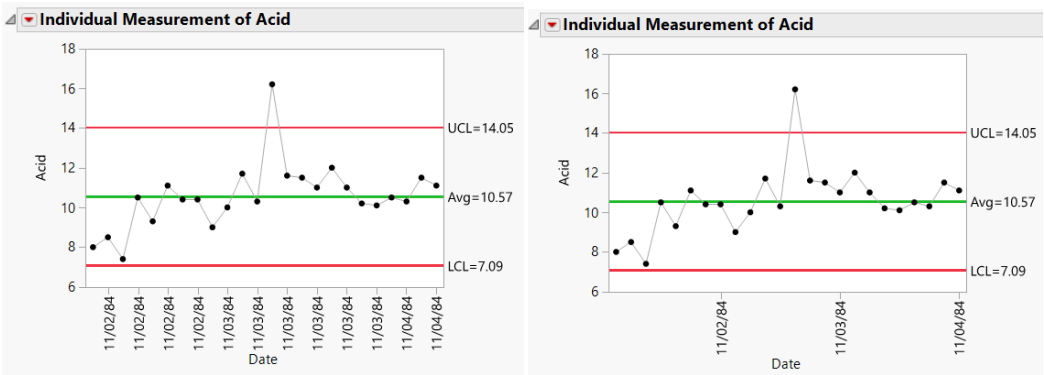
Options within each platform create control charts that can be updated dynamically as samples are received and recorded or added to the data table.

When a control chart signals abnormal variation, action should be taken to return the process to a state of statistical control if the process degraded. If the abnormal variation indicates an improvement in the process, the causes of the variation should be studied and implemented.

When you double-click the horizontal or vertical axis, the appropriate Axis Specification window appears for you to specify the format, axis values, number of ticks, gridline, reference lines, and other options to display on the axis.

For example, the Pickles.jmp data lists measurements taken each day for three days. In Figure 12.8, by default, the horizontal axis is labeled at every other tick. Sometimes this gives redundant labels, as shown to the left in Figure 12.8. If you specify a label at an increment of eight, the horizontal axis is labeled once for each day, as shown in the chart on the right.

Figure 12.8 Example of Labeled x Axis Tick Marks



Tip: For information about warnings and rules, see "Tests" and "Westgard Rules".

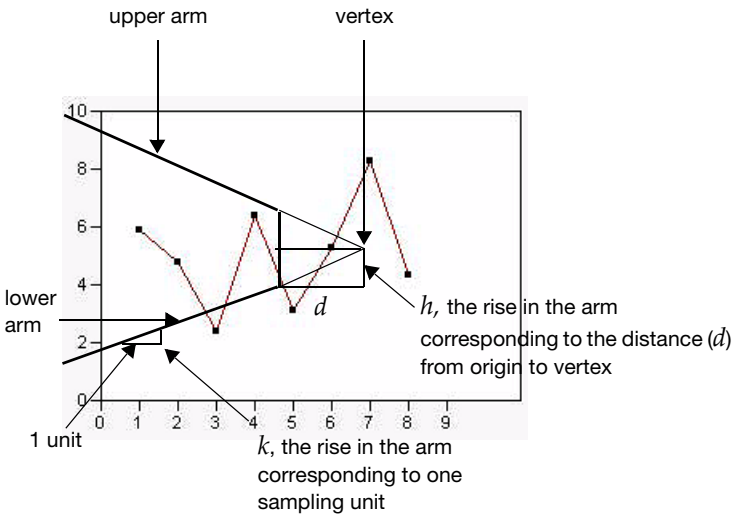
V-Mask CUSUM Chart Reports

Interpret a Two-Sided V-Mask CUSUM Chart

Note: See also "V-Mask CUSUM Chart Example".

To interpret a two-sided CUSUM chart, compare the points with limits that compose a V-mask. A V-mask is a shape in the form of a V on its side that is superimposed on the graph of the cumulative sums. The V-mask is formed by plotting V-shaped limits. The origin of a V-mask is the most recently plotted point, and the arms extended backward on the horizontal axis, as in Figure 12.9. As data are collected, the cumulative sum sequence is updated and the origin is relocated at the newest point.

Figure 12.9 V-Mask for a Two-Sided CUSUM Chart



Shifts in the process mean are visually easy to detect on a CUSUM chart because they produce a change in the slope of the plotted points. The point where the slope changes is the point where the shift occurs. A condition is *out-of-control* if one or more of the points previously plotted crosses the upper or lower arm of the V-mask. Points crossing the lower arm signal an increasing process mean, and points crossing the upper arm signal a downward shift.

There are important differences between CUSUM charts and Shewhart charts:

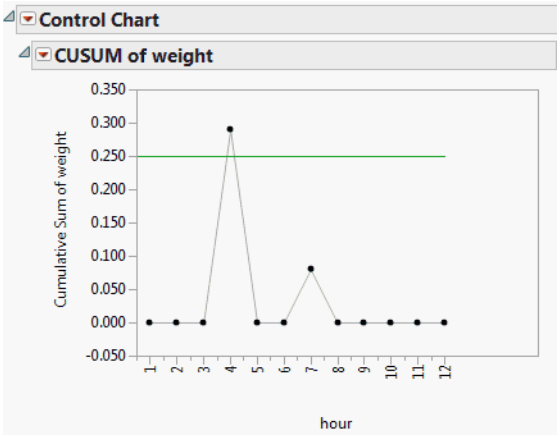
- A Shewhart control chart plots points based on information from a single subgroup sample. In CUSUM charts, each point is based on information from all samples taken up to and including the current subgroup.
- On a Shewhart control chart, horizontal control limits define whether a point signals an out-of-control condition. On a CUSUM chart, the limits can be either in the form of a V-mask or a horizontal decision interval.
- The control limits on a Shewhart control chart are commonly specified as 3σ limits. On a CUSUM chart, the limits are determined from average run length.

A CUSUM chart is more efficient for detecting small shifts in the process mean. Lucas (1976) states that a V-mask detects a 1σ shift about four times as fast as a Shewhart control chart.

Interpret a One-Sided CUSUM Chart

Use a one-sided CUSUM chart to identify data approaching or exceeding the side of interest.

Figure 12.10 Example of a One-Sided CUSUM Chart



The *decision interval* or horizontal line is set at the H value that you entered in the launch window. In this example, it is 0.25. Any values exceeding the decision interval of 0.25 indicate a shift or out-of-control condition. In this example, observation 4 appears to be where a shift occurred. Also note that no V-mask appears for one-sided CUSUM charts.

Legacy Control Chart Platform Options

Legacy control charts have red triangle menus that affect various parts of the platform:

- The menu on the top-most title bar affects the whole platform window. Its items vary with the type of chart that you select. See "Window Options for Legacy Control Charts".
- There is a menu of items on the chart type title bar with options that affect each chart individually. See "Chart Options for Legacy Control Charts".

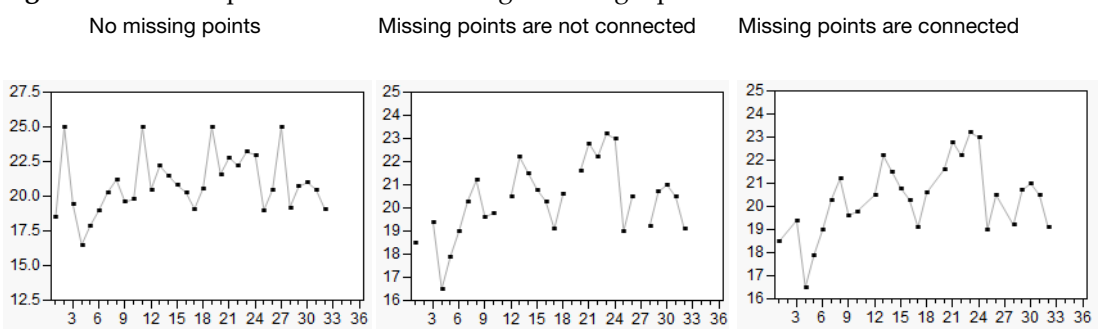
Window Options for Legacy Control Charts

The red triangle menu on the window title bar lists options that affect the report window. If you request **XBar** and **R** at the same time, you can check each chart type to show or hide it. The specific options that are available depend on the type of control chart you request. Unavailable options show as grayed menu items.

Show Limits Legend Shows or hides the Avg, UCL, and LCL values to the right of the chart.

Connect Through Missing Connects points when some samples have missing values. In Figure 12.11, the left chart has no missing points. The middle chart has samples 2, 11, 19, and 27 missing with the points not connected. The right chart appears if you select the **Connect Through Missing** option, which is the default.

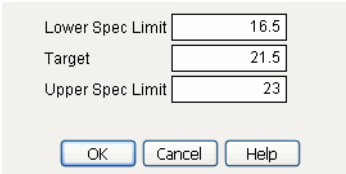
Figure 12.11 Example of Connected through Missing Option



Use Median For Runs Charts, when you select the **Show Center Line** option in the individual Runs Chart red triangle menu, a line is drawn through the center value of the column. The center line is determined by the **Use Median** setting of the main Runs Chart red triangle menu. When **Use Median** is selected, the median is used as the center line. Otherwise, the mean is used. When saving limits to a file, both the overall mean and median are saved.

Capability (Not available when a Phase variable is specified.) Performs a Capability Analysis for your data. A pop-up window is first shown, where you can enter the Lower Spec Limit, Target, and Upper Spec Limit values for the process variable.

Figure 12.12 Capability Analysis Window



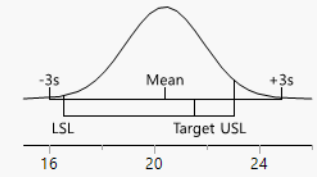
An example of a capability analysis report is shown in Figure 12.13 for Coating.jmp when the Lower Spec Limit is set as 16.5, the Target is set to 21.5, and the Upper Spec Limit is set to 23.

Figure 12.13 Capability Analysis Report for Coating.jmp

Capability Analysis

Specification	Value	Portion	% Actual
Lower Spec Limit	16.5	Below LSL	0.0000
Spec Target	21.5	Above USL	3.1250
Upper Spec Limit	23	Total Outside	3.1250

Long Term Sigma

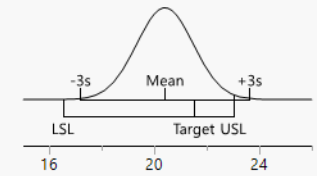


Sigma = 1.47506

Capability	Index	Lower CI	Upper CI
CP	0.734	0.552	0.916
CPK	0.588	0.402	0.775
CPM	0.271	.	.
CPL	0.881	0.632	1.126
CPU	0.588	0.400	0.772

Portion	Percent	PPM	Sigma Quality
Below LSL	0.4123	4122.7878	4.142
Above USL	3.8802	38801.970	3.265
Total Outside	4.2925	42924.758	3.218

Control Chart Sigma



Sigma = 1.07468

Capability	Index
CP	1.008
CPK	0.807
CPM	0.325
CPL	1.209
CPU	0.807

Portion	Percent	PPM	Sigma Quality
Below LSL	0.0144	143.8801	5.126
Above USL	0.7713	7712.7760	3.922
Total Outside	0.7857	7856.6561	3.916

For additional information, see "Statistical Details for Capability Analysis".

Save Sigma Saves the computed value of sigma as a column property in the process variable column in the JMP data table.

Save Limits Saves the control limits in one of the following ways:

in Column Saves control limits as a column property in the existing data table for the response variable. If the limits are constant, LCL, Avg, and UCL values for each chart type in the report are saved. This option is not available with phase charts. In addition, the option has no effect if the sample sizes are not constant for each chart.

in New Table Saves the standard deviation and mean for each chart into a new data table. If the limits are constant, the LCL, Avg, and UCL for each chart are saved as well. If there are phases, a new set of values is saved for each phase. You can use this data table to use the limits later. In the Control Chart launch window, click **Get Limits** and then select the saved data table. See the section "Saving and Retrieving Limits".

Save Summaries Creates a new data table that contains the sample label, sample sizes, the statistic being plotted, the center line, and the control limits. The specific statistics included in the table depend on the type of chart.

Alarm Script Enables you to write and run a script that indicates when the data fail special causes tests. Results can be written to the log or spoken. See "Tests" of this guide. See the *Scripting Guide* for more information about writing custom Alarm Scripts.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Chart Options for Legacy Control Charts

The red triangle menu of chart options appears when you click the icon next to the chart name. Some options are also available under **Chart Options** when you right-click the chart. Not all of the options below are available for all control chart types.

Box Plots Superimposes box plots on the subgroup means plotted in a Mean chart. The box plot shows the subgroup maximum, minimum, 75th percentile, 25th percentile, and median. Markers for subgroup means show unless you deselect the **Show Points** option. The control limits displayed apply only to the subgroup mean. The **Box Plots** option is available only for $\bar{X}$ charts. It is most appropriate for larger subgroup sample sizes (more than 10 samples in a subgroup).

Needle Connects plotted points to the center line with a vertical line segment.

Connect Points Shows or hides the line that connects the data points.

Show Points Shows or hides the points representing summary statistics. Initially, the points show. You can use this option to suppress the markers denoting subgroup means when the **Box Plots** option is in effect.

Connect Color Displays the JMP color palette for you to choose the color of the line segments used to connect points.

Center Line Color Displays the JMP color palette for you to choose the color of the line segments used to draw the center line.

Limits Color Displays the JMP color palette for you to choose the color of the line segments used in the upper and lower limits lines.

Line Width Enables you to select the width of the control lines. Options are **Thin**, **Medium**, or **Thick**.

Point Marker Enables you to select the marker used on the chart.

Show Center Line Shows or hides the center line in the control chart.

Show Control Limits Shows or hides the chart control limits and their legends.

Limits Precision Sets the decimal limit for labels.

Tests Shows a submenu that enables you to choose which tests to mark on the chart when the test is positive. Tests apply only for charts whose limits are 3σ limits. Tests 1 to 4 apply to Mean, Individual, and attribute charts. Tests 5 to 8 apply to Mean charts, Presummarize, and Individual Measurement charts only. If tests do not apply to a chart, the Tests option is dimmed. When sample sizes are unequal, the Test options are grayed out. If the samples change while the chart is open and they become equally sized, and the zone or test option is selected, the zones or tests are applied immediately and appear on the chart. These special tests are also referred to as the *Western Electric Rules*. For more information about special causes tests, see "Tests".

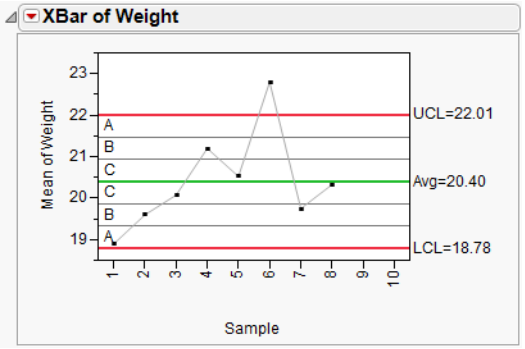
Westgard Rules Westgard rules are control rules that help you decide whether a process is in or out of control. The different tests are abbreviated with the decision rule for the particular test. See the text and chart in "Westgard Rules".

Test Beyond Limits Flags as a "*" any point that is beyond the limits. This test works on all charts with limits, regardless of the sample size being constant, and regardless of the size of *k* or the width of the limits. For example, if you had unequal sample sizes, and wanted to flag any points beyond the limits of an R chart, you could use this command.

Show Zones Shows or hides the *zone lines*. The zones are labeled A, B, and C as shown here in the Mean plot for weight in the Coating.jmp sample data. Control Chart tests use the

zone lines as boundaries. The seven zone lines are set one sigma apart, centered on the center line.

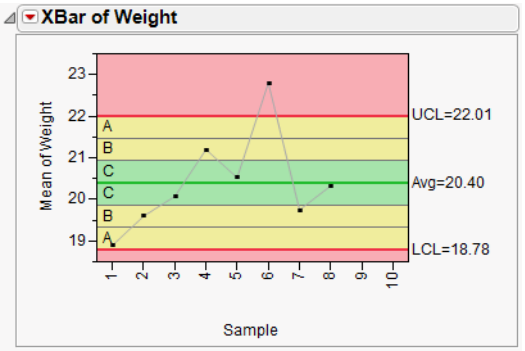
Figure 12.14 Show Zones



Shade Zones Shows or hides the default green, yellow, and red colors for the three zone areas and the area outside the zones. Green represents the area one sigma from the center line, yellow represents the area two and three sigmas from the center line, and red represents the area beyond three sigmas. Shades can be shown with or without the zone lines.

Tip: To change the colors used to shade the zones, right-click in the control chart and select Customize. In the Customize Graph window, you can specify colors for each of the three zones.

Figure 12.15 Shade Zones



OC Curve (Available when subgroup sizes are equal.) Opens a new window that contains the operating characteristic (OC) curve, using all the calculated values directly from the active control chart. See "Operating Characteristic Curves Utility".

Chart Options for V-Mask CUSUM Control Charts

Mask Color (Available only when the Show V Mask option is selected.) Enables you to select a line color for the V-mask.

Show Shift Shows or hides the shift that you entered in the launch window.

Show V Mask Shows or hides the V-mask based on the statistics that you specified in the CUSUM Control Charts launch window.

Show Parameters Shows or hides a report that summarizes the CUSUM charting parameters.

Show ARL Shows or hides the average run length (ARL) information. The average run length is the expected number of samples taken before an out-of-control condition is signaled:

- ARL (Delta), sometimes denoted ARL1, is the average run length for detecting a shift in the size of the specified Delta.
- ARL(0), sometimes denoted ARL0, is the in-control average run length for the specified parameters (Montgomery 2013).

Saving and Retrieving Limits

JMP can use previously established control limits for control charts:

- Upper and lower control limits, and a center line value.
- Parameters for computing limits such as a mean and standard deviation.

The control limits or limit parameter values must be either in a JMP data table, referred to as the *Limits Table*, or stored as a column property in the process column. When you specify the **Control Chart** command, you can retrieve the Limits Table with the **Get Limits** button on the Control Chart launch window.

Tip: To add specification limits to several columns at once, see "Manage Spec Limits Utility".

The easiest way to create a Limits Table is to save results computed by the Control Chart platform. The **Save Limits** command in the red triangle menu for each control chart automatically saves limits from the sample values. The type of data saved in the table varies according to the type of control chart in the analysis window. You can also use values from any source and create your own Limits Table.

All Limits Tables must have:

- A column of special keywords that identify each row.

- A column for each of the variables whose values are the known standard parameters or limits. This column name must be the same as the corresponding process variable name in the data table to be analyzed by the Control Chart platform.

The following table describes the limit keywords and their associated control chart for both legacy control charts and charts created with Control Chart Builder.

Table 12.1 Limits Table Keys with Appropriate Charts and Meanings

Keywords	For Charts	Meaning
_KSigma	All *except* Control Chart Builder and V-Mask CUSUM	multiples of the standard deviation of the statistics to calculate the control limits; set to missing if the limits are in terms of the alpha level
_Alpha	All *except* Control Chart Builder	Type I error probability used to calculate the control limits
_Range Span	IM, MR, MMR	number of consecutive measurements for which moving ranges are computed. Not applicable in the Control Chart Builder platform, where the range span is always equal to 2.
_Sample Size	All *except* Levey-Jennings and Presummarize	subgroup size
_Std Dev	XBar, R, S, IM, MR, G, T, V-Mask CUSUM, and Levey-Jennings	known process standard deviation
_U	C, U	known average number of nonconformities per unit
_P	NP, P	known value of average proportion nonconforming
_LCL, _UCL	XBar, IM, P, NP, C, U, G, T, and Levey-Jennings	lower and upper control limit for Mean Chart, Individual Measurement chart, or any attribute or rare event chart
_AvgR	R, MR	average range or average moving range

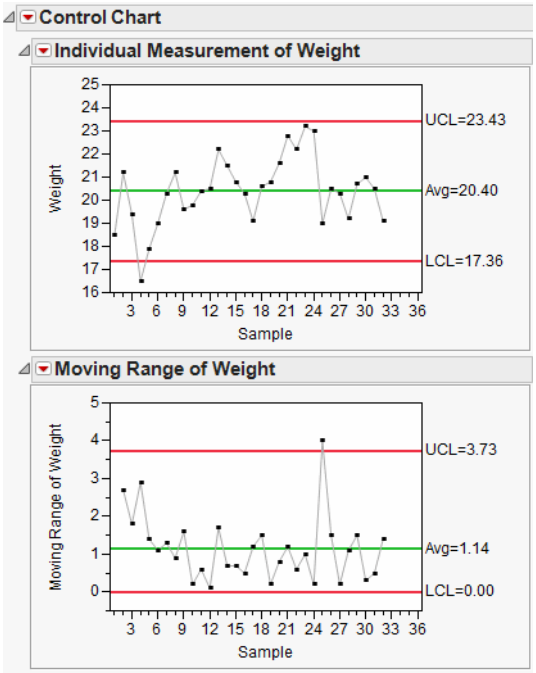
Table 12.1 Limits Table Keys with Appropriate Charts and Meanings *(Continued)*

Keywords	For Charts	Meaning
_LCLR, _UCLR	R, MR	lower control limit for R or MR chart upper control limit for R or MR chart
_AvgS, _LCLS, _UCLS	S Chart	average standard deviation, upper and lower control limits for S chart
_AvgR_PreMeans _AvgR_PreStdDev _LCLR_PreMeans _LCLR_PreStdDev _UCLR_PreMeans _UCLR_PreStdDev _Avg_PreMeans _Avg_PreStdDev _LCL_PreMeans _LCL_PreStdDev _UCL_PreMeans _UCL_PreStdDev	IM, MR	Mean, upper, and lower control limits based on pre-summarized group means or standard deviations.
_Data Units _Two Sided _Headstart _Beta _Delta	V-Mask CUSUM	specifications for V-Mask CUSUM chart

You can save limits in a new data table or as properties of the response column. When you save control limits using the **in New Table** command, the limit keywords written to the table depend on the current chart types displayed.

Figure 12.16 shows examples of control limits saved to a data table using Coating.jmp. The rows with values _Mean, _LCL, and _UCL are for the Individual Measurement chart. The values with the R suffix (_AvgR, _LCLR, and _UCLR) are for the Moving Range chart. If you create these charts again using this Limits Table, the Control Chart platform identifies the appropriate limits from keywords in the _LimitsKey column.

Figure 12.16 Example of Saving Limits in a Data Table



	_LimitsKey	Weight
1	_KSigma	3
2	_Alpha	0.0026997961
3	_Range Span	2
4	_Std Dev	1.0120139729
5	_Mean	20.396875
6	_LCL	17.360833081
7	_UCL	23.432916919
8	_AvgR	1.1419354839
9	_LCLR	0
10	_UCLR	3.7301687077

Note that values for _KSigma, _Alpha, and _Range Span can be specified in the Control Chart Launch window. JMP always looks at the values from the window first. Values specified in the window take precedence over those in an active Limits Table.

Rows with unknown keywords and rows marked with the excluded row state are ignored. Except for _Range Span, _KSigma, _Alpha, and _Sample Size, any needed values not specified are estimated from the data.

Excluded, Hidden, and Deleted Samples

The following table summarizes the effects of various conditions on samples and subgroups:

Table 12.2 Excluded, Hidden, and Deleted Samples

All rows of the sample are excluded before creating the chart.	Sample is not included in the calculation of the limits, but it appears on the graph.
Sample is excluded after creating the chart.	Sample is included in the calculation of the limits, and it appears in the graph. Nothing changes on the output by excluding a sample with the graph open.

Table 12.2 Excluded, Hidden, and Deleted Samples *(Continued)*

Sample is hidden before creating the chart.	Sample is included in the calculation of the limits, but does not appear on the graph.
Sample is hidden after creating the chart.	Sample is included in the calculation of the limits, but does not appear on the graph. The sample marker disappears from the graph, the sample label still appears on the axis, but limits remain the same.
All rows of the sample are both excluded and hidden before creating the chart.	Sample is not included in the calculation of the limits, and it does not appear on the graph.
All rows of the sample are both excluded and hidden after creating the chart.	Sample is included in the calculation of the limits, but does not appear on the graph. The sample marker disappears from the graph, the sample label still appears on the axis, but limits remain the same.
Data set is subsetted with Sample deleted before creating chart.	Sample is not included in the calculation of the limits, the axis does not include a value for the sample, and the sample marker does not appear on the graph.
Data set is subsetted with Sample deleted after creating chart.	Sample is not included in the calculation of the limits, and does not appear on the graph. The sample marker disappears from the graph, the sample label is removed from the axis, the graph shifts, and the limits change.

Some additional notes:

- Hide and Exclude operate only on the row state of the first observation in the sample. For example, if the second observation in the sample is hidden, but the first observation is not hidden, the sample still appears on the chart.

  Note: Excluded rows in Presummarize charts are excluded from calculations, regardless of which position they are within a sample.

- An exception to the exclude/hide rule: Both hidden and excluded rows are included in the count of points for Tests for Special Causes. An excluded row can be labeled with a special cause flag. A hidden point cannot be labeled. If the flag for a Tests for Special Causes is on a hidden point, it will not appear in the chart.
- Because of the specific rules in place (Table 12.2), the control charts do not support the Automatic Recalc script.

Additional Examples of the Control Chart Platform

- "Presummarize Chart Example"
- "V-Mask CUSUM Chart Example"
- "One-Sided CUSUM Chart Example"
- "UWMA Chart Example"

Presummarize Chart Example

This example uses the Coating.jmp data table.

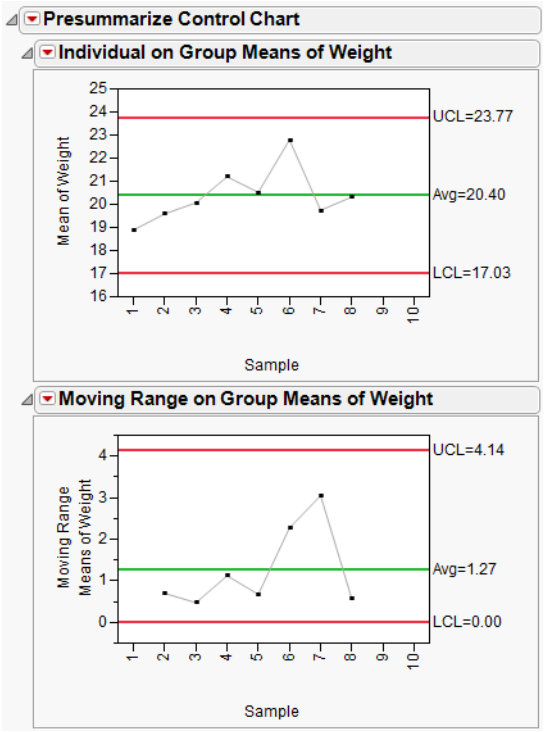
1. Select **Help > Sample Data Library** and open Quality Control/Coating.jmp.
2. Select **Analyze > Quality and Process > Legacy Control Charts > Presummarize**.
3. Select Weight and click **Process.**
4. Select Sample and click **Sample Label**.
5. Select both **Individual on Group Means** and **Moving Range on Group Means**. The **Sample Grouped by Sample Label** button is automatically selected when you choose a Sample Label variable.

   When using **Presummarize** charts, you can select either **On Group Means** options or **On Group Std Devs** options or both. Each option creates two charts (an Individual Measurement, also known as an X chart, and a Moving Range chart) if both IR chart types are selected.

   The **On Group Means** options compute each sample mean and then plot the means and create an Individual Measurement and a Moving Range chart on the means.

   The **On Group Std Devs** options compute each sample standard deviation and plot the standard deviations as individual points. Individual Measurement and Moving Range charts for the standard deviations then appear.
6. Click **OK**.

Figure 12.17 Example of Charting Presummarized Data



Although the points for XBar and S charts are the same as the Individual on Group Means and Individual on Group Std Devs charts, the limits are different because they are computed as Individual charts.

Another way to generate the presummarized charts, with the Coating.jmp data table:

1. Choose **Tables > Summary**.
2. Select Sample and click **Group**.
3. Select Weight, and then click **Statistics** > **Mean** and **Statistics** > **Std Dev**.
4. Click **OK**.
5. Select **Analyze > Quality and Process > Legacy Control Charts > IR**.
6. Select Mean(Weight) and Std Dev(Weight) and click **Process**.
7. Click **OK**.

The resulting charts match the presummarized charts.

V-Mask CUSUM Chart Example

A machine fills 8-ounce cans of two-cycle engine oil additive. The filling process is believed to be in statistical control. The process is set so that the average weight of a filled can (μ_0) is 8.10 ounces. Previous analysis shows that the standard deviation of fill weights (σ_0) is 0.05 ounces.

Subgroup samples of four cans are selected and weighed every hour for twelve hours. Each observation in the Oil1 Cusum.jmp data table contains one value of weight and its associated value of hour. The observations are sorted so that the values of hour are in increasing order.

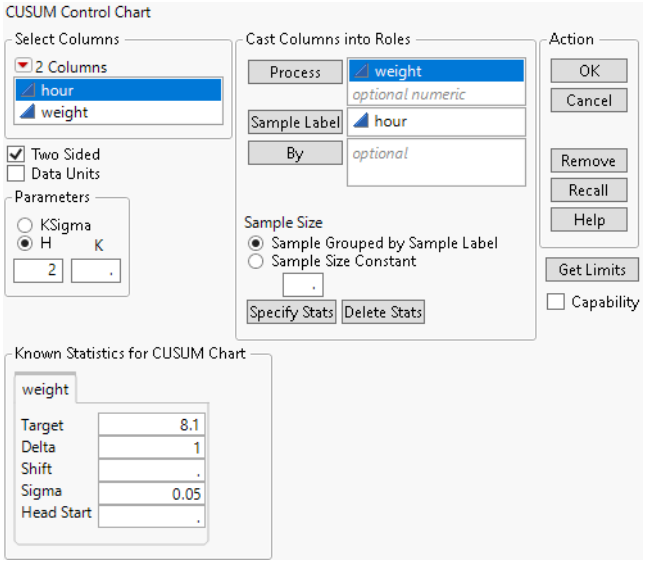
1. Select **Help > Sample Data Library** and open Quality Control/Oil1 Cusum.jmp.
2. Select **Analyze > Quality And Process > Legacy Control Charts > CUSUM**.
3. Select weight and click **Process**.
4. Select hour and click **Sample Label**.
5. Select the **Two Sided** check box if it is not already checked.
6. In the Parameters area, click the **H** button and type 2.
7. Click **Specify Stats**.
8. Type 8.1 next to **Target**.

   8.1 is the average weight in ounces of a filled can. This is the target mean.
9. Type 1 next to **Delta**.

   1 is the absolute value of the smallest shift to be detected as a multiple of the process standard deviation or of the standard error.
10. Type 0.05 next to **Sigma**.

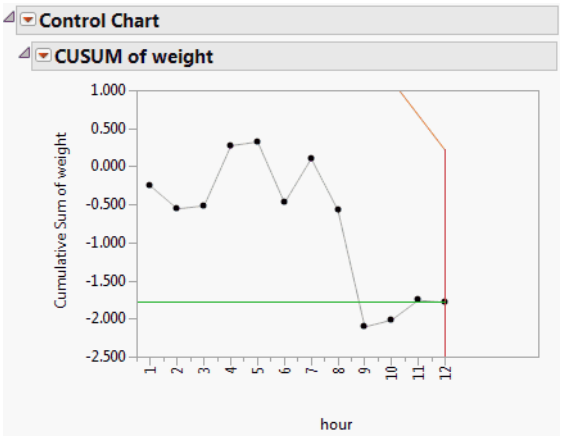
    0.05 is the known standard deviation of fill weights (σ_0) in ounces.

Figure 12.18 Completed Launch Window



11. Click **OK**.

Figure 12.19 Two-Sided CUSUM Chart for Oil1 Cusum.jmp Data



You can interpret the chart by comparing the points with the V-mask. The right edge of the V-mask is centered at the most recent point (the 12th hour). Because none of the points cross the arms of the V-mask, there is no evidence that a shift in the process has occurred. See "V-Mask CUSUM Chart Reports".

One-Sided CUSUM Chart Example

Consider the data used in "V-Mask CUSUM Chart Example", where the machine fills 8-ounce cans of engine oil. In order to cut costs, the manufacturer is now concerned about significant over-filling (and not so concerned about under-filling). Use a one-sided CUSUM chart to identify any instances of over-filling. Anything that is 0.25 ounces beyond the mean of 8.1 is considered a problem.

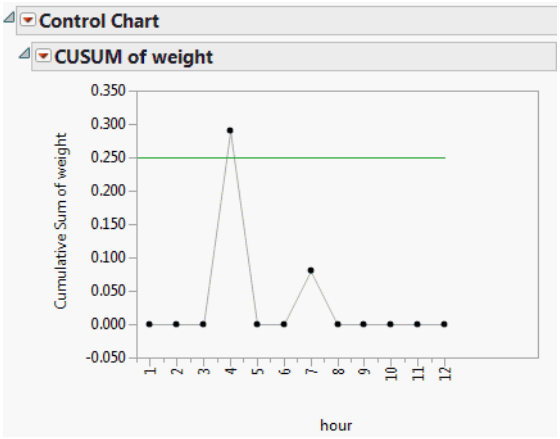
1. Select **Help > Sample Data Library** and open Quality Control/Oil1 Cusum.jmp.
2. Select **Analyze > Quality And Process > Legacy Control Charts > CUSUM**.
3. Deselect **Two Sided**.
4. Select weight and click **Process**.
5. Select hour and click **Sample Label**.
6. Click **H** and type 0.25.
7. Click **Specify Stats**.
8. Type 8.1 next to **Target**.

   8.1 is the average weight in ounces of a filled can. This is the target mean.
9. Type 1 next to **Delta**.

   1 is the absolute value of the smallest shift to be detected as a multiple of the process standard deviation or of the standard error.
10. Type 0.05 next to **Sigma**.

    0.05 is the known standard deviation of fill weights (σ_0) in ounces.
11. Click **OK**.

Figure 12.20 One-Sided CUSUM Chart for Oil1 Cusum.jmp Data



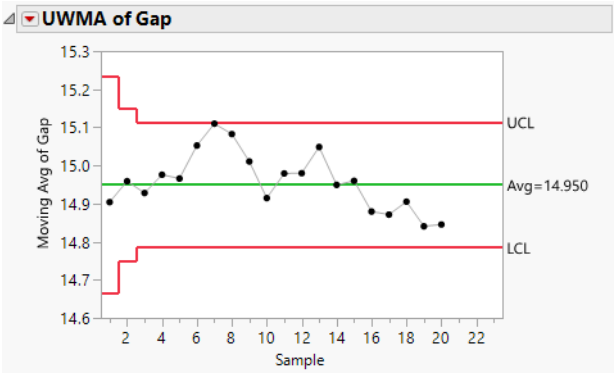
The decision interval is set at the H value that you entered (0.25). You can see that at the fourth hour, some significant over-filling occurred.

UWMA Chart Example

In the sample data table Clips1.jmp, the measure of interest is the gap between the ends of manufactured metal clips. To monitor the process for a change in the average gap, subgroup samples of five clips are selected daily. A UWMA chart with a moving average span of three is examined.

1. Select **Help > Sample Data Library** and open Quality Control/Clips1.jmp.
2. Select **Analyze > Quality and Process > Legacy Control Charts > UWMA**.
3. Select Gap and click **Process**.
4. Select Sample and click **Sample Label**.
5. Change the **Moving Average Span** to 3.
6. Click **OK**.

Figure 12.21 UWMA Charts for the Clips1 data



The point for the first day is the mean of the five subgroup sample values for that day. The plotted point for the second day is the average of subgroup sample means for the first and second days. The points for the remaining days are the average of subsample means for each day and the two previous days.

The average clip gap appears to be decreasing, but no sample point falls outside the 3σ limits.

Statistical Details for the Control Chart Platform

- "Control Limits for Median Moving Range Charts"
- "Statistical Details for Capability Analysis"
- "Statistical Details for V-Mask CUSUM Control Charts"
- "Statistical Details for Weighted Moving Average Charts"

Note: For more information about other types of charts (such as XBar and R charts, P and NP charts, and more) see the "Statistical Details for Control Chart Builder".

Control Limits for Median Moving Range Charts

Control limits for Median Moving Range charts are computed as follows:

$$LCL_{MMR} = \max(0, MMR - kd_3(n)\ \hat{\sigma}\ )$$

$$UCL_{MMR} = MMR + kd_3(n)\ \hat{\sigma}$$

where:

MMR is the median of the nonmissing moving ranges.

$\hat{\sigma}$ = MMR/0.954

$d_3(n)$ is the standard deviation of the range of n independent normally distributed variables with unit standard deviation.

Statistical Details for Capability Analysis

This section contains details about the computation of the statistics in the Capability Analysis report.

Variation Statistics

All capability analyses use the same formulas. Options differ in how sigma (σ) is computed:

Long Term Sigma Uses the overall sigma. This option is used for P_{pk} statistics, and computes sigma as follows:

$$\hat{\sigma} = \sqrt{\sum_{i=1}^{n} \frac{(x_i - \bar{x})^2}{n-1}}$$

Note: By default, the capability indices in the Long Term Sigma report use the Cp labeling that is used in the other sigma reports. To use Ppk labeling in the Long Term Sigma report, select the **File > Preferences > Platforms > Distribution > PpK Capability Labeling** preference.

Control Chart Sigma Uses a sigma that is determined by the control chart settings.

- If you specify a value for Sigma using the Specify Stats button in the control launch window, the specified value is used for computing capability indices.
- In an IR chart that uses the Moving Range (Average) option, the value for sigma is computed as follows:

$$\hat{\sigma} = \frac{\bar{R}}{d_2(n)}$$

where:

$\bar{R}$ is the average of the moving ranges.

$d_2(n)$ is the expected value of the range of n independent normally distributed variables with unit standard deviation, where n is the value of the Range Span option.

- In an IR chart that uses the Median Moving Range option, the value for sigma is computed as follows:

$$\hat{\sigma} = \frac{MMR}{d_4(n)}$$

where:

MMR is the median of the nonmissing moving ranges.

$d_4(n)$ is the median of the range of n independent normally distributed variables with unit standard deviation, where n is the value of the Range Span option.

- In an XBar chart that uses the R option, the value for sigma is computed as follows:

$$\hat{\sigma} = \frac{\frac{R_1}{d_2(n_1)} + \ldots + \frac{R_N}{d_2(n_N)}}{N}$$

where:

R_i = range of i^{th} subgroup

n_i = sample size of i^{th} subgroup

$d_2(n_i)$ = expected value of the range of n_i independent normally distributed variables with unit standard deviation

N = number of subgroups for which $n_i \geq 2$

– In an XBar chart that uses the S option, the value for sigma is computed as follows:

$$\hat{\sigma} = \frac{\frac{s_1}{c_4(n_1)} + \ldots + \frac{s_N}{c_4(n_N)}}{N}$$

where:

n_i = sample size of i^{th} subgroup

$c_4(n_i)$ = expected value of the standard deviation of n_i independent normally distributed variables with unit standard deviation

N = number of subgroups for which $n_i \geq 2$

s_i = sample standard deviation of the i^{th} subgroup

Capability Indices for Normal Distributions

This section provides details about the calculation of capability indices for normal data.

For a process characteristic with mean μ and standard deviation σ, the population-based capability indices are defined as follows:

$$Cp = \frac{USL - LSL}{6\sigma}$$

$$Cpl = \frac{\mu - LSL}{3\sigma}$$

$$Cpu = \frac{USL - \mu}{3\sigma}$$

$$Cpk = \min(Cpl, Cpu)$$

$$Cpm = \frac{\min(T - LSL, USL - T)}{3\sigma\sqrt{1 + \left(\frac{T - \mu}{\sigma}\right)^2}}$$

where:

LSL is the lower specification limit.

USL is the upper specification limit.

T is the target value.

For sample-based capability indices, the parameters are replaced by their estimates. The estimate for σ uses the method that you specified in the Capability Analysis window. See "Variation Statistics".

If either of the specification limits is missing, the capability indices containing the missing specification limit are reported as missing.

Tip: A capability index of 1.33 is often considered to be the minimum value that is acceptable. For a normal distribution, a capability index of 1.33 corresponds to an expected number of nonconforming units of about 6 per 100,000.

Confidence Intervals for Capability Indices

Note: Confidence intervals for capability indices appear only in the Long Term Sigma report.

The 100(1 - α)% confidence interval for Cp is calculated as follows:

$$\left(\hat{CP}\sqrt{\frac{\chi^2_{\alpha/2,\,n-1}}{n-1}}, \hat{CP}\sqrt{\frac{\chi^2_{1-\alpha/2,\,n-1}}{n-1}}\right)$$

where:

$\hat{CP}$ is the estimated value for Cp.

$\chi^2_{\alpha/2,\,n-1}$ is the $(\alpha/2)^{th}$ quantile of a chi-square distribution with n - 1 degrees of freedom.

n is the number of observations.

The 100(1 - α)% confidence interval for Cpk is calculated as follows:

$$\left(\hat{C}_{pk}\left[1-\Phi^{-1}{}_{1-\alpha/2}\sqrt{\frac{1}{9n\hat{C}_{pk}^2}+\frac{1}{2(n-1)}}\right],\ \hat{C}_{pk}\left[1+\Phi^{-1}{}_{1-\alpha/2}\sqrt{\frac{1}{9n\hat{C}_{pk}^2}+\frac{1}{2(n-1)}}\right]\right)$$

where:

$\hat{C}_{pk}$ is the estimated value for Cpk.

$\Phi^{-1}{}_{1-\alpha/2}$ is the $(1 - \alpha/2)^{th}$ quantile of a standard normal distribution.

n is the number of observations.

The 100(1 - α)% confidence interval for CPM is calculated as follows:

$$\left(\hat{\text{CPM}}\sqrt{\frac{\chi^2_{\alpha/2,\gamma}}{\gamma}},\ \hat{\text{CPM}}\sqrt{\frac{\chi^2_{1-\alpha/2,\gamma}}{\gamma}}\right)$$

where:

$\hat{\text{CPM}}$ is the estimated value for CPM.

$\chi^2_{\alpha/2,\gamma}$ is the $(\alpha/2)^{th}$ quantile of a chi-square distribution with γ degrees of freedom.

$$\gamma = \frac{n\left(1+\left(\frac{\bar{x}-T}{s}\right)^2\right)^2}{1+2\left(\frac{\bar{x}-T}{s}\right)^2}$$

n is the number of observations.

$\bar{x}$ is the mean of the observations.

T is the target value.

s is the long-term sigma estimate.

Note: The confidence interval for CPM is computed only when the target value is centered between the lower and upper specification limits.

Lower and upper confidence limits for CPL and CPU are computed using the method of Chou et al. (1990).

The 100(1 - α)% confidence limits for CPL (denoted by CPL_L and CPL_U) satisfy the following equations:

$$\Pr[t_{n-1}(\delta_L) \geq 3\hat{C}_{pl}\sqrt{n}] = \alpha/2 \qquad \text{where} \quad \delta_L = 3CPL_L\sqrt{n}$$

$$\Pr[t_{n-1}(\delta_U) \leq 3\hat{C}_{pl}\sqrt{n}] = \alpha/2 \qquad \text{where} \quad \delta_U = 3CPL_U\sqrt{n}$$

where:

$t_{n\text{-}1}(\delta)$ has a non-central *t*-distribution with *n* - 1 degrees of freedom and noncentrality parameter δ.

$\hat{C}_{pl}$ is the estimated value for Cpl.

The 100(1 - α)% confidence limits for CPU (denoted by CPU_L and CPU_U) satisfy the following equations:

$$\Pr[t_{n-1}(\delta_L) \geq 3\hat{C}_{pu}\sqrt{n}] = \alpha/2 \qquad \text{where} \quad \delta_L = 3CPU_L\sqrt{n}$$

$$\Pr[t_{n-1}(\delta_U) \leq 3\hat{C}_{pu}\sqrt{n}] = \alpha/2 \qquad \text{where} \quad \delta_U = 3CPU_U\sqrt{n}$$

where:

$t_{n\text{-}1}(\delta)$ has a non-central *t*-distribution with *n* - 1 degrees of freedom and noncentrality parameter δ.

$\hat{C}_{pu}$ is the estimated value for Cpu.

Capability Indices for Nonnormal Distributions

This section describes how capability indices are calculated for nonnormal distributions. These generalized capability indices are defined as follows:

$$\text{Cp} = \frac{USL - LSL}{P_{0.99865} - P_{0.00135}}$$

$$\text{Cpk} = \min(C_{pl}, C_{pu})$$

$$\text{Cpm} = \frac{\min\left(\frac{T - LSL}{P_{0.5} - P_{0.00135}}, \frac{USL - T}{P_{0.99865} - P_{0.5}}\right)}{\sqrt{1 + \left(\frac{\mu - T}{\sigma}\right)^2}}$$

$$\text{Cpl} = \frac{P_{0.5} - LSL}{P_{0.5} - P_{0.00135}}$$

$$\text{Cpu} = \frac{USL - P_{0.5}}{P_{0.99865} - P_{0.5}}$$

where:

LSL is the lower specification limit.

USL is the upper specification limit.

T is the target value.

P_α is the α*100th percentile of the fitted distribution.

For the calculation of Cpm, μ and σ are estimated using the expected value and square root of the variance of the fitted distribution. For more information about the relationship between the parameters in the Parameter Estimates report and the expected value and variance of the fitted distributions, see *Basic Analysis*.

Sigma Quality Statistics

The Sigma Quality statistics for each Portion (Below LSL, Above USL, and Total Outside) are calculated as follows:

$$\text{Sigma Quality} = \Phi^{-1}{}_{1 - \text{Pct}/100} + 1.5$$

where:

Pct is the value in the Percent column of the report.

$\Phi^{-1}{}_{1 - \text{Pct}/100}$ is the (1 - Pct/100)th quantile of a standard normal distribution.

Note: Even though the Percent Below LSL and Percent Above USL sum to the Percent Total Outside value, the Sigma Quality Below LSL and Sigma Quality Above USL values do not sum to the Sigma Quality Total Outside value. This is because calculating Sigma Quality involves finding normal distribution quantiles, and is therefore not additive.

Benchmark Z Statistics

Benchmark Z statistics are available only for capability analyses based on the normal distribution. The Benchmark Z statistics are calculated as follows:

$$\text{Z Bench} = \Phi^{-1}{}_{1 - P(LSL) - P(USL)}$$

$$\text{Z LSL} = \frac{\mu - LSL}{\sigma} = 3 * \text{Cpl}$$

$$\text{Z USL} = \frac{USL - \mu}{\sigma} = 3 * \text{Cpu}$$

where:

LSL is the lower specification limit.

USL is the upper specification limit.

μ is the sample mean.

σ is the sample standard deviation.

$\Phi^{-1}{}_{1 - P(LSL) - P(USL)}$ is the $(1 - P(LSL) - P(USL))^{th}$ quantile of a standard normal distribution.

$P(LSL) = \text{Prob}(X < LSL) = 1 - \Phi(\text{Z LSL})$.

$P(USL) = \text{Prob}(X > USL) = 1 - \Phi(\text{Z USL})$.

Φ is the standard normal cumulative distribution function.

Statistical Details for V-Mask CUSUM Control Charts

The following notation is used in these formulas:

- μ denotes the mean of the population, also referred to as the process mean or the process level.
- μ_0 denotes the target mean (or goal) for the population. Sometimes, the symbol $\overline{X}_0$ is used for μ_0. See the American Society for Quality Statistics Division (2004). You can provide μ_0 as the Target in the Known Statistics for CUSUM Chart area on the launch window.
- σ denotes the population standard deviation. $\hat{\sigma}$ denotes an estimate of σ.
- σ_0 denotes a known standard deviation. You can provide σ_0 as the Sigma in the Known Statistics for CUSUM Chart area on the launch window.
- n denotes the nominal sample size for the CUSUM chart.

- δ denotes the shift in μ to be detected, expressed as a multiple of the standard deviation. You can provide δ as the Delta in the Known Statistics for CUSUM Chart area on the launch window.
- Δ denotes the shift in μ to be detected, expressed in data units. If the sample size n is constant across subgroups, then the following computation applies:

$$\Delta = \delta\sigma_{\bar{X}} = (\delta\sigma)/\sqrt{n}$$

You can provide Δ as the Shift in the Known Statistics for CUSUM Chart area on the launch window.

Note: Some authors use the symbol D instead of Δ.

One-Sided CUSUM Charts

Positive Shifts

If the shift δ to be detected is positive, the CUSUM for the t^{th} subgroup is computed as follows:

$$S_t = \max(0, S_{t-1} + (z_t - k))$$

$t = 1, 2, \ldots, n$, where $S_0 = 0$, z_t is defined as for two-sided charts, and the parameter k, termed the *reference value*, is positive. If the parameter k is not specified in the launch window, k is set to $\delta/2$. The CUSUM S_t is referred to as an *upper cumulative sum*. S_t can be computed as follows:

$$\max\left(0, S_{t-1} + \frac{\bar{X}_i - (\mu_0 + k\sigma_{\bar{X}_t})}{\sigma_{\bar{X}_t}}\right)$$

The sequence S_t cumulates deviations in the subgroup means greater than k standard errors from μ_0. If S_t exceeds a positive value h (referred to as the *decision interval*), a shift or out-of-control condition is signaled.

Negative Shifts

If the shift to be detected is negative, the CUSUM for the t^{th} subgroup is computed as follows:

$$S_t = \max(0, S_{t-1} - (z_t + k))$$

$t = 1, 2, \ldots, n$, where $S_0 = 0$, z_t is defined as for two-sided charts, and the parameter k, termed the *reference value*, is positive. If the parameter k is not specified in the launch window, k is set to $\delta/2$. The CUSUM S_t is referred to as a *lower cumulative sum*. S_t can be computed as follows:

$$\max\left(0, S_{t-1} - \frac{\bar{X}_i - (\mu_0 - k\sigma_{\bar{X}_t})}{\sigma_{\bar{X}_t}}\right)$$

The sequence S_t cumulates the absolute value of deviations in the subgroup means less than k standard errors from μ_0. If S_t exceeds a positive value h (referred to as the *decision interval*), a shift or out-of-control condition is signaled.

Note that S_t is always positive and h is always positive, regardless of whether δ is positive or negative. For charts designed to detect a negative shift, some authors define a reflected version of S_t for which a shift is signaled when S_t is less than a negative limit.

Lucas and Crosier (1982) describe the properties of a fast initial response (FIR) feature for CUSUM charts in which the initial CUSUM S_0 is set to a "head start" value. Average run length calculations given by them show that the FIR feature has little effect when the process is in control and that it leads to a faster response to an initial out-of-control condition than a standard CUSUM chart. You can provide a Head Start value in the Known Statistics for CUSUM Chart area on the launch window.

Constant Sample Sizes

When the subgroup sample sizes are constant (= n), it might be preferable to compute CUSUMs that are scaled in the same units as the data. CUSUMs are then computed as follows:

$$S_t = \max(0, S_{t-1} + (\bar{X}_t - (\mu_0 + k\sigma/\sqrt{n})))$$

where $\delta > 0$

$$S_t = \max(0, S_{t-1} - (\bar{X}_t - (\mu_0 - k\sigma/\sqrt{n})))$$

where $\delta < 0$. In either case, the parameter k is rescaled to $k' = k\sigma/\sqrt{n}$. If the parameter k is not specified in the launch window, k' is set to $\delta/2$. A shift is signaled if S_t exceeds $h' = h\sigma/\sqrt{n}$. Some authors use the symbol H for h'.

Two-Sided CUSUM Charts

If the CUSUM chart is two-sided, the cumulative sum S_t plotted for the t^{th} subgroup is defined as follows:

$$S_t = S_{t-1} + z_t$$

$t = 1, 2, \ldots, n$. Here $S_0=0$, and the term z_t is calculated as follows:

$$z_t = (\bar{X}_t - \mu_0)/(\sigma/\sqrt{n_t})$$

where $\bar{X}_t$ is the t^{th} subgroup average, and n_t is the t^{th} subgroup sample size. If the subgroup samples consist of individual measurements x_t, the term z_t simplifies to the following computation:

$$z_t = (x_t - \mu_0)/\sigma$$

The first equation can be rewritten as follows:

$$S_t = \sum_{i=1}^{t} z_i = \sum_{i=1}^{t} (\bar{X}_i - \mu_0)/\sigma_{\bar{X}_i}$$

where the sequence S_t cumulates standardized deviations of the subgroup averages from the target mean μ_0.

In many applications, the subgroup sample sizes n_i are constant ($n_i = n$), and the equation for S_t can be simplified:

$$S_t = (1/\sigma_{\bar{X}}) \sum_{i=1}^{t} (\bar{X}_i - \mu_0) = (\sqrt{n}/\sigma) \sum_{i=1}^{t} (\bar{X}_i - \mu_0)$$

In some applications, it might be preferable to compute S_t as follows:

$$S_t = \sum_{i=1}^{t} (\bar{X}_i - \mu_0)$$

which is scaled in the same units as the data. In this case, the procedure rescales the V-mask parameters h and k to $h' = h\sigma/\sqrt{n}$ and $k' = k\sigma/\sqrt{n}$, respectively. Some authors use the symbols F for k' and H for h'.

If the process is in control and the mean μ is at or near the target μ_0, the random walk model applies. Therefore, the points might wander away from zero, but they will not exhibit a large trend since positive and negative displacements from μ_0 tend to cancel each other. If μ shifts in the positive direction, the points exhibit an upward trend, and if μ shifts in the negative direction, the points exhibit a downward trend.

Statistical Details for Weighted Moving Average Charts

Control Limits for UWMA Charts

Control limits for UWMA charts are computed for each subgroup i as follows:

$$\text{LCL}_i = \mu_0 - k\frac{\hat{\sigma}}{min(i,w)}\sqrt{\frac{1}{n_i} + \frac{1}{n_{i-1}} + \ldots + \frac{1}{n_{1+max(i-w,0)}}}$$

$$\text{UCL}_i = \mu_0 + k\frac{\hat{\sigma}}{min(i,w)}\sqrt{\frac{1}{n_i} + \frac{1}{n_{i-1}} + \ldots + \frac{1}{n_{1+max(i-w,0)}}}$$

where:

w is the span parameter (number of terms in moving average)

n_i is the sample size of the i^{th} subgroup

k is the number of standard deviations

μ_0 is the weighted average of the subgroup means

$\hat{\sigma}$ is the estimated process standard deviation

Control Limits for EWMA Charts

Control limits for EWMA charts are computed as follows:

$$\text{LCL} = \mu_0 - k\hat{\sigma}r\sqrt{\sum_{j=0}^{i-1}\frac{(1-r)^{2j}}{n_{i-j}}}$$

$$\text{UCL} = \mu_0 + k\hat{\sigma}r\sqrt{\sum_{j=0}^{i-1}\frac{(1-r)^{2j}}{n_{i-j}}}$$

where:

r is the EWMA weight parameter ($0 < r \leq 1$)

x_{ij} is the j^{th} measurement in the i^{th} subgroup, with $j = 1, 2, 3, \ldots, n_i$

n_i is the sample size of the i^{th} subgroup

k is the number of standard deviations

μ_0 is the weighted average of the subgroup means

$\hat{\sigma}$ is the estimated process standard deviation

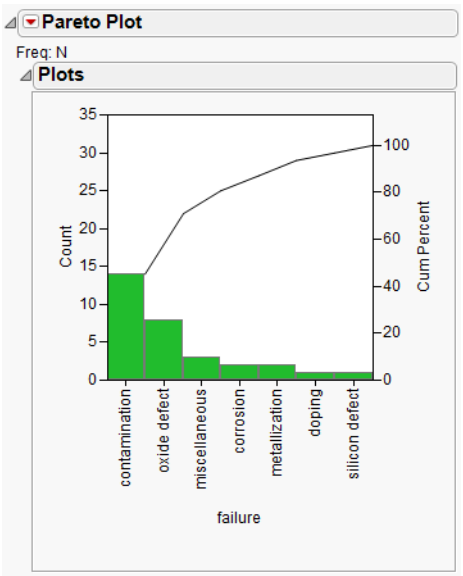
Chapter 13

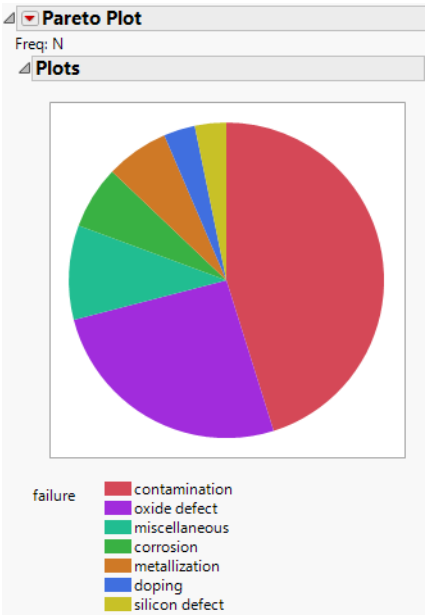
Pareto Plots

Focus Improvement Efforts on the Vital Few

Improve the statistical quality of your process or operation using Pareto plots. A Pareto plot is a chart that shows severity (frequency) of problems in a quality-related process or operation. Pareto plots help you decide which problems to solve first by highlighting the frequency and severity of problems.

Figure 13.1 Pareto Plot Examples





Contents

Overview of the Pareto Plot Platform 375
Example of the Pareto Plot Platform 375
Launch the Pareto Plot Platform 378
The Pareto Plot Report 379
Pareto Plot Platform Options 380
Causes Options 382
Additional Examples of the Pareto Plot Platform 382
Threshold of Combined Causes Example 383
Using a Constant Size across Groups Example 384
Using a Non-Constant Sample Size across Groups Example 386
One-Way Comparative Pareto Plot Example 387
Two-Way Comparative Pareto Plot Example 389
Statistical Details for the Pareto Plot Platform 390
Likelihood Ratio Chi-Square Test 390

Overview of the Pareto Plot Platform

The Pareto Plot platform produces charts to display the relative frequency or severity of problems in a quality-related process or operation. The Pareto plot is displayed initially as a bar chart that shows the classification of problems arranged in decreasing order. The column whose values are the cause of a problem is assigned the *Y* role and is called the *process variable*.

You can also generate a comparative Pareto plot, which combines two or more Pareto plots for the same process variable. The single display shows plots for each value in a column assigned the *X* role, or combination of levels from two *X* variables. Columns assigned the *X* role are called *classification variables*.

The Pareto plot can chart a single *Y* (process) variable with no *X* classification variables, with a single *X*, or with two *X* variables. The Pareto function does not distinguish between numeric and character variables or between modeling types. You can switch between a bar chart and a pie chart. All values are treated as discrete, and bars or wedges represent either counts or percentages.

Example of the Pareto Plot Platform

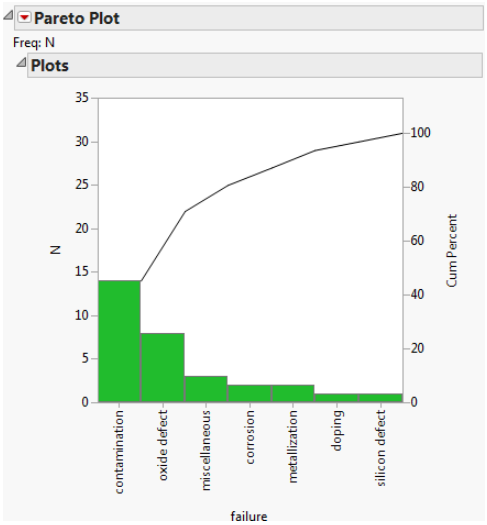
This example uses the Failure.jmp sample data table, which contains failure data and a frequency column. It lists causes of failure during the fabrication of integrated circuits and the number of times each type of defect occurred. From the analysis, you can determine which factors contribute most toward process failure.

1. Select **Help > Sample Data Library** and open Quality Control/Failure.jmp.
2. Select **Analyze > Quality and Process > Pareto Plot**.
3. Select failure and click **Y, Cause**.

   This column lists the causes of failure. It is the variable that you want to inspect.
4. Select N and click **Freq.**

   This column list the number of times that each type of failure occurred.
5. Click **OK.**

Figure 13.2 Pareto Plot Report Window



The left axis represents the count of failures, and the right axis represents the percent of failures in each category. The bars are in decreasing order with the most frequently occurring failure to the left. The curve indicates the cumulative failures from left to right.

6. Click the Pareto Plot red triangle and select **Label Cum Percent Points**.

   Note that Contamination accounts for approximately 45% of the failures. The point above the Oxide Defect bar shows that Contamination and Oxide Defect together account for approximately 71% of the failures.

7. Click the Pareto Plot red triangle and deselect **Label Cum Percent Points** and **Show Cum Percent Curve**.
8. Click the label for the y-axis labeled **N** and rename it **Count**.
9. Double-click the y-axis to display the **Y Axis Settings** window.
   - In the **Maximum** field, type 15.
   - In the **Increment** field, type 2.
   - In the **Axis Label Row** panel, select **Grid Lines** for the **Major** grid line.
   - Click **OK**.
10. Click the Pareto Plot red triangle and select **Category Legend**.

Figure 13.3 Pareto Plot with Display Options

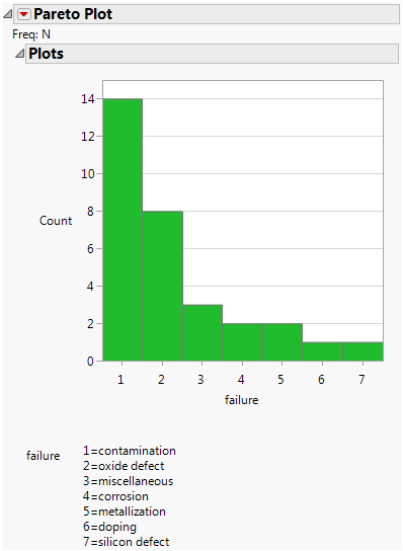
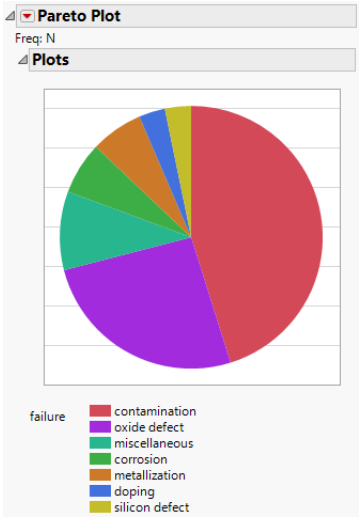


Figure 13.3 shows the counts of different types of failures and has a category legend. The vertical count axis is rescaled and has grid lines at the major tick marks.

11. To view the data as a pie chart, click the Pareto Plot red triangle and select **Pie Chart**.

Figure 13.4 Pareto Plot as a Pie Chart

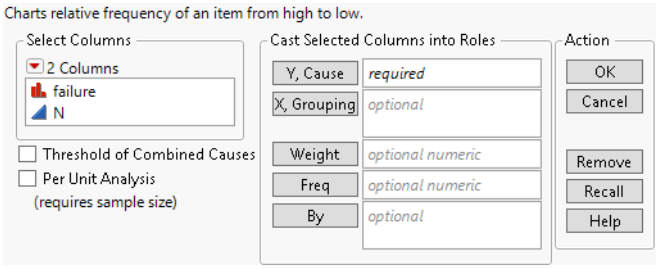


Contamination and Oxide Defect clearly represent the majority of the failures.

Launch the Pareto Plot Platform

Launch the Pareto Plot platform by selecting **Analyze > Quality and Process > Pareto Plot**.

Figure 13.5 The Pareto Plot Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Pareto Plot launch window contains the following options:

Y, Cause Identifies the column whose values are the cause of a problem. It is called the process variable and is the variable that you want to inspect.

X, Grouping Identifies the grouping factor. The grouping variable produces one Pareto plot window with side-by-side plots for each value. You can have no grouping variable, one grouping variable (see "One-Way Comparative Pareto Plot Example"), or two grouping variables (see "Two-Way Comparative Pareto Plot Example").

Weight Assigns a variable to give the observations different weights.

Freq Identifies the column whose values hold the frequencies.

By Identifies a variable to produce a separate analysis for each value that appears in the column.

Threshold of Combined Causes Enables you to specify a threshold for combining causes by specifying a minimum rate or count. Select the option and then select **Tail %** or **Count** and enter the threshold value. The Tail percent option combines smaller count groups against the percentage specified of the total (combined small groups count/total group count). The Count option enables you to specify a specific count threshold. For an example, see "Threshold of Combined Causes Example".

Per Unit Analysis Enables you to compare defect rates across groups. JMP calculates the defect rate as well as 95% confidence intervals of the defect rate. Select the option and then select **Constant** or **Value in Freq Column** and enter the sample size value or cause code, respectively. The Constant option enables you to specify a constant sample size on the

launch window. The Value In Freq Column option enables you to specify a unique sample size for a group through a special cause code to designate the rows as cause rows.

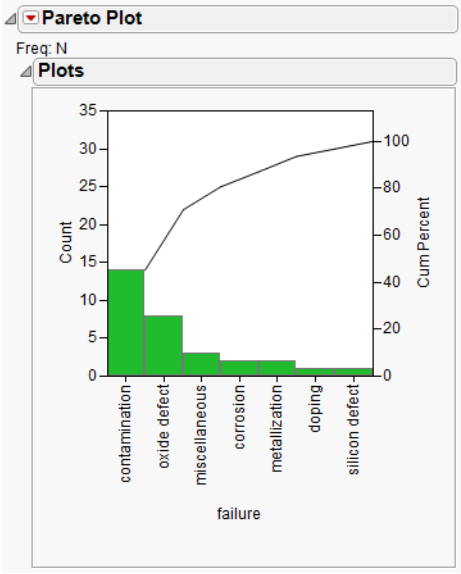
Although causes are allowed to be combined in Pareto plots, the calculations for these analyses do not change correspondingly.

For examples, see "Using a Constant Size across Groups Example" and "Using a Non-Constant Sample Size across Groups Example".

The Pareto Plot Report

The Pareto plot combines a bar chart displaying percentages of variables in the data with a line graph showing cumulative percentages of the variables.

Figure 13.6 Pareto Plot Example



The Pareto plot can chart a single *Y* (process) variable with no *X* classification variables, with a single *X*, or with two *X* variables. The Pareto plot does not distinguish between numeric and character variables or between modeling types. All values are treated as discrete, and bars represent either counts or percentages. The following list describes the arrangement of the Pareto plot:

- A *Y* variable with no *X* classification variables produces a single chart with a bar for each value of the *Y* variable. For an example, see "Example of the Pareto Plot Platform".
- A *Y* variable with one *X* classification variable produces a row of Pareto plots. There is a plot for each level of the *X* variable with bars for each *Y* level. These plots are referred to as

the *cells* of a comparative Pareto plot. There is a cell for each level of the *X* (classification) variable. Because there is only one *X* variable, this is called a *one-way comparative Pareto plot*. For an example, see "One-Way Comparative Pareto Plot Example".

- A *Y* variable with two *X* variables produces rows and columns of Pareto plots. There is a row for each level of the first *X* variable and a column for each level of the second *X* variable. Because there are two *X* variables, this is called a *two-way comparative Pareto plot*. The rows have a Pareto plot for each value of the first *X* variable, as described previously. The upper left cell is called the *key cell*. Its bars are arranged in descending order. The bars in the other cells are in the same order as the key cell. You can reorder the rows and columns of cells. The cell that moves to the upper left corner becomes the new key cell and the bars in all other cells rearrange accordingly. For an example, see "Two-Way Comparative Pareto Plot Example".
- Each bar is the color for which the rows for that *Y* level are assigned in the associated data table. Otherwise, a single color is used for all of the bars whose *Y* levels do not have rows with an assigned color. If the rows for a *Y* level have different colors, the bar for that *Y* level is the color of the first row for that *Y* level in the data table.

You can change the type of scale and arrangement of bars and convert the bars into a pie chart using the options in the Pareto Plot red triangle menu. See "Pareto Plot Platform Options".

Pareto Plot Platform Options

The Pareto Plot red triangle menu contains options that customize the appearance of the plots. It also has options in the **Causes** submenu that affect individual bars within a Pareto plot. The following commands affect the appearance of the Pareto plot as a whole:

Percent Scale Shows or hides the count and percent left vertical axis display.

N Legend Shows or hides the total sample size in the plot area.

Category Legend Shows or hides labeled bars and a separate category legend.

Pie Chart Shows or hides the bar chart and pie chart representation.

Reorder Horizontal, Reorder Vertical Reorders grouped Pareto plots when there is one or more grouping variables.

Ungroup Plots Splits up a group of Pareto charts into separate plots.

Count Analysis Performs defect per unit analyses. Enables you to compare defect rates and perform ratio tests across and within groups:

Per Unit Rates Compares defect rates across groups. If a sample size is specified, Defects Per Unit (DPU) and Parts Per Million (PPM) columns are added to the report.

Test Rate Within Groups Performs a likelihood ratio Chi-square test to determine whether the rates across causes are the same within a group. See "Statistical Details for the Pareto Plot Platform".

Test Rates Across Groups Performs a likelihood ratio Chi-square test to determine whether the rate for each cause is the same across groups. See "Statistical Details for the Pareto Plot Platform".

Show Cum Percent Curve Shows or hides the cumulative percent curve above the bars and the cumulative percent axis on the vertical right axis.

Show Cum Percent Axis Shows or hides the cumulative percent axis on the vertical right axis.

Show Cum Percent Points Shows or hides the points on the cumulative percent curve.

Label Cum Percent Points Shows or hides the labels on the points on the cumulative curve.

Cum Percent Curve Color Changes the color of the cumulative percent curve.

Causes Has options that affect one or more individual chart bars. See "Causes Options", for a description of these options.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

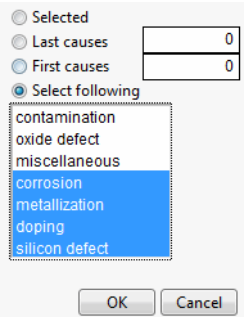
Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Causes Options

You can highlight a bar by clicking on it. Use Control-click to select multiple bars that are not contiguous. When you select bars, you can access the commands on the red triangle menu that affect Pareto plot bars. They are found on the **Causes** submenu on the red triangle menu. These options are also available with a right-click anywhere in the plot area. The following options apply to highlighted bars instead of to the chart as a whole:

Combine Causes Combines selected (highlighted) bars. You can select either **Selected**, **Last Causes**, **First Causes** or select from a list of variables.

Figure 13.7 Combine Causes Window



Separate Causes Separates selected bars into their original component bars.

Move to First Moves one or more highlighted bars to the left (first) position.

Move to Last Moves one or more highlighted bars to the right (last) position.

Colors Shows the colors palette for coloring one or more highlighted bars.

Markers Shows the markers palette for assigning a marker to the points on the cumulative percent curve, when the **Show Cum Percent Points** command is in effect.

Label Displays the bar value at the top of all highlighted bars.

Additional Examples of the Pareto Plot Platform

- "Threshold of Combined Causes Example"
- "Using a Constant Size across Groups Example"
- "Using a Non-Constant Sample Size across Groups Example"
- "One-Way Comparative Pareto Plot Example"
- "Two-Way Comparative Pareto Plot Example"

Threshold of Combined Causes Example

This example uses the Failure.jmp sample data table, which contains failure data and a frequency column. It lists causes of failure during the fabrication of integrated circuits and the number of times each type of defect occurred. A threshold value of 2 is specified for this example.

1. Select **Help > Sample Data Library** and open Quality Control/Failure.jmp.
2. Select **Analyze** > **Quality and Process > Pareto Plot**.
3. Select failure and click **Y, Cause**.
4. Select N and click **Freq.**
5. Select **Threshold of Combined Causes** and then select **Count.**
6. Enter 2 as the threshold value**.**
7. Click **OK.**

Figure 13.8 Pareto Plot with a Threshold Count of 2

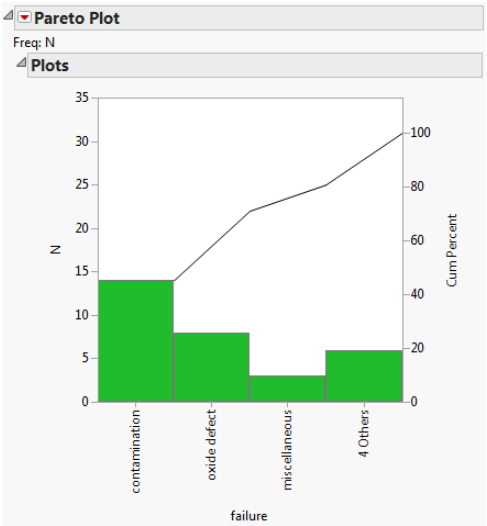
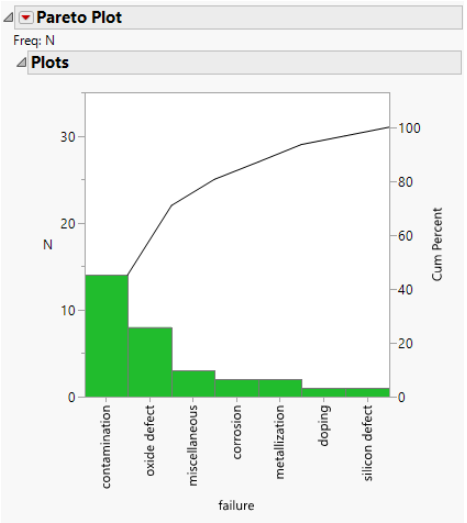


Figure 13.8 displays the plot after specifying a count of 2. All causes with counts 2 or fewer are combined into the final bar labeled 4 Others.

8. To separate the combined bars into original categories as shown in Figure 13.9, select **Causes** > **Separate Causes**.

Figure 13.9 Pareto Plot with Separated Causes

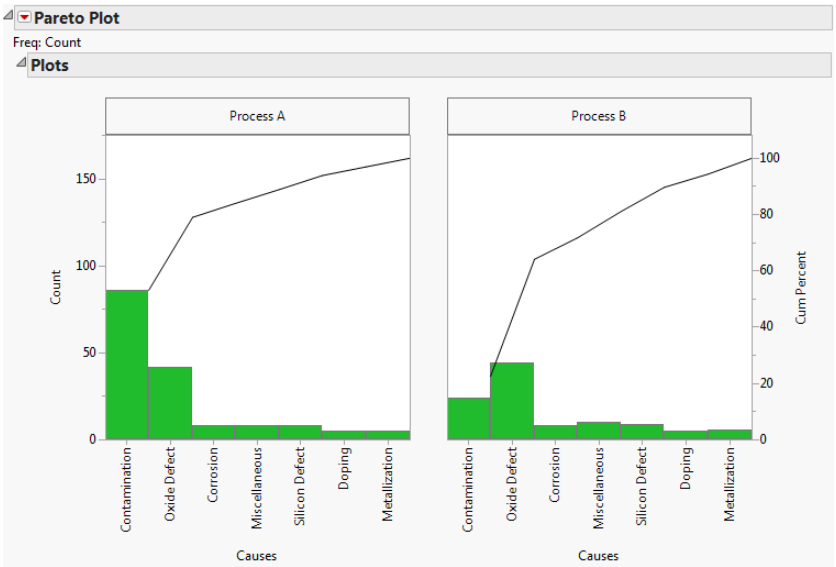


Using a Constant Size across Groups Example

This example uses the Failures.jmp sample data table, which contains failure data and a frequency column. It lists causes of failure during the fabrication of integrated circuits and the number of times each type of defect occurred for two processes. A constant sample size of 1000 is specified for this example.

1. Select **Help > Sample Data Library** and open Quality Control/Failures.jmp.
2. Select **Analyze** > **Quality and Process > Pareto Plot**.
3. Select Causes and click **Y, Cause**.
4. Select Process and click **X, Grouping.**
5. Select Count and click **Freq.**
6. Select **Per Unit Analysis** and then select **Constant.**
7. Enter 1000 in **Sample Size.**
8. Click **OK.**

Figure 13.10 Pareto Plot Report Window



Process A indicates Contamination as the top failure while Process B indicates Oxide Defect as the leading failure.

9. Click the Pareto Plot red triangle and select **Count Analysis** > **Test Rates Across Groups**.

Figure 13.11 Test Rates across Groups Results

Test Rate Across Groups

Test rate across group: Process

Cause	DPU Diff	-.8 -.6 -.4 -.2 0 .2 .4 .6 .8	Std Error	ChiSquare	DF	Prob>ChiSq
Contamination	0.0620		0.0105	37.0810	1	<.0001*
Oxide Defect	-0.0020		0.0093	0.0465	1	0.8292
Corrosion	0.0000		0.0040	0.0000	1	1.0000
Miscellaneous	-0.0020		0.0042	0.2227	1	0.6370
Silicon Defect	-0.0010		0.0041	0.0589	1	0.8083
Doping	0.0000		0.0032	0.0000	1	1.0000
Metallization	-0.0010		0.0033	0.0910	1	0.7629
Pooled Total	0.0080		0.0023	11.7882	1	0.0006*

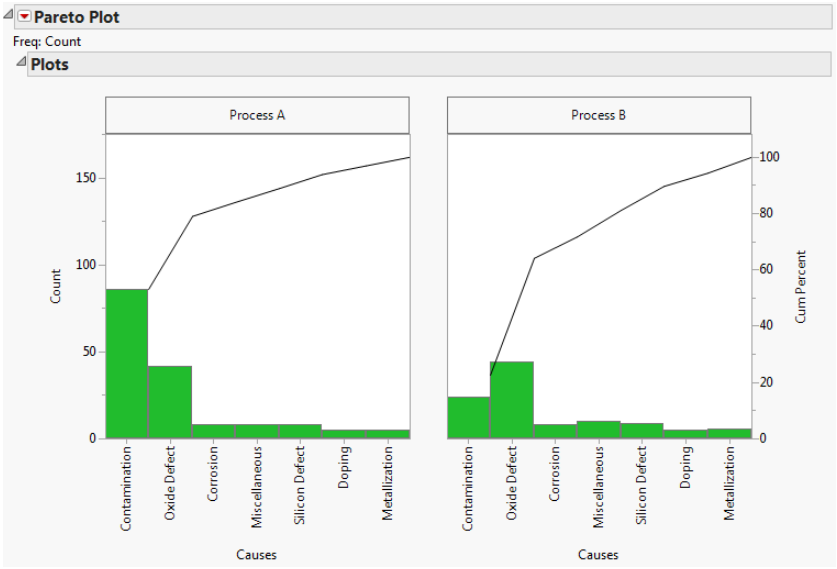
Note that the DPU for Contamination across groups (Process A and B) is around 0.06.

Using a Non-Constant Sample Size across Groups Example

This example uses the Failuressize.jmp sample data table, which contains failure data and a frequency column. It lists causes of failure during the fabrication of integrated circuits and the number of times each type of defect occurred for two processes. Among the other causes (Oxide Defect, Silicon Defect, and so on) is a cause labeled *size*. Specifying size as the cause code designates the rows as size rows.

1. Select **Help > Sample Data Library** and open Quality Control/Failuressize.jmp.
2. Select **Analyze** > **Quality and Process > Pareto Plot**.
3. Select Causes and click **Y, Cause**.
4. Select Process and click **X, Grouping.**
5. Select Count and click **Freq.**
6. Select **Per Unit Analysis** and then select **Value in Freq Column.**
7. Enter *size* in **Cause Code.**
8. Click **OK.**

Figure 13.12 Pareto Plot Report Window



9. Click the Pareto Plot red triangle and select **Count Analysis** > **Per Unit Rates** and **Count Analysis** > **Test Rates Across Groups**.

Figure 13.13 Per Unit Rates and Test Rates across Groups Results

Per Unit Rates

Process	Cause	Count	DPU	PPM	Lower 95%	Upper 95%
Process A	Contamination	86	0.8515	851485.15	0.6811	1.0516
	Oxide Defect	42	0.4158	415841.58	0.2997	0.5621
	Corrosion	8	0.0792	79207.92	0.0342	0.1561
	Miscellaneous	8	0.0792	79207.92	0.0342	0.1561
	Silicon Defect	8	0.0792	79207.92	0.0342	0.1561
	Doping	5	0.0495	49504.95	0.0161	0.1155
	Metallization	5	0.0495	49504.95	0.0161	0.1155
	Pooled Total	162	0.2291	229137.20	0.1952	0.2673
	size	101				
Process B	Contamination	24	0.1655	165517.24	0.1061	0.2463
	Oxide Defect	44	0.3034	303448.28	0.2205	0.4074
	Corrosion	8	0.0552	55172.41	0.0238	0.1087
	Miscellaneous	10	0.0690	68965.52	0.0331	0.1268
	Silicon Defect	9	0.0621	62068.97	0.0284	0.1178
	Doping	5	0.0345	34482.76	0.0112	0.0805
	Metallization	6	0.0414	41379.31	0.0152	0.0901
	Pooled Total	106	0.1044	104433.50	0.0855	0.1263
	size	145				

Test Rate Across Groups

Test rate across group: Process

Cause	DPU Diff	-.8 -.6 -.4 -.2 0 .2 .4 .6 .8	Std Error	ChiSquare	DF	Prob>ChiSq
Contamination	0.6860		0.0978	63.0776	1	<.0001*
Oxide Defect	0.1124		0.0788	2.1195	1	0.1454
Corrosion	0.0240		0.0341	0.5202	1	0.4707
Miscellaneous	0.0102		0.0355	0.0847	1	0.7710
Silicon Defect	0.0171		0.0348	0.2500	1	0.6171
Doping	0.0150		0.0270	0.3251	1	0.5685
Metallization	0.0081		0.0278	0.0871	1	0.7679
Pooled Total	0.1247		0.0207	40.7524	1	<.0001*

Note that the sample size of 101 is used to calculate the DPU for the causes in group A. However, the sample size of 145 is used to calculate the DPU for the causes in group B.

If there are two group variables (say, Day and Process), Per Unit Rates lists DPU or rates for every combination of Day and Process for each cause. However, Test Rate Across Groups only tests overall differences between groups.

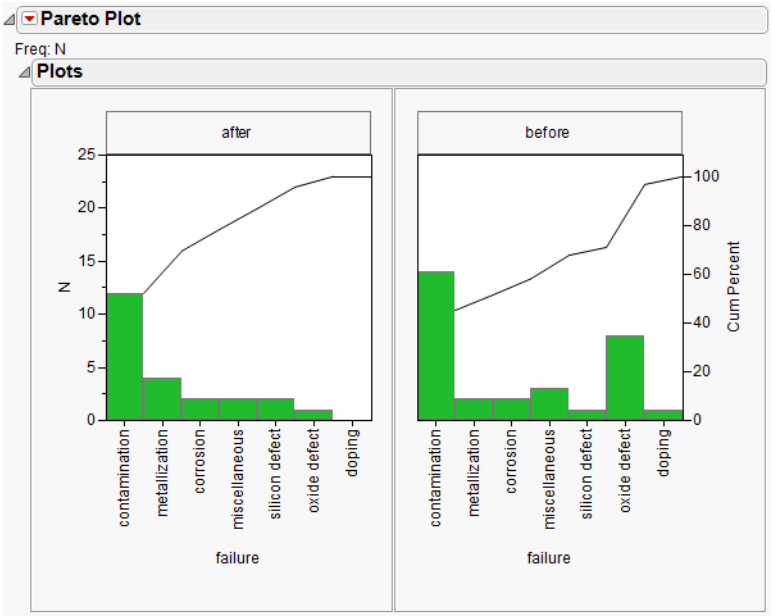
One-Way Comparative Pareto Plot Example

This example uses the Failure2.jmp sample data table. This table records failures in a sample of capacitors manufactured before cleaning a tube in the diffusion furnace and in a sample manufactured after cleaning the furnace. For each type of failure, the variable clean identifies the samples with the values “before” or “after.”

1. Select **Help > Sample Data Library** and open Quality Control/Failure2.jmp.
2. Select **Analyze** > **Quality and Process > Pareto Plot**.
3. Select failure and click **Y, Cause**.
4. Select clean and click **X, Grouping.**
5. Select N and click **Freq.**
6. Click **OK.**

Figure 13.14 displays the side-by-side plots for each value of the variable, clean.

Figure 13.14 One-way Comparative Pareto Plot

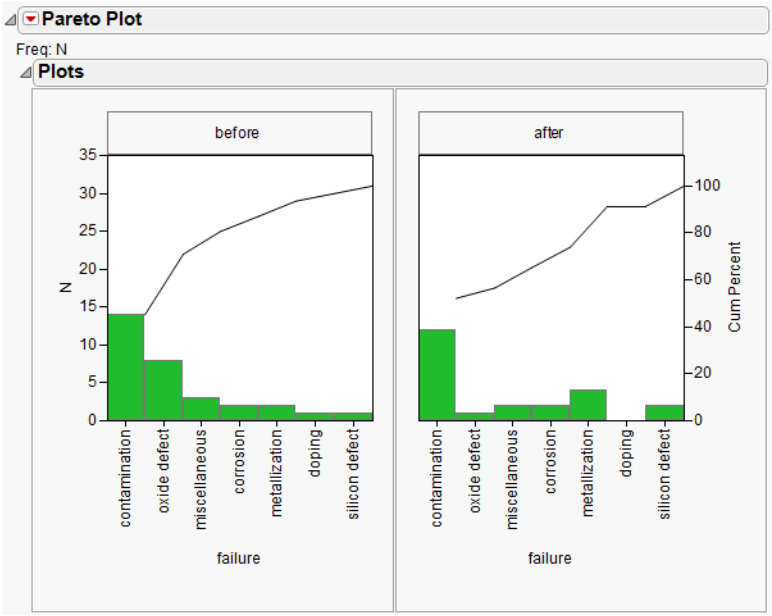


The horizontal and vertical axes are scaled identically for both plots. The bars in the first plot are in descending order of the *y*-axis values and determine the order for all cells.

7. Rearrange the order of the plots by clicking the title (*after*) in the first tile and dragging it to the title of the next tile (*before*).

   A comparison of these two plots shows a reduction in oxide defects after cleaning. However, the plots are easier to interpret when presented as the before-and-after plot shown in Figure 13.15. Note that the order of the causes changes to reflect the order based on the first cell.

Figure 13.15 One-way Comparative Pareto Plot with Reordered Cells



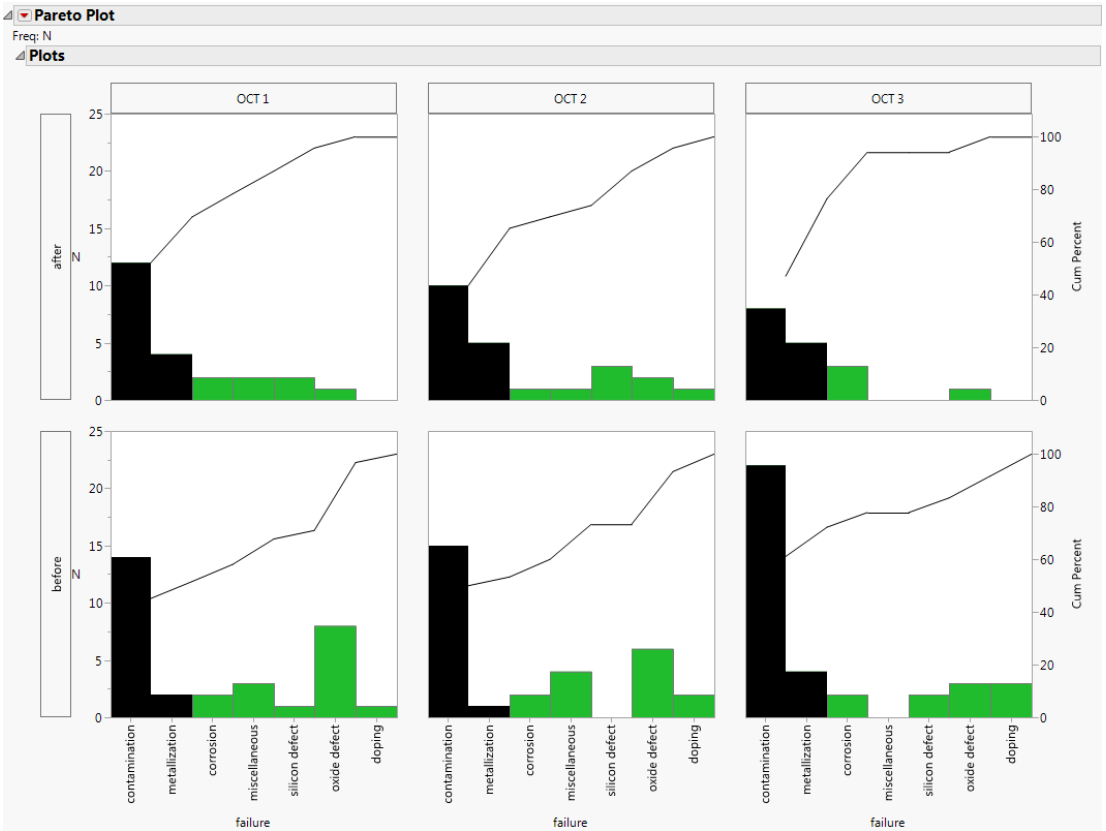
Two-Way Comparative Pareto Plot Example

This example uses the Failure3.jmp sample data table. The data monitors production samples before and after a furnace cleaning for three days for a capacitor manufacturing process. The data table has a column called date with values OCT 1, OCT 2, and OCT 3.

1. Select **Help > Sample Data Library** and open Quality Control/Failure3.jmp.
2. Select **Analyze** > **Quality and Process > Pareto Plot**.
3. Select failure and click **Y, Cause**.
4. Select clean and date and click **X, Grouping.**
5. Select N and click **Freq.**
6. Click **OK.**

   Figure 13.16 displays the Pareto plot with a two-way layout of plots that show each level of both *X* variables. The upper left cell is called the *key cell*. Its bars are arranged in descending order. The bars in the other cells are in the same order as the key cell.
7. Click Contamination and Metallization in the key cell and the bars for the corresponding categories highlight in all other cells.

Figure 13.16 Two-way Comparative Pareto Plot



The Pareto plot illustrates highlighting the *vital few*. In each cell of the two-way comparative plot, the bars representing the two most frequently occurring problems are selected. Contamination and Metallization are the two vital categories in all cells. After furnace cleaning, Contamination is less of a problem.

Statistical Details for the Pareto Plot Platform

Likelihood Ratio Chi-Square Test

Notation

The likelihood ratio Chi-square test statistic computed in the Pareto Plot platform uses the following notation:

- n_{ij} is the count for Cause i in Group j.

- E_j is the expected count for Group j. This is the mean count of each group, across causes.
- E_i is the expected count for Cause i. This is the mean count of each cause, across groups.

Likelihood Ratio Chi-Square Test Statistic within Groups

$$G_j^2 = 2 \sum_{i=1}^{K} n_{ij} \ln(n_{ij} / E_j)$$

Likelihood Ratio Chi-Square Test Statistic across Groups

$$G_i^2 = 2 \sum_{j=1}^{J} n_{ij} \ln(n_{ij} / E_i)$$

Chapter 14

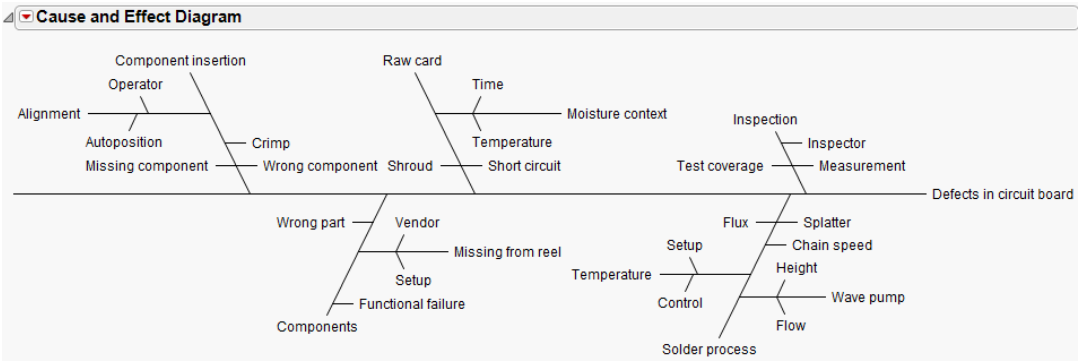
Cause-and-Effect Diagrams

Identify Root Causes

Use the Diagram platform to construct cause-and-effect diagrams, also known as *Ishikawa charts* or *fishbone charts*. Use these diagrams to:

- Organize the causes of an effect (sources of a problem)
- Brainstorm
- Identify variables in preparation for further experimentation

Figure 14.1 Example of a Cause-and-Effect Diagram



Contents

Overview of Cause-and-Effect Diagrams 395
Example of a Cause-and-Effect Diagram 395
Prepare the Data 396
Launch the Diagram Platform 396
The Cause-and-Effect Diagram 397
Right-Click Menus 397
Cause and Effect Diagram Menu Options 400
Save the Diagram 401
Save the Diagram as a Data Table 401
Save the Diagram as a Journal 401
Save the Diagram as a Script 402

Overview of Cause-and-Effect Diagrams

Use the Diagram platform to construct cause-and-effect diagrams, also known as *Ishikawa charts* or *fishbone charts*. Use these diagrams to:

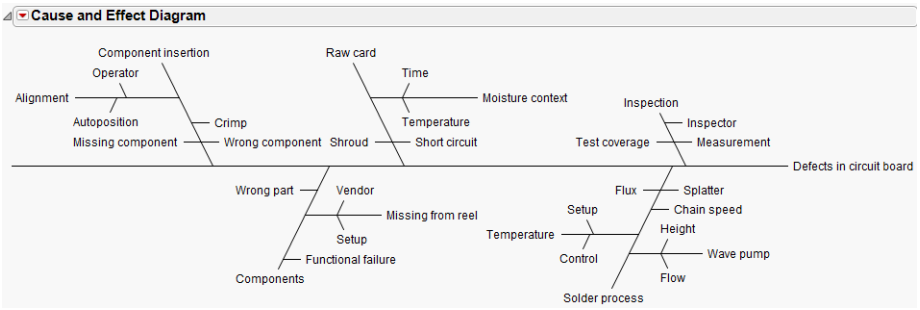
- Organize the causes of an effect (sources of a problem)
- Brainstorm
- Identify variables in preparation for further experimentation

Example of a Cause-and-Effect Diagram

You have data about defects in a circuit board. You want to examine the major factors and possible causes of the defects in a diagram.

1. Select **Help > Sample Data Library** and open Ishikawa.jmp.
2. Select **Analyze > Quality and Process > Diagram**.
3. Select Parent and click **X, Parent**.
4. Select Child and click **Y, Child**.
5. Click **OK**.

Figure 14.2 Ishikawa.jmp Diagram



The major factors are Inspection, Solder process, Raw card, Components, and Component insertion. From each major factor, possible causes branch off, such as Inspection, Measurement, and Test coverage for the Inspection factor.

You can focus on one area at a time to further examine the possible causes or sources of variation for each major factor.

Prepare the Data

Before you produce the diagram, begin with your data in two columns of a data table.

Figure 14.3 Example of the Ishikawa.jmp Data Table

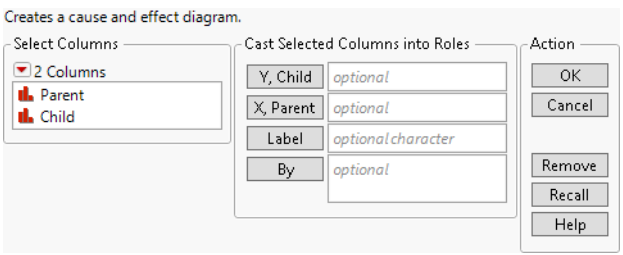
	Parent	Child
1	Defects in circuit board	Inspection
2	Defects in circuit board	Solder process
3	Defects in circuit board	Raw card
4	Defects in circuit board	Components
5	Defects in circuit board	Component insertion
6	Inspection	Measurement
7	Inspection	Test coverage
8	Inspection	Inspector
9	Solder process	Splatter
10	Solder process	Flux
11	Solder process	Chain speed
12	Solder process	Temperature
13	Solder process	Wave pump
14	Temperature	Setup

Notice that the Parent value Defects in circuit board (the effect) has five major factors, listed in the Child column. One of these major factors is Inspection, which has its own causes listed in the Child column. Parent values have children, and children can have their own children (and therefore be listed in both the Parent and Child columns.)

Launch the Diagram Platform

Launch the Diagram platform by selecting **Analyze > Quality And Process > Diagram**.

Figure 14.4 The Diagram Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Tip: To create a basic diagram that is not based on a data table, leave the **Y, Child**, and **X, Parent** fields empty and click **OK**. Then edit the nodes using the options in the right-click menu. See "Right-Click Menus".

Y, Child Represents the child factors contributing to the parent factors.

X, Parent Represents the parent factors (including the effect) that have child factors.

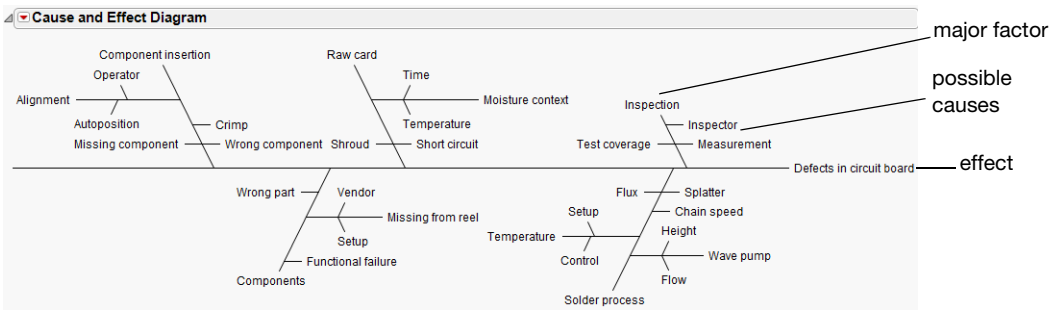
Label Includes the text from the Label columns in the nodes of the diagram.

By Produces separate diagrams for each value of the By variable.

The Cause-and-Effect Diagram

In Figure 14.5, the effect or problem, Defects in circuit board, appears on the right as the center line. The major contributing factors appear at the end of the branches (Inspection, Solder process, Raw Card, and so on.) Possible causes branch off each major factor.

Figure 14.5 Cause-and-Effect Diagram



Right-Click Menus

Right-click a highlighted node to modify text, insert new nodes, change the diagram type, and more. Note the following:

- Right-click a title to change the font and color, positioning, visibility, or formatting.
- Click and highlight a node to rename it.
- Click and drag a node to move it.

Text Menu

The Text menu contains the following options:

Font Select the font of the text or numeric characters.

Color Select the color of the text or numeric characters.

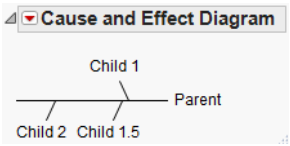
Rotate Left, Rotate Right, Horizontal Rotates the text or numbers to be horizontal, 90 degrees left, or 90 degrees right.

Insert Menu

Use the **Insert** menu to insert items onto existing nodes. The Insert menu contains the following options:

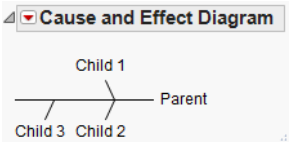
Before Inserts a new node to the right of the highlighted node. For example, Figure 14.6 inserts Child 1.5 before Child 2.

Figure 14.6 Insert Before



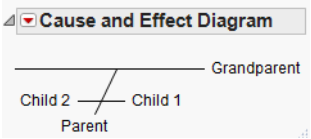
After Inserts a new node to the left of the highlighted node. For example, Figure 14.7 inserts Child 3 after Child 2.

Figure 14.7 Insert After



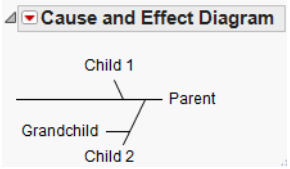
Above Inserts a new node at a level above the current node. For example, Figure 14.8 inserts Grandparent at a level above Parent.

Figure 14.8 Insert Above



Below Inserts a new node at a level below the current node. For example, Figure 14.9 inserts Grandchild at a level below Child 2.

Figure 14.9 Insert Below



Move Menu

Use the Move menu to move nodes or branches. The Move menu contains the following options:

First Moves the highlighted node to the first position under its parent.

Last Moves the highlighted node to the last position under its parent.

Other Side Moves the highlighted node to the opposite side of its parent line.

Force Left Makes all horizontally drawn elements appear to the left of their parent.

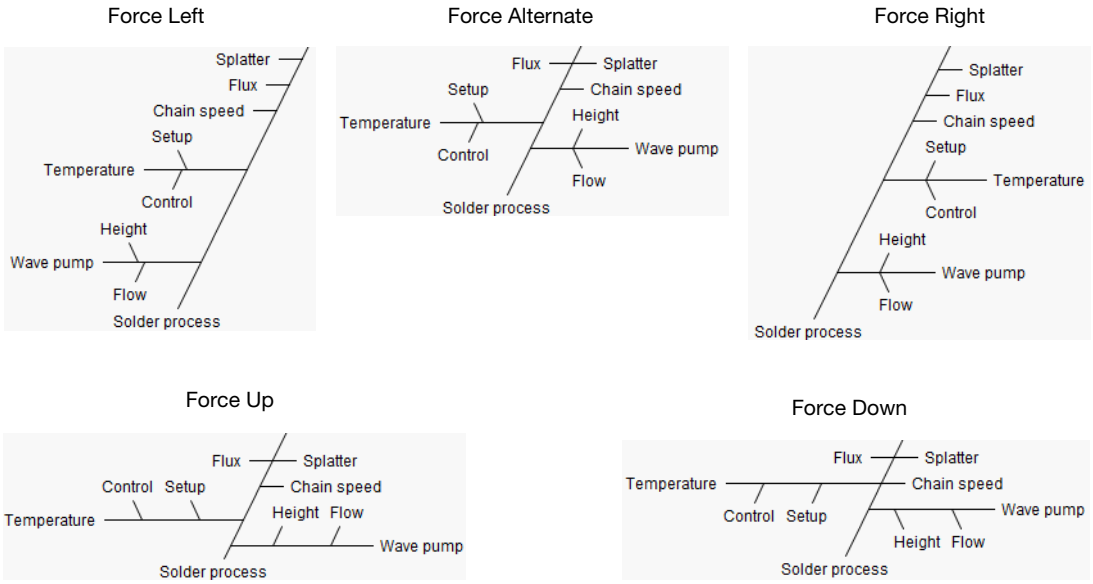
Force Right Makes all horizontally drawn elements appear to the right of their parent.

Force Up Makes all vertically drawn elements appear above their parent.

Force Down Makes all vertically drawn elements appear below their parent.

Force Alternate Draws children on alternate sides of the parent line.

Figure 14.10 Force Options



Other Menu Options

The right-click menu for a highlighted node also contains these options:

Change Type Changes the entire chart type to **Fishbone**, **Hierarchy**, or **Nested**.

Uneditable Disables all other commands except **Move** and **Change Type**.

Text Wrap Width Specifies the width of labels where text wrapping occurs.

Make Into Data Table Converts the currently highlighted node into a data table. Convert the all nodes by highlighting the whole diagram (effect).

Close Shows the highlighted node.

Delete Deletes the highlighted node and all of its children.

Cause and Effect Diagram Menu Options

The Cause and Effect Diagram red triangle menu contains the following options:

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Save the Diagram

There are different ways to save your diagram. Choose from one of the following:

- save the diagram as a data table
- save the diagram as a journal
- save the diagram as a script

Save the Diagram as a Data Table

Note the following about this approach:

- If you have other processes that need to update the data table, this can be a good approach to choose.
- Very little customization is available, because the data table cannot represent the customization.

To save the diagram as a data table:

1. Highlight the entire diagram.
2. Right-click and select **Make Into Data Table**.
3. Save the new data table.

Save the Diagram as a Journal

Note the following about this approach:

- This option can be a good choice for impromptu work. For example, you can manually build the diagram, save it as a journal, then reopen the journal later and continue building and editing the diagram.
- Any customization exists only in the journal, and the journal is not connected to the data table.

To save the diagram as a journal:

1. Highlight the entire diagram.
2. Right-click and select **Edit > Journal**.
3. Save the new journal.

Save the Diagram as a Script

Note the following about this approach:

- If you have other processes that need to update the data table, this can be a good approach to choose.
- If you created the diagram from a data table, a simple script appears that relaunches against the data table with no customization.
- If you created the diagram without using a data table (or from a journal), a more complex script appears that contains all the commands needed to add and customize each area of the diagram.

To save the diagram as a script:

1. Click the red triangle next to Cause and Effect Diagram and select **Save Script > To Script Window**.
2. Save the new script.

Chapter 15

Quality Utilities

Manage Specification Limits and Create OC Curves

This chapter covers utilities in the Analyze > Quality and Process menu. Specifically, the Manage Spec Limits utility and OC Curves utility are discussed.

Contents

Manage Spec Limits Utility 405
Example of the Manage Spec Limits Utility 405
Manage Spec Limits Options. 408
Operating Characteristic Curves Utility 409
Launch the OC Curves Utility. 409
OC Curves for Control Chart Options 410
OC Curves for Acceptance Sampling Options 410

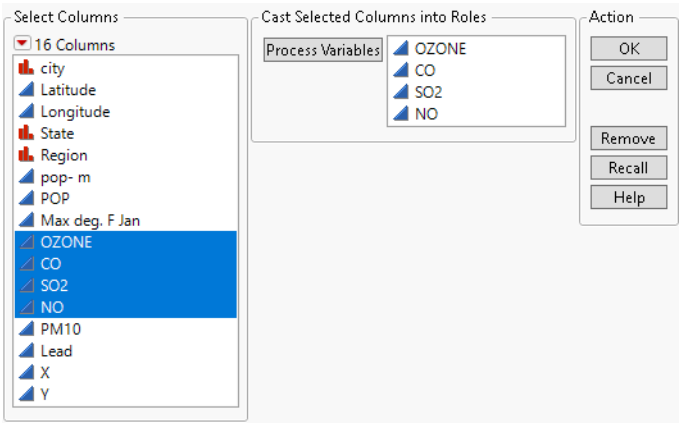
Manage Spec Limits Utility

The Manage Spec Limits utility enables you to quickly add or edit many specification limits for several columns at once. The specification limits are then used in any future analyses. You can also specify importance values for each process and indicate whether limits should appear in graphs as reference lines.

Example of the Manage Spec Limits Utility

1. Select **Help > Sample Data Library** and open Cities.jmp.
2. Select **Analyze > Quality and Process > Manage Spec Limits**.
3. Specify the columns that you want to set specification limits on. For this example, select OZONE, CO, SO2, and NO, and click **Process Variables**.

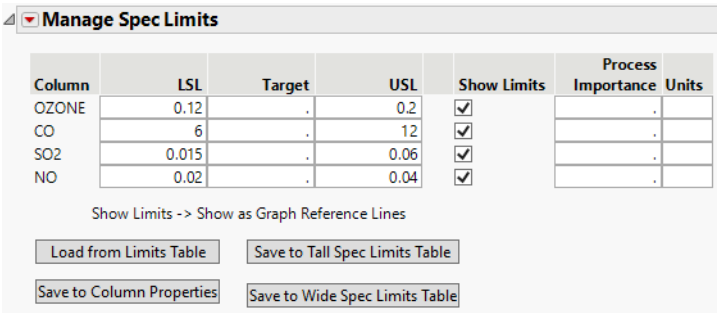
Figure 15.1 Specify Columns



4. Click **OK**.
5. Add your specification limits. You can do this by loading existing limits from a JMP data table (**Load from Limits Table**) or by entering limits manually. For this example, enter the following limits manually:
   - OZONE: LSL 0.12, USL 0.2
   - CO: LSL 6, USL 12
   - SO2: LSL 0.015, USL 0.06
   - NO: LSL 0.02, USL 0.04
6. Click the red triangle next to Manage Spec Limits and select **Show Limits All**.

Specification limits for all columns will appear in graphs for any future analyses. If you want to show the specification limits only for individual columns, check the **Show Limits** box next to those columns.

Figure 15.2 Set Specification Limits

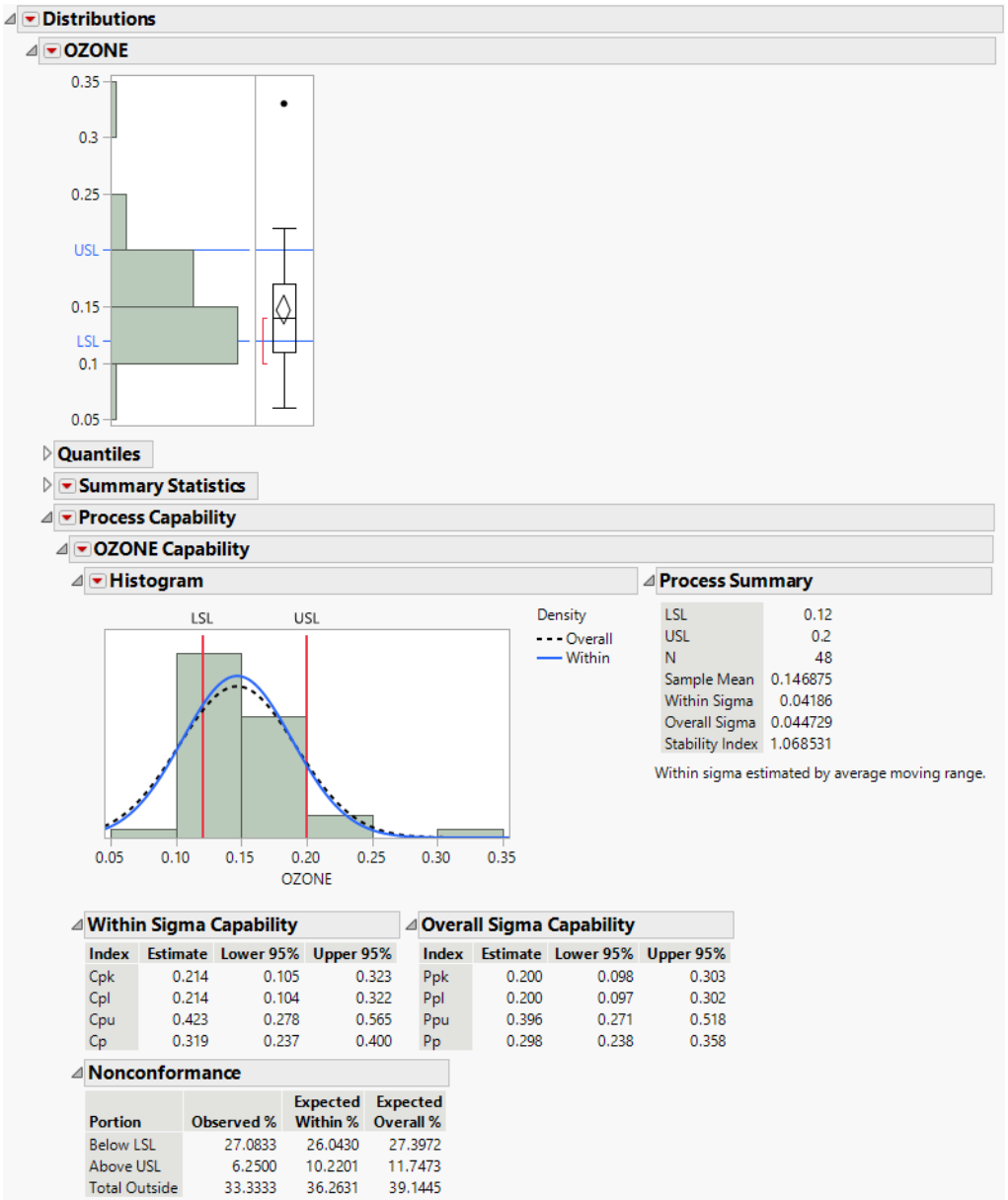


7. Choose how you want to save the specification limits. For this example, click **Save to Column Properties**. This saves them as column properties in the corresponding data table. You could also save them to a new data table (tall or wide format).

   In the Cities.jmp data table Columns panel, notice that asterisks indicating the Spec Limits column property appear next to OZONE, CO, SO2, and NO.

8. To see values that are outside the limits in the data table, click the red triangle next to Manage Spec Limits and select **Color Out of Spec Values**. Go to the Cities.jmp data table, and you can see that any values that are outside the limits are now colored.
9. Now, you can run any analysis. For this example, select **Analyze > Distribution**.
10. Select OZONE, CO, SO2, and NO, and click **Y, Columns**.
11. Click **OK**.

Figure 15.3 Specification Limits for OZONE in Distribution



The specification limits that you added to the OZONE column appear in the histogram. Because the column contains a Spec Limits column property, the Distribution report also contains a Capability Analysis report.

Manage Spec Limits Options

In the window where you set specification limits, there are buttons to save and load specification limits, and options in the Manage Spec Limits red triangle menu.

Report Table Columns

The table at the top of the Manage Spec Limits report contains a row for each process column specified in the launch window. This table enables you to specify specification limits as well as the following options for each column:

Show Limits Specifies that the Show as Graph Reference Lines option is selected in the Spec Limits column property for the specified column.

Process Importance Specifies the process importance value for each column. Process importance values provide a mechanism to sort processes in the order that you prefer. Process importance values are used to size markers in many platform graphs.

Units Specifies the units for each column.

Buttons

Load from Limits Table Loads specification limits from a JMP data table.

Save to Column Properties Saves the specification limits as column properties in the associated data table.

Save to Tall Spec Limits Table Saves the specification limits to a new data table in tall format.

Save to Wide Spec Limits Table Saves the specification limits to a new data table in wide format.

Red Triangle Options

Show Limits All Selects boxes under Show Limits for all of the columns. If Show Limits is selected for a column, the Show as Graph Reference Lines option is selected in the Spec Limits column property. The Show as Graph Reference Lines option displays the specification limits and target that you specify as reference lines in select analysis plots.

Note: If all boxes under Show Limits are selected, the Show Limits All option deselects all of the boxes under Show Limits.

Round Decimals Sets the number of decimal places to which you want the specification limits rounded.

Color Out of Spec Values Colors any values in the data table that are outside the specification limits for the columns.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Operating Characteristic Curves Utility

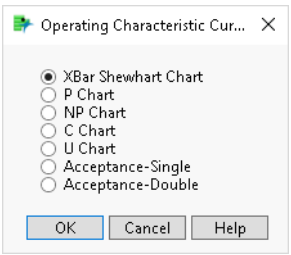
The Operating Characteristic (OC) Curves utility enables you to construct OC curves for control charts and attribute acceptance sampling plans. OC curves are available for XBar, P, NP, C, and U charts. For specified control charts, the OC curve shows the probability of failing to detect a shift of a particular size. In addition, there are OC curves for single and double attribute acceptance sampling plans. For a specified acceptance sampling plan, the OC curve shows how the probability of accepting a lot changes with the lot quality.

Note: The OC curves for control charts are two-sided curves. They are drawn for negative and positive shifts. Often OC curves display one curve for the absolute shift.

Launch the OC Curves Utility

Launch the Operating Characteristic Curves by selecting **Analyze > Quality and Process > OC Curves.** Select the OC curve of interest and click OK to launch.

Figure 15.4 OC Curves Launch Window



OC Curves for Control Chart Options

For control charts, you can use OC curves to explore how chart parameter settings impact β, which is the probability of failing to detect a specified shift of interest. Use the text boxes, sliders, or the LCL, Shift, and UCL handles on the OC curve to set and adjust chart parameters. The chart parameters are as follows:

Lower Control Limit Specifies the lower control limit from your control chart.

Upper Control Limit Specifics the upper control limit from your control chart.

Sample Size (Not available for C charts.) Specifies the sample size used for your control chart measure.

Sigma (Available only for XBar charts.) Specifies your control chart sigma.

Shift Specifies the shift to detect.

Beta Specifies the probability of failing to detect the specified shift given the control chart specifications. Beta updates as the specifications are changed.

OC Curves for Acceptance Sampling Options

For attributes acceptance sampling, you can use OC curves to explore how sampling plans and assumed product quality impact the probability of accepting a lot. Use the text boxes, sliders, or fraction defective handle on the OC curve to set and adjust your sampling plan.

Single Sampling OC Curve Options

Sampling Type Enables you to select Lot Sampling or Binomial Sampling.

Lot Sampling Enables you to specify and explore an acceptance plan based on a fixed lot size.

Binomial Sampling Enables you to specify and explore an acceptance plan for a continuous process or other situation where the binomial distribution is appropriate.

Lot size (N) (Available only for Lot Sampling.) Specifies the size of the lot that you are sampling from.

Sample Size (n) Specifies the number of units for inspection.

Acceptable failures (c1) specifies the number of allowable failures. If the number of observed defects is greater than c1, then the lot is rejected.

Fraction Defective Specifies the expected fraction defective in the lot.

Probability of Acceptance Specifies the probability of accepting the lot given the sampling plan as defined. The Probability of Acceptance updates as the specifications are changed, but you can also adjust this value directly. When you adjust the Probability of Acceptance directly, the Fraction Defective value is updated.

Double Sampling OC Curve Options

First Sample Contains the following settings for the first sample:

Number inspected (n1) Specifies the number of units inspected in the first sample.

Acceptable failures (c1) specifies the number of allowable failures in the first sample.

Second Sample Contains the following settings for the second sample:

Number inspected (n2) Specifies the number of units inspected in the second sample.

Acceptable failures (c1+c2) specifies the total number of allowable failures.

Fraction Defective Specifies the expected fraction defective in the lot.

Probability of Acceptance Specifies the probability of accepting the lot given the sampling plan as defined. The Probability of Acceptance updates as the specifications are changed, but you can also adjust this value directly. When you adjust the Probability of Acceptance directly, the Fraction Defective value is updated.

Appendix A

References

Agresti, A., and Coull, B. (1998). "Approximate is Better Than 'Exact' for Interval Estimation of Binomial Proportions." *The American Statistician* 52:119–126.

American Society for Quality Statistics Division (2004). *Glossary and Tables for Statistical Quality Control*. 4th ed. Milwaukee, WI: ASQC Quality Press.

American Society for Testing and Materials (1976). *ASTM Manual on Presentation of Data and Control Chart Analysis*. Philadelphia: ASTM.

Automotive Industry Action Group (2002). *Measurement Systems Analysis Reference Manual*. 3rd ed. Troy, MI: Automotive Industry Action Group.

AIAG (2005). *Statistical Process Control*. 2nd ed. Troy, MI: Automotive Industry Action Group.

Barrentine, L. B. (1991). *Concepts for R&R Studies*. Milwaukee, WI: ASQC Quality Press.

Bissell, A. F. (1990). "How Reliable Is Your Capability Index?" *Journal of the Royal Statistical Society, Series C* 39:331–340.

Box, G. E. P, Luceño, A., and Paniagua-Quinones, M. d. C. (2009). *Statistical Control by Monitoring and Adjustment*. 2nd ed. New York: John Wiley & Sons.

Burdick, P. K., Borror, C. M., and Montgomery, D. C. (2005). *Design and Analysis of Gauge R&R Studies: Making Decisions with Confidence Intervals in Random Mixed ANOVA Models.* Philadelphia, PA and Alexandria, VA: SIAM and ASA.

Chou, Y.-M., Owen, D. B., and Borrego, S. A. (1990). "Lower confidence limits on process capability indices." *Journal of Quality Technology* 22:223–229.

Crowder, S. V. (1987). "Average Run Lengths of Exponentially Weighted Moving Average Charts." *Journal of Quality Technology* 29:401–408.

David, H. A. (1951). "Further Applications of Range to the Analysis of Variance." *Biometrika* 38:393–407.

Fleiss, J. L. (1981). *Statistical Methods for Rates and Proportions*. New York: John Wiley & Sons.

Goel, A. L., and Wu, S. M. (1971). "Determination of A.R.L. and a Contour Nomogram for Cusum Charts to Control Normal Mean." *Technometrics* 13:221–230.

Harter, H. L. (1960). "Tables of Range and Studentized Range." *Annals of Mathematical Statistics* 31:1122–1147.

Hoffman, D. (2003). "Negative Binomial Control Limits for Count Data with Extra-Poisson Variation." *Pharmaceutical Statistics* 2:127–132.

Jackson, J. E., and Mudholkar, G. S. (1979). "Quality Control Procedures for Residuals Associated with Principal Component Analysis." *Annals of Mathematical Statistics* 25:290–302.

Knoth, S. (2004). "Fast Initial Response Features for EWMA Control Charts." *Statistical Papers* 46:47–64.

Kourti, T., and MacGregor, J. F. (1996). "Multivariate SPC Methods for Process and Product Monitoring." *Journal of Quality Technology* 28:409–428.

Levey, S., and Jennings, E. R. (1950). "The Use of Control Charts in the Clinical Laboratory." *American Journal of Clinical Pathology* 20:1059–1066.

Li, G., Qin, S., Ji, Y., Zhou, D.(2009). "Total PLS Based Contribution Plots for Fault Diagnosis." *Acta Automatica Sinica* 35:759 - 765.

Lucas, J. M. (1976). "The Design and Use of V–Mask Control Schemes." *Journal of Quality Technology* 8:1–12.

Lucas, J. M., and Crosier, R. B. (1982). "Fast Initial Response for CUSUM Quality Control Schemes: Give Your CUSUM a Head Start." *Technometrics* 24:199–205.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Montgomery, D. C. (2013). *Introduction to Statistical Quality Control.* 7th ed. New York: John Wiley & Sons.

Nair, V. N. (1984). "Confidence Bands for Survival Functions with Censored Data: A Comparative Study." *Technometrics* 26:265–275.

Nelson, L. (1984). "The Shewhart Control Chart—Tests for Special Causes." *Journal of Quality Technology* 15:237–239.

Nelson, L. (1985). "Interpreting Shewhart X Control Charts." *Journal of Quality Technology* 17:114–116.

Nomikos, P., and MacGregor, J. F., (1995). "Multivariate SPC Charts for Monitoring Bach Processes." *Technometrics* 37:41–59.

Pearn, W. L., and Kotz, S. (2006). *Encyclopedia and Handbook of Process Capability Indices*. Vol. 12 of Series on Quality, Reliability, and Engineering Statistics. New Jersey: World Scientific.

Portnoy, S. (1971). "Formal Bayes Estimation with Application to a Random Effects Model." *The Annals of Mathematical Statistics* 42:1379–1402.

Sahai, H. (1974). "Some Formal Bayes Estimators of Variance Components in the Balanced Three-Stage Nested Random Effects Model." *Communication in Statistics – Simulation and Computation* 3:233–242.

Slifker, J. F., and Shapiro, S. S. (1980). "The Johnson System: Selection and Parameter Estimation." *Technometrics* 22:239–246.

Sullivan, J. H., and Woodall, W. H. (2000). "Change-point detection of mean vector or covariance matrix shifts using multivariate individual observations." *IIE Transactions* 32:537–549.

Tracy, N. D., Young, J. C., and Mason, R. L. (1992). "Multivariate Control Charts for Individual Observations." *Journal of Quality Technology* 24:88–95.

Westgard, J. O. (2002). *Basic QC Practices.* 2nd ed. Madison, WI: Westgard QC.

Wheeler, D. J. (2004). *Advanced Topics in Statistical Process Control.* 2nd ed. Knoxville, TN: SPC Press.

Wheeler, D. J. (2006). *EMP III Using Imperfect Data*. Knoxville, TN: SPC Press.

Wludyka, P., and Sa, P. (2004). "A robust I-Sample analysis of means type randomization test for variances for unbalanced designs." *Journal of Statistical Computation and Simulation* 74:701–726.

Woodall, W. H., and Adams, B. M. (1998). *Handbook of Statistical Methods for Engineers and Scientists*. 2nd ed. Chapter 7. New York: McGraw-Hill.

Appendix B

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

  - You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - You must cause any modified files to carry prominent notices stating that You changed the files; and
  - You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
  - You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Reliability and Survival Methods

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Reliability and Survival Methods*. Cary, NC: SAS Institute Inc.

JMP® 16 Reliability and Survival Methods

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Reliability and Survival Methods

1 Learn about JMP 13
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 15
JMP Help 16
JMP Documentation Library 16
Additional Resources for Learning JMP 22
JMP Tutorials 22
Sample Data Tables 22
Learn about Statistical and JSL Terms 23
Learn JMP Tips and Tricks 23
JMP Tooltips 23
JMP User Community 24
Free Online Statistical Thinking Course 24
JMP New User Welcome Kit 24
Statistics Knowledge Portal 24
JMP Training 24
JMP Books by Users 25
The JMP Starter Window 25
JMP Technical Support 25

2 Introduction to Reliability and Survival 27
Lifetime and Failure Analysis

3 Life Distribution 29
Fit Distributions to Lifetime Data
Overview of the Life Distribution Platform 31
Example of the Life Distribution Platform 31
Launch the Life Distribution Platform 34
Life Distribution Report 37
Event Plot 38
Compare Distributions 41
Life Distribution: Statistics 45
Life Distribution Report Options 52

Mixture ... 54
Fit Competing Risk Mixture ... 57
Weibayes Report ... 58
Competing Cause Report ... 59
Competing Cause Workflow ... 60
Competing Cause Model ... 60
Cause Combination ... 61
Competing Cause: Statistics ... 62
Individual Causes ... 63
Competing Cause Report Options ... 63
Life Distribution - Compare Groups Report ... 65
Compare Groups: Statistics ... 66
Individual Group ... 67
Life Distribution - Compare Groups Report Options ... 67
Additional Examples of the Life Distribution Platform ... 68
Omit Competing Causes ... 69
Change the Scale ... 70
Examine the Same Distribution across Groups ... 72
Weibayes Estimates ... 74
Fit Mixture Example ... 76
Fit Competing Risk Mixture Example ... 79
Statistical Details for the Life Distribution Platform ... 81
Distributions ... 81
Competing Cause Details ... 96
Notes Regarding Median Rank Regression ... 101

4 Fit Life by X ... 103
Fit Single-Factor Models to Time-to Event Data

Overview of the Fit Life by X Platform ... 105
Example of the Fit Life by X Platform ... 106
Launch the Fit Life by X Platform ... 108
The Fit Life by X Report ... 110
Summary of Data ... 111
Scatterplot ... 111
Nonparametric Overlay ... 113
Comparisons ... 114
Results ... 118
Custom Relationship ... 129
Fit Life by X Platform Options ... 130
Additional Examples of the Fit Life by X Platform ... 131

Capacitor Example ... 131
Custom Relationship Example ... 133

5 Cumulative Damage ... 137
Model Product Deterioration with Variable Stress over Time

Overview of the Cumulative Damage Platform ... 139
Example of the Cumulative Damage Platform ... 139
Launch the Cumulative Damage Platform ... 143
Time to Event ... 144
Stress Pattern ... 145
The Cumulative Damage Report ... 148
Event Plot ... 148
Stress Patterns Report ... 148
Model List ... 149
Model Results ... 149
Cumulative Damage Platform Options ... 150
Stress Patterns Options ... 150
Additional Example of the Cumulative Damage Platform ... 152
Example of Simulating New Data ... 152

6 Recurrence Analysis ... 155
Model the Frequency or Cost of Recurrent Events over Time

Overview of the Recurrence Analysis Platform ... 157
Example of the Recurrence Analysis Platform ... 157
Launch the Recurrence Analysis Platform ... 160
Recurrence Analysis Platform Options ... 161
Fit Model ... 162
Additional Examples of the Recurrence Analysis Platform ... 166
Bladder Cancer Recurrences Example ... 167
Diesel Ship Engines Example ... 170

7 Degradation ... 175
Model Product Deterioration over Time

Overview of the Degradation Platform ... 177
Example of the Degradation Platform ... 177
Launch the Degradation Platform ... 179
The Degradation Reports ... 181
Model Specification ... 184
Simple Linear Path ... 184
Nonlinear Path ... 186
Inverse Prediction ... 194

Prediction Graph ... 196
Degradation Platform Options ... 197
Model Reports ... 199
Model Lists ... 199
Reports ... 200
Destructive Degradation ... 200
Stability Analysis ... 205

8 Destructive Degradation ... 209
Model Product Deterioration over Time

Example of the Destructive Degradation Platform ... 211
Launch the Destructive Degradation Platform ... 217
Specify Two Y Columns ... 218
The Destructive Degradation Plot Options and Models ... 219
Plot Options ... 220
Models ... 221
The Destructive Degradation Report ... 225
Model List ... 226
Model Outlines ... 226
Destructive Degradation Platform Options ... 229
Statistical Details for the Destructive Degradation Platform ... 229

9 Reliability Forecast ... 231
Forecast Product Failure Using Production and Failure Data

Overview of the Reliability Forecast Platform ... 233
Example Using the Reliability Forecast Platform ... 233
Launch the Reliability Forecast Platform ... 237
The Reliability Forecast Report ... 240
Observed Data Report ... 240
Life Distribution Report ... 240
Forecast Report ... 241
Reliability Forecast Platform Options ... 245
Additional Example Using the Reliability Forecast Platform ... 246

10 Reliability Growth ... 249
Model System Reliability as Changes Are Implemented

Overview of the Reliability Growth Platform ... 251
Example Using the Reliability Growth Platform ... 252
Launch the Reliability Growth Platform ... 256
Launch Window Roles ... 257
Data Table Structure ... 258

The Reliability Growth Report ... 261
Model Descriptions and Feasibilities ... 261
Observed Data Report ... 261
Reliability Growth Platform Options ... 264
Model Reports ... 265
Crow AMSAA ... 266
Crow AMSAA with Modified MLE ... 272
Fixed Parameter Crow AMSAA ... 273
Piecewise Weibull NHPP ... 274
Reinitialized Weibull NHPP ... 278
Piecewise Weibull NHPP Change Point Detection ... 281
Additional Examples of the Reliability Growth Platform ... 282
Piecewise NHPP Weibull Model Fitting with Interval-Censored Data ... 282
Piecewise Weibull NHPP Change Point Detection with Time in Dates Format ... 285
Statistical Details for the Reliability Growth Platform ... 287
Statistical Details for the Crow-AMSAA Report ... 287
Statistical Details for the Piecewise Weibull NHPP Change Point Detection Report ... 288

11 Reliability Block Diagram ... 289
Engineer System Reliabilities

Overview of the Reliability Block Diagram Platform ... 291
Example Using the Reliability Block Diagram Platform ... 291
Add Components ... 293
Align Shapes ... 295
Connect Shapes ... 296
Configure Components ... 297
The Reliability Block Diagram Window ... 299
Preview Window ... 300
Reliability Block Diagram Platform Options ... 301
Workspace Options ... 304
Configuration Settings ... 305
Distribution Configurations ... 305
Specify a Nonparametric Distribution ... 307
Options for Design and Library Items ... 308
Profilers ... 309
Component Importance and Time to Failure ... 311
Component Plots ... 313
Print Algebraic Reliability Formula ... 316
Generate Algebraic Expression Data Table ... 317
Clone and Delete ... 317

12 Repairable Systems Simulation 319
Estimate Outage Time for a Complex System
Overview of the Repairable Systems Simulation Platform 321
Example Using the Repairable Systems Simulation Platform 321
The Repairable Systems Simulation Window 323
System Diagram 324
Shapes Toolbar 325
Configuration Panel 326
Add Event Panel 329
Add Action Panel 330
Repairable Systems Simulation Platform Options 333
Options for Block Items 334
Distribution Options 334
Specify a Nonparametric or Estimated Distribution 335
Event Settings 336
Action Settings 337
Repairable Systems Simulation Results 338
Results Table 338
Results Explorer 339
Repairable Systems Simulation Results Options 341
RSS Results Explorer Point Estimation Profilers 342

13 Survival Analysis 343
Analyze Survival Time Data
Overview of the Survival Analysis Platform 345
Example of Survival Analysis 346
Launch the Survival Platform 348
The Survival Plot 349
Survival Platform Options 350
Exponential, Weibull, and Lognormal Plots and Fits 352
Fitted Distribution Plots 356
Competing Causes 357
Additional Examples of the Survival Platform 358
Example of Survival Analysis 358
Example of Competing Causes 361
Example of Interval Censoring 363
Statistical Reports for Survival Analysis 365

14 Fit Parametric Survival 369
Fit Survival Data Using Regression Models
Overview of the Fit Parametric Survival Platform 371

Example of Parametric Regression Survival Fitting ... 371
Launch the Fit Parametric Survival Platform ... 373
The Parametric Survival Fit Report ... 375
The Parametric Survival - All Distributions Report ... 376
Parametric Competing Cause Report ... 378
Fit Parametric Survival Options ... 378
Nonlinear Parametric Survival Models ... 380
Additional Examples of Fitting Parametric Survival ... 381
Arrhenius Accelerated Failure LogNormal Model ... 381
Interval-Censored Accelerated Failure Time Model ... 385
Analyze Censored Data Using the Nonlinear Platform ... 386
Left-Censored Data ... 388
Weibull Loss Function Using the Nonlinear Platform ... 389
Fitting Simple Survival Distributions Using the Nonlinear Platform ... 392
Statistical Details for the Fit Parametric Survival Platform ... 395
Loss Formulas for Survival Distributions ... 395

15 Fit Proportional Hazards ... 397
Fit Survival Data Using Semiparametric Regression Models

Overview of the Fit Proportional Hazards Platforms ... 399
Example of the Fit Proportional Hazards Platform ... 399
Risk Ratios for One Nominal Effect with Two Levels ... 401
Launch the Fit Proportional Hazards Platform ... 402
The Fit Proportional Hazards Report ... 404
Fit Proportional Hazards Options ... 405
Example Using Multiple Effects and Multiple Levels ... 406
Risk Ratios for Multiple Effects and Multiple Levels ... 408

A References ... 411

B Technology License Notices ... 415

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation 15
JMP Help 16
JMP Documentation Library 16
Additional Resources for Learning JMP 22
JMP Tutorials 22
Sample Data Tables 22
Learn about Statistical and JSL Terms 23
Learn JMP Tips and Tricks 23
JMP Tooltips 23
JMP User Community 24
Free Online Statistical Thinking Course 24
JMP New User Welcome Kit 24
Statistics Knowledge Portal 24
JMP Training 24
JMP Books by Users 25
The JMP Starter Window 25
JMP Technical Support 25

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction to Reliability and Survival

Lifetime and Failure Analysis

Reliability and Survival Methods describes a number of methods and tools that are available in JMP to help you evaluate and improve reliability in a product or system and analyze survival data for people and products:

- The Life Distribution platform enables you to analyze the lifespan of a product, component, or system to improve quality and reliability. This analysis helps you determine the best material and manufacturing process for the product, thereby increasing the quality and reliability of the product. See "Life Distribution".
- The Fit Life by X platform helps you analyze lifetime events when only one factor is present. You can choose to model the relationship between the event and the factor using various transformations, or create a custom transformation of your data. See "Fit Life by X".
- The Cumulative Damage platform enables you to analyze an accelerated life test where the stress levels might have changed over time. See "Cumulative Damage".
- The Recurrence Analysis platform analyzes event times where the events can recur several times for each unit. Typically, these events occur when a unit breaks down, is repaired, and then put back into service after the repair. See "Recurrence Analysis".
- The Degradation platform analyzes degradation data to predict pseudo failure times. These pseudo failure times can then be analyzed by other reliability platforms to estimate failure distributions. You can include an explanatory factor. You can perform stability analysis to set product expiration dates. You can also fit custom destructive degradation models. See "Degradation".
- The Destructive Degradation platform models failure data for product characteristics whose measurement requires that the product be destroyed. This results in a single observation per product unit. You can also include an acceleration factor. A wide range of common degradation models is available. See "Destructive Degradation".
- The Reliability Forecast platform helps you predict the number of future failures. The analysis estimates the parameters for a life distribution using production dates, failure dates, and production volume. See "Reliability Forecast".
- The Reliability Growth platform models the change in reliability of a single repairable system over time as improvements are incorporated into its design. See "Reliability Growth".

- JMP PRO The Reliability Block Diagram platform displays the reliability relationship between a system's components and, if reliability distributions are given to the components, analytically obtains the reliability behavior. See "Reliability Block Diagram".
- JMP PRO The Repairable Systems Simulation platform enables you to interactively define the relationships between the components of a repairable system. It can also simulate the down time of the system. See "Repairable Systems Simulation".
- The Survival platform computes survival estimates for one or more groups. It can be used as a complete analysis or is useful as an exploratory analysis to gain information for more complex model fitting. See "Survival Analysis".
- The Fit Parametric Survival platform fits the time to event variable using linear regression models that can involve both location and scale effects. The fit is performed using several distributions. See "Fit Parametric Survival".
- The Fit Proportional Hazards platform fits the Cox proportional hazards model, which assumes a multiplying relationship between predictors and the hazard function. See "Fit Proportional Hazards".

Chapter 3

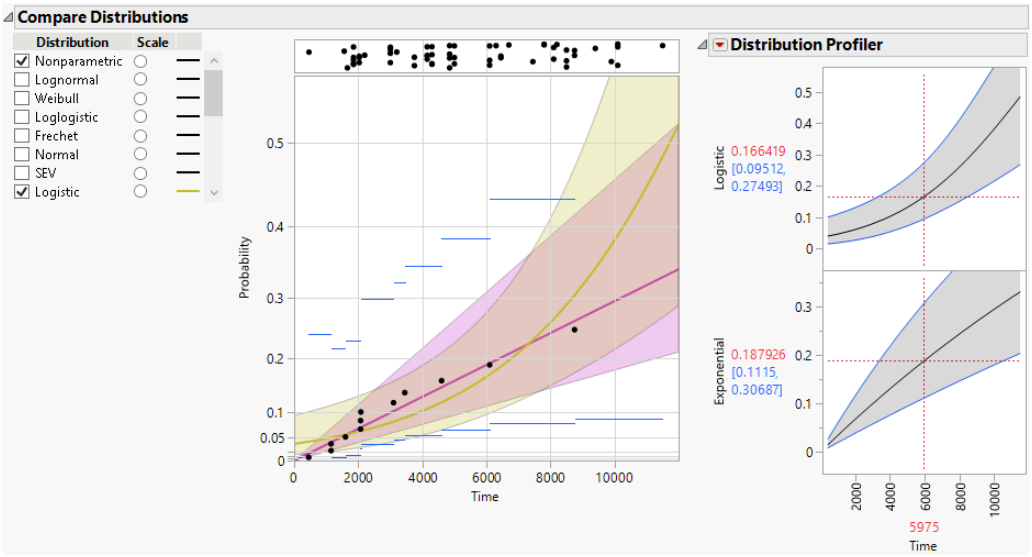
Life Distribution

Fit Distributions to Lifetime Data

The Life Distribution platform models time-to-event data. The platform accommodates both right-censored and interval-censored data. Use Life Distribution to do the following:

- Compare multiple distributional fits to determine which distribution best fits your data.
- Construct Bayesian fits.
- Model zero-failure data.
- Compare groups to analyze group differences.
- Analyze multiple causes of failure.
- Estimate the components of a mixture and estimate the probability that an observation comes from a given component.
- Estimate the components of a competing risk mixture and estimate the impact of a component on an observation.

Figure 3.1 Distributional Fits and Comparisons



Contents

Overview of the Life Distribution Platform . . . 31
Example of the Life Distribution Platform . . . 31
Launch the Life Distribution Platform . . . 34
Life Distribution Report . . . 37
Event Plot . . . 38
Compare Distributions . . . 41
Life Distribution: Statistics . . . 45
Life Distribution Report Options . . . 52
Mixture . . . 54
Fit Competing Risk Mixture . . . 57
Weibayes Report . . . 58
Competing Cause Report . . . 59
Competing Cause Workflow . . . 60
Competing Cause Model . . . 60
Cause Combination . . . 61
Competing Cause: Statistics . . . 62
Individual Causes . . . 63
Competing Cause Report Options . . . 63
Life Distribution - Compare Groups Report . . . 65
Compare Groups: Statistics . . . 66
Individual Group . . . 67
Life Distribution - Compare Groups Report Options . . . 67
Additional Examples of the Life Distribution Platform . . . 68
Omit Competing Causes . . . 69
Change the Scale . . . 70
Examine the Same Distribution across Groups . . . 72
Weibayes Estimates . . . 74
Fit Mixture Example . . . 76
Fit Competing Risk Mixture Example . . . 79
Statistical Details for the Life Distribution Platform . . . 81
Distributions . . . 81
Competing Cause Details . . . 96
Notes Regarding Median Rank Regression . . . 101

Overview of the Life Distribution Platform

Life data analysis, or *life distribution analysis*, is the process of modeling the lifespan of a product, component, or system to predict lifetime or time to failure. For example, you can observe failure rates over time to predict when a computer component might fail. This technique enables you to compare materials and manufacturing processes for the product, enabling you to increase the quality and reliability of the product. Decisions on warranty periods and advertising claims can also be more accurate.

With the Life Distribution platform, you can analyze *censored* data in which some time observations are unknown. And if there are potentially multiple causes of failure, you can analyze the *competing causes* to estimate which cause is more influential.

You can use the **Reliability Test Plan** and **Reliability Demonstration** calculators to choose the appropriate sample sizes for reliability studies. These calculators are found at **DOE > Sample Size and Power**. See the *Design of Experiments Guide*.

Example of the Life Distribution Platform

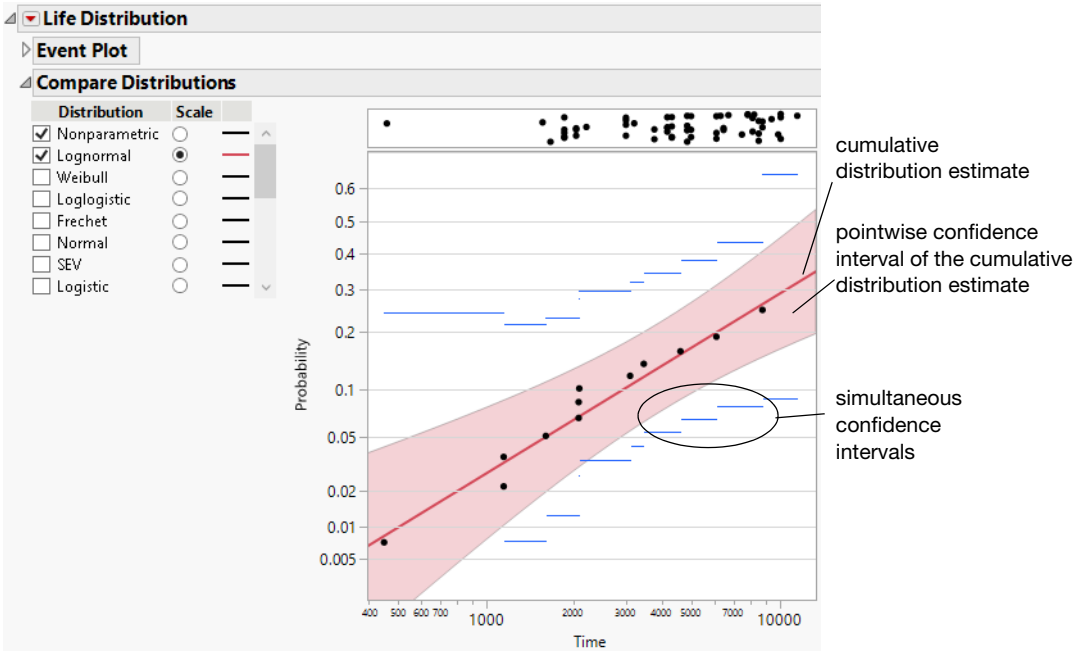
Suppose you have failure times for 70 engine fans, with some of the failure times censored. You want to fit a distribution to the failure times and then estimate various measurements of reliability.

1. Select **Help > Sample Data Library** and open Reliability/Fan.jmp.
2. Select **Analyze > Reliability and Survival > Life Distribution**.
3. Select Time and click **Y, Time to Event**.
4. Select Censor and click **Censor**.
5. Click **OK**.

   The Life Distribution report window appears.
6. In the Compare Distribution report, select **Lognormal** distribution and the corresponding **Scale** radio button.

   A probability plot appears in the report window.

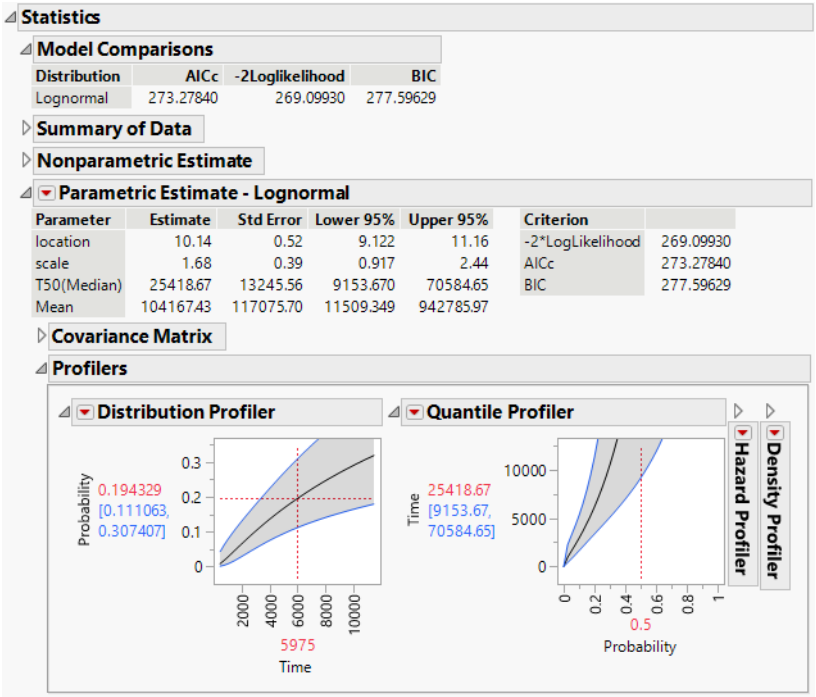
Figure 3.2 Probability Plot



In the probability plot, the data points generally fall along the red line, indicating that the lognormal fit is reasonable.

Below the Compare Distributions report, the Statistics report appears. This report provides a Model Comparison report, nonparametric and parametric estimates, profilers, and more.

Figure 3.3 Statistics Report

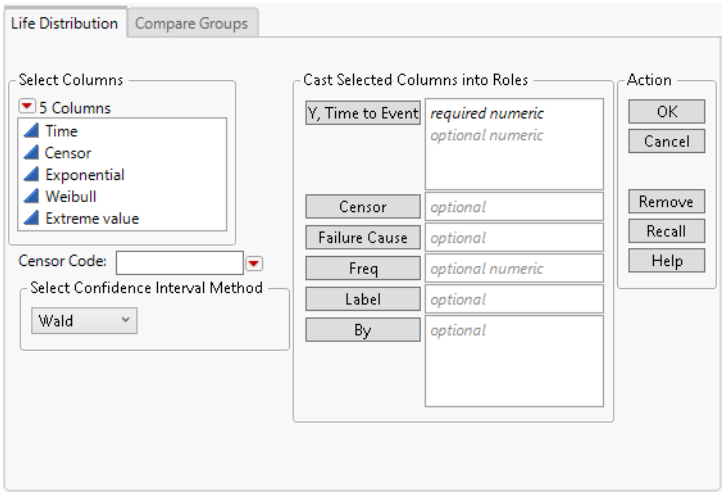


The parameter estimates for the lognormal distribution are provided. The profilers are useful for visualizing the fitted distribution and for estimating probabilities and quantiles. For example, the Quantile Profiler indicates that the estimated median time to failure is 25,418.67 hours.

Launch the Life Distribution Platform

Launch the Life Distribution platform by selecting **Analyze > Reliability and Survival > Life Distribution**. For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Figure 3.4 The Life Distribution Launch Window



Launch Window Tabs

The launch window includes two tabs:

- The Life Distribution tab models ungrouped data. The following types of reports can result:
  - The Life Distribution report appears when you do not specify a Failure Cause role. From this report, you can compare common distributions and examine statistics. See "Life Distribution Report".
  - The Weibayes report appears when you have zero failures in your data. See "Weibayes Report".
  - The Competing Cause report appears when you specify a Failure Cause role. In addition to the features in the Life Distribution report, you can also compare individual failure causes. See "Competing Cause Report".

  Note: You can examine Fixed Parameter and Bayesian models in the Life Distribution and Competing Cause reports.

- The Compare Groups tab enables you to specify a Grouping variable. The Compare Groups report compares different groups using a single specified distribution. For

example, you might compare Weibull fits for components grouped by supplier. In contrast, the Life Distribution tab compares several fitted distributions for a single group. See "Life Distribution - Compare Groups Report".

Launch Window Options

The launch window contains the following options:

Y, Time to Event One or more response columns. The number of response columns specified depends on the censoring structure in the data table.

- If one variable is specified, it is interpreted as the time to event (such as the time to failure) or time to right censoring. Use the Censor column to indicate right-censored responses. For more information about right censoring, see "Single Time to Event Column".
- If two variables are specified, they are interpreted as interval-censored observations. The first Y variable gives the lower limit and the second Y variable gives the upper limit for each unit. For an example of using two response columns to represent various types of censoring, see Figure 3.5. For more information about censoring with two response columns, see "Two Time to Event Columns".

Figure 3.5 Censored Data Types for Two Response Variables

	Start Time	End Time	Count	Censoring Type
1	•	50	50	Left
2	•	100	6	Left
3	25	•	30	Right
4	75	•	10	Right
5	80	150	4	Interval
6	100	250	7	Interval

- If three or more variables are specified, the report contains a separate analysis computed using each specified variable as time to event data.

Grouping (Appears only in the Compare Groups tab.) A column containing the groups that you want to compare. For an example, see "Examine the Same Distribution across Groups".

Censor A column that identifies right-censored observations. Select the value that identifies right-censored observations from the Censor Code menu beneath the Select Columns list. The Censor column is used only when one Y is entered.

Failure Cause A column that contains multiple failure causes. If a Failure Cause column is selected, then a section is added to the window. This section contains check boxes that allow the failure mode to use ZI distributions, TH distributions, DS distributions, fixed parameter models, or Bayesian models for the analysis. The following options are also available:

Distribution Specifies the initial distribution to fit for each failure cause. Select one distribution to fit for all causes; select **Individual Best** to let the platform automatically choose the best fit for each cause; or select **Manual Pick** to manually choose the distribution to fit for each failure cause after JMP creates the Life Distribution report. You can also change the distribution fits in the Life Distribution report itself.

Comparison Criterion (Appears only when you choose the **Individual Best** distribution fit.) Specifies the method by which JMP chooses the best distribution: Corrected Akaike Information Criterion (AICc), Bayesian Information Criterion (BIC), or twice the negative log-likelihood (-2Loglikelihood). See *Fitting Linear Models*. You can change the method later in the Model Comparisons report. See "Model Comparisons".

Censor Indicator in Failure Cause Column Identifies the value used in the **Failure Cause** column to indicate that an observation did not fail. To specify such an indicator, select this option and then enter the indicator in the box that appears. You can also select a value from the list to the right of the box.

See Meeker and Escobar (1998, ch. 15) for a discussion of multiple failure causes. "Omit Competing Causes" illustrates how to analyze multiple causes.

Freq A column that contains frequencies or observation counts when the information in a row represents multiple units. If the value in a row is 0 or a positive number, then the value represents the frequencies or counts of observations for that row.

Label A column that contains identifiers other than the row number. These labels appear on the *Y* axis in the event plot.

By An optional variable whose levels define rows used to create separate models.

Censor Code Identifies the value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

Select Confidence Interval Method Defines the method used for computing confidence intervals for the parameters. The default is Wald, but you can select Likelihood instead. However, all confidence intervals provided in the profilers are based on the Wald method. This is done to reduce computation time. See "Estimation and Confidence Intervals".

Failure Distribution by Cause (Appears only in the Life Distribution tab when a Cause is specified.) Specify which families of distributions should be available to model the life distributions for individual causes. Select an initial distribution, Individual Best, or Manual Pick from the Distribution menu. See "Failure Cause".

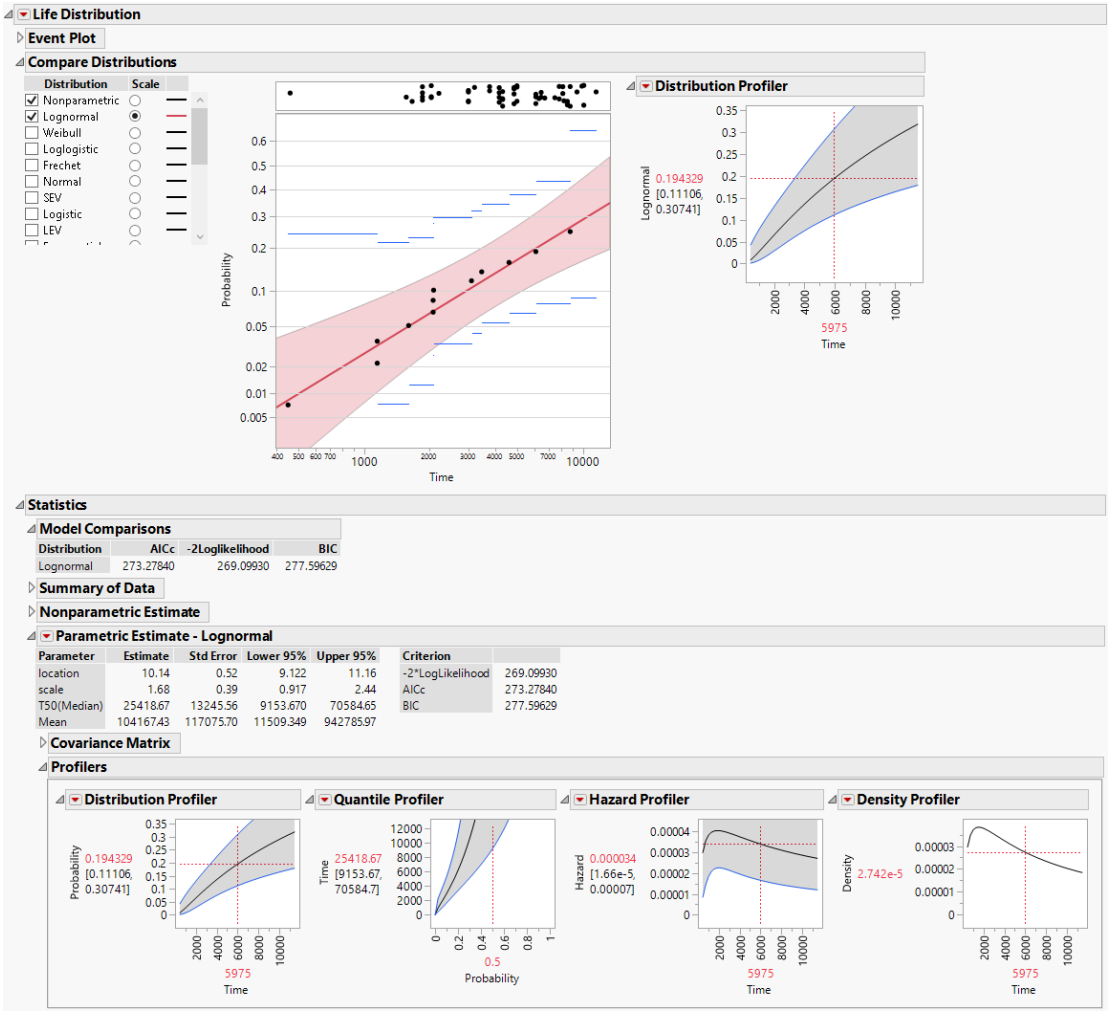
Life Distribution Report

Tip: If you find that the report window is too long, select **Tabbed Report** from the Life Distribution red triangle menu.

If you have not selected a Failure Cause in the launch window, the Life Distribution report appears. Use this report to analyze lifetime data where some time observations might be censored. The Life Distribution report contains the following content and options:

- "Event Plot"
- "Compare Distributions"
- "Life Distribution: Statistics"
- "Life Distribution Report Options"

Figure 3.6 Example of the Life Distribution Report for Fan.jmp



Event Plot

Click the Event Plot disclosure icon to see a plot of the failure or censoring times. For each row in the data table, the Event Plot shows a horizontal line indicating whether the units in the row have been censored. When units have been censored, the line indicates the nature of the censoring.

- The time period when the units in the row are known to be functioning is indicated with a solid line.
- The time period when it is *not* known if the units in the row are functioning is indicated with a dashed line.

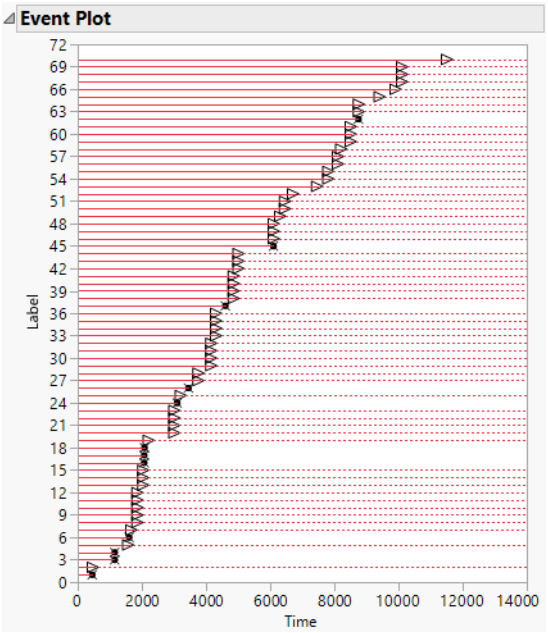
- The line terminates once the units in the row are known to have failed.

Single Time to Event Column

In the Fan.jmp sample data table there is a single Time column indicating failure time. When the failure time is unknown, the value Censored is recorded in the Censor column. All censored units are assumed to be right-censored. Figure 3.7 shows the Event Plot for this data.

Note: To construct the plot in Figure 3.7, select **Help > Sample Data Library** and open Reliability/Fan.jmp. Click the green triangle next to the **Life Distribution - Exponential** script. Click the **Event Plot** disclosure icon.

Figure 3.7 Event Plot for Right-Censored Data



The unit in row 3 failed at Time 1150. Its lifetime is represented by a solid horizontal line that ends at Time 1150. The failure time is marked with an “x”.

The unit in row 5 is right censored. It was last know to be functioning at Time 1560. The time period during which the unit is known to be functioning is represented by a solid horizontal line that ends at Time 1560. At Time 1560, a right arrow is plotted. The line continues as a dashed line, indicating that the failure time is unknown, but greater than 1560.

Two Time to Event Columns

In the Censor Labels.jmp sample data table, there are two columns, Start Time and End Time. Start Time indicates when units in a row were last known to be functioning. End Time indicates when units in that row were first known to have failed.

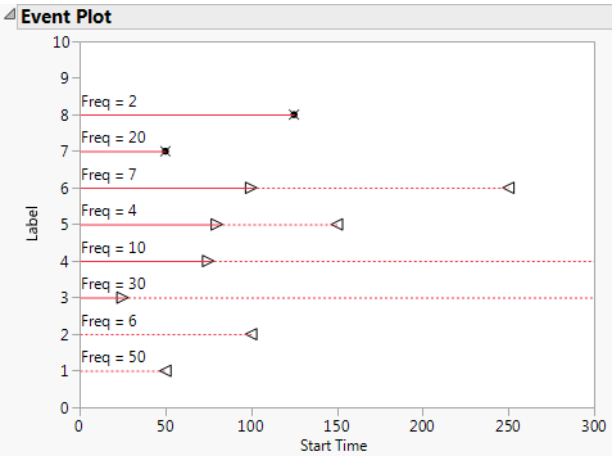
The Start Time and End Time values indicate the following about the units:

- Units in rows 1 and 2 are left censored. They were known to fail before the time in the End Time column, but their exact failure times are unknown.
- Units in rows 3 and 4 are right censored. They were known to be last functioning at the time in the Start Time column, but their failure times are unknown.
- Units in rows 5 and 6 are interval censored. They were known to fail within the interval defined by the Start Time and End Time.
- Units in rows 7 and 8 are not censored. Their failure times are given by the values in the Start Time and End Time columns, which are identical.

Figure 3.8 shows the Event Plot for this data.

Note: To construct the plot in Figure 3.8, select **Help > Sample Data Library** and open Censor Labels.jmp. Click the green triangle next to the **Life Distribution** script.

Figure 3.8 Event Plot for Mixed-Censored Data



The line patterns in the Event Plot represent the various types of censoring:

- The pattern → ····· indicates right censoring. The unit failed after its last inspection.
- The pattern ····· ◁ indicates left censoring. The unit failed after being put on test and prior to the indicated time, but it is not known when it was last functioning.

- The pattern indicates that the unit failed during the time interval marked by the two arrow heads.
- The pattern indicates no censoring. The unit failed at the time marked by the x.

Compare Distributions

The Compare Distributions report lets you fit and compare different failure time distributions. Two lists appear:

Distribution Select a distribution for the response. Different distributions appear based on characteristics of the data. For more information about which distributions are available, see "Available Parametric Distributions".

Scale Select a scale for the probability axis. The probability scale corresponds to the distribution listed to the left of the Scale button. Using this scale, the fitted model is represented by a line. Suppose that you fit a given distribution and then scale the axis using that distribution. If the points generally fall along a line, this indicates that the distribution provides a reasonable fit.

The default plot shows the nonparametric estimates (Kaplan-Meier-Turnbull) for the uncensored data values and their confidence intervals. The confidence intervals are indicated by horizontal blue lines.

Tip: To customize the plot of the nonparametric estimates, select **File > Preferences > Platforms > Life Distribution** and select one or more of the following preferences: **Show Shaded Pointwise Intervals**, **Show Shaded Simultaneous Intervals**, or **Show Staircase Style Function**.

By default, there is a panel at the top of the plot that displays times of right-censored observations.

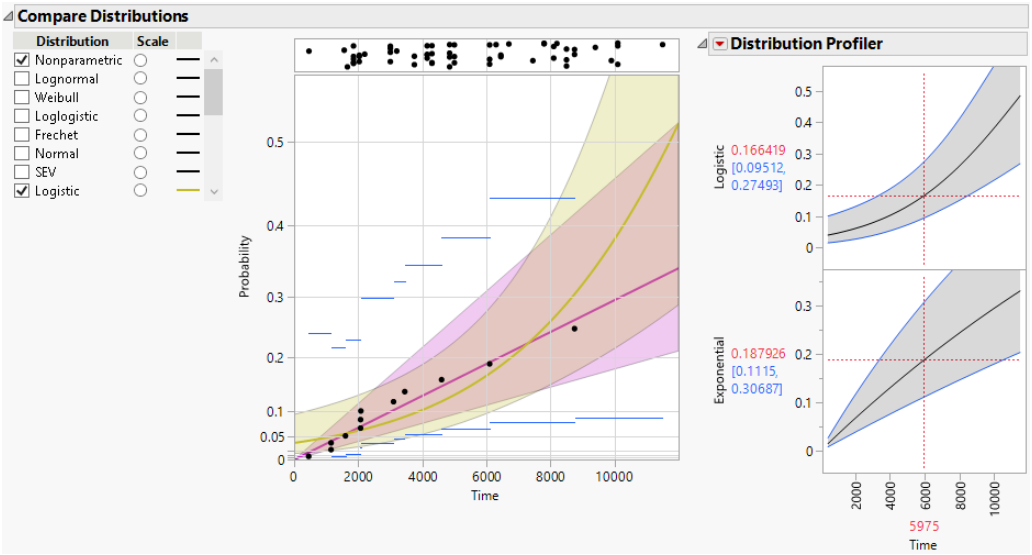
Tip: To hide the panel that contains markers for right-censored observations, select **File > Preferences > Platforms > Life Distribution** and uncheck **Show Markers for Right Censored Observations**.

For each distribution that you select, the Compare Distributions report is updated to show the following:

- the estimated cumulative distribution curve, which appears on the probability plot
- a shaded region that indicates confidence intervals for the cumulative distribution
- a Distribution Profiler that shows the cumulative probability of failure for a given period of time

Figure 3.9 shows an example of the Compare Distributions report. The Logistic (yellow) and Exponential (magenta) distributions are shown. The plot is scaled using the Exponential distribution.

Figure 3.9 Compare Distributions Report and Distribution Profiler



Available Parametric Distributions

This section addresses the distributions available in the Compare Distributions report.

Note: Distributions for the Competing Cause report are covered in "Available Distributions for Competing Cause Compare Distributions Reports".

The available distributions are listed and described in detail in "Parametric Distributions". There are four major groupings of parametric distributions:

- "Basic Failure-Time Distributions"
- "Threshold Distributions"
- "Defective Subpopulation Distributions"
- "Zero-Inflated Distributions"

Tip: To restrict which distributions are available by default, select **File > Preferences > Platforms > Life Distribution** and uncheck the distributions that you do not want to appear. The distributions listed include the Threshold, Defective Subpopulation, Zero-Inflated, LogGenGamma, and GenGamma distributions. By default, all of the distributions are checked and available.

The rules that determine which distributions appear in the Compare Distributions panel depend on the particular implementation. As a general guide, distributions are available if they are not disabled in Preferences and if they are appropriate in the given situation.

Basic Failure-Time Distributions

The basic failure-time distributions are available whenever all failure times are positive. They include the following:

- Lognormal
- Weibull
- Loglogistic
- Fréchet
- Normal
- SEV
- Logistic
- LEV
- Exponential
- LogGenGamma
- GenGamma

Note: When there are negative or zero failure times, only the Normal, SEV, Logistic, LEV, and LogGenGemma are available.

Threshold Distributions

The threshold (TH) distributions are always available. Threshold distributions are log-location-scale distributions with threshold parameters. The threshold parameter shifts the distribution away from 0. These distributions assume that all units survive until the threshold value. Threshold distributions are useful for fitting moderate to heavily shifted distributions. The threshold distributions are the following:

- TH Lognormal
- TH Weibull
- TH Loglogistic
- TH Fréchet

Defective Subpopulation Distributions

The defective subpopulation (DS) distributions are available when all failure times are positive. These distributions are useful when only a fraction of the population has a particular defect leading to failure. Use the DS distribution options to model failures that occur on only a subpopulation. The DS distributions are the following:

- DS Lognormal
- DS Weibull
- DS Loglogistic
- DS Fréchet

Zero-Inflated Distributions

When the time-to-event data contain zero as the minimum value in the Life Distribution platform, the following zero-inflated distributions are available:

- Zero-Inflated Lognormal (ZI Lognormal)
- Zero-Inflated Weibull (ZI Weibull)
- Zero-Inflated Loglogistic (ZI Loglogistic)
- Zero-Inflated Fréchet (ZI Fréchet)

Zero-inflated distributions are used when some proportion of units fails at time zero. When the data contain more zeros than expected by a standard model, the number of zeros is inflated.

Zero-Failure Data

In the case of zero-failure data, none of the above distributions are available by default. To obtain Bayesian fits for those distributions where the Bayesian Estimate option is available, select **File > Preferences > Platforms > Life Distribution** and uncheck **Weibayes Only for Zero Failure Data**. See "Weibayes Only for Zero Failure Data".

Parametric Distributions That Allow Bayesian Estimation

Bayesian estimation is available for the following parametric distributions:

- Lognormal
- Weibull
- Loglogistic
- Fréchet
- Normal
- SEV

- Logistic
- LEV

A list of distributions that are available as priors for hyperparameters of these distributions is given in "Prior Distributions for Bayesian Estimation".

Life Distribution: Statistics

The Statistics report includes the following sub-reports:

- "Model Comparisons"
- "Summary of Data"
- "Nonparametric Estimate"
- "Parametric Estimate - <Distribution Name>" (one report appears for each distribution that you select in the Compare Distributions report)

Model Comparisons

The Model Comparisons report provides the AICc, -2Loglikelihood, and BIC statistics for each fitted distribution. Smaller values of each of these statistics indicate a better fit. For more information about these statistics, see *Fitting Linear Models*.

Initially, the rows are sorted by AICc. To change the statistic used to sort the report, click the Life Distribution red triangle and select **Comparison Criterion**. See "Life Distribution Report Options" for more information about this option.

Summary of Data

The Summary of Data report shows the total number of units observed, the number of uncensored units, and the numbers of right-censored, left-censored, and interval-censored units.

Nonparametric Estimate

The Nonparametric Estimate report shows nonparametric estimates for each observation. For right-censored data specified as a single Time to Event column, the report gives the following:

Midpoint Estimate Midpoint-adjusted Kaplan-Meier estimates.

Lower 95%, Upper 95% Pointwise 95% confidence intervals. You can change the confidence level by selecting Change Confidence Level from the report options.

Simultaneous Lower 95% (Nair), Simultaneous Upper 95% (Nair) Simultaneous 95% confidence intervals. You can change the confidence level by selecting Change Confidence Level from the report options. See Nair (1984) and Meeker and Escobar (1998).

Kaplan-Meier Estimate Standard Kaplan-Meier estimates.

If failure times are represented by two Time to Event columns, the report gives Turnbull estimates (in a column called Estimate), pointwise confidence intervals, and simultaneous confidence intervals (Nair).

See "Nonparametric Fit" for more information about nonparametric estimates.

Parametric Estimate - <Distribution Name>

A report called Parametric Estimate - <Distribution Name> appears for each distribution that is fit. The report gives the distribution's parameter estimates, their standard errors, and confidence intervals. The criteria that appear in the Model Comparisons report are shown under Criterion.

Note: Whenever an estimate of the mean is provided, its confidence interval is computed as a Wald interval even if you select Likelihood as the Confidence Interval Method in the launch window. In this case, the notation Mean (Wald CI) appears in the Parameter column to indicate that the confidence interval for the mean is a Wald interval.

For more information about how the distributions are parametrized, see "Parametric Distributions".

The Parametric Estimate report contains the following reports:

- "Covariance Matrix"
- "Profilers"
- Additional reports can be added by selecting report options from the Parametric Estimate red triangle menu. These include the Fix Parameter, Bayesian Estimates, Custom Estimation (Estimate Probability, Estimate Quantile), and Mean Remaining Life reports. See "Parametric Estimate Options".

Covariance Matrix

For each distribution, the Covariance Matrix report shows the covariance matrix for the estimates.

Profilers

Four types of profilers appear for each distribution:

- The Distribution Profiler shows cumulative failure probability as a function of time.

- The Quantile Profiler shows failure time as a function of cumulative probability.
- The Hazard Profiler shows the hazard rate as a function of time.
- The Density Profiler shows the density function for the distribution.

The profilers contain the following red triangle menu options:

Confidence Intervals The Distribution, Quantile, and Hazard profilers show Wald-based confidence curves for the plotted functions. This option shows or hides the confidence curves.

Reset Factor Grid Displays a window for each factor enabling you to enter a specific value for the factor's current setting, to lock that setting, and to control aspects of the grid. See *Profilers*.

Factor Settings Provides a menu that consists of several options. See *Profilers*.

Note: The confidence intervals provided in the profilers are based on the Wald method even if the Likelihood Confidence Interval Method is selected in the launch window. This is done to reduce computation time.

Parametric Estimate Options

The Parametric Estimate red triangle menu contains the following options:

Save Probability Estimates Saves the estimated failure probabilities and confidence intervals to the data table.

Save Quantile Estimates Saves the estimated quantiles and confidence intervals to the data table.

Save Hazard Estimates Saves the estimated hazard values and confidence intervals to the data table.

Show Likelihood Contour Shows or hides a contour plot of the log-likelihood function. If you have selected the Weibull distribution, a second contour plot appears for the alpha-beta parameterization. This option is available only for distributions with two parameters.

Show Likelihood Profiler Shows or hides a profiler of the log-likelihood function. This option is not available for the threshold (TH) distributions.

Fix Parameter Opens a report where you can specify the value of parameters. Enter your fixed parameter values, select the appropriate check box, and then click **Update**. JMP re-estimates the other parameters, covariances, and profilers based on the new parameters, and shows them in the Fix Parameter report. A distribution profiler of the unconstrained model is shown below the distribution profiler for the fixed parameter

model. For an example in a competing cause situation, see "Specify a Fixed Parameter Model as a Distribution for a Cause".

For the Weibull distribution, the **Fix Parameter** option lets you select the Weibayes method. For an example, see "Weibayes Estimates". The Weibayes option is not available for interval-censored data.

Bayesian Estimates Performs Bayesian estimation of parameters for certain distributions based on three methods of specifying prior distributions (Location and Scale Priors, Quantile and Parameter Priors, and Failure Probability Priors). See "Bayesian Estimation - <Distribution Name>". This option is available only for the following distributions: Lognormal, Weibull, Loglogistic, Fréchet, Normal, SEV, Logistic, LEV.

Custom Estimation Provides calculators that enable you to predict failure probabilities, survival probabilities, and quantiles for specific time and failure probability values. Each calculated quantity includes confidence intervals, which can be two-sided or one-sided (in either direction). Two reports appear: Estimate Probability and Estimate Quantile. See "Custom Estimation".

Mean Remaining Life Provides a calculator that enables you to estimate the mean remaining life of a unit. In the Mean Remaining Life Calculator, enter a Time and press **Enter** to see the estimate. Click the plus sign to enter additional times. This calculator is available for the following distributions: Lognormal, Weibull, Loglogistic, Fréchet, Normal, SEV, Logistic, LEV, and Exponential.

Bayesian Estimation - <Distribution Name>

For certain distributions, you can fit Bayesian models. This is done using rejection sampling or a Markov Chain Monte Carlo (MCMC) algorithm. More specifically, the platform attempts a basic rejection sampler. If the rejection sampler produces valid results, these results are reported. If the rejection sampler cannot produce valid results, the platform uses a random walk Metropolis-Hastings algorithm and adds a note to the top of the Bayesian Estimation report. See Robert and Casella (2004).

From the Parametric Estimate - <Distribution Name> report outline, select **Bayesian Estimates**. This opens an outline called Bayesian Estimation - <Distribution Name>. The initial report is a control panel where you can specify the parameters for the priors and control aspects of the simulation.

The following steps describe the workflow:

- Select a prior specification method from the Bayesian Estimation red triangle menu and set values for the parameters of the priors. See "Bayesian Estimation Red Triangle Options".
- Specify the simulation options. See "Bayesian Estimates - Result <N>".
- Select Fit Model to fit a model. See "Bayesian Estimates - Result <N>".

Bayesian Estimation Red Triangle Options

You can choose from the following prior specification methods in the Bayesian Estimation red triangle menu:

Location and Scale Priors Enables you to specify hyperparameters for prior distributions on generic parameters (location and scale parameters). Select the Prior Distribution red triangle menu to select a distribution for each parameter. You can enter new values for the hyperparameters of the priors. The initial values that are provided are estimates consistent with the MLEs. See "Prior Distributions for Bayesian Estimation".

Quantile and Parameter Priors Enables you to specify prior information about a quantile and the scale parameter (or Weibull β if the parametric fit is Weibull). The quantile is defined by the value next to Probability. The default Probability value is 0.10, but you can specify a value that corresponds to the quantile of interest. Specify information about the prior information in terms of Lower and Upper 99% limits on the range of each prior distribution. See Meeker and Escobar (1998). The initial values that are provided are estimates consistent with the MLEs. See "Prior Distributions for Bayesian Estimation".

Failure Probability Priors Enables you to specify prior information about failure probabilities at two distinct time points. You can specify the two time points. The prior distribution for each time point is Beta. You can specify the prior distributions using either of two synchronized approaches:

1. Specify failure probability by estimates and error percentages. The prior information for each Beta prior distribution can be specified using a probability estimate and an estimate error. See Kaminskiy and Krivtsov (2005).
2. Specify failure probability estimate ranges. You can specify the 99% range for the two Beta distributions in the following ways:

– For each failure time, enter an initial value for the Lower and Upper 99% Limits.

– Click the vertical line segments in the graph and drag them to your two time points. Adjust the vertical spread of each marker to specify the 99% limits.

Simulation Options

For any of the prior specification methods that you select in the Bayesian Estimation red triangle menu, the following options appear at the bottom of the panel:

Number of Monte Carlo Iterations Controls the sample size that will be drawn from the posterior distribution after a burn-in procedure.

Random Seed Sets the initial state of the simulation. By default, it is the clock time. The number should be a positive integer greater than 1. If you specify 1, the current clock time is used.

Show Prior Scatter Plot Select this option to draw random samples from the prior distributions and to plot results on a scatter plot. After you select Fit Model, the scatter plot appears in an outline entitled Prior Scatter Plot in the Bayesian Estimates - Results <N> report.

Overlay Likelihood Contour Overlays likelihood-based contours on scatter plots in the Bayesian Estimates Results report.

Fit Model Estimates the posterior lifetime distribution based on prior distributions that JMP fits using the values that you specified. Adds a report entitled Bayesian Estimates - Results <N>, where N is an integer that consecutively numbers the Bayesian Results reports.

Bayesian Estimates - Result <N>

Once you have specified priors using one of the red triangle menu options, select Fit Model. A Bayesian Estimates - Result <N> report is provided for each selection of priors. This report contains these headings:

Priors Documents the specifications that you entered in the Bayesian Estimation report to fit the Bayesian model. The Priors report also specifies the random seed.

Posterior Estimates Shows five marginal statistics that describe the posterior distribution of the generic parameters (location and scale parameters). The marginal statistics are the median, 0.025 quantile (Lower Bound), 0.975 quantile (Upper Bound), mean, and standard deviation computed from the Monte Carlo samples. When the posterior estimates are generated using the Quantile and Parameter Priors specification, this table also includes the posterior estimate of the quantile and Slope β (for the Weibull distribution).

To compute statistics for other derived variables based on the posterior estimates of the generic parameters, click the Export Monte Carlo Samples link.

Prior Scatter Plot Appears when you select Show Prior Scatter Plot before clicking Fit Model. Shows prior scatter plots of parameters or equivalent quantities associated with the prior specification method for the distribution.

Posterior Scatter Plot Shows posterior scatter plots of parameters or equivalent quantities associated with the prior specification method for the distribution.

Profilers Shows two profilers based on samples from the posterior distribution.

The values shown in the Distribution Profiler, at a given time *t*, are calculated as follows:

- For each set of sampled parameter values from the posterior distribution, the value of the cumulative distribution function at time *t* is calculated.
- The predicted value is the median of these calculated values.

– The upper and lower confidence limits are the 0.025 and 0.975 quantiles of these calculated values.

The plot and confidence limits shown in the Quantile Profiler are obtained in a similar fashion. For a given Probability value p, the quantiles corresponding to p are calculated from the distributions associated with the posterior parameter values.

Weibayes Only for Zero Failure Data

In a zero-failure situation, no units fail. All observations are right censored. If you have zero-failure data, it is possible to conduct either Bayesian estimation or Weibayes inference. See "Weibayes Report".

Note: By default, zero-failure data is analyzed using the Weibayes method. If you want to conduct a broader Bayesian analysis on zero-failure data, select **File > Preferences > Platforms > Life Distribution** and uncheck **Weibayes Only for Zero Failure Data**.

Custom Estimation

The Custom Estimation option produces two reports: Estimate Probability and Estimate Quantile. The Estimate Probability report contains a calculator that enables you to predict failure and survival probabilities for specific time values. The Estimate Quantile report contains a calculator that enables you to predict quantiles for specific failure probability values. Both Wald-based and likelihood-based confidence intervals appear for each estimated quantity. The confidence level for these intervals is determined by the Change Confidence Level option in the Life Distribution red triangle menu.

Estimate Probability

In the Estimate Probability calculator, enter a value for Time. Press **Enter** to see the estimates of failure probabilities, survival probabilities, and corresponding confidence intervals. To calculate multiple probability estimates, click the plus sign, enter another Time value in the box, and press **Enter**. Click the minus sign to remove the last entry.

The Estimate Probability calculator contains an option, Side, that enables you to change the form of the intervals. Select one of the following suboptions:

Two Sided Provides two-sided confidence intervals for failure probability and survival probability.

Upper Failure Probability Provides one-sided confidence intervals that contain upper limits for the failure probability and lower limits for the survival probability.

Lower Failure Probability Provides one-sided confidence intervals that contain lower limits for the failure probability and upper limits for the survival probability.

Estimate Quantile

In the Estimate Quantile report, enter a value for Failure Probability. Press **Enter** to see the quantile estimates and corresponding confidence intervals. To calculate multiple quantile estimates, click the plus sign, enter another Failure Probability value in the box, and press **Enter**. Click the minus sign to remove the last entry.

The Estimate Quantile calculator contains an option, Side, that enables you to change the form of the intervals. Select one of the following suboptions:

Two Sided Provides two-sided confidence intervals for quantiles.

Lower Provides one-sided confidence intervals that contain lower limits for quantiles.

Upper Provides one-sided confidence intervals that contain upper limits for quantiles.

Life Distribution Report Options

The Life Distribution red triangle menu contains the following options:

Fit All Distributions Fits all distributions other than the threshold (TH) distributions. The distributions are compared in the Model Comparisons report. See "Compare Distributions".

> **Tip:** Select the **Comparison Criterion** option to change the criterion for finding the best distribution.

Fit All Nonnegative Fits all nonnegative distributions (Exponential, Lognormal, Loglogistic, Fréchet, Weibull, and Generalized Gamma). The distributions are compared in the Model Comparisons report. See "Compare Distributions". Note the following:

- The option does not fit DS or TH distributions.
- If the data have negative values, then the option produces no results.
- If the data have zeros, the option fits the four zero-inflated (ZI) distributions: ZI Lognormal, ZI Weibull, ZI Loglogistic, and ZI Fréchet. For more information about zero-inflated distributions, see "Zero-Inflated Distributions".

Fit All DS Distributions Fits all defective subpopulation (DS) distributions: DS Lognormal, DS Weibull, DS Loglogistic, and DS Fréchet. For more information about defective subpopulation distributions, see "Distributions for Defective Subpopulations".

Fit Mixture Fits a distribution that is a mixture of the distributions other than the threshold (TH) distributions. See "Mixture".

Fit Competing Risk Mixture Fits a competing risk mixture distribution to the data. See "Fit Competing Risk Mixture".

Show Points Shows or hides data points in the probability plot. The Life Distribution platform uses the midpoint estimates of the step function to construct probability plots. When you deselect **Show Points**, the midpoint estimates are replaced by Kaplan-Meier estimates.

Show Event Plot Frequency Label (Appears only if you have specified a Freq variable.) Shows or hides the Frequency label in the Event Plot.

Show Survival Curve Switches between the failure probability and the survival curve on the Compare Distributions probability plot and the Distribution Profiler plots.

Show Quantile Functions Shows or hides a Quantile Profiler that overlays the plots for the selected distributions. The Quantile plot also shows points plotted at time values. The plot appears beneath the Compare Distributions report. If you select distributions in any of the Compare Distributions, Quantile Profiler, and Hazard Profiler plots, they appear in the other two plots.

Show Hazard Functions Shows or hides a Hazard Profiler that overlays the plots for the selected distributions. The plot appears above the Statistics report. If you select distributions in any of the Compare Distributions, Quantile Profiler, and Hazard Profiler plots, they appear in the other two plots.

Show Statistics Shows or hides the Statistics report. See "Life Distribution: Statistics".

Tabbed Report Shows graphs and data on individual tabs rather than in the default outline style.

Show Confidence Area Shows or hides the shaded confidence regions in the plots.

Interval Type Determines the type of confidence interval shown for the Nonparametric fit in the Compare Distributions plot. Select either pointwise or simultaneous confidence intervals.

Change Confidence Level Enables you to change the confidence level for the entire platform. All plots and reports update accordingly.

Comparison Criterion Enables you to select the criterion used to rank models in the Model Comparison report. For all three criteria, smaller values indicate better fit. Burnham and Anderson (2002) and Akaike (1974) discuss using AICc and BIC for model selection. See *Fitting Linear Models*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

If you have specified a By variable, separate Life Distribution reports appear for each level of the By variable.

Save By Group Results For each By group, saves the estimates that appear in all Parametric Estimate reports for that group as a separate row in a new table.

Do Same Analyses for All Groups Applies all of the selected options for the current group to all other By groups.

Mixture

The Fit Mixture option adds the Mixture outline to the report where you can fit a mixture distribution to the data. For an example, see "Fit Mixture Example".

The mixture distribution's probability function F(x) is defined as follows:

$$F(x) = \sum_{i=1}^{k} w_i F_i(x)$$

where $F_i(x)$ is one of the supported distributions, k is the number of components in the mixture, and the w_i are positive weights that sum to 1. The Fit Mixture option attempts to identify clusters of observations that are drawn from each of the component distributions, $F_i(x)$. It estimates the parameters of the mixture and the probability that an observation is drawn from any given component.

Model Fit and Mixture Starting Value Methods

The fitting methodology is based on assumptions about the underlying clusters, called the Starting Value Method. Suppose that you designate k distributions. There are three Starting Value Methods:

- Single Cluster assumes that all observations are affected by all of the ingredient distributions to some extent. None of the densities stand out as affecting only a portion of the observations.

- Separable Clusters assumes that the ingredient distributions affect some observations more profoundly than others. For separable clusters, each of the *k* densities has an identifiable mode and defines a cluster.
- Overlapping Clusters assumes a situation that is intermediate between Single Cluster and Separable Clusters. Some densities stand out, but others jointly affect a portion of the observations. In this case, there are *m* clusters in the data, where *m* is less than *k*, the total number of densities.

The fitting process consists of these steps:

1. Clusters of observations are defined.
2. Assignment of clusters to densities is based on the Starting Value Method:
   - For Separable Clusters, the highest likelihood assignment of clusters to the specified ingredient densities is determined by examining the possible permutations.
   - For Overlapping Clusters, the highest likelihood assignment of clusters to the specified ingredient densities is determined by examining the possible permutations of clusters and combinations of observations.

Note: Suppose that you fit a model using a given Starting Value Method and then select another Starting Value Method. If a better fit based on the likelihood value cannot be achieved, no new model is added.

Mixture Control Panel

The control panel consists of these items:

Ingredient Lists distributions that you can use as components of the fitted mixture distribution.

Quantity Select the number of components in the mixture distribution that have the given distribution. The sum of the Quantity values is *k*, the number of densities in the mixture.

Starting Value Methods Select a method that reflects your assumptions about the mixture. See "Model Fit and Mixture Starting Value Methods".

Overlay Shows the nonparametric estimates (Kaplan-Meier-Turnbull) for the uncensored data values. When you fit a mixture, the plot is updated to show the model and 95% level confidence bands. The confidence level for these bands is determined by the Change Confidence Level option in the Life Distribution red triangle menu. A Legend appears to the right of the plot.

Go Click **Go** to fit the desired mixture. The Model List is updated with the model that you fit, and a report with the name of the mixture model is added.

Fit Mixture Reports

Model List

The Model List report lists the mixture distributions that you fit. The report provides the number of parameters, the number of actual observations, and the AICc, -2Loglikelihood, and BIC statistics for each mixture distribution. For more information about these statistics, see *Fitting Linear Models*.

Note the following:

- Smaller values of each of these statistics indicate a better fit.
- The rows are sorted by AICc.
- The **Comparison Criterion** red triangle option does not affect the order of models in the Model List.
- The AICc, -2Loglikelihood, and BIC statistics also appear in the Model Comparisons table. This enables you to compare mixture distribution to other distributions for your data. See "Model Comparisons".

Mixture Reports

The Model List report is followed by reports for each of the mixture distributions that you have fit. The title of each report describes the corresponding mixture using the specified ingredients and their quantities. The report lists the parameters, their estimates, standard errors, and 95% Wald confidence intervals. These intervals are not affected by the selection of Likelihood as the Confidence Interval Method in the launch window.

Parameter estimates are given for each distribution in the mixture. The Parameter column also includes parameters called Portion <i>, where i = 1, 2,..., k-1. These are estimates of the weights w_i for the mixture. Since the weights sum to 1, the k^{th} weight can be computed from the first k - 1 weights.

Density Overlay Plot

The Density Overlay plot shows estimates of the density functions for each of the components in the mixture. A legend to the right of the plot enables you to select which density functions appear.

Mixture Report Options

The red triangle menu contains the following options:

Remove Removes the model report and the entry for the model in the Model List.

Show Profilers Shows four types of profilers for the combined mixture distribution *F*. See "Mixture Profiler Options" for a description of their red triangle options.

- The Distribution Profiler shows cumulative failure probability as a function of time.
- The Quantile Profiler shows failure time as a function of cumulative probability.
- The Hazard Profiler shows the hazard rate as a function of time.
- The Density Profiler shows the density function for the distribution.

Save Predictions For each mixture density, saves a column to the data table containing the probability that an observation belongs to that density. For the formulas used in the calculation, see "Fit Mixture Save Predictions Formulas".

Mixture Profiler Options

The profilers for each mixture report contain the following red triangle options:

Confidence Intervals The Distribution, Quantile, and Hazard profilers show 95% Wald-based confidence curves for the plotted functions. This option shows or hides the confidence curves. The confidence level for these curves is determined by the Change Confidence Level option in the Life Distribution red triangle menu.

Note: To reduce computation time, the confidence intervals provided in the profilers are based on the Wald method, even if the Likelihood Confidence Interval Method is selected in the launch window.

Reset Factor Grid Displays a window for each factor enabling you to enter a specific value for the factor's current setting, to lock that setting, and to control aspects of the grid. See *Profilers*.

Factor Settings Provides a menu that consists of options relating to profiler settings, scripts, and linking profilers. See *Profilers*.

Fit Competing Risk Mixture

The Fit Competing Risk Mixture option enables you to fit competing risk mixture distributions. It estimates the probability that a given observation fails due to the cause represented by each of the component mixture distributions. For an example, see "Fit Competing Risk Mixture Example".

The competing risk mixture probability distribution function *F(x)* is defined as follows:

$$F(x) = 1 - \prod_{i=1}^{k} (1 - F_i(x))$$

where $F_i(x)$ represents the cumulative failure distributions for the i^{th} risk and k is the number of components (or risks) in the mixture. The Fit Competing Risk Mixture option attempts to identify clusters of observations that are drawn from each of the component distributions, $F_i(x)$. It estimates the parameters of the mixture and the probability that an observation is drawn from any given component.

The Competing Risk Mixture report is structured in a fashion that mirrors the Mixture report. See "Mixture". However, the Fit Competing Risk Mixture reports do not show a Density Overlay plot. They show a Distribution Overlay plot instead.

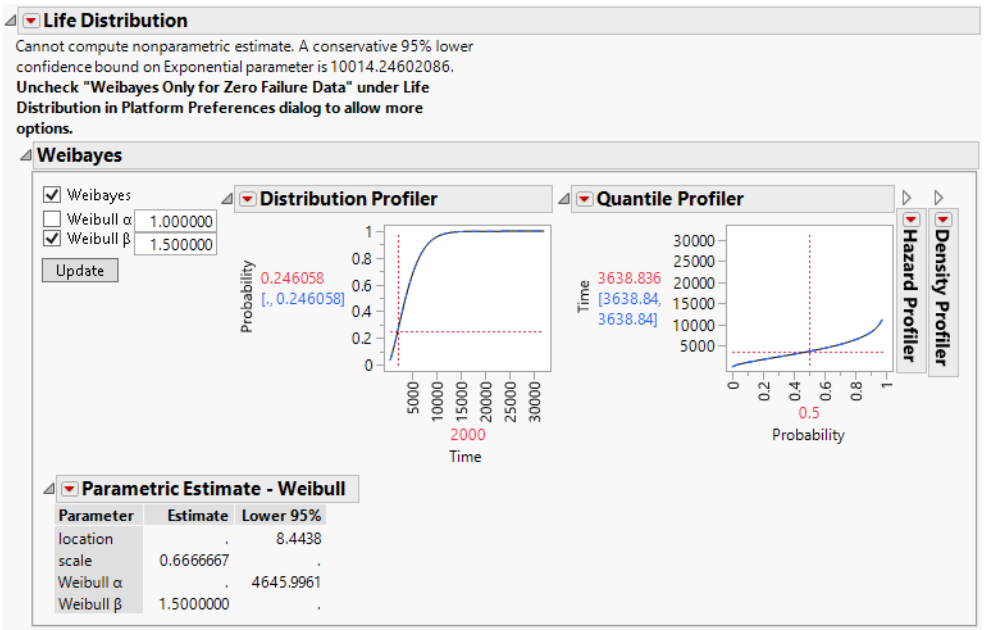
Distribution Overlay Plot

The Distribution Overlay plot shows the cumulative distribution functions for each of the mixture components and for the combined mixture (Aggregated). A legend to the right of the plot enables you to select which cumulative distribution functions appear.

Weibayes Report

If you have data with zero failures (right-centered) and you have not turned off the preference **Weibayes Only for Zero Failure Data**, a special Weibayes report appears. Figure 3.10 shows the Weibayes report for the Weibayes No Failures.jmp sample data table, found in the Reliability folder.

Figure 3.10 Weibayes Report



The analysis begins by assuming an exponential model. The note at the top of the report provides a lower confidence bound for the parameter of an exponential distribution. This lower confidence bound is computed using the method described in Meeker and Escobar (1998, sec. 7.7).

The Weibayes section of the report conducts the Weibayes analysis. For a description of the procedure, see Nelson (1985).

To obtain Weibayes estimates, make sure that Weibayes and Weibull β options are selected. Change the Weibull β value and click Update. The estimates and profilers are updated. The values shown in the profilers use the conservative confidence bound. For an example, see "Weibayes Example for Data with No Failures".

Note: If you deselect the Weibayes option, JMP shows the fixed parameter MLE.

When your data table contains at least one failure, a full Life Distribution report appears, but the maximum likelihood inference might not produce useful results. In this instance, a Weibayes analysis might be preferable. For an example, see "Weibayes Example for Data with One Failure".

Note: To conduct Bayesian inference using the usual Life Distribution options when you have zero-failure data, select **File > Preferences > Platforms > Life Distribution** and deselect **Weibayes Only for Zero Failure Data** before launching the analysis.

Competing Cause Report

Tip: If you find that the report window is too long, select **Tabbed Report** from the Competing Cause red triangle menu.

The Competing Cause report appears if you have assigned a Failure Cause column in the launch window. Use this report to analyze the competing causes to determine which causes are influential. For an example of this report, see "Omit Competing Causes". For technical details, see "Competing Cause Details".

The Competing Cause report contains the following content and options:

- "Cause Combination"
- "Competing Cause: Statistics"
- "Individual Causes"
- "Competing Cause Report Options"

Competing Cause Workflow

Follow these steps to facilitate your use of the Competing Cause report:

1. For convenience, click the Competing Cause red triangle and select the Tabbed Report option.
2. Select the Individual Causes tab. For each failure cause, use the options in its individual Life Distribution outline to select a distributional fit. See "Individual Causes".
3. Select the Cause Combination tab. For each failure cause, specify the desired distribution in each **Distribution** list. See "Cause Combination".
4. Click **Update Model**.
5. Select the Statistics tab to explore and save results for the model. See "Competing Cause: Statistics".

Note: Customizations made to the competing cause model report in the Statistics outline might be lost if you change the model and click **Update Model** again.

Competing Cause Model

In a competing cause situation, the aggregated failure function can be written as follows:

$$F(x) = 1 - \prod_{i=1}^{k} [1 - F_i(x)]$$

where $F_i(x)$ is the cumulative failure distribution for the i^{th} cause and k is the total number of causes. The function $F_i(x)$ is cause-specific. It reflects the probability of failure due to cause i alone and does not account for other causes of failure.

An alternative formulation is defined as follows:

$$F(x) = \sum_{i=1}^{k} \tilde{F}_i(x)$$

where each $\tilde{F}_i$ is a monotone increasing function with values in the interval [0, 1]. The function $\tilde{F}_i$ is called a subdistribution. This form of the aggregated failure distribution is used to predict proportions of failures that are associated with individual causes, accounting for failure due to other causes.

Cause Combination

The Cause Combination report lets you fit and compare different failure time distributions ($F_i(x)$) for the various causes. Different distributions are available based on selections that you have made in the launch window. Negative and zero-failure times are not allowed.

The default plot is based on a Linear scale and shows the following:

- Nonparametric estimates (Kaplan-Meier-Turnbull) for the uncensored data values and their confidence intervals. The confidence intervals are represented by horizontal blue lines.
- Fits of cumulative failure distributions ($F_i(x)$) to each of the causes. The initial distribution is the one that you selected in the launch window. If you selected Individual Best, the best distribution is computed for each group and these fits are shown. (This selection can be time-intensive.) If you selected Manual Pick, the initial Distribution is Nonparametric for all groups and nonparametric fits are shown. A legend appears to the right of the plot.
- The Aggregated cumulative failure distribution, $F(x)$, represented by a black line. This function is computed based on the selected cause distribution. If a nonparametric distribution is specified for a cause, the aggregated cumulative failure distribution extends only as far as the final time observation for that cause.

As you interact with the report, statistics for the aggregated model are re-evaluated.

The Cause Combination report contains these elements:

Scale Specifies the probability scale for the plot's vertical axis. "Change the Scale" illustrates how changing the scale affects the distribution fit.

Omit Check a box to remove the fit for the corresponding cause. Use this when a particular cause has been fixed. The Aggregated model updates to reflect the removal of the failures due to that cause. "Omit Competing Causes" illustrates the effect of omitting causes.

Cause Lists the causes in the Cause column.

Distribution Lists the available distributions for each cause. To change the distribution for a specific failure cause, select the distribution from the **Distribution** list. Click **Update Model** to show the new distribution fit on the plot. The Cause Summary report is also updated.

Count Gives the number of observations with the given failure cause.

Update Model Shows the selected distributions in the plot; updates the Cause Summary report with the selected models; adds the selected model as the most recent one in the Individual Causes report.

Competing Cause: Statistics

The Statistics report for Competing Cause contains the following reports:

- "Cause Summary Report"
- "Profilers"

Cause Summary Report

The Cause Summary report shows information for the fit defined by the current selections under Distribution in the Cause Summary report. The report shows the number of failures for each cause and the parameter estimates for the distribution fit to each failure cause. When you change distribution fits in the Cause Combination report and click Update Model, the Cause Summary report is updated.

The following information is provided:

- The Cause column shows either labels of causes or the censor code.
- The Counts column lists the number of failures for each cause.
- A numerical entry in the Counts column indicates that the cause has enough failure events to consider. A cause with fewer than two events is considered right censored. The column also identifies missing causes.
- The Distribution column specifies the selected distribution for each cause.
- Depending on the selected distributions, various columns display the parameters of the distributions:
  - The location column specifies location parameters for various distributions.
  - The scale column specifies scale parameters for various distributions.
  - The Weibull α and Weibull β columns show Weibull estimates of alpha and beta.
  - Other columns show parameters of other selected distributions.
  - A Convergence column appears if there are convergence issues.

The Cause Summary red triangle menu options enable you to save the probability, quantile, hazard, and density estimates for the aggregated failure distribution to the data table.

Profilers

The Distribution, Quantile, Hazard, and Density profilers help you visualize the aggregated failure distribution. The Distribution, Quantile, and Hazard profilers show 95% level confidence bands. See "Profilers".

Note: If a nonparametric distribution is specified for a cause, the Hazard and Density profilers are not provided. Also, confidence limits are not provided in the Distribution and Quantile profilers.

Individual Causes

The Individual Causes report contains Life Distribution - Failure Cause: <Distribution Name> reports for each of the individual causes. Each Life Distribution - Failure Cause: <Distribution Name> report shows plots and distributional fit statistics for the individual failure cause indicated in the report title.

Whenever you click Update Model in the Cause Combination report, any new cause distribution that you select is added to the Life Distribution - Failure Cause: <Distribution Name> report for that cause. In the Life Distribution - Failure Cause <Distribution Name> report, the following occur:

- The distribution is selected in the Compare Distributions report.
- The distribution is added to the Model Comparisons report.
- A Parametric report is added for that distribution.

Each Life Distribution - Failure Cause <Distribution Name> report is a Life Distribution: Statistics report as described in "Life Distribution: Statistics". However, all confidence intervals are Wald intervals. These intervals are not affected by the selection of Likelihood as the Confidence Interval Method in the launch window.

Available Distributions for Competing Cause Compare Distributions Reports

If you specify a Failure Cause in the launch window, you can specify which groupings of distributions and models you want to allow in the resulting Compare Distributions reports for Individual Causes. You can select ZI (Zero-Inflated), TH (Threshold), and DS (Defective Subpopulation) distributions. You can also select fixed parameter and Bayesian models.

Note: If you have disallowed any distributions in Preferences, these do not appear. Also, rules that govern which distributions appear for Life Distribution apply. See "Available Parametric Distributions".

Competing Cause Report Options

The Competing Cause red triangle menu contains the following options:

Tabbed Report Shows graphs and data on individual tabs rather than in the default outline style.

Tabbed Report for Individual Causes Shows the Life Distribution - Failure Cause: <Distribution Name> reports in tabs, rather than as a stack of Life Distribution reports.

Show Points Shows or hides data points in the Cause Combination plot. The Life Distribution platform uses the midpoint estimates of the step function to construct probability plots. When you deselect **Show Points**, the midpoint estimates are replaced by Kaplan-Meier estimates.

Show Subdistributions Shows the profiler for each individual cause subdistribution $\tilde{F}_i$. See "Individual Subdistribution Profiler for Cause".

Show Remaining Life Distribution Shows the remaining life distribution through the Distribution Profiler. This is conditional upon the unit surviving through a given time.

Mean Remaining Life Estimates the mean remaining life of a unit, given a survival time. In the Mean Remaining Life Calculator, enter a time value and press **Enter** to see the estimate and confidence limits. Click the plus sign to enter additional times. Click the minus sign to remove the most recent entry.

To obtain a confidence interval, select the **Configuration** option from the Mean Remaining Life Calculator. Check **Use bootstrap to construct confidence intervals**. Enter appropriate values, keeping in mind that computation can be time-intensive. See "Mean Remaining Life Calculator".

Export Bootstrap Results Appears only when a Bayesian model is selected and when **Update Model** has been applied.

Bootstrap Sample Size When Bayesian Estimates or Weibayes results are used for any cause, the confidence limits for aggregated functions that appear in the Distribution Profiler must be simulated using parametric bootstrap. Use this option to specify the number of samples to be used in the bootstrap. See "Specify a Bayesian Model for a Cause".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Individual Subdistribution Profiler for Cause

For a given cause, the Individual Subdistribution Profiler for Cause shows the estimated probability of failure, $\tilde{F}_i$, from that cause at time t. The estimated probability takes into account failures from competing causes. See "Competing Cause Model".

To show a profiler of the subdistribution for each cause, click the Competing Cause red triangle and select **Show Subdistributions**. The Individual Subdistribution Profiler for Cause report appears beneath the other profilers. It consists of a profiler and a calculator.

Note: When you select **Show Subdistributions**, the Cause Combination plot is updated to show the subdistribution functions for all causes.

Select a Cause from the list to the right of the profiler to see its profiler. Options that apply to the profiler are provided in the Individual Subdistribution Profiler for Cause red triangle menu. See "Profilers".

Use the Calculator panel to find values of the subdistribution functions for all causes at one or more Time to Event values. Enter a value for the Time to Event variable. When you press Enter (or click outside the text box), values for each of the causes are updated. To add a Time to Event value, click the plus sign. To remove the most recent value, click the minus sign.

Life Distribution - Compare Groups Report

Tip: If you find that the report window is getting too long, select **Tabbed Report** from the red triangle menu next to Life Distribution - Compare Groups.

If you selected the Compare Groups tab in the launch window, the Life Distribution - Compare Groups report appears. This report compares different groups using a single specified distribution. For example, you might compare Weibull fits for components grouped by supplier. In contrast, the Life Distribution tab compares several fitted distributions for a single group.

You can compare the CDF, Quantile, Hazard, and Density functions. You can also consolidate the probability and quantile predictions of all groups.

For an example of this report, see "Examine the Same Distribution across Groups".

The Compare Groups report can contain the following content and options:

- "Compare Distributions" (no Distribution Profiler)
- "Compare Groups: Statistics"
- "Individual Group"

- "Life Distribution - Compare Groups Report Options"

Compare Groups: Statistics

The Statistics report for Compare Groups contains the following reports:

- "Summary"
- "Group Homogeneity Tests"
- "Model Comparisons"
- "Parameter Estimates"

Summary

The Summary report contains a row for each group and for the combined data. Each row shows the number of units that failed and the numbers that were left, interval, or right censored. Each row also gives the corresponding mean and standard error. For more information about how the mean and standard error are computed, see "Statistical Reports for Survival Analysis".

Group Homogeneity Tests

The Group Homogeneity Tests report contains results of three tests for equality of the hazard functions across groups. The tests differ in how the observations are weighted over time.

Log-Rank The Log-Rank test uses equal weights for all time points.

Peto-Peto The Peto-Peto test uses the survival proportion at each time point for weights.

Wilcoxon The generalized Wilcoxon test uses the number at risk at each time point for weights. This test is referred to as the Gehan test in Klein and Moeschberger (1997).

For each of the above tests, the report includes the chi-square approximation, the associated degrees of freedom, and the *p*-value for each test. A small *p*-value suggests that the groups differ. For more information about these tests and comparisons of failure curves, see Klein and Moeschberger (1997, ch. 7).

Parameter Estimates

The Parameter Estimates outline contains reports entitled Parametric Estimate - <Distribution Name> for each distribution that is fit. For each group, the Parametric Estimate - <Distribution Name> report gives the distribution's parameter estimates and their 95% level confidence intervals. The confidence level for these intervals is determined by the Change Confidence Level option in the Life Distribution - Compare Groups menu.

Individual Group

The tabs within the Individual Group report contain Life Distribution reports for each individual group. For more information about these reports, see "Life Distribution Report" and "Life Distribution Report Options".

Life Distribution - Compare Groups Report Options

Many of the options in the Compare Groups red triangle menu can also be found in the Life Distribution red triangle menu. See "Life Distribution Report Options".

However, the following options are specific to Compare Groups:

Show Quantile Functions Shows or hides the Compare Quantile report. Select a distribution. For each group, a curve is plotted showing the estimated quantiles for the time variable. Confidence bands are displayed. A legend is shown to the right of the plot. Only one distribution can be specified at a time.

Show Hazard Functions Shows or hides the Compare Hazard report. Select a distribution. For each group, a curve is plotted showing the hazard function. Confidence bands are displayed. A legend is shown to the right of the plot. Only one distribution can be specified at a time.

Show Density Functions Shows or hides the Compare Density report. Select a distribution. For each group, the density function is displayed. A legend is shown to the right of the plot. Only one distribution can be specified at a time.

Estimate Probability Adds an Estimate Probability report corresponding to the most recently selected distribution under Compare Distribution. Enter a value for the Time to Event variable in the text box and press **Enter**. To add a Time to Event value, click the plus sign. To remove the most recent value, click the minus sign. See "Estimate Probability Report".

Estimate Quantile (Appears only if Compare Quantile is selected.) Adds an Estimate Quantile report corresponding to the most recently selected distribution under Compare Quantile. Enter a value for the probability of interest in the text box and press **Enter**. To add a Prob value, click the plus sign. To remove the most recent value, click the minus sign. For each group and probability value, the Time to Event quantile and 95% Wald and Likelihood confidence intervals are shown.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Estimate Probability Report

For each group and Time to Event value, this report provides the following:

Midpoint Estimate Midpoint-adjusted Kaplan-Meier estimate of failure by the specified time.

Lower 95%, Upper 95% Pointwise 95% confidence intervals for the probability of failure by the specified time.

Simultaneous Lower 95% (Nair), Simultaneous Upper 95% (Nair) Simultaneous 95% confidence intervals for the probability of failure by the specified time. See Nair (1984) and Meeker and Escobar (1998).

Survival Probability Midpoint-adjusted estimate of survival beyond the specified time.

Survival Probability Lower 95%, Upper 95% Pointwise 95% confidence intervals for the probability of survival beyond the specified time.

Survival Probability Simultaneous Lower 95% (Nair), Simultaneous Upper 95% (Nair)

Simultaneous 95% confidence intervals for the probability of survival beyond the specified time. See Nair (1984) and Meeker and Escobar (1998).

Additional Examples of the Life Distribution Platform

- "Omit Competing Causes"
- "Change the Scale"
- "Examine the Same Distribution across Groups"
- "Weibayes Estimates"
- "Fit Mixture Example"
- "Fit Competing Risk Mixture Example"

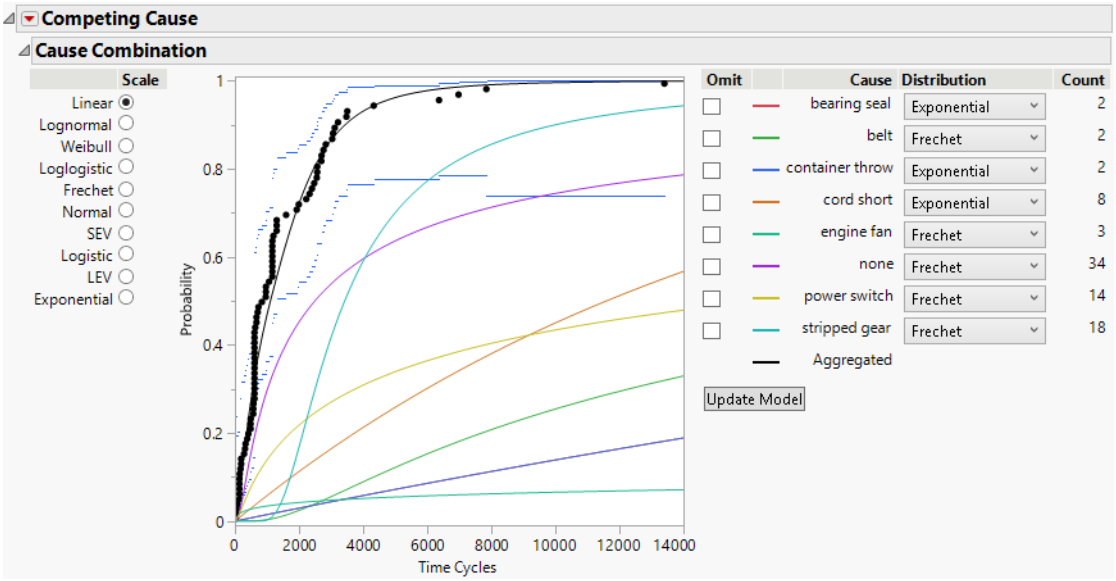
Omit Competing Causes

This example illustrates how to decide on the best fit for competing causes.

1. Select **Help > Sample Data Library** and open Reliability/Blenders.jmp.
2. Select **Analyze** > **Reliability and Survival** > **Life Distribution**.
3. Select Time Cycles and click **Y, Time to Event**.
4. Select Causes and click **Failure Cause**.
5. Select Censor and click **Censor**.
6. Select **Individual Best** as the Distribution.
7. Make sure that **AICc** is the Comparison Criterion.
8. Click **OK**.

   On the Competing Cause report, JMP shows the best distribution fit for each failure cause.

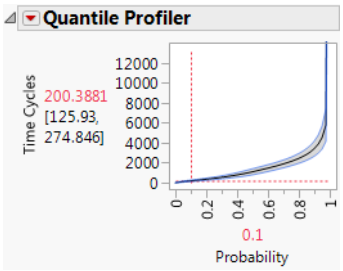
Figure 3.11 Initial Competing Cause Report



9. In the Quantile Profiler, type 0.1 for the probability.

   The estimated time by which 10% of the failures occur is 200.

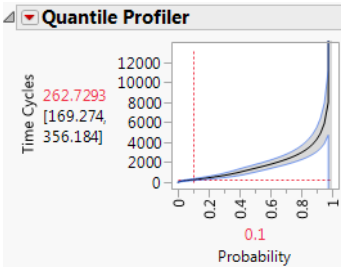
Figure 3.12 Estimated Failure Time for 10% of the Units



10. Select **Omit** for bearing seal, belt, container throw, cord short, and engine fan (the causes with the fewest failures).

    The estimated time by which 10% of the failures occur is now 263.

Figure 3.13 Updated Failure Time



When power switch and stripped gear are the only causes of failure, the estimated time by which 10% of the failures occur increases by approximately 31%.

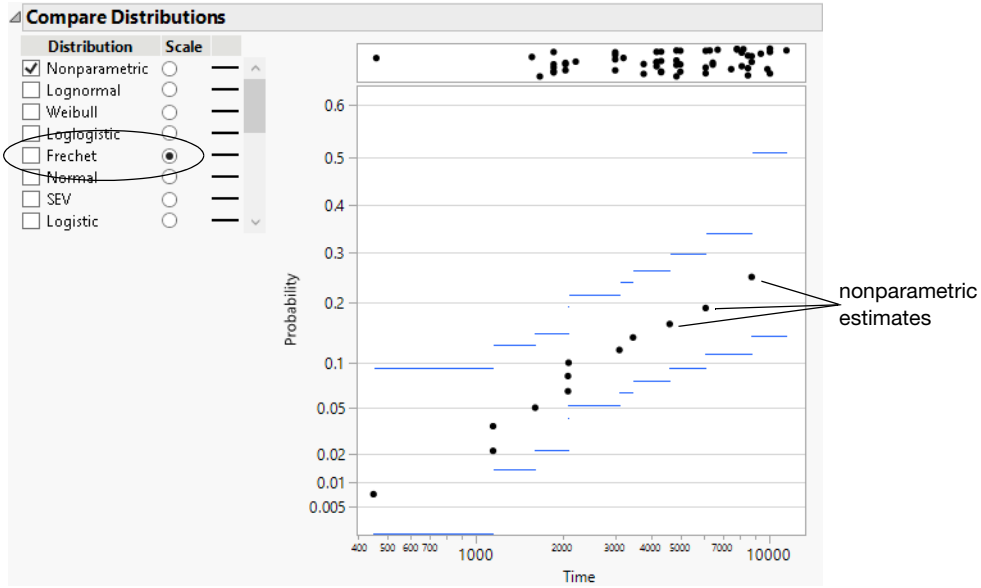
Change the Scale

In the initial Compare Distributions report, the probability and time axes are linear. But suppose that you want to see distribution estimates on a Fréchet scale.[1]

1. Follow step 1 through step 5 in "Example of the Life Distribution Platform".
2. In the Compare Distributions report, select **Fréchet** in the Scale column.
3. Click the Life Distribution red triangle and select **Interval Type > Pointwise**.

1. Using different scales is sometimes referred to as drawing the distribution on different types of *probability paper*.

Figure 3.14 Nonparametric Estimates with a Fréchet Probability Scale

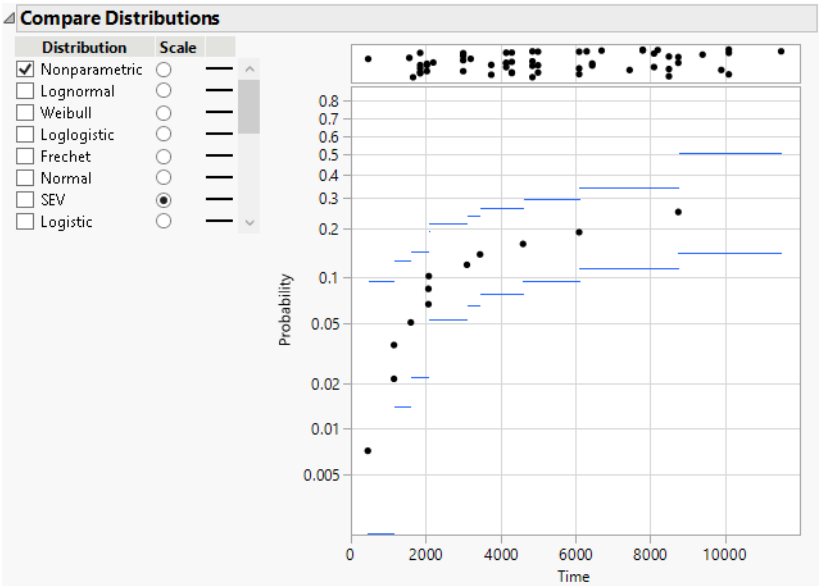


Using a Fréchet scale, the nonparametric estimates approximate a straight line, meaning that a Fréchet fit might be reasonable.

4. Select **SEV** in the Scale column.

The nonparametric estimates no longer approximate a straight line. You now know that the SEV distribution is not appropriate.

Figure 3.15 Nonparametric Estimates with a SEV Probability Scale

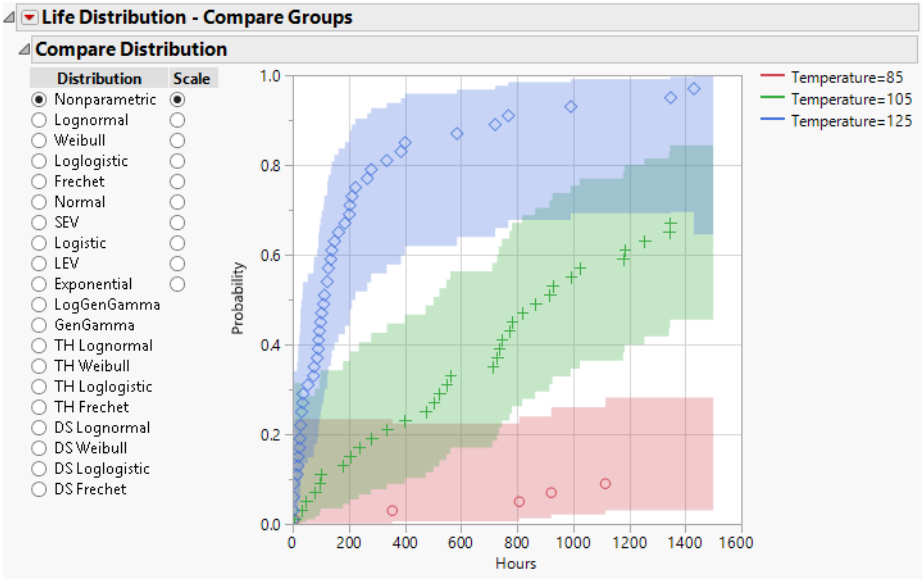


Examine the Same Distribution across Groups

Suppose you want to compare the same distribution across different groups. You want to examine estimates of failure probabilities for a single type of capacitor operating at three different temperatures.

1. Select **Help > Sample Data Library** and open Reliability/Capacitor ALT.jmp.
2. Select **Analyze** > **Reliability and Survival** > **Life Distribution**.
3. Click the Compare Groups tab.
4. Select Hours and click **Y, Time to Event**.
5. Select Temperature and click **Grouping**.
6. Select Censor and click **Censor**.
7. Select Freq and click **Freq**.
8. Click **OK**.

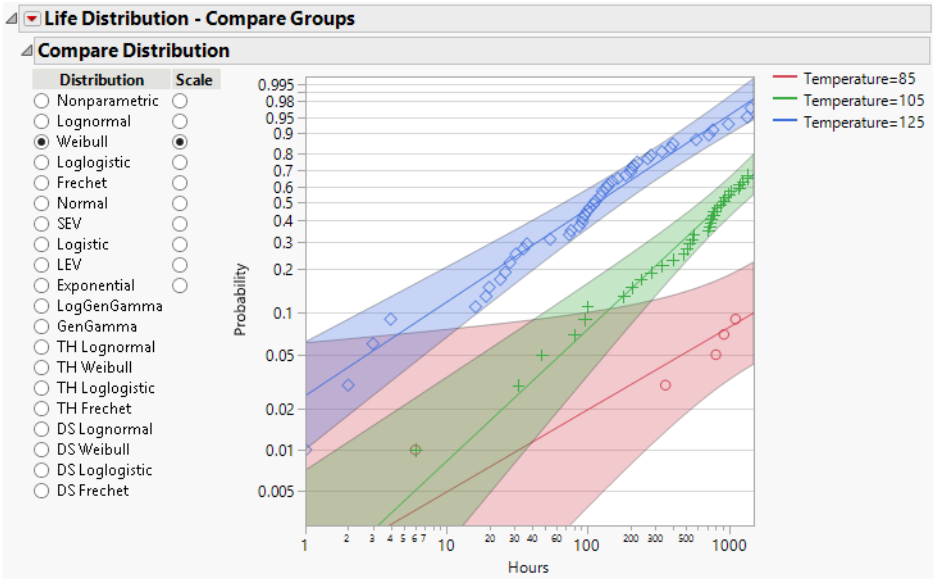
Figure 3.16 Compare Distribution for Groups



The default graph shows the nonparametric estimates. At a higher temperature, the capacitor has a higher probability of failure. You want to try fitting a parametric distribution.

9. Select **Weibull** for Distribution and Scale.

Figure 3.17 Compare Weibull Distribution for Groups



When plotted against a Weibull probability scale, the points come close to following three lines. This indicates that a Weibull distribution provides a reasonable fit for each of the Temperature groups.

Weibayes Estimates

There are two possible ways to obtain a Weibayes analysis:

- You have no failures (all observations are right-censored) and the preference **Weibayes Only for Zero Failure Data** is checked. Then the Weibayes report appears. See "Weibayes Example for Data with No Failures".
- You have few failures. A full Life Distribution report is presented. Fit a Weibull distribution. In the Parametric Estimate - Weibull report, select the Fix Parameter option. Then select the Weibayes option in the Fixed Parameter report. See "Weibayes Example for Data with One Failure".

Weibayes Example for Data with No Failures

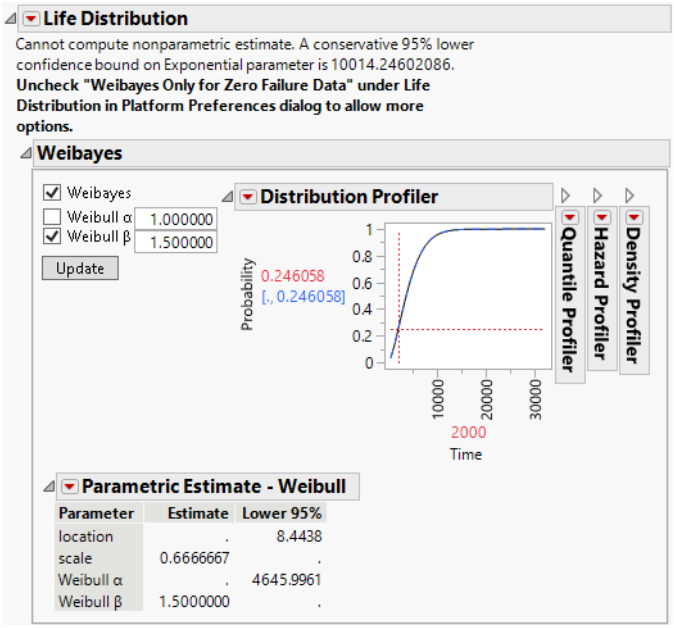
You have data for a product that is mostly reliable. Thirty were tested for 1,000 hours with no failures occurring. You want to predict the failure probability at 2,000 hours.

1. Select **Help > Sample Data Library** and open Reliability/Weibayes No Failures.jmp.
2. Select **Analyze** > **Reliability and Survival** > **Life Distribution**.
3. Select Time and click **Y, Time to Event.**
4. Select Censor and click **Censor**.
5. Select Freq and click **Freq**.
6. Select **Likelihood** as the Confidence Interval Method.
7. Click **OK.**

   A special Life Distribution report appears. **Weibayes** and **Weibull beta** should be selected.
8. Type 1.5 as the known Weibull β value.

   The value 1.5 is considered appropriate for this example.
9. Click **Update**.
10. In the Distribution Profiler, type 2000 for Time.

Figure 3.18 Life Distribution Report for Zero Failures



From the Distribution Profiler, you can see that at 2,000 hours, the conservative probability is 24.6058%. That means that the one-tailed conservative 95% confidence limit for the failure probability is 24.6058%.

Weibayes Example for Data with One Failure

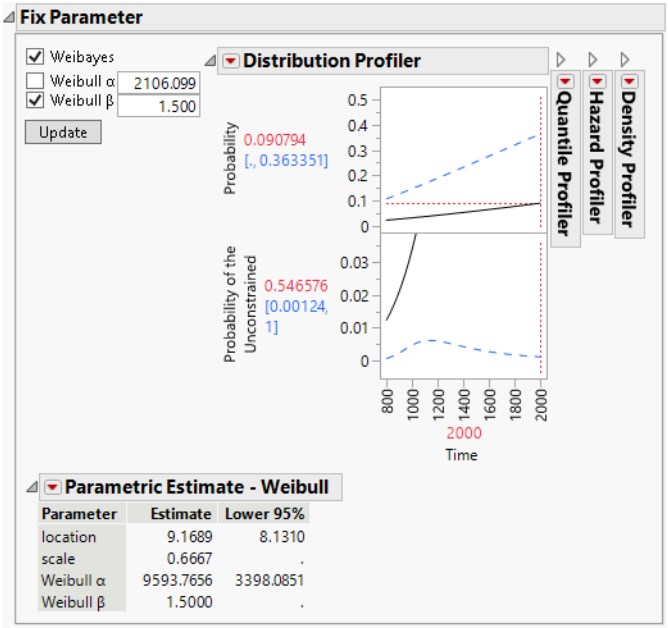
Suppose you have the same data, but this time, one failure occurred at 800 hours. Again, you want to predict the failure probability at 2,000 hours.

1. Select **Help > Sample Data Library** and open Reliability/Weibayes One Failure.jmp.
2. Select **Analyze** > **Reliability and Survival** > **Life Distribution**.
3. Select Time and click **Y, Time to Event.**
4. Select Censor and click **Censor**.
5. Select Freq and click **Freq**.
6. Select **Likelihood** as the Confidence Interval Method.
7. Click **OK.**

   The Life Distribution report appears.
8. Select the **Weibull** distribution in the Compare Distributions plot.
9. Click the red triangle next to Parametric Estimate - Weibull and select **Fix Parameter**.
10. Select **Weibayes** and **Weibull beta** in the Fix Parameter report.

11. Type 1.5 as the known Weibull β value.
12. Click **Update**.
13. In the Distribution Profiler, type 2000 for Time.
14. Hover over the top of the *Y* axis. The cursor becomes a hand. Drag the axis downward until it reaches 0.5 as the top number.

Figure 3.19 Life Distribution Report for One Failure



In the Distribution Profiler, the solid line shows the MLE. The dashed line shows the Weibayes conservative limit. You can see that at 2,000 hours, the conservative probability is 36.3351%. That means that the one-tailed conservative 95% confidence limit for the failure probability is 36.3351%.

Fit Mixture Example

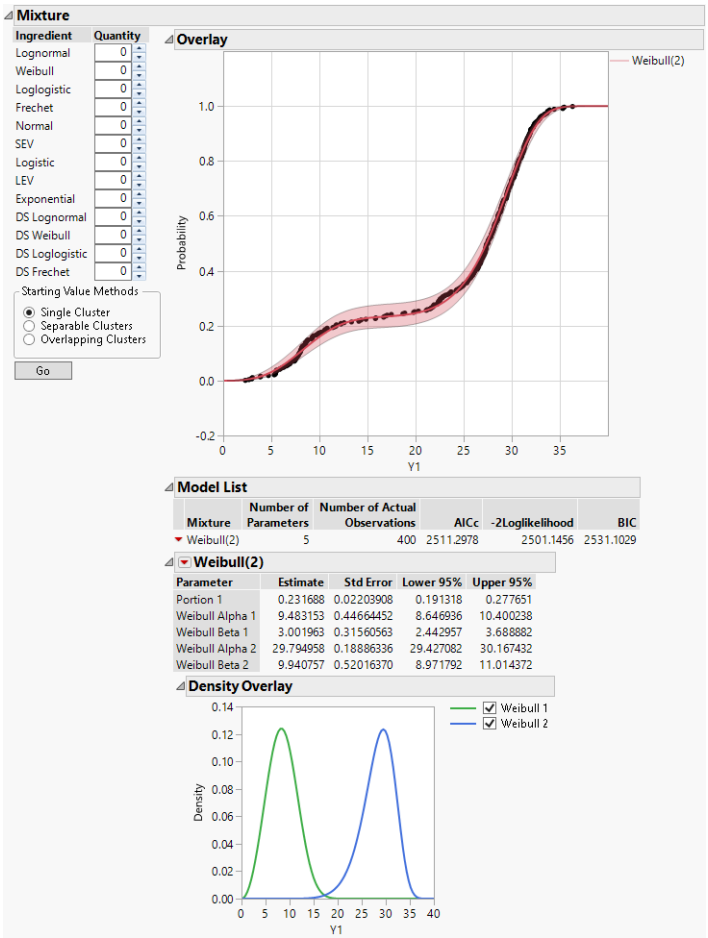
In this example, you fit two mixture distributions and then identify observations belonging to one of the clusters for the second mixture.

Fit Two Mixture Distributions

1. Select **Help > Sample Data Library** and open Reliability/Mixture Demo.jmp.
2. Select **Analyze > Reliability and Survival > Life Distribution**.
3. Select Y1 and click **Y, Time to Event**.

4. Click **OK**.
5. Click the Life Distribution red triangle and select **Fit Mixture**.
6. Type 2 in the **Quantity** box next to **Weibull**.
7. Select **Separable Clusters** in the Starting Value Methods panel.
8. Click **Go**.

Figure 3.20 Fit Mixture for Weibull (2)

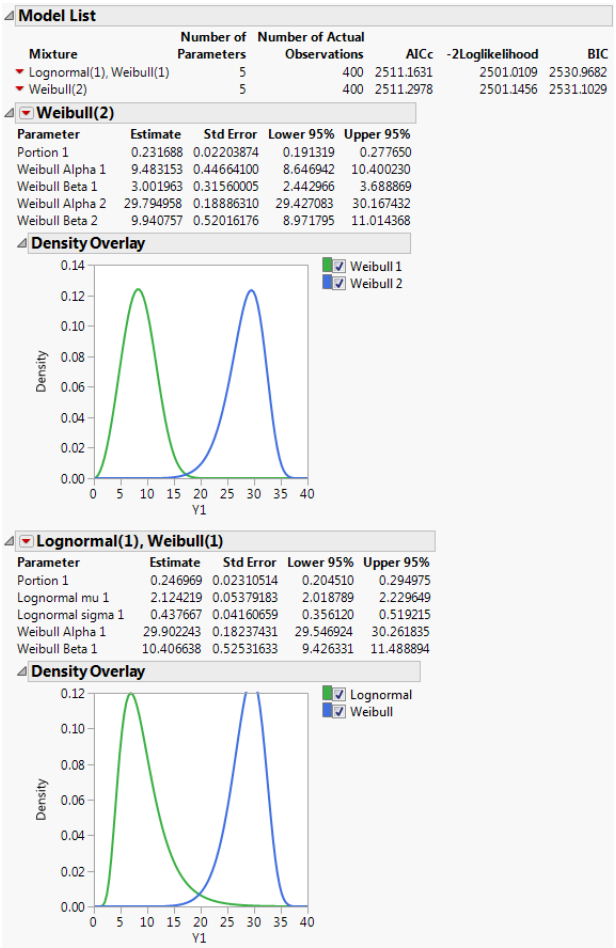


JMP fits a mixture model consisting of two Weibull components. Portion 1 is estimated as 0.231688, indicating that approximately 23% of observations have the Weibull distribution with alpha = 9.483153 and beta = 3.001963. The remaining 77% are estimated to come from the second Weibull distribution.

To compare this model to another, you can change the Ingredient selections and the Quantity of components.

9. Type 1 next to **Lognormal** and 1 next to **Weibull**.
10. Click **Go**.

Figure 3.21 Fit Mixture for Lognormal(1), Weibull(1)



The Overlay plot is updated to show both mixture models. The plots and statistics in the Model List indicate that the Lognormal(1), Weibull(1) mixture seems to give a fit that is very similar to the Weibull(2) mixture.

Identify Observations Belonging to a Cluster

1. Click the red triangle next to Lognormal(1), Weibull(1) and select **Save Predictions**.

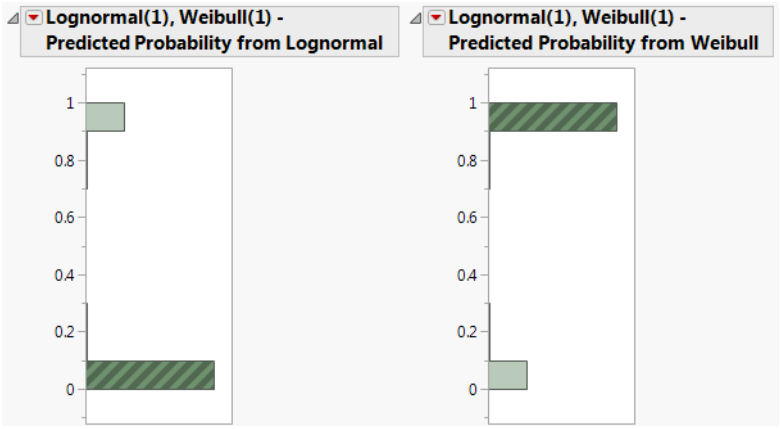
   Two columns are added to the data table:

   – Lognormal(1), Weibull(1) - Predicted Probability from Lognormal

– Lognormal(1), Weibull(1) - Predicted Probability from Weibull

2. Select **Analyze > Distribution**.
3. Select the two new columns from the Select Columns list and click **Y, Columns**.
4. Check **Histograms Only**.
5. Click **OK**.
6. In the histogram for Lognormal(1), Weibull(1) - Predicted Probability from Weibull, click in the bar corresponding to the value near 1.

Figure 3.22 Histograms for Mixture Probabilities



In the data table, the 297 corresponding rows are selected. These are the observations that are likely to have come from the Weibull distribution with parameters alpha = 29.90 and beta = 10.41.

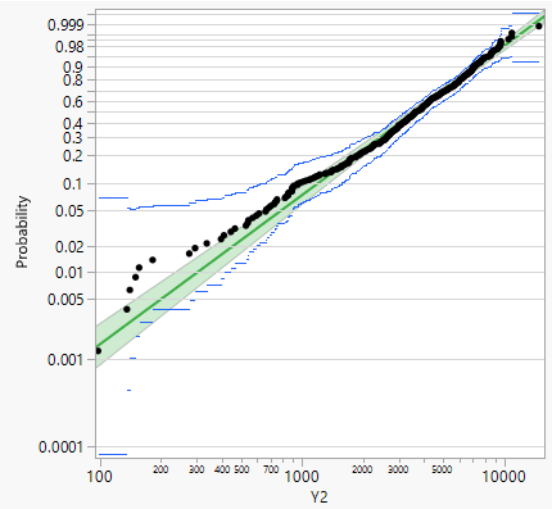
Fit Competing Risk Mixture Example

In this example, you compare the fit of a single Weibull distribution to a competing risk mixture distribution with two Weibull fits.

1. Select **Help > Sample Data Library** and open Reliability/Mixture Demo.jmp.
2. Select **Analyze > Reliability and Survival > Life Distribution**.
3. Select Y2 and click **Y, Time to Event**.
4. Click **OK**.
5. In the Compare Distributions report, select **Weibull** distribution and the corresponding **Scale** radio button.

   A probability plot for a single Weibull distribution fit appears. Note that the fit is not very good in the lower part of the range of Y2.

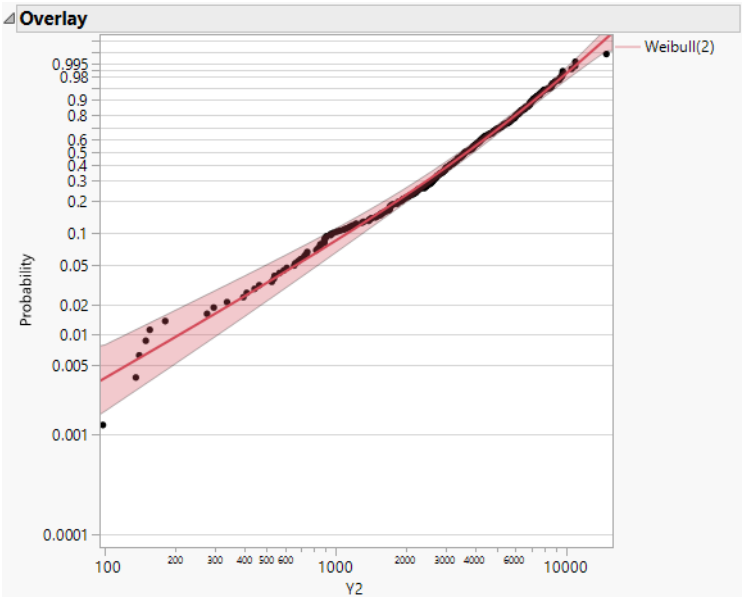
Figure 3.23 Weibull Distribution Fit



6. Click the Life Distribution red triangle and select **Fit Competing Risk Mixture**.
7. Scroll down to the Competing Risk Mixture report. Next to **Weibull**, type 2 in the **Quantity** box.
8. Click **Go**.
9. Scroll up to the Compare Distributions report. In the probability plot, right-click the vertical axis and select **Edit > Copy Axis Settings**.
10. Scroll down to the Competing Risk Mixture report. In the overlay plot, right-click the vertical axis and select **Edit > Paste Axis Settings**.
11. Do the same for the horizontal axis. In the probability plot, right-click the horizontal axis and select **Edit > Copy Axis Settings**.
12. In the overlay plot, right-click the horizontal axis and select **Edit > Paste Axis Settings**.

    The probability plot for the Weibull(2) distribution fit appears. Note that the mixture of two Weibull distributions helps better capture the distribution in the lower part of the range of Y2.

Figure 3.24 Weibull(2) Competing Risk Mixture Distribution Fit



Statistical Details for the Life Distribution Platform

This section covers the following topics:

- The distributions used in the Life Distribution platform. See "Distributions".
- Technical information about the Competing Cause report. See "Competing Cause Details".
- Technical information about median rank regression. See "Notes Regarding Median Rank Regression".

Distributions

This section provides details for the distributional fits in the Life Distribution platform. Meeker and Escobar (1998, ch. 2-5) is an excellent source of theory, application, and discussion for both the nonparametric and parametric details that follow.

Estimation and Confidence Intervals

The parameters of all distributions, unless otherwise noted, are estimated using maximum likelihood estimates (MLEs). The only exceptions are the threshold distributions. If the smallest observation is an exact failure, then this observation is treated as interval-censored with a small interval. The parameter estimates are the MLEs estimated from this slightly modified data set. Without this modification, the likelihood can be unbounded, so an MLE might not exist. This approach is similar to that described in Meeker and Escobar (1998, p. 275), except that only the smallest exact failure is censored. This is the minimal change to the data that guarantees boundedness of the likelihood function.

The Life Distribution platform offers two methods for calculating confidence intervals for the distribution parameters. These methods are labeled as Wald or Likelihood and can be selected in the launch window for the Life Distribution platform. Wald confidence intervals are used as the default setting. The computations for the confidence intervals for the cumulative distribution function (cdf) start with Wald confidence intervals on the standardized variable. Next, the intervals are transformed to the cdf scale (Nelson 1982, pp. 332–333 and pp. 346-347). The confidence intervals given in the other graphs and profilers are transformed Wald intervals (Meeker and Escobar 1998, ch. 7). Joint confidence intervals for the parameters of a two-parameter distribution are shown in the log-likelihood contour plots. They are based on approximate likelihood ratios for the parameters (Meeker and Escobar 1998, ch. 8).

Nonparametric Fit

A nonparametric fit describes the basic curve of a distribution. For data with no censoring (failures only) and for data where the observations consist of both failures and right-censoring, JMP uses Kaplan-Meier estimates. For mixed, interval, or left censoring, JMP uses Turnbull estimates. When your data set contains only right-censored data, the Nonparametric Estimate report indicates that the nonparametric estimate cannot be calculated.

The Life Distribution platform uses the midpoint estimates of the step function to construct probability plots. The midpoint estimate is halfway between (or the average of) the current and previous Kaplan-Meier estimates.

Parametric Distributions

Parametric distributions provide a more concise distribution fit than nonparametric distributions. The estimates of failure-time distributions are also smoother. Parametric models are also useful for extrapolation (in time) to the lower or upper tails of a distribution.

Note: Many distributions in the Life Distribution platform are parameterized by location and scale. For lognormal fits, the median is also provided. A threshold parameter is also included in threshold distributions. Location corresponds to μ, scale corresponds to σ, and threshold corresponds to γ.

Lognormal

Lognormal distributions are used commonly for failure times when the range of the data is several powers of 10. This distribution is often considered as the multiplicative product of many small positive identically independently distributed random variables. It is reasonable when the log of the data values appears normally distributed. Examples of data appropriately modeled by the lognormal distribution include hospital cost data, metal fatigue crack growth, and the survival time of bacteria subjected to disinfectants. The pdf curve is usually characterized by strong right-skewness. The lognormal pdf and cdf are:

$$f(x;\mu,\sigma) = \frac{1}{x\sigma}\phi_{nor}\left[\frac{\log(x)-\mu}{\sigma}\right], \quad x > 0$$

$$F(x;\mu,\sigma) = \Phi_{nor}\left[\frac{\log(x)-\mu}{\sigma}\right]$$

where

$$\phi_{nor}(z) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z^2}{2}\right)$$

and

$$\Phi_{nor}(z) = \int_{-\infty}^{z}\phi_{nor}(w)dw$$

are the pdf and cdf, respectively, for the standardized normal, or nor($\mu = 0$, $\sigma = 1$) distribution.

Weibull

The Weibull distribution can be used to model failure time data with either an increasing or a decreasing hazard rate. It is used frequently in reliability analysis because of its tremendous flexibility in modeling many different types of data, based on the values of the shape parameter, β. This distribution has been successfully used for describing the failure of electronic components, roller bearings, capacitors, and ceramics. Various shapes of the Weibull distribution can be revealed by changing the scale parameter, α, and the shape parameter, β. The Weibull pdf and cdf are commonly represented as follows:

$$f(x;\alpha,\beta) = \frac{\beta}{\alpha^{\beta}} x^{(\beta-1)} \exp\left[-\left(\frac{x}{\alpha}\right)^{\beta}\right]; \qquad x > 0, \alpha > 0, \beta > 0$$

$$F(x;\alpha,\beta) = 1 - \exp\left[-\left(\frac{x}{\alpha}\right)^{\beta}\right]$$

where α is a scale parameter, and β is a shape parameter. The Weibull distribution is particularly versatile because it reduces to an exponential distribution when β = 1. An alternative parameterization commonly used in the literature and in JMP is to use σ as the scale parameter and μ as the location parameter. These are easily converted to an α and β parameterization using the following definitions:

$$\alpha = \exp(\mu)$$

and

$$\beta = \frac{1}{\sigma}$$

The pdf and the cdf of the Weibull distribution are also expressed as a log-transformed smallest extreme value distribution (SEV). This uses a location scale parameterization, with μ = log(α) and σ = 1/β,

$$f(x;\mu,\sigma) = \frac{1}{x\sigma}\phi_{\text{sev}}\left[\frac{\log(x)-\mu}{\sigma}\right], \qquad x > 0, \sigma > 0$$

$$F(x;\mu,\sigma) = \Phi_{\text{sev}}\left[\frac{\log(x)-\mu}{\sigma}\right]$$

where

$$\phi_{\text{sev}}(z) = \exp[z - \exp(z)]$$

and

$$\Phi_{sev}(z) = 1 - \exp[-\exp(z)]$$

are the pdf and cdf, respectively, for the standardized smallest extreme value ($\mu = 0$, $\sigma = 1$) distribution.

Loglogistic

The pdf of the loglogistic distribution is similar in shape to the lognormal distribution but has heavier tails. It is often used to model data exhibiting non-monotonic hazard functions, such as cancer mortality and financial wealth. The loglogistic pdf and cdf are:

$$f(x;\mu,\sigma) = \frac{1}{x\sigma}\phi_{logis}\left[\frac{\log(x)-\mu}{\sigma}\right]$$

$$F(x;\mu,\sigma) = \Phi_{logis}\left[\frac{\log(x)-\mu}{\sigma}\right]$$

where

$$\phi_{logis}(z) = \frac{\exp(z)}{[1+\exp(z)]^2}$$

and

$$\Phi_{logis}(z) = \frac{\exp(z)}{[1+\exp(z)]} = \frac{1}{1+\exp(-z)}$$

are the pdf and cdf, respectively, for the standardized logistic or logis distribution($\mu = 0$, $\sigma = 1$).

Fréchet

The Fréchet distribution is known as a log-largest extreme value distribution or sometimes as a Fréchet distribution of maxima when it is parameterized as the reciprocal of a Weibull distribution. This distribution is commonly used for financial data. The pdf and cdf are:

$$f(x;\mu,\sigma) = \exp\left[-\exp\left(-\frac{\log(x)-\mu}{\sigma}\right)\right]\exp\left(-\frac{\log(x)-\mu}{\sigma}\right)\frac{1}{x\sigma}$$

$$F(x;\mu,\sigma) = \exp\left[-\exp\left(-\frac{\log(x)-\mu}{\sigma}\right)\right]$$

and are more generally parameterized as follows:

$$f(x;\mu,\sigma) = \frac{1}{x\sigma}\phi_{lev}\left[\frac{\log(x)-\mu}{\sigma}\right]$$

$$F(x;\mu,\sigma) = \Phi_{lev}\left[\frac{\log(x)-\mu}{\sigma}\right]$$

where

$$\phi_{lev}(z) = \exp[-z-\exp(-z)]$$

and

$$\Phi_{lev}(z) = \exp[-\exp(-z)]$$

are the pdf and cdf, respectively, for the standardized largest extreme value LEV($\mu = 0$, $\sigma = 1$) distribution.

Normal

The normal distribution is the most widely used distribution in most areas of statistics because of its relative simplicity and the ease of applying the central limit theorem. However, it is rarely used in reliability. It is most useful for data where $\mu > 0$ and the coefficient of variation (σ / μ) is small. Because the hazard function increases with no upper bound, it is particularly useful for data exhibiting wear-out failure. Examples include incandescent light bulbs, toaster heating elements, and mechanical strength of wires. The pdf and cdf are:

$$f(x;\mu,\sigma) = \frac{1}{\sigma}\phi_{nor}\left(\frac{x-\mu}{\sigma}\right), \quad -\infty < x < \infty$$

$$F(x;\mu,\sigma) = \Phi_{nor}\left(\frac{x-\mu}{\sigma}\right)$$

where

$$\phi_{nor}(z) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z^2}{2}\right)$$

and

$$\Phi_{nor}(z) = \int_{-\infty}^{z} \phi_{nor}(w)dw$$

are the pdf and cdf, respectively, for the standardized normal, or nor($\mu = 0$, $\sigma = 1$) distribution.

Smallest Extreme Value (SEV)

This non-symmetric (left-skewed) distribution is useful in two cases. The first case is when the data indicate a small number of weak units in the lower tail of the distribution (the data indicate the smallest number of many observations). The second case is when σ is small relative to μ, because probabilities of being less than zero, when using the SEV distribution, are small. The smallest extreme value distribution is useful to describe data whose hazard rate becomes larger as the unit becomes older. Examples include human mortality of the aged and rainfall amounts during a drought. This distribution is sometimes referred to as a Gumbel distribution. The pdf and cdf are:

$$f(x;\mu,\sigma) = \frac{1}{\sigma}\phi_{sev}\left(\frac{x-\mu}{\sigma}\right), \quad -\infty < \mu < \infty, \quad \sigma > 0$$

$$F(x;\mu,\sigma) = \Phi_{sev}\left(\frac{x-\mu}{\sigma}\right)$$

where

$$\phi_{sev}(z) = \exp[z - \exp(z)]$$

and

$$\Phi_{sev}(z) = 1 - \exp[-\exp(z)]$$

are the pdf and cdf, respectively, for the standardized smallest extreme value, SEV($\mu = 0$, $\sigma = 1$) distribution.

Logistic

The logistic distribution has a shape similar to the normal distribution, but with longer tails. The logistic distribution is often used to model life data when negative failure times are not an issue. Logistic regression models for a binary or ordinal response assume the logistic distribution as the latent distribution. The pdf and cdf are:

$$f(x;\mu,\sigma) = \frac{1}{\sigma}\phi_{logis}\left(\frac{x-\mu}{\sigma}\right), \quad -\infty < \mu < \infty \text{ and } \sigma > 0$$

$$F(x;\mu,\sigma) = \Phi_{logis}\left(\frac{x-\mu}{\sigma}\right)$$

where

$$\phi_{logis}(z) = \frac{\exp(z)}{[1 + \exp(z)]^2}$$

and

$$\Phi_{\text{logis}}(z) = \frac{\exp(z)}{[1+\exp(z)]} = \frac{1}{1+\exp(-z)}$$

are the pdf and cdf, respectively, for the standardized logistic or logis distribution ($\mu = 0$, $\sigma = 1$).

Largest Extreme Value (LEV)

This right-skewed distribution can be used to model failure times if σ is small relative to $\mu > 0$. This distribution is not commonly used in reliability but is useful for estimating natural extreme phenomena, such as a catastrophic flood heights or extreme wind velocities. The pdf and cdf are:

$$f(x;\mu,\sigma) = \frac{1}{\sigma}\phi_{\text{lev}}\left(\frac{x-\mu}{\sigma}\right), \quad -\infty < \mu < \infty \text{ and } \sigma > 0$$

$$F(x;\mu,\sigma) = \Phi_{\text{lev}}\left(\frac{x-\mu}{\sigma}\right)$$

where

$$\phi_{\text{lev}}(z) = \exp[-z-\exp(-z)]$$

and

$$\Phi_{\text{lev}}(z) = \exp[-\exp(-z)]$$

are the pdf and cdf, respectively, for the standardized largest extreme value LEV($\mu = 0$, $\sigma = 1$) distribution.

Exponential

Both one- and two-parameter exponential distributions are used in reliability. The pdf and cdf for the two-parameter exponential distribution are:

$$f(x;\theta,\gamma) = \frac{1}{\theta}\exp\left(-\frac{x-\gamma}{\theta}\right), \quad \theta > 0$$

$$F(x;\theta,\gamma) = 1-\exp\left(-\frac{x-\gamma}{\theta}\right)$$

where θ is a scale parameter and γ is both the threshold and the location parameter. Reliability analysis frequently uses the one-parameter exponential distribution, with $\gamma = 0$. The exponential distribution is useful for describing failure times of components exhibiting wear-out far beyond their expected lifetimes. This distribution has a constant failure rate, which means that for small time increments, failure of a unit is independent of the unit's age. The exponential distribution should not be used for describing the life of mechanical components that can be exposed to fatigue, corrosion, or short-term wear. This distribution is, however, appropriate for modeling certain types of robust electronic components. It has been used successfully to describe the life of insulating oils and dielectric fluids (Nelson 1990, p. 53).

Log Generalized Gamma (LogGenGamma)

The log generalized gamma distribution contains the SEV, LEV, and Normal. The pdf and cdf are:

$$f(x;\mu,\sigma,\lambda) = \begin{cases} \frac{|\lambda|}{\sigma}\phi_{lg}[\lambda\omega + \log(\lambda^{-2});\lambda^{-2}] & \text{if } \lambda \neq 0 \\ \frac{1}{\sigma}\phi_{nor}(\omega) & \text{if } \lambda = 0 \end{cases}$$

$$F(x;\mu,\sigma,\lambda) = \begin{cases} \Phi_{lg}[\lambda\omega + \log(\lambda^{-2});\lambda^{-2}] & \text{if } \lambda > 0 \\ \Phi_{nor}(\omega) & \text{if } \lambda = 0 \\ 1 - \Phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda < 0 \end{cases}$$

where $-\infty < x < \infty$, $\omega = [x - \mu]/\sigma$, and

$$-\infty < \mu < \infty, \quad -12 < \lambda < 12, \text{ and } \sigma > 0.$$

Note that

$$\phi_{lg}(z;\kappa) = \frac{1}{\Gamma(\kappa)}\exp[\kappa z - \exp(z)]$$

$$\Phi_{lg}(z;\kappa) = \Gamma_I[\exp(z);\kappa]$$

are the pdf and cdf, respectively, for the log-gamma variable and $\kappa > 0$ is a shape parameter. The standardized distributions above are dependent upon the shape parameter κ.

Note: In JMP, the shape parameter, λ, for the generalized gamma distribution is bounded between [-12,12] to provide numerical stability.

Extended Generalized Gamma (GenGamma)

The extended generalized gamma distribution can include many other distributions as special cases, such as the generalized gamma, Weibull, lognormal, Fréchet, gamma, and exponential. It is particularly useful for cases with little or no censoring. This distribution has been successfully modeled for human cancer prognosis. The pdf and cdf are:

$$f(x;\mu,\sigma,\lambda) = \begin{cases} \frac{|\lambda|}{x\sigma}\phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda \neq 0 \\ \frac{1}{x\sigma}\phi_{nor}(\omega) & \text{if } \lambda = 0 \end{cases}$$

$$F(x;\mu,\sigma,\lambda) = \begin{cases} \Phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda > 0 \\ \Phi_{nor}(\omega) & \text{if } \lambda = 0 \\ 1 - \Phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda < 0 \end{cases}$$

where $x > 0$, $\omega = [\log(x) - \mu]/\sigma$, and

$$-\infty < \mu < \infty, \quad -12 < \lambda < 12, \text{ and } \sigma > 0.$$

Note that

$$\phi_{lg}(z;\kappa) = \frac{1}{\Gamma(\kappa)}\exp[\kappa z - \exp(z)]$$

$$\Phi_{lg}(z;\kappa) = \Gamma_I[\exp(z);\kappa]$$

are the pdf and cdf, respectively, for the standardized log-gamma variable and $\kappa > 0$ is a shape parameter.

The standardized distributions above are dependent upon the shape parameter κ. Meeker and Escobar (1998, ch. 5) give a detailed explanation of the extended generalized gamma distribution.

Note: In JMP, the shape parameter, λ, for the generalized gamma distribution is bounded between [-12,12] to provide numerical stability.

Distributions with Threshold Parameters

Threshold Distributions are log-location-scale distributions with threshold parameters. Some of the distributions above are generalized by adding a threshold parameter, denoted by γ. The addition of this threshold parameter shifts the left endpoint of the distribution away from 0. Threshold parameters are sometimes called shift, minimum, or guarantee parameters because all units survive at least until threshold time. Note that while adding a threshold parameter shifts the distribution on the time axis, the shape, and spread of the distribution are not affected. Threshold distributions are useful for fitting moderately to heavily shifted distributions. The general forms for the pdf and cdf of a log-location-scale threshold distribution are:

$$f(x;\mu,\sigma,\gamma) = \frac{1}{\sigma(x-\gamma)}\phi\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right], \quad x > \gamma$$

$$F(x;\mu,\sigma,\gamma) = \Phi\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right]$$

where ϕ and Φ are the pdf and cdf, respectively, for the specific distribution. Examples of specific threshold distributions are shown below for the Weibull, lognormal, Fréchet, and loglogistic distributions, where, respectively, the SEV, Normal, LEV, and logis pdfs and cdfs are appropriately substituted.

Note: If the smallest observation is a failure (not censored), JMP creates a small interval around the point and treats the observation as interval censored. This padding around the failure bounds the log-likelihood function and improves estimation. If the smallest observation is censored, then no extra padding is added to the observation.

TH Weibull

The pdf and cdf of the three-parameter Weibull distribution are:

$$f(x;\mu,\sigma,\gamma) = \frac{1}{(x-\gamma)\sigma}\phi_{sev}\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right], \quad x > \gamma,\ \sigma > 0$$

$$F(x;\mu,\sigma,\gamma) = \Phi_{sev}\left(\frac{\log(x-\gamma)-\mu}{\sigma}\right) = 1 - \exp\left[-\left(\frac{x-\gamma}{\alpha}\right)^{\beta}\right], \quad x > \gamma$$

where $\mu = \log(\alpha)$, and $\sigma = 1/\beta$ and where

$$\phi_{sev}(z) = \exp[z - \exp(z)]$$

and

$$\Phi_{sev}(z) = 1 - \exp[-\exp(z)]$$

are the pdf and cdf, respectively, for the standardized smallest extreme value, SEV($\mu = 0$, $\sigma = 1$) distribution.

TH Lognormal

The pdf and cdf of the three-parameter lognormal distribution are:

$$f(x;\mu, \sigma, \gamma) = \frac{1}{\sigma(x-\gamma)}\phi_{nor}\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right], \quad x > \gamma$$

$$F(x;\mu,\sigma,\gamma) = \Phi_{nor}\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right]$$

where

$$\phi_{nor}(z) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z^2}{2}\right)$$

and

$$\Phi_{nor}(z) = \int_{-\infty}^{z}\phi_{nor}(w)dw$$

are the pdf and cdf, respectively, for the standardized normal, or N($\mu = 0$, $\sigma = 1$) distribution.

TH Fréchet

The pdf and cdf of the three-parameter Fréchet distribution are:

$$f(x;\mu, \sigma, \gamma) = \frac{1}{\sigma(x-\gamma)}\phi_{lev}\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right], \quad x > \gamma$$

$$F(x;\mu,\sigma,\gamma) = \Phi_{lev}\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right]$$

where

$$\phi_{lev}(z) = \exp[-z - \exp(-z)]$$

and

$$\Phi_{lev}(z) = \exp[-\exp(-z)]$$

are the pdf and cdf, respectively, for the standardized largest extreme value LEV($\mu = 0$, $\sigma = 1$) distribution.

TH Loglogistic

The pdf and cdf of the three-parameter loglogistic distribution are:

$$f(x;\mu, \sigma, \gamma) = \frac{1}{\sigma(x-\gamma)}\phi_{\text{logis}}\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right], \quad x > \gamma$$

$$F(x;\mu,\sigma,\gamma) = \Phi_{\text{logis}}\left[\frac{\log(x-\gamma)-\mu}{\sigma}\right]$$

where

$$\phi_{\text{logis}}(z) = \frac{\exp(z)}{[1+\exp(z)]^2}$$

and

$$\Phi_{\text{logis}}(z) = \frac{\exp(z)}{[1+\exp(z)]} = \frac{1}{1+\exp(-z)}$$

are the pdf and cdf, respectively, for the standardized logistic or logis distribution ($\mu = 0$, $\sigma = 1$).

Distributions for Defective Subpopulations

In reliability experiments, there are times when only a fraction of the population has a particular defect leading to failure. Because all units are not susceptible to failure, using the regular failure distributions is inappropriate and might produce misleading results. Use the DS distribution options to model failures that occur on only a subpopulation. The following DS distributions are available:

- DS Lognormal
- DS Weibull
- DS Loglogistic
- DS Fréchet

The pdf and cdf for defective subpopulation distributions are defined as follows:

$$f(t) = \left[p\frac{1}{t\sigma}\right]\phi\left[\frac{(\log(t)-\mu)}{\sigma}\right]$$

$$F(t) = p\Phi\left[\left(\frac{(\log(t)-\mu)}{\sigma}\right)\right]$$

where:

p is the defective subpopulation fraction

t is the time of measurement for the lifetime event

μ and σ are estimated by calculating the usual maximum likelihood estimations using the pdf and cdf of the corresponding defective subpopulation

$\phi(z)$ and $\Phi(z)$ are the density and cumulative distribution function, respectively, for a standard distribution. For example, for a Weibull distribution,

$\phi(z) = \exp(z\text{-}\exp(z))$ and $\Phi(z) = 1 - \exp(\text{-}\exp(z))$.

See Tobias and Trindad (2012, p. 321) for more information about defective subpopulation models.

The defective subpopulation model is also known as a *limited failure population* model in Meeker and Escobar (1998, ch. 11).

Zero-Inflated Distributions

Zero-inflated distributions are used when some proportion (p) of the data fail at $t = 0$. When the data contain more zeros than expected by a standard model, the number of zeros is inflated. When the time-to-event data contain zero as the minimum value in the Life Distribution platform, four zero-inflated distributions are available. These distributions include:

- Zero-Inflated Lognormal (ZI Lognormal)
- Zero-Inflated Weibull (ZI Weibull)
- Zero-Inflated Loglogistic (ZI Loglogistic)
- Zero-Inflated Fréchet (ZI Fréchet)

The pdf and cdf for zero-inflated distributions are defined as follows:

$$f(t) = \left[(1-p)\frac{1}{t\sigma}\right]\phi\left[\frac{(\log(t)-\mu)}{\sigma}\right]$$

$$F(t) = p + (1-p)\Phi\left[\left(\frac{(\log(t)-\mu)}{\sigma}\right)\right]$$

where:

p is the proportion of zero data values

t is the time of measurement for the lifetime event

μ and σ are estimated by calculating the usual maximum likelihood estimations after removing zero values from the original data

$\phi(z)$ and $\Phi(z)$ are the density and cumulative distribution function, respectively, for a standard distribution. For example, for a Weibull distribution,

$\phi(z) = \exp(z\text{-}\exp(z))$ and $\Phi(z) = 1 - \exp(\text{-}\exp(z))$.

See Lawless (2003, p. 34) for more information about zero-inflated distributions. Substitute $p = 1 - p$ and $S_1(t) = 1 - \Phi(t)$ to obtain the form shown above.

See Tobias and Trindade (1995, p. 232) for more information about reliability distributions. This reference gives the general form for mixture distributions. Using the parameterization in Tobias and Trindade, the form above can be found by substituting $\alpha = p$, $F_d(t) = 1$, and $F_N(t) = \Phi(t)$.

Prior Distributions for Bayesian Estimation

The following distributions are available for Location Scale Priors:

- Normal/Lognormal, with hyperparameters Location (mu) and Scale (sigma). For a definition, see "Lognormal" and "Normal".
- Uniform, with hyperparameters Low and End, which define the support of a Uniform distribution.
- Gamma, with hyperparameters Shape and Scale. The k/theta parameterization and the probability density function is used.
- Point Mass, with hyperparameter Location. This is a degenerated prior; there is only one possible value for the parameter that we are assigning a prior distribution to. The only possible value equals the value that is entered to this Location hyperparameter.

The following distributions are available for Quantile Parameter Priors:

- Normal/Lognormal, with a 99% probability range, specifies the prior distribution using the 0.005 and 0.995 percentiles of the distribution. JMP backs out the mu and sigma.
- Uniform, with hyperparameters Lower and Upper Limits, which define the support of a Uniform distribution.
- Log-Uniform, with Lower (a) and Upper (b) Limits. This distribution is uniform on the log scale between Log(a) and Log(b).
- Point Mass, with hyperparameter Location. This is a degenerated prior; there is only one possible value for the parameter that we are assigning a prior distribution to. The only possible value equals the value that is entered to this Location hyperparameter.

The following distributions are available for Failure Probability Priors:

- Beta, characterized by the probability density function.

- Specify the Beta prior using estimates and error percentages (mean and variance). The mean equals the number entered in to the Estimate, and the variance equals (Error Percentage / 100 * Estimate)^2.
- Specify the Beta prior using 0.005 and 0.995 percentiles of the distribution. JMP backs out the hyperparameters.

Competing Cause Details

For a competing cause model, a closed form for the aggregated distribution is defined as follows:

$$F(x) = 1 - \prod_{i=1}^{k} [1 - F_i(x)]$$

where the $F_i(x)$, $i = 1, \ldots, k$, are individual failure distributions corresponding to causes. Confidence limits are readily available, because all involved estimates are MLEs.

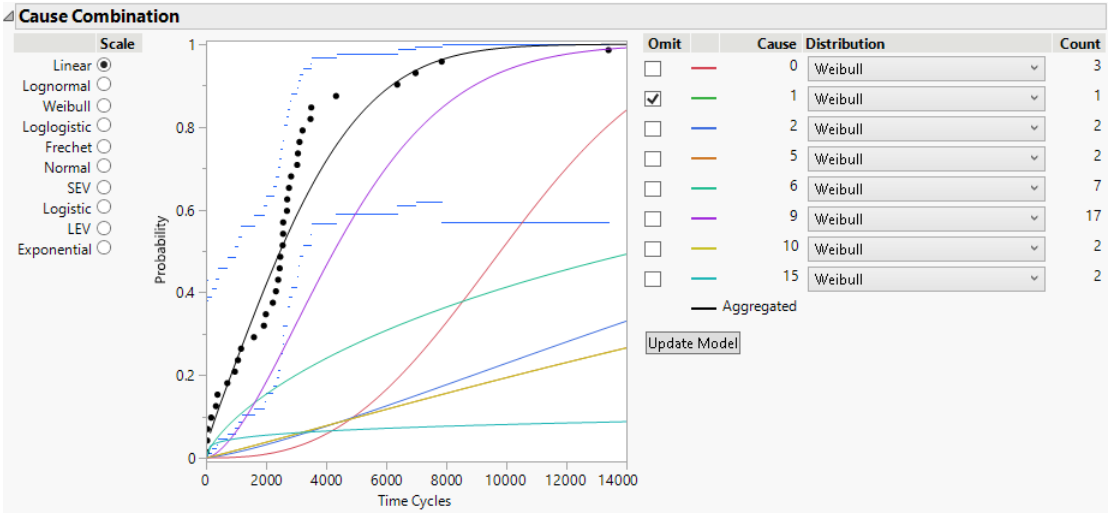
Specify a Fixed Parameter Model as a Distribution for a Cause

If a Fixed Parameter model is specified for a cause, you must fix the parameter in the Individual Causes report for that cause. Fix the parameter in the desired Parametric Estimate report in the Life Distribution - Failure Cause: <Name> report. The fixed parameter becomes part of the aggregated distribution when you click Update Model.

This example illustrates how to include the Fixed Parameter model into the aggregated distribution:

1. Select **Help > Sample Data Library** and open Reliability/Appliance.jmp.
2. Select **Analyze** > **Reliability and Survival** > **Life Distribution**.
3. Select Time Cycles and click **Y, Time to Event.**
4. Select Cause Code and click **Failure Cause**.
5. Select **Likelihood** as the Confidence Interval Method.
6. Select **Allow failure mode to use fixed parameter models**.
7. Click **OK.**

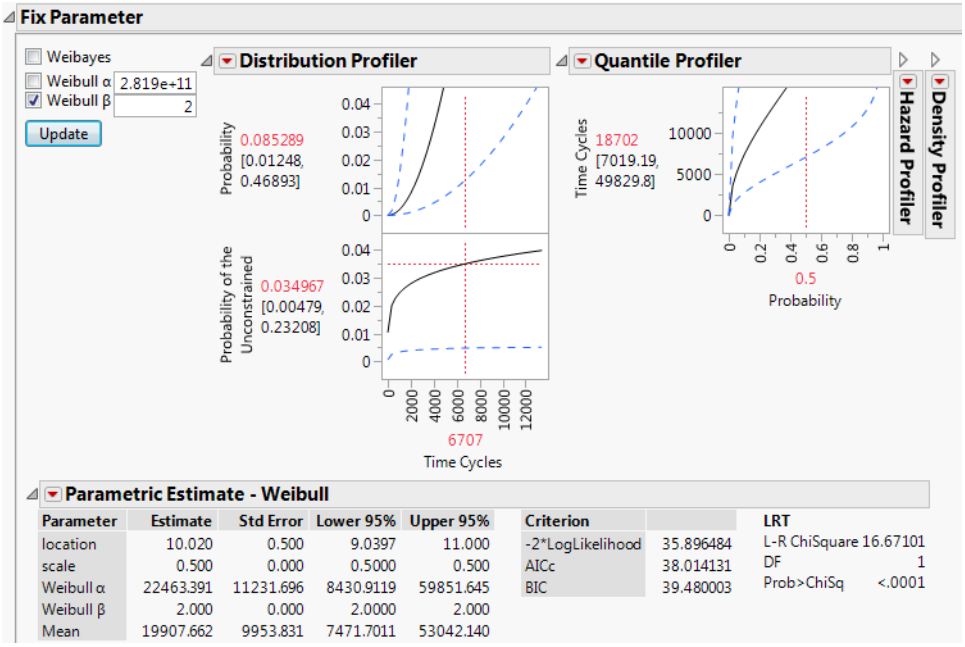
Figure 3.25 Fixed Parameter Model with Cause 1 Omitted



By default, Cause = 1 is omitted, because there are not enough data. However, you do not want this cause to be omitted.

8. Open the Individual Causes report for Cause 1. The report is called Life Distribution - Failure Cause: 1 Failure Counts: 1.
9. Click the red triangle next to Parametric Estimate - Weibull and select **Fix Parameter**.
10. Select **Weibull beta** and type 2.
11. Click **Update**.

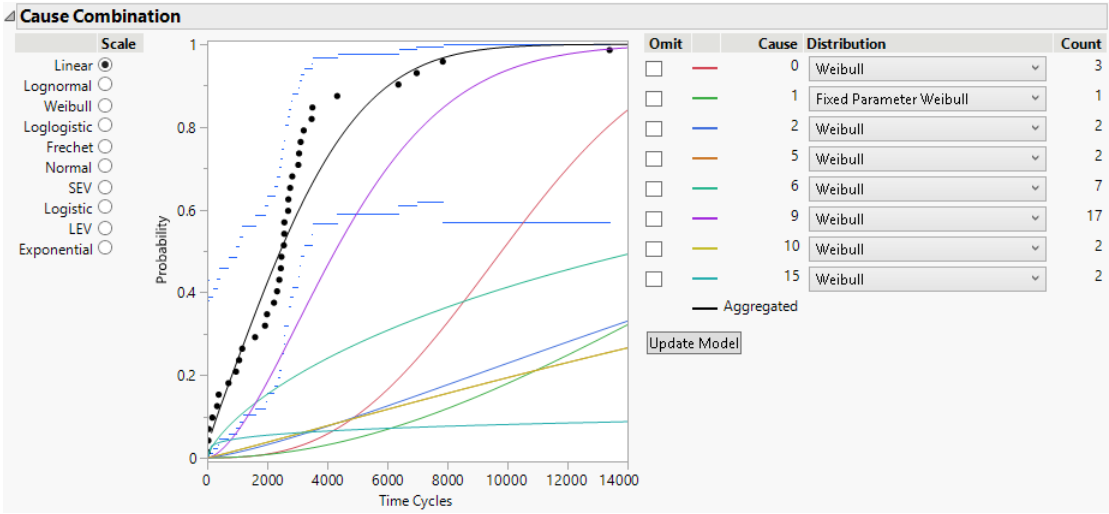
Figure 3.26 Fixed Parameter Model with Weibull Beta Specified



In the Parametric Estimate - Weibull report, assuming β equals 2, the alpha parameter is estimated to be 22463.391. Now you can use this for the failure distribution for Cause=1.

12. Scroll up to Cause Combination at the top of the report window.
13. Deselect **Omit** for Cause 1.
14. For the distribution for Cause 1, select **Fixed Parameter Weibull**.
15. Click **Update Model**.

Figure 3.27 Updated Model Showing Cause 1



Now the aggregated model uses the Fixed Parameter Weibull results for Cause 1 in the overall competing cause model.

Specify a Bayesian Model for a Cause

The steps for specifying a Bayesian model for a cause are similar to those described in "Specify a Fixed Parameter Model as a Distribution for a Cause". Define the model in the desired Bayesian Estimation report found in the corresponding Parametric Estimate outline under Statistics in the Life Distribution report for the individual cause. See "Bayesian Estimation - <Distribution Name>".

To incorporate a Bayesian model into the aggregated model, non-Bayesian distributions for other causes must be amenable to a simulation-based framework. For example, suppose that a model has two failure causes. One is modeled using a Weibull distribution and the other using a Bayesian approach for estimating the parameters of a second Weibull. The parameters for the first Weibull distribution, denoted by the vector θ_1, are estimated using maximum likelihood. The parameters for the second Weibull, θ_2, are estimated using the Bayesian approach.

The quantiles and median of the aggregated mixture distribution, denoted $F(x, \theta_1, \theta_2)$, are obtained as follows:

- A parametric bootstrap is performed for the first Weibull, yielding random samples from the asymptotic distribution of the maximum likelihood estimate $\hat{\theta}_1$. Denote a sampled value from the asymptotic distribution of $\hat{\theta}_1$ by θ_1^*.
- A sample is drawn from the posterior distribution of θ_2, denoted by θ_2^*.

- For each set of values θ_1^* and θ_2^*, an estimate of $F(x, \theta_1, \theta_2)$, denoted by $F^*(x, \theta_1, \theta_2)$, is obtained.
- The values $F^*(x, \theta_1, \theta_2)$ are used to obtain estimates of the quantiles and median of the aggregated distribution. These are the values displayed in the Distribution profiler at a given value of x.

Specify a Weibayes Model for a Cause

The steps for specifying a Weibayes model for a cause are similar to those described in "Specify a Fixed Parameter Model as a Distribution for a Cause". Select the Fix Parameter option in the Parametric Estimate - Weibull outline under Statistics in the Life Distribution report for the cause. In the Fix Parameter report, check the Weibayes option. The Weibayes model is treated as a Bayesian model and a bootstrap sample is drawn from the posterior distribution of the parameter alpha. See Liu and Wang (2013).

Mean Remaining Life Calculator

Use the Configuration option in the red triangle menu to set a value for the number of simulated failure times used in computing the mean remaining life. Denote this value by m.

To obtain an estimate of the mean remaining life at time t, m samples are drawn from the aggregated distribution conditioned on survival to time t. Their average is computed.

To compute the confidence limits for the mean remaining life, you must select the box in the Configuration window. You then have the option to set the number of bootstrap samples. Denote this value by n.

To compute the confidence interval, n samples of parameter estimates are drawn from either the asymptotic distributions of the MLEs, or the posterior distributions derived using Bayesian inference. For each sample of parameter values, an aggregated distribution is formed, from which m samples are drawn to compute a mean remaining life. The samples of n mean remaining life values are used to construct the confidence interval.

Fit Mixture Save Predictions Formulas

This section gives the formulas used in calculating values in the columns saved by the Fit Mixture report option Save Predictions.

Consider the following notation:

$\hat{p}_i$ is an estimate of the mixture proportion, w_i

$\hat{F}_i$ is the estimated probability distribution function F_i

$\hat{f}_i$ is the estimated probability density function for F_i

- If the observation y is not censored, the saved value is given by the following:

$$\frac{\hat{p}_i \hat{f}_i(y)}{\sum_{i=1}^{k} \hat{p}_i \hat{f}_i(y)}$$

- If the observation is censored, the saved value is obtained by replacing the estimated density values in the formula for an uncensored observation by the following:

$\hat{F}_i(y)$ for right censoring

$1 - \hat{F}_i(y)$ for left censoring

$\hat{F}_i(y_{high}) - \hat{F}_i(y_{low})$ for interval censoring

Notes Regarding Median Rank Regression

When there are no censored rows and no Weight column, median rank regression (MRR) for Weibull parameter estimates is available in the Life Distribution platform. The following conditions must be met before MRR estimates appear in the Life Distribution report:

- This Life Distribution platform preference is selected: Report Median Rank Regression Based Weibull Parameter Estimates When There Are No Censored Observations.
- There are no censored observations in the analysis. In other words, all observations represent failure times.
- There is no Weight column specified.

If all of the above conditions are met, the Parametric Estimate - Weibull report is modified to show MRR estimates as well as the usual maximum likelihood estimates (MLE). The column headings in the parameter estimates table are prefixed with MLE or MRR to denote which estimation method was used to produce each column of estimates. There are no confidence intervals available for the MRR estimates.

Caution: The use of MRR estimates is not recommended. Median rank regression uses least squares estimation to produce estimates, instead of maximum likelihood. There are no commonly accepted methods to apply least squares estimation to censored data. Further, Genschel and Meeker (2010) show that the MLE method is more precise. They provide simulation results that are based on the fact that the true estimates are known, but in reality, the accuracy of MRR estimates is unknown. Because MRR is a loosely defined and ad hoc estimation procedure, different software packages produce different MRR estimates for the same data.

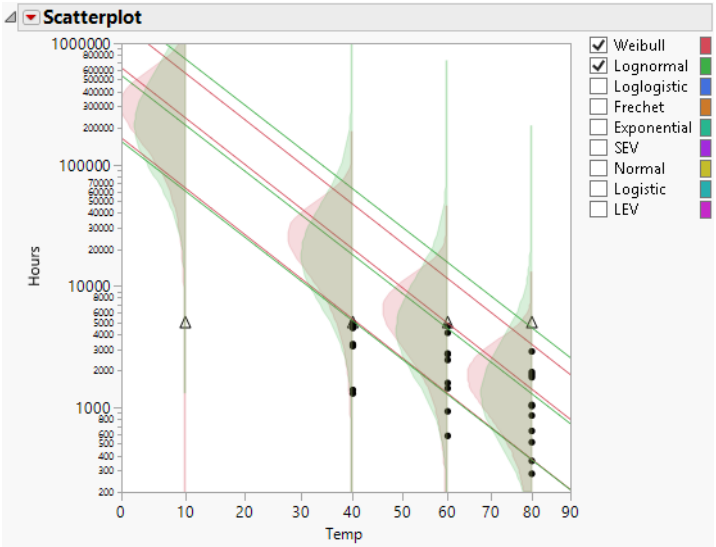
Chapter 4

Fit Life by X

Fit Single-Factor Models to Time-to Event Data

The Fit Life by X platform helps you analyze lifetime events when only one factor is present. You can choose to model the relationship between the event and the factor using various transformations, or create a custom transformation of your data. You also have the flexibility of comparing different distributions at the same factor level and comparing the same distribution across different factor levels.

Figure 4.1 Scatterplot Showing Varying Distributions and Factor Levels



Contents

Overview of the Fit Life by X Platform 105
Example of the Fit Life by X Platform 106
Launch the Fit Life by X Platform 108
The Fit Life by X Report 110
Summary of Data 111
Scatterplot 111
Nonparametric Overlay 113
Comparisons 114
Results 118
Custom Relationship 129
Fit Life by X Platform Options 130
Additional Examples of the Fit Life by X Platform 131
Capacitor Example 131
Custom Relationship Example 133

Overview of the Fit Life by X Platform

The Fit Life by X platform provides the tools needed for accelerated life-testing analysis. Accelerated tests are routinely used in industry to provide failure-time information about products or components in a relatively short time frame. Common accelerating factors include temperature, voltage, pressure, and usage rate. Results are extrapolated to obtain time-to-failure estimates at lower, normal operating levels of the accelerating factors. These results are used to assess reliability, detect and correct failure modes, compare manufacturers, and certify components.

The Fit Life by X platform includes many commonly used transformations to model physical and chemical relationships between the event and the factor of interest. Examples include transformation using Arrhenius (Celsius, Fahrenheit, and Kelvin) relationship time-acceleration factors and Voltage-acceleration mechanisms. Linear, Log, Logit, Reciprocal, Square Root, Box-Cox, Location, Location and Scale, and Custom acceleration models are also included in this platform.

You can use the DOE > Accelerated Life Test Design platform to design accelerated life test experiments. See the *Design of Experiments Guide*.

Meeker and Escobar (1998, p. 495) offer the following strategy for analyzing accelerated lifetime data:

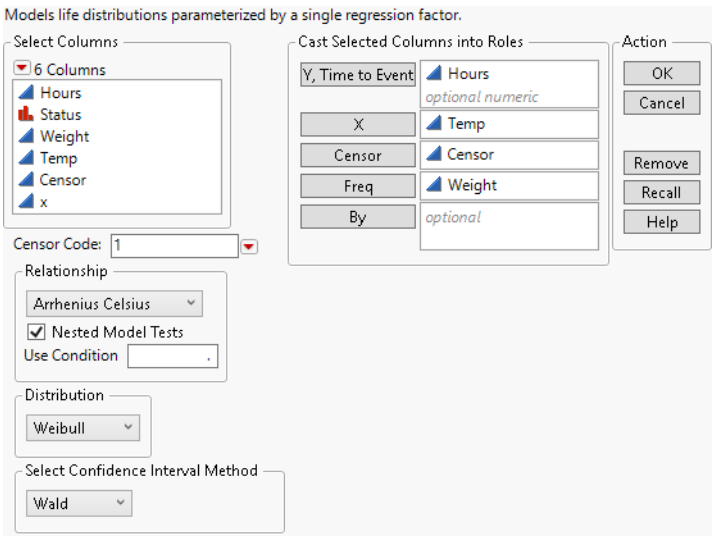
1. Examine the data graphically. One useful way to visualize the data is by examining a scatterplot of the time-to-failure variable versus the accelerating factor.
2. Fit distributions individually to the data at different levels of the accelerating factor. Repeat for different assumed distributions.
3. Fit an overall model with a plausible relationship between the time-to-failure variable and the accelerating factor.
4. Compare the model in Step 3 with the individual analyses in Step 2, assessing the lack of fit for the overall model.
5. Perform residual and various diagnostic analyses to verify model assumptions.
6. Assess the plausibility of the data to make inferences.

Example of the Fit Life by X Platform

This example uses the Devalt.jmp sample data table, from Meeker and Escobar (1998), and can be found in the Reliability folder of the sample data. It contains time-to-failure data for a device at accelerated operating temperatures. No time-to-failure observation is recorded for the normal operating temperature of 10 degrees Celsius; all other observations are shown as time-to-failure or censored values at accelerated temperature levels of 40, 60, and 80 degrees Celsius.

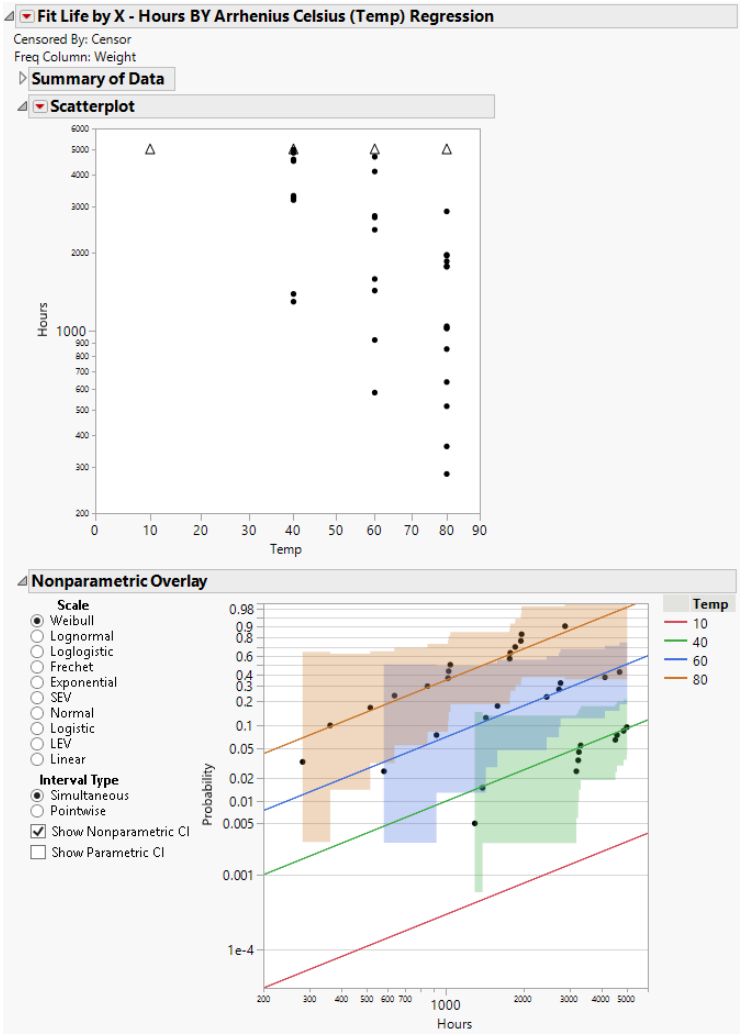
1. Select **Help > Sample Data Library** and open Reliability/Devalt.jmp.
2. Select **Analyze > Reliability and Survival > Fit Life by X**.
3. Select Hours and click **Y, Time to Event**.
4. Select Temp and click **X**.
5. Select Censor and click Censor.
6. Leave the **Censor Code** as 1.
7. Select Weight and click **Freq**.
8. Keep **Arrhenius Celsius** as the relationship, and keep the **Nested Model Tests** option selected.
9. Select **Weibull** as the distribution.
10. Keep **Wald** as the confidence interval method.

Figure 4.2 Fit Life by X Launch Window



11. Click **OK**.

Figure 4.3 Fit Life by X Report Window for Devalt.jmp Data

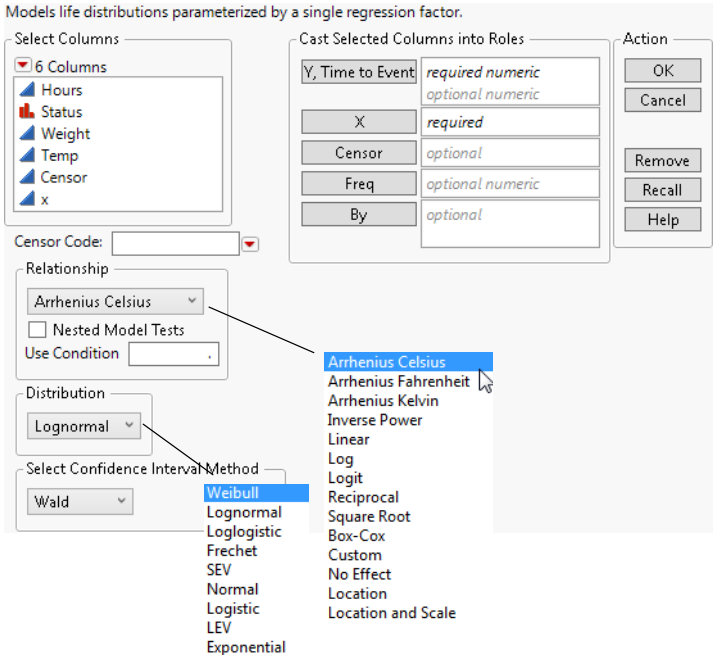


The report window shows summary data, diagnostic plots, comparison data and results, including detailed statistics and prediction profilers. Separate result sections are shown for each selected distribution. Distribution, Quantile, Hazard, Density, and Acceleration Factor Profilers are included for each of the specified distributions.

Launch the Fit Life by X Platform

Launch the Fit Life by X platform by selecting **Analyze > Reliability and Survival > Fit Life by X**. For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Figure 4.4 The Fit Life by X Launch Window



The Fit Life by X launch window contains the following options:

Y, Time to Event Identifies the time to event (such as the time to failure) or time to censoring. With interval censoring, specify two Y variables, where one Y variable gives the lower limit and the other Y variable gives the upper limit for each unit. For more information about censoring, see "Event Plot".

X Identifies the accelerating factor.

Censor Identifies censored observations. Select the value that identifies right-censored observations from the Censor Code menu beneath the Select Columns list. The Censor column is used only when one Y is entered.

Freq Identifies frequencies or observation counts when there are multiple units. If the value is 0 or a positive integer, then the value represents the frequencies or counts of observations for each row when there are multiple units recorded.

By Identifies a column that creates a report consisting of separate analyses for each level of the variable.

Censor Code Identifies the value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

Relationship Identifies the relationship between the event and the accelerating factor. Table 4.1 defines the model for each relationship.

Table 4.1 Models for Relationship Options

Relationship	Model
Arrhenius Celsius	μ = b0 + b1 * 11604.5181215503 / (X + 273.15)
Arrhenius Fahrenheit	μ = b0 + b1 * 11604.5181215503 / ((X + 459.67) / 1.8)
Arrhenius Kelvin	μ = b0 + b1 * 11604.5181215503 / X
Voltage	μ = b0 + b1 * log(X)
Linear	μ = b0 + b1 * X
Log	μ = b0 + b1 * log(X)
Logit	μ = b0 + b1 * log(X / (1 - X))
Reciprocal	μ = b0 + b1 / X
Square Root	μ = b0 + b1 * sqrt(X)
Box-Cox	μ = b0 + b1 * BoxCox(X)
Location	means that μ is different for every level of X
Location and Scale	means that μ and σ are both different for every level of X (equivalent to a Life Distribution fit with X as a By variable)
Custom	user-defined μ and σ

If you select Box-Cox, a text edit box appears below the Use Condition option. Use this box to specify a lambda value. The BoxCox(X) transformation for a specified λ is defined as follows:

$$x_i^{(\lambda)} = \begin{cases} \dfrac{x_i^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln(x_i) & \text{if } \lambda = 0 \end{cases}$$

If you want to use a Custom relationship for your model, see "Custom Relationship".

Nested Model Tests Appends a nonparametric overlay plot, nested model tests, and a multiple probability plot to the report window.

Use Condition Enables you to enter a value for the explanatory variable, X, of the acceleration factor. You can also set the use condition value after launching the platform using the Set Time Acceleration Use Condition option from the Fit Life by X red triangle menu.

Distribution Specifies one distribution (Weibull, Lognormal, Loglogistic, Fréchet, SEV, Normal, Logistic, LEV, or Exponential distributions) at a time. Lognormal is the default setting.

Select Confidence Interval Method Displays the method for computing confidence intervals for the parameters. The default is Wald, but you can select Likelihood instead. The Wald method is an approximation and runs faster. The Likelihood method provides more precise parameters but takes longer to compute.

Note: The Confidence Interval Method preference enables you to select the Likelihood method as the default confidence interval method. You can change this preference in **Preferences > Platforms > Fit Life by X**.

The Fit Life by X Report

The initial report window includes the following sections:

- "Summary of Data"
- "Scatterplot"
- "Nonparametric Overlay"
- "Comparisons"

  Distribution, Quantile, Hazard, Density, and Acceleration Factor profilers, along with criteria values under Comparison Criterion can be viewed and compared.
- "Results"

Parametric estimates, covariance matrices, nested model tests, and diagnostics can be examined and compared for each of the selected distributions. You can also perform custom estimation and obtain Bayesian estimates for the distribution parameters.

Summary of Data

The Summary of Data section gives the total number of observations, the number of uncensored values, and the number of censored values (right, left, and interval). Figure 4.5 shows the summary data for the Devalt.jmp sample data table.

Figure 4.5 Summary of Data Example

Summary of Data

Observation Used	165
Uncensored Values	33
Right Censored Values	132

Scatterplot

The Scatterplot of the lifetime event versus the explanatory variable is shown at the top of the report window. For the Devalt.jmp sample data, the Scatterplot shows Hours versus Temp. Table 4.2 indicates how each type of failure is represented on the Scatterplot in the report window. To increase the size of the markers on the graph, right-click the graph, select **Marker Size**, and then select one of the marker sizes listed.

Figure 4.6 Scatterplot of Hours versus Temp

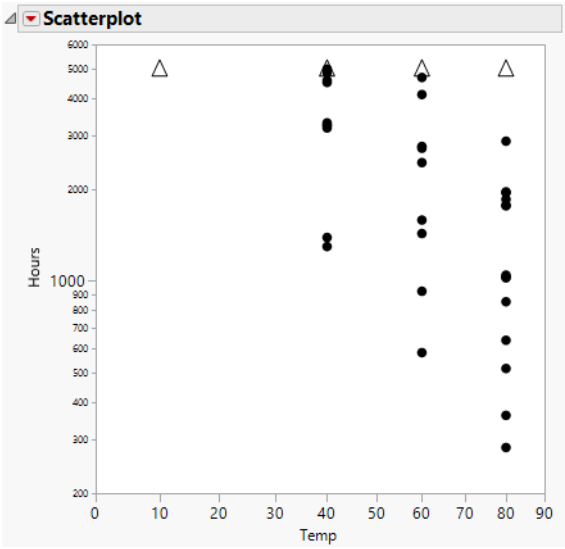


Table 4.2 Scatterplot Representation for Failure and Censored Observations

Event	Scatterplot Representation
failure	dots
right-censoring	upward triangles
left-censoring	downward triangles
interval-censoring	downward triangle on top of an upward triangle, connected by a solid line

Scatterplot Options

The Scatterplot red triangle menu contains the following options:

Add Density Curve Specify the density curve that you want, one at a time, by entering any value within the range of the accelerating factor. You can then select different distributions by selecting the appropriate check box(es) that appear after you add a curve.

Remove Density Curves Displays previously entered density curve values. Remove curves by selecting the appropriate check box.

Show Density Curves Displays the density curves. If the **Location** or the **Location and Scale** model is fit, or if **Nested Model Tests** is selected in the launch window, then the density curves for all of the given explanatory variable levels are shown. After the curves have been created, the **Show Density Curves** option toggles the curves on and off the plot.

Add Quantile Lines Specify the quantile lines that you want, three at a time. You can add more quantiles by continually selecting **Add Quantile Lines**. Default quantile values are 0.1, 0.5, and 0.9. Invalid quantile values, such as missing values, are ignored. If desired, you can enter just one quantile value, leaving the other entries blank.

Show Quantile Line CI Bands Shows or hides confidence intervals around the quantile lines.

Set Level of Quantile Line CI Bands Specifies the confidence level for the confidence intervals around the quantile lines.

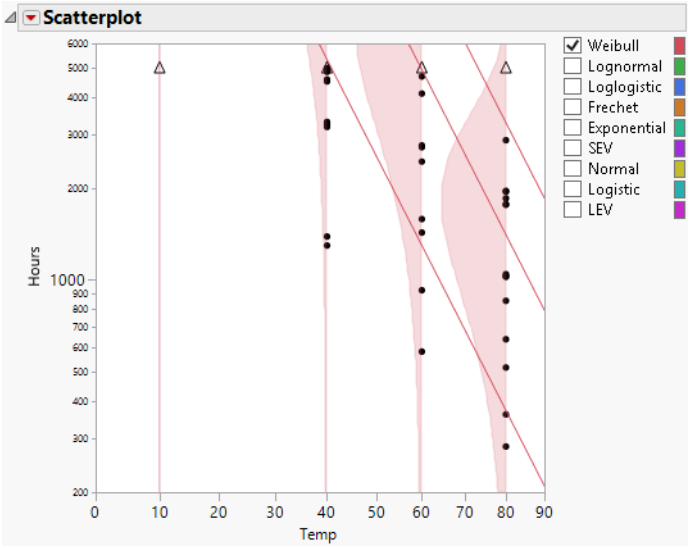
Remove Quantile Lines Displays previously entered quantile values. Remove lines by selecting the appropriate check box.

Transposed Axes Swaps the horizontal and vertical axes.

Use Transformation Scale The default view of the scatterplot incorporates the transformation scale. Select this option to switch between the linear and nonlinear scales for the horizontal axis.

Figure 4.6 shows the initial scatterplot; Figure 4.7 shows the resulting scatterplot with the **Show Density Curves** and **Add Quantile Lines** options selected displaying the curves and the lines for the various Temp levels for the Weibull distribution. You can also view density curves across all the levels of Temp for the other distributions. These distributions can be selected one at a time or can be viewed simultaneously by checking the boxes to the left of the desired distribution name(s).

Figure 4.7 Scatterplot with Density Curve and Quantile Line Options



Nonparametric Overlay

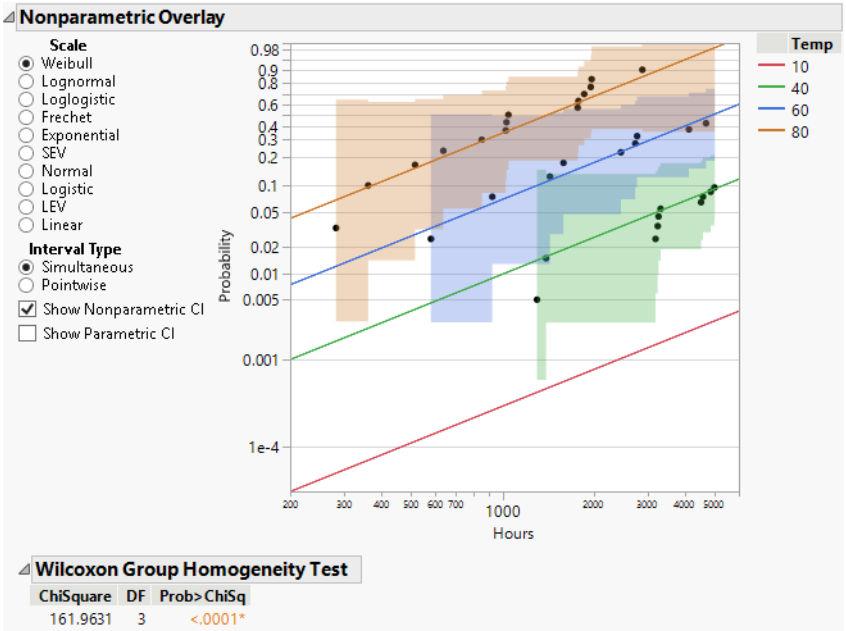
The Nonparametric Overlay plot is displayed after the scatterplot. Differences among groups can readily be detected by examining this plot. You can choose different scales for viewing these differences. You can change the interval type on a Nonparametric fit probability plot between **Simultaneous** and **Pointwise** (results displayed when **Show Nonparametric CI** is selected). You can also select whether to show parametric or nonparametric confidence intervals.

Pointwise estimates show the pointwise 95% confidence bands on the plot while simultaneous confidence intervals show the simultaneous confidence bands for all groups on the plot. Meeker and Escobar (1998, ch. 3) discuss pointwise and simultaneous confidence intervals and the motivation for simultaneous confidence intervals in a lifetime analysis.

Wilcoxon Group Homogeneity Test

For this example, the **Wilcoxon Group Homogeneity Test**, shown in Figure 4.8, indicates that there is a difference among groups. The high chi-square value and low *p*-value are consistent with the differences seen among the Temp groups in the Nonparametric Overlay plot.

Figure 4.8 Nonparametric Overlay Plot and Wilcoxon Test for Devalt.jmp



Comparisons

The Comparisons report section, shown in Figure 4.9, shows profilers for the selected distributions in the Nonparametric Overlay section, and includes the following tabs:

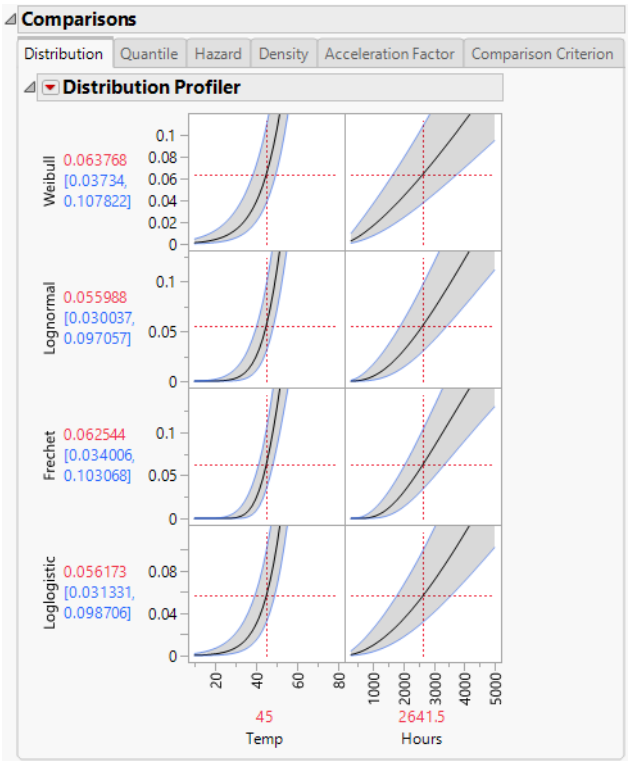
- Distribution
- Quantile
- Hazard
- Density
- Acceleration Factor
- Comparison Criterion

To show a specific profiler, select the appropriate distribution option in the Nonparametric Overlay section.

Profilers

The first five tabs show profilers for the selected distributions. Curves shown in the first four profilers correspond to both the time-to-event and explanatory variables. The Acceleration Factor profiler tab corresponds only to the acceleration factor (explanatory variable). Figure 4.9 shows the Distribution Profiler for the Weibull, Lognormal, Fréchet, and Loglogistic distributions.

Figure 4.9 Distribution Profiler

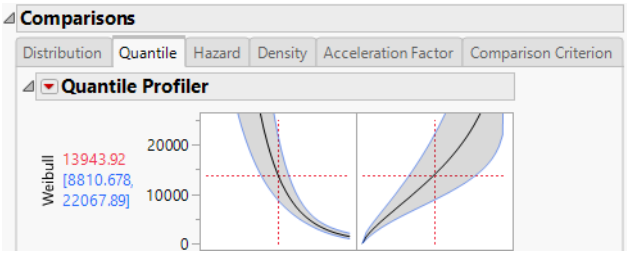


Comparable results appear on the **Quantile**, **Hazard**, and **Density** tabs. The Distribution, Quantile, Hazard, Density, and Acceleration Factor Profilers behave similarly to the Prediction Profiler in other platforms. For example, the vertical lines of Temp and Hours can be dragged to see how each of the distribution values change with temperature and time. For a detailed explanation of the Prediction Profiler, see *Profilers*.

Quantile

You can use the Quantile profiler for extrapolation. Suppose that the data are represented by a Weibull distribution. From viewing the Weibull Acceleration Factor Profiler in Figure 4.11, you see that the acceleration factor at 45 degrees Celsius is 17.42132 for a use condition temperature of 10 degrees Celsius. Select the **Quantile** tab to see the Quantile Profiler for the Weibull distribution. Select and drag the vertical line in the probability plot so that the probability reads 0.5. From viewing Figure 4.10, where the Probability is set to 0.5, you find that the quantile for the failure probability of 0.5 at 45 degrees Celsius is 13943.92 hours. So, at 10 degrees Celsius, you can expect that 50% of the units fail by 13943.92 * 17.42132 = 242921 hours.

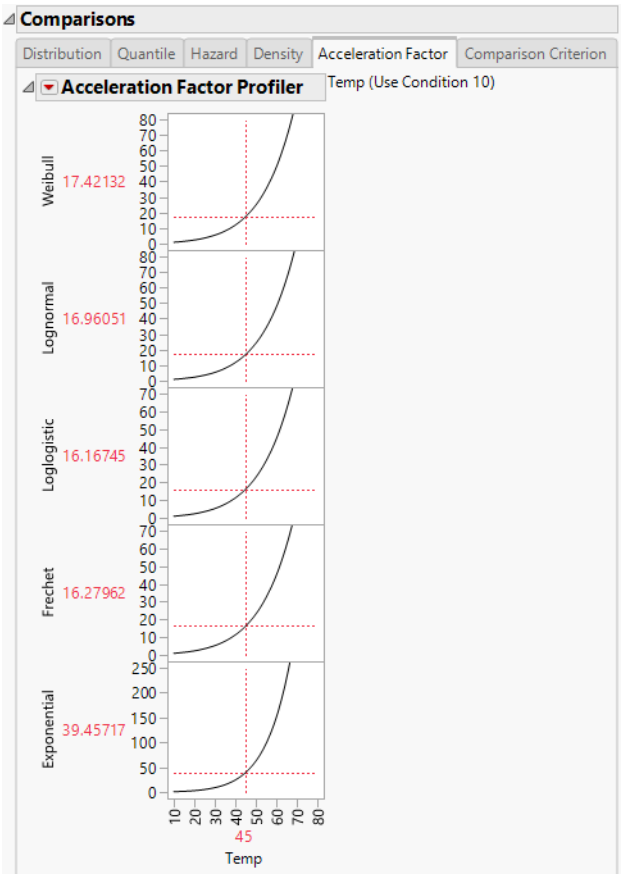
Figure 4.10 Weibull Quantile Profiler for Devalt.jmp



Acceleration Factor

Selecting the **Acceleration Factor** tab shows the Acceleration Factor Profiler for the time-to-event variable for each specified distribution. To produce Figure 4.11, select **Fit All Distributions** from the Fit Life by X red triangle menu. Modify the use condition value for the explanatory variable by selecting **Set Time Acceleration Use Condition** from the Fit Life by X red triangle menu and entering the desired value. Note that the explanatory variable and the use condition value appear beside the profiler title.

Figure 4.11 Acceleration Factor Profiler for Devalt.jmp



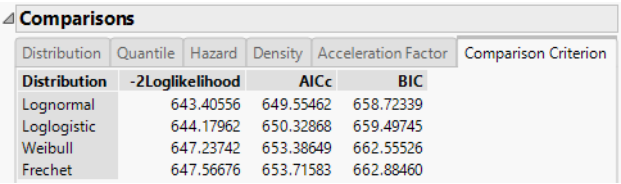
The Acceleration Factor Profiler lets you estimate time-to-failure for accelerated test conditions when compared with the use condition value and a parametric distribution assumption. The interpretation of a time-acceleration plot is generally the ratio of the p^{th} quantile of the use condition to the p^{th} quantile of the accelerated test condition. This relation applies only when the distribution is Lognormal, Weibull, Loglogistic, or Fréchet, and the scale parameter is constant for all levels. For more information about the parameterizations of the distributions, see "Distributions".

Note: The Acceleration Factor Profiler does not appear in the following instances: when the explanatory variable is discrete; the explanatory variable is treated as discrete; a customized formula does not use a unity scale factor; or the distribution is Normal, SEV, Logistic, or LEV.

Comparison Criterion

The **Comparison Criterion** tab shows the -2Loglikelihood, AICc, and BIC criteria for the distributions of interest. Figure 4.12 shows these values for the Weibull, Lognormal, Loglogistic, and Fréchet distributions. Distributions providing better fits to the data are shown at the top of the Comparisons report, sorted by AICc.

Figure 4.12 Comparison Criterion Report Tab



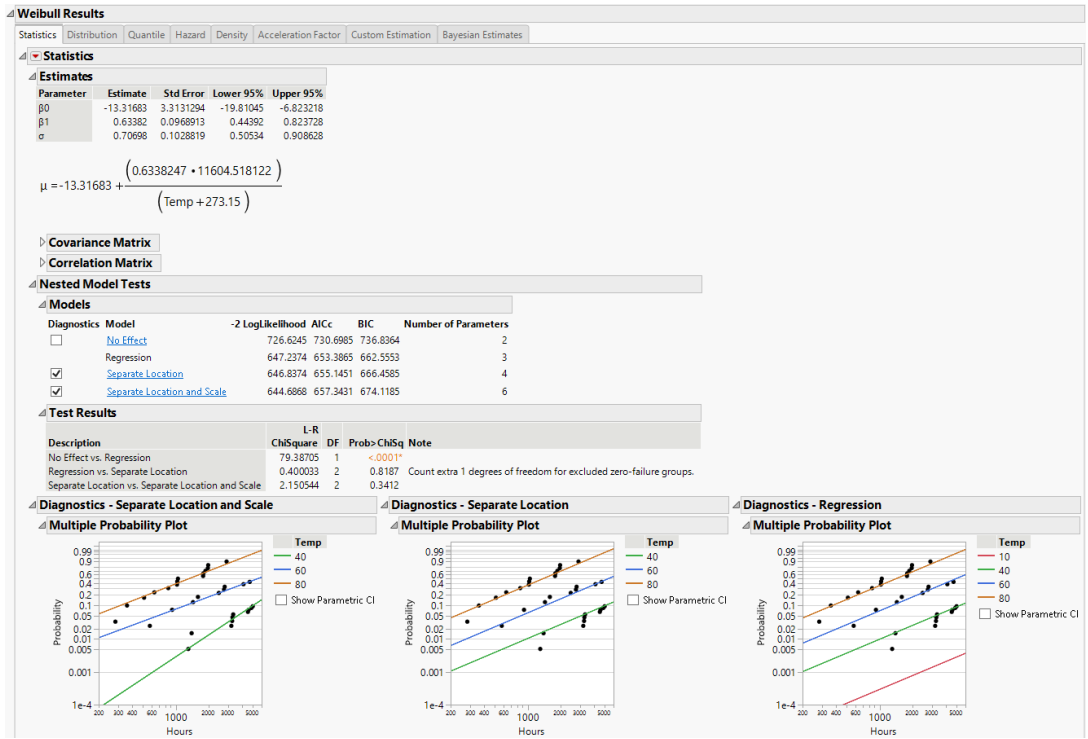
This report suggests that the Lognormal and Loglogistic distributions provide the best fits for the data, because the lowest criteria values are seen for these distributions. For more information about the criteria, see *Fitting Linear Models*.

Results

The Results section of the report window shows more detailed statistics and prediction profilers than those shown in the Comparisons report. Separate result sections are shown for each selected distribution. Figure 4.13 shows a portion of the Weibull results, Nested Model Tests, and Diagnostics plots for Devalt.jmp.

Statistical results, diagnostic plots, and Distribution, Quantile, Hazard, Density, and Acceleration Factor Profilers are included for each of your specified distributions. The Custom Estimation tab lets you estimate specific failure probabilities and quantiles, using both Wald and Profile interval methods. When the Box-Cox Relationship is selected on the platform launch window, the Sensitivity tab appears. This tab shows how the Relative Likelihood and B10 Life change as a function of Box-Cox lambda.

Figure 4.13 Weibull Distribution Nested Model Tests for Devalt.jmp Data



Statistics

For each parametric distribution, there is a Statistics section that shows parameter estimates, a covariance matrix, confidence intervals, summary statistics, and diagnostic plots. You can save probability, quantile, and hazard estimates by selecting any or all of these options from the Statistics red triangle menu for each parametric distribution. The estimates and the corresponding lower and upper confidence limits are saved as columns in your data table.

Nested Model Tests

Nested Model Tests are included, if you selected the option on the platform launch window. The Nested Model Tests include statistics and diagnostic plots for the following models:

Separate Location and Scale Assumes that the location and scale parameters are different for all levels of the explanatory variable. This option is equivalent to fitting the distribution by the levels of the explanatory variable. The Separate Location and Scale model has multiple location parameters and multiple scale parameters (Figure 4.14).

Separate Location Assumes that the location parameters are different, but the scale parameters are the same for all levels of the explanatory variable. The Separate Location model has multiple location parameters and only one scale parameter (Figure 4.15).

Regression The default model shown in the initial Fit Life by X report window (Figure 4.16).

No Effect Assumes that the explanatory variable does not affect the response. This option is equivalent to fitting all of the data values to the selected distribution. The No Effect Model has one location parameter and one scale parameter (Figure 4.17).

Separate Location and Scale, Separate Location, and Regression analyses results are shown by default. Regression parameter estimates and the location parameter formula are shown under the Estimates section, by default. The Diagnostics plots for the No Effect model can be displayed by selecting the check box to the left of No Effect under the Nested Model Tests title.

To see results for each of the models (independently of the other models), click the underlined model of interest (listed under Nested Model Tests) and then uncheck the check boxes for the other models.

If the Nested Model Tests option was not checked in the launch window, then the Separate Location and Scale and Separate Location models are not assessed. In this case, estimates are given for the regression model for each distribution that you select and the Cox-Snell Residual P-P Plot is the only diagnostic plot.

Note: When Separate Location and Scale or Separate Location models are fit for the Weibull distribution, both parameterizations of the Weibull distribution are shown in the Estimates table, as is the case in Figure 4.14 and Figure 4.15. For more information about the Weibull parameterizations, see "Weibull".

Diagnostics

The Multiple Probability Plots shown in Figure 4.13 are used to validate the distributional assumption for the different levels of the accelerating variable. If the line for each level does not run through the data points for that level, the distributional assumption might not hold. Side-by-side comparisons of the diagnostic plots provide a visual comparison for the validity of the different models. See Meeker and Escobar (1998, sec. 19.2.2) for a discussion of multiple probability plots. Each multiple probability plot has an option below the legend that enables you to show or hide shaded parametric confidence intervals for each line in the plot.

The Cox-Snell Residual P-P Plots are used to validate the distributional assumption for the data. If the data points deviate far from the diagonal, then the distributional assumption might be violated. The Cox-Snell Residual P-P Plot red triangle menu has an option called **Save Residuals** that enables you to save the residual data to the data table. See Meeker and Escobar (1998, sec. 17.6.1) for a discussion of Cox-Snell residuals.

The Residuals versus Fitted Plots are used to validate the distributional assumption for the different levels of the accelerating variable. The plots show standardized residuals on the vertical axis and the X variable on the horizontal axis. The Residuals vs Fitted red triangle menu has an option called **Save Residuals** that enables you to save the standardized residual data to the data table.

Figure 4.14 Separate Location and Scale Model with the Weibull Distribution for Devalt.jmp Data

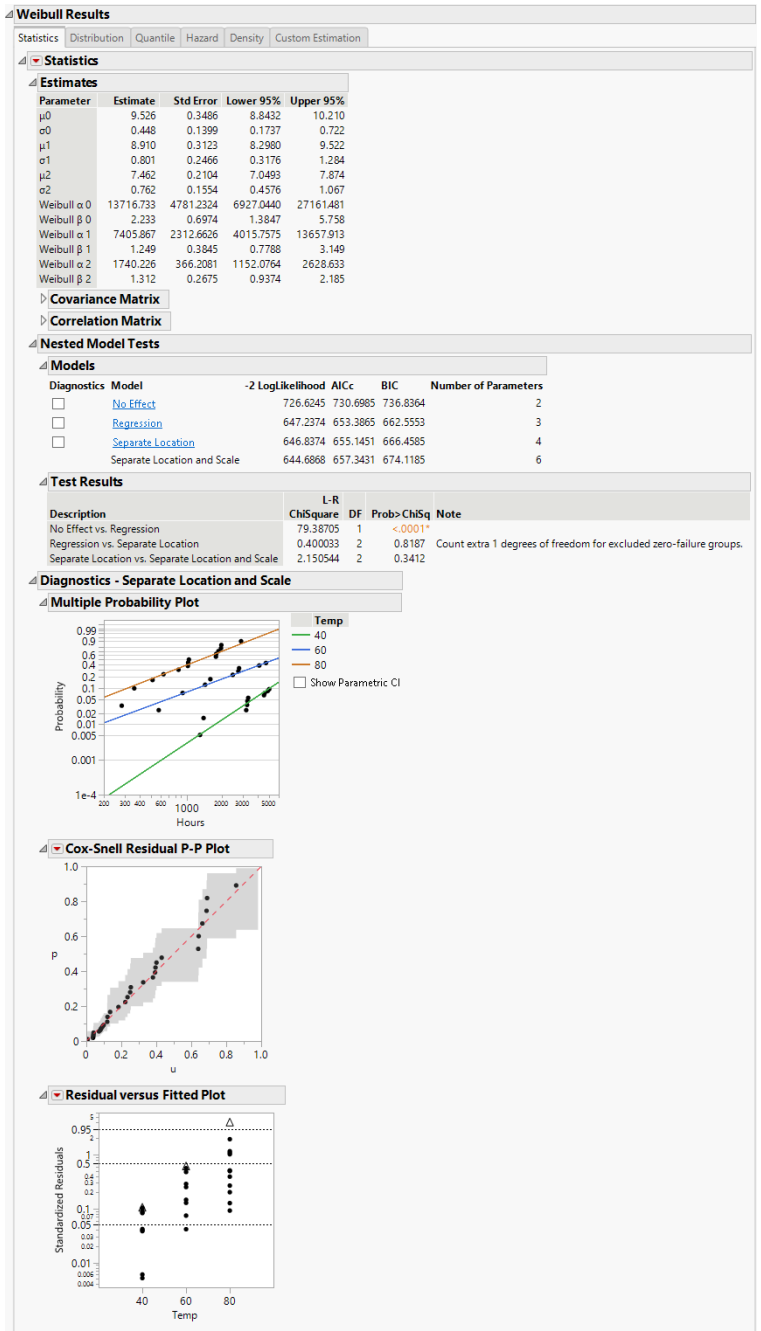


Figure 4.15 Separate Location Model with the Weibull Distribution for Devalt.jmp Data

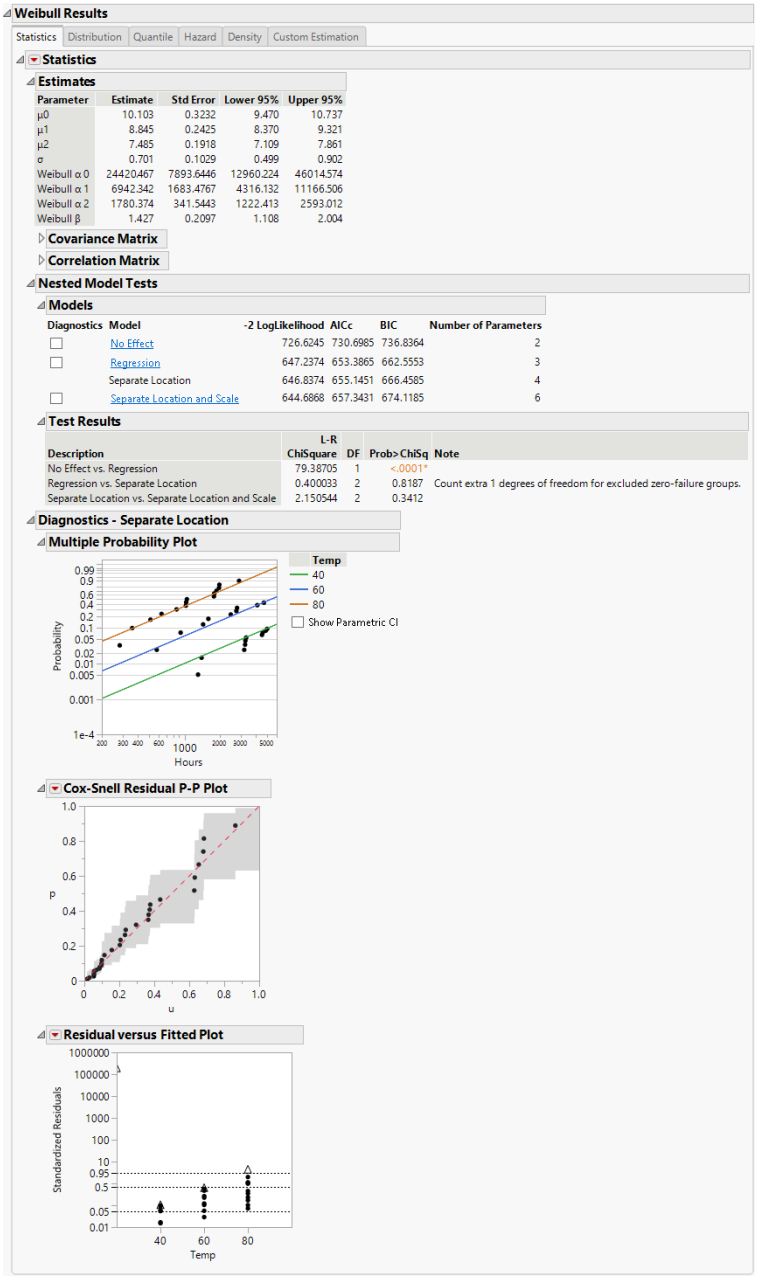


Figure 4.16 Regression Model with the Weibull Distribution for Devalt.jmp Data

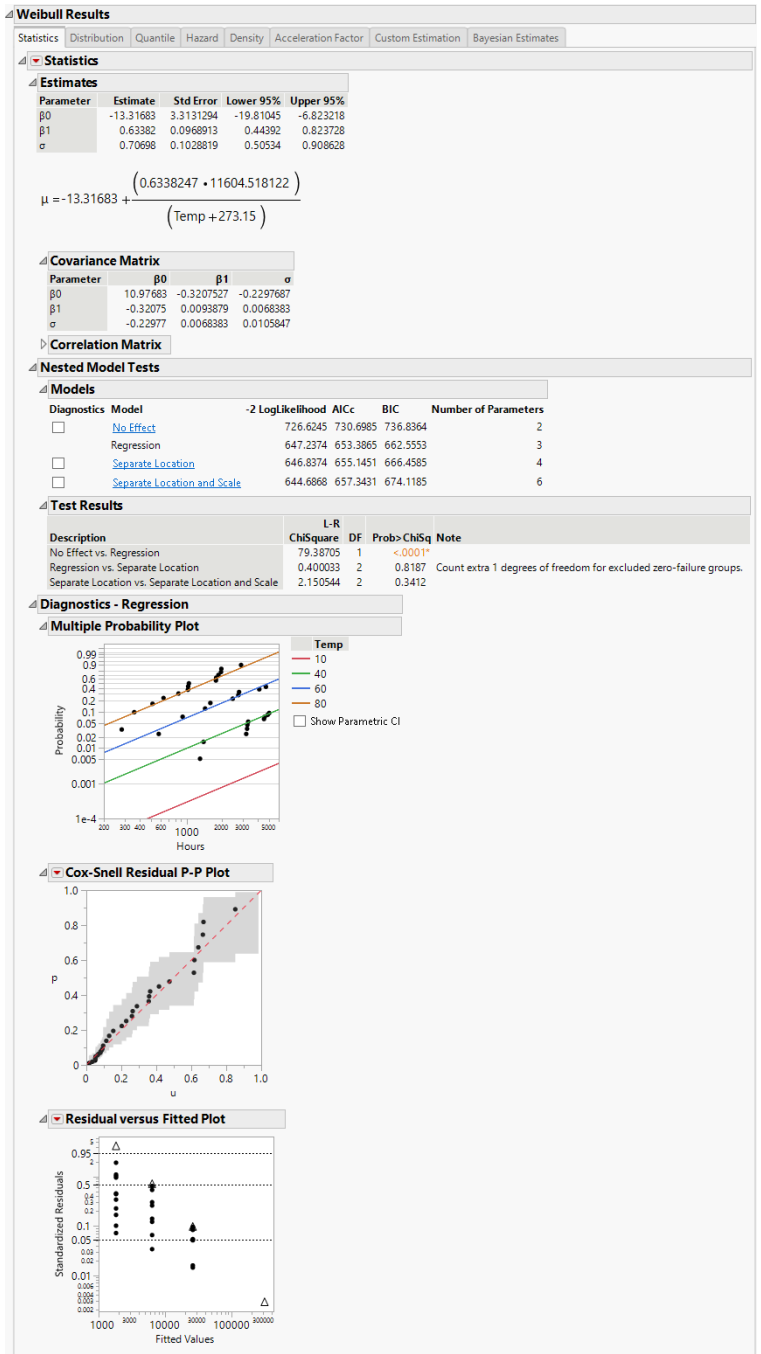
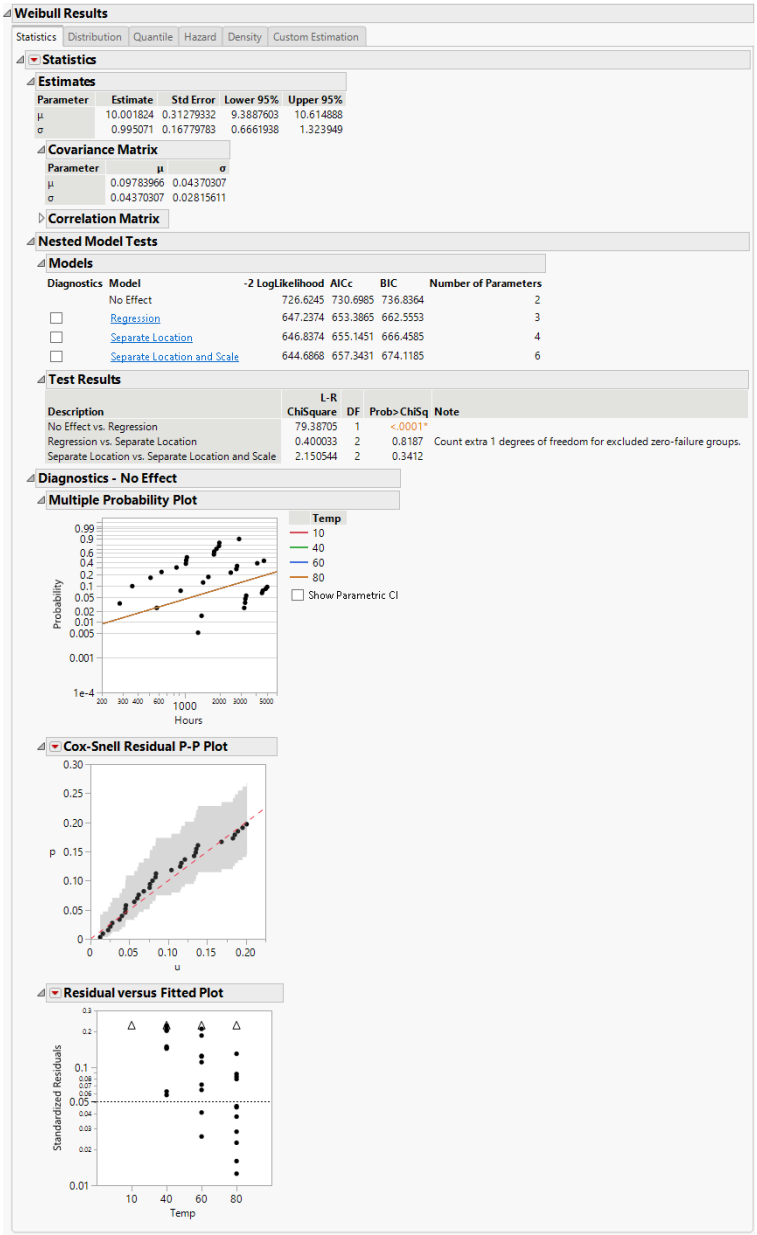


Figure 4.17 No Effect Model with the Weibull Distribution for Devalt.jmp Data

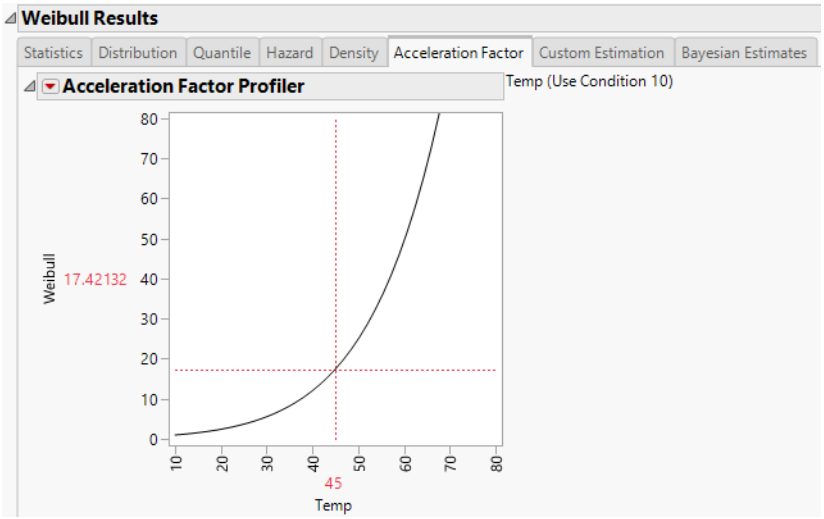


Profilers and Surface Plots

In addition to a statistical summary and diagnostic plots, the Fit Life by X report window also includes profilers and surface plots for each of your specified distributions. To view the Weibull time-accelerating factor and explanatory variable profilers, click the **Distribution** tab under Weibull Results. To see the surface plot, click the disclosure icon to the left of the Weibull title (under the profilers). The profilers and surface plot behave similarly to other platforms. See *Profilers*.

The report window also includes a tab labeled **Acceleration Factor**. Clicking the **Acceleration Factor** tab shows the Acceleration Factor Profiler. This profiler is an enlargement of the Weibull plot shown under the **Acceleration Factor** tab in the Comparisons section of the report window. Figure 4.18 shows the Acceleration Factor Profiler for the Weibull distribution of Devalt.jmp. The use condition level for the explanatory variable can be modified by selecting the **Set Time Acceleration Use Condition** option in the Fit Life by X red triangle menu.

Figure 4.18 Weibull Acceleration Factor Profiler for Devalt.jmp



Custom Estimation

For each parametric distribution, there is a Custom Estimation section that contains two reports: Estimate Quantile and Estimate Probability. For distributions with positive support, the Custom Estimation section also contains an Estimate Mean Remaining Life (MRLF) report.

Estimate Quantile

The Estimate Quantile report contains a calculator that enables you to predict quantiles for specific failure probability values. In the Estimate Quantile calculator, enter a value for Prob and the X variable. Press **Enter** to see the quantile estimates and corresponding confidence intervals. To calculate multiple quantile estimates, click the plus sign, enter another Prob value, another X value, or both, and press **Enter**. Click the minus sign to remove the last entry. If you enter more than one value in either column, the table contains all combinations of Prob and X values.

By default, Wald-based intervals are shown. Click **Likelihood CI** in the Interval Type outline to switch to likelihood-based confidence intervals. The confidence level for these intervals is determined by the **Change Confidence Level** option in the Fit Life by X red triangle menu.

Estimate Probability

The Estimate Probability report contains a calculator that enables you to predict failure and survival probabilities for specific time values. In the Estimate Probability calculator, enter a value for Time and the X variable. Press **Enter** to see the failure probability estimates and corresponding confidence intervals. To calculate multiple failure probability estimates, click the plus sign, enter another Time value, another X value, or both, and press **Enter**. Click the minus sign to remove the last entry. If you enter more than one value in either column, the table contains all combinations of Time and X values.

By default, Wald-based intervals are shown. Click **Likelihood CI** in the Interval Type outline to switch to likelihood-based confidence intervals. The confidence level for these intervals is determined by the **Change Confidence Level** option in the Fit Life by X red triangle menu.

Estimate Mean Remaining Life (MRLF)

The Estimate Mean Remaining Life (MRLF) report contains a calculator that enables you to predict mean remaining life for specific time values. In the Estimate Mean Remaining Life (MRLF) calculator, enter a value for Survival Time and the X variable. Press **Enter** to see the mean remaining life estimates and corresponding Wald-based confidence intervals. The confidence level for these intervals is determined by the **Change Confidence Level** option in the Fit Life by X red triangle menu. To calculate multiple mean remaining life estimates, click the plus sign, enter another Survival Time value, another X value, or both and press **Enter**. Click the minus sign to remove the last entry. If you enter more than one value in either column, the table contains all combinations of Survival Time and X values.

Note: When the survival time equals zero, the calculated mean remaining life (MRLF) value is equivalent to the mean time to failure (MTTF) value.

Bayesian Estimates

For each parametric distribution other than Exponential, there is a Bayesian Estimates section that enables you to obtain Bayesian parameter estimates. The Bayesian Estimates section is not available if the Relationship in the Fit Life by X launch window is Custom, No Effect, Location, or Location and Scale.

Bayesian estimation in the Fit Life by X platform is done using rejection sampling or a Markov Chain Monte Carlo (MCMC) algorithm. More specifically, the platform attempts a basic rejection sampler. If the rejection sampler produces valid results, these results are reported. If the rejection sampler cannot produce valid results, the platform uses a random walk Metropolis-Hastings algorithm and adds a note to the top of the Bayesian Estimation report. See Robert and Casella (2004).

The initial report is a control panel where you can specify prior distributions for the parameters and control aspects of the simulation. To obtain posterior estimates of the parameters, specify the prior distributions and the simulation options, and then click Fit Model.

To specify the prior distributions of the parameters, you must specify information about a quantile of the distribution and the slope β_1 and scale σ parameters. (For the Weibull distribution, you specify the Weibull β rather than σ.) The quantile is defined by two values: the probability of the quantile and the value of the X variable at the specified quantile. The default Probability value is 0.10, but you can specify a value that corresponds to the quantile of interest. Specify information about the range of the prior distribution. For Normal and Lognormal prior distributions, the range is specified in terms of Lower and Upper 99% limits. For Uniform and Log-Uniform prior distributions, the range is specified in terms of the Lower and Upper limits. See Meeker and Escobar (1998). The initial values that are provided are estimates consistent with the maximum likelihood estimates in the Statistics section of the report.

The following options for the simulation appear below the prior distribution specification table:

Number of Monte Carlo Iterations Sets the sample size that will be drawn from the posterior distribution after a burn-in procedure is completed.

Random Seed Sets the initial state of the simulation. By default, it is the clock time. The number should be a positive integer greater than 1. If you specify 1, the current clock time is used.

Bayesian Estimates - Result <N> Report

After you specify prior distributions and the simulation options, click **Fit Model** to perform the simulation. A Bayesian Estimates - Result <N> report is provided for each simulation. This report contains the following headings:

Priors Shows the specifications that you entered in the Bayesian Estimates report to run the simulation. The Prior report also contains the random seed.

Posterior Estimates Shows five marginal statistics that describe the posterior distribution of β_0, β_1, σ, and the quantile. The marginal statistics are the median, 0.025 quantile (Lower Bound), 0.975 quantile (Upper Bound), mean, and standard deviation computed from the Monte Carlo samples. If the Weibull distribution is specified, this table contains the posterior estimate of the Weibull β instead of σ.

To compute statistics for other derived variables based on the posterior estimates of the generic parameters, click the Export Monte Carlo Samples link.

Posterior Scatter Plot Shows two scatter plots of values from the Monte Carlo simulation. The scatter plot on the left shows the values of the posterior parameters as they are specified in the Priors report. The scatter plot on the right shows the values of the posterior parameters as they are specified in the Posterior Estimates report.

Profilers Shows two profilers based on samples from the posterior distribution. The values shown in the profilers, at the specified values of the X and Time variables, are calculated using the following steps:

- For each set of sampled parameter values from the posterior distribution, the values of the cumulative distribution function and the quantile function are calculated at the specified values of the X and Time variables.
- The predicted values of the cumulative distribution function and the quantile function are the medians of the calculated values.
- The upper and lower confidence limits are the 0.025 and 0.975 quantiles of the calculated values. The confidence level for these limits is determined by the **Change Confidence Level** option in the Fit Life by X red triangle menu.

Distribution Profiler Shows the parametric cumulative distribution function as a function of the X variable and Time.

Quantile Profiler Shows the parametric quantile function as a function of the X variable and a specified probability.

Bayesian Estimates - Result <N> Options

The Bayesian Estimates - Result <N> red triangle menu contains the following options:

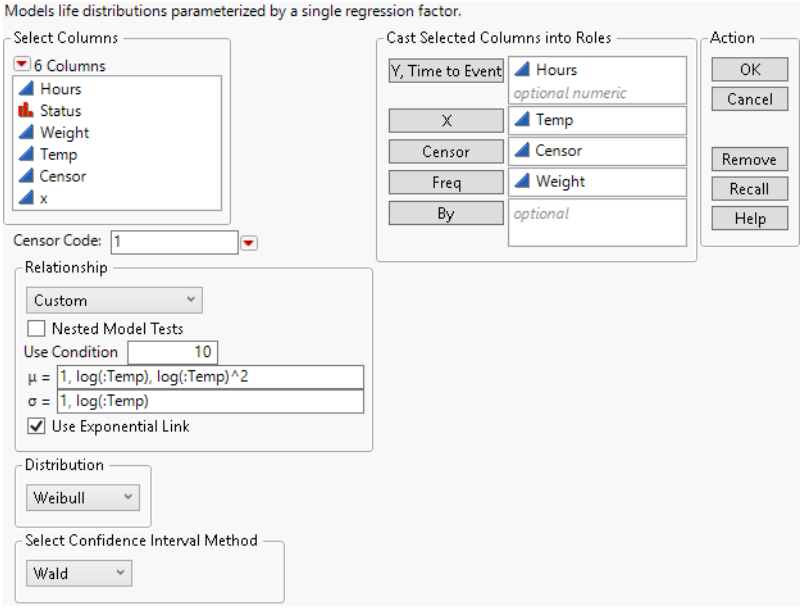
Remove Removes the current Bayesian Estimates report from the Fit Life by X report.

Export Monte Carlo Samples Saves the results of the Monte Carlo simulation to a new data table. You can use this table to compute statistics of the posterior estimates.

Custom Relationship

If you want to use a custom transformation to model the relationship between the lifetime event and the accelerating factor, use the **Custom** option. This option is found in the list under Relationship in the launch window. Enter comma delimited values into the entry fields for the location (μ) and scale (σ) parameters. For the Devalt.jmp sample data, an example entry for μ could be "1, log(:Temp), log(:Temp)^2", and an entry for σ could be "1, log(:Temp)", where 1 indicates that an intercept is included in the model. Select the **Use Exponential Link** check box to ensure that the sigma parameter is positive.

Figure 4.19 Custom Relationship Specification in Fit Life by X Launch Window



After selecting **OK**, location and scale transformations are created and included at the bottom of the Estimates report section.

Figure 4.20 Weibull Estimates and Formulas for Custom Relationship

Estimates

Parameter	Estimate	Std Error	Lower 95%	Upper 95%
β0	-0.603333	9.9924940	-20.18826	18.98160
β1	8.047095	4.8377341	-1.43469	17.52888
β2	-1.412340	0.5989816	-2.58632	-0.23836
λ0	-2.317331	2.0821317	-6.39823	1.76357
λ1	0.474054	0.5015868	-0.50904	1.45715

$$\mu = -0.6033328 + 8.047095 \cdot \text{Log}(\text{Temp}) + -1.41234 \cdot \text{Log}(\text{Temp})^2$$

$$\sigma = \text{Exp}\left(-2.317331 + 0.4740541 \cdot \text{Log}(\text{Temp})\right)$$

For an example of how to use a custom transformation, see "Custom Relationship Example". Analysis proceeds similarly to the "Example of the Fit Life by X Platform", where the **Arrhenius Celsius** Relationship was specified.

Fit Life by X Platform Options

The Fit Life by X red triangle menu contains the following options:

Fit Lognormal Fits a lognormal distribution to the data.

Fit Weibull Fits a Weibull distribution to the data.

Fit Loglogistic Fits a loglogistic distribution to the data.

Fit Fréchet Fits a Fréchet distribution to the data.

Fit Exponential Fits an exponential distribution to the data.

Fit SEV Fits a smallest extreme value (SEV) distribution to the data.

Fit Normal Fits a normal distribution to the data.

Fit Logistic Fits a logistic distribution to the data.

Fit LEV Fits a largest extreme value (LEV) distribution to the data.

Fit All Distributions Fits all distributions to the data.

Set Time Acceleration Use Condition Enables you to enter a use condition value for the explanatory variable of the acceleration factor in a pop-up window.

Change Confidence Level Enables you to enter a desired confidence level, for the plots and statistics, in a pop-up window. The default confidence level is 0.95.

Tabbed Report Lets you specify how you want the report window displayed. Two options are available: Tabbed Overall Report and Tabbed Individual Report. Tabbed Individual Report is checked by default. You can select one, both or none.

Show Surface Plot Shows or hides the surface plot for the distribution on and off in the individual distribution results section of the report. The surface plot is shown in the Distribution, Quantile, Hazard, and Density sections for the individual distributions, and it is on by default.

Show Points Shows or hides the data points on and off in the Nonparametric Overlay plot and in the Multiple Probability Plots. The points are shown in the plots by default. If this option is unchecked, step functions are shown instead.

Scatterplot Shows a scatterplot of a lifetime event versus an explanatory variable.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Examples of the Fit Life by X Platform

- "Capacitor Example"
- "Custom Relationship Example"

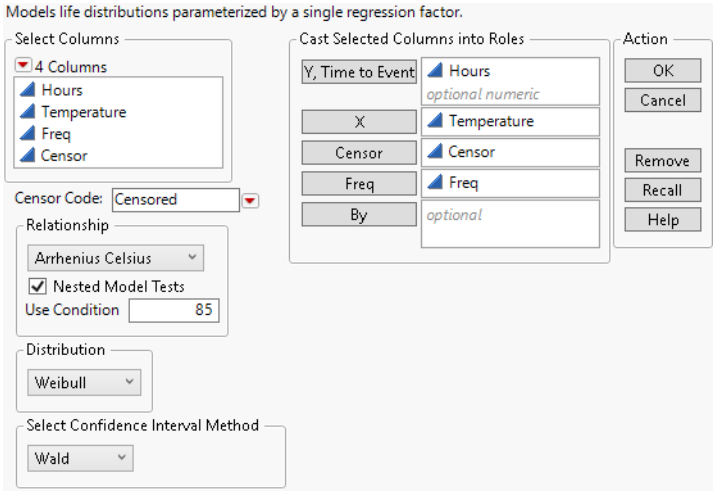
Capacitor Example

This example uses Capacitor ALT.jmp and can be found in the Reliability folder of the sample data. It contains simulated data that gives censored observations for three levels of temperature, for a reliability study. Observations are shown as censored values at temperature levels of 85, 105, and 125 degrees Celsius.

1. Select **Help > Sample Data Library** and open Reliability/Capacitor ALT.jmp.

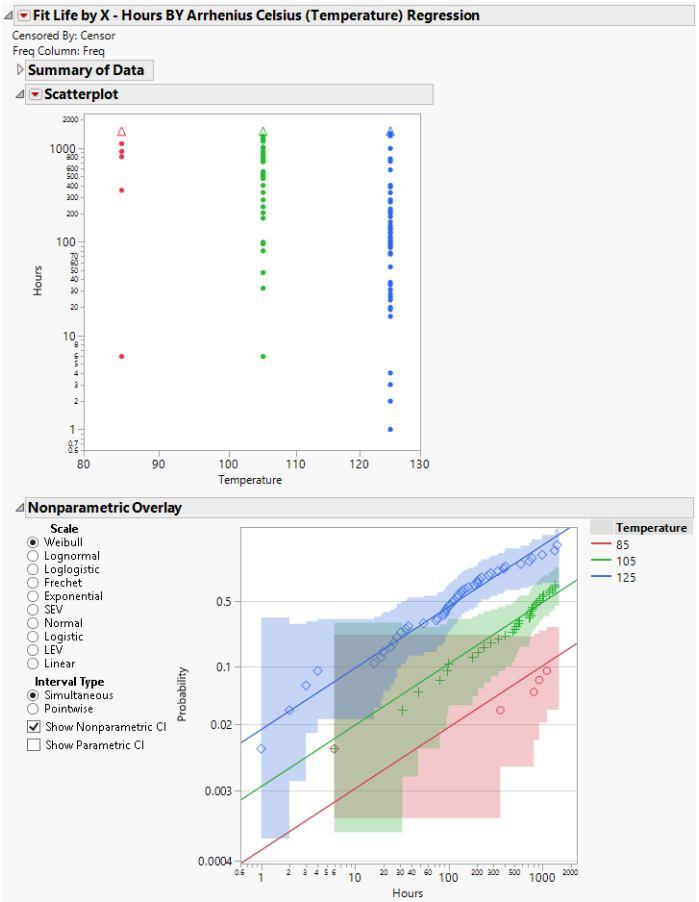
2. Select **Analyze** > **Reliability and Survival > Fit Life by X**.
3. Select Hours and click **Y, Time to Event**.
4. Select Temperature and click **X**.
5. Select Censor and click Censor.
6. Leave the **Censor Code** as Censored.
7. Select Freq as **Freq**.
8. Keep **Arrhenius Celsius** as the relationship, and keep the **Nested Model Tests** option selected.
9. Select **Weibull** as the distribution.
10. Keep **Wald** as the confidence interval method.

Figure 4.21 Fit Life by X Launch Window



11. Click **OK**.

Figure 4.22 Fit Life by X Report Window for Capacitor ALT.jmp Data



The report window shows summary data, diagnostic plots, comparison data and results, including detailed statistics and prediction profilers. Separate result sections are shown for each selected distribution. Distribution, Quantile, Hazard, Density, and Acceleration Factor Profilers are included for each of the specified distributions.

Custom Relationship Example

To create a quadratic model with Log(Temp) for the Weibull location parameter and a log-linear model with Log(Temp) for the Weibull scale parameter, follow these steps:

1. Select **Help > Sample Data Library** and open Reliability/Devalt.jmp.
2. Select **Analyze** > **Reliability and Survival** > **Fit Life by X**.
3. Select Hours and click **Y, Time to Event**.
4. Select Temp and click **X**.

5. Select Censor and click **Censor**.
6. Select Weight and click **Freq**.
7. Select **Custom** as the Relationship from the list.
8. In the entry field for μ, enter 1, log(:Temp), log(:Temp)^2.

   (The 1 indicates that an intercept is included in the model.)
9. In the entry field for σ, enter 1, log(:Temp).
10. Select the check box for **Use Exponential Link**.
11. Deselect the check box for **Nested Model Tests**.
12. In the entry field for Use Condition, enter 10.
13. Select **Weibull** as the Distribution.

    Figure 4.23 shows the completed launch window using the **Custom** option.

Note: The Nested Model Tests check box is not checked for non-constant scale models. Nested Model test results are not supported for this option.

14. Click **OK**.

Figure 4.23 Custom Relationship Specification in Fit Life by X Launch Window

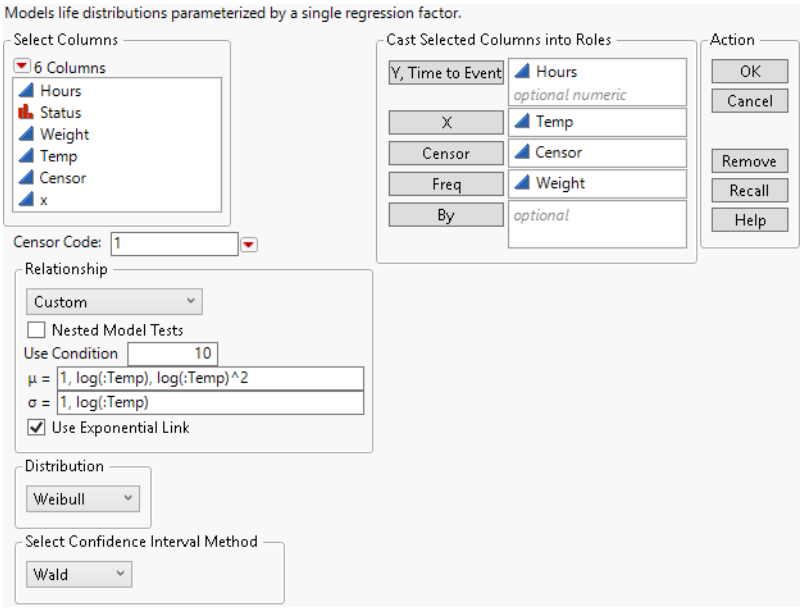


Figure 4.24 shows the location and scale transformations, which are subsequently created and included at the bottom of the Estimates report section.

Analysis proceeds similarly to the "Example of the Fit Life by X Platform", where the **Arrhenius Celsius** Relationship was specified.

Figure 4.24 Weibull Estimates and Formulas for Custom Relationship

Estimates

Parameter	Estimate	Std Error	Lower 95%	Upper 95%
β0	-0.603333	9.9924940	-20.18826	18.98160
β1	8.047095	4.8377341	-1.43469	17.52888
β2	-1.412340	0.5989816	-2.58632	-0.23836
λ0	-2.317331	2.0821317	-6.39823	1.76357
λ1	0.474054	0.5015868	-0.50904	1.45715

$$\mu = -0.6033328 + 8.047095 \cdot \text{Log}\left(\text{Temp}\right) + -1.41234 \cdot \text{Log}\left(\text{Temp}\right)^2$$

$$\sigma = \text{Exp}\left(-2.317331 + 0.4740541 \cdot \text{Log}\left(\text{Temp}\right)\right)$$

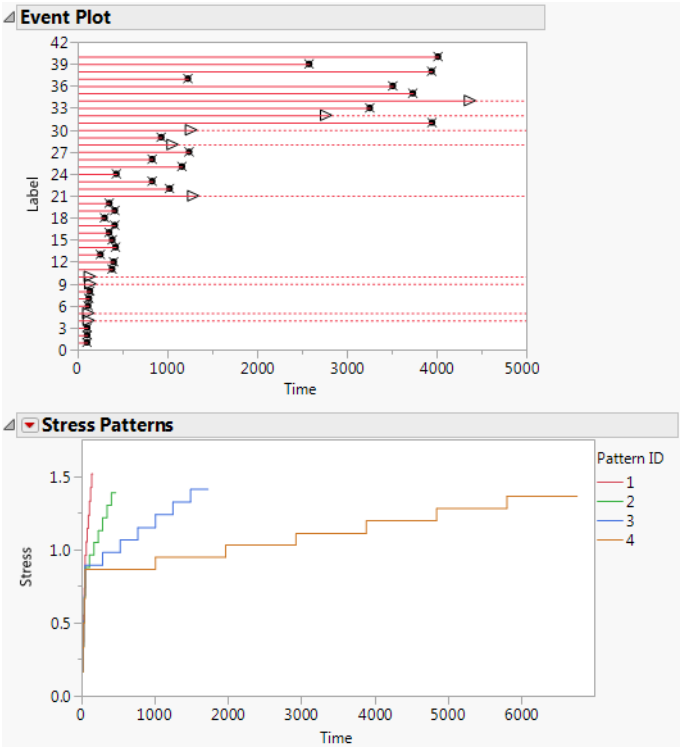
Chapter 5

Cumulative Damage

Model Product Deterioration with Variable Stress over Time

Cumulative damage models, which include step-stress models, enable you to analyze an accelerated life test where the stress levels might be changed over time. The stress can be applied by many different forces: load, temperature, pressure. A typical cumulative damage experiment consists of multiple test units. Each unit has an initial stress level, and the stress level can be changed throughout the experiment. The response is the failure time or time-to-event. The platform plots the failure events and enables you to fit multiple distributions to your data.

Figure 5.1 Example of Cumulative Damage Report



Contents

Overview of the Cumulative Damage Platform 139
Example of the Cumulative Damage Platform 139
Launch the Cumulative Damage Platform 143
Time to Event 144
Stress Pattern 145
The Cumulative Damage Report 148
Event Plot 148
Stress Patterns Report 148
Model List 149
Model Results 149
Cumulative Damage Platform Options 150
Stress Patterns Options 150
Additional Example of the Cumulative Damage Platform 152
Example of Simulating New Data 152

Overview of the Cumulative Damage Platform

A cumulative damage experiment, also called a *varying-stress experiment*, is an accelerated life test where the stress levels can change over time. The stress can be applied by many different forces: load, temperature, pressure. A typical cumulative damage experiment consists of multiple test units. Each unit has an initial stress level, and the stress level can be changed throughout the experiment.

The most common cumulative damage experiment is a *step-stress experiment*. A step-stress experiment uses multiple units with varying levels of stress applied. Stress can be applied using factors such as temperature, pressure, or voltage. For each unit, there is an initial stress level. At specified time points, the stress levels are adjusted based on different patterns of stress levels. Between stress level changes, the stress level remains constant.

The Cumulative Damage platform also includes three other varying-stress pattern models:

- In a ramp-stress experiment, the stress levels start at an initial value and then increase linearly over time at a specified slope.
- In a sinusoid-stress experiment, the stress levels fluctuate in a periodic fashion that is defined by a sine wave.
- In a piecewise ramp-stress experiment, the stress levels are defined at specified time points similar to the step-stress case. However, the stress level is not required to stay constant between time points. Rather, it changes linearly from a starting stress level to an ending stress level between time points. If a pair of starting and ending stress levels are equal, the interval is equivalent to a step-stress interval.

For more information about varying-stress and step-stress models, see Nelson (2004, ch. 10).

Example of the Cumulative Damage Platform

A step-stress experiment is conducted on 40 test units at varying stress conditions. Your goal is to estimate the probability of failure at 10,000 time units given a stress level of 0.75. These data tables are based on data from Nelson (2004, ch. 10).

The Reliability/CD Step Stress Pattern.jmp data table contains a column called Pattern ID that identifies four different stress patterns. The stress level at a particular step is the ratio of Voltage to Thickness. (Note that these two columns are hidden.) Thickness is held constant for each stress pattern. However, Voltage is set to different levels and increases within each pattern.

The Reliability/CD Step Stress.jmp data table contains the time to failure data.

1. Select **Help > Sample Data Library** and open Reliability/CD Step Stress.jmp and Reliability/CD Step Stress Pattern.jmp.

   The CD Step Stress table contains failure time data:

   – The Time column gives the failure times.

   – The Pattern ID column identifies the stress pattern.

   – The Censor column indicates whether the failure time is exact or censored.

   Each row of the table corresponds to one test unit.

   The CD Step Stress Pattern table contains the four stress patterns (identified as 1 through 4). The levels of the stress factor, Stress, are varied within each value of the Pattern ID column. The Duration column represents how many time units a particular level of the stress factor lasted.

2. Select **Analyze > Reliability and Survival > Cumulative Damage**.

   The launch window has two sections: one for the failure time data (Time-to-Event) and one for the stress pattern data (Stress Pattern).

3. Click **Select Table** in the Time-to-Event panel.

   A Time-to-Event Data Table window appears, which prompts you to specify the data table for the failure time data.

4. Select CD Step Stress and click **OK**.

   The columns from this table now populate the **Select Columns** list in the Time-to-Event panel.

5. Select Time and click **Time to Event**.

6. Select Censor and click **Censor**.

7. Select Pattern ID for **Pattern ID**.

8. Click **Select Table** in the Stress Pattern panel.

9. Select CD Step Stress Pattern and click **OK**.

   The columns from this table now populate the **Select Columns** list in the Stress Pattern panel.

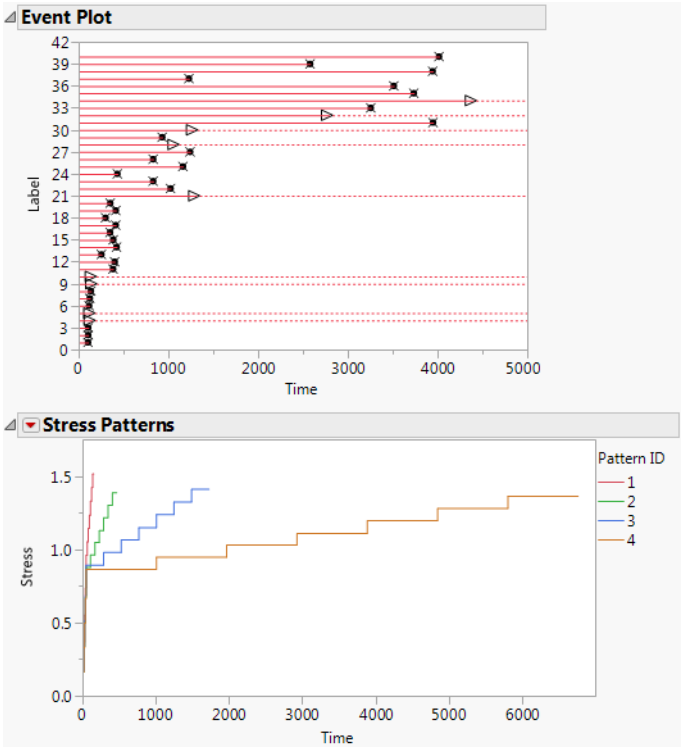
10. Select Duration and click **Stress Duration**.

11. Select Stress and click **Stress**.

12. Select Pattern ID and click **Pattern ID**.

13. Click **OK**.

Figure 5.2 Event Plot and Stress Patterns Plot



The initial report contains the Event Plot and a plot of the defined stress patterns. All four stress patterns increase the stress level quickly over the first 40 time units, after which they increase at much different rates.

14. Click the Cumulative Damage red triangle and select **Fit All**.

Figure 5.3 Model List Report

Model List

Distribution	-2 Log Likelihood	Nparm	AICc	BIC
Exponential	398.48312	2	402.80745	405.86088
Weibull	398.4822	3	405.14887	409.54884
Loglogistic	400.93264	3	407.59931	411.99928
Lognormal	402.3488	3	409.01547	413.41544
Frechet	410.92577	3	417.59243	421.99241

From the Model List report, you determine that the best fitting distribution is the Exponential distribution.

15. In the Results report, scroll to the Exponential report.
16. In the Distribution Profiler report, set the current value of Stress to 0.75.
17. Set the current value of Time to 10000.

Figure 5.4 Distribution Profiler for Exponential Distribution at Specified Settings

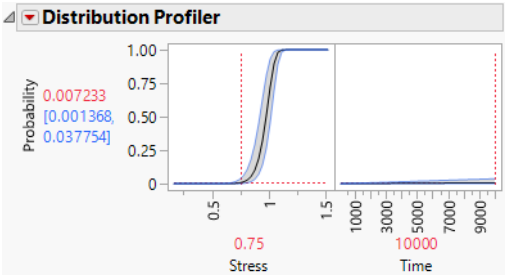
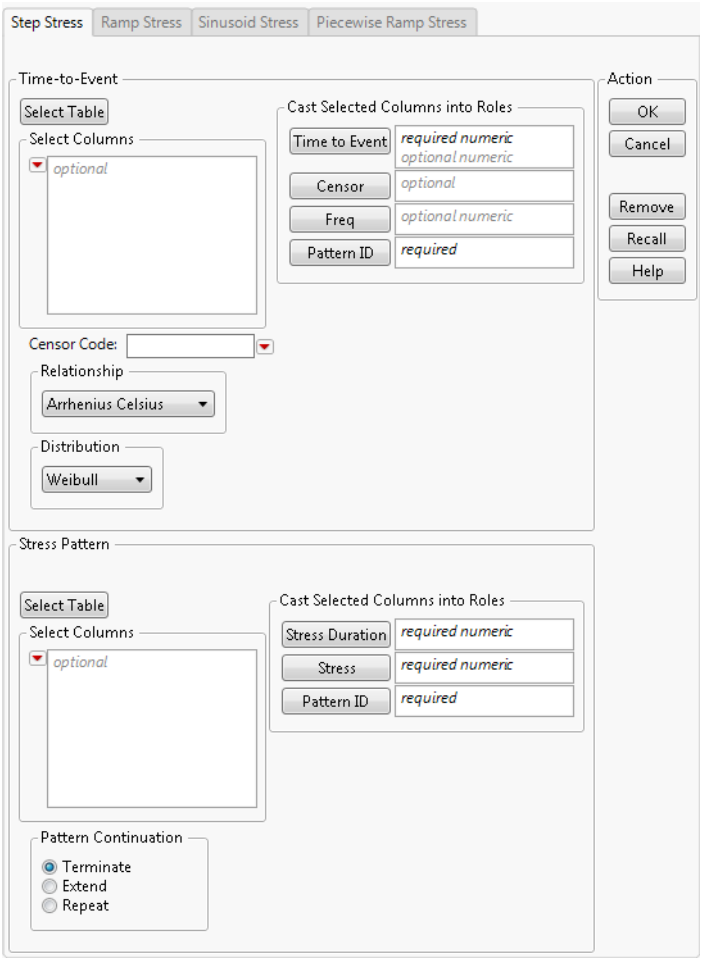


Figure 5.4 shows that the predicted probability of failure for a test unit under constant stress of 0.75 at 10000 time units is 0.007233, with a 95% confidence interval of 0.001368 to 0.037754.

Launch the Cumulative Damage Platform

Launch the Cumulative Damage platform by selecting **Analyze > Reliability and Survival > Cumulative Damage**. You must supply the Cumulative Damage platform with two data tables as input. The first data table is time-to-event data for each unit under test. The second data table defines the stress patterns used for each unit.

Figure 5.5 The Cumulative Damage Launch Window



The launch window includes a separate tab for each step-stress data format. For more information about the various stress patterns, see "Stress Pattern". For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Each of the step-stress format tabs contains two panels for specifying variables for the model:

- The Time-to-Event panel is common to all of the step-stress formats. This panel is similar to the Fit Life by X platform launch window.
- The Stress Pattern panel is used to describe the type of stress. Consequently, its format depends on the selected step-stress tab.

Time to Event

The Time to Event panel in the Cumulative Damage launch window contains the following options:

Time to Event Identifies the time to event (such as the time to failure) or time to censoring. For interval censoring, your data table should contain two columns, where one gives the lower bound and the other gives the upper bound of the failure time for each unit. Enter the two censoring columns as Time to Event. For more information about censoring, see "Event Plot".

Censor Identifies right-censored observations. Select the value that identifies right-censored observations from the Censor Code menu beneath the Select Columns list. The Censor column is used only when one Y is entered.

Freq Identifies frequencies or observation counts when there are multiple units. If the value is 0 or a positive integer, then the value represents the frequencies or counts of observations with the given row's settings.

Pattern ID Contains values that specify the stress pattern in the Stress Pattern data table that was used for the given row.

Censor Code Identifies the value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

Relationship Identifies the relationship between the event and the stress factor. Table 5.1 defines the model for each relationship.

Table 5.1 Models for Relationship Options

Relationship	Model
Arrhenius Celsius	$\mu = b0 + b1 * 11604.5181215503 / (X + 273.15)$

Table 5.1 Models for Relationship Options *(Continued)*

Relationship	Model
Arrhenius Fahrenheit	μ = b0 + b1 * 11604.5181215503 / ((X + 459.67) / 1.8)
Arrhenius Kelvin	μ = b0 + b1 * 11604.5181215503 / X
Inverse Power (default)	μ = b0 + b1 * log(X)
Linear	μ = b0 + b1 * X
Log	μ = b0 + b1 * log(X)
Logit	μ = b0 + b1 * log(X / (1 - X))
Reciprocal	μ = b0 + b1 / X
Square Root	μ = b0 + b1 * sqrt(X)
Box-Cox	μ = b0 + b1 * BoxCox(X)
Custom	user-defined (Available only in the Step Stress panel.)

The BoxCox(X) transformation is defined as follows:

$$x_i^{(\lambda)} = \begin{cases} \dfrac{x_i^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln(x_i) & \text{if } \lambda = 0 \end{cases}$$

If you select **Custom**, additional controls appear that require you to define the Custom transformation that models the relationship between the lifetime event and the stress factor.

If you want to use a Custom relationship for your model, see "Custom Relationship".

Distribution Specifies an initial time-to-failure distribution. Select from Weibull, Lognormal, Loglogistic, Fréchet, or Exponential. Weibull is the default setting. For more information about the distributions, see "Distributions".

Stress Pattern

Specify the stress patterns used in the experiment using the second panel.

Step Stress Pattern

The Step Stress pattern has stress levels that are changed at arbitrary time points. The duration of each stress step and associated stress level must be specified in ascending time order.

The Stress Pattern panel in the Step Stress tab contains the following options:

Stress Duration The column that contains the length in time units of each stress step.

Stress The column that contains the level of the stress setting.

Pattern ID The column that contains a unique identifier for the stress pattern. This column is used to match stress patterns in the Stress Pattern data table with the observations in the Time-to-Event data table.

Pattern Continuation Specifies how to handle failures that occur after the final time period in the defined stress pattern. This panel contains the following options:

Terminate A failure that occurs at a time beyond the final time period in the defined stress pattern produces an error, and the model is not fit.

Extend A failure that occurs at a time beyond the final time period in the defined stress pattern assumes the same stress level as the level in the final time period.

Repeat A failure that occurs at a time beyond the final time period in the defined stress pattern assumes the same stress level as if the stress pattern were being repeated. For example, if a failure occurs 10 time units after the final time period in the defined stress pattern, then the stress level at that failure time is set to the stress level at 10 time units after the beginning of the defined stress pattern.

Note: The default Pattern Continuation setting for the Step Stress Pattern is Terminate.

Ramp Stress Pattern

The Ramp Stress pattern defines stress as a linear function of time. Each pattern is defined by an intercept (the stress level at time zero) and a slope (the increase in the stress level for every one time unit). Each pattern is described in a single row in the stress pattern data table.

Intercept The column that contains the intercept for each pattern.

Slope The column that contains the slope for each pattern.

Pattern ID The column that contains a unique identifier for the stress pattern. This column is used to match stress patterns in the Stress Pattern data table with the observations in the Time-to-Event data table.

Sinusoid Stress Pattern

The Sinusoid Stress pattern defines stress as a periodic function. The pattern is defined by a level, an amplitude, a period, and a phase. Each pattern is described in a single row in the stress pattern data table. The pattern is defined as follows:

$$S(t) = \text{level} + \text{amplitude} * \sin(\text{phase} + (2*\pi*t) / \text{period})$$

Level The column that contains the level for each pattern.

Amplitude The column that contains the amplitude for each pattern.

Period The column that contains the period for each pattern.

Phase The column that contains the phase for each pattern.

Pattern ID The column that contains a unique identifier for the stress pattern. This column is used to match stress patterns in the Stress Pattern data table with the observations in the Time-to-Event data table.

Piecewise Ramp Stress Pattern

The Piecewise Ramp Stress pattern defines stress as a piecewise linear function of time. The line segments for the stress level over time can be disjoint or continuous. Line segments can also be flat, so that step stress and ramp stress can be combined. The line segments are defined in the stress pattern data table by the time duration of the segment and the start and end levels of the stress setting.

Stress Duration The column that contains the length in time units of each stress step.

Stress Ramp The two columns that contain the stress levels at the start and end of the step.

Pattern ID The column that contains a unique identifier for the stress pattern. This column is used to match stress patterns in the Stress Pattern data table with the observations in the Time-to-Event data table.

Pattern Continuation The Pattern Continuation panel enables you to specify the stress levels that occur after the final time period in the defined stress pattern. This panel contains the following options:

Terminate A failure that occurs at a time beyond the final time period in the defined stress pattern produces an error, and the model is not fit.

Extend A failure that occurs at a time beyond the final time period in the defined stress pattern assumes the same stress level as the level in the final time period.

Repeat A failure that occurs at a time beyond the final time period in the defined stress pattern assumes the same stress level as if the stress pattern were being repeated. For example, if a failure occurs 10 time units after the final time period in the defined stress pattern, then the stress level at that failure time is set to the stress level at 10 time units after the beginning of the defined stress pattern.

Note: The default Pattern Continuation setting for the Piecewise Ramp Stress Pattern is Terminate.

The Cumulative Damage Report

After you click OK, the Cumulative Damage report window appears. By default, the Cumulative Damage report contains an event plot, a stress patterns report, a model list, and model results.

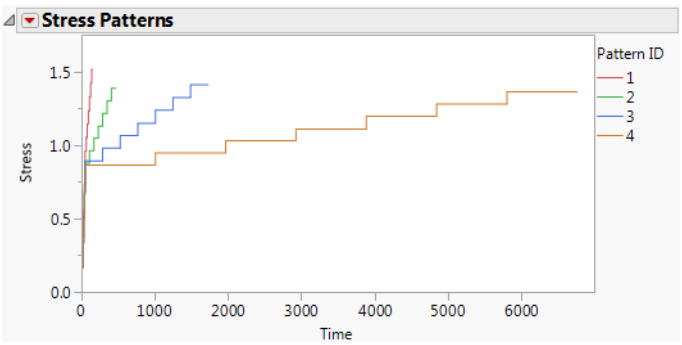
Event Plot

The Event Plot in Cumulative Damage displays time to failure or censoring. See "Event Plot".

Stress Patterns Report

The Stress Patterns report shows a plot of the stress level over time for each of the stress pattern IDs. The Simulate option in this report enables you to simulate new data. Figure 5.6 shows an example of the Stress Patterns report plot. See "Stress Patterns Options".

Figure 5.6 Stress Patterns Report



Model List

The Model List report provides the -2Loglikelihood, number of parameters, AICc, and BIC statistics for each fitted distribution. Smaller values of each of these statistics (other than number of parameters) indicate a better fit. For more information about these statistics, see *Fitting Linear Models*.

Model Results

The Results report contains a separate report for each fitted distribution. Each report contains the following:

Parameter Estimates Shows the estimates, standard errors, and Wald-based 95% confidence intervals.

The fitted equation appears below the Parameter Estimates table. This equation takes into account the fitted parameter estimates and the relationship specified in the launch window.

Distribution Profiler Shows cumulative failure probability as a function of time.

Quantile Profiler Shows failure time as a function of cumulative probability.

Hazard Profiler Shows the hazard rate as a function of time.

Density Profiler Shows the density function for the distribution.

Probability Plot of Standardized Residuals Shows a plot of standardized residuals for the model on a probability scale axis.

The Probability Plot of Standardized Residuals red triangle menu has a Save Residuals option that saves the standardized residuals to three new columns in the failure time data table. The three columns are Left Residuals, Right Residuals, and Residual Weight.

Cox-Snell Residual P-P Plot Shows a residual plot that is used to validate the distributional assumption for the data.

The Cox-Snell Residual P-P Plot red triangle menu has a Save Residuals option that saves the Cox-Snell residuals to three new columns in the failure time data table. The three columns are Left Residuals, Right Residuals, and Residual Weight. See Meeker and Escobar (1998, sec. 17.6.1) for a discussion of Cox-Snell residuals.

Cumulative Damage Platform Options

The Cumulative Damage red triangle menu contains the following options:

Fit All Fits the distributions that were not selected in the launch window. The available distributions are listed under "Distribution".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

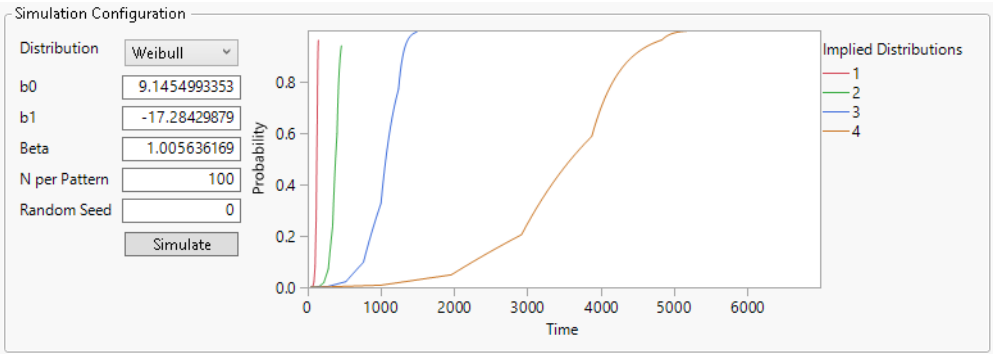
Stress Patterns Options

The Simulate option in the Stress Pattern red triangle menu shows or hides the Simulation Configuration panel.

The Simulation Control Panel

Figure 5.7 shows the Simulation Configuration Panel. The initial values for Distribution and the parameter settings are determined by the parameter estimates for the distribution specified in the Cumulative Damage launch window. The graph shows the estimated failure distribution functions over time for the stress patterns defined in the Stress Pattern data table.

Figure 5.7 Simulation Configuration Panel



The Simulation Configuration panel enables you to simulate new failure time data based on a distribution and stress pattern. The stress pattern defined in the Stress Pattern data table used in the launch of the platform is also used for the simulation. This panel contains the following options:

Distribution The distribution to be used for the simulation. The available distributions are the same as in the Cumulative Damage launch window. For more information about the distributions, see "Distributions".

b0 The intercept for the location parameter of the distribution.

b1 The slope for the location parameter of the distribution.

lambda (Available only for the Box-Cox relationship.) The lambda value for the Box-Cox relationship.

b2, s0, s1, and so on (Available only for the Custom relationship.) Other parameters that are defined in the Custom relationship in the Cumulative Damage launch window.

Beta (Available only for the Weibull distribution.) The Beta parameter of the Weibull distribution.

sigma (Available only for the Lognormal, Loglogistic, and Fréchet distributions.) The sigma parameter of the distribution.

N per Pattern The number of points generated in the simulation for each stress pattern.

Random Seed (Optional) A nonzero random seed that ensures the reproducibility of simulation results.

Termination (Not available when the specified Pattern Continuation in the Cumulative Damage launch window is Terminate.) A time beyond which surviving test units are censored.

Simulate

The plot in the Simulation Configuration panel shows the implied distributions for each of the stress patterns over time. Click the Simulate button to generate a new JMP data table that contains the results of the simulation.

Additional Example of the Cumulative Damage Platform

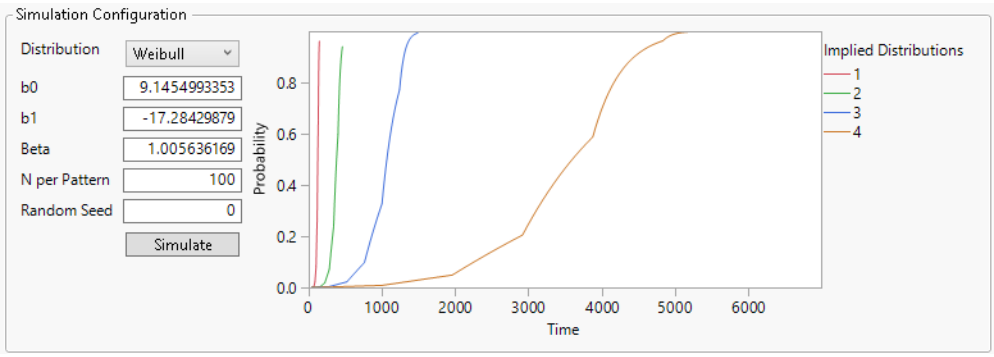
This section contains additional examples using the Cumulative Damage platform.

Example of Simulating New Data

This example illustrates using the Simulation Configuration panel in the Cumulative Damage report window to generate new step stress data. This example uses the same data as the example in "Example of the Cumulative Damage Platform".

1. Select **Help > Sample Data Library** and open Reliability/CD Step Stress.jmp and Reliability/CD Step Stress Pattern.jmp.
2. In the CD Step Stress data table, run the script Cumulative Damage.
3. Click the Stress Patterns red triangle and select **Simulate**.

Figure 5.8 Simulation Configuration Panel



The Simulation Configuration panel appears in the Stress Patterns report. The selection for Distribution is Weibull. The fitted values for b0 and b1 are used as initial values for the simulation.

4. Select Exponential for **Distribution**.
5. Enter 10 for **b0**.

6. Enter -18 for **b1**.
7. (Optional) Enter 14678 for **Random Seed**.
8. Click **Simulate**.

Figure 5.9 Partial Results of Simulation

	Time Left	Time Right	Pattern ID
1	140.67948106	140.67948106	1
2	139.34770497	139.34770497	1
3	136.95430716	136.95430716	1
4	136.30050091	136.30050091	1
5	145	•	1
6	118.5123739	118.5123739	1
7	119.03562567	119.03562567	1
8	145	•	1
9	127.19420127	127.19420127	1
10	118.48391055	118.48391055	1
11	103.54004804	103.54004804	1
12	145	•	1
13	121.04292162	121.04292162	1
14	136.15486483	136.15486483	1
15	143.65666248	143.65666248	1
16	123.04501594	123.04501594	1
17	145	•	1
18	124.26692664	124.26692664	1
19	120.96407573	120.96407573	1

Figure 5.9 shows a partial listing of the simulated data table. The stress pattern with Pattern ID equal to 1 is defined only up to 145 time units. Since the Pattern Configuration setting in the launch window was set to Terminate, the simulation censors any simulated values at 145 for stress pattern 1.

Chapter 6

Recurrence Analysis

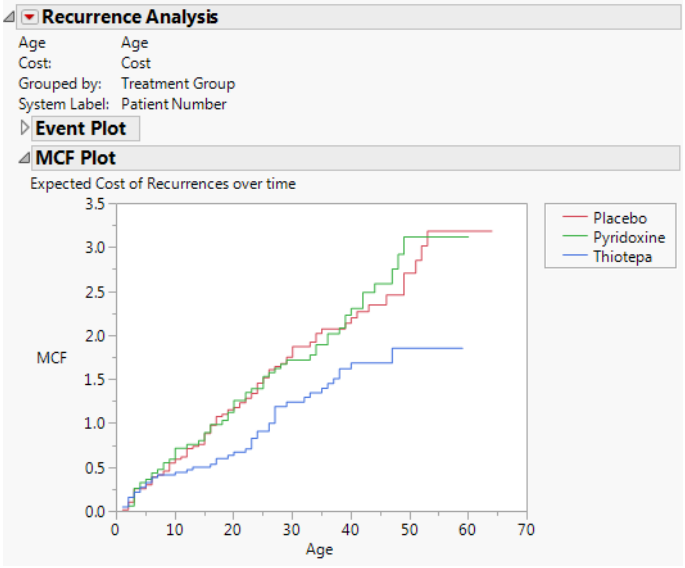
Model the Frequency or Cost of Recurrent Events over Time

The Recurrence Analysis platform analyzes event times, where the events can recur several times for each unit, item, or person. In an industrial setting, these events can occur when a unit breaks down, is repaired, and then put back into service after the repair. The units are followed until they are ultimately taken out of service.

In a medical setting, recurrence analysis can be used to analyze data from continuing treatments of a long-term disease, such as the recurrence of tumors in patients receiving treatment for bladder cancer.

The goal of the analysis is to obtain the mean cumulative function (MCF), which shows the total cost per unit as a function of time. Cost can be a count of the number of repairs, or it can be the actual cost of repair.

Figure 6.1 Recurrence Analysis Example



Contents

Overview of the Recurrence Analysis Platform 157
Example of the Recurrence Analysis Platform 157
Launch the Recurrence Analysis Platform 160
Recurrence Analysis Platform Options 161
Fit Model 162
Additional Examples of the Recurrence Analysis Platform 166
Bladder Cancer Recurrences Example 167
Diesel Ship Engines Example 170

Overview of the Recurrence Analysis Platform

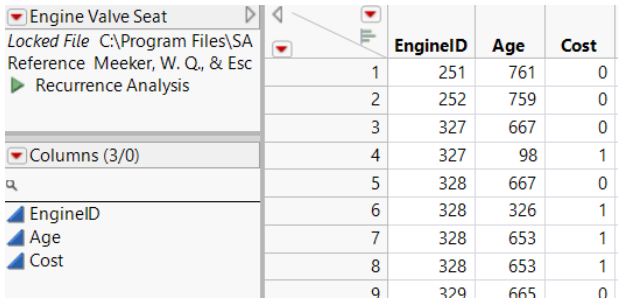
Recurrent event data involves the cumulative frequency or cost of repairs as units age. In JMP, the Recurrence Analysis platform analyzes recurrent events data.

The data for recurrence analysis have one row for each observed event and a closing row with the last observed age of a unit. Any number of units or systems can be included. In addition, these units or systems can include any number of recurrences.

Example of the Recurrence Analysis Platform

A typical unit might be a system, such as a component of an engine or appliance. For example, consider the sample data table Engine Valve Seat.jmp, which records valve seat replacements in locomotive engines. See Meeker and Escobar (1998, p. 395) and Nelson (2003). A partial listing of this data is shown in Figure 6.2. The EngineID column identifies a specific locomotive unit. Age is time in days from beginning of service to replacement of the engine valve seat. Note that an engine can have multiple rows with its age at each replacement and its cost, corresponding to multiple repairs. Here, Cost=0 indicates the last observed age of a locomotive.

Figure 6.2 Partial Engine Valve Seat Data Table

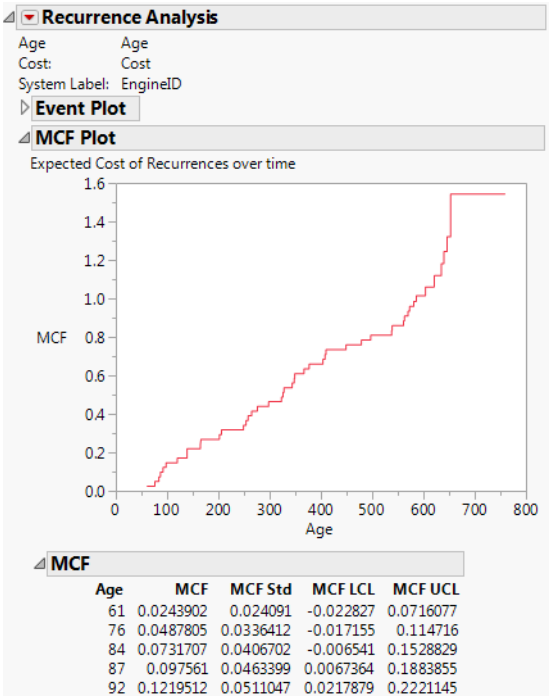


	EngineID	Age	Cost
1	251	761	0
2	252	759	0
3	327	667	0
4	327	98	1
5	328	667	0
6	328	326	1
7	328	653	1
8	328	653	1
9	329	665	0

Complete the launch window as shown in Figure 6.5.

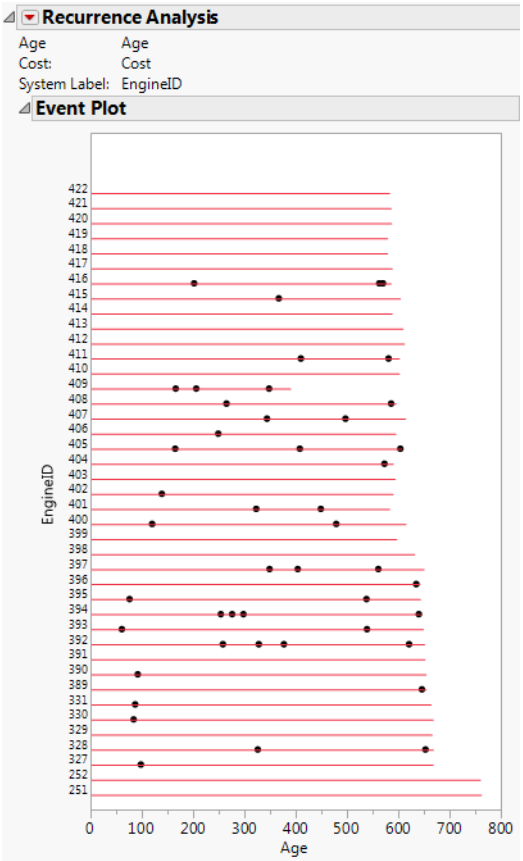
When you click **OK**, the Recurrence platform shows the reports in Figure 6.3 and Figure 6.4. The MCF plot shows the sample mean cumulative function. For each age, this is the nonparametric estimate of the mean cumulative cost or number of events per unit. This function goes up as the units get older and total costs grow. The plot in Figure 6.3 shows that about 580 days is the age that averages one repair event.

Figure 6.3 MCF Plot and Partial Table for Recurrence Analysis



The event plot in Figure 6.4 shows a time line for each unit. There are markers at each time of repair, and each line extends to that unit's last observed age. For example, unit 409 was last observed at 389 days and had three valve replacements.

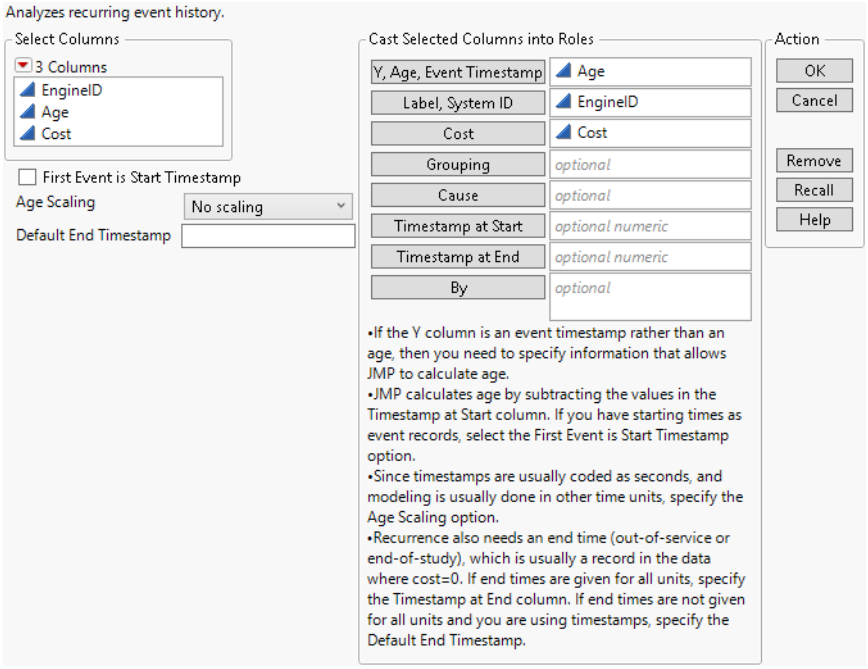
Figure 6.4 Event Plot for Valve Seat Replacements



Launch the Recurrence Analysis Platform

Launch the Recurrence Analysis platform by selecting **Analyze > Reliability and Survival > Recurrence Analysis**.

Figure 6.5 Recurrence Analysis Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Age, Event Timestamp Specifies either the unit's age at the time of an event or the timestamp of the event. If the Y column is an event timestamp, then you must specify the start and the end timestamp so that JMP can calculate age.

Label, System ID Identifies the unit for each event and censoring age.

Cost Identifies a column that must contain one of the following values:

- A 1, indicating that an event has occurred (a unit failed or was repaired, replaced, or adjusted). When indicators (1s) are specified, the MCF is the mean cumulative count of events per unit as a function of age.
- A cost for the event (the cost of the repair, replacement, or adjustment). When costs are specified, the MCF is a mean cumulative cost per unit as a function of age and the markers in the Event Plot are sized by the cost values.

– A zero, indicating that the unit went out-of-service, or is no longer being studied. All units (each System ID) must have one row with a zero for this column, with the **Y, Age, Event Timestamp** column containing the final observed age. If each unit does not have exactly one last observed age in the table (where the Cost column cell is zero), then an error message appears.

Note: Cost indicators for **Recurrence Analysis** are the reverse of censor indicators seen in **Life Distribution** or **Survival Analysis**. For the cost variable, the value of 1 indicates an event, such as repair; the value of 0 indicates that the unit is no longer in service. For the censor variable, the value of 1 indicates censored values, and the value of 0 indicates the event or failure of the unit (non-censored value).

Grouping Produces separate MCF estimates for the different groups that are identified by this column.

Cause Specifies multiple failure modes.

Timestamp at Start Specifies the column with the origin timestamp. If you have starting times as event records, select the **First Event is Start Timestamp** option instead. JMP calculates age by subtracting the values in this column.

Timestamp at End Specifies the column with the end-of-service timestamp. If end times are given for all units, specify that column here. If end times are not given for all units, specify the **Default End Timestamp** option instead. But if you have a record in which Cost is equal to zero, JMP uses that record as the end timestamp and you do not need to specify this role.

Age Scaling Specifies the time units for modeling. For example, if your timestamps are coded in seconds, you can change them to hours.

Recurrence Analysis Platform Options

The Recurrence Analysis red triangle menu contains the following options:

MCF Plot Shows or hides the mean cumulative function (MCF) plot.

MCF Confid Limits Shows or hides lines corresponding to the approximate 95% confidence limits of the mean cumulative function (MCF).

Event Plot Shows or hides the Event Plot. If a Cost column is specified in the launch window, the markers in the Event Plot are sized by the values in the Cost column.

Calendar Event Plot (Available only when the events are designated by a timestamp rather than an age.) Shows or hides the Calendar Event Plot, which shows events with calendar

date on the horizontal axis. This plot is next to the Event Plot and the units in each plot are aligned vertically.

Plot Interarrival by Age Shows or hides the Interarrival by Age plot, which plots the time between successive events on the vertical axis and the event times on the horizontal axis. You can use this plot to determine whether there are changes in the time between events in your data. For a recurrence analysis, the interarrival times should be independent and identically distributed over time. For more information about interarrival plots, see Tobias and Trindade (2012, p. 420).

Plot MCF Differences (Available only when you specify a grouping variable.) Shows or hides a plot of the difference of MCFs, including a 95% confidence interval for that difference. The MCFs are significantly different where the confidence interval lines do not cross the zero line. This option is available only when you specify a grouping variable.

MCF Plot Each Group (Available only when you specify a grouping variable.) Shows or hides a report that contains a mean cumulative function (MCF) plot for each level of the grouping variable.

This option can be used to get an MCF Plot for each unit if the **Label, System ID** variable is also specified as the **Grouping** variable.

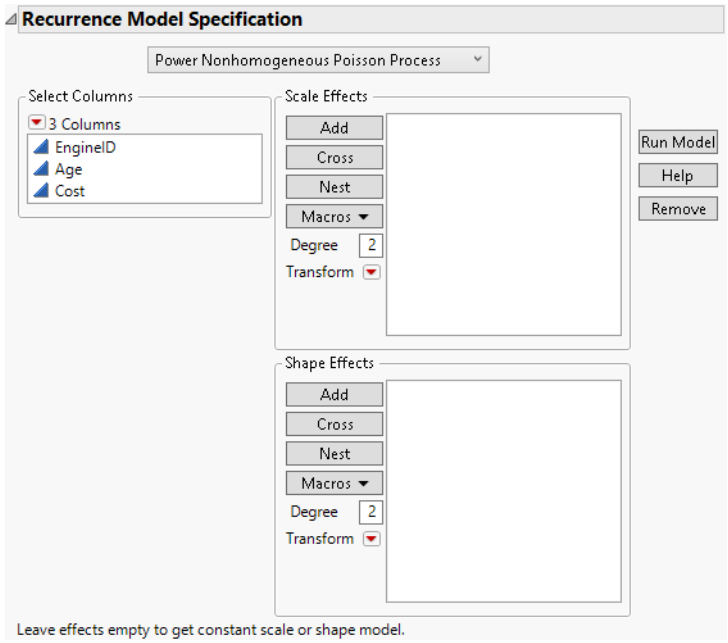
Fit Model Enables you to fit models for the Recurrence Intensity and Cumulative functions. See "Fit Model".

Fit Model

The Fit Model option is used to fit models for the Recurrence Intensity and Cumulative functions. There are four models available for describing the intensity and cumulative functions. You can fit the models with constant parameters, or with parameters that are functions of effects.

Select Fit Model from the Recurrence Analysis red triangle menu to produce the Recurrence Model Specification window shown in Figure 6.6.

Figure 6.6 Recurrence Model Specification



You can select one of four models, with the following Intensity and Cumulative functions:

Power Nonhomogeneous Poisson Process

$$I(t) = \left(\frac{\beta}{\theta}\right)\left(\frac{t}{\theta}\right)^{\beta - 1}$$

$$C(t) = \left(\frac{t}{\theta}\right)^{\beta}$$

Proportional Intensity Poisson Process

$$I(t) = \delta t^{\delta - 1} e^{\gamma}$$

$$C(t) = t^{\delta} e^{\gamma}$$

Loglinear Nonhomogeneous Poisson Process

$$I(t) = e^{\gamma + \delta t}$$

$$C(t) = \frac{I(t) - I(0)}{\delta} = \frac{e^{\gamma + \delta t} - e^{\gamma}}{\delta}$$

Homogeneous Poisson Process

$$I(t) = e^{\gamma}$$

$$C(t) = te^{\gamma}$$

where t is the age of the product.

Table 6.1 defines each model parameter as a scale parameter or a shape parameter.

Table 6.1 Scale and Shape Parameters

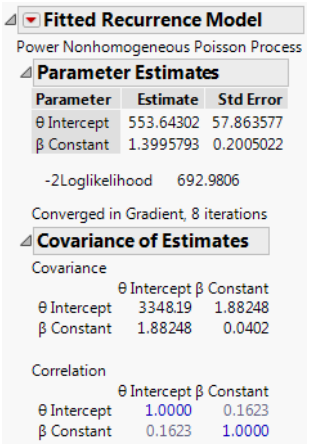
Model	Scale Parameter	Shape Parameter
Power NHPP	θ	β
Proportional Intensity PP	γ	δ
Loglinear NHPP	γ	δ
Homogeneous PP	γ	none

Note the following:

- For the Recurrence Model Specification window (Figure 6.6), if you include Scale Effects or Shape Effects, the scale and shape parameters in Table 6.1 are modeled as functions of the effects. To fit the models with constant scale and shape parameters, do not include any Scale Effects or Shape Effects.
- The Homogeneous Poisson Process is a special case compared to the other models. The Power NHPP and the Proportional Intensity Poisson Process are equivalent for one-term models, but the Proportional Intensity model seems to fit more reliably for complex models.

Click **Run Model** to fit the model and see the model report.

Figure 6.7 Model Report



The Fitted Recurrence Model red triangle menu contains the following options:

Profiler Shows or hides the Profiler showing the Intensity and Cumulative functions.

Effect Marginals Evaluates the parameter functions for each level of the categorical effect, holding other effects at neutral values. This helps you see how different the parameter functions are between groups. This is available only when you specify categorical effects.

Test Homogeneity Tests if the process is homogeneous. This option is not available for the Homogeneous Poisson Process model.

Effect Likelihood Ratio Test Produces a test for each effect in the model. This option is available only if there are effects in the model.

Specific Intensity and Cumulative Computes the intensity and cumulative values associated with particular time and effect values. The confidence intervals are profile likelihood intervals.

Specific Time for Cumulative Computes the time associated with a particular number of recurrences and effect values.

Save Intensity Formula Saves the Intensity formula to the data table.

Save Cumulative Formula Saves the Cumulative formula to the data table.

Publish Intensity Formula Creates the Intensity formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Publish Cumulative Formula Creates the Cumulative formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Simulate from Model Enables you to simulate new data from the estimated recurrence model. This option creates a new data table of simulated observations that are based on options that are specified in the Simulate from Model window. See "Simulate from Model".

Remove Fit Removes the model report.

Simulate from Model

When you select the Simulate from Model option from the Fitted Recurrence Model red triangle menu, the Simulate from Model window appears. This window contains the following specifications for the simulation:

Maximum Number of Events Specifies the maximum number of events to be simulated in each level of the System ID.

Maximum Age Specifies the maximum time for a simulated event.

Number of Units Specifies the number of units in each level of the System ID.

> **Note:** If the model contains terms other than the Intercept and Constant terms, the simulated data table contains this number of units for all level combinations of the regression terms. If the regression term is continuous, it is divided into 5 levels.

After you click OK, a new data table that contains the results appears. This table contains a script that enables you to analyze the simulated observations in the Recurrence platform.

Additional Examples of the Recurrence Analysis Platform

- "Bladder Cancer Recurrences Example"
- "Diesel Ship Engines Example"

Bladder Cancer Recurrences Example

The sample data file Bladder Cancer.jmp contains data on cancerous bladder tumor recurrences from the Veteran's Administration Co-operative Urological Research Group. See Andrews and Herzberg (1985, table 45). All patients presented with superficial bladder tumors, which were removed upon entering the trial. Each patient was then assigned to one of three treatment groups: placebo pills, pyridoxine (vitamin B6) pills, or periodic chemotherapy with Thiotepa. The following analysis of tumor recurrence explores the progression of the disease, and whether there is a difference among the three treatments.

Launch the platform with the options shown in Figure 6.8.

Figure 6.8 Bladder Cancer Launch Window

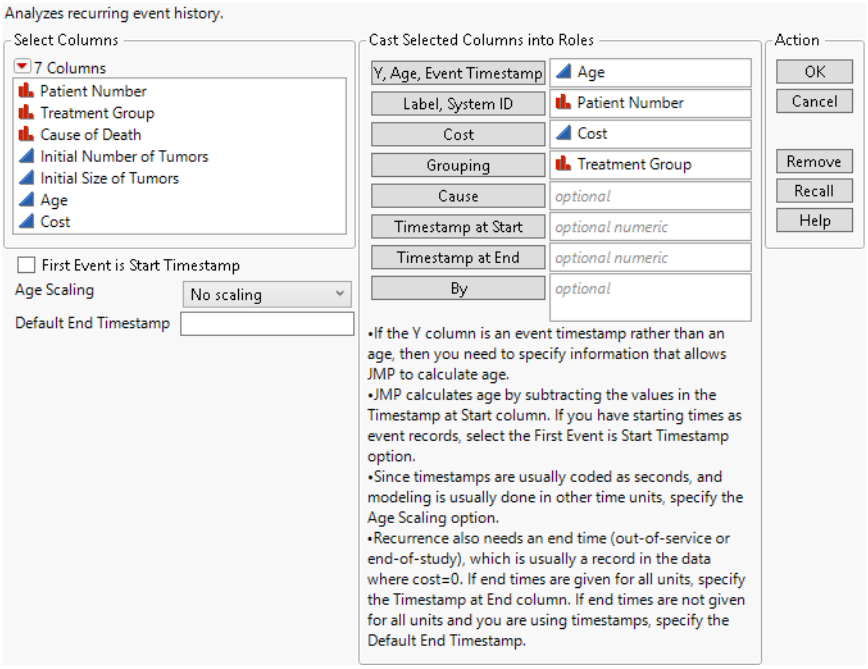
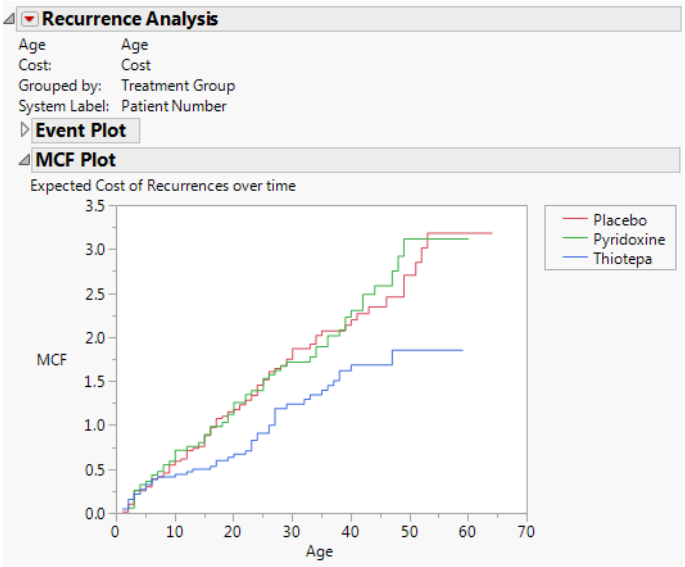


Figure 6.9 shows the MCF plots for the three treatments.

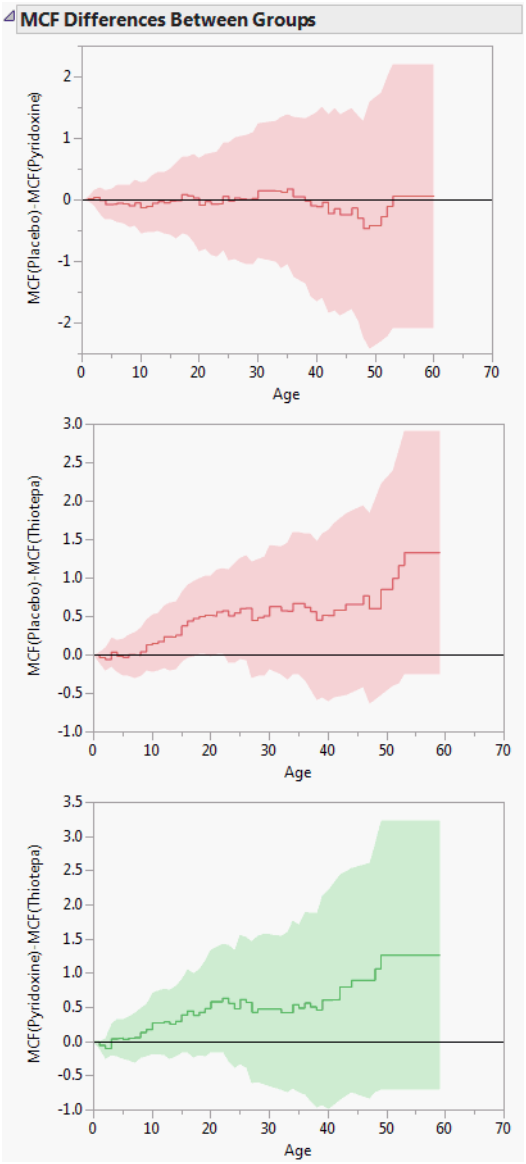
Figure 6.9 Bladder Cancer MCF Plot



Note that all three of the MCF curves are essentially straight lines. The slopes (rates of recurrence) are therefore constant over time, implying that patients do not seem to get better or worse as the disease progresses.

To examine if there are differences among the treatments, click the Recurrence Analysis red triangle and select **Plot MCF Differences**.

Figure 6.10 MCF Differences



To determine whether there is a statistically significant difference between treatments, examine the confidence limits on the differences plot. If the limits do not include zero, the treatments are convincingly different. The graphs in Figure 6.10 show there is no significant difference among the treatments.

Diesel Ship Engines Example

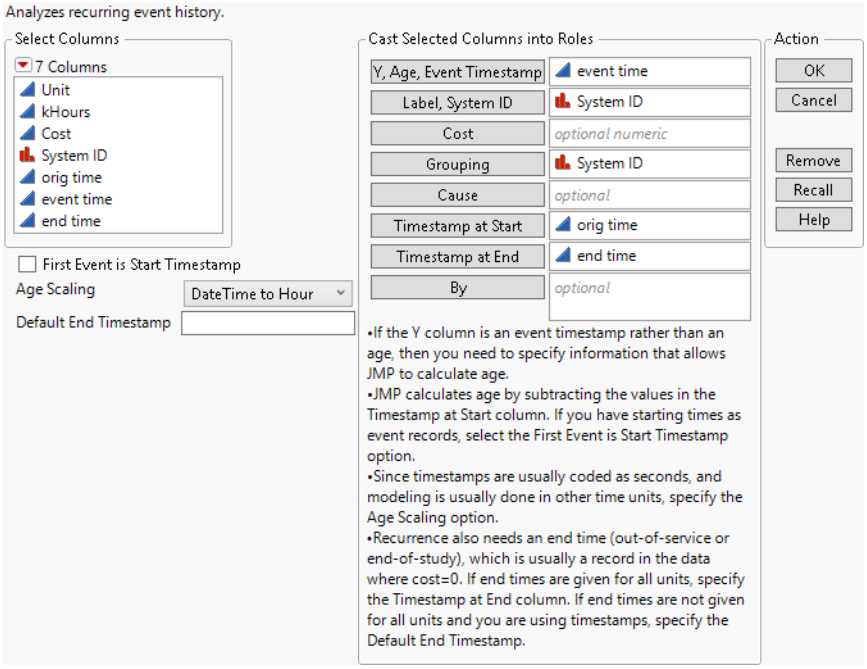
The sample data table Diesel Ship Engines.jmp contains data on engine repair times for two ships (Grampus4 and Halfbeak4) that have been in service for an extended period of time. See Meeker and Escobar (1998). You want to examine the progression of repairs and gain a sense of how often repairs might need to be done in the future. These observations can help you decide when an engine should be taken out of service.

1. Select **Help > Sample Data Library** and open Reliability/Diesel Ship Engines.jmp.
2. Ensure that rows 57 and 129 are set as Excluded.

> **Note:** If they are not set to Excluded, select rows 57 and 129 and select **Rows > Exclude/Unexclude**.

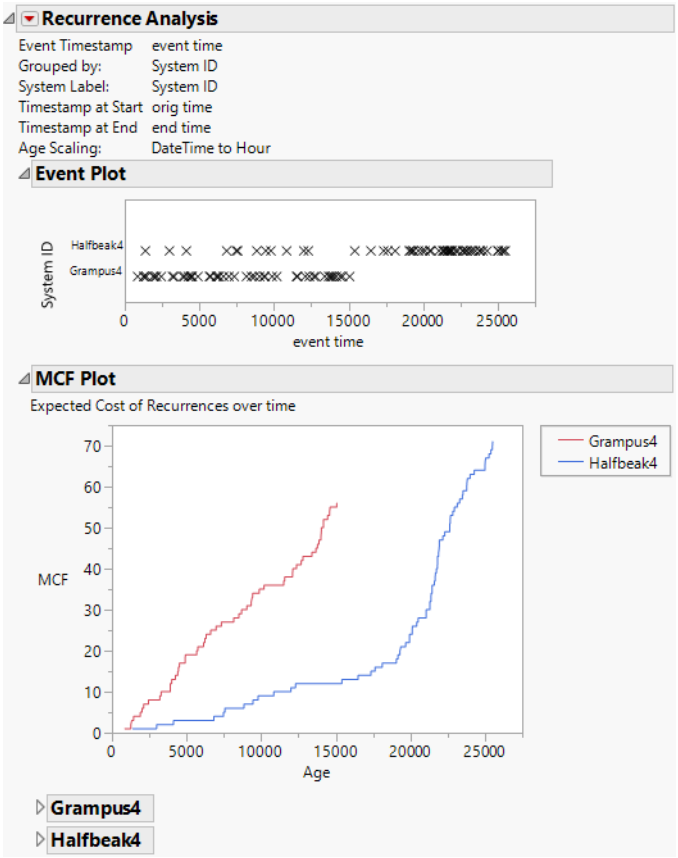
3. Select **Analyze > Reliability and Survival > Recurrence Analysis**.
4. Complete the launch window as shown in Figure 6.11.

Figure 6.11 Diesel Ship Engines Launch Window



5. Click **OK**.

Figure 6.12 Diesel Ship Engines Report

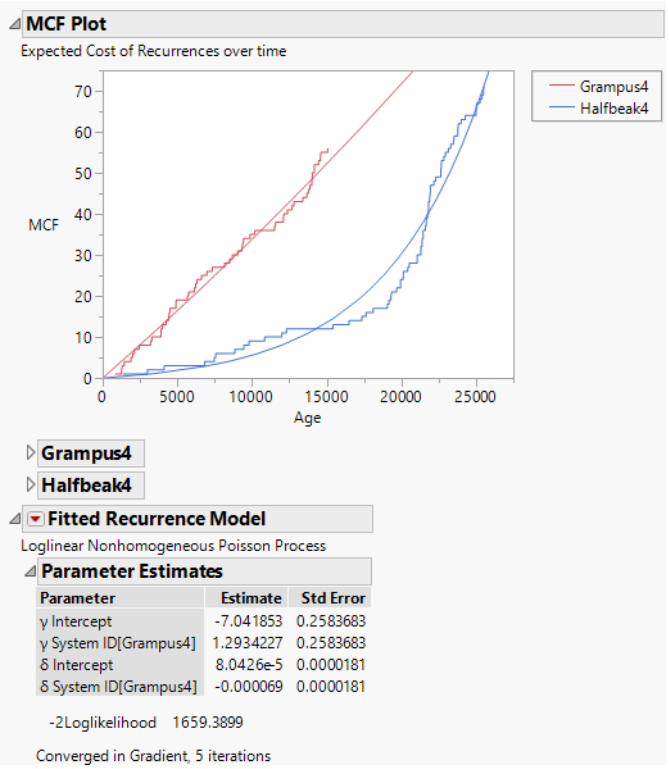


Looking at the Event Plot, you can see that repairs for the Grampus4 engine have been relatively consistent. Repairs for the Halfbeak4 engine have been more sporadic, and there appears to be an abrupt increase in repairs somewhere around the 19,000 hour mark. This increase is even more obvious in the MCF Plot.

Continue your analysis by fitting a parametric model to help predict future performance.

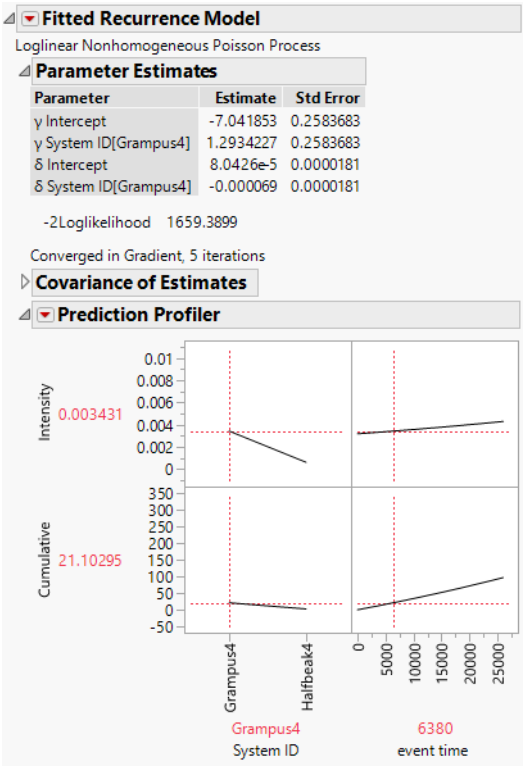
6. Click the Recurrence Analysis red triangle and select **Fit Model**.
7. In the Recurrence Model Specification, select the **Loglinear Nonhomogeneous Poisson Process**.
8. Add the System ID column as both a Scale Effect and a Shape Effect.
9. Click **Run Model**.

Figure 6.13 Diesel Ship Engines Fitted Model



10. Click the Fitted Recurrence Model red triangle and select **Profiler**.

Figure 6.14 Diesel Ship Profiler



Compare the number of future repairs for the Grampus4 engine to the Halfbeak4 engine. Change the event time value to see the effect on the cumulative number of future repairs.

- To see how many repairs will be needed after 30,000 hours of service, type 30,000 for the event time. The Grampus4 engine will require about 114 repairs. To see the values for Halfbeak4, click and drag the dotted line from Grampus4 to Halfbeak4. The Halfbeak4 engine will require about 140 repairs.
- To see how many repairs will be needed after 80,000 hours of service, type 80,000 for the event time. The Halfbeak4 engine will require about 248,169 repairs. Click and drag the dotted line from Halfbeak4 to Grampus4. The Grampus4 engine will require about 418 repairs.

You can conclude that in the future, the Halfbeak4 engine will require many more repairs than the Grampus4 engine.

Chapter 7

Degradation

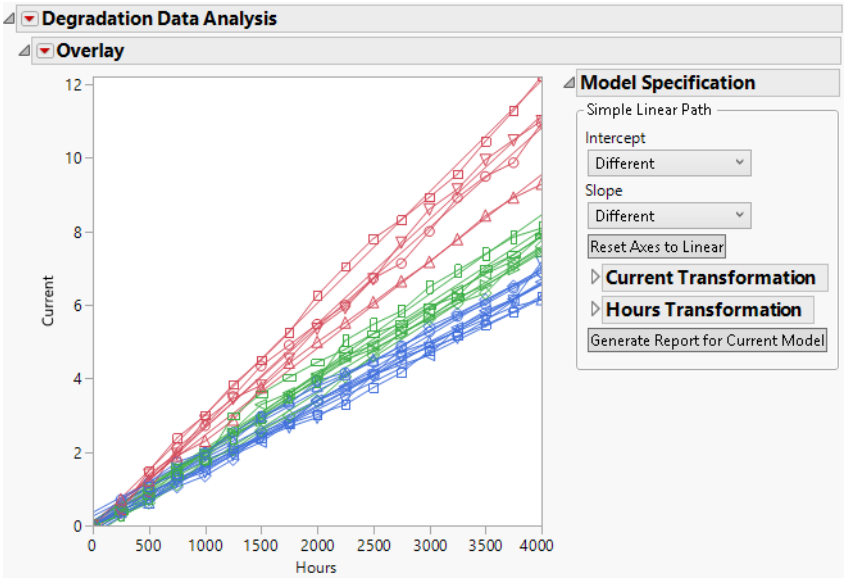
Model Product Deterioration over Time

Using the Degradation platform, you can analyze degradation data to predict pseudo failure times. These pseudo failure times can then be analyzed by other reliability platforms to estimate failure distributions.

Both linear and nonlinear degradation paths can be modeled. You can specify an accelerating factor to analyze accelerated degradation data.

You can also perform stability analysis, which is useful when setting pharmaceutical product expiration dates.

Figure 7.1 Degradation Analysis Example



Contents

Overview of the Degradation Platform 177
Example of the Degradation Platform 177
Launch the Degradation Platform 179
The Degradation Reports 181
Model Specification 184
Simple Linear Path 184
Nonlinear Path 186
Inverse Prediction 194
Prediction Graph 196
Degradation Platform Options 197
Model Reports 199
Model Lists 199
Reports 200
Destructive Degradation 200
Stability Analysis 205

Overview of the Degradation Platform

In reliability analyses, the primary objective is to model the failure times of the product under study. In many situations, these failures occur because the product degrades (weakens) over time. But, sometimes failures do not occur. In these situations, modeling the product degradation over time is helpful in making predictions about failure times. When an accelerating factor is included in the data, you can fit an accelerated degradation model.

The Degradation platform can model data that follows linear or nonlinear degradation paths. If a path is nonlinear, transformations are available to linearize the path. If linearization is not possible, you can specify a nonlinear model.

You can also use the Degradation platform to perform stability analysis. Three types of linear models are fit, and an expiration date is estimated. Stability analysis is used in setting pharmaceutical product expiration dates.

Example of the Degradation Platform

This example uses the GaAs Laser.jmp data table from Meeker and Escobar (1998), which contains measurements of the percent increase in operating current taken on several gallium arsenide lasers. When the percent increase reaches 10%, the laser is considered to have failed.

1. Select **Help > Sample Data Library** and open Reliability/GaAs Laser.jmp.
2. Select **Analyze > Reliability and Survival > Degradation**.
3. Select Current and click **Y, Response**.
4. Select Hours and click **Time**.
5. Select Unit and click **Label, System ID**.
6. Type 10 in the text box for **Upper Spec Limit**.
7. Click **OK**.

Figure 7.2 Initial Degradation Report

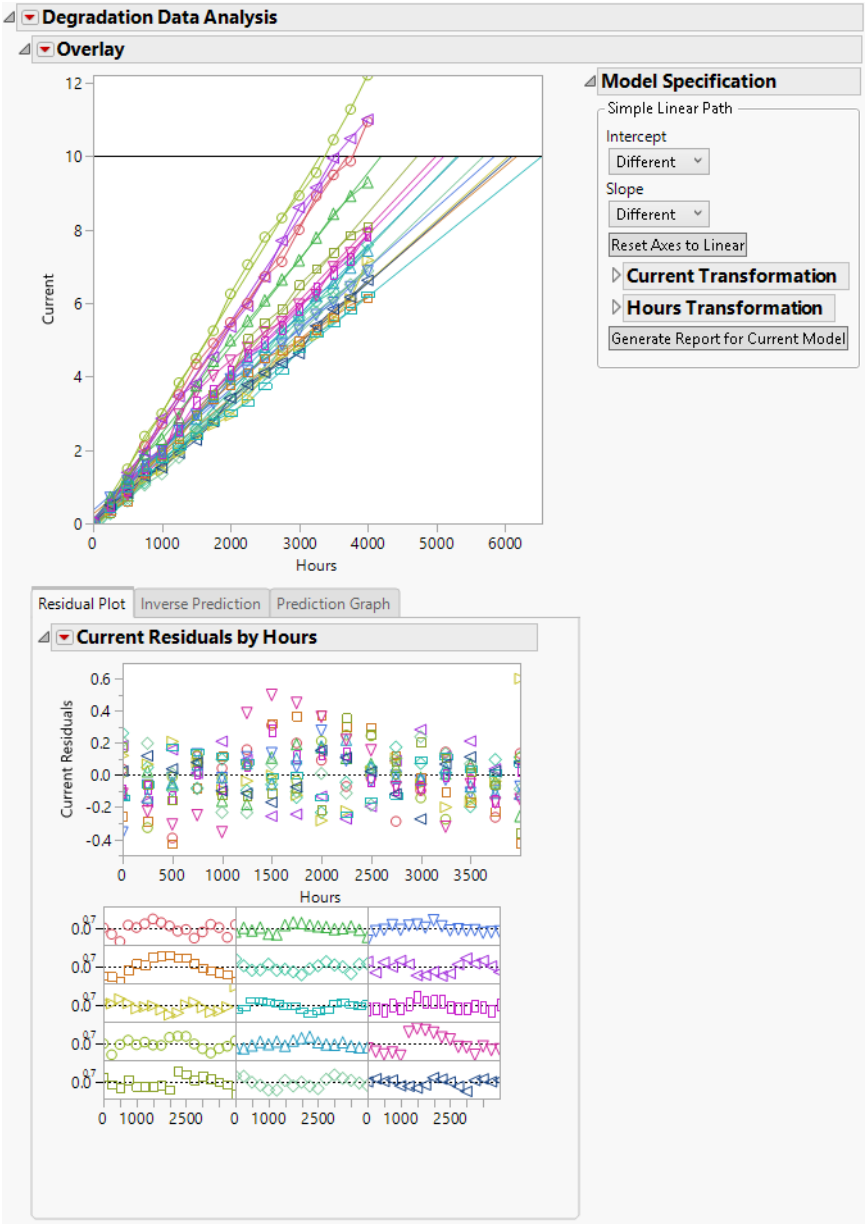


Figure 7.2 shows the initial Degradation report. The Overlay plot shows the measurements of Current versus Time for each unit in the data. The horizontal line at Current = 10 corresponds to the upper specification limit at 10%. Units with values above this limit are considered to have failed. Three of the fifteen units have reached that point by the end of the study period. The Inverse Prediction outline shows the predicted Hours value for which each unit fails, based on your specified model.

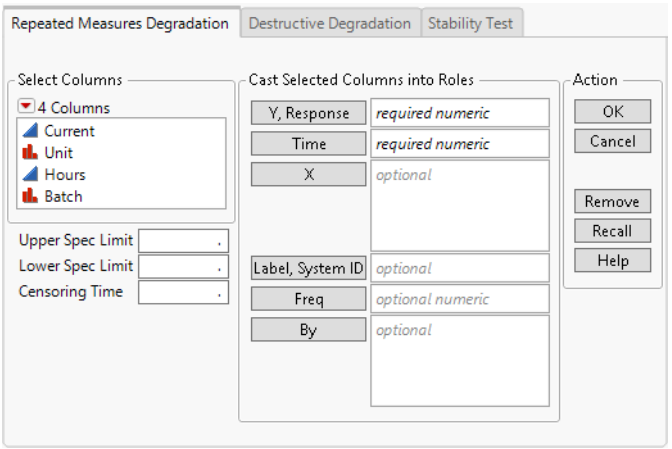
The default model fits a separate slope and intercept for each unit, using linear transformations. You can fit other models using the Model Specification outline.

The Residual Plot tab in Figure 7.2 shows residuals based on your specified model. The top plot shows residuals for all units plotted against Hours and overlaid on one plot. The bottom plot shows individual plots of the residuals for each unit in a rectangular array.

Launch the Degradation Platform

Launch the Degradation platform by selecting **Analyze > Reliability and Survival > Degradation**. Figure 7.3 shows the Degradation launch window using the GaAs Laser.jmp data table (located in the Reliability folder). For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Figure 7.3 The Degradation Launch Window



Analysis Types

The launch window is split into three tabs, representing three different types of analyses:

Repeated Measures Degradation Performs linear or nonlinear degradation analysis. This option allows only one Y, Response variable. It does not allow censoring.

Destructive Degradation Choose this type of analysis if units are destroyed during the measurement process. This option allows censoring. See "Destructive Degradation".

Note: The Destructive Degradation platform provides a flexible collection of predefined models for destructive testing. See "Destructive Degradation".

Stability Test Performs a stability analysis for setting pharmaceutical product expiration dates. This option allows only one Y, Response variable. See "Stability Analysis".

Launch Window Options

The launch window contains the following options:

Y, Response Assign the column with degradation measurements.

Time Assign the column containing the time values.

X (Available only in the Repeated Measures Degradation and Destructive Degradation tabs.) Assign an explanatory variable. Use this role to specify the accelerating factor in an accelerated degradation model.

Label, System ID (Available only in the Repeated Measures Degradation and Stability Test tabs.) Assign the column that designates the unit IDs.

Freq Assign a column giving a frequency for each row.

Censor (Available only in the Destructive Degradation tab.) Assign a column that designates if a unit is censored.

By Assign a variable to produce an analysis for each level of the variable.

Censor Code (Available only in the Destructive Degradation tab.) Specify the value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

Upper Spec Limit Assign an upper specification limit. (Optional except for Stability Test tab.)

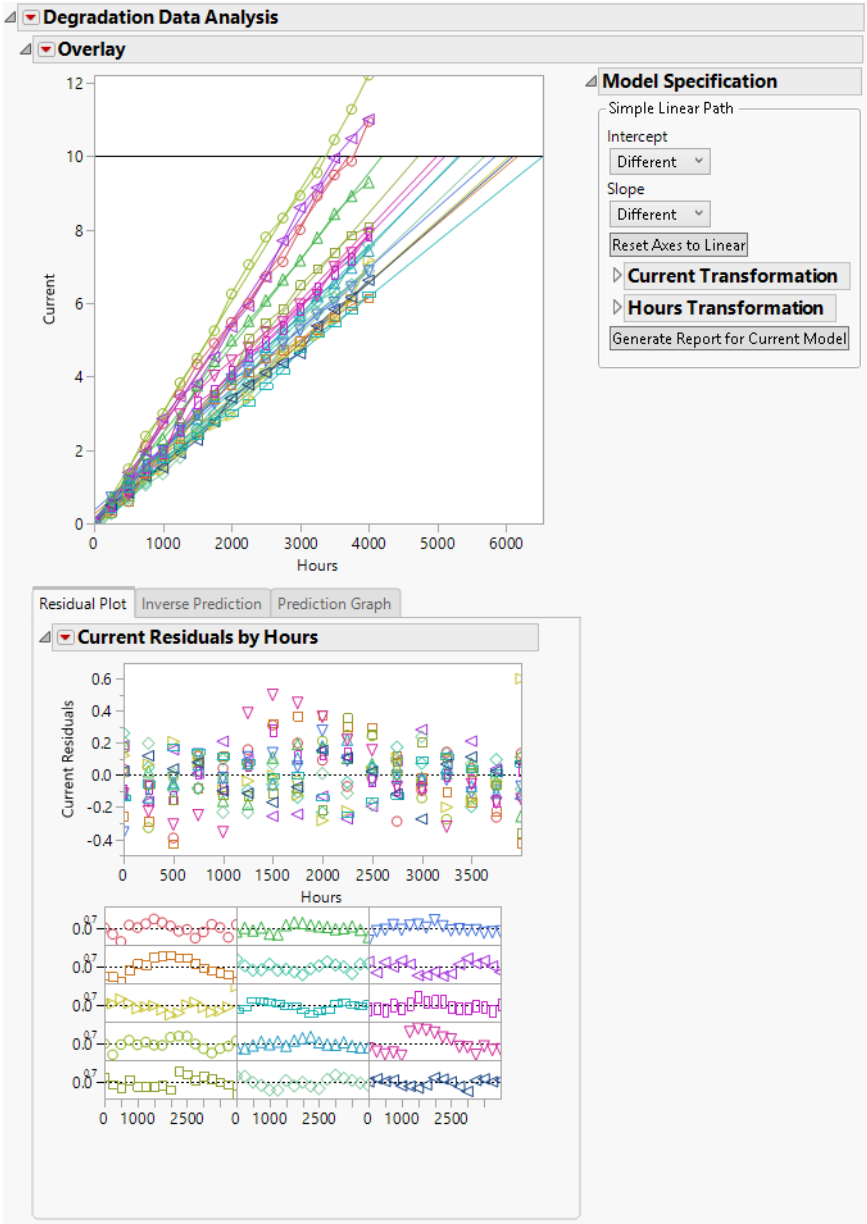
Lower Spec Limit Assign a lower specification limit. (Optional except for Stability Test tab.)

Censoring Time (Available only in the Repeated Measures Degradation and Destructive Degradation tabs.) Assign a Time value that represents censoring of pseudo failures when you use Inverse Prediction. See "Inverse Prediction".

The Degradation Reports

The Repeated Measures Degradation and Destructive Degradation methods show a report that fits a default model. As shown in the Model Specification outline in Figure 7.4, this model fits each unit with its own intercept and slope, using a linear transformation of the response and time columns. Separate intercepts and slopes are fit for each value of the Label, System ID variable, or, if only an X variable is specified, separate intercepts and slopes are fit for each level of the X variable. The Stability Test method fits three models.

Figure 7.4 Initial Repeated Measures Degradation Report with Transformation Outlines Open



To reproduce this example, see "Example of the Degradation Platform".

The reports for Repeated Measures Degradation, Destructive Degradation, and Stability Test include the following:

Overlay

An Overlay plot of the Y, Response variable versus the Time variable. In this example, the plot is of Current versus Hours. The Save Estimates option in the Overlay plot red triangle menu creates a new data table containing the estimated slopes and intercepts for all units.

Model Specification

Specify your model and generate a report for that model. See "Model Specification". (Available only for the Repeated Measures Degradation and Destructive Degradation methods.)

Stability Tests Outline

Compare models and estimate expiration dates. See "Stability Analysis". (Available only for the Stability Test method.)

Reports

Shows analysis results for three different models and the best model. See "Stability Analysis". (Available for the Stability Test method by default. Also available for the Repeated Measures Degradation and Destructive Degradation methods after you click Generate Report for Current Model.)

Tabbed Reports

Residual Plot Shows a single residual plot with all the units overlaid and separate residual plots for each unit arranged in a rectangular grid. The red triangle menu contains the following options:

Save Residuals Saves the residuals of the current model to a new data table.

Jittering Adds random noise to the points in the time direction. This is useful for visualizing the data if there are a lot of points clustered together.

Separate Groups Adds space between the groups to visually separate the groups. This option appears only when an X variable is specified on the platform launch window.

Jittering Scale Changes the magnitude of the jittering and group separation. This option appears only if Jittering is selected.

Inverse Prediction Enables you to predict the time at which the Y variable reaches a specified value. See "Inverse Prediction".

Prediction Graph Enables you to predict the Y variable for a specified Time value. See "Prediction Graph".

Model Specification

You can use the Model Specification outline to specify the model that you want to fit to the degradation data. There are two types of Model Specifications:

Simple Linear Path Used to model linear degradation paths, or nonlinear paths that can be transformed to linear. See "Simple Linear Path".

Nonlinear Path Used to model nonlinear degradation paths, especially those that cannot be transformed to linear. See "Nonlinear Path".

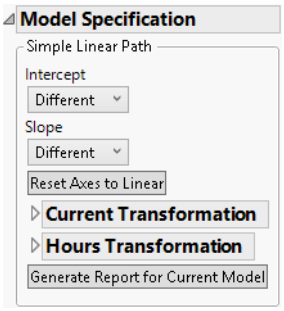
To change between the two specifications, use the Degradation Path Style submenu from the Degradation Data Analysis red triangle menu.

Simple Linear Path

To model linear degradation paths, select **Degradation Path Style > Simple Linear Path** from the Degradation Data Analysis red triangle menu.

Use the Simple Linear Path Model specification to specify the form of the linear model that you want to fit to the degradation path. You can model linear paths, or nonlinear paths that can be transformed to linear.

Figure 7.5 Simple Linear Path Model Specification



The Simple Linear Path model specification contains the following options:

Intercept Specifies the form of the intercept in the model:

Different Fits a different intercept for each level of the ID variable.

Common in Group Fits the same intercept for each level of the ID variable in the same level of the X variable, and fits different intercepts between levels.

Common Fits the same intercept for all levels of the ID variable.

Zero Restricts the intercept to be zero for all levels of the ID variable.

Slope Specifies the form of the slope in the model:

Different Fits a different slope for each level of the ID variable.

Common in Group Fits the same slope for each level of the ID variable in the same level of the X variable, and fits different slopes between levels.

Common Fits the same slope for all levels of the ID variable.

Reset Axes to Linear Resets the Overlay plot axes to their initial settings.

<Y, Response> Transformation If a transformation on the Y variable can linearize the degradation path, select the transformation (Linear, ln(x), exp(x), x^2, sqrt(x) or Custom) here. For more information about the Custom option, see "Custom Transformations".

<Time> Transformation If a transformation for the Time variable can linearize the degradation path, select the transformation (Linear, ln(x), x^2, sqrt(x) or Custom) here. For more information about the Custom option, see "Custom Transformations".

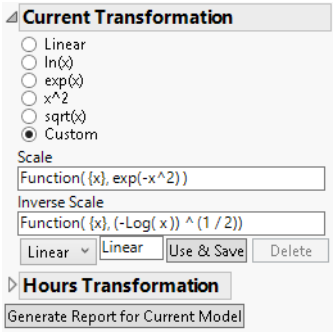
Generate Report for Current Model Creates a report for the current model settings. This includes a Model Summary report, and Estimates report giving the parameter estimates. See "Model Reports".

Custom Transformations

If you need to perform a transformation that is not given, use the Custom option. For example, to transform the response variable using $\exp(-x^2)$, enter the transformation as shown in the Scale box in Figure 7.6. Also, enter the inverse transformation in the Inverse Scale box.

Note: JMP automatically attempts to solve for the inverse transformation. If it can solve for the inverse, it automatically enters it in the Inverse Scale box. If it cannot solve for the inverse, you must enter it manually.

Figure 7.6 Custom Transformation Options



Name the transformation using the text box. When finished, click the **Use & Save** button to apply the transformation. Select a transformation from the menu if you have created multiple custom transformations. Click the **Delete** button to delete a custom transformation.

Nonlinear Path

To model nonlinear degradation paths, select **Degradation Path Style > Nonlinear Path** from the Degradation Data Analysis red triangle menu. This is useful if a degradation path cannot be linearized using transformations, or if you have a custom nonlinear model that you want to fit to the data.

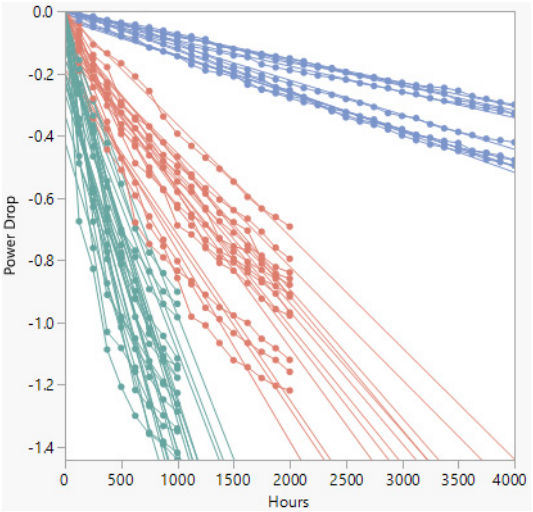
To facilitate explaining the Nonlinear Path Model Specification, open the Device B.jmp data table. The data consists of power decrease measurements taken on 34 units, across four levels of temperature. Follow these steps:

1. Select **Help > Sample Data Library** and open Reliability/Device B.jmp.
2. Select **Analyze > Reliability and Survival > Degradation**.
3. Select Power Drop and click **Y, Response**.
4. Select Hours and click **Time**.
5. Select Degrees C and click **X**.

   The temperature setting is the accelerating factor in the experiment.
6. Select Device and click **Label, System ID**.
7. Click **OK**.

The initial overlay plot of the data appears.

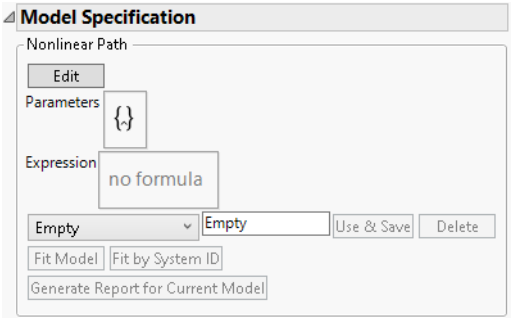
Figure 7.7 Device B Overlay Plot



The degradation paths appear linear for the first several hundred hours, but then start to curve. To fit a nonlinear model, select **Degradation Path Style > Nonlinear Path** from the Degradation Data Analysis red triangle menu to show the Nonlinear Path Model Specification outline (Figure 7.8).

Note: To view the Edit button displayed in Figure 7.8, you must select the interactive formula editor preference (**File > Preferences > Platforms > Degradation > Use Interactive Formula Editor**) before launching the Degradation platform.

Figure 7.8 Initial Nonlinear Model Specification Outline



The first step to create a model is to select one of the options on the menu initially labeled **Empty**:

- For more information about Reaction Rate models, see "Reaction Rate Models".
- For more information about Constant Rate models, see "Constant Rate Models".

- For more information about using a Prediction Column, see "Prediction Columns".

Reaction Rate Models

The Reaction Rate options are applicable when the degradation occurs from a single chemical reaction, and the reaction rate is a function of temperature only. Select **Reaction Rate** or **Reaction Rate Type 1** from the menu shown in Figure 7.8. Although similar to the Reaction Rate model, the Reaction Rate Type 1 model contains an offset term that changes the basic assumption concerning the response value's sign.

The Setup window prompts you to select the temperature scale, and the baseline temperature. The baseline temperature is used to generate initial estimates of parameter values. The baseline temperature should be representative of the temperatures used in the study.

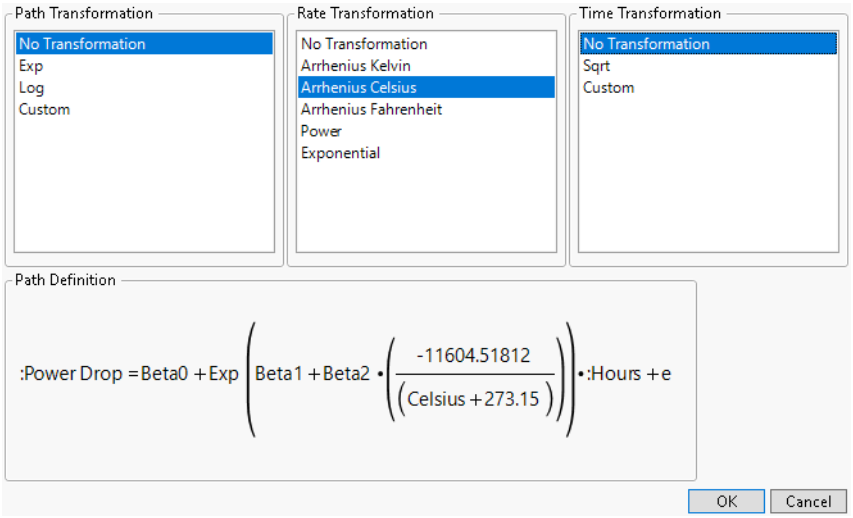
For this example, select **Reaction Rate** and then select Celsius as the Temperature Unit. Click **OK** to return to the report. For more information about all the features for Model Specification, see "Model Specification Details".

Constant Rate Models

The Constant Rate option is for modeling degradation paths that are linear with respect to time (or linear with respect to time after transforming the response or time). The reaction rate is a function of temperature only.

Select **Constant Rate** from the menu shown in Figure 7.8. The Constant Rate Model Settings window prompts you to enter transformations for the Path, Rate, and Time.

Figure 7.9 Constant Rate Transformation



Once a selection is made for each transformation, the associated formula appears in the lower left corner as shown in Figure 7.9.

After all selections are made, click **OK** to return to the report. For more information about all the features for Model Specification, see "Model Specification Details".

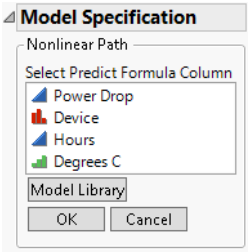
Prediction Columns

The Prediction Column option enables you to use a custom model that is stored as a formula in a data table column. The easiest approach is to create the formula column before launching the Degradation platform. You can also create the formula column from within the Degradation platform if you want to use one of the built-in models in the Nonlinear Model Library.

For more information about how to create a custom model and store it as a column formula, see "Fit a Custom Model" or *Predictive and Specialized Modeling*.

Select **Prediction Column** from the list that appears beneath the Expression area in Figure 7.8. The Model Specification outline changes to prompt you to select the column that contains the model.

Figure 7.10 Column Selection



At this point, do one of three things:

- If the model that you want to use already exists as a formula in a column of the data table, select the corresponding column here, and then click **OK**. You are returned to the Nonlinear Path Model Specification. For more information about all the features for that specification, see "Model Specification Details".
- If the model that you want to use does not already exist in the data table, you can click the **Model Library** button to use one of the built-in models. For more information about using the Model Library button, see "Model Library" or *Predictive and Specialized Modeling*. After the model is created, select **Redo > Redo Analysis** from the Degradation Data Analysis red triangle menu. Then, return to the column selection shown in Figure 7.10. Select the column that contains the model, and then click **OK**. You are returned to the Nonlinear Path Model Specification. For more information about all the features for that specification, see "Model Specification Details".

- If the model that you want to use is not in the data table, and you do not want to use one of the built-in models, then you are not ready to use this model specification. First, create the model, relaunch the Degradation platform, and then return to the column selection (Figure 7.10). Select the column containing the model, and then click **OK**. You are returned to the Nonlinear Path Model Specification. For more information about all the features for that specification, see "Model Specification Details".

Model Specification Details

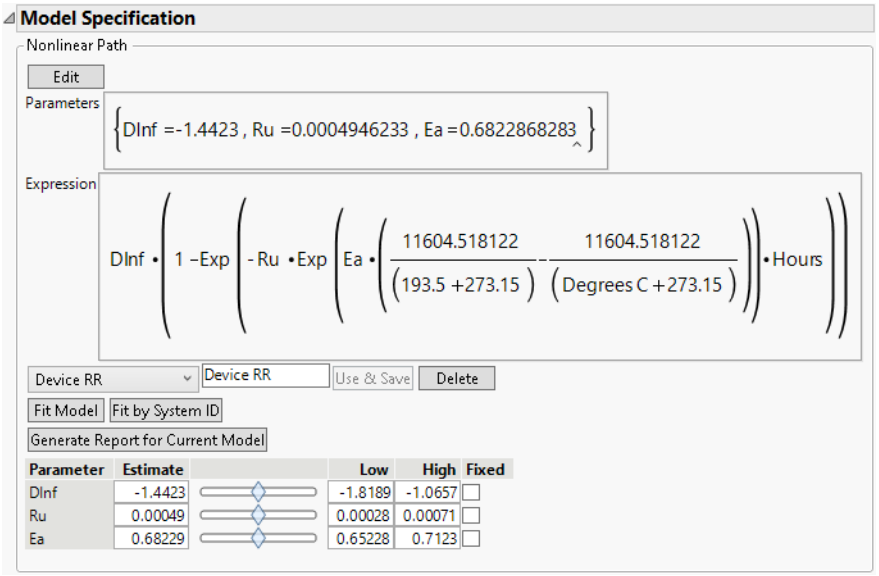
After you select one of the model types and supply the required information, you are returned to the Nonlinear Path Model Specification window.

Note: To view the Edit button displayed in Figure 7.11, you must select the interactive formula editor preference (**File > Preferences > Platforms > Degradation > Use Interactive Formula Editor**) before launching the Degradation platform.

A model is now shown in the script box that uses the `Parameter` statement. Initial values for the parameters are estimated from the data. For more information about creating models that use parameters, see "Fit a Custom Model" or *Predictive and Specialized Modeling*.

If desired, enter a name in the text box to name the model. For this example, use the name "Device RR". After that, click the **Use & Save** button to enter the model and activate the other buttons and features. Figure 7.11 shows the Model Specification window after clicking the Use & Save button.

Figure 7.11 Model Specification



- The **Fit Model** button is used to fit the model to the data.
- The **Fit by System ID** is used to fit the model to every level of **Label, System ID**.
- The **Delete** button is used to delete a model from the model menu.
- The **Generate Report for Current Model** button creates a report for the current model settings. See "Model Reports".

The initial parameter values are shown at the bottom, along with sliders for visualizing how changes in the parameters affect the model. The fitted lines are shown on the Overlay plot. Move the parameter sliders to see how changes affect the fitted lines.

Here are the parameters for the Reaction Rate model (Meeker and Escobar 1998):

- **Dinf** (D_∞) - asymptotic degradation level
- **Ru** (R_U) - reaction rate at use temperature ($temp_U$)
- **Ea** (Ea) - reaction-specific activation energy

The above parameters are calculated as follows:

$$D(t; \text{temp}) = D_\infty \text{ x } \{1\text{-exp}[-R_U \text{ x } AF(\text{temp}) \text{ x } t]\}$$

where R_U is the reaction rate at use temperature $temp_U$, R_U x AF(temp) is the reaction rate at a general temperature temp, and for temp > $temp_U$, AF(temp) > 1

and

$$AF(\text{temp}) = \frac{R(temp)}{R(temp_U)} = \exp\left[Ea\left(\frac{11604.5181215503}{(temp_U K)} - \frac{11604.5181215503}{(tempK)}\right)\right]$$

where $temp_U K$ and $tempK$ are temperatures expressed on the Kelvin scale.

To compute the optimal values for the parameters, click the **Fit Model** or **Fit by System ID** button.

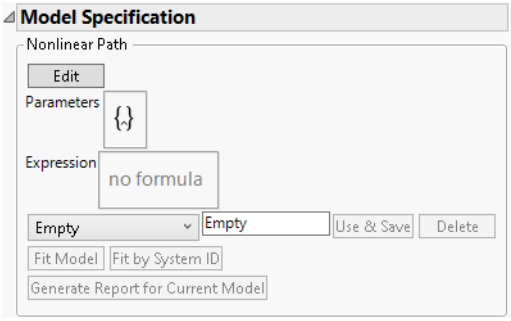
To fix a value for a parameter, check the box under **Fixed** for the parameter. When fixed, that parameter is held constant in the model fitting process.

Entering a Model with the Formula Editor

You can use the Formula Editor to enter a model. Click the **Edit** button to open the Formula Editor to enter parameters and the model. For more information about entering parameters and formulas in the Formula Editor, see *Using JMP*.

Note: To view the Edit button displayed in Figure 7.12, you must select the interactive formula editor preference (**File > Preferences > Platforms > Degradation > Use Interactive Formula Editor**) before launching the Degradation platform.

Figure 7.12 Alternate Model Specification Report



Fit a Custom Model

If you want to fit a custom model, you must first create a formula column with initial parameter estimates. This method requires a few more steps than fitting a built-in model, but it allows any nonlinear model to be fit. Also, you can provide a custom loss function, and specify several other options for the fitting process.

1. Open your data table.
2. Create a new column in the data table.
3. Open the Formula Editor for the new column.
4. Select **Parameters** from the list in the lower left corner.
5. Click **New Parameter**.
6. Enter the name of the parameter.
7. Enter the initial value of the parameter.

   Repeat steps 4 to 6 to create all the parameters in the model.
8. Build the model formula using the data table columns, parameters, and formula editor functions.
9. Click **OK**.

Parameters for Models with a Grouping Variable

In the formula editor, when you add a parameter, note the check box for **Expand Into Categories, selecting column**. This option is used to add several parameters (one for each level of a categorical variable for example) at once. When you select this option, a window appears that enables you to select a column. After selection, a new parameter appears in the Parameters list with the name D_*column*, where D is the name that you gave the parameter. When you use this parameter in the formula, a Match expression is inserted, containing a separate parameter for each level of the grouping variable.

Model Library

The Model Library can assist you in creating a formula column with parameters and initial values. Click **Model Library** under Model Specification to open the library. Select a model in the list to see its formula in the **Formula** box.

Click **Show Graph** to show a 2-D theoretical curve for one-parameter models and a 3-D surface plot for two-parameter models. No graph is available for models with more than two explanatory (*X*) variables. Use the slider bars to change the default starting values for the parameters. You can also click the values and enter new values directly.

The **Reset** button sets the starting values for the parameters back to their default values.

Click **Show Points** to overlay the actual data points to the plot. A window opens, asking you to assign columns into *X* and *Y* roles, and an optional Group role. The Group role allows for fitting the model to every level of a categorical variable. If you specify a Group role here, also specify the same column in the Label, System ID role in the platform launch window.

For most models, the starting values are constants. Showing points enables you to adjust the parameter values to see how well the model fits for different values of the parameters.

Click **Make Formula** to create a new column in the data table. This column has the formula as a function of the specified *X* variable and uses the parameter values specified in the graph window.

Note: If you click **Make Formula** before you click the **Show Graph** or **Show Points** buttons, you are asked to provide the *X* and *Y* roles, and an optional Group role. After that, you are brought back to the plot so that you have the option to adjust the starting values for the parameters. Once the starting values for the parameters are satisfactory, click **Make Formula** again to create the new column.

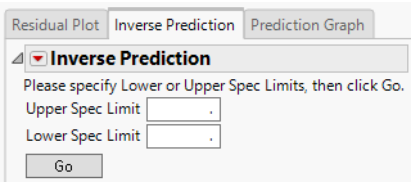
Once the formula is created in the data table, click the Degradation Data Analysis red triangle and select **Redo > Redo Analysis**. Then, return to the column selection shown in Figure 7.10. Select the column that contains the model, and then click **OK**. You are returned to the Nonlinear Path Model Specification. For more information about all the features for that specification, see "Model Specification Details".

Note: You can customize the models included in the Nonlinear Model Library by modifying the built-in script named NonlinLib.jsl. This script is located in the Resources/Builtins folder in the folder that contains JMP (Windows) or in the Application Package (macOS).

Inverse Prediction

Use the Inverse Prediction tab to predict the time at which the Y variable reaches a specified value. These times are sometime called pseudo failure times.

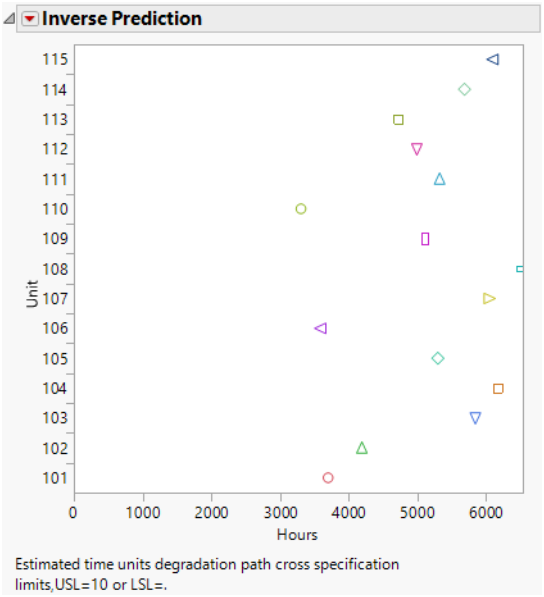
Figure 7.13 Inverse Prediction Tab



Enter either the Lower or Upper Spec Limit. Generally, if your Y variable decreases over time, then enter a Lower Spec Limit. If the Y variable increases over time, then enter an Upper Spec Limit.

For the GaAs Laser example, enter 10 for the Upper Spec Limit and click **Go**. A plot is produced showing the estimated times until the units reach a 10% increase in operating current.

Figure 7.14 Inverse Prediction Plot



The Inverse Prediction red triangle menu contains the following options:

Save Crossing Time Saves the pseudo failure times to a new data table. The table contains a Life Distribution or Fit Life by X script that can be used to fit a distribution to the pseudo failure times. When one of the Inverse Prediction Interval options is enabled, the table also includes the intervals.

Set Upper Spec Limit Sets the upper specification limit.

Set Lower Spec Limit Sets the lower specification limit.

Set Censoring Time Sets the censoring time. The plot updates to show the Censoring Time as a dotted vertical line. If Inverse Prediction Interval > No Interval is selected, observations that exceed the Censoring Time are displayed on horizontal lines starting at the Censoring Time. If Confidence Interval or Prediction Interval is selected, horizontal lines extend indefinitely to the right of observations whose upper limits exceed the Censoring Time. The Censoring Time is reflected in data tables constructed using Save Crossing Time and Generate Pseudo Failure Data.

Use Interpolation through Data Specifies the use of linear interpolation between points (instead of the fitted model) to predict when a unit crosses the specification limit. The behavior depends on whether a unit has observations that exceed the specification limit.

- If a unit has observations exceeding the specification limit, the inverse prediction is the linear interpolation between the observations that surround the specification limit.
- If a unit does not have observations exceeding the specification limit, the inverse prediction is censored and has a value equal to the maximum observed time for that unit.

Inverse Prediction Interval Specifies that confidence or prediction intervals for the pseudo failure times are shown on the Inverse Prediction plot. When intervals are enabled, the intervals are also included in the data table that is created when using the Save Crossing Time option.

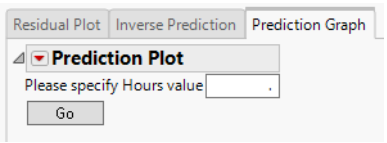
Inverse Prediction Alpha Specifies the alpha level used for the intervals.

Inverse Prediction Side Specifies one or two sided intervals.

Prediction Graph

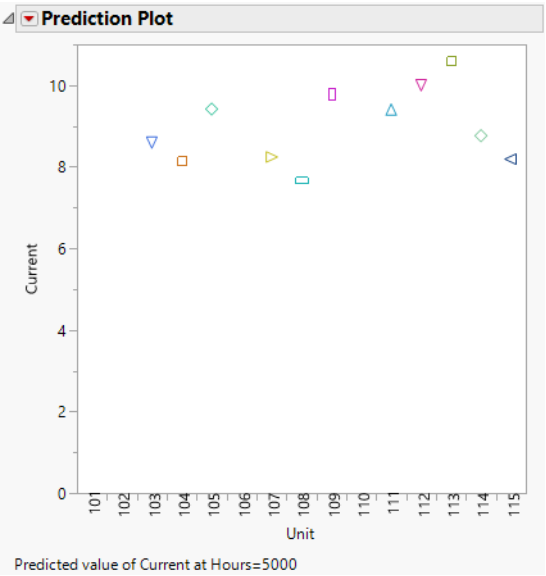
Use the Prediction Graph tab to predict the Y variable for a specified Time value.

Figure 7.15 Prediction Plot Tab



For the GaAs Laser example, no data was collected after 4000 hours. If you want to predict the percent increase in operating current after 5000 hours, enter 5000 and click **Go**. A plot is produced showing the estimated percent decrease after 5000 hours for all the units.

Figure 7.16 Prediction Plot



The Prediction Plot red triangle menu contains the following options:

Save Predictions Saves the predicted Y values to a data table. When one of the Longitudinal Prediction Interval options is enabled, the table also includes the intervals.

Longitudinal Prediction Interval Shows or hides confidence or prediction intervals for the estimated Y on the Prediction Plot. When intervals are enabled, the intervals are also included in the data table that is created when using the Save Predictions option.

Longitudinal Prediction Time Specifies the time value for which you want to predict the Y.

Longitudinal Prediction Alpha Specifies the alpha level used for the intervals.

Degradation Platform Options

The Degradation Data Analysis red triangle menu contains the following options:

Path Definition The Y variable at a given time is assumed to have a distribution. You can model the mean, location parameter, or median of that distribution.

Mean Path Specifies that the mean is the parameter to be modeled.

Location Parameter Path Specifies that the location parameter of the distribution is the parameter to be modeled.

Median Path Specifies that the median of the distribution is the parameter to be modeled.

When the Location Parameter or Median Path option is selected, a menu appears in the Model Specification that enables you to select the distribution of the response.

Degradation Path Style Contains options for selecting the style of degradation path to fit.

Simple Linear Path Enables you to fit linear degradation paths and nonlinear paths that can be transformed to linear paths. See "Simple Linear Path".

Nonlinear Path Enables you to fit nonlinear degradation paths. See "Nonlinear Path".

Graph Options Contains options for modifying the appearance of graphs in the report.

Connect Data Markers Shows or hides lines connecting the points on the Overlay plot.

Show Fitted Lines Shows or hides the fitted lines on the Overlay plot.

Show Spec Limits Shows or hides the specification limits on the Overlay plot.

Show Residual Plot Shows or hides the residual plot.

Show Inverse Prediction Plot Shows or hides the inverse prediction plot

Show Curve Interval Shows or hides the confidence intervals on the fitted lines on the Overlay plot.

Curve Interval Alpha Enables you to change the alpha used for the confidence interval curves.

Show Median Curves Shows or hides median lines on the plot when the Path Definition is set to Location Parameter Path.

Show Legend Shows or hides a legend for the markers used on the Overlay plot.

No Tab List Shows or hides the Residual Plot, Inverse Prediction, and Prediction Graph in tabs or in stacked reports.

Prediction Settings Contains options for modifying the settings used in the model predictions.

Upper Spec Limit Specifies the upper specification limit.

Lower Spec Limit Specifies the lower specification limit.

Censoring Time Specifies the censoring time. See "Inverse Prediction".

Baseline Specifies the normal use conditions for an X variable in nonlinear degradation paths. A path with this value usually results in an Overlay plot.

Inverse Prediction Specifies the interval type, alpha level, and one- or two-sided intervals for inverse prediction. To do inverse prediction, you must also specify the lower or upper specification limit. See "Inverse Prediction".

Longitudinal Prediction Specifies the Time value, interval type, and alpha level for longitudinal prediction. See "Prediction Graph".

Applications Contains options for further analysis of the degradation data.

Generate Pseudo Failure Data Creates a data table giving the predicted time each unit crosses the specification limit. The table contains a Life Distribution or Fit Life by X script that can be used to fit a distribution to the pseudo failure times.

Test Stability Performs stability analysis. See "Stability Analysis".

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Model Reports

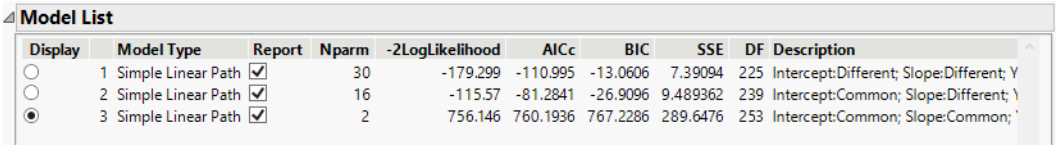
When the **Generate Report for Current Model** button is clicked, summary reports are added in two places:

- An entry is added to the Model List report. See "Model Lists".
- An entry is added to the Reports report. See "Reports".

Model Lists

The Model List report gives summary statistics and other options for every fitted model. Figure 7.17 shows an example of the Model List with summaries for three models.

Figure 7.17 Model List



Model List

Display		Model Type	Report	Nparm	-2LogLikelihood	AICc	BIC	SSE	DF	Description
○	1	Simple Linear Path	☑	30	-179.299	-110.995	-13.0606	7.39094	225	Intercept:Different; Slope:Different; Y
○	2	Simple Linear Path	☑	16	-115.57	-81.2841	-26.9096	9.489362	239	Intercept:Common; Slope:Different; \
◉	3	Simple Linear Path	☑	2	756.146	760.1936	767.2286	289.6476	253	Intercept:Common; Slope:Common;

Display Select the model that you want represented in the Overlay plot, Residual Plot, Inverse Prediction plot, and Prediction Graph.

Model Type Gives the type of path, either linear or nonlinear.

Report Select the check boxes to display the report for a model. For more information about the reports, see "Reports".

Nparm Gives the number of parameters estimated for the model.

-2LogLikelihood Gives twice the negative of the log-likelihood. See *Fitting Linear Models*.

AICc Gives the corrected Akaike Criterion. See *Fitting Linear Models*.

BIC Gives the Bayesian Information Criterion. See *Fitting Linear Models*.

SSE Gives the error sums-of-squares for the model.

DF Gives the error degrees of freedom.

Description Gives a description of the model.

Reports

The Reports outline node gives details about each model fit. The report includes a Model Summary report, and an Estimate report.

The Model Summary report contains the following information:

<Y, Response> Scale The transformation on the response variable.

<Time> Scale The transformation on the time variable.

SSE The error sums-of-squares.

Nparm The number of parameters estimated for the model.

DF The error degrees of freedom.

RSquare The R-square.

MSE The mean square error.

The Estimate report contains the following information:

Parameter The name of the parameter.

Estimate The estimate of the parameter.

Std Error The standard error of the parameter estimate.

t Ratio The t statistic for the parameter, computed as the Estimate divided by the Std Error.

Prob>|t| The *p*-value for a two-sided test for the parameter.

Destructive Degradation

To measure a product characteristic, sometimes the product must be destroyed. For example, when measuring breaking strength, the product is stressed until it breaks. Regular degradation analysis no longer applies in these situations. You can handle these situations in one of two ways:

- If your failure time model is a standard one, it might be covered by the Destructive Degradation platform. See "Destructive Degradation".
- If you are using custom transformations of the time or response variables and custom nonlinear models, use the Degradation platform. In the launch window, select the Destructive Degradation tab.

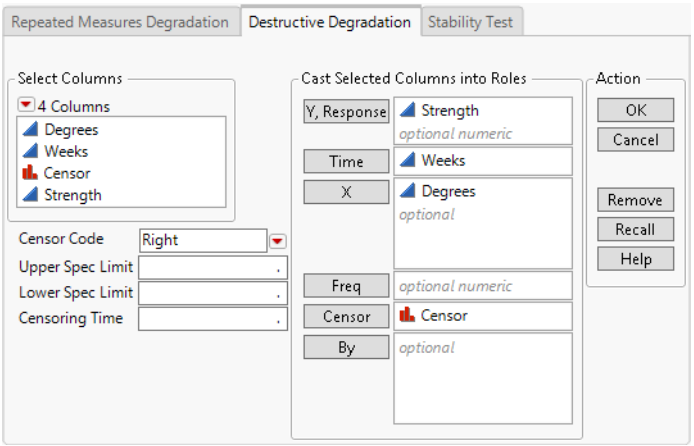
Example of Accelerated Destructive Degradation

This example fits a custom nonlinear model. The data consist of measurements on the strength of an adhesive bond. The product is stressed until the bond breaks, and the required breaking strength is recorded. Because units at normal use conditions are unlikely to break, the units were tested at several levels of an acceleration factor (temperature). You want to estimate the strength (in newtons) of units after 52 weeks (one year) at use conditions of 25°C.

Complete the Launch Window

1. Select **Help > Sample Data Library** and open Reliability/Adhesive Bond.jmp.
2. Select **Analyze > Reliability and Survival > Degradation**.
3. Select the **Destructive Degradation** tab.
4. Select Strength and click **Y, Response**.
5. Select Weeks and click **Time**.
6. Select Degrees and click **X**.
7. Select Censor and click **Censor**.

Figure 7.18 Completed Launch Window



8. Click **OK**.

Define and Fit the Model

1. Select **Lognormal** from the menu in the Location Parameter Path Specification panel in the Model Specification outline.

   This specifies a lognormal transformation for the response, Strength.
2. Click the Degradation Data Analysis red triangle and select **Degradation Path Style > Nonlinear Path**.

This adds a script window to the report. You insert a script that specifies your model in this window.

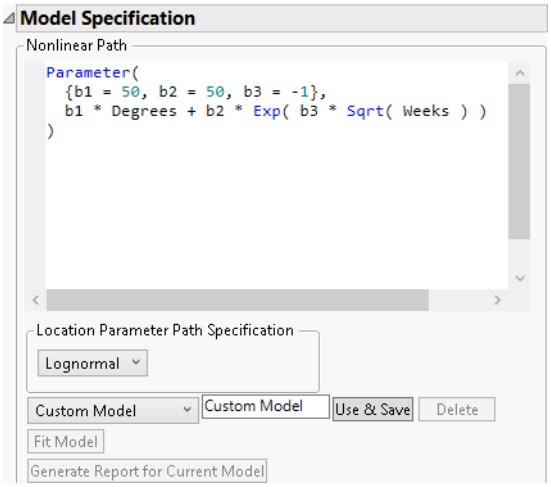
3. Copy the JSL formula below and paste it into the script window under Nonlinear Path:

```
Parameter(
   {b1 = 50, b2 = 50, b3 = -1},
   b1 * :Degrees + b2 * Exp( b3 * Sqrt( :Weeks ) )
)
```

The script defines a model for Strength in terms of parameters b1, b2, and b3. The script specifies initial values for the parameters.

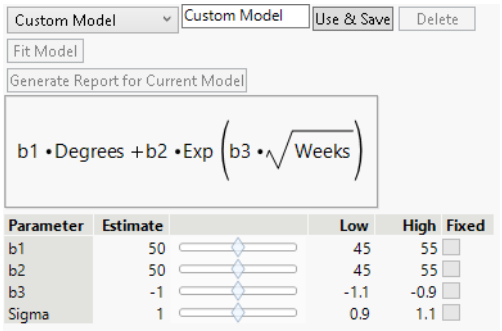
4. In the text box at the bottom of the report, change Empty to Custom Model.

Figure 7.19 Nonlinear Path Script



5. Click **Use & Save**.

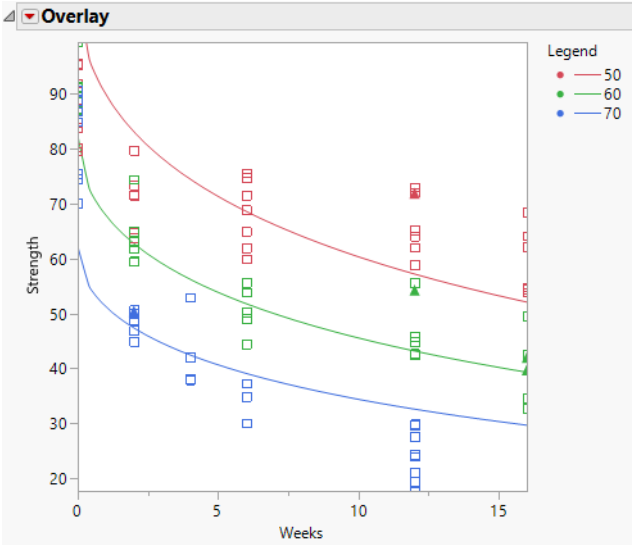
Figure 7.20 Updated Model Specification Outline



The report is updated to include controls for changing initial values for the parameters. Initial values for the parameters are set to the values specified in the formula in the script editor.

6. Click **Fit Model**.

Figure 7.21 Plot of Fitted Model



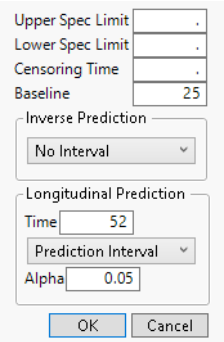
The Parameter panel in the Model Specification outline is updated to show the parameter estimates for the fitted model. The model fit is shown in the Overlay plot. You can drag the axes to show the points, as is done in Figure 7.21. The legend identifies the curves.

Obtain Predicted Values and Prediction Intervals

Next, you obtain a predicted value and prediction interval for Strength after 52 weeks at the baseline use condition of 25°C.

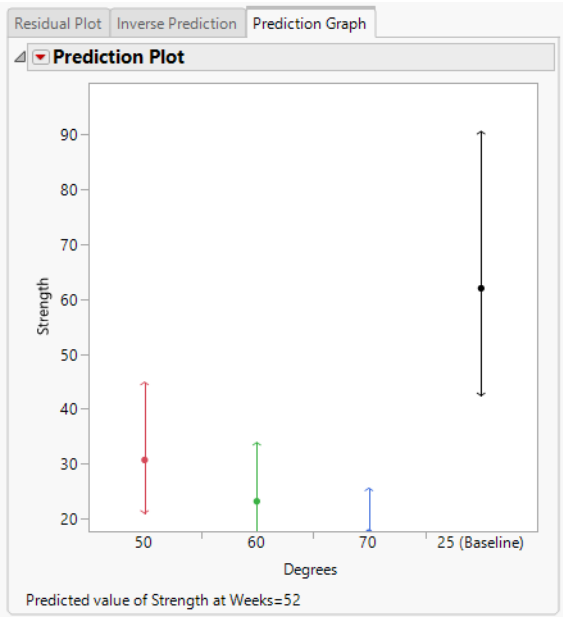
1. Click the Degradation Data Analysis red triangle and select **Prediction Settings**.
2. In the **Prediction Settings** window:
   - Enter 25 for Baseline.
   - Enter 52 for Time in the Longitudinal Prediction panel.
   - Select Prediction Interval from the menu in the Longitudinal Prediction panel.

Figure 7.22 Prediction Settings



3. Click **OK**.
4. Select the **Prediction Graph** tab in the report.

Figure 7.23 Prediction Plot at Weeks = 52



The Prediction Plot shows the predicted value and its 95% level prediction interval above the axis label of 25 (Baseline).

5. Click the Prediction Plot red triangle and select **Save Predictions**.

   Predicted values for the three values of Degrees and for the desired baseline of 25 degrees are saved to a data table. The predicted strength of the adhesive bond after 1 year at 25° C is 61.96, with a prediction interval of 42.42 to 90.50.

Stability Analysis

Stability analysis is used in setting pharmaceutical product expiration dates. Three linear degradation models are fit, and an expiration date is estimated following International Conference on Harmonisation (ICH) guidelines. The ICH guidelines are used for the general framework of determining if batches can be pooled for expiration dating (ICH Q1E 2003). For specific implementation details, see the STAB macro and FDA guidelines in Chow (2007, Appendix B).

The Stability Tests report summarizes three degradation models and their corresponding earliest crossing times. The best model is selected and displayed in the Overlay plot. The models are listed in the order of complexity.

The first model fits different intercepts and different slopes to each batch. (In the procedure steps and model comparisons described below, this is the *full model*.) When this model is used for estimating the expiration date, the mean square error (MSE) is not pooled across batches. Prediction intervals are computed for each batch using individual mean squared errors, and the interval that crosses the specification limit first is used to estimate the expiration date. The earliest crossing times are based on 95% two-sided prediction intervals when there are two specification limits provided; the earliest crossing times are based on 95% one-sided prediction intervals when there is only one specification limit provided.

The second model fits different intercepts to each batch, but fits a common slope across all batches. The third model fits a common intercept and a common slope across all batches. When this model is appropriate, it provides the expiration date furthest into the future.

Figure 7.24 Stability Tests Summary

Stability Tests

The best model accepted at the significance level of 0.25 has Different intercepts and Common slopes. The model suggests the earliest crossing time at 23.47505 with 95 percent confidence. ICH Guidelines indicate an expiration time of 23.47505.

Display	Intercept	Slope	Earliest Crossing Time
○	Different	Different	23.11596
◉	Different	Common	23.47505
○	Common	Common	29.98567

The Model Comparisons section of the Stability Tests report summarizes the tests of significance for each of the stability models. The Legend describes each source. The procedure for determining the best model for expiration dating considers the *p*-values for Sources C and B. The sources are listed below in reverse order to accommodate the order of steps in the procedure.

Tip: The ICH guidelines specify using a significance level of 0.25 to determine the appropriate model. You can follow the steps below and compare the *p*-values at each source to a different significance level. If this results in a different model being selected, you can use the radio buttons in the Display column of the table in the Stability Tests summary report.

Source E Specifies the sum of squared responses minus the Source D sum of squares. The value in the Mean Square column for Source E is the Source E SS value divided by the Source E degrees of freedom, which are equal to the number of parameters in the full model (different intercepts and different slopes).

Source D Specifies the error sum of squares and corresponding MSE for the full model (different intercepts and different slopes).

Source C Specifies the test of equal slopes. This is a test of the second model (different intercepts and common slope) versus the full model (different intercepts and different slopes).

- If the *p*-value is less than 0.25, the slopes are assumed to be different across batches. The procedure stops and the full model (different intercepts and different slopes) is used to estimate the expiration date.
- If the *p*-value is greater than or equal to 0.25, the slopes are assumed to be common across batches and you then evaluate the intercepts using Source B.

Source B Specifies the test for equal intercepts. This is a test of the third model (common intercepts and common slopes) versus the second model (different intercepts and common slope).

- If the *p*-value is less than 0.25, the intercepts are assumed to be different across batches, and the second model (different intercepts and common slope) is used to estimate the expiration date.
- If the *p*-value is greater than or equal to 0.25, the intercepts are assumed to be common across batches, and the third model (common intercepts and common slopes) is used to estimate the expiration date.

Source A Specifies the test of the third model (common intercepts and common slopes) versus the full model (different intercepts and different slopes). This test is not used in the procedure to determine the best model for expiration dating.

Figure 7.25 Stability Model Comparisons

Model Comparisons

Source	DF	SS	Mean Square	F Statistic	Prob>F
A	6	61.48956	10.24826	10.10836	<.0001*
B	3	60.48866	20.16289	19.88764	<.0001*
C	3	1.000906	0.333635	0.329081	0.8043
D	28	28.38752	1.01384		
E	8	360214.5	45026.81		

Legend

Source		Intercept	Slope	Intercept	Slope
A		Different	Different	Common	Common
B		Different	Common	Common	Common
C		Different	Different	Different	Common
D	Residual				
E	Whole Model				

The Reports section contains models fit by the Test Stability option. The following four models are fit by the Test Stability option:

- A different intercept, different slope model where the MSE (mean squared error) is pooled across batches. (This is an alternative to the first model in the Summary report.)
- A different intercept, common slope model. (This is the second model in Summary report.)
- A common intercept, common slope model. (This is the third model in Summary report.)
- A different intercept, different slope model where the MSE (mean squared error) is not pooled across batches. (This is the first model in the Summary report.)

Caution: In addition to the four models fit by the Test Stability option, the Reports section also contains any models fit in the Degradation platform prior to running the Test Stability option.

Example

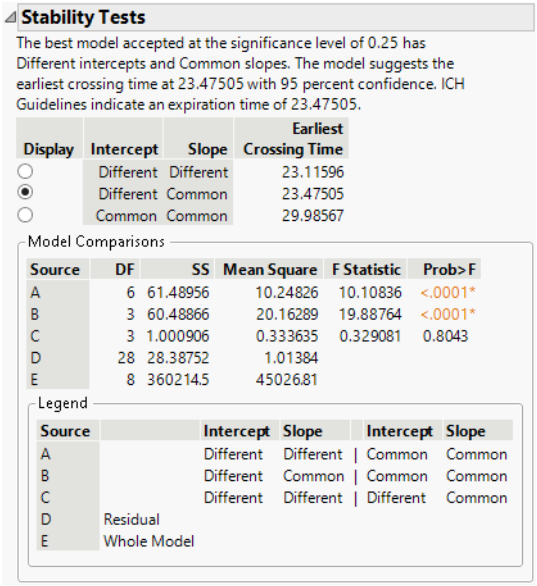
Use the data in the Stability.jmp sample data table to establish an expiration data for a new product. The data consists of product concentration measurements on four batches. A concentration of 95 is considered the end of the product's usefulness.

To perform the stability analysis, do the following steps:

1. Select **Help > Sample Data Library** and open Reliability/Stability.jmp.
2. Select **Analyze > Reliability and Survival > Degradation**.
3. Select the **Stability Test** tab.
4. Select Concentration (mg/Kg) and click **Y, Response**.
5. Select Time and click **Time**.
6. Select Batch Number and click **Label, System ID**.
7. Enter 95 for the Lower Spec Limit.

8. Click **OK**.

Figure 7.26 Stability Models



Stability Tests

The best model accepted at the significance level of 0.25 has Different intercepts and Common slopes. The model suggests the earliest crossing time at 23.47505 with 95 percent confidence. ICH Guidelines indicate an expiration time of 23.47505.

Display	Intercept	Slope	Earliest Crossing Time
○	Different	Different	23.11596
◉	Different	Common	23.47505
○	Common	Common	29.98567

Model Comparisons

Source	DF	SS	Mean Square	F Statistic	Prob>F
A	6	61.48956	10.24826	10.10836	<.0001*
B	3	60.48866	20.16289	19.88764	<.0001*
C	3	1.000906	0.333635	0.329081	0.8043
D	28	28.38752	1.01384		
E	8	360214.5	45026.81		

Legend

Source		Intercept	Slope	Intercept	Slope
A		Different	Different	Common	Common
B		Different	Common	Common	Common
C		Different	Different	Different	Common
D	Residual				
E	Whole Model				

The test for equal slopes has a *p*-value of 0.8043. Because this is larger than a significance level of 0.25, the test is not rejected, and you conclude the degradation slopes are equal between batches.

The test for equal intercepts and slopes has a *p*-value of <.0001. Because this is smaller than a significance level of 0.25, the test is rejected, and you conclude that the intercepts are different between batches.

Because the test for equal slopes was not rejected, and the test for equal intercepts was rejected, the chosen model is the one with Different Intercepts and Common Slope. This model is the one selected in the report, and gives an estimated expiration date of 23.475.

Chapter 8

Destructive Degradation

Model Product Deterioration over Time

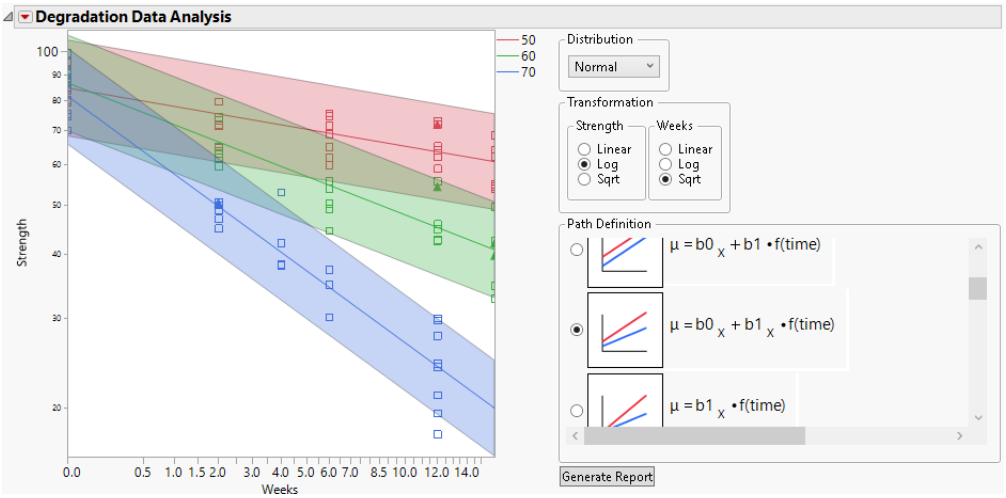
To measure a product characteristic, sometimes the product must be destroyed. For example, when measuring breaking strength, the product is stressed until it breaks. Because the test is destructive, there is only one observation per product unit. In such a situation, you can model product reliability using the Destructive Degradation platform.

The platform models how a (typically) nonnegative response changes over time. Observations are assumed to be independent and measure the value of the response and the time at failure. A large and flexible collection of pre-defined models is provided. The models include location-scale and log-location-scale distributions whose location parameters are functions of time. The models allow explanatory variables and additional parameters. When an explanatory variable is specified, the platform fits an accelerated destructive degradation model.

Note: If you require a model that is not represented among the models provided, you can use the Degradation platform. See "Destructive Degradation".

For more information about destructive degradation and reliability, see Escobar et al. (2003) and Meeker and Escobar (1998).

Figure 8.1 Destructive Degradation Example of Model for Adhesive Bond.jmp



Contents

Example of the Destructive Degradation Platform 211
Launch the Destructive Degradation Platform 217
Specify Two Y Columns 218
The Destructive Degradation Plot Options and Models 219
Plot Options 220
Models 221
The Destructive Degradation Report 225
Model List 226
Model Outlines 226
Destructive Degradation Platform Options 229
Statistical Details for the Destructive Degradation Platform 229

Example of the Destructive Degradation Platform

This example of an accelerated destructive degradation model is patterned after an example from Escobar et al. (2003). The data consist of measurements on the strength (measured in newtons) of an adhesive bond. Temperature is considered to be an acceleration factor. The product is stressed until the bond breaks and the required breaking stress is recorded. Because units at normal temperatures are unlikely to break, the units were tested at several levels over a wide range of temperatures. Strength less than 50 newtons is considered failure. You want to estimate the proportion of units with a strength below 50 newtons after 260 weeks (5 years) at a reference temperature of 35 degrees Celsius.

This example has three stages:

- "Perform the Initial Analysis"
- "Change the Model and Generate the Report"
- "Using the Profilers for Prediction"

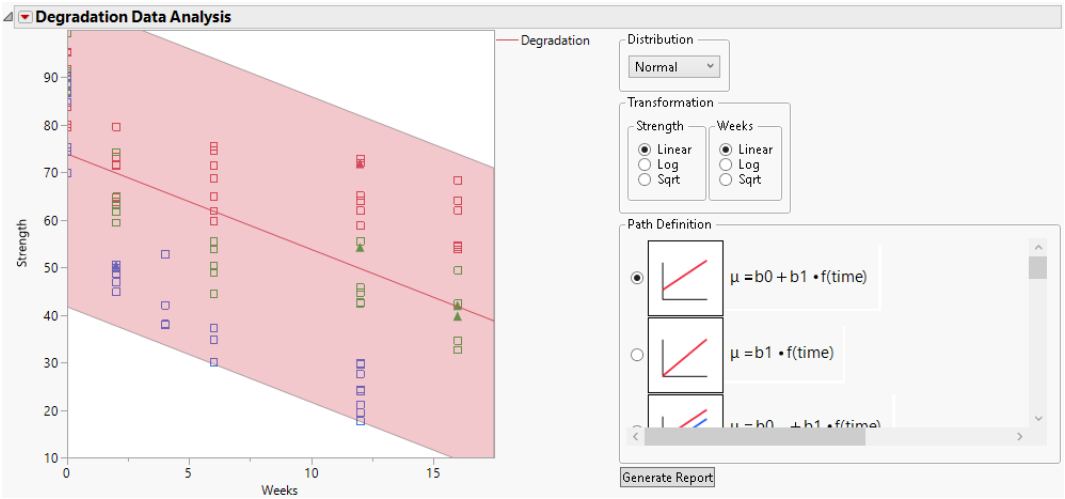
Perform the Initial Analysis

1. Select **Help > Sample Data Library** and open Reliability/Adhesive Bond.jmp.
2. Select **Analyze > Reliability and Survival > Destructive Degradation**.
3. Select Strength and click **Y, Response**.
4. Select Weeks and click **Time**.
5. Select Degrees and click **X**.

   The temperature is the accelerating factor in the experiment.
6. Select Censor and click **Censor**.

   Notice that the **Censor Code** is set to Right.
7. Click **OK**.

Figure 8.2 Initial Degradation Plot



The platform specifies a default model. The default model assumes that the data are described by a single Normal distribution, whose location parameter is a linear function of time.

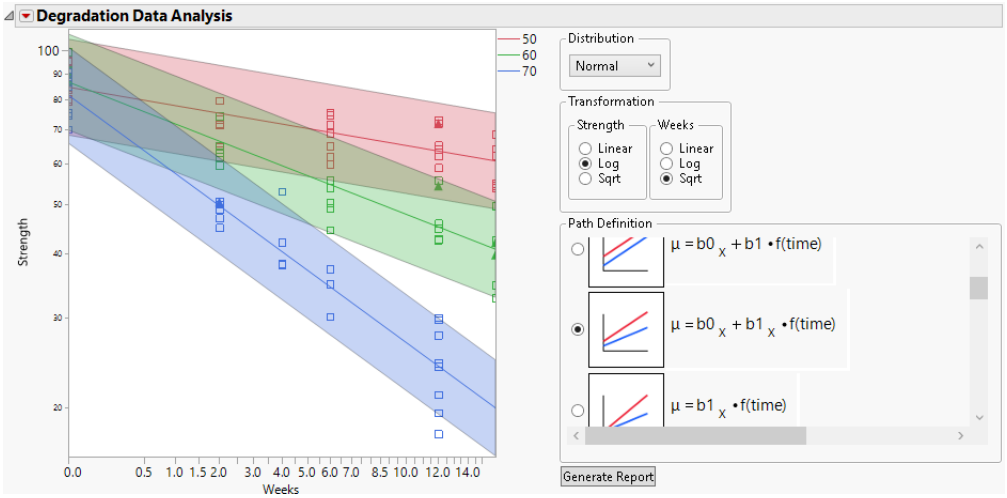
Change the Model and Generate the Report

1. Select **Log** for the Y (Strength) Transformation.
2. Select **Sqrt** for the Time (Weeks) Transformation.
3. Select $\mu = b0_x + b1_x * f(time)$ for the Path Definition.

   The subscript "x" denotes the accelerating variable, which is Degrees in this example.

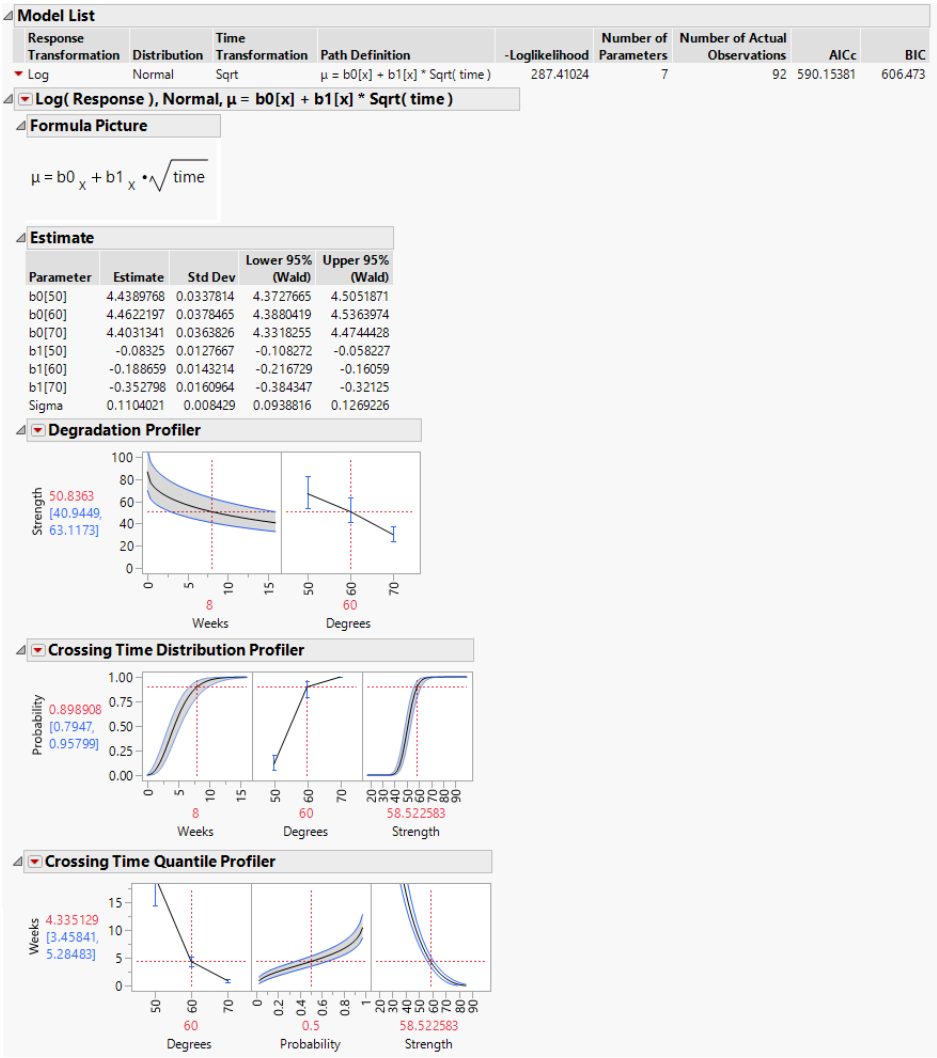
Note: This model is linear in all parameters.

Figure 8.3 Plot Showing Model



4. Click **Generate Report**.

Figure 8.4 Report for Basic Model



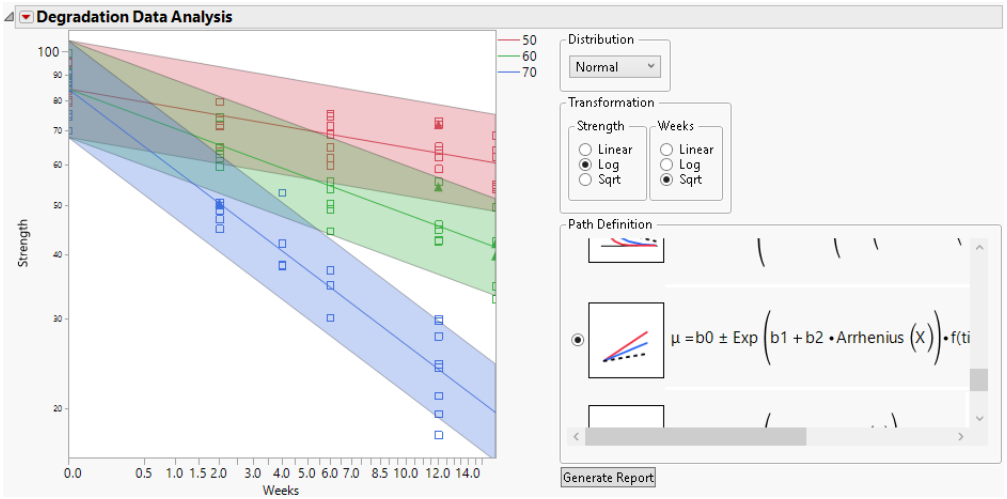
The estimates of the slope b1 at the three values of Degrees suggest that degradation occurs more quickly at higher temperatures. Failure mechanisms that depend on chemical processes are often well modeled using the Arrhenius model for temperature. For this reason, you now fit a model where an Arrhenius transformation is applied to Degrees, which is measured on a Celsius scale.

5. Select μ = b0 ± Exp(b1 + b2*Arrhenius(X))*f(time) for the Path Definition.

Note: This model is not linear in the parameters.

6. Select **Celsius** and click **OK**.

Figure 8.5 Plot Showing Model with Arrhenius Transformation



7. Click **Generate Report**.

Figure 8.6 Report Including Second Model with Arrhenius Transformation

Model List

Response Transformation	Distribution	Time Transformation	Path Definition	-Loglikelihood	Number of Parameters	Number of Actual Observations	AICc	BIC
Log	Normal	Sqrt	μ = b0[x] + b1[x] * Sqrt(time)	287.41024	7	92	590.15381	606.473
Log	Normal	Sqrt	μ = b0 - Exp(b1 + b2 * Arrhenius(X)) * Sqrt(time)	288.09559	4	92	584.65095	594.27834

Log(Response), Normal, μ = b0[x] + b1[x] * Sqrt(time)

Log(Response), Normal, μ = b0 - Exp(b1 + b2 * Arrhenius(X)) * Sqrt(time)

Formula Picture

$$\mu = b0 - \text{Exp}\left(b1 + b2 \bullet \text{Arrhenius}\left(X\right)\right) \bullet \sqrt{\text{time}}$$

Estimate

Parameter	Estimate	Std Dev	Lower 95% (Wald)	Upper 95% (Wald)
b0	4.4354686	0.0205347	4.3952214	4.4757158
b1	22.825503	1.4029332	20.075805	25.575202
b2	-0.704728	0.0414658	-0.786	-0.623457
Sigma	0.1114956	0.0085011	0.0948337	0.1281574

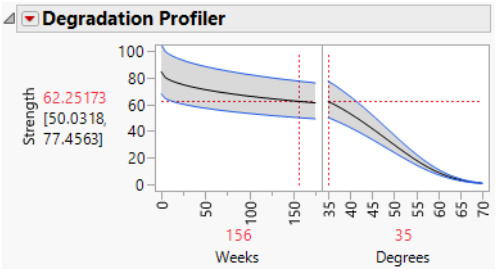
Using the Profilers for Prediction

Because the Arrhenius model shows a better fit, as indicated by its smaller AICc and BIC values (Figure 8.6), you continue your analysis using this model.

Recall that strength less than 50 newtons is considered failure. You are interested in units lasting 156 weeks (three years) at a reference temperature of 35 degrees Celsius. Change the settings in the profilers to reflect these values. Click the value in red beneath each plot's horizontal axis and enter the new value.

1. In the Degradation profiler for the Arrhenius model, set **Weeks** to 156 and **Degrees** to 35.

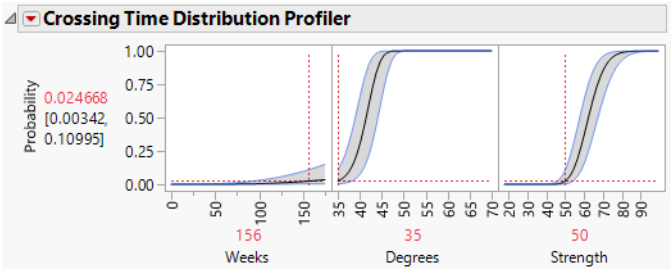
Figure 8.7 Degradation Profiler



The predicted **Strength** at these settings is 62.25173, with a 95% prediction interval ranging from 50.0318 to 77.4563. Failures are not very likely at these or less extreme settings.

2. In the Crossing Time Distribution Profiler, set **Weeks** to 156, **Degrees** to 35, and **Strength** to 50.

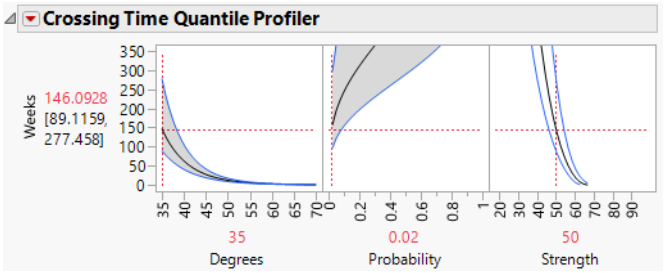
Figure 8.8 Crossing Time Distribution Profiler



At 156 weeks at a temperature of 35 degrees Celsius, the probability that the value of Strength is less than 50 is 0.024668. The 95% confidence interval ranges from 0.00342 to 0.10995. The probability of failure at these or less extreme conditions is about 2%.

3. In the Crossing Time Quantile Profiler, set **Degrees** to 35, **Probability** to 0.02, and **Strength** to 50.
4. Adjust the vertical axis of the Crossing Time Quantile Profiler so that the maximum value is about 350.

Figure 8.9 Crossing Time Quantile Profiler

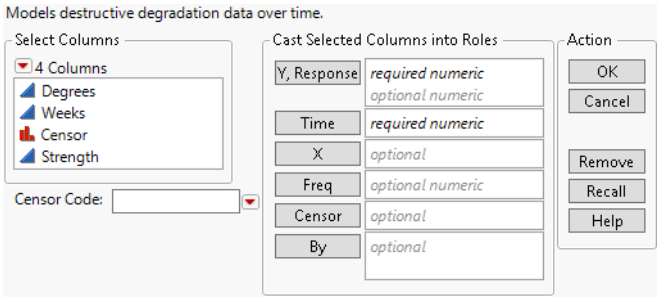


The number of weeks within which 2% of units fail at a temperature of 35 degrees Celsius is estimated to be 146.0928.The 95% confidence interval ranges from 89.1159 to 277.458.

Launch the Destructive Degradation Platform

Launch the Destructive Degradation platform by selecting **Analyze** > **Reliability and Survival** > **Destructive Degradation**. Figure 8.10 shows the Destructive Degradation launch window using the Adhesive Bond.jmp data table (located in the Reliability folder).

Figure 8.10 The Destructive Degradation Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Destructive Degradation launch window contains the following options:

Y, Response Identifies the column containing the degradation measurements. When your response values are interval-censored, you can enter two columns. See "Specify Two Y Columns".

Time Identifies the column containing the time values.

X Identifies an optional explanatory variable. If the distribution of Y changes not only over time, but is also impacted by some other variable, that additional variable can be supplied

as X. Use this role to specify the accelerating factor in an accelerated destructive degradation model.

Freq Identifies frequencies or observation counts when there are multiple units. If the value is 0 or a positive integer, then the value represents the frequencies or counts of observations with the given row's settings.

Censor Identifies an optional column that identifies censored response measurements. When there is only one column in the Y role, this column indicates whether the response Y in a given row is exact or right censored. A right-censored observation is one where the exact measurement is unknown, but is known to be larger than the Y value in the corresponding row.

Select the value that identifies right-censored observations from the Censor Code menu. Rows corresponding to other values in the Censor column are treated as uncensored. Rows with missing censor code values are excluded from the analysis. JMP attempts to detect the censor code and display it in the list.

By Identifies an optional By variable. A separate analysis is produced for each level of this variable.

Censor Code Identifies the value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

Specify Two Y Columns

You can specify two Y columns when some of the degradation measurements are interval censored or left censored. For a given row, the values in the two Y columns determine the type of censoring.

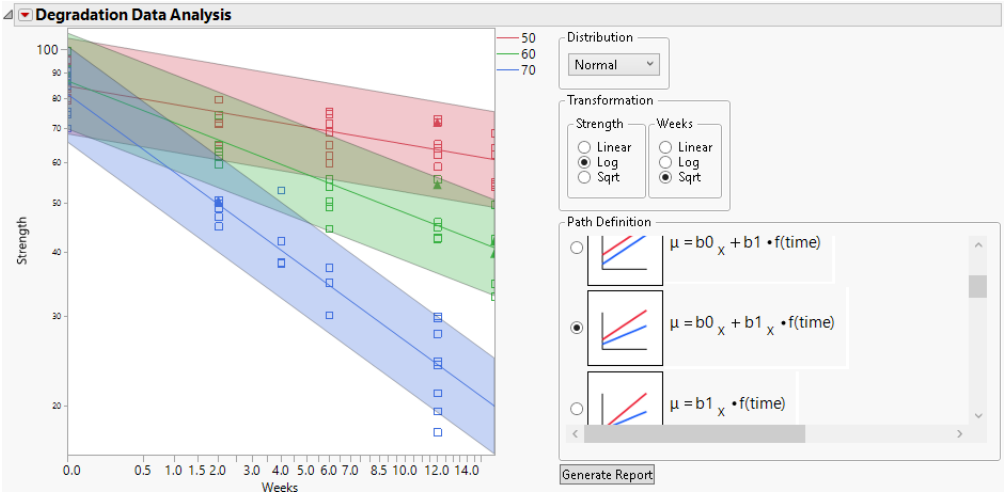
- If the two Y values are equal and neither is missing, then the common measurement is treated as exact.
- If the two Y values are not equal and neither is missing, then the measurement is interval censored and assumed to be between the two values.
- If only the first value is missing, then the measurement is left censored and assumed to be smaller than the second value.
- If only the second value is missing, then the measurement is right censored and assumed to be larger than the first value.

Note: The only way to fit left-censored measurements in the Destructive Degradation platform is through the use of two Y columns.

The Destructive Degradation Plot Options and Models

The Degradation Data Analysis plot shows the data and a graphical representation of the model that is currently specified based on selections of Distribution, Transformation, and Path Definition. The plot shown in Figure 8.11 is for the Adhesive Bond.jmp data table and represents a model that includes an optional X variable.

Figure 8.11 Destructive Degradation Plot and Options



Note the following about the plot:

- Data values are represented by markers.

Table 8.1 Marker Descriptions

Icon	Description
□	An exact measurement.

Table 8.1 Marker Descriptions *(Continued)*

Icon	Description
▼	Left-censored observation, indicating that the censored measurement is below the triangle. **Note:** In the Degradation platform, left-censoring arises only when observations are interval censored. See "Specify Two Y Columns".
▲	Right-censored observation, indicating that the censored measurement is above the triangle.
▼ ▲	Interval-censored observation, indicating that the censored value is within the specified interval.

Note: By default, the markers are not colored. To color them to match the model color scheme, select **Rows > Color or Mark by Column**. Select the X column. Deselect **Continuous Scale**. Select **JMP Default** from the Colors menu.

- For each level of X, a colored band appears. If the model does not include an X variable, then a single band appears. For a given value of Time, the upper and lower bounds of the band are the 0.025 and 0.975 percentiles of the fitted distribution of Y. The colors of the bands correspond to the values of X, as indicated by the legend to the upper right of the plot.

Note: Marker colors correspond to the color states assigned in the data table.

- The solid curve in the center of a band is the median of the fitted distribution of Y for the corresponding value of X over time. If the model does not include an X variable, then the curve plots the median of Y over time.

Plot Options

Distribution Choose a location-scale or a log-location-scale distribution.

Note: It is not recommended to fit a log-location-scale distribution model when you specify a Log transformation for the response column.

Transformation Choose a transformation function for the response Y and for the Time variable.

Note: If you apply the Log transformation to a column that contains nonpositive values, the rows with nonpositive values are omitted from the model fit. If you apply the Sqrt transformation to a column that contains negative values, the rows with negative values are omitted from the model fit.

Path Definition Choose a linear or a nonlinear path for the regression model. For more information about each model, see "Models".

Generate Report Creates a report for the specified model. The first time you select Generate Report, a Model List outline is created. When you select Generate Report to fit other models, the Model List outline is updated and an outline is added for each model.

Models

The following table provides the equations for each model in the Path Definition list. For a description of each model, follow the link.

Note: The thumbnail sketch shown to the left of each equation shows a generic plot of the behavior of the location parameter, μ, over time. In the report's main plot, the plot of the estimated median can differ from the thumbnail based on your selections for Distribution and Transformation.

Table 8.2 Model Equations

Model	Equation
"Common Path with Intercept"	$\mu = b0 + b1 * f(time)$
"Common Path without Intercept"	$\mu = b1 * f(time)$
"Common Slope"	$\mu = b0_X + b1 * f(time)$
"Individual Path with Intercept"	$\mu = b0_X + b1_X * f(time)$
"Individual Path without Intercept"	$\mu = b1_X * f(time)$

Table 8.2 Model Equations *(Continued)*

Model	Equation
"Common Intercept"	$\mu = b0 + b1_X * f(time)$
"First-Order Kinetics Type 1"	$\mu = b0 - b1 * Exp[-b2 * Exp[b3 * [Arrhenius(X_0) - Arrhenius(X)]] * f(time)]$
"First-Order Kinetics Type 2"	$\mu = b0 * [1 - Exp[-b1 * Exp[b2 * [Arrhenius(X_0) - Arrhenius(X)]] * f(time)]]$
"First-Order Kinetics Type 3"	$\mu = b0 + b1 * Exp[-b2 * Exp[b3 * [Arrhenius(X_0) - Arrhenius(X)]] * f(time)]$
"First-Order Kinetics Type 4"	$\mu = b0 * Exp[-b1 * Exp[b2 * [Arrhenius(X_0) - Arrhenius(X)]] * f(time)]$
"Arrhenius Rate"	$\mu = b0 \pm Exp[b1 + b2 * Arrhenius(X)] * f(time)$
"Polynomial Rate"	$\mu = b0 \pm Exp[b1 + b2 * Log(X)] * f(time)$
"Exponential Rate"	$\mu = b0 \pm Exp[b1 + b2 * X] * f(time)$

Models That Are Linear in Transformed Time

Common Path with Intercept

This model fits a single distribution whose location parameter changes linearly over transformed time. This model fits a common intercept and a common slope, regardless of whether there is an X variable.

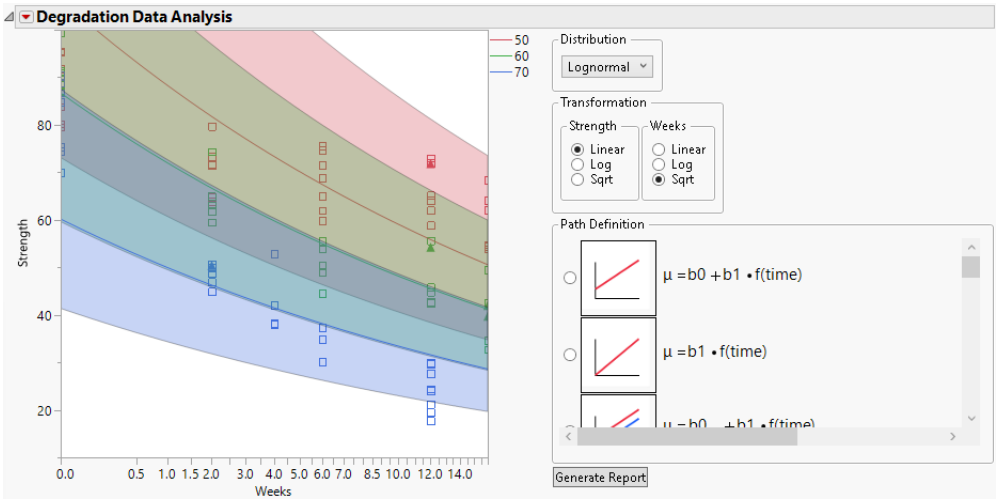
Common Path without Intercept

This model fits a single distribution whose location parameter changes linearly over transformed time but whose value at time zero is zero. Based on selections for Distribution and Transformation, the median curve might not be a straight line and might not pass through origin. This model fits a zero intercept and a common slope, regardless of whether there is an X variable.

Common Slope

In this model, the location parameters are linear functions of transformed time with separate intercepts for the values of X but a common slope. Based on selections for Distribution and Transformation, the model fits might appear as curves. For example, selecting a Lognormal distribution gives the plot in Figure 8.12.

Figure 8.12 Common Slope Model Using a Lognormal Distribution



Individual Path with Intercept

In this model, the location parameters are linear functions of transformed time with separate intercepts and separate slopes for the values of X.

Individual Path without Intercept

In this model, the location parameters are linear functions of transformed time with zero intercepts and separate slopes for the values of X.

Common Intercept

In this model, the location parameters are linear functions of transformed time with a common intercept and separate slopes for the values of X.

First-Order Kinetics Models

Four first-order kinetics models where the location parameter is a nonlinear function based on an Arrhenius transformation of temperature are provided. Each of these location models fit separate models for each value of the optional explanatory variable X.

When you first select any of these models, you must specify the measurement scale for temperature and a reference temperature value, X_0. The specified reference temperature affects the interpretation of *b2* in these models. The *b2* parameter is the rate constant at X_0. The value for X_0 is used to construct a time acceleration factor (Meeker and Escobar 1998). If you subsequently select another of the first-order kinetics models or the Arrhenius Rate model (see "Arrhenius Rate"), the platform remembers and uses these specifications.

First-Order Kinetics Type 1

In this model, *b1* and *b2* are positive. On a linear scale, the curves have an upper asymptote at *b0* as time approaches infinity.

First-Order Kinetics Type 2

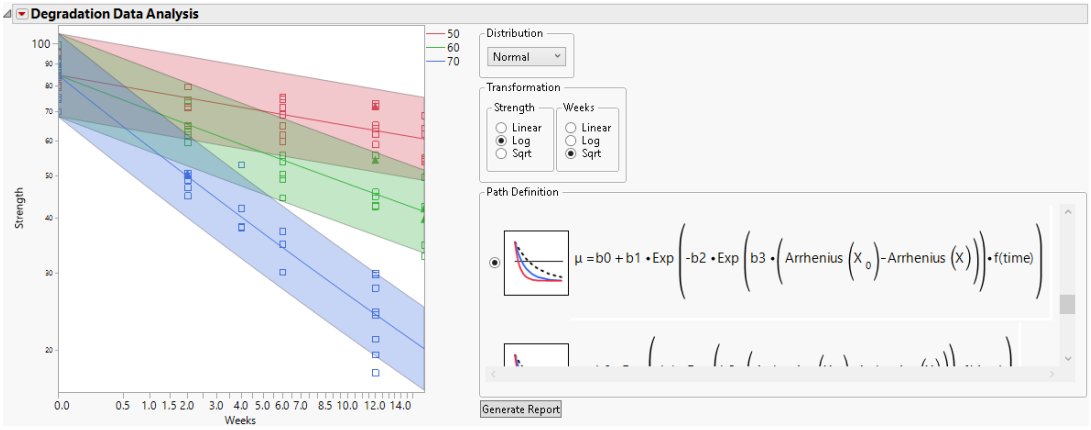
In this model, the location parameter is zero at time zero. Both *b0* and *b1* are positive. On a linear scale, the curves have an upper asymptote at *b0* as time approaches infinity. You can think of the Type 2 model as a vertically shifted version of the Type 1 model.

First-Order Kinetics Type 3

In this model, both *b1* and *b2* are positive. Because of the sign preceding *b1* is the opposite of the sign preceding *b1* in the Type 1 model, this model is an inverted version of the Type 1 model. On a linear scale, it has a lower asymptote at *b0* as time approaches infinity.

Given data exhibiting a negative slope over time, the fitted model can produce a plot similar to Figure 8.13. The figure is for Adhesive Bond.jmp. The selected temperature measurement scale is Celsius and the specified reference temperature under typical use conditions is 35 degrees.

Figure 8.13 Example of First-Order Kinetics Model Type 3



First-Order Kinetics Type 4

This model is a vertically shifted version of the Type 3 model. On a linear scale, the curves have a lower asymptote at 0 as time approaches infinity.

Rate Models

Three models where the location parameter is an exponential function of the transformed X variable are provided. Each of these location models fits common intercept and separate slope models for each value of the optional explanatory variable X. For each of these models, on a linear scale, the location parameter is linear in the transformed time values.

Arrhenius Rate

This model involves an exponential function of the Arrhenius transformation multiplied by transformed time. When you select this model, you are asked to specify the measurement scale for temperature, unless you have already supplied this information.

Polynomial Rate

This model involves an exponential function of a linear function of the log of X multiplied by transformed time.

Exponential Rate

This model involves an exponential function of a linear function of X multiplied by transformed time.

The Destructive Degradation Report

The Destructive Degradation report contains an outline for each model that you fit. When you fit a model, the Model List is updated with a row for that model.

Note: All models are fit using the maximum likelihood method.

Figure 8.14 Model List and Model Outlines

Model List

Response Transformation	Distribution	Time Transformation	Path Definition	-Loglikelihood	Number of Parameters	Number of Actual Observations	AICc	BIC
Log	Normal	Sqrt	μ = b0[x] + b1[x] * Sqrt(time)	287.41024	7	92	590.15381	606.473
Log	Normal	Sqrt	μ = b0 - Exp(b1 + b2 * Arrhenius(X)) * Sqrt(time)	288.09559	4	92	584.65095	594.27834

Log(Response), Normal, μ = b0[x] + b1[x] * Sqrt(time)

Log(Response), Normal, μ = b0 - Exp(b1 + b2 * Arrhenius(X)) * Sqrt(time)

Model List

The first four columns in the Model List reflect the choices that you made in the plot options. The -Loglikelihood, AICc, and BIC statistics are information-based measures that can be used for model comparisons. For descriptions of these measures, see *Fitting Linear Models*.

The three information-based measures in the Model List are comparable across models as long as the models being compared have the same Number of Actual Observations. If this is not the case, exercise caution because different models might use different subsets. The Number of Actual Observations might be reduced due to the choice of distribution or the choice of transformation. Choosing a log-location-scale distribution excludes all non-positive Y values. Also, the Log and Sqrt transformations exclude all non-positive values.

Each row of the Model List table has a red triangle menu with the following options:

Scroll To Scrolls the report window to the corresponding model outline.

Remove Removes the model from the Model List and removes the corresponding model outline from the report.

Model Outlines

The outline for each model contains a red triangle menu with the following option:

Remove Removes the model outline from the report and removes the model from the Model List.

The outline for each model contains the following reports:

Formula Picture Shows the equation for the location parameter.

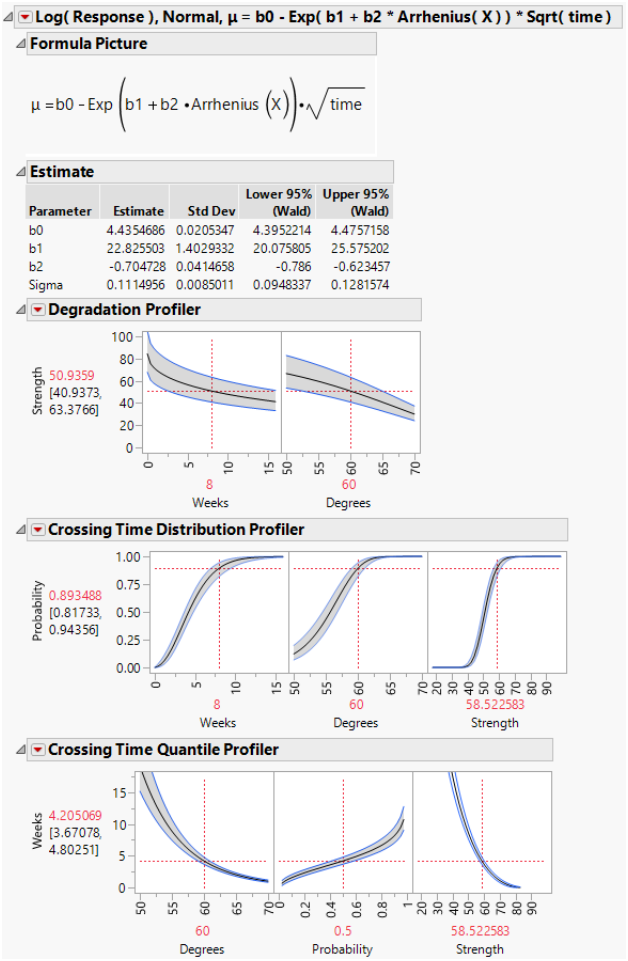
Estimate Shows parameter estimates and their standard deviations. This report also contains 95% Wald-based confidence intervals for the parameters.

Note: If you specify a reference temperature (X_0), this value is also displayed at the top of the Estimate report.

Distribution, Quantile, and Inverse Prediction For more information about the profilers, see "Profilers".

Residual Plots For more information about the residual plots, see "Residual Plots".

Figure 8.15 Reports within a Model Outline



Profilers

Three profilers appear in the model report.

Degradation Profiler

The Degradation Profiler shows a profiler view of the Degradation Data Analysis plot for the given model. The response is the degradation response Y. The profiler includes a plot against the Time variable and a plot against the optional explanatory variable X (if you have specified one). The plot against Time shows the median of the fitted distribution of Y as a solid curve. The dashed curves show the 0.025 and 0.975 percentiles of the fitted distribution of Y.

Crossing Time Distribution Profiler

Use the Crossing Time Distribution Profiler to determine the probability that the degradation measurement falls below a given threshold at some point in time.

The profiler plots the estimated cumulative distribution function of the response Y as a function of Time, the optional X variable, and Y. The plots for Time and Y show Wald confidence intervals.

Crossing Time Quantile Profiler

Use the Crossing Time Quantile Profiler to determine the time at which a specified proportion of measurements falls below a given threshold value.

The profiler plots the estimated Time as a function of the optional X variable, quantile values for Y (Probability), and Y. The plots for Probability and Y show Wald confidence intervals.

Residual Plots

Four residual plots appear in the model report. Use these plots to validate the distributional assumption for the model. For more information about the standardized residuals, see "Statistical Details for the Destructive Degradation Platform".

Cox-Snell Residual P-P Plot

If the points deviate far from the diagonal, then the distributional assumption might be violated. The Cox-Snell Residual P-P Plot red triangle menu has an option called Save Residuals that enables you to save the residual data to the data table. See Meeker and Escobar (1998, sec. 17.6.1) for a discussion of Cox-Snell residuals.

Probability Plot of Standardized Residuals

If the points deviate far from the diagonal, then the distributional assumption might be violated. The Probability Plot of Standardized Residuals red triangle menu has an option called Save Residuals that enables you to save the residual data to the data table.

Standardized Residuals versus Time

Use the Standardized Residuals versus Time plot to examine differences in variability over time. The Standardized Residuals versus Time red triangle menu has an option called Save Residuals that enables you to save the residual data to the data table.

Standardized Residuals versus Predicted

Use the Standardized Residuals versus Predicted plot to examine differences in variability over the range of predicted values. The Standardized Residuals versus Predicted red triangle menu has an option called Save Residuals that enables you to save the residual data to the data table.

Destructive Degradation Platform Options

The Degradation Data Analysis red triangle menu contains the following options:

Graph Options Provides options for modifying the platform graphs.

Shade Turns the shading on or off. By default, the upper and lower bounds of each shaded band at a given value of time correspond to the 0.025 and 0.975 quantiles of the distribution of Y.

Shade Coverage If shading is on, you can enter a proportion to increase or decrease the amount of shading coverage.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Statistical Details for the Destructive Degradation Platform

The destructive degradation model can be expressed as follows:

$$g(Y) \sim F(\mu, \sigma)$$

$$\mu = h(f(Time), X)$$

where:

- $g(Y)$ is the transformed Y variable
- F is the selected probability distribution
- μ is the location parameter, defined by h
- h is a function that relates the transformed Time variable and the explanatory variable X
- σ is the scale parameter of the distribution
- f(Time) is the transformed Time variable
- X is an optional explanatory variable

The standardized residuals are obtained as follows:

- For location-scale distributions, the standardized residuals are $(y\text{-}\mu)/\sigma$.
- For log-location-scale distributions, the standardized residuals are $(log(y)\text{-}\mu)/\sigma$.

Chapter 9

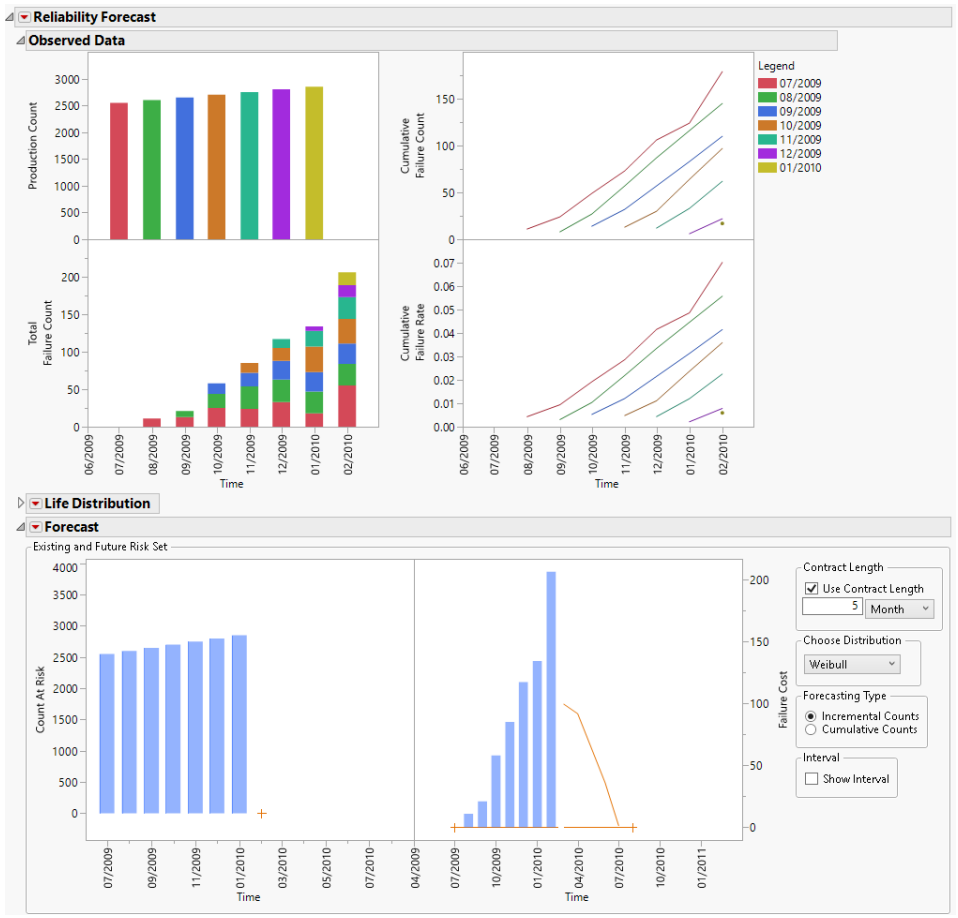
Reliability Forecast

Forecast Product Failure Using Production and Failure Data

The Reliability Forecast platform helps you predict the number of future failures. JMP estimates the parameters for a life distribution using production dates, failure dates, and production volume.

Using the interactive graphs, you can adjust factors such as future production volumes and contract length to estimate future failures. Repair costs can be incorporated into the analysis to forecast the total cost of repairs across all failed units.

Figure 9.1 Example of a Reliability Forecast



Contents

Overview of the Reliability Forecast Platform 233
Example Using the Reliability Forecast Platform 233
Launch the Reliability Forecast Platform 237
The Reliability Forecast Report 240
Observed Data Report 240
Life Distribution Report 240
Forecast Report 241
Reliability Forecast Platform Options 245
Additional Example Using the Reliability Forecast Platform 246

Overview of the Reliability Forecast Platform

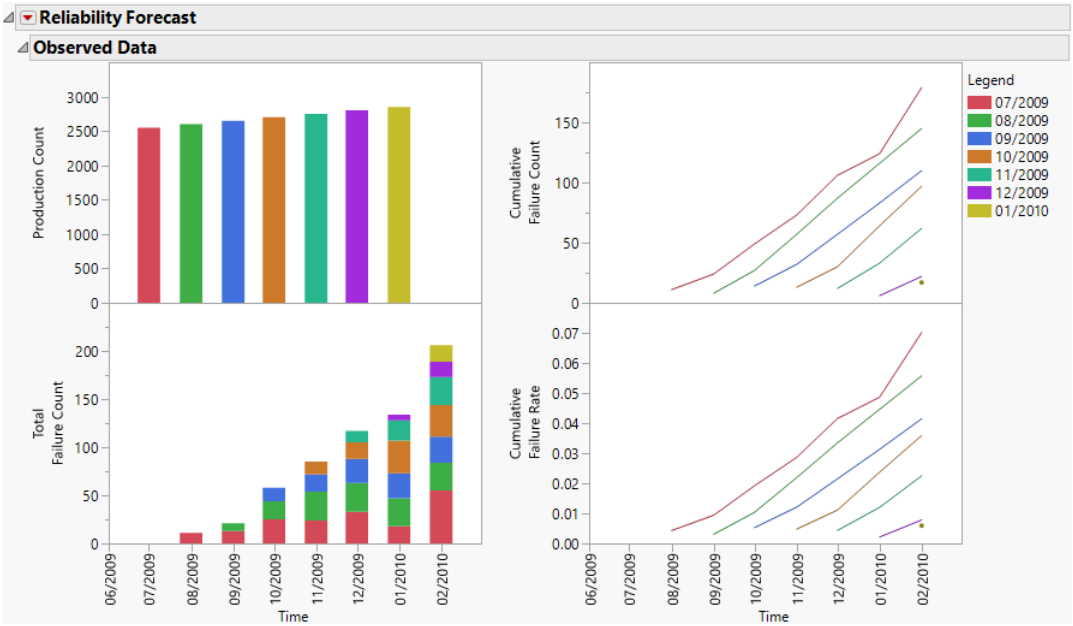
The Reliability Forecast platform helps you predict the number of future failures. JMP estimates the parameters for a life distribution using production dates, failure dates, and production volume. Using the interactive graphs, you can adjust factors such as future production volumes and contract length to estimate future failures. Repair costs can be incorporated into the analysis to forecast the total cost of repairs across all failed units.

Example Using the Reliability Forecast Platform

You have data on seven months of production and returns. You want to use this information to forecast the total number of units that will be returned for repair through February 2011. The product has a 12-month contract.

1. Select **Help > Sample Data Library** and open Reliability/Small Production.jmp.
2. Select **Analyze > Reliability and Survival > Reliability Forecast**.
3. On the **Nevada Format** tab, select Sold Quantity and click **Production Count**.
4. Select Sold Month and click **Timestamp**.
5. Select the other columns and click **Failure Count**.
6. Click **OK**.

Figure 9.2 Observed Data Report

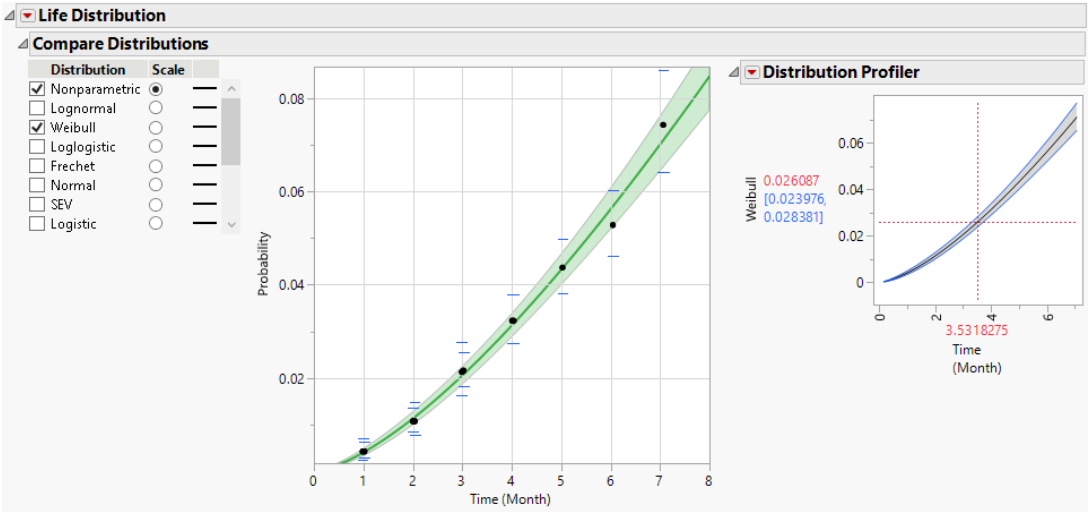


In the Observed Data report, the bottom left shows bar charts of previous failures. Cumulative failures are shown in the line graphs on the right. Note that production levels are fairly consistent. As production accumulates over time, more units are at risk of failure, so the cumulative failure rate gradually increases. The consistent production levels also result in similar cumulative failure rates and counts from month to month.

7. Click the **Life Distribution** disclosure icon.

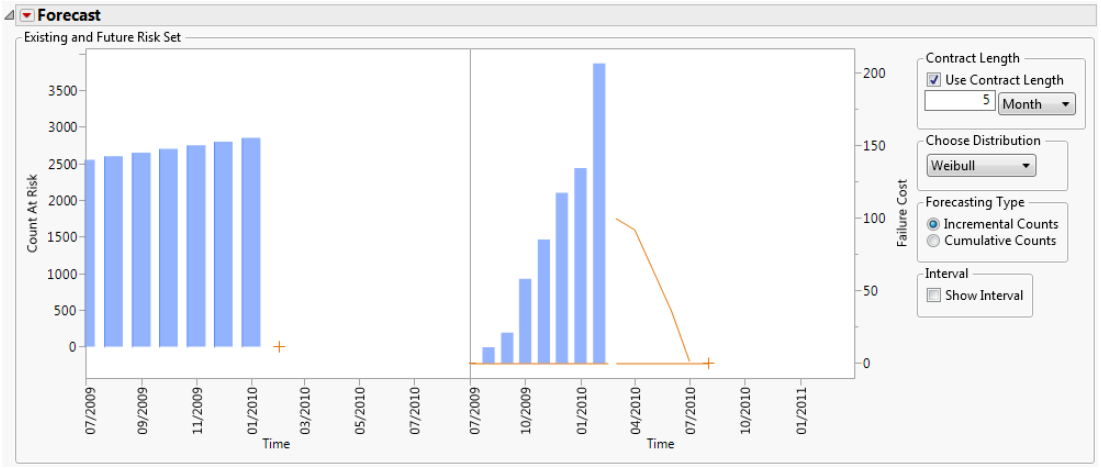
   JMP fits production and failure data with a Weibull distribution using the Life Distribution platform (Figure 9.3). JMP then uses the fitted Weibull distribution to forecast returns for the next five months (Figure 9.4).

Figure 9.3 Life Distribution Report



The Forecast report shows previous production in the left graph (Figure 9.4). In the right graph, you see that the number of previous failures increased steadily over time.

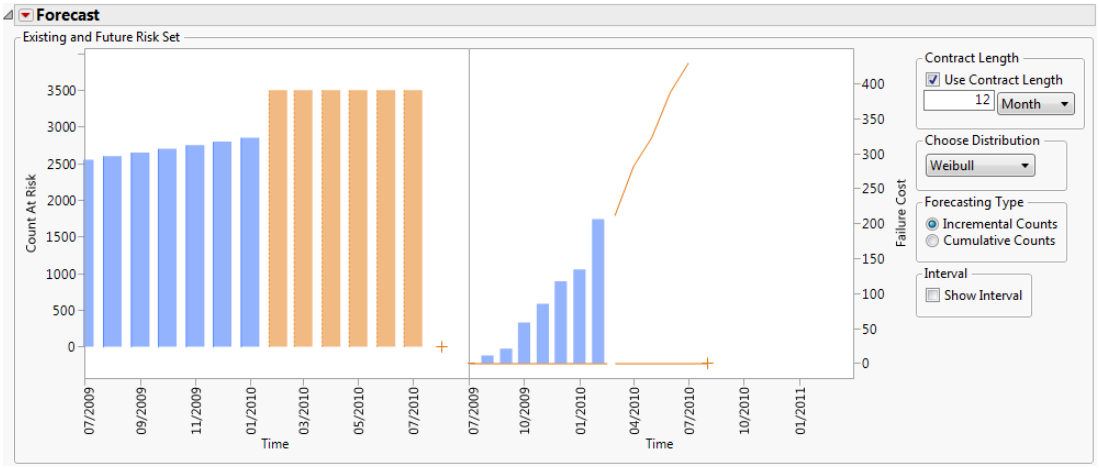
Figure 9.4 Forecast Report



8. In the Forecast report, type 12 for the Contract Length.
9. In the left Forecast graph, drag the animated hotspot over to July 2010 and upward to approximately 3500.

   Orange bars appear in the left graph to represent future production. The monthly returned failures in the right graph increase gradually through August 2010.

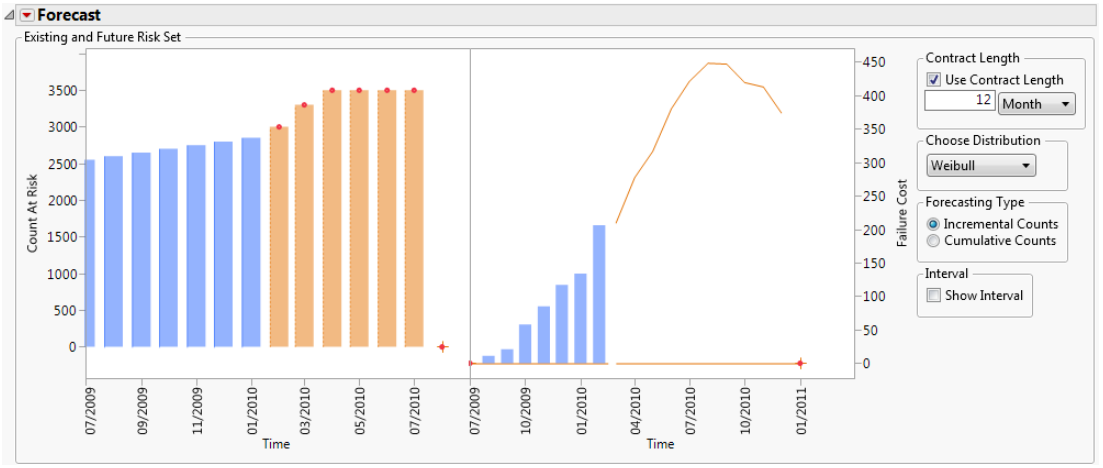
Figure 9.5 Production and Failure Estimates



10. In the left graph, drag the February 2010 hotspot to approximately 3000 and then drag the March 2010 hotspot to approximately 3300.
11. In the right graph, drag the right hotspot to February 2011.

    JMP estimates that the number of returns will gradually increase through August 2010 and decrease by February 2011.

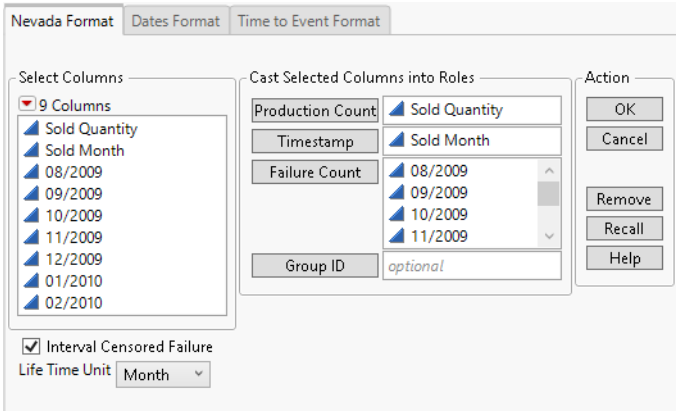
Figure 9.6 Future Production Counts and Forecasted Failures



Launch the Reliability Forecast Platform

Launch the Reliability Forecast platform by selecting **Analyze > Reliability and Survival > Reliability Forecast**.

Figure 9.7 The Reliability Forecast Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The launch window includes a tab for each contract data format: Nevada, Dates, and Time to Event (for time to failure data). The following sections describe these formats.

Nevada Format

Contract data is commonly stored in the Nevada format: shipment or production dates and failure counts within specified periods are shaped like the state of Nevada. Figure 9.8 shows the Small Production.jmp sample data table.

Figure 9.8 Example of the Nevada Format

	Sold Quantity	Sold Month	08/2009	09/2009	10/2009	11/2009	12/2009	01/2010	02/2010
1	2550	07/2009	11	13	25	24	33	18	55
2	2600	08/2009	0	8	19	30	30	29	29
3	2650	09/2009	0	0	14	18	25	26	27
4	2700	10/2009	0	0	0	13	17	34	33
5	2750	11/2009	0	0	0	0	12	21	29
6	2800	12/2009	0	0	0	0	0	6	16
7	2850	01/2010	0	0	0	0	0	0	17

The Nevada Format tab contains the following options:

Interval Censored Failure Considers the returned quantity to be interval censored. The interval is between the last recorded time and the time that the failure was observed. This option is selected by default.

Life Time Unit Physical date-time format of all time stamps, including column titles for return counts. The platform uses this setting in forecasting step increments.

Production Count Column that identifies the number of units produced.

Timestamp Column that contains the production date.

Failure Count Column that contains the number of failed units.

Group ID Column by which observations are grouped. Each group has its own distribution fit and forecast. A combined forecast is also included.

Dates Format

The Dates format focuses on production and failure dates. One data table specifies the production counts for each time period. The other table provides failure dates, failure counts, and the corresponding production times of the failures.

Figure 9.9 shows the Small Production part1.jmp and Small Production part2.jmp sample data tables.

Figure 9.9 Example of the Dates Format

production data

	Sold Quantity	Sold Month
1	2550	07/2009
2	2600	08/2009
3	2650	09/2009
4	2700	10/2009
5	2750	11/2009
6	2800	12/2009
7	2850	01/2010

failure data

	Return Month	Return Quantity	Sold Month
1	08/2009	11	07/2009
2	09/2009	13	07/2009
3	10/2009	25	07/2009
4	11/2009	24	07/2009
5	12/2009	33	07/2009
6	01/2010	18	07/2009
7	02/2010	55	07/2009

The Dates Format tab is divided into Production Data and Failure Data sections.

Production Data

Select Table Select the table that contains the number of units and the production dates.

Failure Data

Select Table Select the table that contains failure data, such as the number of failed units, production dates, and failure dates.

Left Censor Column that identifies censored observations.

Timestamp Column that links production observations to failure observations, indicating which batch a failed unit came from.

Censor Code Value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

Other options are identical to those on the Nevada Format tab. See "Nevada Format".

Time to Event Format

The Time to Event format shows production and failure data. Unlike the Nevada and Dates formats, Time to Event data does not include date-time information, such as production or failure dates. This format can also accommodate arbitrary censoring schemes in the data. See "Defining Risk Sets with Time to Event Data".

Figure 9.10 shows the Small Production Time to Event.jmp sample data table. For an example that uses time-to-event data, see "Additional Example Using the Reliability Forecast Platform".

Figure 9.10 Example of the Time to Event Format

start time | end time | failure counts

	Time (Month)	Time Right	Freq	Timestamp
1	0	1.0184804928	11	07/2009
2	1.0184804928	2.0369609856	13	07/2009
3	2.0369609856	3.022587269	25	07/2009
4	3.022587269	4.0410677618	24	07/2009
5	4.0410677618	5.0266940452	33	07/2009
6	5.0266940452	6.045174538	18	07/2009
7	6.045174538	7.0636550308	55	07/2009
8	7.0636550308	•	2371	07/2009

The Time to Event Format tab contains the following options:

Forecast Start Time Time at which the forecast begins. Enter the first value that you want on the horizontal axis. To enable forecasting, you must select Numeric for the Life Time Unit and set the Forecast Start Time to zero.

Censor Code Code for censored observations. Available only when you assign a **Censor** variable.

Other options are identical to those on the Nevada Format tab. See "Nevada Format".

The Reliability Forecast Report

The Reliability Forecast report contains the Observed Data report, Life Distribution report, and Forecast report. Use these reports to view the current data, compare distributions to find the right fit, and then adjust factors that affect forecasting. You can also save the forecast to a data table for use in financial forecasts. See Figure 9.1 for an example of the report.

The Reliability Forecast red triangle menu contains options for filtering the observed data by date and saving the data in another format. See "Reliability Forecast Platform Options".

Observed Data Report

The Observed Data report gives you a quick view of Nevada and Dates data (Figure 9.11).

- Bar charts show the production and failure counts for the specified production periods.
- Line charts show the cumulative forecast by production period.

Note: Time-to-event data does not include date-time information, such as production or failure dates, so the platform does not create an Observed Data report for this format.

Life Distribution Report

The Life Distribution report lets you compare distributions and work with profilers to find the right fit. When you select a distribution in the Forecast report, the Life Distribution report is updated. For more information about this report, see "Life Distribution". (Some of the options described in the Life Distribution chapter are not available in the Life Distribution report in the Reliability Forecast platform.)

Forecast Report

The Forecast report provides interactive graphs that help you forecast failures. By dragging hotspots, you can add anticipated production counts and see how they affect the forecast.

Adjusting Risk Sets

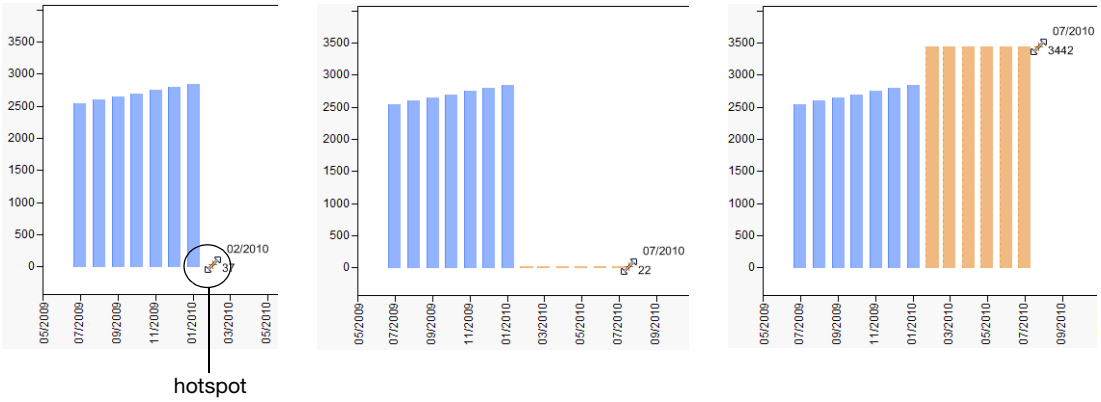
Future Production

In the left graph, the blue bars represent previous production counts. To add anticipated production, follow these steps:

1. Drag a hotspot to the right to add one or more production periods.

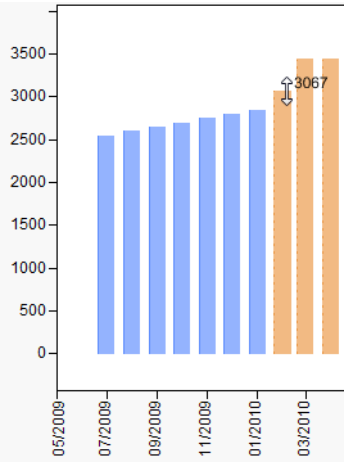
   The orange bars represent future production.

Figure 9.11 Add Production Periods



2. Drag each bar upward or downward to change the production count for each period.

Figure 9.12 Adjust Production Counts



Tip: To adjust future production counts, click the Forecast red triangle and select **Spreadsheet Configuration of Risk Sets**. See "Spreadsheet Configuration of Risk Sets".

Existing Production

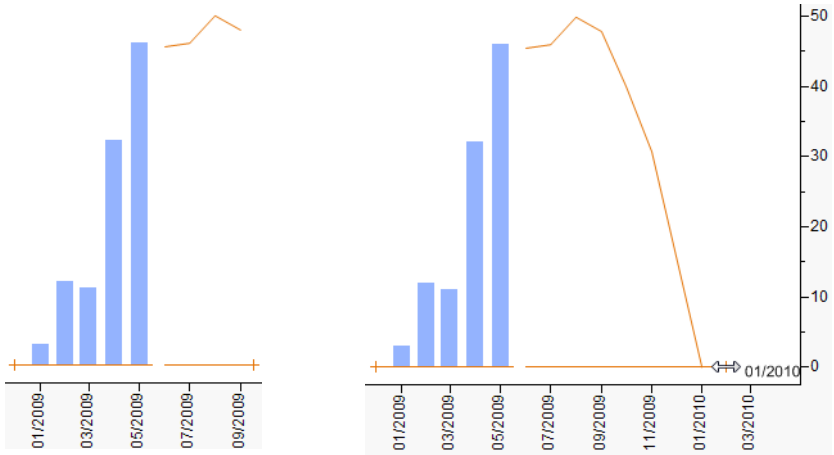
To remove a risk set from the forecast results, right-click a blue bar and select **Exclude**. To return that data to the risk set, right-click and select **Include**.

Tip: To adjust existing production counts, click the Forecast red triangle and select **Spreadsheet Configuration of Risk Sets**. See "Spreadsheet Configuration of Risk Sets".

Forecasting Failures

When you adjust production in the left graph, the right graph is updated to estimate future failures (Figure 9.13). Dragging a hotspot lets you change the forecast period. The orange line then shortens or lengthens to show the estimated failure counts.

Figure 9.13 Adjust the Forecast Period



Defining Risk Sets with Time to Event Data

You can obtain forecasts for arbitrary risk sets using the Time to Event Format tab. In the launch window's Time to Event Format tab, you must select Numeric for the Life Time Unit and set the Forecast Start Time to zero. Enter appropriate columns for Time to Event, Censor, Freq, and Group ID.

The plot on the left provides locations for bars representing existing production. Drag the hotspot at 0 to the left to create existing risk sets. Drag the hotspot at 1 to the right to create future production risk sets.

You can specify existing and future risk sets using the Spreadsheet Configuration of Risk Sets option. For time-to-event data, you must enter negative time values for the existing risk set in the Future Risk area. See "Spreadsheet Configuration of Risk Sets".

For an example that uses time-to-event data, see "Additional Example Using the Reliability Forecast Platform".

Forecast Graph Options

To explore your data further, you can change the contract length, distribution type, and other options.

- To forecast failures for a different contract period, enter the number next to **Use Contract Length**. Change the time unit if necessary.
- To change the distribution fit, select a distribution from the **Choose Distribution** list. The distribution is then fit to the future graph of future risk. The distribution fit appears in the Life Distribution report plot, and a new profiler is added. Changing the distribution fit in the Life Distribution report does not change the fit in the Forecast graph.

- If you are more interested in the total number of failures over time, select **Cumulative Counts**. Otherwise, JMP shows failures incrementally, which can make trends easier to identify.
- To show 95% confidence limits for the anticipated number of failures, select **Show Interval**.

Forecast Report Options

The Forecast red triangle menu contains the following options:

Animation Controls the flashing of the hotspots. You can also right-click a blue bar in the existing risk set and select or deselect **Animation**.

Interactive Configuration of Risk Sets Determines whether you can drag hotspots in the graphs.

Spreadsheet Configuration of Risk Sets Lets you specify production counts and periods (rather than adding them to the interactive graphs). You can also exclude production periods from the analysis.

- To remove an existing time period from analysis, highlight the period in the Existing Risk area, click, and then select **Exclude**. Or select **Include** to return the period to the forecast.
- To edit production, double-click in the appropriate Future Risk field and enter the new values.
- To add a production period to the forecast, right-click in the Future Risk area and select one of the **Append** options. (**Append Rows** adds one row; **Append N Rows** lets you specify the number of rows.)

As you change these values, the graphs update accordingly.

Note: If you launched the platform using the Time to Event Format tab, values for the existing risk set must be entered in the Future Risk area using negative time values.

Import Future Risk Set Lets you import future production data from another open data table. The new predictions then appear in the future risk graph. The imported data must have a column for timestamps and for quantities.

Show Interval Shows or hides 95% confidence limits in the graph. This option works the same as selecting **Show Interval** next to the graphs.

After you select **Show Interval**, the **Forecast Interval Type** option appears in the menu. Select one of the following interval types:

Plugin Interval Considers only forecasting errors given a fixed distribution.

Prediction Interval Considers forecasting errors when a distribution is estimated with estimation errors (for example, with a non-fixed distribution).

If you select Prediction Interval, the **Prediction Interval Settings** option appears in the menu. Approximate intervals are initially shown in the graph. Select **Monte Carlo Sample Size** or **Random Seed** to specify those values instead. To use the system clock, enter a missing number.

Use Contract Length Determines whether the specified contract length is considered in the forecast. This option works the same as selecting **Use Contract Length** next to the graphs.

Use Failure Cost Shows failure cost instead of the failure count in the future risk graph.

After you select **Use Failure Cost**, the **Set Failure Cost** option appears in the menu. This option enables you to set a cost for each failure. If you specified a Group variable in the launch window, the Set Failure Cost window enables you to specify separate costs for failures in each group.

Save Forecast Data Table Saves the cumulative and incremental number of returns in a new data table, along with the variables that you selected in the launch window. For grouped analyses, table names include the group ID and the word "Aggregated". Existing returns are also included in the aggregated data tables.

Reliability Forecast Platform Options

The Reliability Forecast red triangle menu contains the following options:

Save Data in Time to Event Format Saves Nevada or Dates data in a Time to Event formatted table.

Show Legend Shows or hides a legend for the Observed Data report. Unavailable for Time to Event data.

Show Graph Filter Shows or hides the Graph Filter so that you can select which production periods to show in the Observed Data graphs. Bars fade for deselected periods. Deselect the periods to show the graph in its original state. Unavailable for Time to Event data.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Additional Example Using the Reliability Forecast Platform

You have seven months of production and returns data that are stored in time-to-event format. You want to use this information to forecast the total number of units that will be returned for repair in the next 6 months. The product has a 4-month contract.

1. Select **Help > Sample Data Library** and open Reliability/Small Production Time to Event.jmp.
2. Select **Analyze > Reliability and Survival > Reliability Forecast**.
3. On the **Time to Event Format** tab, select Time (Month) and Time Right. Click **Time to Event**.
4. Select Freq and click **Freq**.
5. Click **OK**.
6. In the Forecast report, type 4 for the Contract Length.
7. (Optional.) Click the **Life Distribution** disclosure icon.

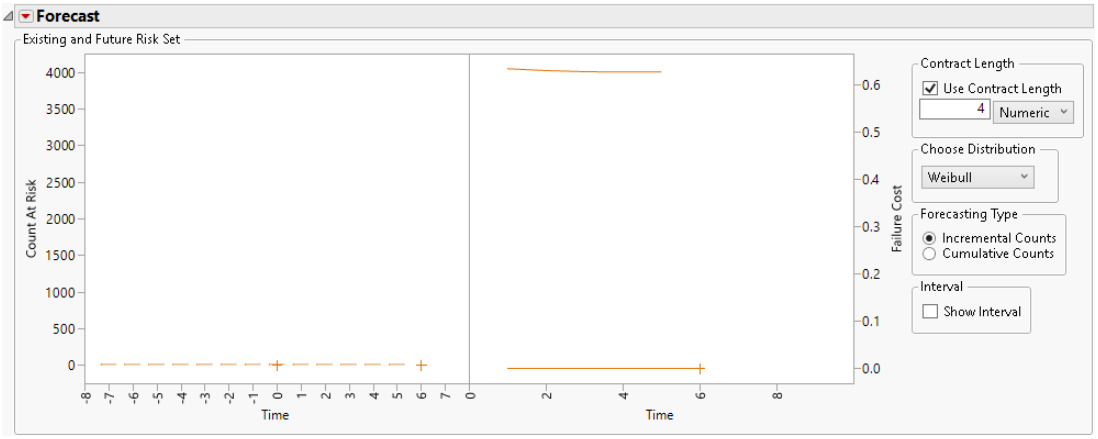
   The Life Distribution report shows a Weibull fit for the production and failure data. The Reliability Forecast platform then uses the fitted Weibull distribution to forecast returns.

Specify the Current and Future Production Counts

8. Set the vertical axis settings to prepare for past and future production counts. Right-click the vertical axis in the left Forecast graph and select **Axis Settings**. Specify the following axis settings:
   1. Type -250 for the Minimum.
   2. Type 4250 for the Maximum.
   3. Type 500 for the Increment.
   4. Click **OK**.
9. Set the horizontal axis to cover the previous 8 and next 8 months. Right-click the horizontal axis in the left Forecast graph and select **Axis Settings**. Specify the following axis settings:
   1. Type -8 for the Minimum.
   2. Type 8 for the Maximum.
   3. Click **OK**.

10. Set the past 7 months of production counts. In the left Forecast graph, drag the leftmost animated hotspot left to -7.
11. Set the future 5 months of production estimates. In the left Forecast graph, drag the rightmost animated hotspot right to 5.

Figure 9.14 Risk Set after Dragging Animated Hotspots



12. In the Forecast red triangle menu, select **Spreadsheet Configuration of Risk Sets**.
13. Drag the bottom of the Future Risk panel down so that you can see all 13 rows of the Future Risk table.

> **Note:** When using time-to-event formatted data, the existing production quantities must be specified in the Future Risk table using negative numeric time values.

14. Type in the Count values that are shown in Figure 9.15.

Figure 9.15 Future Risk Count Specifications

Future Risk

	Time	Count
1	-7	2550
2	-6	2600
3	-5	2650
4	-4	2700
5	-3	2750
6	-2	2800
7	-1	2850
8	0	3022
9	1	3307
10	2	3502
11	3	3502
12	4	3502
13	5	3502

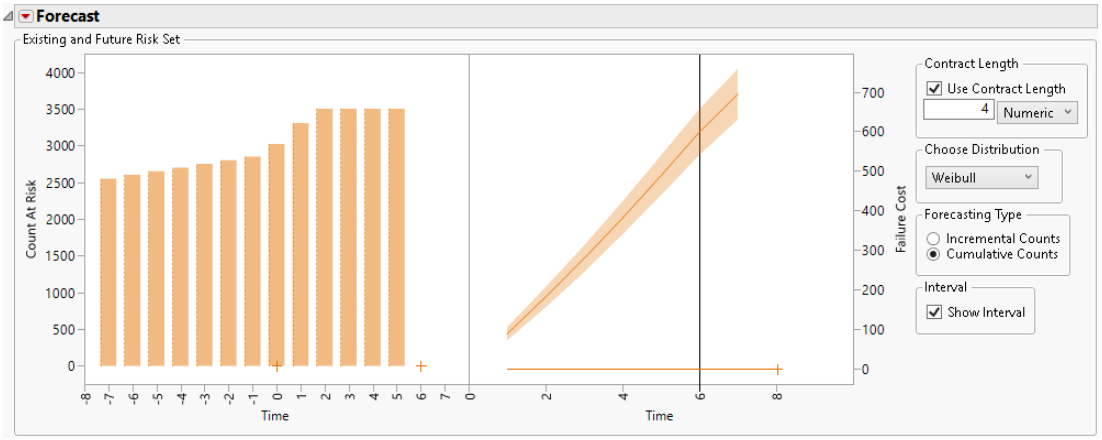
optional

15. In the right Forecast graph, drag the animated hotspot right to 8.
16. In the Forecasting Type panel, select **Cumulative Counts**.
17. In the Interval panel, select **Show Interval**.
18. Right-click the horizontal axis in the right Forecast graph and select **Axis Settings**. Add a reference line at month 6 using the following steps:
    1. Type 6 for the Value.
    2. Click **Add**.
    3. Click **OK**.

The forecast for the total number of units that will be returned for repair in the next 6 months is approximately 600 with confidence interval about 550 to 650.

Tip: You can use the **Save Forecast Data Table** option in the Forecast red triangle menu to see exact values for the forecasts and intervals.

Figure 9.16 Forecast Results



Chapter 10

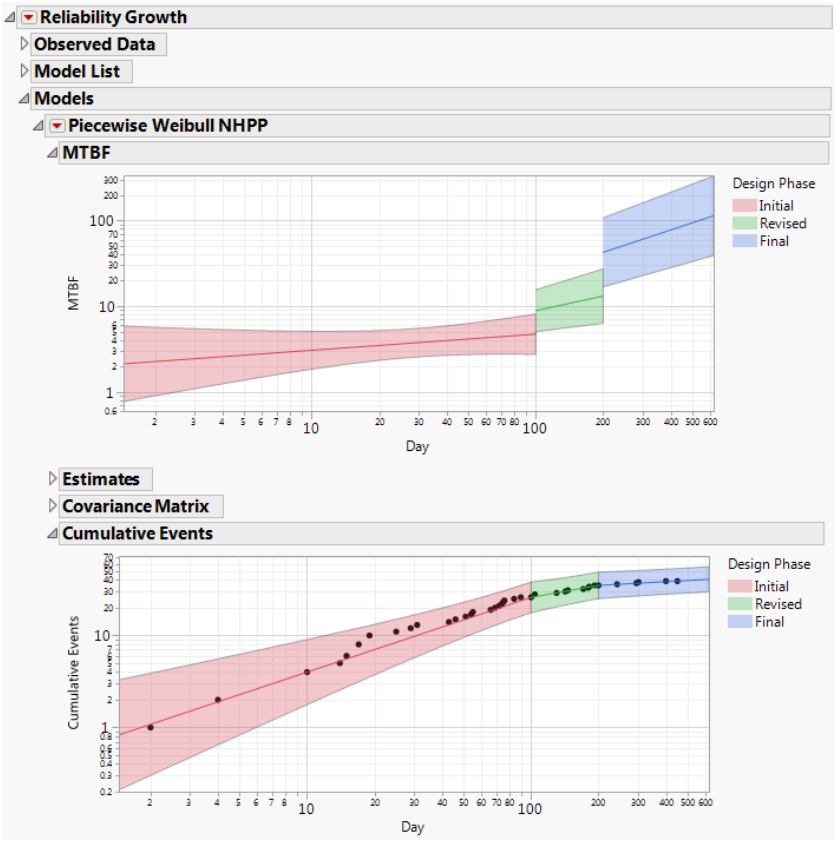
Reliability Growth

Model System Reliability as Changes Are Implemented

The Reliability Growth platform models the change in reliability of a single repairable system over time as improvements are incorporated into its design. A reliability growth testing program attempts to increase the system's mean time between failures (MTBF) by integrating design improvements as failures are discovered.

The Reliability Growth platform fits Crow-AMSAA models. These are non-homogeneous Poisson processes with Weibull intensity functions. Separate models can accommodate various phases of a reliability growth program. The platform also fits models for multiple systems.

Figure 10.1 Example of Plots for a Three-Phase Reliability Growth Model



Contents

Overview of the Reliability Growth Platform 251
Example Using the Reliability Growth Platform 252
Launch the Reliability Growth Platform 256
Launch Window Roles 257
Data Table Structure 258
The Reliability Growth Report 261
Model Descriptions and Feasibilities 261
Observed Data Report 261
Reliability Growth Platform Options 264
Model Reports 265
Crow AMSAA 266
Crow AMSAA with Modified MLE 272
Fixed Parameter Crow AMSAA 273
Piecewise Weibull NHPP 274
Reinitialized Weibull NHPP 278
Piecewise Weibull NHPP Change Point Detection 281
Additional Examples of the Reliability Growth Platform 282
Piecewise NHPP Weibull Model Fitting with Interval-Censored Data 282
Piecewise Weibull NHPP Change Point Detection with Time in Dates Format 285
Statistical Details for the Reliability Growth Platform 287
Statistical Details for the Crow-AMSAA Report 287
Statistical Details for the Piecewise Weibull NHPP Change Point Detection Report 288

Overview of the Reliability Growth Platform

The Reliability Growth platform fits Crow-AMSAA models, described in MIL-HDBK-189 (1981). A Crow-AMSAA model is a non-homogeneous Poisson process (NHPP) model with Weibull intensity; it is also known as a power law process. Such a model allows the failure intensity to vary over time. The failure intensity is defined by the two parameters beta and lambda.

For single-prototype data, the platform fits four classes of models and performs automatic change-point detection. The following reports are available:

- Simple Crow-AMSAA model, where both parameters are estimated using maximum likelihood.
- Crow-AMSAA with Modified MLE, where the maximum likelihood estimate for beta is corrected for bias.
- Fixed Parameter Crow-AMSAA model, where the user is allowed to fix either or both parameters.
- Piecewise Weibull NHPP model, where the parameters are estimated for each test phase, taking failure history from previous phases into account.
- Reinitialized Weibull NHPP model, where both parameters are estimated for each test phase in a manner that ignores failure history from previous phases.
- Automatic estimation of a change-point and the associated piecewise Weibull NHPP model, for reliability growth situations where different failure intensities can define two distinct test phases.

For multiple-prototype data, the platform fits the following classes of models:

- Piecewise Weibull NHPP model, where each system in a multi-phase study follows the same piecewise Weibull NHPP model. The differences among the systems are assumed to be due to the randomness of individual realizations of the same model. This model contains one beta parameter for each phase and one lambda parameter.
- Piecewise Weibull NHPP with Different Intercepts model, where each system in a multi-phase study follows a separate piecewise Weibull NHPP model. This model contains one beta parameter for each phase and one lambda parameter for each system.
- Distinct Phase Weibull NHPP model, where each system in a multi-phase study follows the same Crow-AMSAA model in each phase. This model contains one beta parameter and one lambda parameter for each phase.
- Distinct Weibull NHPP model, where each system in a multi-phase study follows a separate Crow-AMSAA model in each phase. This model contains one beta parameter and one lambda parameter for each combination of system and phase in the study.

- Distinct System Weibull NHPP model, where each system in the study follows a separate Crow-AMSAA model with different parameters.
- Identical System Weibull NHPP model, where each system in the study follows a single Crow-AMSAA model. The differences among the systems are assumed to be due to the randomness of individual realizations of the same model.

Interactive profilers enable you to explore changes in MTBF, failure intensity, and cumulative failures over time. When you suspect a change in intensity over the testing period, you can use the change-point detection option to estimate a change-point and its corresponding model.

Example Using the Reliability Growth Platform

Suppose that you are testing a prototype for a new class of turbine engines. The testing program has been ongoing for over a year and has been through three phases.

The data are given in the TurbineEngineDesign1.jmp sample data table, found in the Reliability subfolder. For each failure that occurred, the number of days since test initiation was recorded in the column Day. The number of failures on a given day, or equivalently, the number of required fixes, was recorded in the column Fixes.

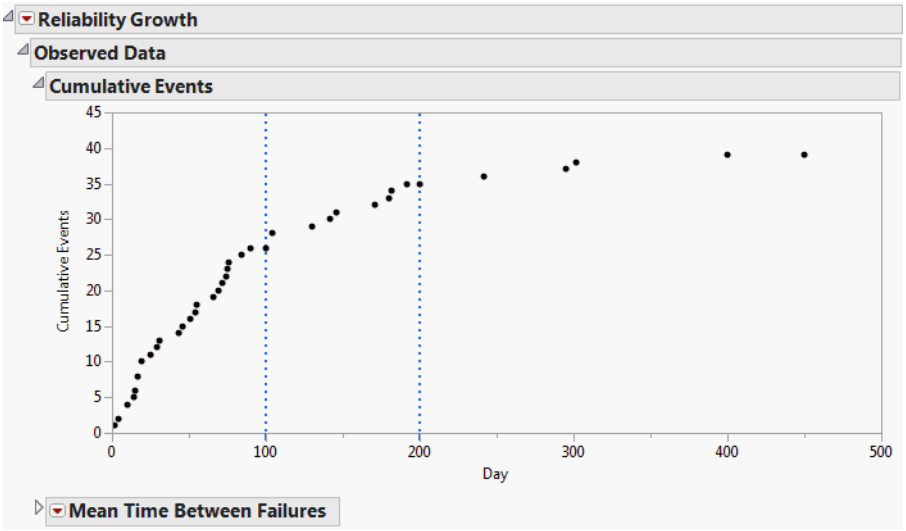
The first 100-day phase of the program was considered the initial testing phase. Failures were addressed with aggressive design changes, resulting in a substantially revised design. This was followed by another 100-day phase, during which failures in the revised design were addressed with design changes to subsystems. The third and final testing phase ran for 250 days. During this final phase, failures were addressed with local design changes.

Each phase of the testing was terminated based on the designated number of days, so that the phases are time terminated. Specifically, a given phase is time terminated at the start time of the next phase. However, the failure times are exact (not censored).

1. Select **Help > Sample Data Library** and open Reliability/TurbineEngineDesign1.jmp.
2. Select **Analyze > Reliability and Survival > Reliability Growth**.
3. On the **Time to Event Format** tab, select Day and click **Time to Event**.
4. Select Fixes and click **Event Count**.
5. Select Design Phase and click **Phase**.
6. Click **OK**.

   The Reliability Growth report appears. The Cumulative Events plot shows the cumulative number of failures by day. Vertical dashed blue lines show the two transitions between the three phases.

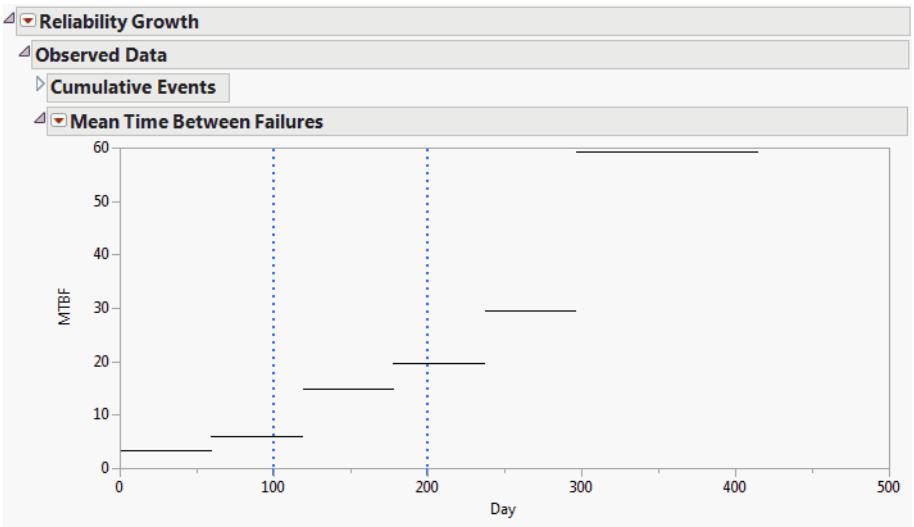
Figure 10.2 Observed Data Report



7. Click the **Mean Time Between Failures** disclosure icon.

   This provides a plot with horizontal lines at the mean times between failures computed over intervals of a predetermined size. An option in the red triangle menu enables you to specify the interval size.

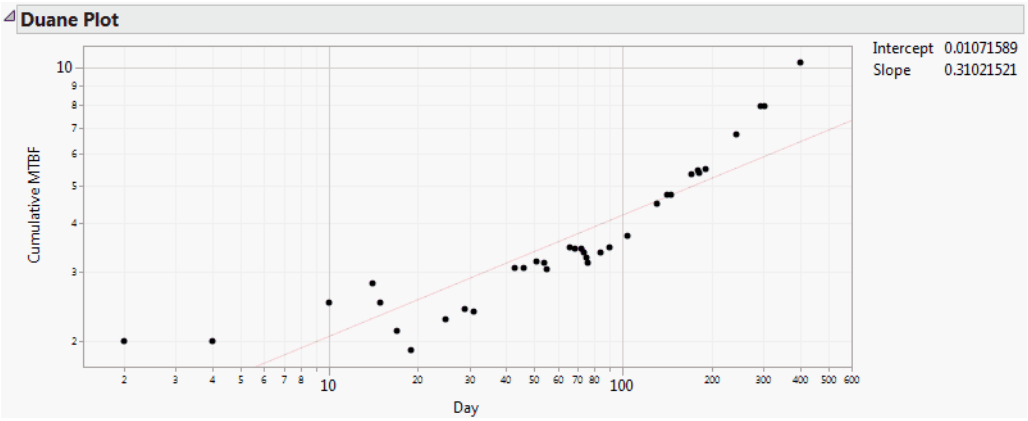
Figure 10.3 Mean Time between Failures Plot



8. Click the **Duane Plot** disclosure icon.

This provides a plot that displays the Cumulative MTBF estimates on the vertical axis versus the time to event variable on the horizontal axis. If the data follow the Crow-AMSAA model, the points should fall along a line when plotted on log-log paper.

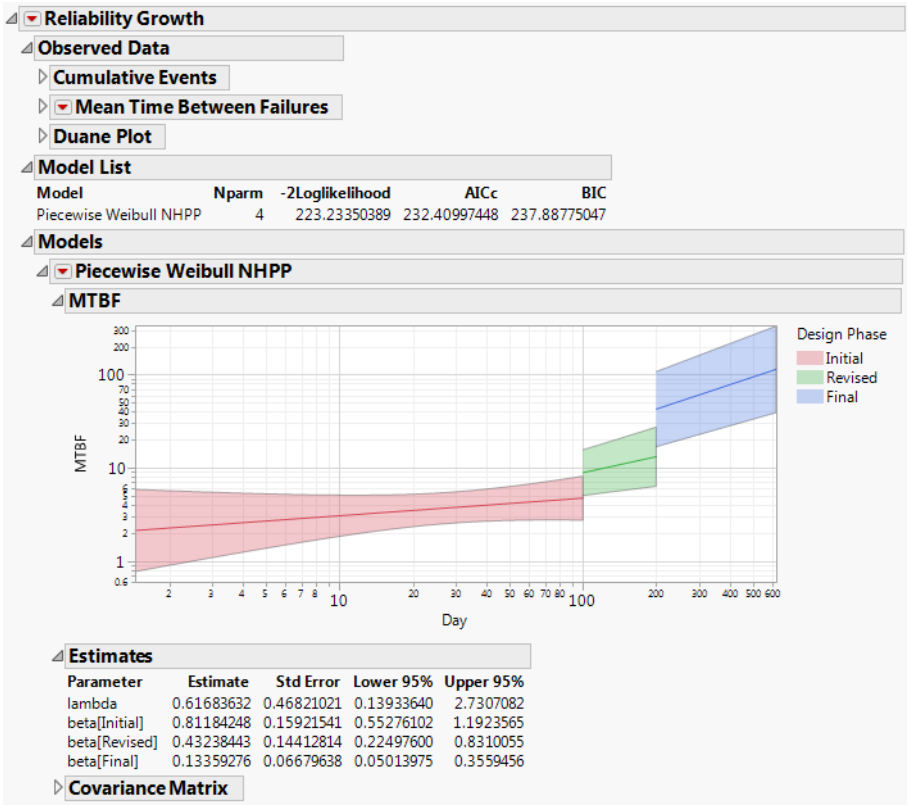
Figure 10.4 Duane Plot



9. Click the Reliability Growth red triangle and select **Fit Model > Piecewise Weibull NHPP**.

   This fits Weibull NHPP models to the three phases of the testing program, treating these phases as multiple stages of a single reliability growth program (Figure 10.5). Options in the Piecewise Weibull NHPP red triangle menu provide various other plots and reports.

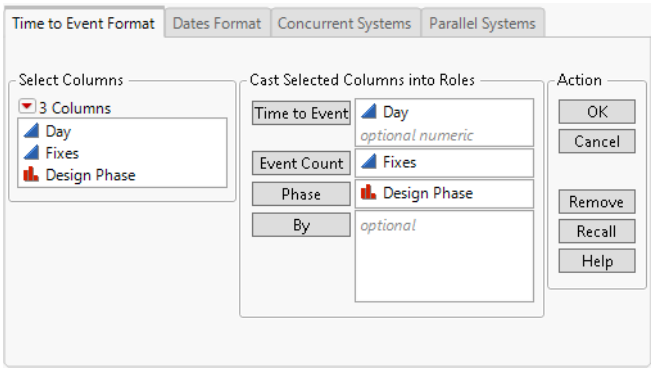
Figure 10.5 Piecewise Weibull NHPP Report



Launch the Reliability Growth Platform

Launch the Reliability Growth platform by selecting **Analyze > Reliability and Survival > Reliability Growth**. The launch window, using data from TurbineEngineDesign1.jmp, is shown in Figure 10.6.

Figure 10.6 Reliability Growth Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The launch window includes a tab for each of four data formats: Time to Event Format, Dates Format, Concurrent Systems, and Parallel Systems.

- The Dates Format assumes that time is recorded in a date/time format, indicating an absolute date or time. On the Dates Format tab, the time column or columns are given the Timestamp role.
- The other three data formats assume that time is recorded as the number of time units; for example, days or hours, since initial start-up of the system. The test start time is assumed to be time zero. On the Time to Event Format tab, the time column or columns are given the Time to Event role.

Rows with missing data in the columns Time to Event, Timestamp, Event Count, Phase, or System ID are not included in the analysis.

Note: All data formats for the Reliability Growth platform require that times or time intervals be specified in non-decreasing order.

Launch Window Roles

The roles that are available for variables in the Reliability Growth launch window are determined by the specified data format tab. The roles are explained in this section.

Time to Event

Time to Event is the number of time units that elapse between the start of the test and the occurrence of an event (a failure or test termination). The test start time is assumed to be time zero. Note that the Time to Event role is available only on the Time to Event Format tab.

Two conventions are allowed (See "Exact Failure Times versus Interval Censoring".):

- A single column can be entered. In this case, it is assumed that the column gives the exact elapsed times at which events occurred.
- Two columns can be entered, giving interval start and end times. If an interval's start and end times differ, it is assumed that the corresponding events given in the Event Count column occurred at some unknown time within that interval. We say that the data are interval-censored. If the interval start and end times are identical, it is assumed that the corresponding events occurred at that specific time point, so that the times are exact (not censored).

The platform requires that the time columns be sorted in non-decreasing order. When two columns giving interval start and end times are provided, these intervals must not overlap (except at their endpoints). Intervals with zero event counts that fall strictly within a phase can be omitted, as they do not affect the likelihood function.

Timestamp

Timestamp is an absolute time (for example, a date). As with Time to Event, Timestamp allows times to be entered using either a single column or two columns. Note that the Timestamp role is available only on the Dates Format tab.

For times entered as Timestamp, the first row of the table is considered to give the test start time:

- When a single column is entered, the timestamp corresponding to the test start, with an event count of 0, should appear in the first row.
- When two columns, giving time intervals, are entered, the first entry in the first column should be the test start timestamp. See "Phase".

Other details are analogous to those described for Time to Event Format in the section "Time to Event". See also "Exact Failure Times versus Interval Censoring".

Event Count

Event Count is the number of events, usually failures addressed by corrective actions (fixes), occurring at the specified time or within the specified time interval. If no column is entered as Event Count, it is assumed that the Event Count for each row is one.

System ID

System ID identifies the prototype for an observation for multiple-prototype data. The System ID role is available only in the Concurrent Systems and Parallel Systems data formats. It is a required role for both data formats.

Phase

Reliability growth programs often involve several periods, or phases, of active testing. These testing phases can be specified in the optional Phase column. For the Time to Event Format and Dates Format data formats, the Phase variable can be of any data or modeling type. For the Parallel Systems data format, the data type of the Phase variable must be Numeric. For more information about structuring multi-phase data, see "Test Phases". For an example, see "Piecewise NHPP Weibull Model Fitting with Interval-Censored Data".

By

This produces a separate analysis for each value that appears in the column.

Data Table Structure

The Time to Event Format and the Dates Format enable you to enter either a single column *or* two columns as Time to Event or Timestamp, respectively. The Concurrent Systems and the Parallel Systems enable you to enter one column corresponding to each level of the System ID variable. There are examples of multiple-prototype data tables in the Reliability folder of the Sample Data folder: Concurrent Systems.jmp for Concurrent Systems and four tables with the prefix Parallel Systems for Parallel Systems.

This section describes how to use these two approaches to specify the testing structure.

Exact Failure Times versus Interval Censoring

In some testing situations, the system being tested is checked periodically for failures. In this case, failures are known to have occurred within time intervals, but the precise time of a failure is not known. We say that the failure times are *interval-censored*.

The Reliability Growth platform accommodates both exact, non-censored failure-time data, and interval-censored data. When a single column is entered as Time to Event or Timestamp, the times are considered exact times (not censored).

When two columns are entered, the platform views these as defining the start and end points of time intervals. If an interval's start and end times differ, then the times for failures occurring within that interval are considered to be interval-censored. If the end points are identical, then the times for the corresponding failures are assumed to be exact and equal to that common time value. So, you can represent both exact and interval-censored failure times by using two time columns.

In particular, *exact failures times* can be represented in one of two ways: As times given by a single time column, or as intervals with identical endpoints, given by two time columns.

Model-fitting in the Reliability Growth platform relies on the likelihood function. The likelihood function takes into account whether interval-censoring is present or not. So, mixing interval-censored with exact failure times is permitted.

Failure and Time Termination

A test plan can call for test termination once a specific number of failures has been achieved or once a certain span of time has elapsed. For example, a test plan might terminate testing after 50 failures occur. Another plan might terminate testing after a six-month period.

If testing terminates based on a specified number of failures, we say that the test is *failure terminated*. If testing is terminated based on a specified time interval, we say that the test is *time terminated*. The likelihood functions used in the Reliability Growth platform reflect whether the test phases are failure or time terminated.

Test Phases

Reliability growth testing often involves several *phases* of testing. For example, the system being developed or the testing program might experience substantial changes at specific time points. The data table conveys the start time for each phase and whether each phase is failure or time terminated, as described below.

Single Test Phase

When there is a single test phase, the platform infers whether the test is failure or time terminated from the time and event count entries in the last row of the data table.

- If the last row contains an exact failure time with a nonzero event count, the test is considered failure terminated.
- If the last row contains an exact failure time with a zero event count, the test is considered time terminated.

- If the last row contains an interval with nonzero width, the test is considered time terminated with termination time equal to the right endpoint of that interval.

Note: To indicate that a test has been time terminated, be sure to include a last row in your data table showing the test termination time. If you are entering a single column as Time to Event or Timestamp, the last row should show a zero event count. If you are entering two columns as Time to Event or Timestamp, the right endpoint of the last interval should be the test-termination time. In this case, if there were no failures during the last interval, you should enter a zero event count.

Multiple Test Phases

When using Time to Event Format, the start time for any phase other than the first should be included in the time column(s). When using Dates Format, the start times for all phases should be included in the time column(s). If no events occurred at a phase start time, the corresponding entry in the Event Count column should be zero. For times given in two columns, it might be necessary to reflect the phase start time using an interval with identical endpoints and an event count of zero.

In a multi-phase testing situation, the platform infers whether each phase, other than the last, is failure or time terminated from the entries in the last row preceding a phase change. Suppose that Phase A ends and that Phase B begins at time t_B. In this case, the first row corresponding to Phase B contains an entry for time t_B.

- If the failure time for the last failure in Phase A is exact and if that time differs from t_B, then Phase A is considered to be time terminated. The termination time is equal to t_B.
- If the failure time for the last failure in Phase A is exact and is equal to t_B, then Phase A is considered to be failure terminated.
- If the last failure in Phase A is interval-censored, then Phase A is considered to be time terminated with termination time equal to t_B.

The platform infers whether the final phase is failure or time terminated from the entry in the last row of the data table.

- If the last row contains an exact failure time with a nonzero event count, the test is considered failure terminated.
- If the last row contains an exact failure time with a zero event count, or an interval with nonzero width, the test is considered time terminated. In the case of an interval, the termination time is taken as the right endpoint.

The Reliability Growth Report

The Observed Data report appears by default. If the Parallel Systems data format is specified in the launch window, the Model Descriptions and Feasibilities report also appears by default.

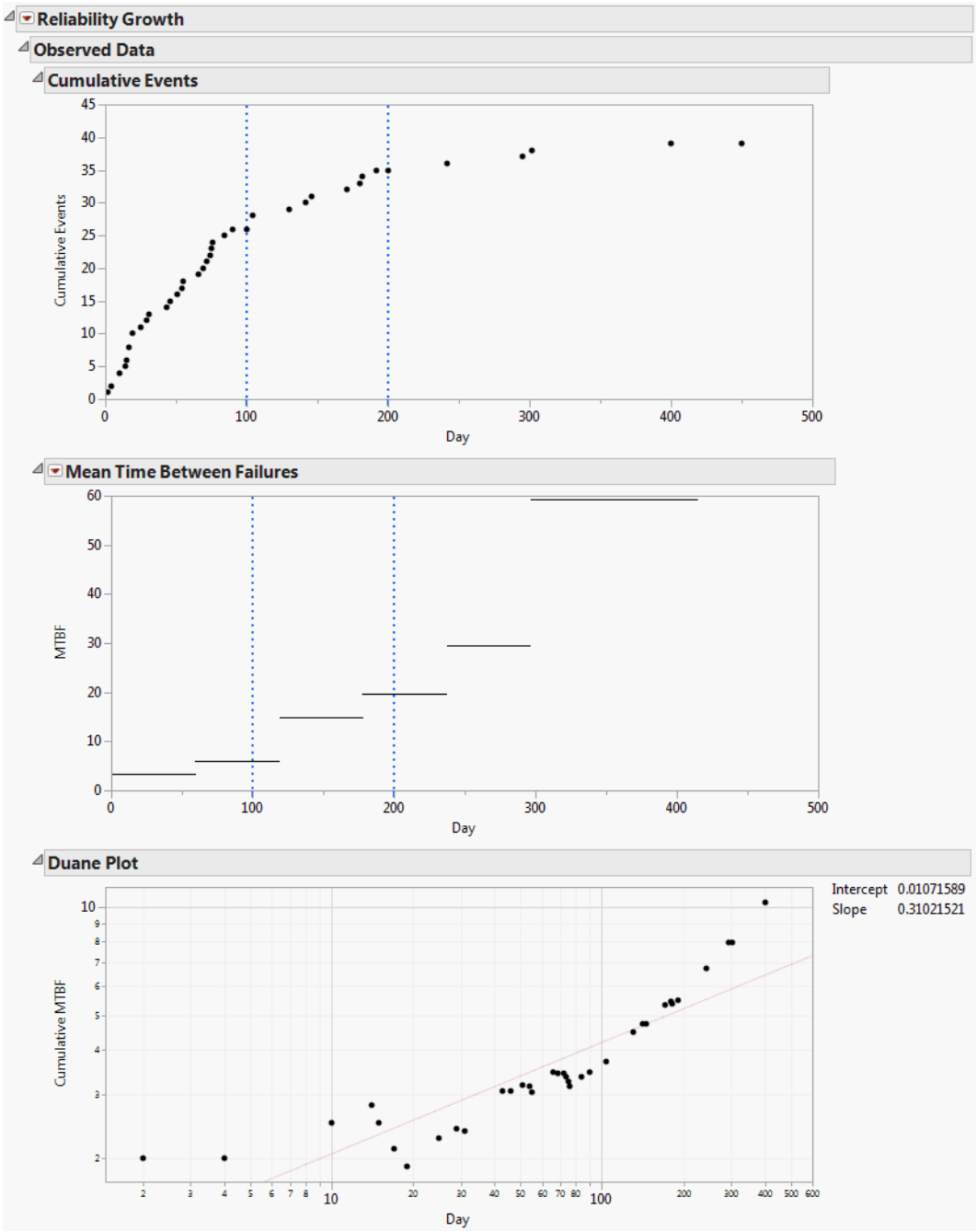
Model Descriptions and Feasibilities

The Model Description and Feasibilities report lists the name and descriptions of the parallel systems models. It also contains a column that shows if a model is available for the current data. If a model is listed as not available, the reason is reported in the right-most column of this table.

Observed Data Report

The Observed Data report contains the Cumulative Events plot, the Mean Time Between Failures plot, and the Duane plot. These are shown in Figure 10.7, where we have opened the Mean Time Between Failures and Duane Plot reports. To produce this report, follow the instructions in "Example Using the Reliability Growth Platform".

Figure 10.7 Observed Data Report



Cumulative Events Plot

This plot shows how events are accumulating over time. The vertical coordinate for each point on the Cumulative Events plot equals the total number of events that have occurred by the time given by the point's horizontal coordinate.

Whenever a model is fit, that model is represented on the Cumulative Events plot. Specifically, the cumulative events estimated by the model are shown by a curve, and 95% confidence intervals are shown by a solid band. Check boxes to the right of the plot enable you to control which models are displayed.

For the Parallel Systems data format, the Cumulative Events plot has a separate panel in the plot for each level of the System ID variable. It also contains a panel at the bottom of the plot that overlays the cumulative events for all levels of the System ID variable. The levels of the System ID variable are designated by colors.

Mean Time Between Failures Plot

The Mean Time Between Failures plot shows mean times between failures averaged over small time intervals of equal length. These are not tied to the Phases. The default number of equal length intervals is based on the number of rows.

Mean Time Between Failures Plot Options

Click the Mean Time Between Failures red triangle and select Options to open a window that enables you to specify intervals over which to average.

Two types of averaging are offered:

- Equal Interval Average MTBF (Mean Time Between Failures) enables you to specify a common interval size.
- Customized Average MTBF enables you to specify cut-off points for time intervals.
  - Double-click within a table cell to change its value.
  - Right-click in the table to open a menu that enables you to add and remove rows.

Duane Plot

The Duane Plot displays Cumulative MTBF estimates plotted against the Time to Event variable, with both axes on a log10 scale. If the data follow the Crow-AMSAA model, the points should follow a line when plotted on log-log paper.

Note: The Duane Plot is available only if failure times are exact and Time to Event Format is used. The plot is not available for interval-censored data or data entered in Dates Format.

The line displayed on the plot is the least squares regression line for the regression of log10 of Cumulative MTBF on log10 of the Time to Event variable.

Note: The Duane Plot does not reflect Phase variables. Rows where the Time to Event variable defines Phase changes are ignored in constructing the plot and fitting the regression line.

Intercept and Slope

Intercept and Slope values are displayed to the right of the plot.

- The Intercept value is given, for historical reasons, as the intercept for a fit in the *natural* logarithmic scale. Specifically, the natural logarithm of Cumulative MTBF is regressed on the natural logarithm of Time to Event. The value of Intercept is the value predicted by this regression equation at log(1) = 0, where *log* is the natural logarithm. To obtain the intercept for a fit in terms of base 10 logarithms, divide the Intercept value by log(10). See Tobias and Trindade (2012, ch. 13).
- The Slope value is the slope for a fit in either the natural or the logarithmic scale. This follows from the properties of logarithms.

Reliability Growth Platform Options

The Reliability Growth red triangle menu contains the following options:

Fit Model If the Time to Event Format, Dates Format, or Concurrent Systems data formats are specified in the launch window, this menu contains options to fit various Non-Homogeneous Poisson Process (NHPP) models. These options are described in detail below. Depending on the choices made in the launch window, the possible options are:

- "Crow AMSAA"
- "Crow AMSAA with Modified MLE"
- "Fixed Parameter Crow AMSAA"
- "Piecewise Weibull NHPP"
- "Reinitialized Weibull NHPP"
- "Piecewise Weibull NHPP Change Point Detection"

Fit Parallel System Model If the Parallel Systems data format is specified in the launch window, this menu contains options to fit various models for multiple-prototype data. The possible options in this menu are dependent on choices made in the launch window.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Model List

Once a model is fit, the Model List report appears. This report provides various statistical measures that describe the fit of the model. As additional models are fit, they are added to the Model List, which provides a convenient summary for model comparison. The models are sorted in ascending order based on AICc. The Model List report contains the following statistics:

Nparm The number of parameters in the model.

-2Loglikelihood The likelihood function is a measure of how probable the observed data are, given the estimated model parameters. In a general sense, the higher the likelihood, the better the model fit. It follows that smaller values of twice the negative of the log-likelihood (-2Loglikelihood) indicate better model fits.

AICc The Corrected Akaike's Information Criterion.

BIC The Bayesian Information Criterion.

See *Fitting Linear Models* for more information about -2Loglikelihood, AICc, and BIC.

Model Reports

This section describes the reports generated when the available models are fit as well as the options available in those model reports.

- "Crow AMSAA"
- "Crow AMSAA with Modified MLE"
- "Fixed Parameter Crow AMSAA"
- "Piecewise Weibull NHPP"
- "Reinitialized Weibull NHPP"
- "Piecewise Weibull NHPP Change Point Detection"

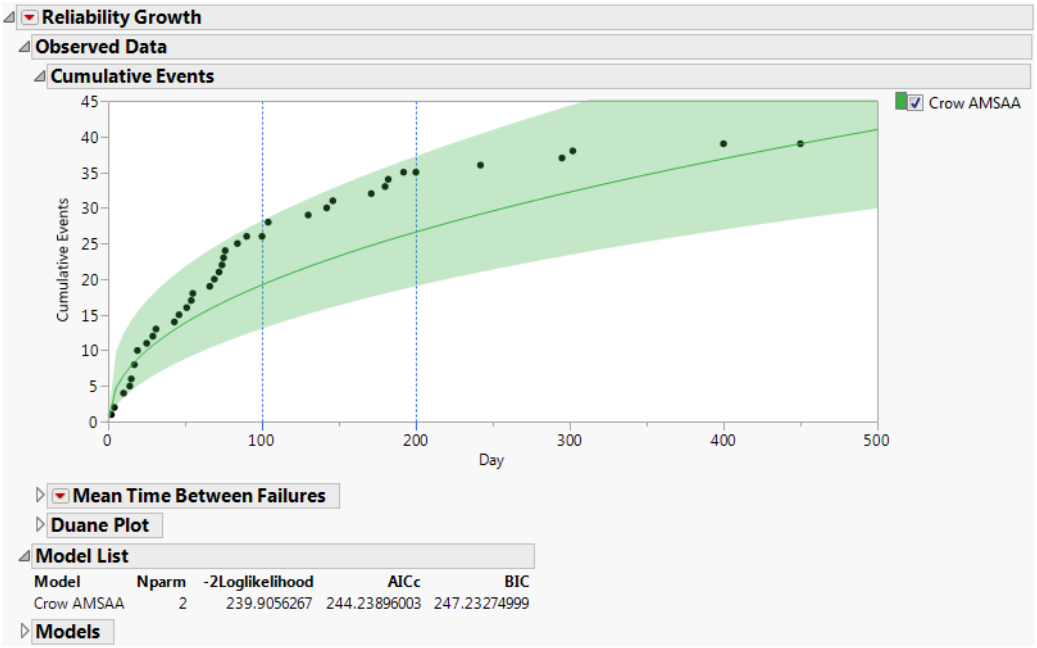
Crow AMSAA

This option fits a Crow-AMSAA model (MIL-HDBK-189 1981). A Crow-AMSAA model is a nonhomogeneous Poisson process with failure intensity as a function of time t given by $\rho(t) = \lambda\beta t^{\beta-1}$. Here, λ is a scale parameter and β is a growth parameter. This function is also called a Weibull intensity, and the process itself is also called a power law process (Rigdon and Basu 2000; Meeker and Escobar 1998). Note that the Recurrence platform fits the Power Nonhomogeneous Poisson Process. The Power Nonhomogeneous Poisson Process is equivalent to the Crow-AMSAA model, although it uses a different parameterization. See "Fit Model".

The intensity function is a concept applied to repairable systems. Its value at time t is the limiting value of the probability of a failure in a small interval around t, divided by the length of this interval; the limit is taken as the interval length goes to zero. You can think of the intensity function as measuring the likelihood of the system failing at a given time. If $\beta < 1$, the system is improving over time. If $\beta > 1$, the system is deteriorating over time. If $\beta = 1$, the rate of occurrence of failures is constant.

When the Crow AMSAA option is selected, the Cumulative Events plot updates to show the cumulative events curve estimated by the model. For each time point, the shaded band around this curve defines a 95% confidence interval for the true cumulative number of events at that time. The Model List report also updates. Figure 10.8 shows the Observed Data report for the data in TurbineEngineDesign1.jmp.

Figure 10.8 Crow AMSAA Cumulative Events Plot and Model List Report



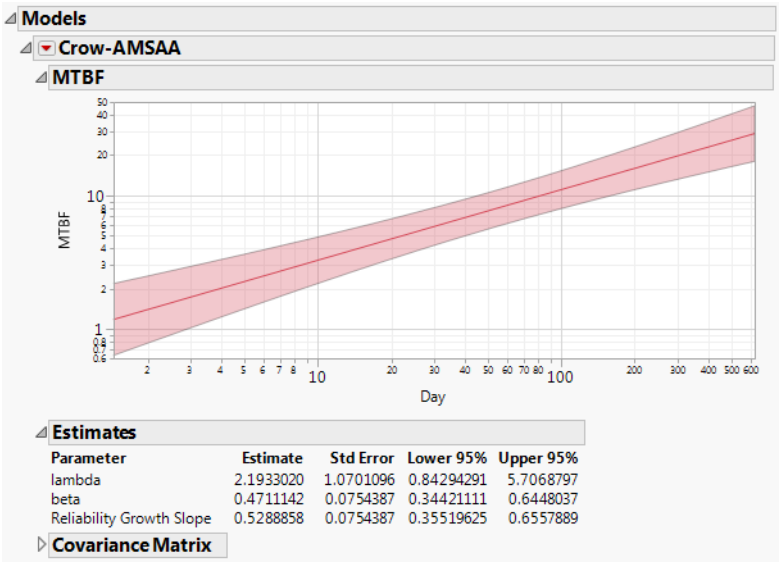
Crow-AMSAA Report

A Crow-AMSAA report opens within the Models report. If Time to Event Format is used, the Crow-AMSAA report shows an MTBF plot with both axes scaled logarithmically. See "MTBF Plot".

MTBF Plot

The mean time between failures (MTBF) plot is displayed by default (Figure 10.9). For each time point, the shaded band around the MTBF plot defines a 95% confidence interval for the true MTBF at time *t*. The plot is shown with both axes logarithmically scaled. With this scaling, the MTBF plot is linear.

Figure 10.9 MTBF Plot



To see why the MTBF plot is linear when logarithmic scaling is used, consider the following. The mean time between failures is the reciprocal of the intensity function. For the Weibull intensity function, the MTBF is $1/(\lambda\beta t^{\beta-1})$, where t represents the time since testing initiation. It follows that the logarithm of the MTBF is a linear function of $\log(t)$, with slope $1 - \beta$. The estimated MTBF is defined by replacing the parameters λ and β by their estimates. So the log of the estimated MTBF is a linear function of $\log(t)$.

Estimates

Maximum likelihood estimates for lambda (λ), beta (β), and the Reliability Growth Slope ($1 - \beta$), appear in the Estimates report below the plot (Figure 10.9). Standard errors and 95% confidence intervals for λ, β, and $1 - \beta$ are given. For more information about the calculations, see "Parameter Estimates for Crow-AMSAA Models".

Covariance Matrix

Estimated covariance matrix for the estimates of the parameters of the fitted model. This report is closed by default.

Crow-AMSAA Options

This section describes the options that are available in the Crow-AMSAA red triangle menu when a Crow AMSAA model is fit.

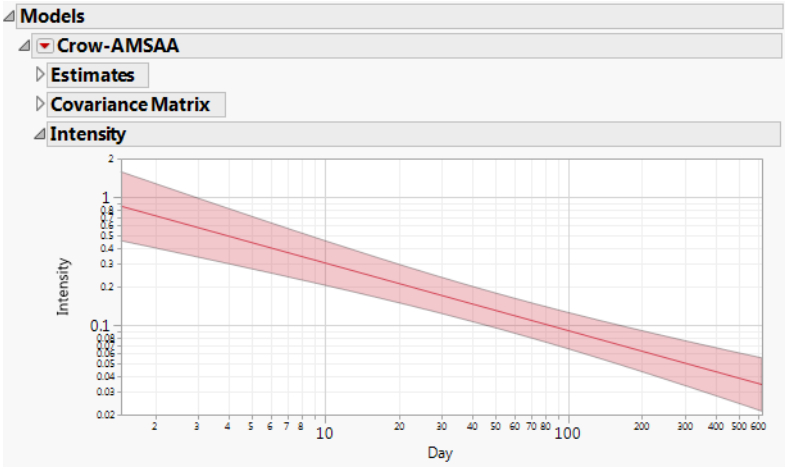
Show MTBF Plot

This option shows or hides the MTBF Plot. See "MTBF Plot".

Show Intensity Plot

This plot shows the estimated intensity function. The Weibull intensity function is given by $\rho(t) = \lambda\beta t^{\beta-1}$, so it follows that log(Intensity) is a linear function of log(t). Both axes are scaled logarithmically.

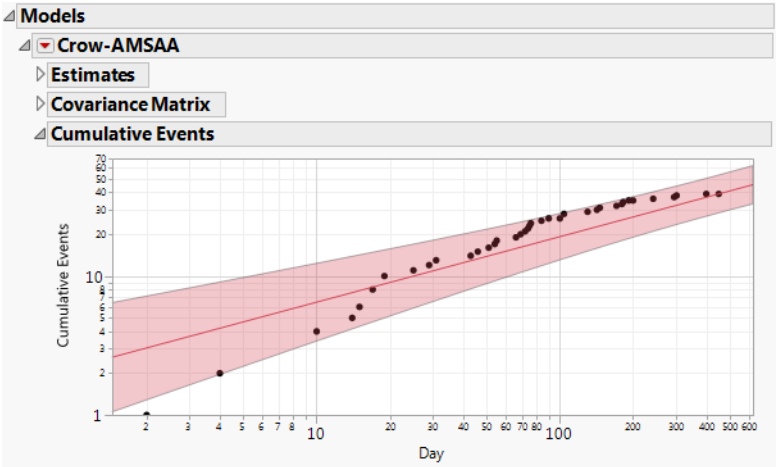
Figure 10.10 Intensity Plot



Show Cumulative Events Plot

This plot shows the estimated cumulative number of events. The observed cumulative numbers of events are also displayed on this plot. Both axes are scaled logarithmically.

Figure 10.11 Cumulative Events Plot



For the Crow-AMSAA model, the cumulative number of events at time t is given by λt^{β}. It follows that the logarithm of the cumulative number of events is a linear function of $\log(t)$. So, the plot of the estimated Cumulative Events is linear when plotted against logarithmically scaled axes.

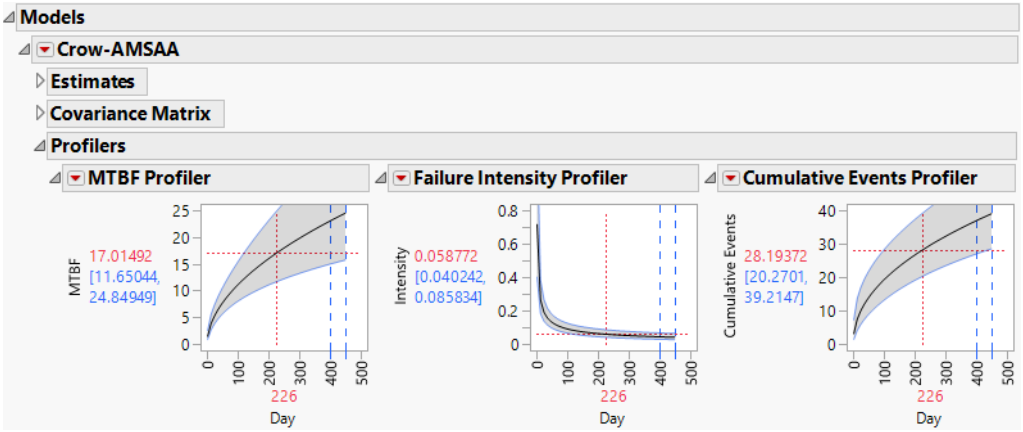
Show Profilers

Three profilers are displayed, showing estimated MTBF, Failure Intensity, and Cumulative Events. These profilers do not use logarithmic scaling. By dragging the red vertical dashed line in any profiler, you can explore model estimates at various time points; the value of the selected time point is shown in red beneath the plot. Also, you can set the time axis to a specific value by pressing Ctrl while you click in the plot. A blue vertical dashed line denotes the time point of the last observed failure.

The profilers also display 95% confidence bands for the estimated quantities. For the specified time setting, the estimated quantity (in red) and 95% confidence limits (in black) are shown to the left of the profiler. See "Profilers".

Note that you can link these profilers by selecting **Factor Settings > Link Profilers** from any of the profiler red triangle menus. For more information about the use and interpretation of profilers, see *Fitting Linear Models*. See also *Profilers*.

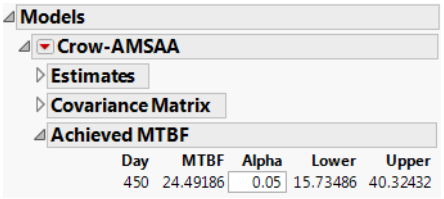
Figure 10.12 Profilers



Achieved MTBF

A confidence interval for the MTBF at the point when testing concludes is often of interest. For uncensored failure time data, this report gives an estimate of the Achieved MTBF and a 95% confidence interval for the Achieved MTBF. You can specify a 100*(1-α)% confidence interval by entering a value for Alpha. The report is shown in Figure 10.13. For censored data, only the estimated MTBF at test termination is reported.

Figure 10.13 Achieved MTBF Report



There are infinitely many possible failure-time sequences from an NHPP; the observed data represent only one of these. Suppose that the test is failure terminated at the n^{th} failure. The confidence interval computed in the Achieved MTBF report takes into account the fact that the n failure times are random. If the test is time terminated, then the number of failures as well as their failure times are random. Because of this, the confidence interval for the Achieved MTBF differs from the confidence interval provided by the MTBF Profiler at the last observed failure time. See Crow (1982) and Lee and Lee (1978).

When the test is failure terminated, the confidence interval for the Achieved MTBF is exact. However, when the test is time terminated, an exact interval cannot be obtained. In this case, the limits are conservative in the sense that the interval contains the Achieved MTBF with probability at least 1-α.

Goodness of Fit

The Goodness of Fit report tests the null hypothesis that the data follow a Crow-AMSAA model. Depending on whether one or two time columns are entered, either a Cramér-von Mises (see "Cramér-von Mises Test for Data with Uncensored Failure Times") or a chi-squared test (see "Chi-Squared Goodness of Fit Test for Interval-Censored Failure Times") is performed.

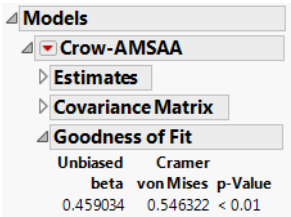
Cramér-von Mises Test for Data with Uncensored Failure Times

When the data are entered in the launch window as a single Time to Event or Timestamp column, the goodness of fit test is a Cramér-von Mises test. For the Cramér-von Mises test, large values of the test statistic lead to rejection of the null hypothesis and the conclusion that the model does not fit adequately. The test uses an unbiased estimate of beta, given in the report. The value of the test statistic is found below the Cramér-von Mises heading.

The entry below the p-Value heading indicates how unlikely it is for the test statistic to be as large as what is observed if the data come from a Crow-AMSAA model. The platform computes *p*-values up to 0.25. If the test statistic is smaller than the value that corresponds to a *p*-value of 0.25, the report indicates that its *p*-value is >=0.25. For more information about this test, see Crow (1975).

Figure 10.14 shows the goodness-of-fit test for the fit of a Crow-AMSAA model to the data in TurbineEngineDesign1.jmp. The computed test statistic corresponds to a *p*-value that is less than 0.01. We conclude that the Crow-AMSAA model does not provide an adequate fit to the data.

Figure 10.14 Goodness of Fit Report - Cramér-von Mises Test



Chi-Squared Goodness of Fit Test for Interval-Censored Failure Times

When the data are entered in the launch window as two Time to Event or Timestamp columns, a chi-squared goodness of fit test is performed. The chi-squared test is based on comparing observed to expected numbers of failures in the time intervals defined. Large values of the test statistic lead to rejection of the null hypothesis and the conclusion that the model does not fit.

In the Reliability Growth platform, the chi-squared goodness of fit test is intended for interval-censored data where the time intervals specified in the data table cover the entire time period of the test. This means that the start time of an interval is the end time of the preceding interval. In particular, intervals where no failures occurred should be included in the data table. If some intervals are not consecutive, or if some intervals have identical start and end times, the algorithm makes appropriate accommodations. But the resulting test is only approximately correct.

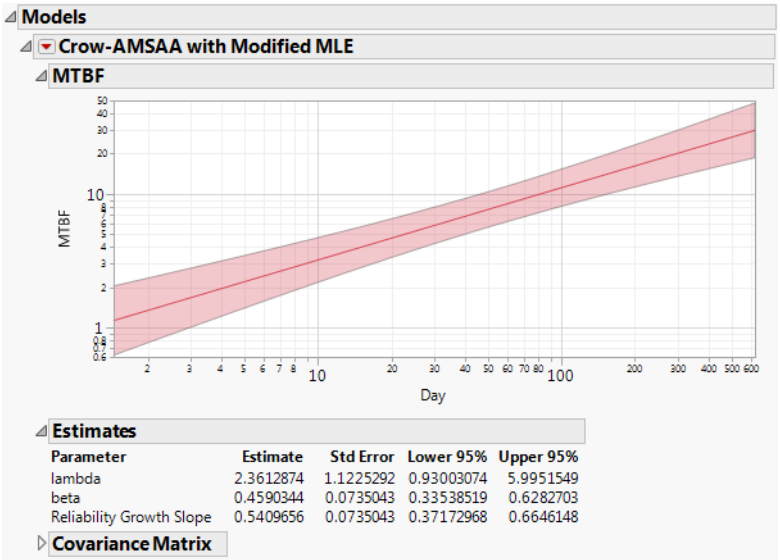
Crow AMSAA with Modified MLE

In the Crow-AMSAA model, the maximum likelihood estimate (MLE) of β is biased. This option fits a Crow AMSAA model where β is adjusted for bias.

Note: This option is available only when the data are entered in the launch window as a single Time to Event or Timestamp column. It is not available for interval-censored data.

Figure 10.15 shows a Crow-AMSAA with Modified MLE fit to the data in TurbineEngineDesign1.jmp.

Figure 10.15 Crow-AMSAA with Modified MLE Report



The formula for the bias-corrected estimate of β depends on whether the test is failure terminated or time terminated. See "Parameter Estimates for Crow-AMSAA with Modified MLE".

When the Crow-AMSAA with Modified MLE option is selected, the Cumulative Events Plot updates to display this model. The Model List also updates. The Crow-AMSAA with Modified MLE report opens to show the MTBF plot, Estimates, and Covariance Matrix for the Crow-AMSAA with Modified MLE fit; this plot is described in the section "MTBF Plot".

In addition to Show MTBF plot, available options are Show Intensity Plot, Show Cumulative Events Plot, Show Profilers, Achieved MTBF, and Goodness of Fit. These reports are described under "Crow AMSAA". For more information about how the modified MLEs are used to construct these reports, see "Parameter Estimates for Crow-AMSAA with Modified MLE". Details about the Goodness of Fit and Achieved MTBF reports specific to the modified MLE option are given below.

Goodness of Fit

Because the Crow-AMSAA with Modified MLE option is available only when the data are entered as a single Time to Event or Timestamp column, the Goodness of Fit test is a Cramér-von Mises test. Because the estimate of β is used in this test is bias-corrected, the test results are identical to those of the Goodness of Fit test for the Crow-AMSAA model.

Achieved MTBF

The achieved MTBF is estimated using the modified MLEs. However, the confidence interval for the achieved MTBF uses the true MLE and is identical to the interval given by the Crow AMSAA model.

Fixed Parameter Crow AMSAA

This option enables you to specify parameter values for a Crow-AMSAA fit. If a Crow-AMSAA report has not been obtained before choosing the Fixed Parameter Crow-AMSAA option, then both a Crow-AMSAA report and a Fixed Parameter Crow-AMSAA report are provided.

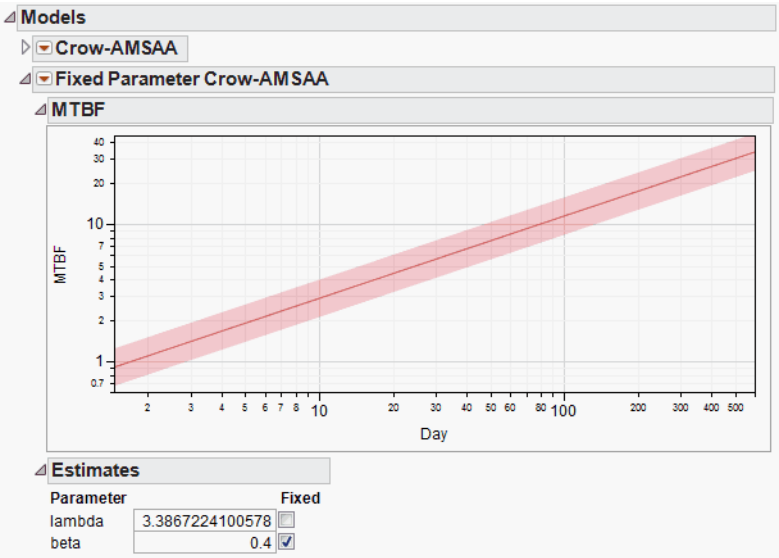
When the Fixed Parameter Crow-AMSAA option is selected, the Cumulative Events Plot updates to display this model. The Model List also updates. The Fixed Parameter Crow-AMSAA report opens to show the MTBF plot for the Crow-AMSAA fit; this plot is described in the section "MTBF Plot".

In addition to Show MTBF plot, available options are Show Intensity Plot, Show Cumulative Events Plot, and Show Profilers. The construction and interpretation of these plots is described under "Crow AMSAA".

Estimates

The initial parameter estimates are the MLEs from the Crow-ASMAA fit. Either parameter can be fixed by checking the box next to the desired parameter and then entering the desired value. The model is re-estimated and the MTBF plot updates to describe this model. Figure 10.16 shows a fixed-parameter Crow-AMSAA fit to the data in TurbineEngineDesign1.jmp, with the value of beta set at 0.4.

Figure 10.16 Fixed Parameter Crow-AMSAA Report

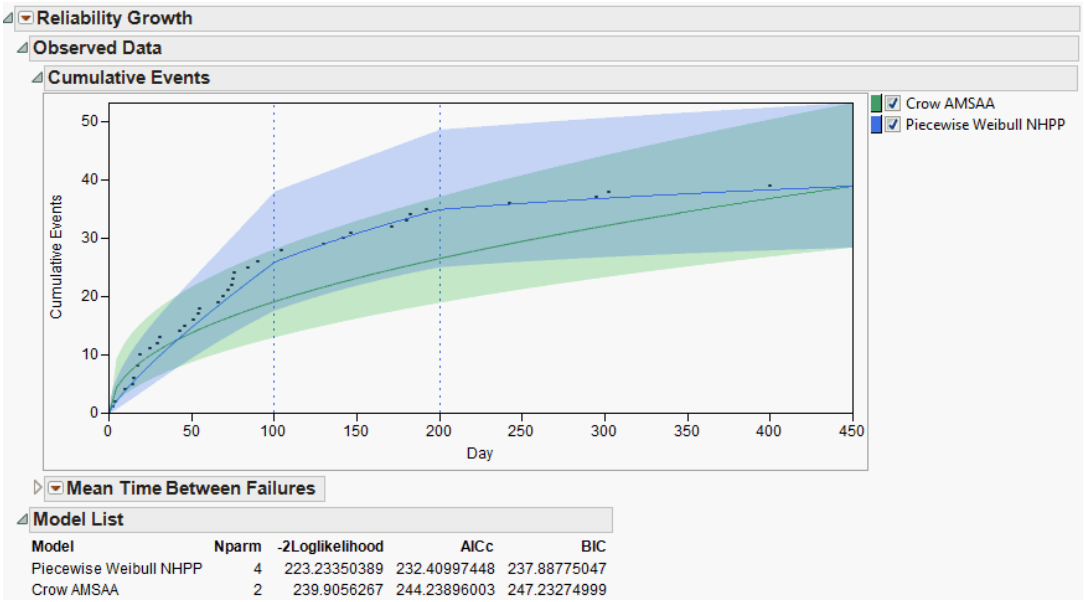


Piecewise Weibull NHPP

The Piecewise Weibull NHPP model can be fit when a Phase column specifying at least two values has been entered in the launch window. Crow-AMSAA models are fit to each of the phases under the constraint that the cumulative number of events at the start of a phase matches that number at the end of the preceding phase. For proper display of phase transition times, the first row for every Phase other than the first must give that phase's start time. See "Multiple Test Phases".

When the report is run, the Cumulative Events plot updates to show the piecewise model. Blue vertical dashed lines show the transition times for each of the phases. The Model List also updates. See Figure 10.17, where both a Crow-AMSAA model and a piecewise Weibull NHPP model have been fit to the data in TurbineEngineDesign1.jmp. Note that both models are compared in the Model List report.

Figure 10.17 Cumulative Events Plot and Model List Report

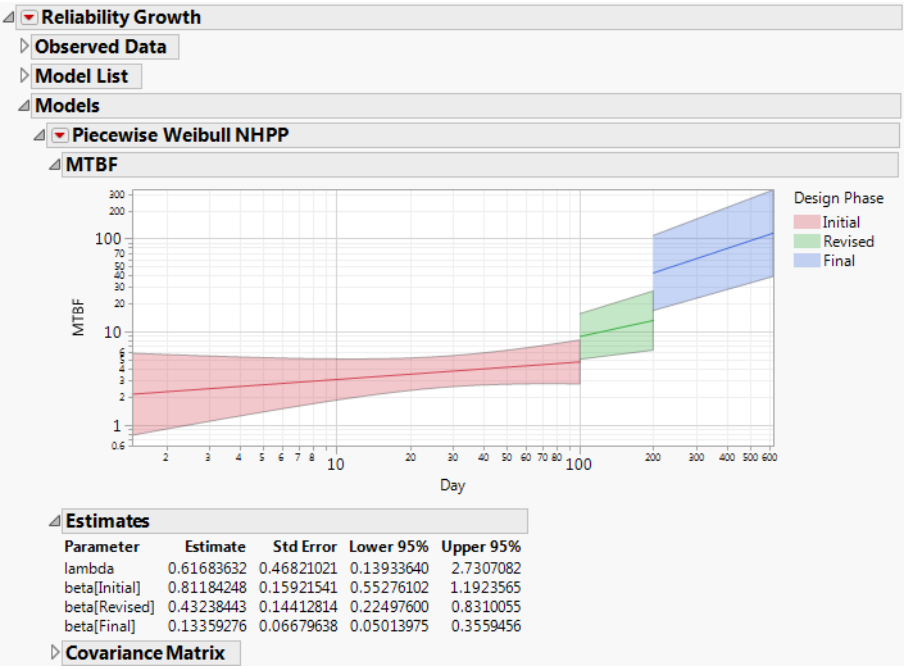


Model	Nparm	-2Loglikelihood	AICc	BIC
Piecewise Weibull NHPP	4	223.23350389	232.40997448	237.88775047
Crow AMSAA	2	239.9056267	244.23896003	247.23274999

Piecewise Weibull NHPP Report

By default, the Piecewise Weibull NHPP report shows the estimated MTBF plot. The phases are differentiated with different colors. The Estimates and Covariance Matrix reports are shown below the plot.

Figure 10.18 Piecewise Weibull NHPP Report



Parameter	Estimate	Std Error	Lower 95%	Upper 95%
lambda	0.61683632	0.46821021	0.13933640	2.7307082
beta[Initial]	0.81184248	0.15921541	0.55276102	1.1923565
beta[Revised]	0.43238443	0.14412814	0.22497600	0.8310055
beta[Final]	0.13359276	0.06679638	0.05013975	0.3559456

MTBF Plot

The MTBF plot and the Estimates report appear by default when the Piecewise Weibull NHPP option is chosen (Figure 10.18). When Time to Event Format is used, the axes are logarithmically scaled. For more information about the plot, see "MTBF Plot".

Estimates

The Estimates report gives estimates of the model parameters. Note that only the estimate for the value of λ corresponding to the first phase is given. In the piecewise model, the cumulative events at the end of one phase must match the number at the beginning of the subsequent phase. Because of these constraints, the estimate of λ for the first phase and the estimates of the βs determine the remaining λs.

The method used to calculate the estimates, their standard errors, and the confidence limits is similar to that used for the simple Crow-AMSAA model. See "Parameter Estimates for Crow-AMSAA Models". The likelihood function reflects the additional parameters and the constraints on the cumulative numbers of events.

Covariance Matrix

Estimated covariance matrix for the estimates of the parameters of the fitted model. This report is closed by default.

Piecewise Weibull NHPP Options

This section describes the options available in the Piecewise Weibull NHPP red triangle menu.

Show MTBF Plot

This option shows or hides the MTBF plot. See "MTBF Plot".

Show Intensity Plot

The Intensity plot shows the estimated intensity function and confidence bounds over the design phases. The intensity function is generally discontinuous at a phase transition. Color coding facilitates differentiation of phases. If Time to Event Format is used, the axes are logarithmically scaled. See "Show Intensity Plot".

Show Cumulative Events Plot

The Cumulative Events plot shows the estimated cumulative number of events, along with confidence bounds, over the design phases. The model requires that the cumulative events at the end of one phase match the number at the beginning of the subsequent phase. Color coding facilitates differentiation of phases. If Time to Event Format is used, the axes are logarithmically scaled. See "Show Cumulative Events Plot".

Show Profilers

Three profilers are displayed, showing estimated MTBF, Failure Intensity, and Cumulative Events. These profilers do not use logarithmic scaling. For more information about interpreting and using these profilers, see the section "Show Profilers".

It is important to note that, due to the default resolution of the profiler plot, discontinuities are not displayed clearly in the MTBF or Failure Intensity Profilers. In the neighborhood of a phase transition, the profiler trace shows a nearly vertical, but slightly sloped, line; this line represents a discontinuity (Figure 10.19). Such a line at a phase transition should not be used for estimation. Obtain a higher-resolution display to make these lines appear more vertical:

1. Press Ctrl while clicking in the profiler plot.
2. Enter a larger value for Number of Plotted Points in the window. (See Figure 10.20, where we have specified 500 as the Number of Plotted Points.)

Figure 10.19 Profilers

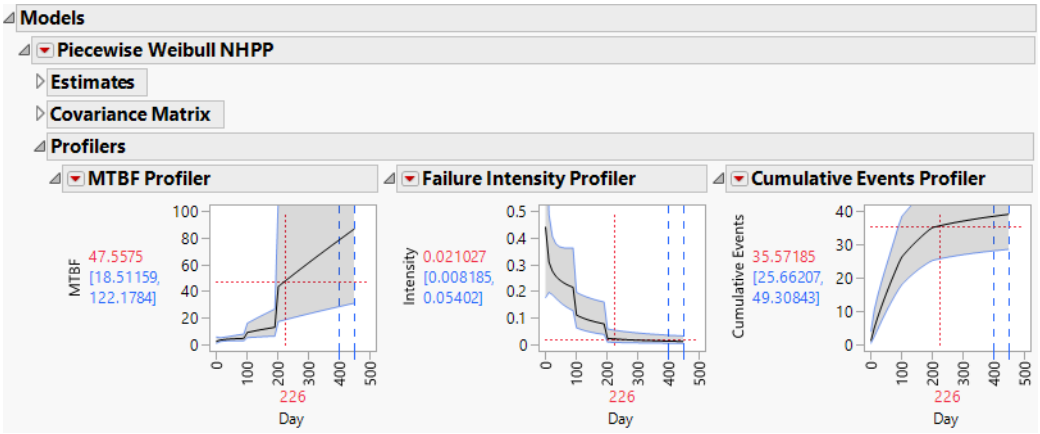
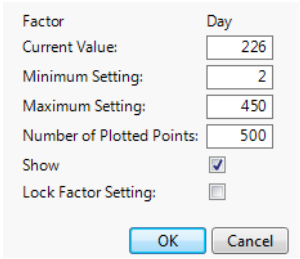


Figure 10.20 Factor Settings Window



Reinitialized Weibull NHPP

This option fits an independent growth model to the data from each test phase. Fitting models in this fashion can be useful when the factors influencing the growth rate, either in terms of testing or engineering, have changed substantially between phases. In such a situation, you might want to compare the test phases independently. The Reinitialized Weibull NHPP option is available when a Phase column specifying at least two phases has been entered in the launch window.

For the algorithm to fit this model, each row that contains the first occurrence of a new phase must contain the start date.

- Suppose that a single column is entered as Time to Event or Timestamp. Then the start time for a new phase, with a zero Event Count, must appear in the first row for that phase. See the sample data table ProductionEquipment.jmp, in the Reliability subfolder, for an example.
- If two columns are entered, then an interval whose left endpoint is that start time must appear in the first row, with the appropriate event count. The sample data table

TurbineEngineDesign2.jmp, found in the Reliability subfolder, provides an example. Also, see "Piecewise NHPP Weibull Model Fitting with Interval-Censored Data".

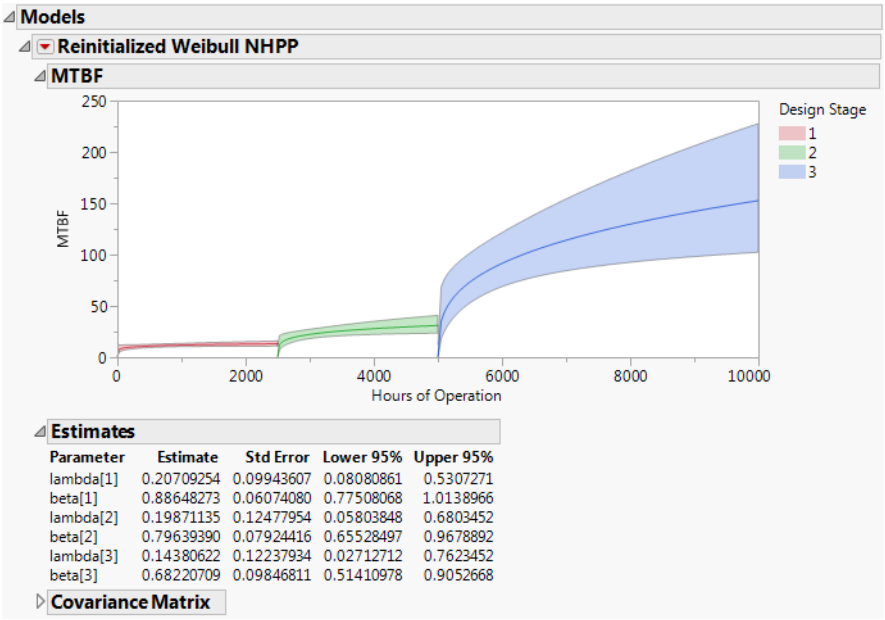
See "Multiple Test Phases".

Independent Crow-AMSAA models are fit to the data from each of the phases. When the report is run, the Cumulative Events plot updates to show the reinitialized models. Blue vertical dashed lines show the transition points for each of the phases. The Model List also updates.

Reinitialized Weibull NHPP Report

By default, the Reinitialized Weibull NHPP report shows the estimated MTBF plot. The phases are differentiated with different colors. The Estimates and Covariance Matrix reports are shown below the plot. (See Figure 10.21, which uses the ProductionEquipment.jmp sample data file from the Reliability subfolder.)

Figure 10.21 Reinitialized Weibull NHPP Report



MTBF Plot

The MTBF plot opens by default when the Reinitialized Weibull NHPP option is chosen. For more information s about the plot, see "MTBF Plot".

Estimates

The Estimates report gives estimates of λ and β for each of the phases. For a given phase, λ and β are estimated using only the data from that phase. The calculations assume that the phase begins at time 0 and reflect whether the phase is failure or time terminated, as defined by the data table structure. See "Test Phases". Also shown are standard errors and 95% confidence limits. These values are computed as described in "Parameter Estimates for Crow-AMSAA Models".

Covariance Matrix

Estimated covariance matrix for the estimates of the parameters of the fitted model. This report is closed by default.

Reinitialized Weibull NHPP Options

This section describes the options available in the Reinitialized Weibull NHPP red triangle menu.

Show MTBF Plot

This option shows or hides the MTBF plot. See "MTBF Plot".

Show Intensity Plot

The Intensity plot shows the estimated intensity functions for the phases, along with confidence bands. Because the intensity functions are computed based only on the data within a phase, they are discontinuous at phase transitions. Color coding facilitates differentiation of phases. See "Show Intensity Plot".

Show Cumulative Events Plot

The Cumulative Events plot for the Reinitialized Weibull NHPP model portrays the estimated cumulative number of events, with confidence bounds, over the design phases in the following way. Let t represent the time since the first phase of testing began. The model for the phase that is in effect at time t is evaluated at time t. In particular, the model for the phase that is in effect is not evaluated at the time since the beginning of the specific phase; rather it is evaluated at the time since the beginning of the *first* phase of testing.

At phase transitions, the cumulative events functions are discontinuous. The Cumulative Events plot matches the estimated cumulative number of events at the beginning of one phase to the cumulative number at the end of the previous phase. This matching allows the user to compare the observed cumulative events to the estimated cumulative events functions. Color coding facilitates differentiation of phases.

Show Profilers

Three profilers are displayed, showing estimated MTBF, Failure Intensity, and Cumulative Events. Note that the Cumulative Events Profiler is constructed as described in the Cumulative Events Plot section. In particular, the cumulative number of events at the beginning of one phase is matched to the number at the end of the previous phase. See "Show Profilers".

Piecewise Weibull NHPP Change Point Detection

The Piecewise Weibull NHPP Change Point Detection option attempts to find a time point where the reliability model changes. This might be useful if you suspect that a change in reliability growth has occurred over the testing period. Note that detection seeks only a single change point, corresponding to two potential phases.

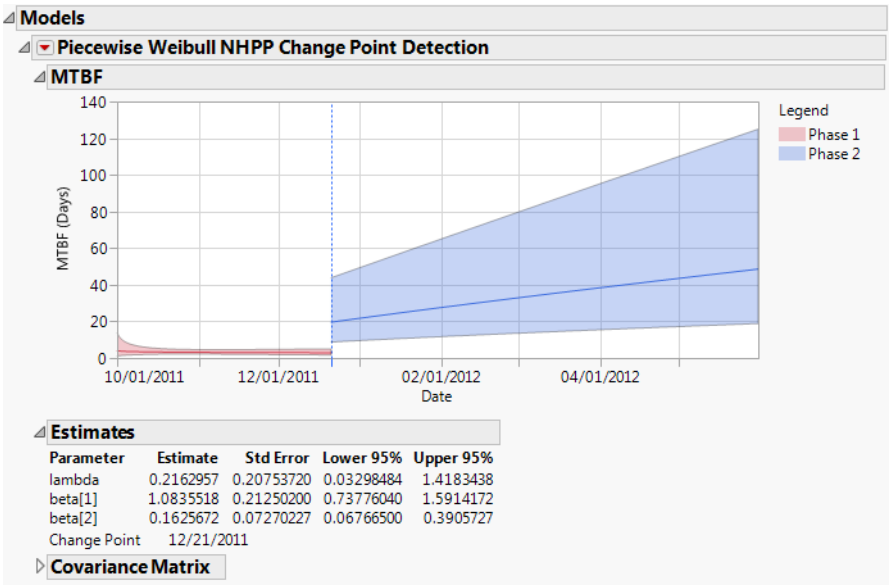
This option is available only when a Phase has *not* been entered in the launch window and one of the following is true:

- a single column has been entered as Time to Event or Timestamp in the launch window (indicating that failure times are exact), and
- two columns have been entered as Time to Event in the Concurrent Systems panel of the launch window.

When the Piecewise Weibull NHPP Change Point Detection option is selected, the estimated model plot and confidence bands are added to the Cumulative Events report under Observed Data. The Model List updates, giving statistics that are conditioned on the estimated change point. Under Models, a Piecewise Weibull NHPP Change Point Detection report is provided.

The default Piecewise Weibull NHPP Change Point Detection report shows the MTBF plot, Estimates, and Covariance Matrix. (See Figure 10.22, which uses the data in BrakeReliability.jmp, found in the Reliability subfolder.) Note that the Change Point, shown at the bottom of the Estimates report, is estimated as 12/21/2011. The standard errors and confidence intervals consider the change point to be known. The plot and the Estimates report are described in the section "Piecewise Weibull NHPP".

Figure 10.22 Piecewise Weibull NHPP Change Point Detection Report



Available options are: Show MTBF Plot, Show Intensity Plot, Show Cumulative Events Plot, Show Profilers. These options are described in the section "Reinitialized Weibull NHPP Options".

The procedure used in estimating the change point is described in "Statistical Details for the Piecewise Weibull NHPP Change Point Detection Report".

Additional Examples of the Reliability Growth Platform

- "Piecewise NHPP Weibull Model Fitting with Interval-Censored Data"
- "Piecewise Weibull NHPP Change Point Detection with Time in Dates Format"

Piecewise NHPP Weibull Model Fitting with Interval-Censored Data

The sample data file TurbineEngineDesign2.jmp, found in the Reliability subfolder, contains data on failures for a turbine engine design over three phases of a testing program. The first two columns give time intervals during which failures occurred. These intervals are recorded as days since the start of testing. The exact failure times are not known; it is known only that failures occurred within these intervals.

The reports of failures are provided generally at weekly intervals. Intervals during which there were no failures and which fell strictly within a phase are not included in the data table (for example, the interval 106 to 112 is not represented in the table). Because these make no contribution to the likelihood function, they are not needed for estimation of model parameters.

However, to fit a Piecewise Weibull NHPP or Reinitialized Weibull NHPP model, it is important that the start times for all phases be provided in the Time to Event or Timestamp columns.

Here, the three phases began at days 0 (Initial phase), 91 (Revised phase), and 200 (Final phase). There were failures during the weeks that began the Initial and Revised phases. However, no failures were reported between days 196 and 231. For this reason, an interval with beginning and ending days equal to 200 was included in the table (row 23), reflecting 0 failures. This is necessary so that JMP knows the start time of the Final phase.

The test was terminated at 385 days. This is an example of interval-censored failure times with time terminated phases.

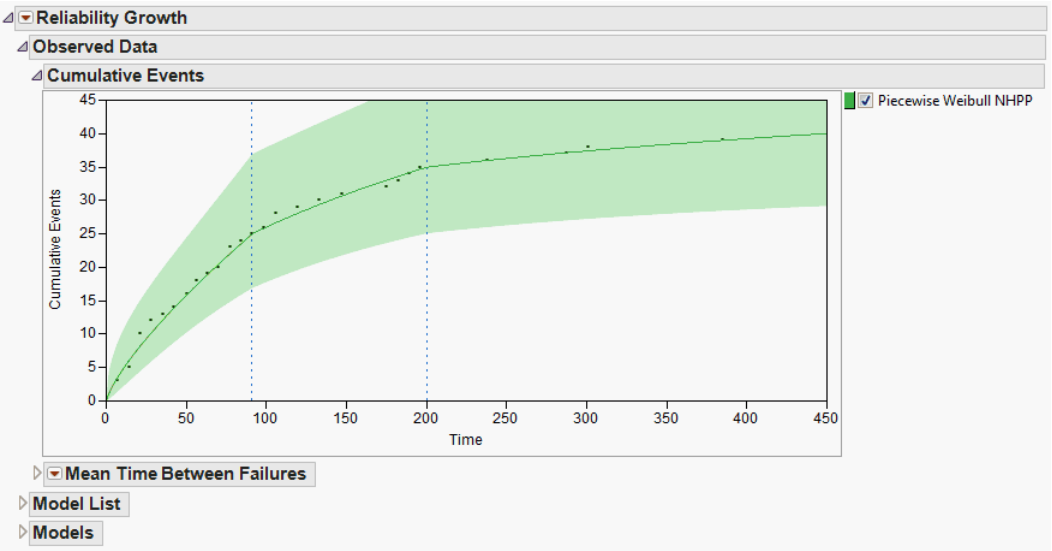
Note: The phase start times are required for proper display of the transition times for the Piecewise Weibull NHPP model; they are required for estimation of the Reinitialized Weibull NHPP model. For interval-censored data, the algorithm defines the beginning time for a phase as the start date recorded in the row containing the first occurrence of that phase designation. In our example, if row 23 were not in the table, the beginning time of the Final phase would be taken as 231.

---

1. Select **Help > Sample Data Library** and open Reliability/TurbineEngineDesign2.jmp.
2. Select **Analyze > Reliability and Survival > Reliability Growth**.
3. On the **Time to Event Format** tab, select the columns Interval Start and Interval End, and click **Time to Event**.
4. Select Fixes and click **Event Count**.
5. Select Design Phase and click **Phase**.
6. Click **OK**.
7. Click the Reliability Growth red triangle and select **Fit Model > Piecewise Weibull NHPP**.

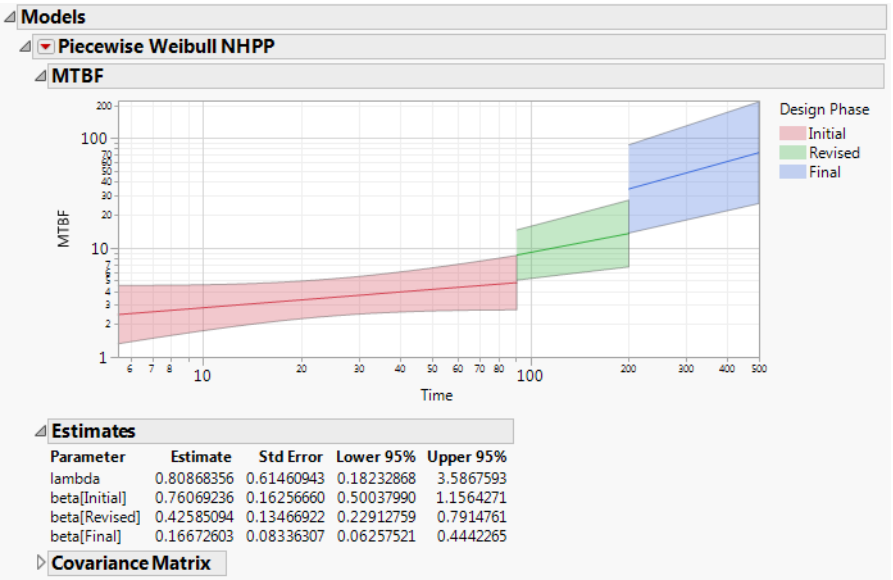
   The Cumulative Events plot from the Observed Data report is shown in Figure 10.23. The vertical dashed blue lines indicate the phase transition points. The first occurrence of Revised in the column Design Phase is in row 14. So, the start of the Revised phase is taken to be the Interval Start value in row 14, namely, day 91. Similarly, the first occurrence of Final in the column Design Phase is in row 23. So, the start of the Final phase is taken to be the Interval Start value in row 23, namely, day 200.

Figure 10.23 Cumulative Events Plot



The Piecewise Weibull NHPP report is found under the Models outline node. Here we see the mean time between failures increasing over the three phases. From the Estimates report, we see that the estimates of beta decrease over the three testing phases.

Figure 10.24 MTBF Plot



Parameter	Estimate	Std Error	Lower 95%	Upper 95%
lambda	0.80868356	0.61460943	0.18232868	3.5867593
beta[Initial]	0.76069236	0.16256660	0.50037990	1.1564271
beta[Revised]	0.42585094	0.13466922	0.22912759	0.7914761
beta[Final]	0.16672603	0.08336307	0.06257521	0.4442265

Piecewise Weibull NHPP Change Point Detection with Time in Dates Format

The file BrakeReliability.jmp, found in the Reliability subfolder, contains data on fixes to a braking system. The Date column gives the dates when Fixes, given in the second column, were implemented. For this data, the failure times are known. Note that the Date column must be in ascending order.

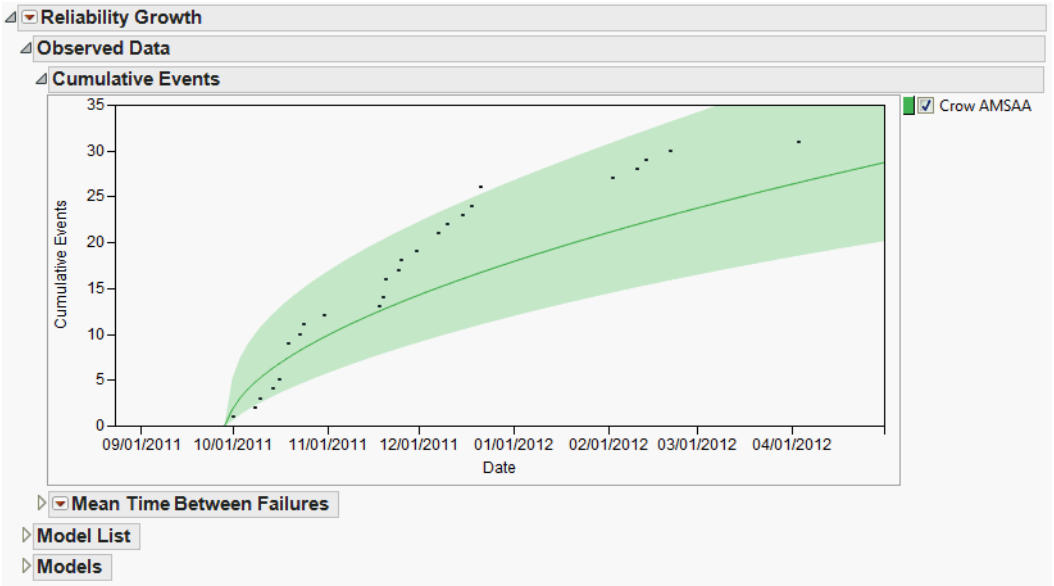
The test start time is the first entry in the Date column, 09/29/2011, and the corresponding value for Fixes is set at 0. This is needed in order to convey the start time for testing. If there had been a nonzero value for Fixes in this first row, the corresponding date would have been treated as the test start time. However, the value of Fixes would have been treated as 0 in the analysis.

The test termination time is given in the last row as 05/31/2012. Because the value in Fixes in the last row is 0, the test is considered to be time terminated on 5/31/2012. If there had been a nonzero value for Fixes in this last row, the test would have been considered failure terminated.

1. Select **Help > Sample Data Library** and open Reliability/BrakeReliability.jmp.
2. Select **Analyze > Reliability and Survival > Reliability Growth**.
3. Select the **Dates Format** tab.
4. Select Date and click **Timestamp**.
5. Select Fixes and click **Event Count**.
6. Click **OK**.
7. Click the Reliability Growth red triangle and select **Fit Model > Crow AMSAA**.

   The Cumulative Events plot in the Observed Data report updates to show the model. The model does not seem to fit the data very well.

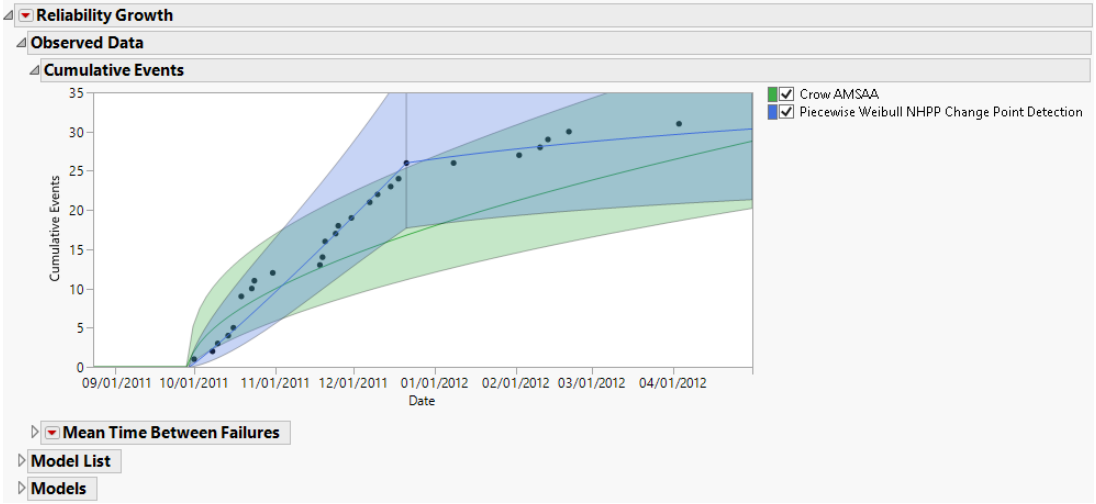
Figure 10.25 Cumulative Events Plot with Crow AMSAA Model



8. Click the Reliability Growth red triangle and select **Fit Model > Piecewise Weibull NHPP Change Point Detection**.

   The Cumulative Events plot in the Observed Data report updates to show the piecewise model fit using change-point detection. Both models are shown in Figure 10.26. Though the data are rather sparse, the piecewise model seems to provide a better fit to the data.

Figure 10.26 Cumulative Events Plot with Two Models



Statistical Details for the Reliability Growth Platform

- "Statistical Details for the Crow-AMSAA Report"
- "Statistical Details for the Piecewise Weibull NHPP Change Point Detection Report"

Statistical Details for the Crow-AMSAA Report

This section contains details for the parameter estimates and profilers that appear in the Crow-AMSAA Report.

Parameter Estimates for Crow-AMSAA Models

With the exception of the Crow-AMSAA with Modified MLE option, the estimates for λ and β are maximum likelihood estimates, which are computed as follows. The likelihood function is derived using the methodology in Meeker and Escobar (1998). It is reparametrized in terms of $param_1 = \log(\lambda)$ and $param_2 = \log(\beta)$. This is done to enable the use of an unconstrained optimization algorithm, namely, an algorithm that searches from -∞ to +∞. The MLEs for $param_1$ and $param_2$ are obtained.

The standard errors for λ and β are obtained from the Fisher information matrix. Confidence limits for $param_1$ and $param_2$ are calculated based on the asymptotic distribution of the MLEs, using the Wald statistic. These estimates and their confidence limits are then transformed back to the original units using the exponential function.

Parameter Estimates for Crow-AMSAA with Modified MLE

For the Crow AMSAA with Modified MLE option, the estimate for β is corrected for bias. The formula for the bias-corrected estimate of β depends on whether the test is failure terminated or time terminated.

Denote the MLE for β by $\hat{\beta}$, let n be the number of observations, and let T be the total test time.

The bias-corrected estimate (*modified MLE*) of β is $\bar{\beta}$, where:

$$\bar{\beta} = \left(\frac{n-2}{n}\right)\hat{\beta} \quad \text{for a failure-terminated test}$$

$$\bar{\beta} = \left(\frac{n-1}{n}\right)\hat{\beta} \quad \text{for a time-terminated test}$$

The modified MLE for λ, denoted $\bar{\lambda}$, is calculated according to the expression given by the likelihood function, but based on the adjusted value of beta:

$$\bar{\lambda} = n / T^{\bar{\beta}}$$

The covariance matrix for the parameters is estimated using the Fisher information matrix. See "Parameter Estimates for Crow-AMSAA Models". However, the bias-corrected estimates for λ and β are substituted for the MLEs in the resulting formulas. All confidence bands in plots and confidence intervals in reports are based on this procedure.

Profilers

For the Crow-AMSAA models, the estimates for the MTBF, Intensity, and Cumulative Events given in the profilers are obtained by replacing the parameters λ and β in their theoretical expressions by their MLEs. In the case of the Crow-AMSAA with Modified MLE option, the modified MLEs are used. Confidence limits are obtained by applying the delta method to the log of the expression of interest.

For example, consider the cumulative events function. The cumulative number of events at time t since testing initiation is given by $N(t) = \lambda t^{\beta}$. It follows that $\log(N(t)) = \log(\lambda) + \beta\log(t)$. The parameters λ and β in $\log(N(t))$ are replaced by their MLEs (or modified MLEs) to estimate $\log(N(t))$. The delta method is applied to this expression to obtain an estimate of its variance. This estimate is used to construct a 95% Wald-based confidence interval. The resulting confidence limits are then transformed using the exponential function to give confidence limits for the estimated cumulative number of events at time t.

Statistical Details for the Piecewise Weibull NHPP Change Point Detection Report

The change point is estimated as follows:

- Using consecutive event times, disjoint intervals are defined.
- Each point within such an interval can be considered to be a change point defining a piecewise Weibull NHPP model with two phases. So long as the two phases defined by that point each consist of at least two events, the algorithm can compute MLEs for the parameters of that model. The log-likelihood for that model can also be computed.
- Within each of the disjoint intervals, a constrained optimization routine is used to find a local optimum for the log-likelihood function.
- These local optima are compared, and the point corresponding to the largest is chosen as the estimated change point.

Note that this procedure differs from the grid-based approach described in Guo et al. (2010).

Chapter 11

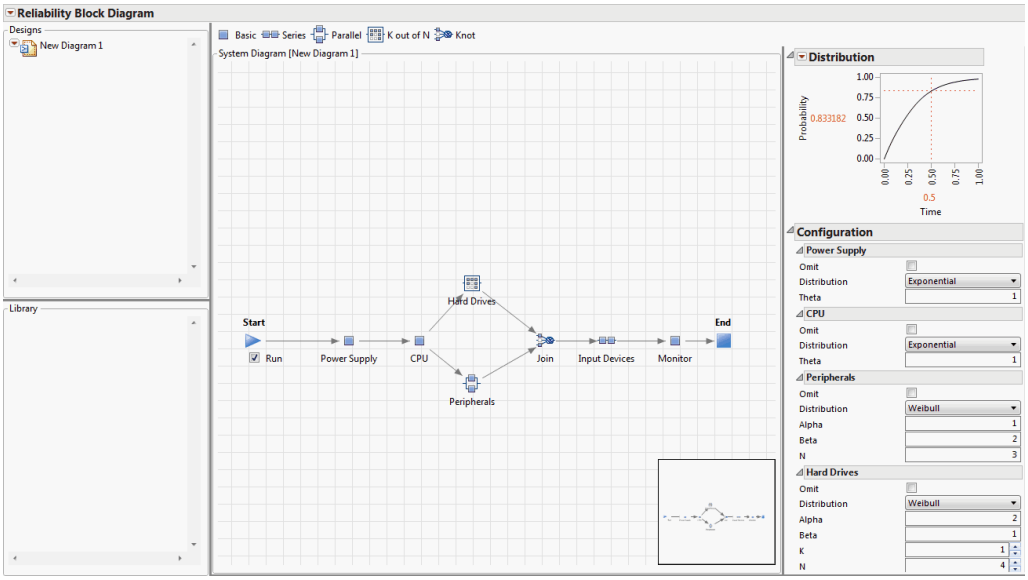
JMP PRO Reliability Block Diagram

Engineer System Reliabilities

The Reliability Block Diagram platform is available only in JMP Pro.

The Reliability Block Diagram platform displays the reliability relationships among a system's components. If reliability distributions are assigned to the components, the platform analytically models the reliability behavior. A Reliability Block Diagram (also known as a dependence diagram) illustrates how component reliability contributes to the success or failure of a system.

Figure 11.1 Example of a Reliability Block Diagram



Contents

Overview of the Reliability Block Diagram Platform 291
Example Using the Reliability Block Diagram Platform 291
Add Components 293
Align Shapes 295
Connect Shapes 296
Configure Components 297
The Reliability Block Diagram Window 299
Preview Window 300
Reliability Block Diagram Platform Options 301
Workspace Options 304
Configuration Settings 305
Distribution Configurations 305
Specify a Nonparametric Distribution 307
Options for Design and Library Items 308
Profilers 309
Component Importance and Time to Failure 311
Component Plots 313
Print Algebraic Reliability Formula 316
Generate Algebraic Expression Data Table 317
Clone and Delete 317

JMP PRO Overview of the Reliability Block Diagram Platform

The Reliability Block Diagram platform enables you to diagram a system and its related components to show how component reliability affects the success of the whole system. Each block in a Reliability Block Diagram represents a component in the system and is connected to other components in the system.

A Reliability Block Diagram can include components connected either in series or in parallel. A failure in any series component causes the entire system to fail. In a parallel-connected (or redundant) system, all parallel components must fail for the entire system to fail. In addition, the diagram can include K out of N components where *k*-out-of-*n* components must function for the system to function.

The Reliability Block Diagram template places a Start block on the left side and an End block on the right side of the diagram. Using the Shape tools, you diagram the system to be analyzed beginning at the Start block and connecting components to reach the End block.

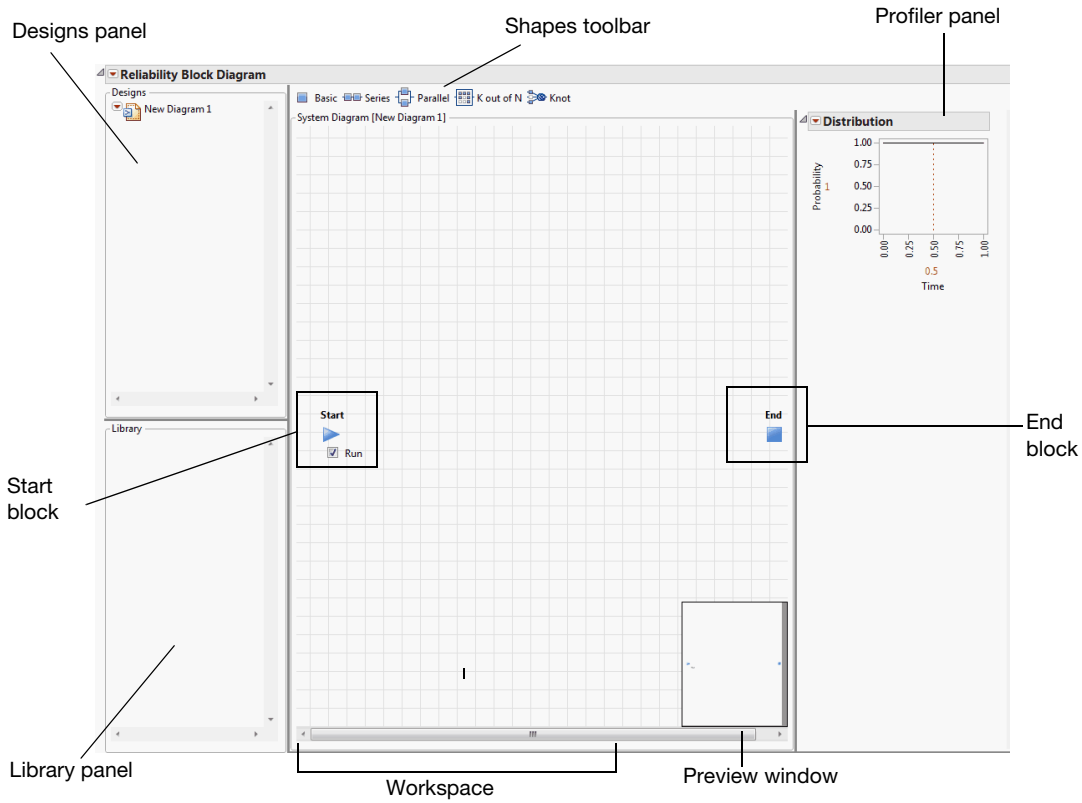
JMP PRO Example Using the Reliability Block Diagram Platform

In this example, you learn how to create a new Reliability Block Diagram.

1. Select **Analyze > Reliability and Survival > Reliability Block Diagram**.

   A blank Reliability Block Diagram window appears.

Figure 11.2 New Reliability Block Diagram



Note: The Distribution profiler appears by default.

2. In the Designs panel, select and rename New Diagram 1 to Computer.

   The Workspace is now named System Diagram [Computer].

3. Deselect **Run** in the Start block.

   With **Run** selected, the platform updates the diagram's reliability calculations after each change to the system diagram. These changes can include adding or deleting components, changing a component's configuration, and adding or deleting a connection.

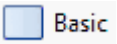
   With **Run** deselected, the platform does not update the reliability calculations after any changes.

Tip: Deselect **Run** when you are diagramming large systems. Select **Run** when the diagram is complete.

4. Proceed with "Add Components".

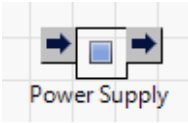
JMP PRO Add Components

The Reliability Block Diagram drawing elements that are located in the toolbar are called *shapes*. The term *component* refers to a shape that represents a constituent part of the system.

1. Click the Basic icon 

on the Shape toolbar and drag the shape to the System Diagram to the right of the Start block.
2. Select the label, replace New Basic 1 by Power Supply, and press Enter.

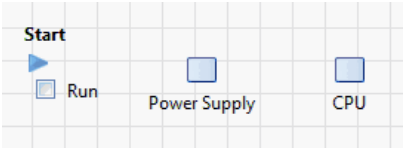
Figure 11.3 Basic Shape



When you click the label or the shape, connection arrows appear. The arrows disappear when you click elsewhere in the template.

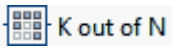
3. Drag a second Basic shape to the right of the Power Supply shape.
4. Select the label and type CPU.

Figure 11.4 Example System Diagram



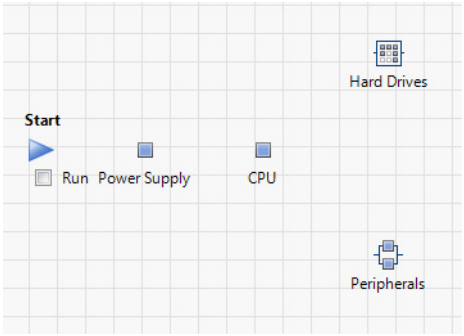
Note: You will align the shapes later, in the section "Align Shapes".

5. Drag a Parallel shape

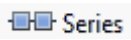
to a position to the right and below the CPU shape.
6. Select the label and type Peripherals.
7. Drag a K out of N shape 

to a position to the right and above the CPU shape.
8. Select the label and type Hard Drives.

Figure 11.5 Partial System Diagram

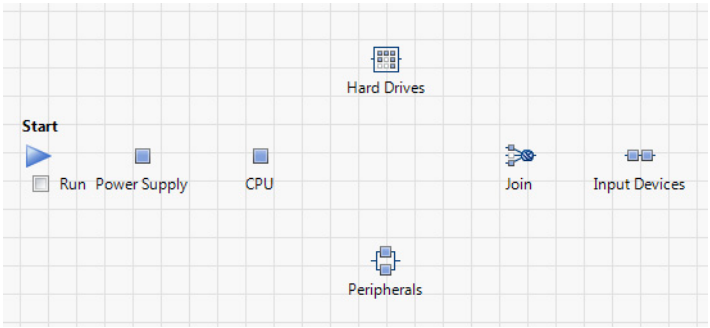


9. Drag a Knot shape

to the right of the previous shapes.
10. Select the label and type Join.
11. Drag a Series shape 

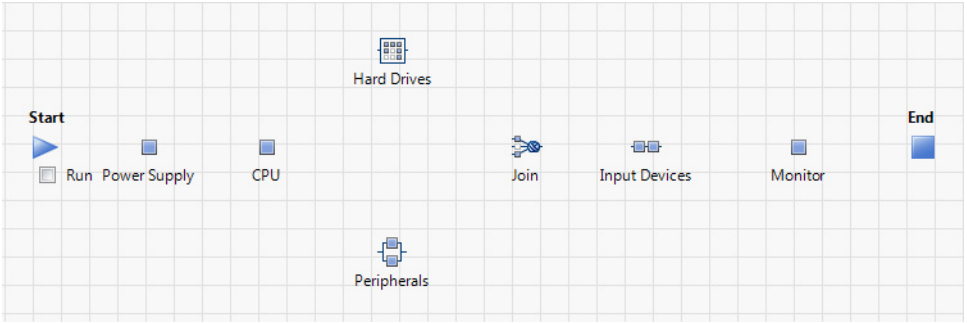
to a position to the right of the Knot shape.
12. Select the label and type Input Devices.

Figure 11.6 Partial System Diagram



13. Drag a Basic shape to a position to the right of the Input Devices shape.
14. Select the label and type Monitor.

Figure 11.7 System Diagram Showing All Shapes



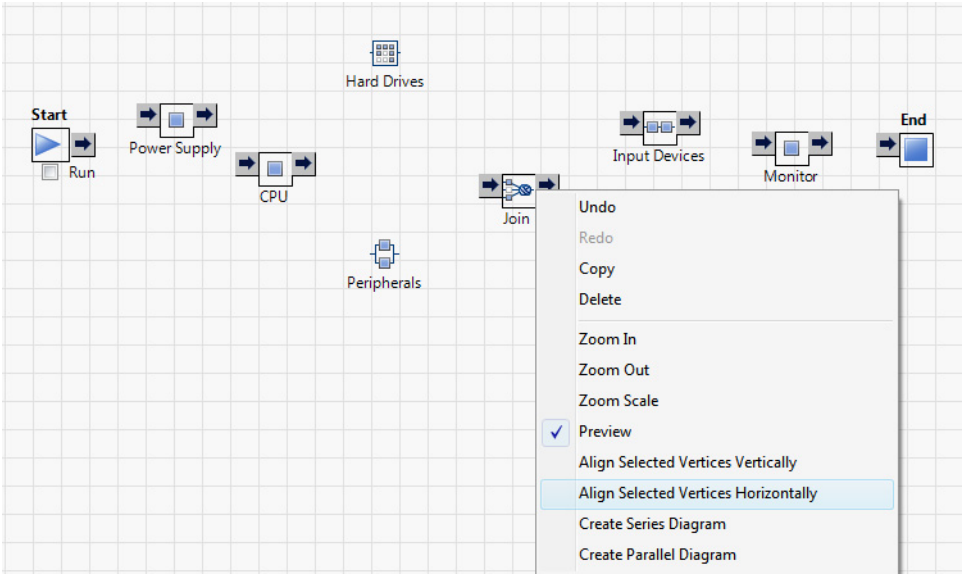
15. Proceed with "Align Shapes".

JMP PRO Align Shapes

1. To vertically align the shapes for Hard Drives and Peripherals, select the components:
   - Hard Drives
   - Peripherals

   Tip: To select shapes, drag the cursor around the shapes or press Shift and click each shape.
2. With the shapes selected, right-click one of the shapes and select **Align Selected Vertices Vertically**.
3. To horizontally align the remaining shapes, select the following components:
   - Start
   - Power Supply
   - CPU
   - Join
   - Input Devices
   - Monitor
   - End
4. With the shapes selected, right-click one of the shapes and select **Align Selected Vertices Horizontally**.

Figure 11.8 Align Shapes Horizontally



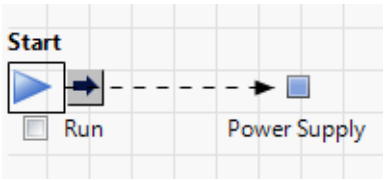
5. Proceed with "Connect Shapes".

JMP PRO Connect Shapes

To connect shapes, select a shape to display its connection arrows. Suppose you want to connect shape A to shape B. Select shape A. Drag the right arrow to shape B to indicate that shape A precedes shape B. Drag the left arrow to shape B to indicate that shape B precedes shape A. To connect the shapes in your diagram, select the right arrows to connect to the next shape in the sequence.

1. Select the Start block (blue arrow) to display the connection arrow.
2. Select the single connection arrow and drag it to the Power Supply component.

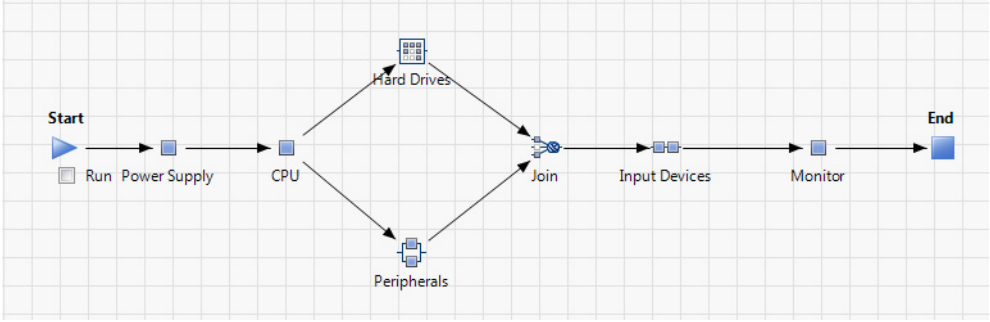
Figure 11.9 Connecting Shapes



3. For each of the following components, click the first component, select its right connection arrow, and drag the arrow to the second component:
    1. Power Supply → CPU

2. CPU → Hard Drive
3. CPU → Peripherals
4. Hard Drives → Join
5. Peripherals → Join
6. Join → Input Devices
7. Input Devices → Monitor
8. Monitor → End block

Figure 11.10 Completed System Diagram



4. Proceed with "Configure Components".

JMP PRO Configure Components

1. In the Configuration panel, enter the Configuration settings for the components. See "Configuration Settings".

Table 11.1 Configuration Settings

Component	Settings
Power Supply	• Distribution—Exponential • Theta—1
CPU	• Distribution—Exponential • Theta—1

Table 11.1 Configuration Settings *(Continued)*

Component	Settings
Peripherals	• Distribution—Weibull • Alpha—1 • Beta—2 • N—3
Hard Drives	• Distribution—Weibull • Alpha—2 • Beta—1 • K—1 • N—4
Join	Minimum available—1
Input Devices	• Distribution—Fréchet • Location—0 • Scale—1 • N—2
Monitor	• Distribution—Exponential • Theta—1

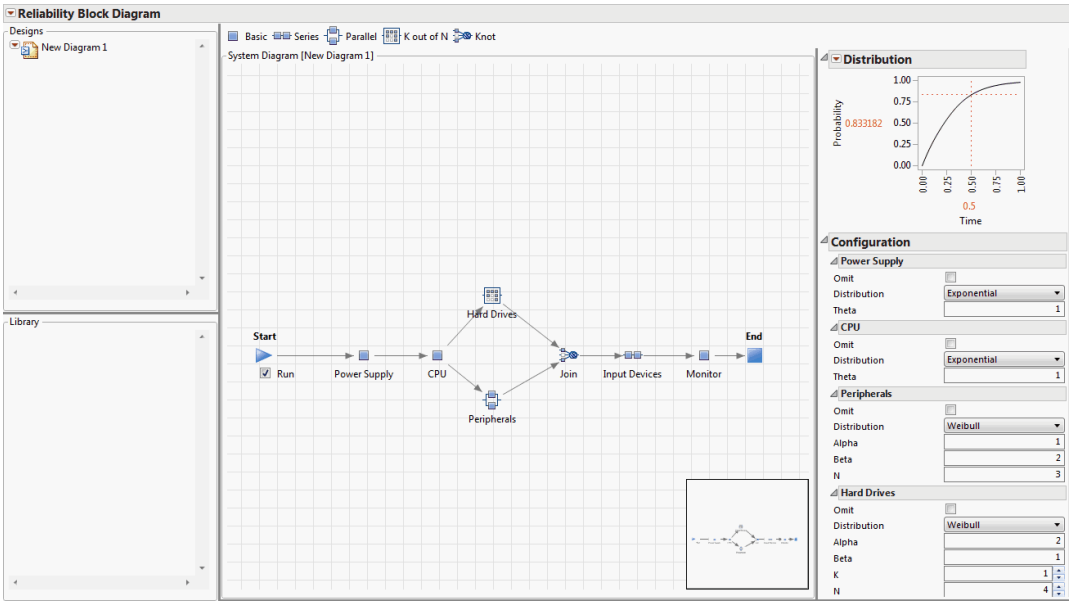
The Reliability Block Diagram is complete (Figure 11.11).

2. Select **Run**.

   The system's reliability information is updated. This is reflected in the Distribution plot in the Profiler pane.

3. To save the Reliability Block Diagram as a JMP Scripting Language (JSL) file, select **File > Save** and name it exampleRBDcomplete.jsl.

Figure 11.11 Example Reliability Block Diagram



The Reliability Block Diagram Window

The Reliability Block Diagram window is divided into the following panels:

- Designs: Lists system diagrams that were created using the Reliability Block Diagram platform.
- Library: Lists sub-system designs that are available for reuse in creating new system diagrams.

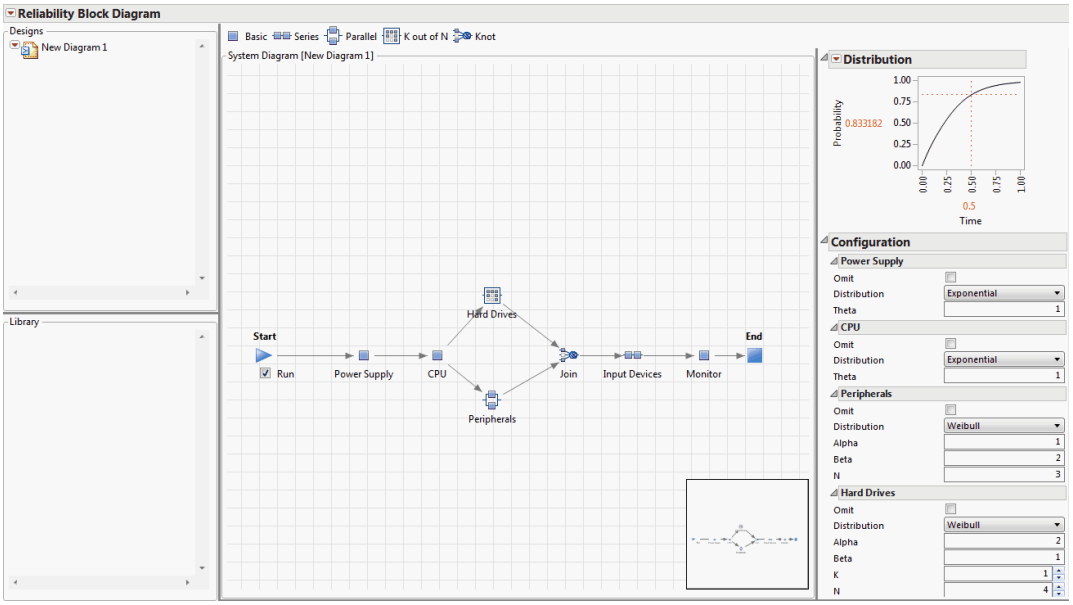
Note: Each system diagram that appears in the Designs and the Library panels contains its own red triangle menu. The options in each of those red triangle menus are described in "Options for Design and Library Items".

- Workspace: Displays the Shape toolbar, the System Diagram, and the Preview window.

Tip: To hide the Preview window, right-click in the Workspace and deselect Preview.

- Profiler Panel: Displays the Distribution profiler, Configuration settings, and various system and component profilers and plots that you select from the Diagram red triangle menu.

Figure 11.12 Example of a Reliability Block Diagram



The Shape toolbar includes the following drawing tools:

Basic Adds a single block shape to the system diagram.

Series Adds a series block shape that represents a group of components connected in a series. All components must function for the system to function.

Parallel Adds a parallel block shape that represents a group of components that are in parallel. At least one of the components must function for the system to function.

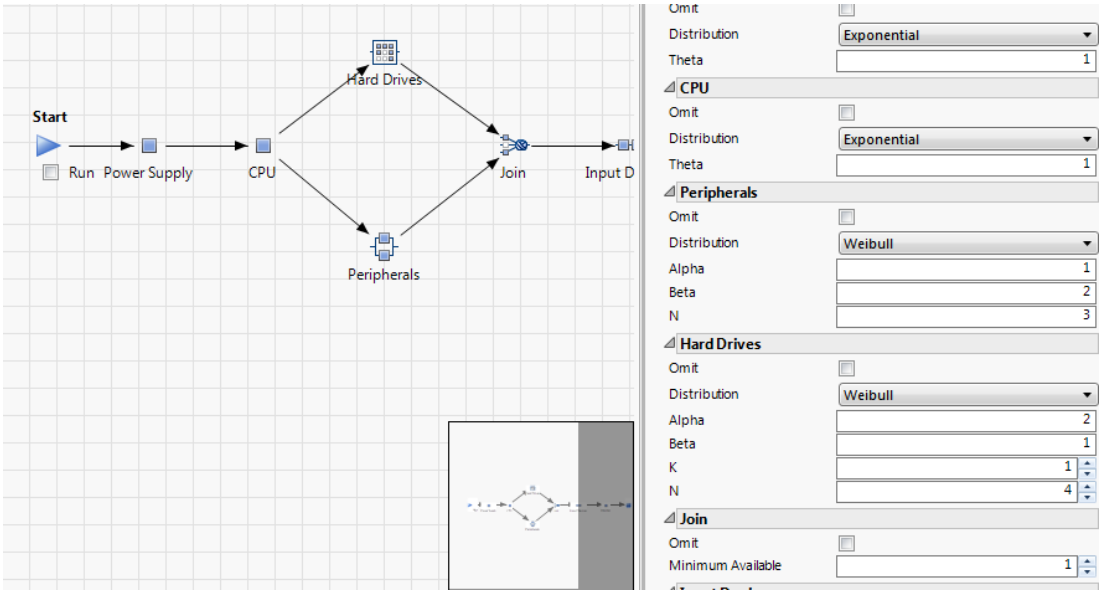
K out of N Adds a *k*-out-of-*n* block shape where you specify *k* and *n*. At least k of the components must function for the system to function.

Knot Adds a knot shape to the system diagram that enables you to join Parallel or K out of N components that have different Distribution property settings.

JMP PRO Preview Window

When the system diagram is too large to appear in the workspace, the Preview window enables you to reposition the portion of the system diagram that appears in the workspace. The viewing area is indicated in the Preview window by a white background. Drag the viewing area to reposition the view of the Workspace to another part of the system diagram.

Figure 11.13 Preview Window with Visible Part of Diagram on White Background



JMP PRO Reliability Block Diagram Platform Options

The Reliability Block Diagram red triangle menu contains the following options for the Reliability Block Diagram window:

Save and Save As Enables you to save a Reliability Block Diagram, or save an existing Reliability Block Diagram with a new name, to a JMP Scripting Language (JSL) script that is automatically executed when it is opened in JMP. See the *Scripting Guide* for more information about Auto-Submit scripts.

Note: The Save and Save As red triangle options are equivalent to File > Save and File > Save As. They are available in the red triangle menu for convenience.

Show Design Diagram If **Show Design Comparisons** is selected, this option enables you to hide or show the system diagram in the Workspace.

Show Design Comparisons Displays a Distribution Overlay profiler and Remaining Life Distribution Overlay profiler for the selected system diagrams. See "Show Design Comparisons".

Import Component Distribution Settings Enables you to import configuration settings for the system diagram from a data table. The table must contain columns for the diagram category, diagram name, component name, distribution, and one or more parameters. The number of parameters depends on the specified distribution.

Note: The strings in the imported table must be exact matches to the strings in the system diagram.

New Design Item Enables you to add a new design diagram to the Designs panel. See "Add a New Design Item".

New Library Item Enables you to add a new item to the Library panel. The Library panel lets you create subsystem diagrams for use in multiple system designs. See "Add a New Library Item".

Note: Library items are available only for use in diagrams that are contained in the Designs panel of a JSL file. They are not available to other diagrams in other JSL files.

JMP PRO Show Design Comparisons

The **Show Design Comparisons** option displays a Distribution Overlay plot and a Remaining Life Distribution Overlay plot. These plots appear in the Workspace below the System Diagram. See Figure 11.14 and Figure 11.15.

- The Distribution Overlay plot shows the probability of system failure for each of the designs in the Designs panel.
- The Remaining Life Distribution Overlay plot shows the conditional probability of failure for each design in the Designs panel, given that the systems have survived to a specified survival time. Enter the Survival Time in the box to the right of the plot. Alternatively, select the small rectangle at the origin and drag it to the right to dynamically set the Survival Time.

Check boxes enable you to select which designs are represented in the plots. This enables you to compare a subset of designs with respect to failure probabilities.

Tip: To hide the system diagram in the Workspace, click the Reliability Block Diagram red triangle and deselect **Show Design Diagram**. Alternatively, to show more of the Design Comparisons, reposition the horizontal splitter upward.

Figure 11.14 Distribution Overlay Example

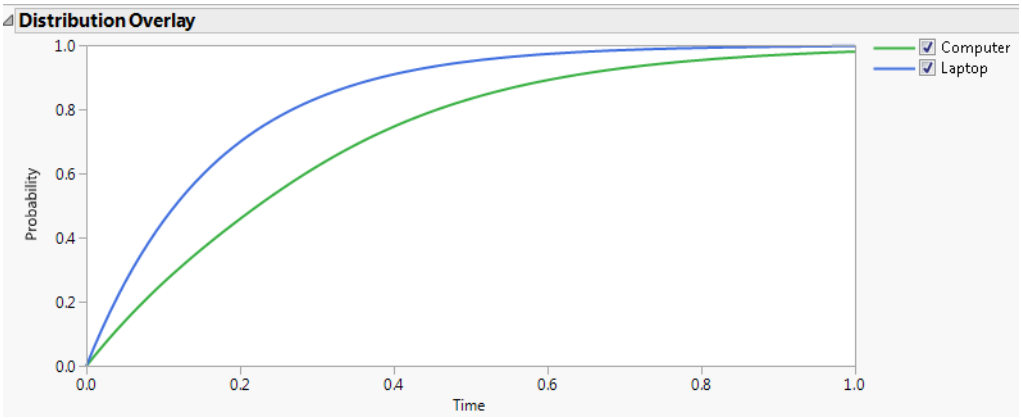
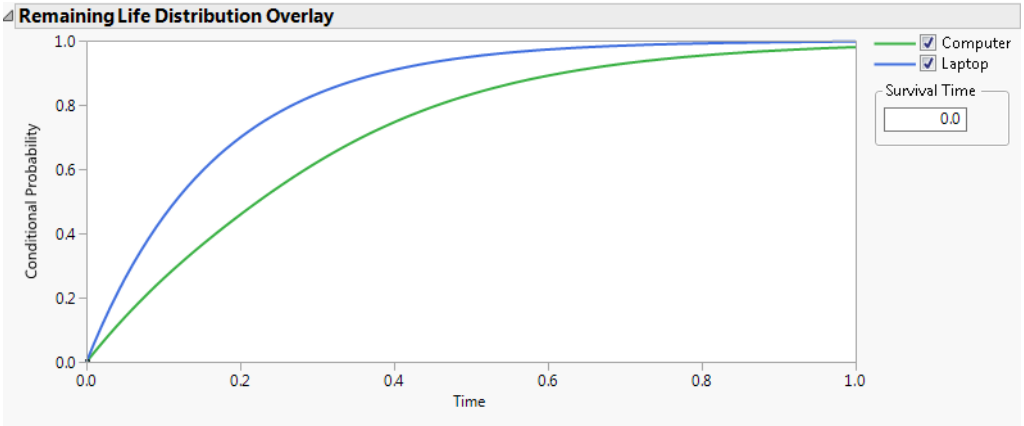


Figure 11.15 Remaining Life Distribution Overlay Example



JMP PRO Add a New Design Item

The **New Design Item** option adds a new system design to the Designs panel list.

- Click the Reliability Block Diagram red triangle and select **New Design Item**.
- A design named New Diagram X, where X is a number that identifies the diagram name, is added to the Designs panel list.
- To name the design, select the label and enter a name.
- Use the Shape toolbar to draw the system diagram. See "Example Using the Reliability Block Diagram Platform".

Tip: You can also copy and paste the body of a diagram from another block diagram template. The Start and End shapes are not copied. After you paste the diagram, you must reconnect the Start and End shapes.

- Configure the components.
- Save the file.

To display a design in the System Diagram window and to view its Profiler panel, double-click its icon in the Designs panel.

JMP PRO Add a New Library Item

The **New Library Item** option adds a new sub-system to the Library panel list.

Tip: Drag a system design from the Designs panel to the Library panel to add it to the Library as a sub-system.

- Click the Reliability Block Diagram red triangle and select **New Library Item**.
- A sub-system called New Diagram X, where X is a number that identifies the diagram name, is added to the Library panel list.
- To name the sub-system, select the label and enter a name.
- Use the Shape toolbar to draw the sub-system diagram.
- Configure the components.
- Create connections from the Start block to the End block.
- Save the file.

JMP PRO Workspace Options

The Workspace panel supports several right-click commands for adjusting the view of the panel. Here are some of the available commands:

Zoom In Enables you to zoom in to the system diagram.

Zoom Out Enables you to zoom out from the system diagram.

Tip: On Windows, you can press Ctrl and use the mouse scroll wheel to zoom in to and out from the diagram.

Zoom Scale Opens the Set Zoom Scale window enabling you to zoom in or out using a scaling factor that is specified as a decimal. By default, the zoom scale is set to 1 (100%).

Preview Enables you to hide or show the Preview window in the Workspace panel.

Align Selected Vertices Vertically Enables you to align shapes on a vertical line.

Align Selected Vertices Horizontally Enables you to align shapes on a horizontal line.

Create Series Diagram Enables you to create a series diagram using the nodes in the diagram. The order of the series is determined by the order of node creation. This option is available only when the diagram does not contain any arrows.

Create Parallel Diagram Enables you to create a parallel diagram using the nodes in the diagram. This option is available only when the diagram does not contain any arrows.

JMP PRO Configuration Settings

Each component in a Reliability Block Diagram can be assigned a failure distribution. The available failure distributions are listed in Figure 11.2. To see the formulas and parameterization for these failure distributions, see "Distributions".

JMP PRO Distribution Configurations

When a component is added to the diagram, an outline for that component appears beneath the Configuration outline.

Note: You can omit a component from the analysis by checking the box next to Omit.

Select a Distribution and enter the required parameter values.

Table 11.2 Distributions and Parameters

Property Type	Required Inputs
Exponential	Theta
Weibull	Alpha, Beta
Lognormal	location, scale
Loglogistic	location, scale
Fréchet	location, scale
GenGamma	mu, sigma, lambda
DS Weibull	Alpha, Beta, Defective Probability
DS Lognormal	location, scale, Defective Probability

Table 11.2 Distributions and Parameters *(Continued)*

Property Type	Required Inputs
DS Loglogistic	location, scale, Defective Probability
DS Fréchet	location, scale, Defective Probability
Nonparametric	data or data file

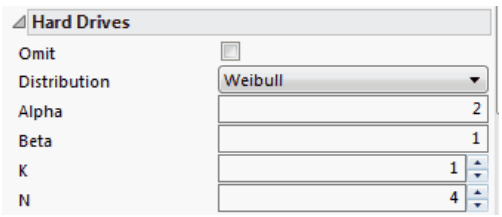
To view Configuration settings for the components in a selected design or subsystem, do the following:

- To view the Configuration settings for a specific component, select the component's shape.
- To view Configuration settings for more than one component, select multiple components' shapes using the Arrow tool or pressing Control and clicking.
- To view Configuration settings for all components, deselect any shapes by clicking in a blank portion of the Workspace.

For each component in the diagram, you can either omit the component from calculations by checking the box next to Omit or:

- From the Distribution list, select the appropriate distribution. For all selections other than Nonparametric, enter parameter values for the distribution. For Nonparametric, see "Specify a Nonparametric Distribution".
- For Series and Parallel components, you must also enter a value for N, the total number of components contained in the series or parallel shape.
- For K out of N components, you must also enter K, the minimum number of components that must function for the system to function, and N, the total number of components in the shape.
- For Knot components, enter the Minimum Available Dependences. The Knot shape enables you to configure a *k*-out-of-*n* shape where the shapes that are joined have different distributions. The Minimum Number of Dependencies is *k*, the minimum number of paths leading to the Knot that need to function in order for the system to function.

Figure 11.16 Example of a Weibull Configuration for a K out of N Shape

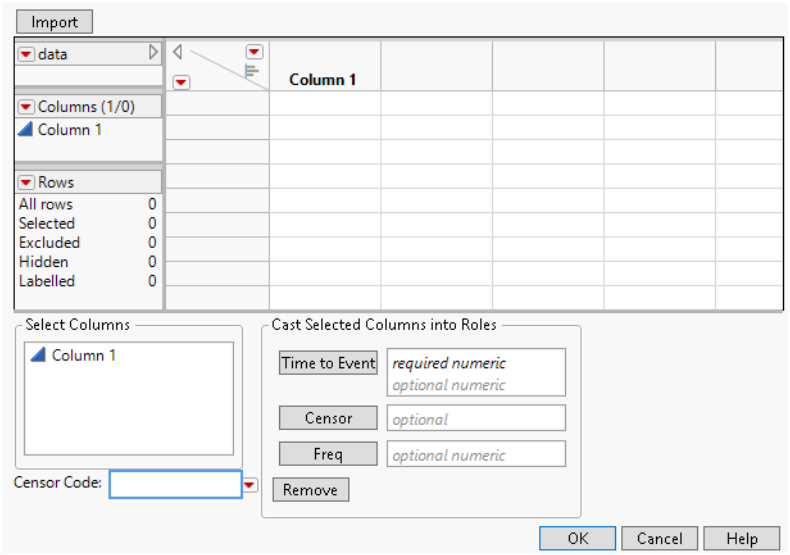


JMP PRO Specify a Nonparametric Distribution

The Nonparametric option under Distribution enables you to approximate an arbitrary distribution. Enter data or import a file containing a large set of data. This data is used to approximate the distribution.

After selecting Nonparametric, click the icon next to Data. The Provide Data window appears, enabling you to either enter data or import a data file. Once you have imported or entered your data, the data is used to calculate a nonparametric distribution for the component.

Figure 11.17 Provide Data Window



To import data from a file, do the following.

1. Open the JMP data table that contains the data to import.
2. In the Provide Data window, click **Import**.

   The Select Data Table window appears.
3. From the Data Table list, select the data table.
4. Click **OK**.
5. In the panel beneath the data grid, specify which columns represent Time to Event data, as well as Censor and Freq data if appropriate.
6. In the Provide Data window, click **OK**.

To enter data manually, do the following.

1. Create columns for Time to Event data, as well as Censor and Freq data, if appropriate.

2. Enter the data into the columns.
3. In the panel beneath the data grid, specify which columns represent Time to Event data, as well as Censor and Freq data if appropriate.
4. In the Provide Data window, click **OK**.

JMP PRO Options for Design and Library Items

The available options for Design and Library items are listed below:

- Show Configuration shows or hides the Configuration outline. See "Configuration Settings".
- Options to show various profilers. See "Profilers" and the following:
  - "Distribution Profiler"
  - "Remaining Life Distribution Profiler"
  - "Reliability Profiler"
  - "Quantile Profiler"
  - "Density Profiler"
  - "Hazard Profiler"
  - "Cumulative Hazard Profiler"
- Options to show plots for component importance measures and mean time to failure. See "Component Importance and Time to Failure" and the following:
  - "Birnbaum's Component Importance"
  - "Remaining Life BCI"
  - "Component Integrated Importance"
  - "Mean Time to Failure"
- Options to show overlay plots for the components in a system diagram. See "Component Plots" and the following:
  - "Component Distribution Functions"
  - "Component Reliability Functions"
  - "Component Density Functions"
  - "Component Hazard Functions"
  - "Component Cumulative Hazard Functions"
- "Print Algebraic Reliability Formula"
- "Generate Algebraic Expression Data Table"

- "Clone and Delete"

JMP PRO Profilers

Various profilers are provided to help you analyze the reliability properties of a system. You can view profilers for each system diagram that is listed in the Designs and the Library panels. This section describes the profilers. The profilers appear in the Profilers panel of the report.

Red Triangle Options for Profilers

Each profiler has a red triangle menu that contains the following commands:

Reset Factor Grid Displays a window where you can enter a current setting and values that control the display. See *Profilers* for more information about setting the factor grid.

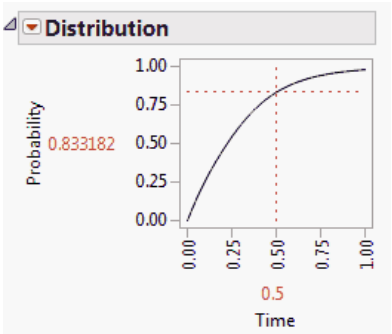
Factor Settings Select this option to configure the profiler's settings and to link the profilers. See *Profilers* for more information about Factor Settings.

Note: Adjust the X and Y axes of a profiler to view the desired portion of the graph.

JMP PRO Distribution Profiler

The Distribution Profiler displays the probability that the system fails as a function of time. The Distribution profiler for the system appears by default.

Figure 11.18 Distribution

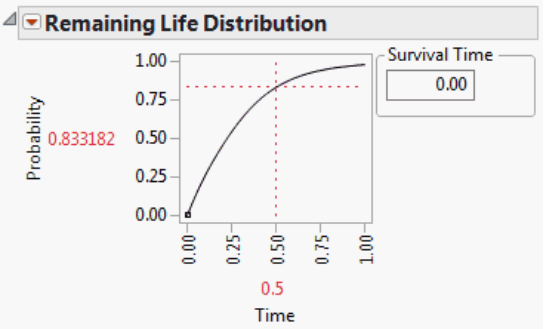


JMP PRO Remaining Life Distribution Profiler

The Remaining Life Distribution profiler for the system shows the probability that the system fails given that it has survived a specified amount of time, designated as Survival Time. By default, Survival Time is set to zero, and the remaining life distribution function is equivalent to the distribution function.

Enter a value for **Survival Time** to indicate the time to which the system has survived without failing. As an alternative to entering a value, select the small rectangle at the origin of the graph and drag it to the right to dynamically set the Survival Time.

Figure 11.19 Remaining Life Distribution



JMP PRO Reliability Profiler

The Reliability profiler for the system shows the probability that the system will function as a function of time. The reliability function is also known as the survival function.

JMP PRO Quantile Profiler

The Quantile profiler for the system shows time as a function of the failure probability. Note that the Quantile function is the inverse of the Distribution function.

JMP PRO Density Profiler

The Density profiler for the system shows the probability density function associated with the system's failure distribution function.

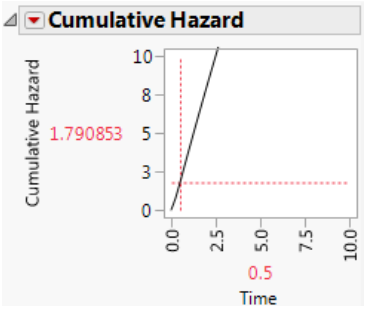
JMP PRO Hazard Profiler

The Hazard profiler for the system shows the instantaneous failure rate at a given time.

JMP PRO Cumulative Hazard Profiler

The Cumulative Hazard profiler for the system shows the cumulative hazard function as a function of time.

Figure 11.20 Cumulative Hazard



JMP PRO Component Importance and Time to Failure

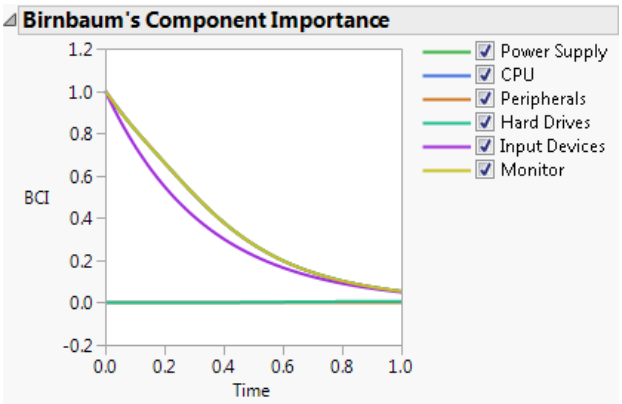
Plots that analyze component importance and mean time to failure are provided for each system diagram that is listed in the Designs and the Library panels. This section describes the component importance measures and mean time to failure. These plots and statistics appear in the profilers panel of the report.

Note: Check boxes in the legend of each component importance plot enable you to select which components to view in that plot.

JMP PRO Birnbaum's Component Importance

Select **Show BCI** to show an overlay plot of the Birnbaum's Component Importance measures for each component of the selected system diagram. A component's BCI at a given time is the probability that the system fails if the component fails. A component with a large BCI is critical to system reliability.

Figure 11.21 Birnbaum's Component Importance

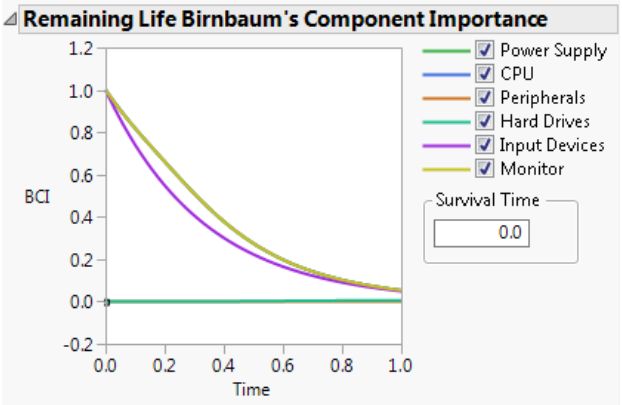


JMP PRO Remaining Life BCI

Select **Show Remaining Life BCI** to show an overlay plot of the Birnbaum's Component Importance for Remaining Life. The BCI for Remaining Life is the probability that the system fails if the component fails, given that the system has survived a specified amount of time, designated as Survival Time. By default, Survival Time is set to zero, and the BCI for Remaining Life is equivalent to the Birnbaum's Component Importance.

Enter a value for **Survival Time** to indicate the time to which the system has survived without failing. As an alternative to entering a value, select the small rectangle at the origin of the graph and drag it to the right to dynamically set the Survival Time.

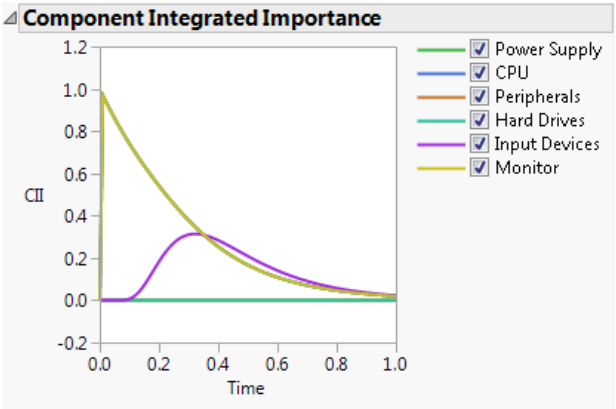
Figure 11.22 Birnbaum's Component Importance for Remaining Life



JMP PRO Component Integrated Importance

Select **Show Component Integrated Importance** to show an overlay plot of the integrated importance measures for the components of the Reliability Block Diagram. The integrated importance measure for each component takes into account the failure rate of the component as well as the likelihood of failing instantaneously. See Si et al. (2012).

Figure 11.23 Component Integrated Importance



JMP PRO Mean Time to Failure

Select **Show MTTF** to view the Mean Time to Failure (MTTF) for the system.

Note: The formula that is used to calculate the Mean Time to Failure depends on the specified failure distributions and Configuration settings for each component in the system.

JMP PRO Component Plots

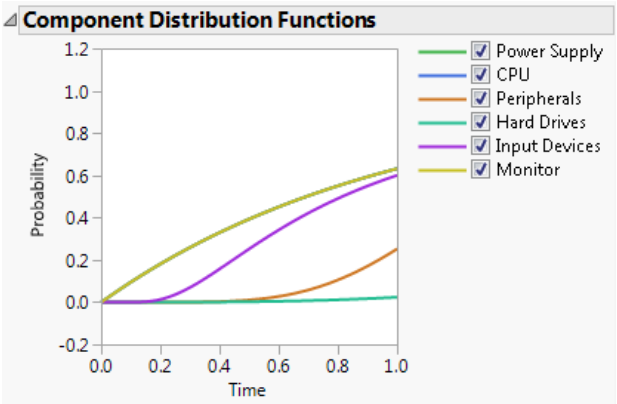
A system diagram usually contains many individual components. The reliability functions of each of these components can be examined in overlay plots. You can view component overlay plots for each system diagram that is listed in the Designs and the Library panels. This section describes the component overlay plots. These plots appear in the profilers panel of the report.

Note: Check boxes in the legend of each component plot enable you to select which components to view in that plot.

JMP PRO Component Distribution Functions

Select **Show Component Distribution Functions** to show an overlay plot of the distribution functions for the components of the Reliability Block Diagram.

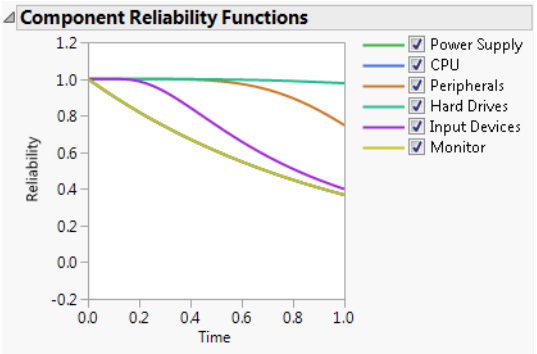
Figure 11.24 Component Distribution Functions



JMP PRO Component Reliability Functions

Select **Show Component Reliability Functions** to show an overlay plot of the reliability functions for the components of the Reliability Block Diagram.

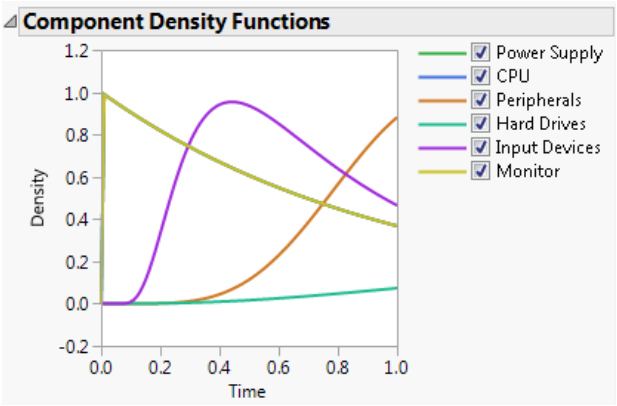
Figure 11.25 Component Reliability Functions



JMP PRO Component Density Functions

Select **Show Component Density Functions** to show an overlay plot of the density functions for the components of the Reliability Block Diagram.

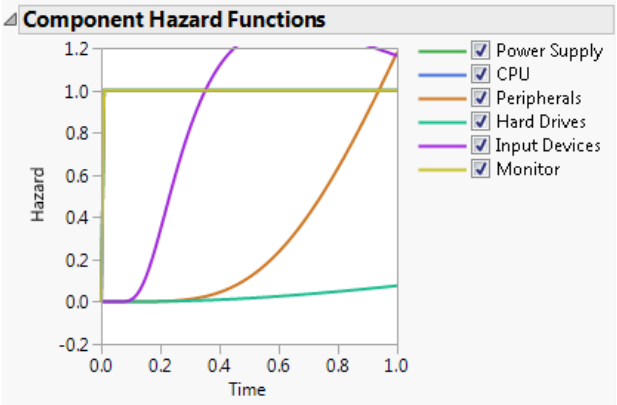
Figure 11.26 Component Density Functions



JMP PRO Component Hazard Functions

Select **Show Component Hazard Functions** to show an overlay plot of the hazard functions for the components of the Reliability Block Diagram.

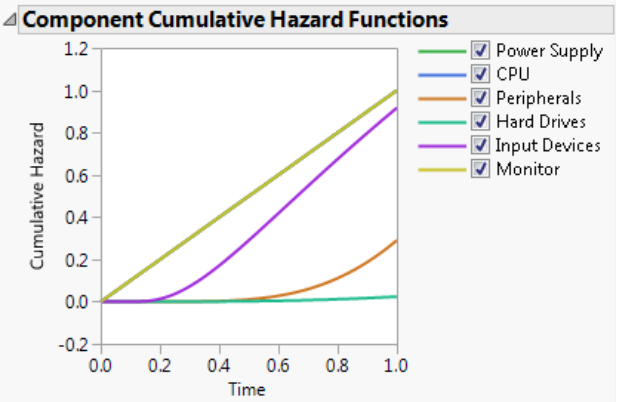
Figure 11.27 Component Hazard Functions



JMP PRO Component Cumulative Hazard Functions

Select **Show Component Cumulative Hazard Functions** to show an overlay plot of the cumulative hazard functions for the components of the Reliability Block Diagram.

Figure 11.28 Component Cumulative Hazard Functions



JMP PRO Print Algebraic Reliability Formula

The Print Algebraic Reliability to the Log Window option prints the reliability formula for the selected system diagram to the Log window.

1. (Windows only) Select **View > Log**.

   Or

   (macOS only) Select **Window > Log**.
2. Open the exampleRBDcomplete.jsl file that you created.
3. In the Designs panel, select **Computer**.
4. Click the Computer red triangle and select **Print Algebraic Reliability to the Log Window**.

   The Log window displays the formula for the Algebraic Reliability of the selected block diagram.

Figure 11.29 Algebraic Reliability in Log Window

```
/*:

Reliability Block Diagram[]Algebraic Reliability:
R["Power Supply"] * R["CPU"] * R["Peripherals"] * R["Input Devices"] * R["Monitor"]
 + R["Power Supply"] * R["CPU"] * F["Peripherals"] * R["Hard Drives"] * R[
"Input Devices"] * R["Monitor"]
```

Note: Reliability formulas use “R” to represent a component’s reliability and “F” to represent probability of a component’s failure.

JMP PRO Generate Algebraic Expression Data Table

The Generate Algebraic Expression Data Table option creates a new data table that contains the algebraic reliability expression as a column formula. The table contains a column for each component in the system. The final column contains the system reliability algebraic expression as a JSL column formula that use the preceding columns as arguments.

JMP PRO Clone and Delete

Each system diagram that is listed in the Designs and the Library panels has options that aid in managing the design and library items. This section describes the Clone and Delete operations.

JMP PRO Clone a Design or Library Item

The Clone option creates a new system design or library sub-system identical to the selected system design or library sub-system.

- Select the **Clone** option from the red triangle menu for the design or library item to be copied.

  A new design or library item is added to the Designs or Library list.
- Save the file.

JMP PRO Delete a Design or Library Item

The Delete option removes the selected system design or a library sub-system from the panel.

- Select the **Delete** option from the red triangle menu for the design or library item to be deleted.

  The specified design or library item is deleted from the Designs or Library list.
- Save the file.

Chapter 12

JMP PRO Repairable Systems Simulation

Estimate Outage Time for a Complex System

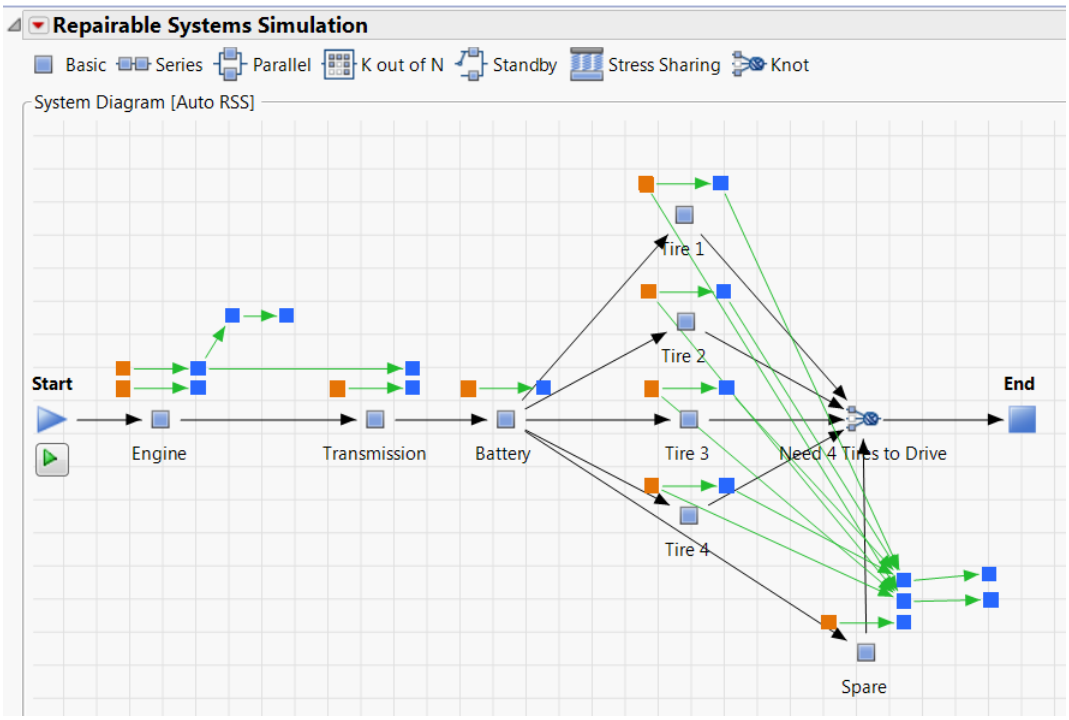
The Repairable Systems Simulation platform is available only in JMP Pro.

The Repairable Systems Simulation (RSS) platform enables you to explore the reliability within complex repairable systems. A *repairable system* consists of individual components that age and require maintenance. The state of a repairable system changes as its components fail and are subsequently repaired.

Repairable systems are involved in many aspects of modern life. They can range in size from refrigerators and houses to power plants and telephone communication networks.

The RSS platform enables you to simulate different system configurations in order to optimize your repairable system. For example, you can extend the useful life of costly components or maximize production outputs while maintaining a safe system operation.

Figure 12.1 Example of a Repairable Systems Simulation Diagram



Contents

Overview of the Repairable Systems Simulation Platform . . . 321
Example Using the Repairable Systems Simulation Platform . . . 321
The Repairable Systems Simulation Window . . . 323
System Diagram . . . 324
Shapes Toolbar . . . 325
Configuration Panel . . . 326
Add Event Panel . . . 329
Add Action Panel . . . 330
Repairable Systems Simulation Platform Options . . . 333
Options for Block Items . . . 334
Distribution Options . . . 334
Specify a Nonparametric or Estimated Distribution . . . 335
Event Settings . . . 336
Action Settings . . . 337
Repairable Systems Simulation Results . . . 338
Results Table . . . 338
Results Explorer . . . 339
Repairable Systems Simulation Results Options . . . 341
RSS Results Explorer Point Estimation Profilers . . . 342

JMP PRO Overview of the Repairable Systems Simulation Platform

The Repairable Systems Simulation (RSS) platform enables you to simulate a repairable system and to analyze its *outage time*. Outage time is the total time a system is not in the On state as a result of either planned maintenance or unintentional failure.

A repairable system is represented by a system diagram. Block shapes in the system diagram represent one or more components that connect to other components. A component that is performing work in the block is said to be *functional*. Components can be connected in series or in parallel. When components are connected in series, the system fails if any of the components in the series fails. When components are connected in parallel, the system fails if all of the parallel components fail.

A component pathway is said to be *uninterrupted* when at least one path between the Start and End blocks does not pass through a block shape failure. During a simulation, the system remains in the On state as long as there is an uninterrupted component pathway. If a block shape failure interrupts the component pathway, the system is set to the Down state. If a block shape fails but the component pathway is not interrupted, the system remains in the On state.

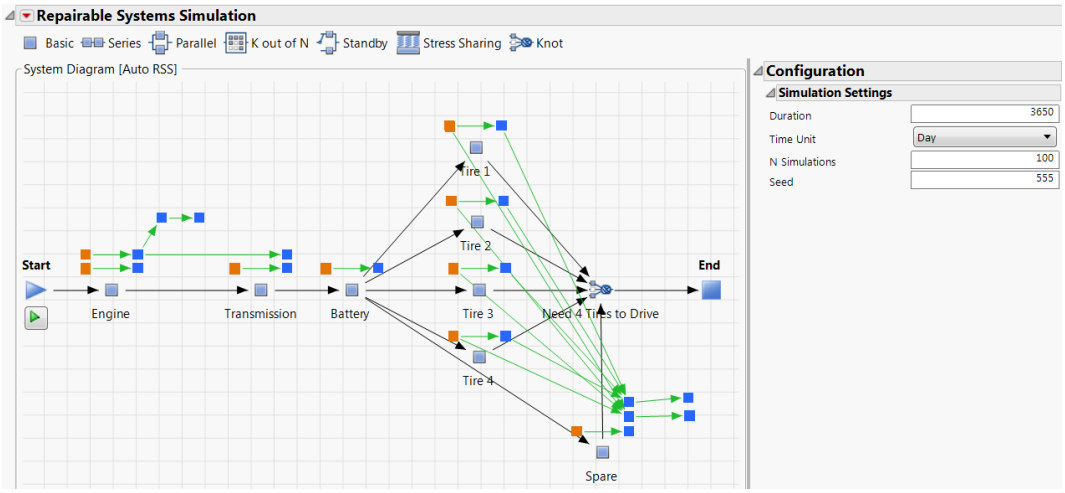
You can add unique events and actions to characterize the repairable nature of individual components. An *event* is a specific occurrence within the system, such as component failure, scheduled maintenance, or the start of a simulation iteration. Events trigger the execution of one or more *actions*, which express how components behave. Actions can alter the state of a component or the entire system.

JMP PRO Example Using the Repairable Systems Simulation Platform

In this example, you run 100 iterations of a simulation of a car system and analyze the system's estimated outage time over the course of 10 years.

1. Select **Help > Sample Data**, click **Open the Sample Scripts Directory**, and open Car Repair Simulation.jsl.

Figure 12.2 Car Repair System Diagram



An RSS window that diagrams a car system appears. In the diagram, the first block on the left is the Start block. Notice in the Configuration panel, on the right side of the RSS window, that the simulation is set to run for 3650 days.

2. (Optional) Enter 555 next to Seed.

   Because the simulation involves random component failures, this action ensures that you obtain the exact results shown below.

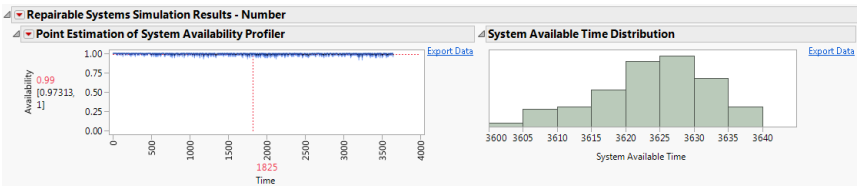
3. Click the green triangle below the Start block to simulate the car system.

   A data table that contains the simulation results appears.

4. Click the green arrow next to the Launch Repairable Systems Simulation Results Explorer script.

   A window appears containing the Repairable Systems Simulation Results report. For more information about how to interpret these results, see "Repairable Systems Simulation Results".

Figure 12.3 Partial RSS Report

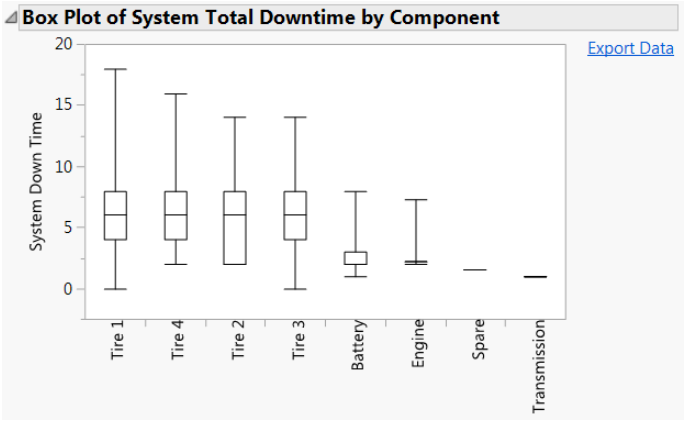


You predict that the car will be available between 3,600 and 3,640 days over the next 10 years. Because the values shown in the Point Estimation of System Availability graph are

close to one, you conclude that the car will be mostly available to drive over the next 10 years. You are interested in which components cause the most downtime for the system.

5. Click the red triangle next to Repairable Systems Simulation Results - Number and select **Box Plot of Total System Downtime by Component**.

Figure 12.4 Partial Box Plot of Total System Downtime by Component Report



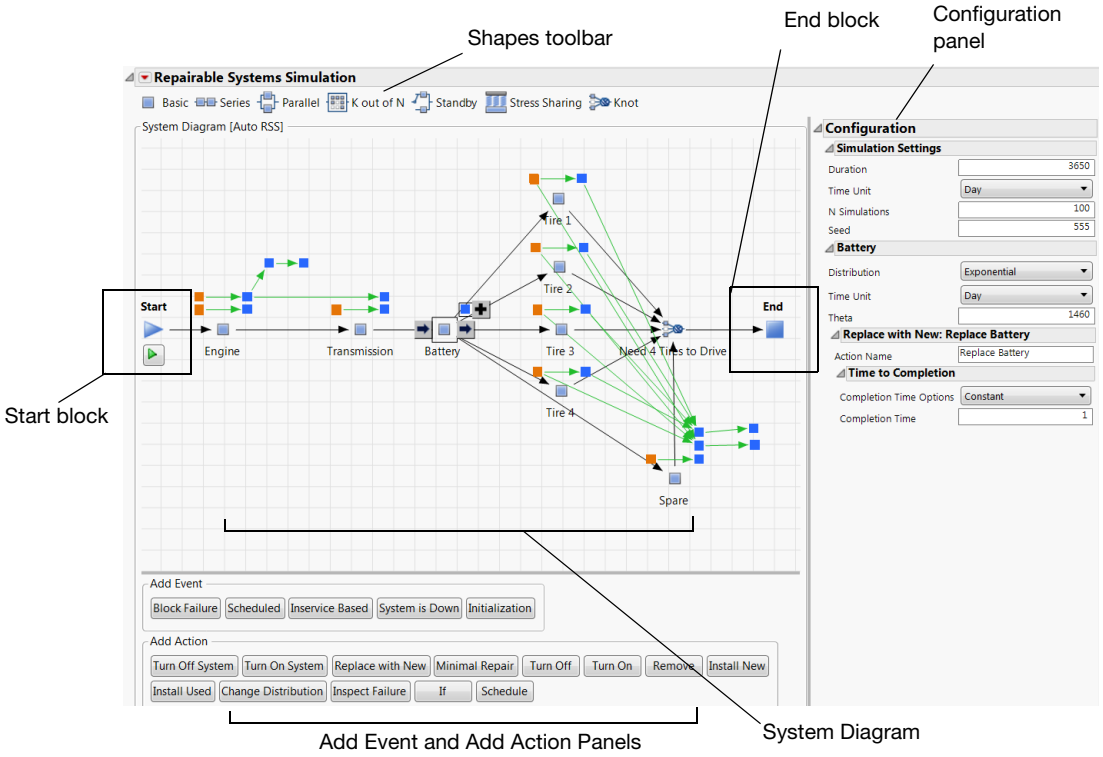
You conclude that the tires cause the most downtime for the car system. To increase the average total system availability, you might consider using more durable tires or carrying another spare tire.

JMP PRO The Repairable Systems Simulation Window

Launch the Repairable Systems Simulation platform by selecting **Analyze > Reliability and Survival > Repairable Systems Simulation**. The Repairable Systems Simulation window is divided into the following panels:

- "System Diagram"
- "Shapes Toolbar"
- "Configuration Panel"
- "Add Event Panel"
- "Add Action Panel"

Figure 12.5 The Repairable Systems Simulation Window



JMP PRO System Diagram

The System Diagram is the space where you can diagram a repairable system. A new System Diagram contains a Start block and an End block.

The following options are available in a pop-up menu when you right-click in the System Diagram:

Run Simulation Runs the simulation using the current Simulation Settings in the Configuration panel. The results of the simulation are reproducible if a nonzero value is specified for Seed.

Run Multithreaded Simulation Runs a multithreaded simulation of the system. The results of the simulation are not reproducible when you run a multithreaded simulation.

Diagram Operation Contains options to control the appearance of the diagram:

Delete (Available only if a block, event, or action is selected.) Removes the selected items from the diagram.

Show Block Names Shows or hides the names that appear below the blocks in the diagram.

Zoom In Increases the Zoom Scale value by a factor of 0.9.

Zoom Out Decreases the Zoom Scale by a factor of 0.9.

Tip: On Windows, you can press Ctrl and use the mouse scroll wheel to zoom in to and out from the diagram.

Zoom Scale Enables the user to set the scale of the zoom on a scale of 0 to 10000. The original Zoom Scale value is 1. The Zoom Scale value increases as you zoom into the diagram.

Preview Shows or hides the Preview window in the lower right corner of the System Diagram.

Show Event Action Link Across Blocks Shows or hides the green arrows that represent action links from one block to another. If this option is turned off, action links for selected events and actions still appear.

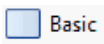
Align Selected Vertices Vertically (Available only if more than one block is selected.) Updates the horizontal position of the selected blocks so that they are aligned vertically.

Align Selected Vertices Horizontally (Available only if more than one block is selected.) Updates the vertical position of the selected blocks so that they are aligned horizontally.

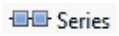
When you right-click an item in the System Diagram, additional submenu items appear in the pop-up menu. The submenu items that appear depend on the type of item. These options enable you to add, remove, or change the events and actions attached to a block item.

JMP PRO Shapes Toolbar

The Shapes toolbar contains the block shapes used to represent components in the System Diagram. Add block shapes to the System Diagram by clicking the icon for the block shape and dragging it onto the System Diagram. The Shapes toolbar includes the following block shape icons:

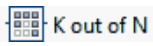


Adds a single block shape that represents a single component.

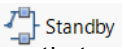


Adds a series block shape that represents a group of identical components that are connected in series. All of the components must be functional for the block to remain functional.

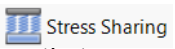
Adds a parallel block shape that represents a group of identical components that are connected in parallel. At least one of the components must be functional for the block to remain functional.



Adds a *k*-out-of-*n* block shape that represents a group of *n* identical components that are connected in parallel. At least *k* of the *n* components must be functional for the block to remain functional.



Adds a standby block shape that represents a group of *n* identical components that are connected in parallel. Only *k* components are *active*, or performing work in the system. The remaining *inactive* components act as standby components that are activated when any of the *k* active components fail. The block must have at least *k* functional and activated components for the block to remain functional.



Adds a stress sharing block shape that represents *n* identical components that are connected in parallel. Components fail one at a time, and remaining components fail at a quicker rate. The block must have at least one functional component and the block must successfully reallocate stress across remaining components for the block to remain functional.

Adds a knot block shape that represents a combination of the block shapes that point to the knot. At least a specified number of the block shapes that point to the knot must be functional for the knot to remain functional.

For information about the settings available for block shapes, see "Configuration Panel".

JMP PRO Configuration Panel

The Simulation Settings report appears at the top of the Configuration panel. The component settings for every selected component appear below the Simulation Settings outline. Configuration settings for selected events and actions appear below the block shape to which the action or event belongs.

JMP PRO Simulation Settings

Alter the settings for the simulation in the Simulation Settings report, which is available in the Configuration panel. The following settings are available:

Duration The length of time that is simulated in each iteration.

Time Unit The unit of time to use for the simulation.

Note: Recurring events and non-immediate actions use the Time Unit specified in Simulation Settings.

N Simulations The number of iterations in the simulation.

Seed (Optional) A random seed that ensures the reproducibility of simulation results. By default, the Seed is set to zero, which does not produce reproducible results. When you save the analysis to a script, the random seed that you enter is saved to the script.

Caution: If you run a simulation by right-clicking and selecting **Run Multithreaded Simulation**, the results are not reproducible, even if you specify a random seed.

JMP PRO Block Settings

Select a block shape in the System Diagram to see its settings in the Configuration panel.

Each block shape, except the knot block, has a Turn On System Exemption option. If this option is selected for a block, the block is not turned on when the Turn On System action is invoked.

Each block shape, except the knot block, has a failure distribution that determines the rate at which the block shape's individual components randomly fail. The failure distribution for a basic block determines the rate at which the block fails, because basic blocks represent only one component. For more information about the available failure distribution options, see "Distribution Options". Under the Distribution option, you can specify the Time Unit and distribution-dependent parameters for the block.

Series and Parallel

A series block fails when one of its components fails. A parallel block fails when all of its components fail. The following option is available for series and parallel blocks:

N Specifies the number of identical components contained in the block.

K-out-of-N

K-out-of-*N* blocks contain *n* identical components. The block fails when fewer than *k* of the components are functional. The following options are available for *K*-out-of-*N* blocks:

K Specifies the minimum number of functional components required for the block to remain functional.

N Specifies the number of identical components contained in the block.

Standby

Standby blocks have secondary components, called standby components, that are inactive. Active components perform work within a standby block. Inactive components do not perform work within a standby block, and are activated one at a time as active components fail. Occasionally, the activation process is not successful. A component switch might fail when activating a standby component. A standby block fails when less than k of its n identical components are active. The following options are available for standby blocks:

K Specifies the number of components that are initially active. This is also the minimum number of active components that is required for the block to remain functional.

N Specifies the total number of identical components within the block. The difference between k and n is equal to the number of standby components.

Switch Type Specifies the mechanism that activates a single standby component if any active component fails.

Single Switch A single switch exists in the block. If the activation of a standby component fails, then the block also fails.

Individual Switches A switch exists for each standby component. If the activation of a standby component fails, that standby component cannot be activated. The standby block attempts to activate the next standby component until a standby component is activated. If no remaining switches are functional and fewer than k of the components are active, then the block fails.

Switch Reliability Specifies the probability of success of activating a standby component when any active component fails.

Standby Type Specifies the state and failure distribution of the standby components.

Cold Standby components do not age until they are activated.

Warm Standby components age according to a secondary failure distribution while they are inactive. When standby components are activated, they age according to the primary failure distribution. Use the secondary failure distribution to mimic reduced stress on standby components that are not performing work in the standby block.

Stress Sharing

A stress sharing block distributes stress equally among its components. As components fail, the components that remain functional experience increased stress and subsequently fail at an increased rate.

N Specifies the total number of identical components contained in the block.

Switch Reliability Specifies the probability of successfully reallocating stress among the remaining functional components. The block fails if the reallocation of stress fails.

Stress Sharing Type Specifies how stress is shared among functional components.

Basic (default) Specifies that stress is shared equally among the remaining functional components. This type of stress sharing is referred to as *Load Sharing*. The characteristic life of individual components is proportional to the number of components that share the work load.

Custom Specifies that components share stress according to the JSL code that appears in the Sharing Formula option. The Sharing Formula defines how stress changes when components fail.

Knot

Knot blocks do not have a failure distribution. Knot blocks fail only if the number of connected blocks shapes that are functional falls below the specified minimum number. The following option is available for knot blocks:

Minimum Available Specifies the minimum number of functional blocks that must point to the knot block for the knot block to remain functional.

JMP PRO Add Event Panel

Events represent discrete occurrences in a simulation. You can use events to trigger actions. There is no limit to how many actions a single event can trigger. The following options are available in the Add Event Panel:

Note: Some block shapes do not have all of the available event types.

Block Failure Occurs when the block fails. If the component pathway is interrupted, the system is set to the Down state.

Scheduled Occurs at a specified recurring interval. The Max Occurrence option sets a limit on the number of occurrences of this event in a simulation. By default, Max Occurrence is set to missing. In this case, the event continues to occur until the end of the simulation. See "Event Settings".

Inservice Based Occurs when the component reaches a specified age. A component's age is the cumulative time that it has been functional and active. Age stops increasing when a component fails or its block is turned off. The Max Occurrence sets a limit on how many times this event can take place in a simulation. By default, the event continues to occur until the end of the simulation. See "Event Settings".

System is Down Occurs when the system is set to either the Down or the Off state. The system state can be changed intentionally or unintentionally.

Initialization Occurs when each simulation iteration begins. Use this event to arrange actions that have to complete before the system runs.

Create an Event

1. To create an event, select a block shape in the System Diagram to display the Add Event and Add Action panels at the bottom of the System Diagram.
2. Select one of the options in the Add Event panel to define an event for the selected block shape.

Figure 12.6 Block Failure Event

An orange square that represents the new event appears above the component. Notice that a connection arrow appears on the right side of the selected event.

JMP PRO Add Action Panel

An action defines component and system behavior that is triggered by either a connected event or a connected action. Connected actions are triggered upon completion of the prior action. The following options are available in the Add Action Panel:

Note: Some block shapes do not have all of the available action types.

Table 12.1 Action Options

Action Name	Blocks that can use this Action	Starting behavior	Completion Behavior
Turn off System	All		All blocks are turned off. The system is set to the Off state if it was not already in the Down state.

Table 12.1 Action Options *(Continued)*

Action Name	Blocks that can use this Action	Starting behavior	Completion Behavior
Turn on System	All		All blocks that were not already failed are turned on. The system is set to the On state if there is an uninterrupted component pathway. Blocks with the Turn On System Exemption option selected are not turned on by this action.
Replace with New	All	Turns off the original block, and sets the system to the Down state if the component pathway is interrupted.	The block becomes new. Its age is reset and the block is turned on. The system is set to the On state if there is an uninterrupted component pathway.
Minimal Repair	Basic and Series	Equivalent to starting behavior for the Replace with New action.	Turns the block on.The system is set to the On state if there is an uninterrupted component pathway. The age of the block is not reset.
Turn On Block	All	Cancels the action if the block is in the Down state or is currently removed from the system.	Turns the block on increments Turn On Count by one.
Turn Off Block	All		Turns the block off. The system is set to the Down state if the component pathway is interrupted.
Remove Block	All		The block is removed from the system. The system is set to the Down state if the component pathway is interrupted.

Table 12.1 Action Options *(Continued)*

Action Name	Blocks that can use this Action	Starting behavior	Completion Behavior
Install New	All	Equivalent to starting behavior for Replace with New action.	Turns the block off. The age of the block is reset.
Install Used	Basic	Equivalent to starting behavior for Replace with New action.	Turns the block off. You specify a new age and failure distribution for the block.
Change Distribution	Basic	Turns the block off. The system is set to the Down state if the component pathway is interrupted.	Changes the failure distribution of the block. Use this to mimic operation changes over time or cumulative damage to the block.
Inspect Failure	All		Triggers connected actions if the block has failed or is removed from the system.
If	All		Triggers connected actions if the specified condition script is true.
Schedule	All		Triggers connected actions at a specified interval. You can limit the number of scheduled intervals or allow the action to continue through the end of the simulation iteration.

Create an Action

1. To create an action, select a block shape in the System Diagram to display the Add Event and Add Action panels at the bottom of the System Diagram.
2. Select one of the options from the Add Action panel to define an action for the selected block shape.

A blue action square is created above the selected block shape. Actions are triggered when a connected event occurs. Create an event that triggers the action that you defined.

3. Select one of the options from the Add Event panel to define an event for the selected block shape.

   An orange event square is created above the block shape. Notice the connection arrow on the right side of the event.

4. Click the connection arrow and drag it to the blue action square that you created in step 2.

   A green arrow connects the event and action squares. When the event occurs in a simulation iteration, it triggers the connected action.

5. Select the blue action square that you created in step 2.

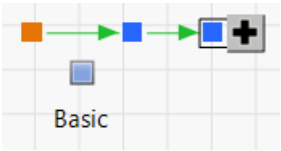
   You can connect additional actions that are triggered upon completion of the previous action by using the addition sign that appears on the right side of the selected action.

6. Click the addition sign and drag it to an empty area in the System Diagram.

   A list of the available actions appears.

7. Select one of the options from the action list to create an action that is connected to the first action.

Figure 12.7 Create an Action



A green arrow connects the two actions. The second action is triggered upon completion of the first action.

JMP PRO Repairable Systems Simulation Platform Options

The Repairable Systems Simulation red triangle menu contains the following options:

Save and Save As Enables you to save a Repairable Systems Simulation to a JMP Scripting Language (JSL) script that is automatically executed when it is opened in JMP. See the *Scripting Guide* for more information about Auto-Submit scripts.

Note: The Save and Save As red triangle options are equivalent to selecting **File > Save** and **File > Save As**, respectively. They are available in the red triangle menu for convenience.

Import Component Distribution Settings Enables you to import configuration settings for the system diagram from a data table. The table must contain columns for the component name, distribution, and one or more parameters. The number of parameters depends on the specified distribution.

Note: The strings in the imported table must be exact matches to the strings in the system diagram.

JMP PRO Options for Block Items

To view settings for the components in a selected design or subsystem, do the following:

- To view the Configuration settings for a block shape, select the block shape in the System Diagram.
- To view Configuration settings for more than one block shape, select multiple block shapes using the Arrow tool, press Ctrl, and click multiple block shapes.

JMP PRO Distribution Options

Each block shape randomly fails according to its specified failure distribution. In addition, you specify a time unit for the distribution. The block shape's time unit can be different from the time unit option that you specify in the simulation settings. The Turn On Count option is based on the number of times the individual block shape has been set to the On state.

The available failure distributions are listed in Table 12.2.

Table 12.2 Distributions and Additional Parameters

Property Type	Required Inputs
Exponential	Theta
Weibull	Alpha, Beta
Lognormal	location, scale
Loglogistic	location, scale
Fréchet	location, scale
GenGamma	mu, sigma, lambda
DS Weibull	Alpha, Beta, Defective Probability

Table 12.2 Distributions and Additional Parameters *(Continued)*

Property Type	Required Inputs
DS Lognormal	location, scale, Defective Probability
DS Loglogistic	location, scale, Defective Probability
DS Fréchet	location, scale, Defective Probability
Nonparametric	data or data file
Estimated	estimated distribution, data, or data file

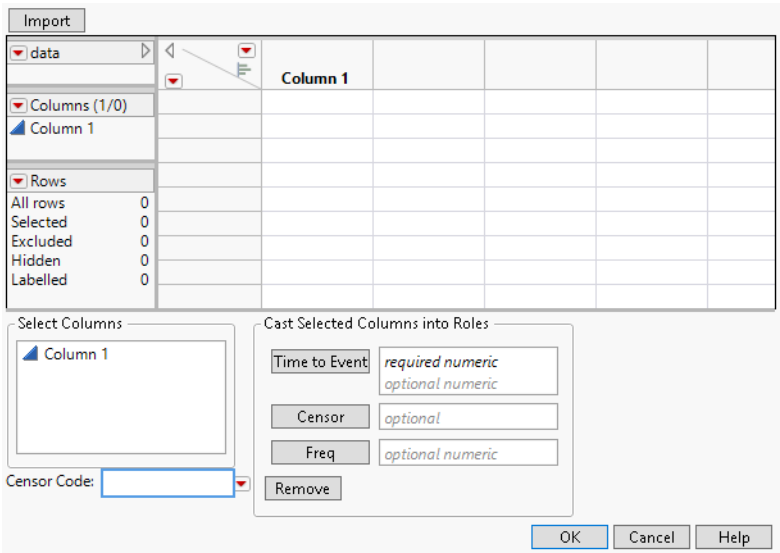
To see the formulas and parameterization for these failure distributions, see "Distributions".

JMP PRO Specify a Nonparametric or Estimated Distribution

The Nonparametric and Estimated options under Distribution enable you to approximate an arbitrary distribution. You can enter data manually or import a file that contains data. These data are used to approximate the distribution.

After selecting Nonparametric or Estimated, click the ... icon next to Data. The Provide Data window appears, which enables you to either enter data or import a data file. After you have imported or entered your data, the data are used to calculate a distribution for the component.

Figure 12.8 Provide Data Window



To import data from a file, do the following:

1. Before clicking the icon, open the JMP data table that contains the data to import.
2. Click the icon next to Data.
3. In the Provide Data window, click **Import**.

   The Select Data Table window appears.
4. From the Data Table list, select a data table.
5. Click **OK**.
6. In the panel beneath the data grid, specify the columns that represent Time to Event data.
7. Specify Censor and Freq columns, if appropriate.
8. In the Provide Data window, click **OK**.

To enter data manually, do the following:

1. Create columns for Time to Event data.
2. Create columns for Censor and Freq data, if appropriate.
3. Enter the data in the columns.
4. In the panel beneath the data grid, specify which columns represent Time to Event data, as well as Censor and Freq data if appropriate. See step 5 and step 6 above.
5. In the Provide Data window, click **OK**.

JMP PRO Event Settings

Select an event in the System Diagram to see its settings in the Configuration panel. You can change the Event Name setting to distinguish the event from others. Only the Scheduled and Inservice events have additional settings, which are listed here:

Scheduled

Scheduled events have the following options:

Recurring Interval Specifies the length of the time interval between event occurrences.

Max Occurrences Specifies the maximum number of times this event can occur.

Inservice

Inservice events have the following options:

Inservice Specifies the interval of component run time to wait before this event occurs.

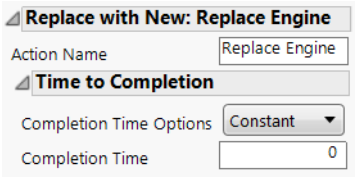
Note: Component run time accumulates only when the system is in the On state and the component is functional.

Max Occurrences Specifies the maximum number of times this event can occur.

JMP PRO Action Settings

Select an action in the System Diagram to see its settings in the Configuration panel. You can change the Action Name setting to distinguish the action from others. By default, actions are completed immediately. Non-immediate completion time options are available in the Completion Time Options menu.

Figure 12.9 Action Settings



The following options appear in the Completion Time Options list:

Immediate (default) Specifies that no time passes between starting and completion behavior.

Constant Specifies that the time lapse is always the specified Completion Time.

Choice Specifies that the time lapse is randomly chosen from the specified list of comma-separated values.

Uniform Specifies that the time lapse is randomly chosen from a uniform distribution with the specified Minimum and Maximum.

Triangle Specifies that the time lapse is randomly chosen from a triangular distribution with the specified Minimum, Mode, and Maximum.

Normal Specifies that the time lapse is randomly chosen from a normal distribution with the specified Mean and Standard Deviation.

JMP PRO Repairable Systems Simulation Results

- "Results Table"
- "Results Explorer"
- "Repairable Systems Simulation Results Options"
- "RSS Results Explorer Point Estimation Profilers"

JMP PRO Results Table

When you click the green arrow under the Start block to run the simulation, the results of the simulation appear in a data table. This table describes the events and subsequent actions that occurred in each simulation iteration.

The results table contains the following columns:

Sim ID Identifies the simulation iteration to which the event or action belongs.

Time Gives the exact time that the event or action took place in the simulation.

Subject Gives the name of the block shape to which the event or action is connected or System in the case of system events and actions.

Predicate Gives the name of the event or action that took place.

State Gives the state of the Subject at that exact Time. The initialization and termination of each action are denoted with Start and Finish, respectively.

Note Gives an additional description of an action. The end of each simulation iteration is denoted with End.

Figure 12.10 RSS Results Table

	Sim ID	Time	Subject	Predicate	State	Note
1	1	0	Spare	Initialization Spar...	Start	
2	1	0	Spare	Initialization Spar...	Off	
3	1	0	Spare	Initialization Spar...	Finish	
4	1	65.024584796	Tire 2	Tire 2 is unrepair...	Down	
5	1	65.024584796	System	Turn Down System	Down	Unintended
6	1	65.024584796	Engine	Turn Off Block by...	Off	
7	1	65.024584796	Transmission	Turn Off Block by...	Off	
8	1	65.024584796	Battery	Turn Off Block by...	Off	
9	1	65.024584796	Tire 1	Turn Off Block by...	Off	
10	1	65.024584796	Tire 3	Turn Off Block by...	Off	
11	1	65.024584796	Tire 4	Turn Off Block by...	Off	
12	1	65.024584796	Tire 2	Replace Tire 2	Removed	
13	1	65.024584796	Tire 2	Replace Tire 2	Start	

Notice that the first entry in Figure 12.10 is the start of the Initialization Spare in Trunk action, which has an immediate completion time. The action turns the Spare component off and then finishes while Time is still 0.

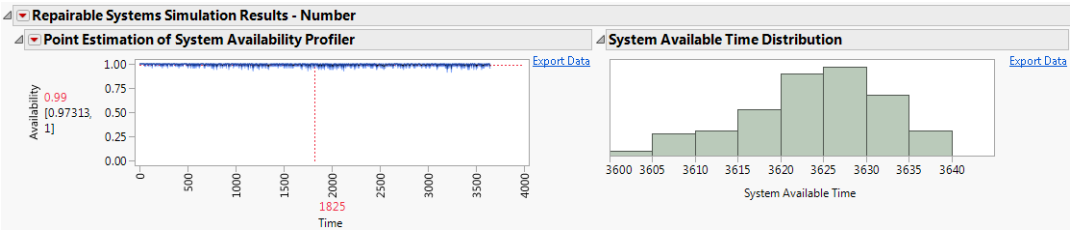
The next event that occurs is Tire 2 is Unrepairable when approximately 65 days have passed in the first simulation iteration. In row five, the system is unintentionally set to the Down state because only three tires are functional. The Need 4 Tires to Drive knot requires at least four tires, including the Spare, to be functional. The Tire 2 is Unrepairable event simultaneously triggers the Replace Tire 2 action and the Use Spare action. The Drive with Spare action is triggered in row eleven, and the system is set to the On state.

Notice that Tire 2 failed and was replaced by the Spare at the same time. No time elapsed between the time that the system was set to the Down state and the time that the system was returned to the On state. Because the subsequent actions had an immediate completion time, the Tire 2 is Unrepairable event did not cause any system outage time. The results explorer shows the system outage time that accumulates when actions are non-immediate.

JMP PRO Results Explorer

When you click the green arrow next to the Launch Repairable Systems Simulation Results Explorer script in the results data table, a report appears that contains an analysis of the simulation results.

Figure 12.11 Partial RSS Explorer Report



By default, the report contains two types of graphs. The first type of graph, shown on the left side of Figure 12.11, is a point estimate graph. The point estimate graph displays aggregated simulation results on the vertical axis plotted against simulation iteration time on the horizontal axis. The range of the horizontal axis is the duration that you specify in the simulation settings. The estimated probability of the system being available at a given time is shown in red next to the vertical axis. Below the point estimate is a 95% confidence interval for the point estimate.

Tip: To see the point estimation of system availability at a specific time in the simulation iteration, click the red number below the horizontal axis. Specify a time within the range of the simulation duration and press Enter.

The second type of graph, shown on the right side of Figure 12.11, is a histogram of the system available time from the simulation iterations. The bins in the histograms are representative of one of the following:

- Total simulation time.
- Proportion of the simulation duration.

Tip: Select the **Export Data** option beside a graph to create a data table with the data that the graph uses.

By default, the report contains the following graphs:

Point Estimation of System Availability Profiler Shows a profiler graph over time of the estimated probability that the system is in the On or the Off state during a simulation iteration.

System Available Time Distribution Shows a histogram of the total time that the system was available for each simulation iteration.

System Availability Distribution Shows a histogram of the total time that the system was available for each simulation iteration divided by the duration of the simulation. This distribution is the same as the System Available Time Distribution, except that it is shown as a proportion of the duration of the simulation.

Point Estimation of System In Service Probability Profiler Shows a profiler graph over time of the estimated probability that the system is in the On state during a simulation iteration.

System In Service Time Distribution Shows a histogram of the total time that the system was in the On state for each simulation iteration.

System In Service Probability Distribution Shows a histogram of the total time that the system was in the On state for each simulation iteration divided by the duration of the simulation. This distribution is the same as System In Service Time Distribution, except that the distribution is shown as a proportion of the duration of the simulation.

Point Estimation of System Unplanned Outage Profiler Shows a profiler graph over time of the estimated probability that the system is in the Down state during a simulation iteration.

System Unplanned Outage Distribution Shows a histogram of the total time that the system was in the Down state for each simulation iteration.

System Unplanned Outage Percentage Distribution Shows a histogram of the total time that the system was in the Down state for each simulation iteration divided by the duration of the simulation. This distribution is the same as System Unplanned Outage Distribution, except that the distribution is shown as a proportion of the duration of the simulation.

JMP PRO Repairable Systems Simulation Results Options

The Repairable Systems Simulation Results red triangle menu contains the following options:

Point Estimation of Component Availability For each block shape in the system diagram, this option shows the following reports:

- Point Estimation of Availability Profiler
- Available Time Distribution
- Availability Distribution
- Point Estimation of Unplanned Outage Profiler
- Unplanned Outage Time Distribution
- Unplanned Outage Distribution

System Availability by Period Shows a graph that enables you to examine the average system availability during specified time periods. Use the By Time panel to define the time period as an interval of time. Use the By Event panel to define the time period as a specified number of occurrences of one event. When you click **Go**, the report shows a bar chart of the average system availability during each period.

Box Plot of System Total Downtime by Component Shows a box plot of the total time the system was in the Down or Off states caused by individual components. The components are ordered by contribution from the most outage time to the least outage time, similar to a Pareto chart.

Box Plot of System Unplanned Total Outage Time by Component Shows a box plot of total time the system was in the Down state caused by individual components. The components are ordered by contribution from the most outage time to the least outage time, similar to a Pareto chart.

See *Using JMP* for more information about the following options:

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

JMP PRO RSS Results Explorer Point Estimation Profilers

Each point estimation profiler within the Repairable Systems Simulation Results Explorer report has a red triangle menu that contains the following options:

Confidence Intervals Shows or hides the 95% confidence intervals on the point estimation graphs.

Reset Factor Grid Opens the Factor Settings window, which enables you to modify the parameters of the point estimation graph. See *Profilers* for more information about setting the factor grid.

Factor Settings Provides additional options affecting the Factor Grid. See *Profilers* for more information about Factor Settings.

Chapter 13

Survival Analysis

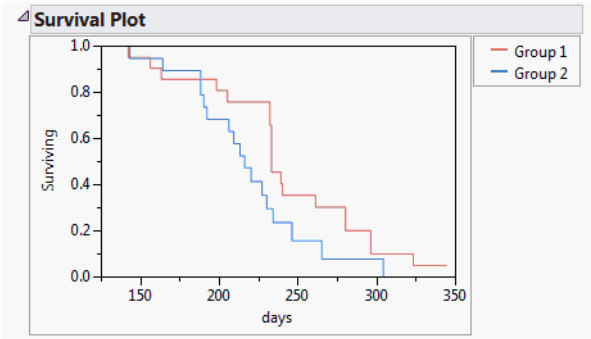
Analyze Survival Time Data

Survival data contain duration times until the occurrence of a specific event and are sometimes referred to as *event-time* response data. The event is usually failure, such as the failure of an engine or death of a patient. If the event does not occur before the end of a study for an observation, the observation is said to be *censored*.

The Survival platform fits a single Y that represents time to event (or time to failure). Use the Survival platform to examine the distribution of the failure times.

Tip: To fit explanatory models, use the Fit Parametric Survival platform or the Fit Proportional Hazards platform.

Figure 13.1 Example of a Survival Plot



Contents

Overview of the Survival Analysis Platform . . . 345
Example of Survival Analysis . . . 346
Launch the Survival Platform . . . 348
The Survival Plot . . . 349
Survival Platform Options . . . 350
Exponential, Weibull, and Lognormal Plots and Fits . . . 352
Fitted Distribution Plots . . . 356
Competing Causes . . . 357
Additional Examples of the Survival Platform . . . 358
Example of Survival Analysis . . . 358
Example of Competing Causes . . . 361
Example of Interval Censoring . . . 363
Statistical Reports for Survival Analysis . . . 365

Overview of the Survival Analysis Platform

Survival data need to be analyzed with specialized methods for two reasons:

1. The survival times usually have specialized non-normal distributions, like the exponential, Weibull, and lognormal.
2. Some of the data could be censored.

Survival functions are calculated using the nonparametric Kaplan-Meier method for one or more groups of either *complete* or *right-censored* data. Complete data have no censored values. Right-censoring is when you do not know the exact survival time, but you know that it is greater than the specified value. Right-censoring occurs when the study ends without all the units failing, or when a patient has to leave the study before it is finished. The censored observations cannot be ignored without biasing the analysis. The elements of a survival model are:

- A time indicating how long until the unit (or patient) either experienced the event or was censored. Time is the model response (*Y*).
- A censoring indicator that denotes whether an observation experienced the event or was censored. JMP uses the convention that the code for a censored unit is 1 and the code for a non-censored event is zero.
- Explanatory variables (if a regression model is used.)
- Interval censoring is when a data point is somewhere on an interval between two values. If interval censoring is needed, then two Y variables hold the lower and upper limits bounding the event time.

Common terms used for reliability and survival data include lifetime, life, survival, failure-time, time-to-event, and duration.

The Survival platform computes product-limit (Kaplan-Meier) survival estimates for one or more groups. It can be used as a complete analysis or is useful as an exploratory analysis to gain information for more complex model fitting. The Kaplan-Meier Survival platform does the following:

- Shows a plot of the estimated survival function for each group. A plot for the whole sample is optional.
- Calculates and lists survival function estimates for each group and for the combined sample.
- Shows exponential, Weibull, and lognormal diagnostic failure plots to graphically check the appropriateness of using these distributions for further regression modeling. Parameter estimates are available on request.
- Computes the Log Rank and generalized Wilcoxon Chi-square statistics to test homogeneity of the estimated survival function across groups.

- Analyzes competing causes, prompting for a cause of failure variable, and estimating a Weibull failure time distribution for censoring patterns corresponding to each cause.

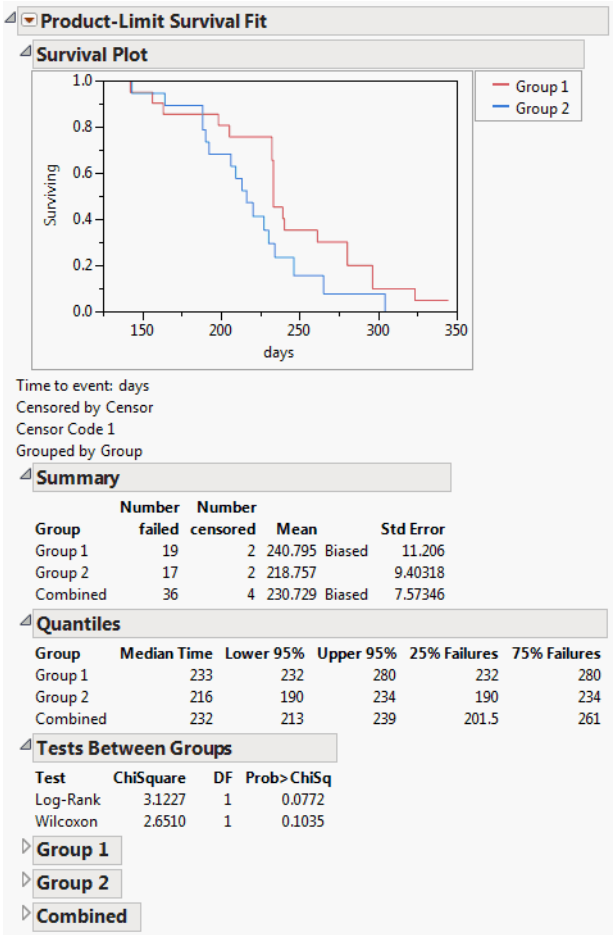
Example of Survival Analysis

An experiment was undertaken to characterize the survival time of rats exposed to a carcinogen in two treatment groups. The data are in the Rats.jmp sample data table. The event in this example is death. The objective is to see whether rats in one treatment group live longer (more days) than rats in the other treatment group.

1. Select **Help > Sample Data Library** and open Rats.jmp.

   The data in the days column is the survival time. Notice that some observations are censored.
2. Select **Analyze > Reliability and Survival > Survival**.
3. Select days and click **Y, Time to Event**.
4. Select Group and click **Grouping**.
5. Select Censor and click **Censor**.
6. Click **OK**.

Figure 13.2 Survival Plot for Rats.jmp Data

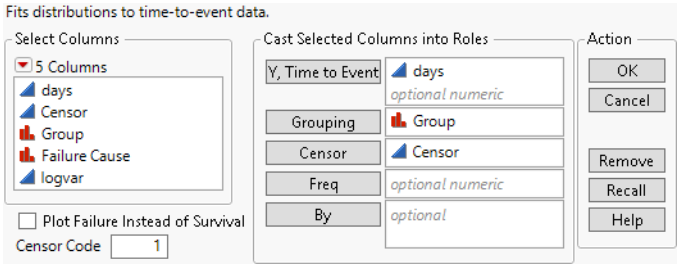


It appears that the rats in treatment group 1 are living longer than the rats in treatment group 2.

Launch the Survival Platform

Launch the Survival platform by selecting **Analyze > Reliability and Survival > Survival**.

Figure 13.3 The Survival Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Survival launch window contains the following options:

Y, Time to Event Identifies the time to event or time to censoring. If you have interval censoring, specify two Y variables, representing the lower and upper limits.

Grouping Classifies the data into groups that are fit separately.

Censor Identifies censored values. Enter the value that identifies censoring in the Censor Code box. This column can contain more than two distinct values under the following conditions:

- All censored rows contain the value that is entered in the Censor Code box.
- Non-censored rows have a value other than what is in the Censor Code box.

Freq Indicates the column whose values are the frequencies of observations for each row when there are multiple units recorded. If the value is 0 or a positive integer, then the value represents the frequencies or counts of observations for each row.

By Performs a separate analysis for each level of a classification or grouping variable.

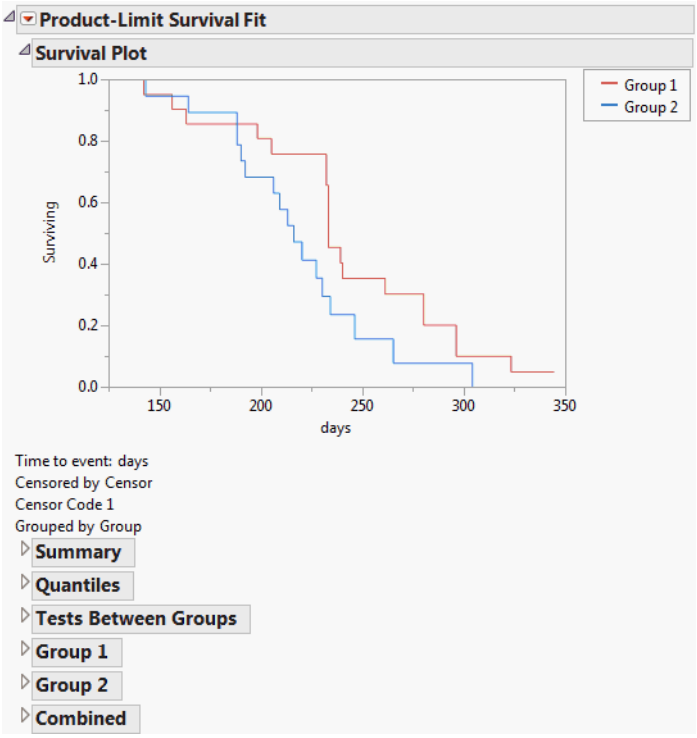
Plot Failure Instead of Survival Shows a failure probability plot instead of its reverse (a survival probability plot).

Censor Code Identifies the value in the Censor column that designates right-censored observations. The default value is 1.

The Survival Plot

The Survival platform shows overlay step plots of estimated survival functions for each group. A legend identifies groups by color and line type.

Figure 13.4 The Survival Plot



Reports beneath the plot show summary statistics and quantiles for survival times. Estimated survival times for each observation are computed within groups. Survival times are computed from the combined sample. When there is more than one group, statistical tests compare the survival curves.

If there are any failures that occur at time zero, the Failures at Time Zero report appears when you request a distribution fit from the Product-Limit Survival Fit red triangle menu. This report contains a table of counts of zero-time failures for each level of the Grouping variable. If there is no Grouping variable specified, there is one row in the table labeled Combined. The table also contains a column labeled Prob Time>0. This column is the proportion of observations in each group that has a nonzero failure time.

Survival Platform Options

The Product-Limit Survival Fit red triangle menu contains the following options:

Survival Plot Shows the overlaid survival plots for each group.

Failure Plot Shows the overlaid failure plots (proportion failing over time) for each group (in the tradition of the reliability literature.) A failure plot reverses the vertical axis to show the number of failures rather than the number of survivors.

Note: The Failure Plot option replaces the Reverse Y Axis option found in older versions of JMP (which is still available in scripts).

Plot Options Contains the following options:

Note: The first seven options (Show Points, Show Kaplan Meier, Show Combined, Show Confid Interval, Show Simultaneous CI, Show Shaded Pointwise CI, and Show Shaded Simultaneous CI) and the last two options (Fitted Survival CI, Fitted Failure CI) pertain to the initial survival plot and failure plot. The other five (Midstep Quantile Points, Connect Quantile Points, Fitted Quantile, Fitted Quantile CI Lines, Fitted Quantile CI Shaded) pertain only to the distributional plots.

Show Points Shows the sample points at each step of the survival plot. Failures appear at the bottom of the steps, and censorings are indicated by points above the steps.

Show Kaplan Meier Shows the Kaplan-Meier curves. This option is on by default.

Show Combined Shows the survival curve for the combined groups in the Survival Plot.

Show Confid Interval Shows the pointwise 95% confidence bands on the survival plot for groups and for the combined plot when it appears with the Show Combined option.

Show Points, Show Combined When you select the Show Points and Show Combined options, the survival plot for the total or combined sample appears as a gray line. The points also appear at the plot steps of each group.

Show Simultaneous CI Shows the simultaneous confidence bands for all groups on the plot. Meeker and Escobar (1998, ch. 3) discuss pointwise and simultaneous confidence intervals and the motivation for simultaneous confidence intervals in survival analysis.

Midstep Quantile Points Changes the plotting positions to use the *modified Kaplan-Meier* plotting positions. These plotting positions are equivalent to taking mid-step positions of the Kaplan-Meier curve, rather than the bottom-of-step positions. This option is recommended, so it is on by default.

Connect Quantile Points Shows the lines in the plot. This option is on by default.

Fitted Quantile Shows the straight-line fit on the fitted Weibull, lognormal, or exponential plot. This option is on by default.

Fitted Quantile CI Lines Shows the 95% confidence bands for the fitted Weibull, lognormal, or exponential plot.

Fitted Quantile CI Shaded Shows the display of the 95% confidence bands for a fit as a shaded area or dashed lines.

Fitted Survival CI Shows the confidence intervals (on the survival plot) of the fitted distribution.

Fitted Failure CI Shows the confidence intervals (on the failure plot) of the fitted distribution.

Exponential Plot Plots the cumulative exponential failure probability by time for each group. Lines that are approximately linear empirically indicate the appropriateness of using an exponential model for further analysis. For example, in Figure 13.5, the lines for Group 1 and Group 2 in the Exponential Plot are curved rather than straight. This indicates that the exponential distribution is not appropriate for this data. See "Exponential, Weibull, and Lognormal Plots and Fits".

Exponential Fit Produces the Exponential Parameters table and the linear fit to the exponential cumulative distribution function in the Exponential Plot (Figure 13.5). The parameter Theta corresponds to the mean failure time. See "Exponential, Weibull, and Lognormal Plots and Fits".

Weibull Plot Plots the cumulative Weibull failure probability by log(time) for each group. A Weibull plot that has approximately parallel and straight lines indicates a Weibull survival distribution model might be appropriate to use for further analysis. See "Exponential, Weibull, and Lognormal Plots and Fits".

Weibull Fit Produces the linear fit to the Weibull cumulative distribution function in the Weibull plot and two popular forms of Weibull estimates. These estimates are shown in the Extreme-value Parameter Estimates table and the Weibull Parameter Estimates tables (Figure 13.5). The Alpha parameter is the 0.632 quantile of the failure-time distribution. The Extreme-value table shows a different parameterization of the same fit, where Lambda = ln(Alpha) and Delta = 1/Beta. See "Exponential, Weibull, and Lognormal Plots and Fits".

LogNormal Plot Plots the cumulative lognormal failure probability by log(time) for each group. A lognormal plot that has approximately parallel and straight lines indicates a lognormal distribution is appropriate to use for further analysis. See "Exponential, Weibull, and Lognormal Plots and Fits".

LogNormal Fit Produces the linear fit to the lognormal cumulative distribution function in the lognormal plot and the LogNormal Parameter Estimates table shown in Figure 13.5. Mu and Sigma correspond to the mean and standard deviation of a normally distributed natural logarithm of the time variable. See "Exponential, Weibull, and Lognormal Plots and Fits".

Fitted Distribution Plots Use in conjunction with the fit options to show three plots corresponding to the fitted distributions: Survival, Density, and Hazard. If you have not performed a fit, no plot appears. See "Fitted Distribution Plots".

Competing Causes Performs an estimation of the Weibull model using the specified causes to indicate a failure event and other causes to indicate censoring. The fitted distribution appears as a dashed line in the Survival Plot. See "Competing Causes".

Estimate Survival Probability Estimates survival probabilities for the time values that you specify.

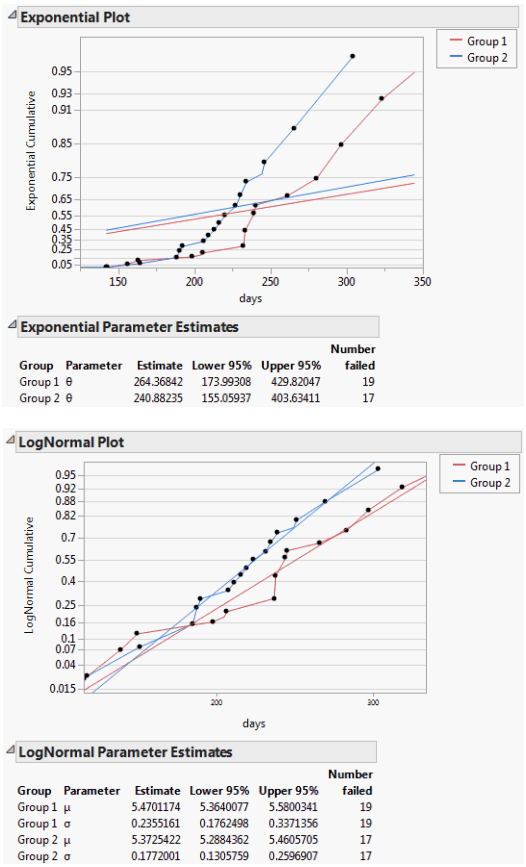
Estimate Time Quantile Estimates a time quantile for each survival probability that you specify.

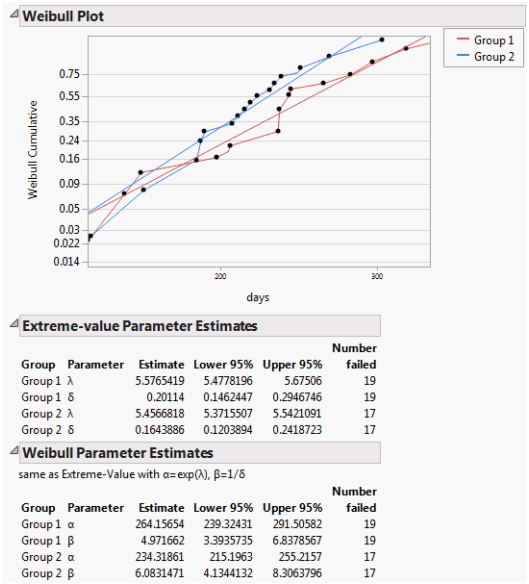
Save Estimates Creates a data table containing survival and failure estimates, along with confidence intervals, and other distribution statistics.

Exponential, Weibull, and Lognormal Plots and Fits

For each of the three supported distributions in the Survival platform, there is a plot command and a fit command. Use the plot command to see whether the event markers seem to follow a straight line. The markers tend to follow a straight line when the distributional fit is suitable for the data. Then, use the fit commands to estimate the parameters.

Figure 13.5 Exponential, Weibull, and Lognormal Plots and Reports





The following table shows what to plot to make a straight line fit for that distribution:

Table 13.1 Straight Line Fits for Distribution

Distribution Plot	Horizontal Axis	Vertical Axis	Interpretation
Exponential	time	-log(S)	slope is 1/theta
Weibull	log(time)	log(-log(S))	slope is beta
Lognormal	log(time)	Probit(1-S)	slope is 1/sigma

Note: S = product-limit estimate of the survival distribution.

Exponential

The exponential distribution is the simplest distribution for modeling time-to-event data. The exponential distribution has only one parameter, theta. It is a constant-hazard distribution, with no memory of how long it has survived to affect how likely an event is. The parameter theta is the expected lifetime.

Weibull

The Weibull distribution is the most popular distribution for modeling time-to-event data. The Weibull distribution can have two or three parameters. The Survival platform fits the two-parameter Weibull distribution. Authors parameterize this distribution in many different ways (Table 13.2). JMP reports two of these parameterizations: the Weibull alpha-beta parameterization and a parameterization based on the smallest extreme value distribution.

The alpha-beta parameterization, shown in the Weibull Parameter Estimates report, is widely used in the reliability literature (Nelson 1990). The *alpha* parameter is interpreted as the quantile at which 63.2% of the units fail. The *beta* parameter determines how the hazard rate changes over time. If *beta* > 1, the hazard rate increases over time; if *beta* < 1, the hazard rate decreases over time; and if *beta* = 1, the hazard rate is constant over time. A Weibull distribution with a constant hazard function is equivalent to an exponential distribution.

The lambda-delta extreme value parameterization is shown in the Extreme-Value Parameter Estimates report. This parameterization is sometimes desirable in a statistical sense because it places the Weibull distribution in a location-scale setting (Meeker and Escobar 1998, p. 86). The location parameter is *lambda*, and the scale parameter is *delta*. In relation to the alpha-beta parameterization, *lambda* is equal to the natural log of alpha, and *delta* is equal to the reciprocal of beta. Therefore, the delta parameter determines how the hazard rate changes over time. If *delta* > 1, the hazard rate decreases over time; if *delta* < 1, the hazard rate increases over time; and if *delta* = 1, the hazard rate is constant over time. A Weibull distribution with a constant hazard function is equivalent to an exponential distribution.

Table 13.2 Various Weibull Parameters in Terms of alpha and beta in JMP

JMP Weibull	*alpha*	*beta*
Wayne Nelson	alpha=*alpha*	beta=*beta*
Meeker and Escobar	eta=*alpha*	beta=*beta*
Tobias and Trindade	c = *alpha*	m = *beta*
Kececioglu	eta=*alpha*	beta=*beta*
Hosmer and Lemeshow	exp(X beta)=*alpha*	lambda=*beta*
Blishke and Murthy	beta=*alpha*	alpha=*beta*

Table 13.2 Various Weibull Parameters in Terms of alpha and beta in JMP *(Continued)*

Kalbfleisch and Prentice	lambda = 1/*alpha*	p = *beta*
JMP Extreme Value	lambda=log(*alpha*)	delta=1/*beta*
Meeker and Escobar s.e.v.	mu=log(*alpha*)	sigma=1/*beta*

Lognormal

The lognormal distribution is also very popular for modeling time-to-event data. The lognormal distribution is equivalent to the distribution where if you take the log of the values, the distribution is normal. If you want to fit a normal distribution to your data, you can take the exp() of it and model your data with a lognormal distribution. See "Additional Examples of Fitting Parametric Survival".

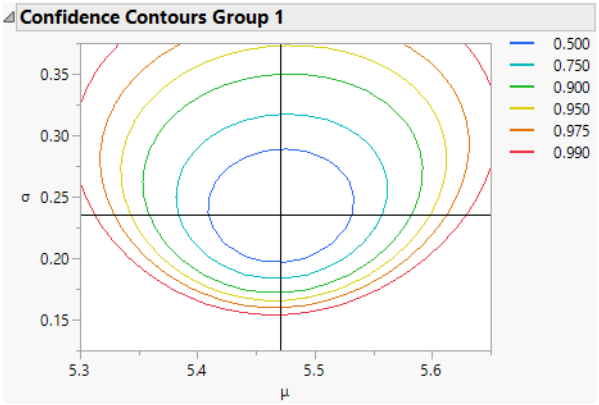
Additional Options

To see additional options for the exponential, Weibull, and lognormal fits, press Shift, click the red triangle next to Product-Limit Survival Fit, and select the desired fit.

Use these options to do the following tasks:

- Set the confidence level for the limits.
- Set the constrained value for theta (in the case of an exponential fit), sigma (in the case of a lognormal fit) or beta (in the case of a Weibull fit). See "WeiBayes Analysis".
- Obtain a Confidence Contour Plot for the Weibull and lognormal fits (when there are no constrained values).

Figure 13.6 Confidence Contour Plot



WeiBayes Analysis

JMP can constrain the values of the Theta (Exponential), Beta (Weibull), and Sigma (LogNormal) parameters when fitting these distributions. This feature is needed in *WeiBayes* situations, for example:

- Where there are few or no failures
- There are existing historical values for beta
- There is still a need to estimate alpha

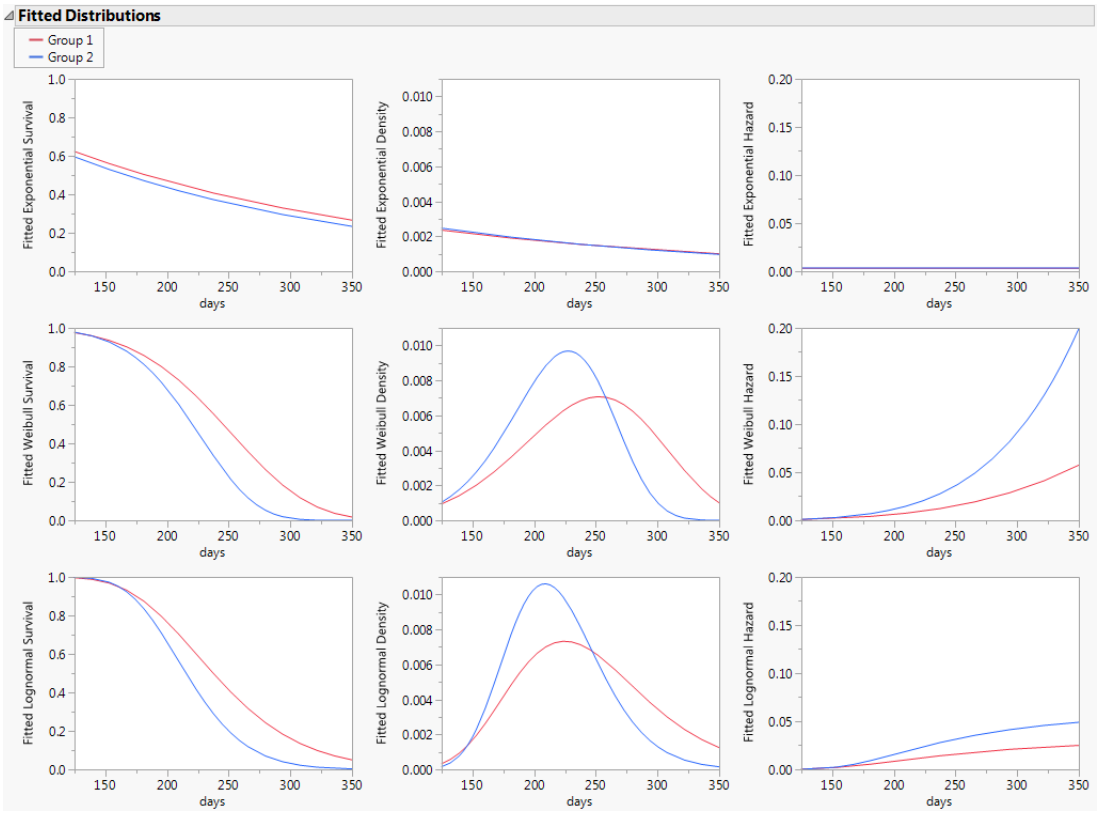
For more information about WeiBayes situations, see Abernethy (1996).

With no failures, the standard technique is to add a failure at the end. Then, the estimates reflect a type of lower bound on the alpha value, rather than a real estimate. However, the WeiBayes feature allows for a true estimation.

Fitted Distribution Plots

Use the Fitted Distribution Plots option to see Survival, Density, and Hazard plots for the exponential, Weibull, and lognormal distributions. The plots share the same axis scaling so that the distributions can be easily compared.

Figure 13.7 Fitted Distribution Plots for Three Distributions



These plots can be transferred to other graphs through the use of graphic scripts. To copy the graph, right-click in the plot to be copied and select **Edit > Copy Frame Contents**. Right-click in the destination plot and select **Edit > Paste Frame Contents**.

Competing Causes

Sometimes there are multiple causes of failure in a system. For example, suppose that a manufacturing process has several stages and the failure of any stage causes a failure of the whole system. If the different causes are independent, the failure times can be modeled by an estimation of the survival distribution for each cause. A censored estimation is undertaken for a given cause by treating all the event times that are not from that cause as censored observations.

The Competing Causes red triangle menu contains the following options:

Omit Causes Removes the specified cause value and recalculates the survival estimates.

Save Cause Coordinates Adds a new column to the current table called log(–log(Surv)). This information is often used to plot against the time variable with a grouping variable, such as the code for type of failure.

Weibull Lines Adds Weibull lines to the plot.

Hazard Plot Adds a Hazard Plot.

Simulate Creates a new data table containing time and cause information from the Weibull distribution, as estimated by the data.

Additional Examples of the Survival Platform

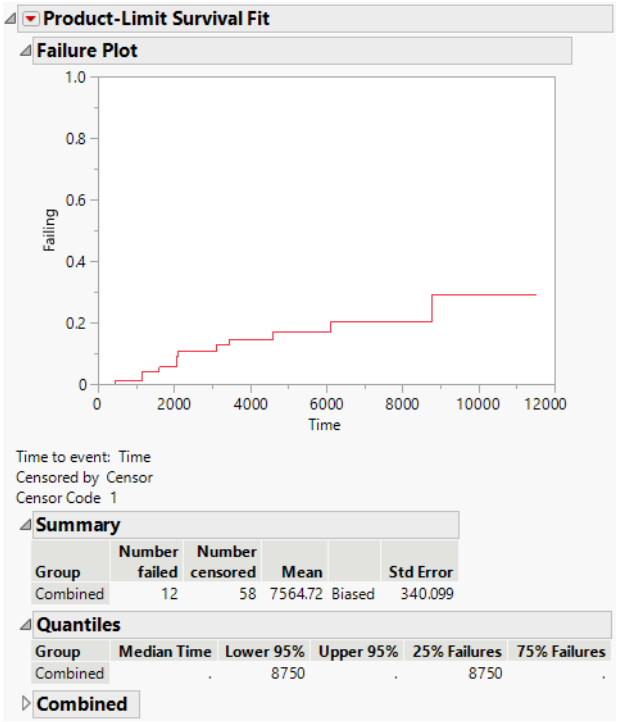
- "Example of Survival Analysis"
- "Example of Competing Causes"
- "Example of Interval Censoring"

Example of Survival Analysis

The failure of diesel generator fans was studied by Nelson (1982, p. 133) and Meeker and Escobar (1998, app. C1).

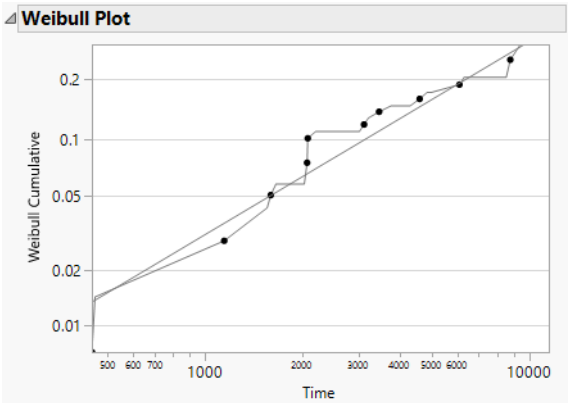
1. Select **Help > Sample Data Library** and open Reliability/Fan.jmp.
2. Select **Analyze > Reliability and Survival > Survival**.
3. Select Time and click **Y, Time to Event**.
4. Select Censor and click **Censor**.
5. Select the check box for **Plot Failure instead of Survival**.
6. Click **OK**.

Figure 13.8 Fan Initial Output



Notice that the probability of failure increases over time. Often the next step is to explore distributional fits, such as a Weibull model. Click the red triangle next to Product-Limit Survival Fit and select **Weibull Plot** and **Weibull Fit**.

Figure 13.9 Weibull Output for Fan Data



Extreme-Value Parameter Estimates

Parameter	Estimate	Lower 95%	Upper 95%	Number failed
λ	10.177204	9.5201194	11.572014	12
δ	0.9447814	0.6031578	1.6502353	12

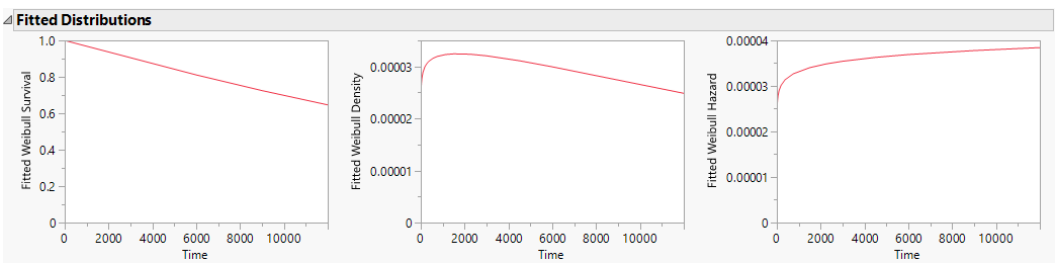
Weibull Parameter Estimates

same as Extreme-Value with α=exp(λ), β=1/δ

Parameter	Estimate	Lower 95%	Upper 95%	Number failed
α	26296.845	13631.239	106086.92	12
β	1.0584459	0.6059742	1.6579409	12

Because the fit is reasonable and the Beta estimate is near 1, you can conclude that this looks like an exponential distribution, which has a constant hazard rate. Click the red triangle next to Product-Limit Survival Fit and select **Fitted Distribution Plots**. Three views of the Weibull fit appear.

Figure 13.10 Fitted Distribution Plots

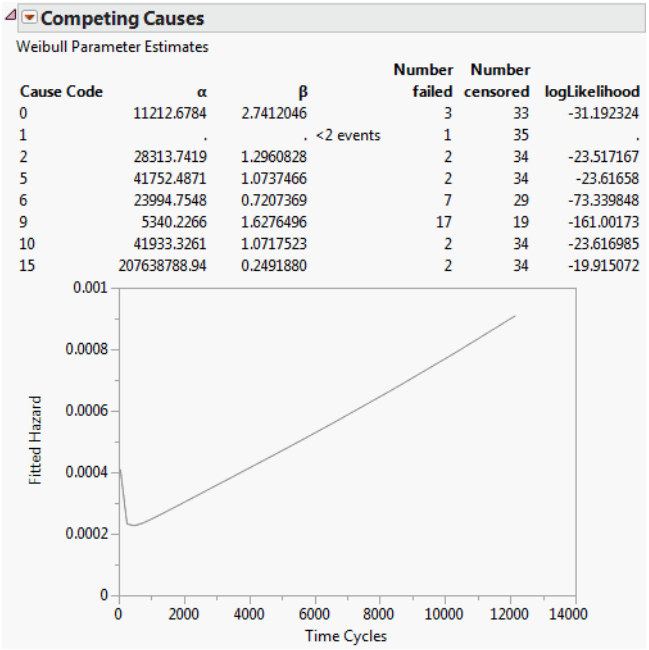


Example of Competing Causes

Nelson (1982) discusses the failure times of a small electrical appliance that has a number of causes of failure. One group (Group 2) of the data is represented in the JMP sample data table Appliance.jmp.

1. Select **Help > Sample Data Library** and open Reliability/Appliance.jmp.
2. Select **Analyze > Reliability and Survival > Survival**.
3. Select Time Cycles and click **Y, Time to Event**.
4. Click **OK**.
5. Click the red triangle next to Product-Limit Survival Fit and select **Competing Causes**.
6. Click Cause Code, and click **OK**.
7. Click the Competing Causes red triangle and select **Hazard Plot**.

Figure 13.11 Competing Causes Report and Hazard Plot

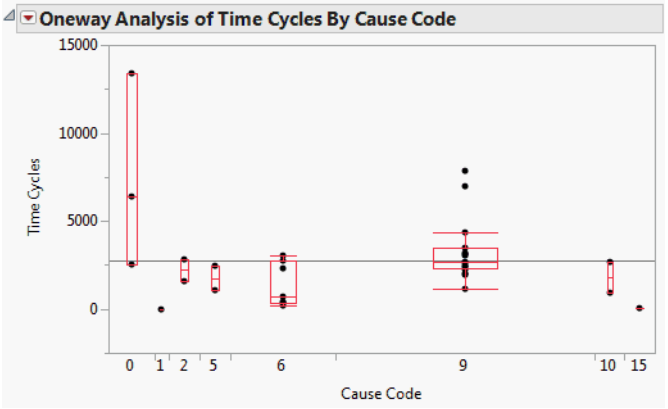


The survival distribution for the whole system is simply the product of the survival probabilities. The Competing Causes table shows the Weibull estimates of Alpha and Beta for each failure cause.

In this example, most of the failures were due to cause 9. Cause 1 occurred only once and could not produce good Weibull estimates. Cause 15 happened for very short times and resulted in a small beta and large alpha. Recall that alpha is the estimate of the 63.2% quantile of failure time, which means that causes with early failures often have very large alphas. If these causes do not result in early failures, then these causes do not usually cause later failures.

Figure 13.12 shows the Fit Y by X plot of Time Cycles by Cause Code with the Quantiles option in effect. This plot further illustrates how the alphas and betas relate to the failure distribution.

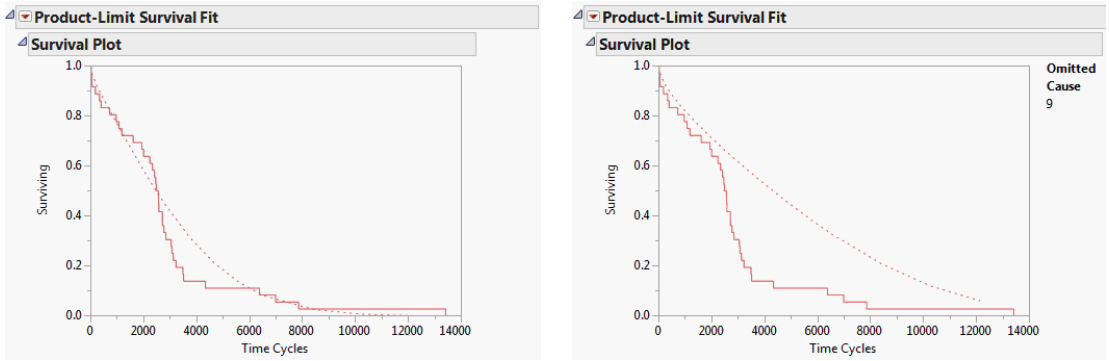
Figure 13.12 Fit Y by X Plot of Time Cycles by Cause Code



In this example, recall that cause 9 was the source of most of the failures. If cause 9 was corrected, how would that affect the survival due to the remaining causes? Select the **Omit Causes** option to remove a cause value and recalculate the survival estimates.

Figure 13.13 shows the survival plots with all competing causes and without cause 9. You can see that the survival rate (represented by the dashed line) without cause 9 does not improve much until 2,000 cycles. It then becomes much better and remains improved, even after 10,000 cycles.

Figure 13.13 Survival Plots with Omitted Causes



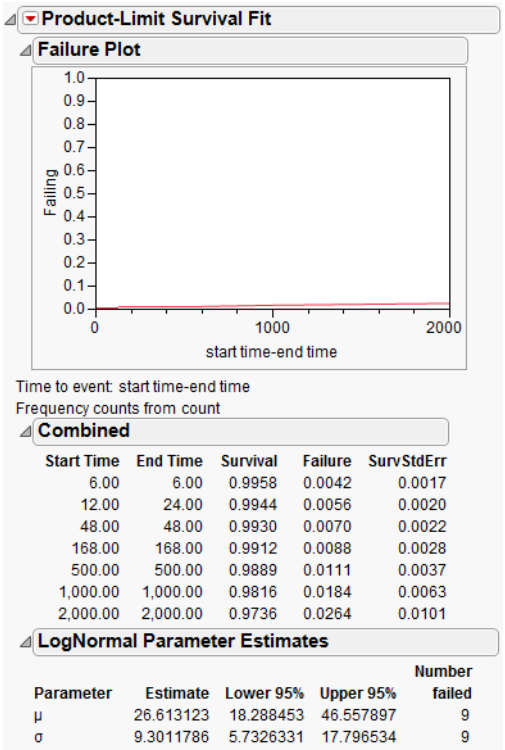
Example of Interval Censoring

With interval censored data, you know only that the events occurred in some time interval. The Turnbull method is used to obtain nonparametric estimates of the survival function.

In this example from Nelson (1990, p. 147), microprocessor units are tested and inspected at various times and the failed units are counted. Missing values in one of the columns indicate that you do not know the lower or upper limit, and therefore the event is left or right censored, respectively.

1. Select **Help > Sample Data Library** and open Reliability/Microprocessor Data.jmp.
2. Select **Analyze > Reliability and Survival > Survival**.
3. Select start time and end time and click **Y, Time to Event**.
4. Select count and click **Freq**.
5. Select the check box next to **Plot Failure instead of Survival**.
6. Click **OK**.
7. Click the red triangle next to Product-Limit Survival Fit and select **LogNormal Fit**.

Figure 13.14 Interval Censoring Output



The resulting Turnbull estimates are shown. Turnbull estimates might have gaps in time where the survival probability is not estimable. In this example, such gaps occur between 6 and 12, 24 and 48, 48 and 168, and so on.

At this point, select a distribution to see its fitted estimates —in this case, a Lognormal distribution is fit. Notice that the failure plot shows very small failure rates for these data.

Statistical Reports for Survival Analysis

For data that is not interval censored, the initial reports show Summary and Quantiles data (Figure 13.15). The Summary data shows the number of failed and number of censored observations for each group (when there are groups) and for the whole study. The mean and standard deviations are also adjusted for censoring. For computational details about these statistics, see the LIFETEST Procedure chapter in SAS Institute Inc. (2020).

The Quantiles data shows time to failure statistics for individual and combined groups. These include the median survival time, with upper and lower 95% confidence limits. The median survival time is the time (number of days) at which half the subjects have failed. The quartile survival times (25% and 75%) are also included.

Figure 13.15 Summary Statistics for the Univariate Survival Analysis

Product-Limit Survival Fit

Time to event: days
Censored by Censor
Censor Code 1
Grouped by Group

Summary

Group	Number failed	Number censored	Mean		Std Error
Group 1	19	2	240.795	Biased	11.206
Group 2	17	2	218.757		9.40318
Combined	36	4	230.729	Biased	7.57346

Quantiles

Group	Median Time	Lower 95%	Upper 95%	25% Failures	75% Failures
Group 1	233	232	280	232	280
Group 2	216	190	234	190	234
Combined	232	213	239	201.5	261

The Summary report gives estimates for the mean survival time, as well as the standard error of the mean. The estimated mean survival time is defined as follows:

$$\hat{\mu} = \sum_{i=1}^{D} \hat{S}(t_{i-1})(t_i - t_{i-1}) \quad \text{with a standard error of} \quad \hat{\sigma}(\hat{\mu}) = \sqrt{\frac{m}{m-1}\sum_{i=1}^{D-1}\frac{A_i^2}{n_i(n_i-d_i)}}$$

where

$$\hat{S}(t_i) = \prod_{j=1}^{i}\left(1 - \frac{d_j}{n_j}\right)$$

$$A_i = \sum_{j=i}^{D-1} \hat{S}(t_j)(t_{j+1} - t_j)$$

$$m = \sum_{j=1}^{D} d_j$$

$\hat{S}(t_i)$ is the survival distribution at time t_i

D is the number of distinct event times

n_i is the number of surviving units just prior to t_i

d_i is the number of units that fail at t_i

t_0 is defined to be 0

When there are multiple groups, the Tests Between Groups table provides statistical tests for homogeneity among the groups. Kalbfleisch and Prentice (1980, ch. 1), Hosmer and Lemeshow (1999, ch. 2), and Klein and Moeschberger (1997, ch. 7) discuss statistics and comparisons of survival curves.

Figure 13.16 Tests between Groups

Tests Between Groups

Test	ChiSquare	DF	Prob>ChiSq
Log-Rank	3.1227	1	0.0772
Wilcoxon	2.6510	1	0.1035

Test Names two statistical tests of the hypothesis that the survival functions are the same across groups.

Chi-Square Provides the Chi-square approximations for the statistical tests.

The **Log-Rank** test places more weight on larger survival times and is more useful when the ratio of hazard functions in the groups being compared is approximately constant. The hazard function is the instantaneous failure rate at a given time. It is also called the *mortality rate* or *force of mortality*.

The **Wilcoxon** test places more weight on early survival times and is the optimum rank test if the error distribution is logistic. See Kalbfleisch and Prentice (1980).

DF Provides the degrees of freedom for the statistical tests.

Prob>ChiSq Lists the probability of obtaining a Chi-square value greater than the one computed if the survival functions are the same for all groups.

Figure 13.17 shows an example of the product-limit survival function estimates for one group.

Figure 13.17 Example of Survival Estimates Table

Group 1

days	Survival	Failure	SurvStdErr	Number failed	Number censored	At Risk
0.000	1.0000	0.0000	0.0000	0	0	21
142.000	0.9524	0.0476	0.0465	1	0	21
156.000	0.9048	0.0952	0.0641	1	0	20
163.000	0.8571	0.1429	0.0764	1	0	19
198.000	0.8095	0.1905	0.0857	1	0	18
204.000	0.8095	0.1905	0.0857	0	1	17
205.000	0.7589	0.2411	0.0941	1	0	16
232.000	0.6577	0.3423	0.1053	2	0	15
233.000	0.4554	0.5446	0.1114	4	0	13
239.000	0.4048	0.5952	0.1099	1	0	9
240.000	0.3542	0.6458	0.1072	1	0	8
261.000	0.3036	0.6964	0.1031	1	0	7
280.000	0.2024	0.7976	0.0902	2	0	6
296.000	0.1012	0.8988	0.0678	2	0	4
323.000	0.0506	0.9494	0.0493	1	0	2
344.000	0.0506	0.9494	0.0493	0	1	1

Note: When the final time recorded is a censored observation, the report indicates a *biased* mean estimate. The biased mean estimate is a lower bound for the true mean.

Chapter 14

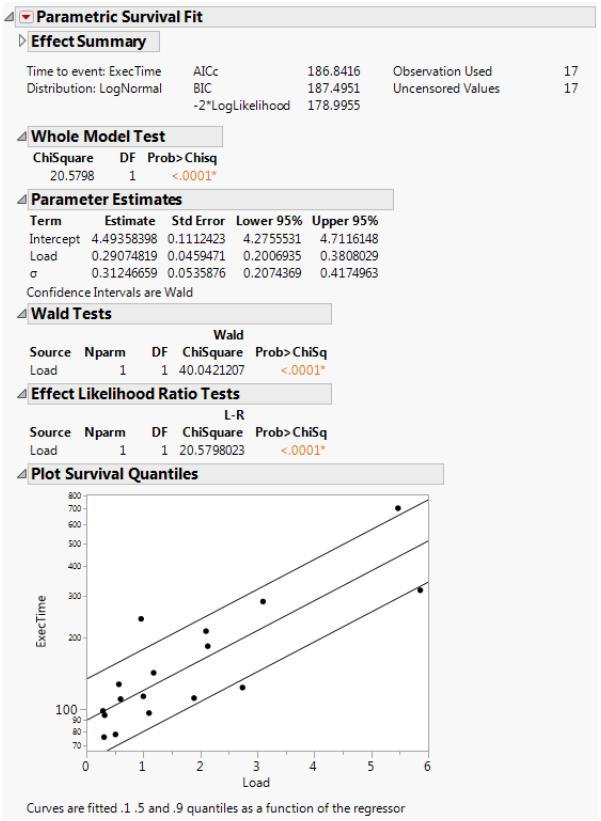
Fit Parametric Survival

Fit Survival Data Using Regression Models

Survival times can be expressed as a function of one or more variables. When this is the case, use a regression platform that fits a linear regression model while taking into account the survival distribution and censoring. The Fit Parametric Survival platform fits the time to event Y (with censoring) using linear regression models that can involve both location and scale effects. The fit is performed using the Weibull, lognormal, exponential, Fréchet, loglogistic, smallest extreme value (SEV), normal, largest extreme value (LEV), and logistic distributions.

Note: The Fit Parametric Survival platform is a slightly customized version of the Fit Model platform. You can also fit parametric survival models using the Nonlinear platform.

Figure 14.1 Example of a Parametric Survival Fit



Contents

Overview of the Fit Parametric Survival Platform . . . 371
Example of Parametric Regression Survival Fitting . . . 371
Launch the Fit Parametric Survival Platform . . . 373
The Parametric Survival Fit Report . . . 375
The Parametric Survival - All Distributions Report . . . 376
Parametric Competing Cause Report . . . 378
Fit Parametric Survival Options . . . 378
Nonlinear Parametric Survival Models . . . 380
Additional Examples of Fitting Parametric Survival . . . 381
Arrhenius Accelerated Failure LogNormal Model . . . 381
Interval-Censored Accelerated Failure Time Model . . . 385
Analyze Censored Data Using the Nonlinear Platform . . . 386
Left-Censored Data . . . 388
Weibull Loss Function Using the Nonlinear Platform . . . 389
Fitting Simple Survival Distributions Using the Nonlinear Platform . . . 392
Statistical Details for the Fit Parametric Survival Platform . . . 395
Loss Formulas for Survival Distributions . . . 395

Overview of the Fit Parametric Survival Platform

Survival times can be expressed as a function of one or more variables. When this is the case, use a regression platform that fits a linear regression model while taking into account the survival distribution and censoring. The Fit Parametric Survival platform fits the time to event Y (with censoring) using linear regression models that can involve both location and scale effects. The fit is performed using the Weibull, lognormal, exponential, Fréchet, loglogistic, SEV, normal, LEV, and logistic distributions.

Example of Parametric Regression Survival Fitting

The data table Comptime.jmp contains data on the analysis of computer program execution time whose lognormal distribution depends on the effect Load.

Note: The data in Comptime.jmp comes from Meeker and Escobar (1998, p. 434).

1. Select **Help > Sample Data Library** and open Reliability/Comptime.jmp.
2. Select **Analyze > Reliability and Survival > Fit Parametric Survival**.
3. Select ExecTime and click **Time to Event**.
4. Select Load and click **Add**.
5. Change the **Distribution** from **Weibull** to **Lognormal**.
6. Click **Run**.

Figure 14.2 Computing Time Output

Parametric Survival Fit

Effect Summary

Time to event: ExecTime	AICc	186.8416	Observation Used	17
Distribution: LogNormal	BIC	187.4951	Uncensored Values	17
	-2*LogLikelihood	178.9955		

Whole Model Test

ChiSquare	DF	Prob>Chisq
20.5798	1	<.0001*

Parameter Estimates

Term	Estimate	Std Error	Lower 95%	Upper 95%
Intercept	4.49358398	0.1112423	4.2755531	4.7116148
Load	0.29074819	0.0459471	0.2006935	0.3808029
σ	0.31246659	0.0535876	0.2074369	0.4174963

Confidence Intervals are Wald

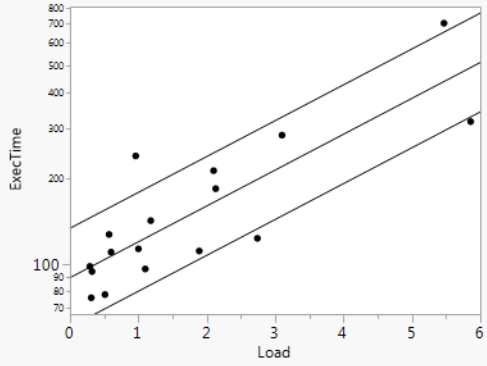
Wald Tests

Source	Nparm	DF	Wald ChiSquare	Prob>ChiSq
Load	1	1	40.0421207	<.0001*

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
Load	1	1	20.5798023	<.0001*

Plot Survival Quantiles



Curves are fitted .1 .5 and .9 quantiles as a function of the regressor

When there is only one effect, a plot of the survival quantiles for three survival probabilities are shown as a function of the effect.

Time quantiles are desired for when 90% of jobs are finished under a system load of 5. See Meeker and Escobar (1998, p. 438).

7. Click the Parametric Survival Fit red triangle and select **Estimate Quantile**.
8. Type 5 in the first row beneath **Load**.
9. Type 0.9 in the first row beneath **p**.
10. Click **Go**.

Figure 14.3 Estimates of Time Quantile

Estimates of Quantile

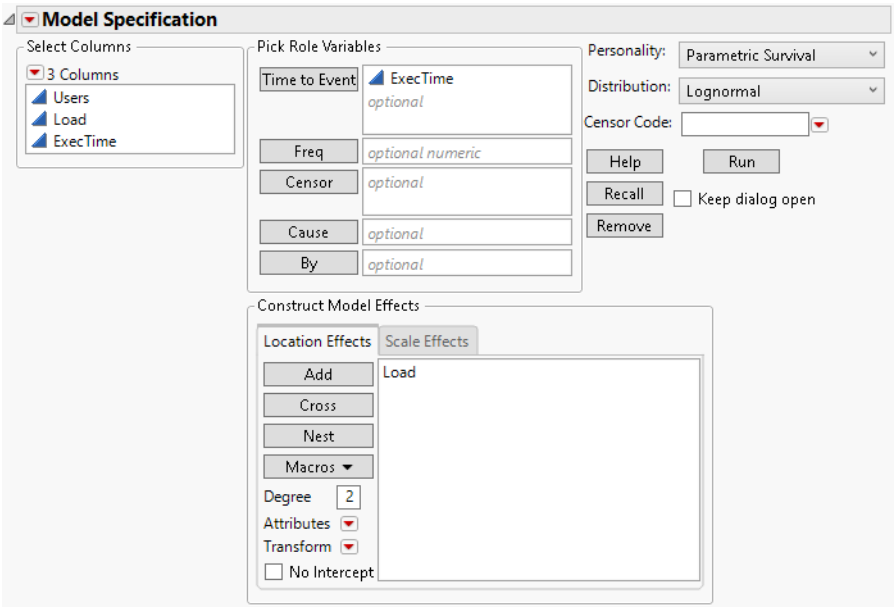
Load	p	Quantile	Lower 95%	Upper 95%
5	0.9	571.21575	401.29076	813.09482

The report estimates that 90% of the jobs will be done by 571 seconds of execution time under a system load of 5.

Launch the Fit Parametric Survival Platform

Launch the Fit Parametric Survival platform by selecting **Analyze > Reliability and Survival > Fit Parametric Survival**.

Figure 14.4 The Fit Parametric Survival Launch Window



Tip: To change the alpha level, click the Model Specification red triangle and select **Set Alpha Level**.

For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Fit Parametric Survival launch window contains the following options:

Time to Event Contains the time to event or time to censoring. With interval censoring, specify two *Y* variables, where one *Y* variable gives the lower limit and the other *Y* variable gives the upper limit for each unit.

Censor Specifies a column with indicators to identify right-censored observations. Select the value that identifies right-censored observations from the Censor Code menu. The Censor column is used only when one Time to Event column is entered.

Freq Specifies a column that contains the frequencies or counts of observations for each row when there are multiple units recorded.

Cause Specifies a column that contains multiple failure causes. This column is particularly useful for estimating competing causes. A separate parametric fit is performed for each cause value. Failure events can be coded with either numeric or categorical (labels) values.

By Performs a separate analysis for each level of a classification or grouping variable.

Location and Scale Effects Specifies location and scale effects. For more information about the Construct Model Effects options, see *Fitting Linear Models*.

Personality Indicates the fitting method. Parametric Survival should always be selected.

Distribution Choose the desired response distribution that is appropriate for your data. Choose the All Distributions option to fit all the distributions and compare the fits. If you choose All Distributions, the report shows a comparison of the distribution fits. See "The Parametric Survival - All Distributions Report".

Note: By default, the All Distributions option fits a model for the log-location-scale distributions that appear above All Distributions in the Distribution menu. Select **Preferences > Platforms > Fit Parametric Survival > Include location-scale distributions in All Distributions** to change the behavior of the All Distributions option to also include the location-scale distributions.

Censor Code Identifies the value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

The Parametric Survival Fit Report

If you select All Distributions in the launch window, a Parametric Survival Fit report appears for each distribution. If you specify a Cause column in the launch window, a Parametric Survival Fit report appears for each cause. Otherwise, only one Parametric Survival Fit report appears. Each Parametric Survival Fit report contains the following:

Effect Summary Shows an interactive report that enables you to add or remove effects from the model. See *Fitting Linear Models*.

Model Fit Details The Time to event shows which Y column is specified, and the Distribution shows which distribution is fit. AICc, BIC, and -2Loglikelihood are all measures of the model fit. These measures allow for comparisons to other model fits. Observation Used and Uncensored Values are summary statistics for the data. See *Fitting Linear Models*.

Whole Model Test Compares the complete fit with an intercept-only fit. If there is only an intercept term, the fit is the same as that from the Life Distribution platform.

Parameter Estimates Shows the estimates of the regression parameters.

JMP PRO A link to launch the Generalized Regression platform appears below the Parameter Estimates table. The link enables you to perform variable selection using the Generalized Regression platform and appears under the following circumstances:

- The model has no scale effects.
- No Cause column is specified in the launch window.
- The Distribution specified in the launch window is Normal, Lognormal, or Weibull.

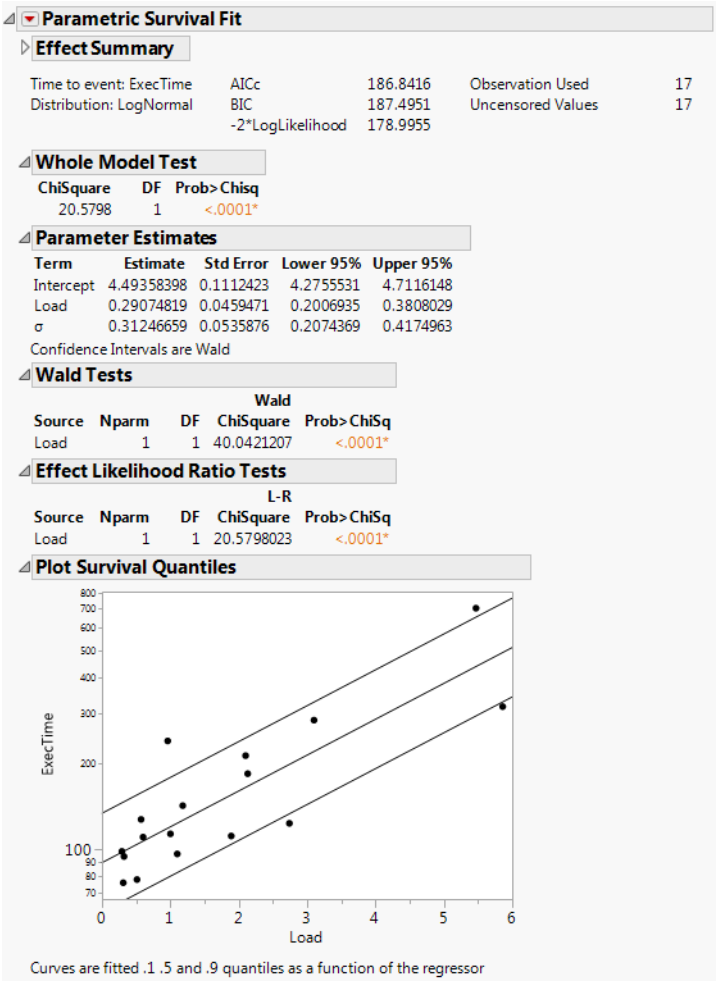
Alternate Parameterization (Available only for the Weibull distribution.) Shows the parameter estimates for the α and β parameterization of the Weibull distribution. For more information about this parameterization, see "Weibull".

Wald Tests Shows a Wald Chi-square test for each term in the model.

Effect Likelihood Ratio Tests Compare the log-likelihood from the fitted model to one that removes each term from the model individually.

Plot Survival Quantiles Shows the data points plotted with the 0.1, 0.5, and 0.9 quantiles.

Figure 14.5 The Parametric Survival Fit Report



The Parametric Survival - All Distributions Report

The Parametric Survival - All Distributions report appears only when you select All Distributions in the launch window. By default, this report contains a Model Comparison report and Distribution Overlay plot. The Quantile Function Overlay plot is available in the red triangle menu next to Parametric Survival - All Distributions.

Model Comparison Table that lists fit statistics (AICc and BIC) for the fitted distributions. The distributions with the smallest AICc and BIC values are labeled in the right-most column. If one distribution has the smallest value for both AICc and BIC, that distribution is labeled “Best”. The Parametric Survival Fit report corresponding to the distribution with

the smallest AICc is open by default. For more information about these statistics, see *Fitting Linear Models*.

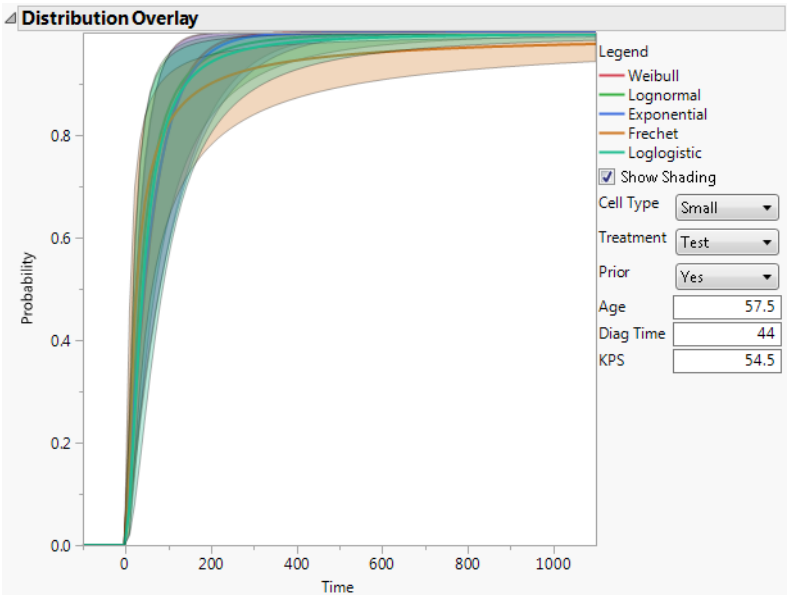
Distribution Overlay Plot of overlaid distribution functions for the fitted distributions at specified values of the effects.

Quantile Function Overlay Plot of overlaid quantile functions for the fitted distributions at specified values of the effects.

Both the Distribution Overlay and Quantile Function Overlay plots show curves for each of the fitted distributions overlaid on the same graph. By default, each curve has a shaded Wald-based confidence interval. To the right of each plot, there is legend, an option to show shading of confidence intervals, and controls that enable you to specify different values of the effects. Figure 14.6 shows an example of a Distribution Overlay plot.

Note: You can change the α level for the shaded confidence intervals by selecting Set Alpha Level from the red triangle menu in the Fit Model launch window. The default α level is 0.05.

Figure 14.6 Distribution Overlay Plot



Parametric Competing Cause Report

If you specify a Cause column in the launch window, the Parametric Competing Cause report appears. If you also specify All Distributions in the launch window, this report is labeled Parametric Competing Cause - All Distributions. The Parametric Competing Cause report contains the following:

Summary by Cause Table that lists fit statistics (AICc and BIC) for each cause. If All Distributions is selected for the Distribution option in the launch window, this table includes fit statistics for each cause within each distribution fit.

Model Comparison (Available only when All Distributions is selected for the Distribution option in the launch window.) Shows a table that lists fit statistics (AICc and BIC) for the fitted distributions. The distributions with the smallest AICc and BIC values are labeled in the right-most column. If one distribution has the smallest value for both AICc and BIC, that distribution is labeled "Best". The Parametric Survival Fit report corresponding to the distribution with the smallest AICc is open by default.

For more information about the AICc and BIC statistics, see *Fitting Linear Models*.

Fit Parametric Survival Options

The Parametric Survival Fit red triangle menu contains the following options:

Likelihood Ratio Tests Produces tests that compare the log-likelihood from the fitted model to one that removes each term from the model individually.

Wald Tests Produces chi-square test statistics and *p*-values for Wald tests of whether each parameter is zero.

Likelihood Confidence Intervals Specifies the type of confidence intervals shown in the Parameter Estimates table for each parameter. When this option is selected, a profile likelihood confidence interval appears. Otherwise, a Wald interval is shown. In the report, the interval type is noted below the Parameter Estimates table. This option is on by default when the computational time for the profile likelihood confidence intervals is not large.

Note: You can change the α level for the confidence intervals by selecting Set Alpha Level from the red triangle menu in the Fit Model launch window. The default α level is 0.05.

Correlation of Estimates Produces a correlation matrix for the model effects with each other and with the parameter of the fitting distribution.

Covariance of Estimates Produces a covariance matrix for the model effects with each other and with the parameter of the fitting distribution.

Estimate Survival Probability Estimates the failure and survival probabilities for the given time values. Specify effect values and one or more time values. JMP then calculates the survival and failure probabilities with 95% confidence limits for all possible combinations of the entries.

Estimate Quantile Estimates the quantiles for the given probabilities. Specify effect values and one or more quantile probabilities. JMP then calculates the time quantiles and 95% confidence limits for all possible combinations of the entries.

Note: For the Estimate Survival Probability and Estimate Quantile options, you can change the alpha level from the default of 0.05.

Residual Probability Plot Shows a probability plot of the standardized residuals. For location-scale distributions, the standardized residuals are defined as $(Y-\mu(X))/\sigma(X)$. For log-location-scale distributions, the standardized residuals are defined as $\exp(\log(Y)-\mu(X))/\sigma(X)$. For all distributions, $\mu(X)$ and $\sigma(X)$ are linear functions of the covariates defined by the model specification for the location and scale effects, respectively. These standardized residuals are a type of Cox-Snell residuals. See Meeker and Escobar (1998, sec. 17.6.1) for a discussion of Cox-Snell residuals.

Save Residuals Saves the residuals to a new column in the data table. For interval-censored observations, two columns of residuals are saved to the data table.

Distribution Profiler Shows the response surfaces of the failure probability versus individual explanatory and response variables.

Quantile Profiler Shows the response surfaces of the response variable versus the explanatory variable and the failure probability.

Distribution Plot by Level Combinations Shows three probability plots for assessing model fit. The plots show different lines for each combination of the X levels.

Separate Location A probability plot assuming equal scale parameters and separate location parameters. This is useful for assessing the parallelism assumption.

Separate Location and Scale A probability plot assuming different scale and location parameters. This is useful for assessing if the distribution is adequate for the data. This plot is not shown for the Exponential distribution.

Regression A probability plot for which the distribution parameters are functions of the X variables.

Save Probability Formula Saves the estimated probability formula to a new column in the data table.

Save Quantile Formula Saves the estimated quantile formula to a new column in the data table. Selecting this option displays a pop-up window, asking you to enter a probability value for the quantile of interest.

Publish Probability Formula Creates a probability formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Publish Quantile Formula Creates a quantile formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

Model Dialog Relaunches the launch window.

Effect Summary Shows the interactive Effect Summary report that enables you to add or remove effects from the model. See *Fitting Linear Models*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Nonlinear Parametric Survival Models

Use the Nonlinear platform for survival models in the following instances:

- The model is nonlinear.
- You need a distribution other than Weibull, lognormal, exponential, Fréchet, loglogistic, SEV, normal, LEV, or logistic.
- You have censoring that is not the usual right, left, or interval censoring.

With the ability to estimate parameters in specified loss functions, the Nonlinear platform becomes a powerful tool for fitting maximum likelihood models. For complete information about the Nonlinear platform, see *Predictive and Specialized Modeling*.

To fit a nonlinear model when data are censored, you must first use the formula editor to create a parametric equation that represents a loss function adjusted for censored observations. Then use the Nonlinear platform to estimate the parameters using maximum likelihood.

Additional Examples of Fitting Parametric Survival

- "Arrhenius Accelerated Failure LogNormal Model"
- "Interval-Censored Accelerated Failure Time Model"
- "Analyze Censored Data Using the Nonlinear Platform"
- "Left-Censored Data"
- "Weibull Loss Function Using the Nonlinear Platform"
- "Fitting Simple Survival Distributions Using the Nonlinear Platform"

Arrhenius Accelerated Failure LogNormal Model

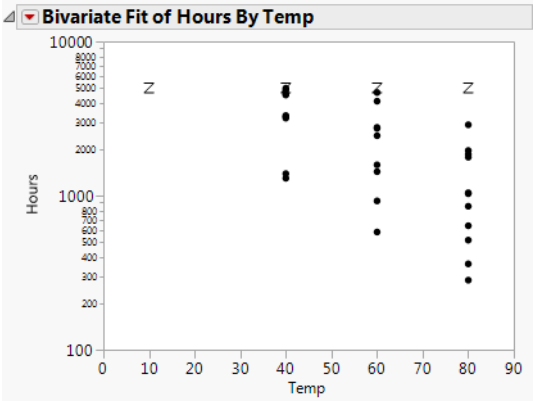
In the Devalt.jmp data, units are stressed by heating, in order to make them fail soon enough to obtain enough failures to fit the distribution.

Note: The data in Devalt.jmp comes from Meeker and Escobar (1998, p. 493).

1. Select **Help > Sample Data Library** and open Reliability/Devalt.jmp.

   First, use the Bivariate platform to see a plot of hours by temperature using the log scale for time.
2. Select **Analyze > Fit Y by X**.
3. Select Hours and click **Y, Response**.
4. Select Temp and click **X, Factor**.
5. Click **OK**.

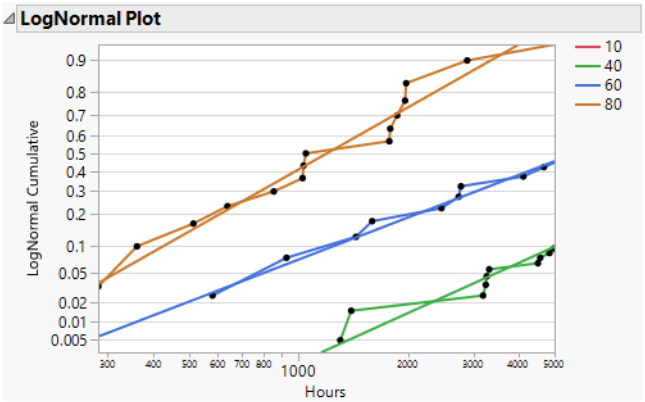
Figure 14.7 Bivariate Plot of Hours by Log Temp



Next, use the survival platform to produce a LogNormal plot of the data for each temperature.

6. Select **Analyze > Reliability and Survival > Survival**.
7. Select Hours and click **Y, Time to Event**.
8. Select Censor and click **Censor**.
9. Select Temp and click **Grouping**.
10. Select Weight and click **Freq**.
11. Click **OK**.
12. Click the red triangle next to Product-Limit Survival Fit and select **LogNormal Plot** and **LogNormal Fit**.
13. Click **OK**.

Figure 14.8 Lognormal Plot



Next, use the Fit Parametric Survival platform to fit one model using an effect for temperature.

14. Select **Analyze > Reliability and Survival > Fit Parametric Survival**.
15. Select Hours and click **Time to Event**.
16. Select x and click **Add**.
17. Select Censor and click **Censor**.
18. Select Weight and click **Freq**.
19. Change the **Distribution** type to **Lognormal**.
20. Click **Run**.

Figure 14.9 Devalt Parametric Output

Parametric Survival Fit

Effect Summary

Time to event: Hours
Distribution: Lognormal
Censored By: Censor
Freq Column: Weight

AICc	649.5546	Observation Used	165
BIC	658.7234	Uncensored Values	33
-2*LogLikelihood	643.4056	Right Censored Values	132

Whole Model Test

ChiSquare	DF	Prob>Chisq
80.7141	1	<.0001*

Parameter Estimates

Term	Estimate	Std Error	Lower 95%	Upper 95%
Intercept	-13.468649	2.8871953	-19.12745	-7.809851
x	0.62787903	0.0828422	0.4655112	0.7902468
σ	0.97782331	0.1326468	0.7178403	1.2378063

Confidence Intervals are Wald

Variable Selection using Generalized Regression

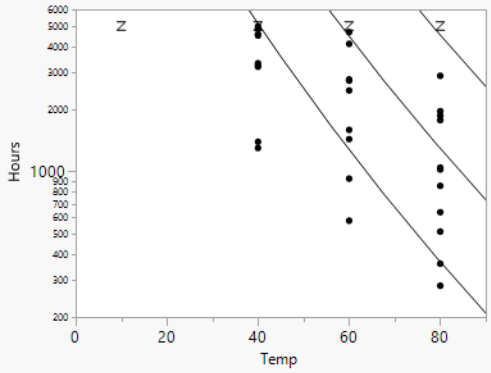
Wald Tests

Source	Nparm	DF	Wald ChiSquare	Prob>ChiSq
x	1	1	57.4444814	<.0001*

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
x	1	1	80.714074	<.0001*

Plot Survival Quantiles



Curves are fitted .1 .5 and .9 quantiles as a function of the regressor
Some of the points are censored times

The result shows the regression fit of the data:

– If there is only one effect and it is continuous, then a plot of the survival as a function of the effect is shown. Lines are at 0.1, 0.5, and 0.9 survival probabilities.

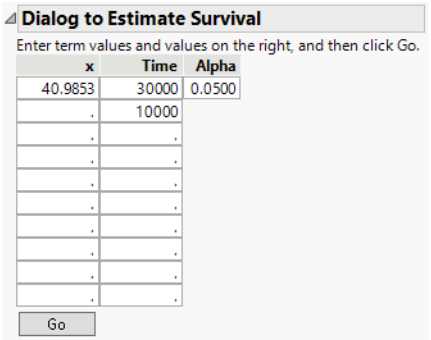
– If the effect column has a formula in terms of one other column, as in this case, the plot is done with respect to the inner column. In this case, the effect was the column x, but the plot is done with respect to Temp, of which x is a function.

Finally, get estimates of survival probabilities extrapolated to a temperature of 10 degrees Celsius for the times 30000 and 10000 hours.

21. Click the Parametric Survival Fit red triangle and select **Estimate Survival Probability**.
22. Enter the values shown in Figure 14.10 into the Dialog to Estimate Survival.

    The Arrhenius transformation of 10 degrees is 40.9853, the effect value.

Figure 14.10 Estimating Survival Probabilities



23. Click **Go**.

Figure 14.11 Survival Probabilities

Estimates of Survival

x	Time	Prob Failure	Lower 95%	Upper 95%	Prob Survival
40.9853	30000	0.0227191	0.0024223	0.1182138	0.9772809
40.9853	10000	0.0008917	4.3289e-5	0.0100954	0.9991083

The Estimates of Survival report shows the estimates and a confidence interval.

Interval-Censored Accelerated Failure Time Model

The ICdevice02.jmp data shows failures that were found to have happened between inspection intervals. The model uses two *y*-variables, containing the upper and lower bounds on the failure times. Right-censored times are shown with missing upper bounds.

Note: The data in ICdevice02.jmp comes from Meeker and Escobar (1998, p. 640).

1. Select **Help > Sample Data Library** and open Reliability/ICdevice02.jmp.
2. Select **Analyze > Reliability and Survival > Fit Parametric Survival**.
3. Select HoursL and HoursU and click **Time to Event**.
4. Select Count and click **Freq**.
5. Select x and click **Add**.
6. Click **Run**.

Figure 14.12 ICDevice Output

Parametric Survival Fit

Effect Summary

Time to event: HoursL .. HoursU
Distribution: Weibull
Freq Column: Count

AICc	185.9584	Observation Used	250
BIC	196.4252	Right Censored Values	194
-2*LogLikelihood	179.8608	Interval Censored Values	56

Whole Model Test

ChiSquare	DF	Prob>Chisq
195.5928	1	<.0001*

Parameter Estimates

Term	Estimate	Std Error	Lower 95%	Upper 95%
Intercept	-10.533672	2.0139264	-14.48089	-6.586449
x	0.85579005	0.0977613	0.6641815	1.0473986
δ	0.4376781	0.0500839	0.3395155	0.5358407

Confidence Intervals are Wald
Variable Selection using Generalized Regression

Alternate Parameterization

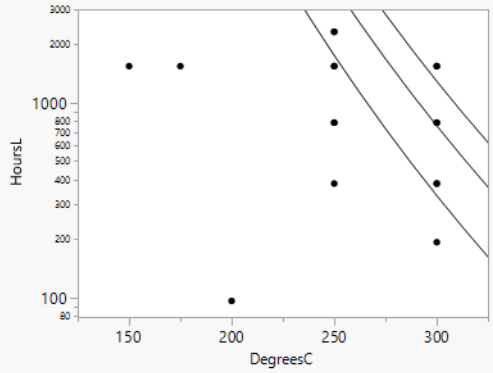
Wald Tests

Source	Nparm	DF	Wald ChiSquare	Prob>ChiSq
x	1	1	76.6303486	<.0001*

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
x	1	1	195.592803	<.0001*

Plot Survival Quantiles



Curves are fitted .1 .5 and .9 quantiles as a function of the regressor
Points are from lower and upper values of interval

The resulting regression shows a plot of time by degrees.

Analyze Censored Data Using the Nonlinear Platform

You can analyze left-censored data using the Nonlinear platform. For the left-censored data, zero is the censored value because it also represents the smallest known time for an observation.

Note: The Tobit model is popular in economics for responses that must be positive or zero, with zero representing a censored point.

Note the following about the Tobit2.jmp data table:

- The response variable is a measure of the durability of a product and cannot be less than zero (Durable, is left-censored at zero).
- Age and Liquidity are independent variables.
- The table also includes the model and Tobit loss function. The model in residual form is `durable-(b0+b1*age+b2*liquidity)`. To see the formula associated with Tobit Loss, right-click the column and select **Formula**.

Fit the Tobit model:

1. Select **Help > Sample Data Library** and open Reliability/Tobit2.jmp.
2. Select **Analyze > Specialized Modeling > Nonlinear**.
3. Select Model and click **X, Predictor Formula**.
4. Select Tobit Loss and click **Loss**.
5. Click **OK**.
6. Click **Go**.
7. Click **Confidence Limits**.

Figure 14.13 Solution Report

Solution

Loss	DFE	Avg Loss	Sqrt Avg Loss
28.92596097	16	1.8078726	1.3445715

Parameter	Estimate	ApproxStdErr	Lower CL	Upper CL
b0	15.277120063	16.0327167	-22.059913	54.7012259
b1	-0.134007501	0.21893135	-0.7365041	0.32841861
b2	-0.045135584	0.05826851	-0.1858906	0.09330941
Sigma	5.5693492202	1.72814365	3.31613117	11.5291865

Solved By: Analytic NR

Correlation of Estimates

Left-Censored Data

The Tobit model from the previous section assumes a normal distribution that is censored at zero. An observation of zero is considered to be left-censored. In the Fit Parametric Survival platform, left-censored observations are specified using two response columns. In this example, you create a new column that indicates observations for which left censoring has occurred. For left-censored observations, the new column contains a missing value. Otherwise, it contains the observed value of durable. The new column is then used as the left side of interval-censored observations, and the existing column durable is used as the right side of the intervals.

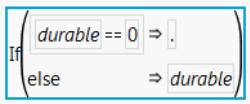
1. Select **Help > Sample Data Library** and open Reliability/Tobit2.jmp.

Create the Left Censoring Column:

2. Select **Cols > New Columns**.
3. Type durable0 for **Column Name**.
4. Select **Column Properties** and click **Formula**.
5. Select **Conditional > If** and select durable.
6. Select **Comparison > a == b**, type 0, and press Enter.
7. Select the box labeled "else clause" and select durable.

   Figure 14.14 shows the completed formula.

Figure 14.14 Column Formula for durable0



8. Click **OK**.
9. Click **OK**.

Fit the Tobit Model:

10. Select **Analyze > Reliability and Survival > Fit Parametric Survival**.
11. Select durable0 and click **Time to Event**.
12. Select durable and click **Time to Event**.

    You must use two response columns to specify left-censored observations. The direction of censoring is determined by the order of the columns in the Time to Event role.
13. Select age and liquidity and click **Add**.
14. Change the **Distribution** type to **Normal**.

15. Click **Run**.

Figure 14.15 Tobit Model Results

Parametric Survival Fit

Effect Summary

Time to event: durable0 .. durable
Distribution: Normal

AICc	68.51859	Observation Used	20
BIC	69.83485	Uncensored Values	7
-2*LogLikelihood	57.85192	Left Censored Values	13

Whole Model Test

ChiSquare	DF	Prob>Chisq
1.1325	2	0.5677

Parameter Estimates

Term	Estimate	Std Error	Lower 95%	Upper 95%
Intercept	15.2771208	16.032721	-16.14643	46.700676
age	-0.1340075	0.2189314	-0.563105	0.2950901
liquidity	-0.0451356	0.0582685	-0.15934	0.0690686
σ	5.56935051	1.7281446	2.1822494	8.9564517

Confidence Intervals are Wald
Variable Selection using Generalized Regression

Wald Tests

Source	Nparm	DF	Wald ChiSquare	Prob>ChiSq
age	1	1	0.37466429	0.5405
liquidity	1	1	0.60002606	0.4386

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
age	1	1	0.3931753	0.5306
liquidity	1	1	0.57102176	0.4499

The report shows the estimated Tobit model fit. Note in the upper right part of the report that there are 13 left-censored observations. This is a good way to check that you have correctly specified the left censoring.

Weibull Loss Function Using the Nonlinear Platform

In this example, models are fit to the survival time using the Weibull, lognormal, and exponential distributions. Model fits include a simple survival model containing only two effects, a more complex model with all the effects, and the creation of dummy variables for the discrete effect Cell Type to be included in the full model.

Nonlinear model fitting is often sensitive to the initial values that you give to the model parameters. In this example, one way to find reasonable initial values is to first use the Nonlinear platform to fit only the linear model. When the model converges, the solution values for the parameters become the initial parameter values for the nonlinear model.

1. Select **Help > Sample Data Library** and open VA Lung Cancer.jmp.

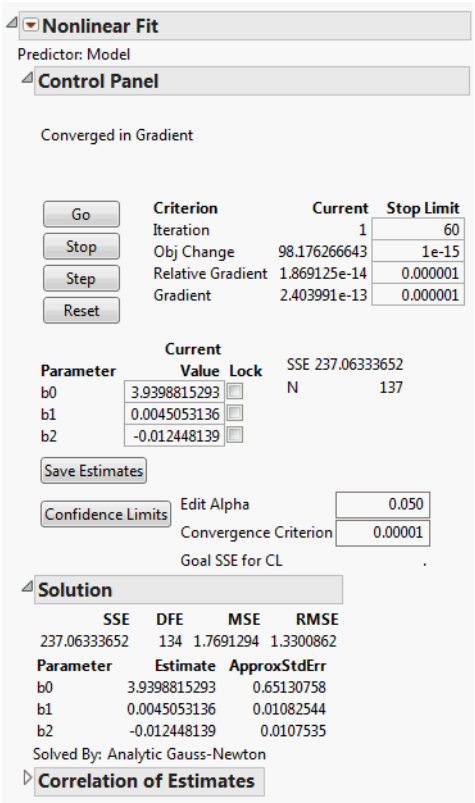
   The first model and all the loss functions have already been created as formulas in the data table. The Model column has the following formula:

   ```
   Log(:Time) - (b0 + b1 * Age + b2 * Diag Time)
   ```

2. Select **Analyze > Specialized Modeling > Nonlinear**.

3. Select Model and click **X, Predictor Formula**.
4. Click **OK**.
5. Click **Go**.

Figure 14.16 Initial Parameter Values in the Nonlinear Fit Control Panel



The report computes the least squares parameter estimates for this model.

6. Click **Save Estimates**.

   The parameter estimates in the column formulas are set to those estimated by this initial nonlinear fitting process.

   The Weibull column contains the Weibull formula, explained in "Weibull Loss Function".

   To continue with the fitting process:

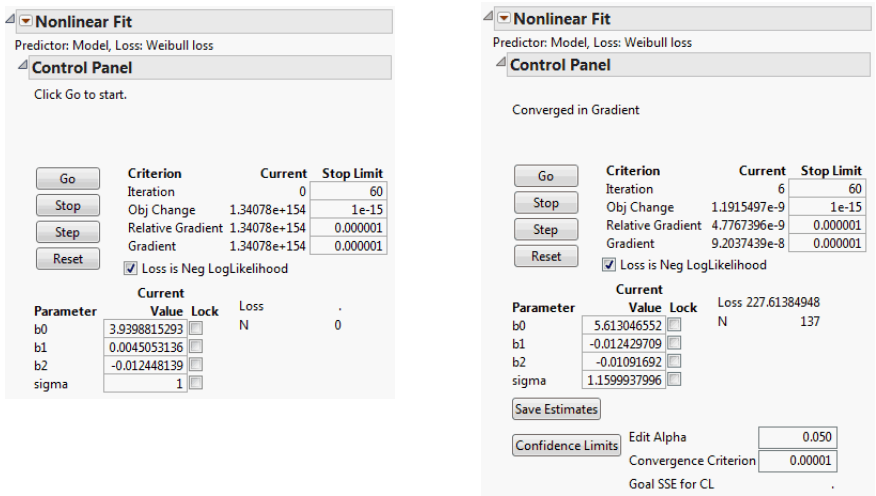
7. Select **Analyze > Specialized Modeling > Nonlinear** again.
8. Select Model and click **X, Predictor Formula**.
9. Select Weibull loss and click **Loss**.
10. Click **OK**.

The Nonlinear Fit Control Panel on the left in Figure 14.17 appears. There is now the additional parameter called sigma in the loss function. Because it is in the denominator of a fraction, a starting value of 1 is reasonable for sigma. When using any loss function other than the default, the **Loss is Neg LogLikelihood** box on the Control Panel is checked by default.

11. Click **Go**.

    The fitting process converges as shown on the right in Figure 14.17.

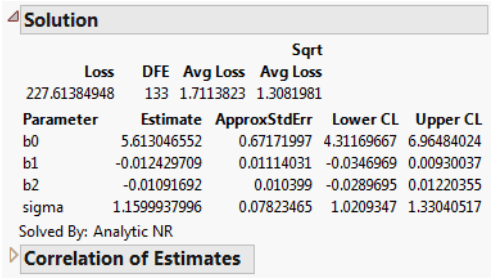
Figure 14.17 Nonlinear Model with Custom Loss Function



The fitting process estimates the parameters by maximizing the negative log of the Weibull likelihood function.

12. (Optional) Click **Confidence Limits** to show lower and upper 95% confidence limits for the parameters in the Solution table.

Figure 14.18 Solution Report



Note: Because the confidence limits are profile likelihood confidence intervals instead of the standard asymptotic confidence intervals, they can take time to compute.

You can also run the model with the predefined exponential and lognormal loss functions. Before you fit another model, reset the parameter estimates to the least squares estimates, as they might not converge otherwise. To reset the parameter estimates:

13. (Optional) Click the Nonlinear Fit red triangle and select **Revert to Original Parameters**.

Fitting Simple Survival Distributions Using the Nonlinear Platform

The following examples show how to use maximum likelihood methods to estimate distributions from time-censored data when there are no effects other than the censor status.

The Loss Function Templates folder has templates with formulas for exponential, extreme value, loglogistic, lognormal, normal, and one-and two-parameter Weibull loss functions. To use these loss functions, copy your time and censor values into the Time and censor columns of the loss function template. To run the model, select **Nonlinear** and assign the loss column as the **Loss** variable. Because both the response model and the censor status are included in the loss function and there are no other effects, you do not need a prediction column (model variable).

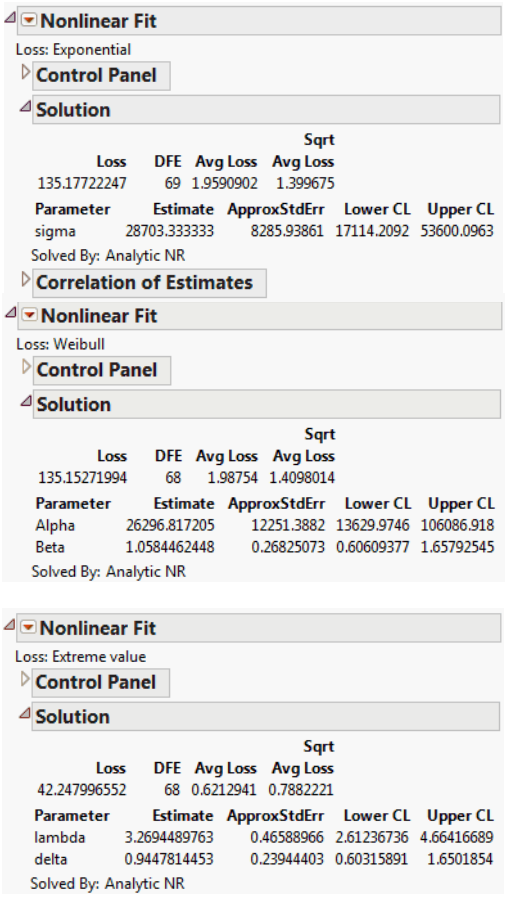
Exponential, Weibull, and Extreme-Value Loss Function

The Fan.jmp data table can be used to illustrate the Exponential, Weibull, and Extreme value loss functions discussed in Nelson (1982). The data are from a study of 70 diesel fans that accumulated a total of 344,440 hours in service. The fans were placed in service at different times. The response is failure time of the fans or run time, if censored.

Tip: To view the formulas for the loss functions, in the Fan.jmp data table, right-click the Exponential, Weibull, and Extreme value columns and select **Formula**.

1. Select **Help > Sample Data Library** and open Reliability/Fan.jmp.
2. Select **Analyze > Specialized Modeling > Nonlinear**.
3. Select Exponential and click **Loss**.
4. Click **OK**.
5. Make sure that the Loss is Neg LogLikelihood check box is selected.
6. Click **Go**.
7. Click **Confidence Limits**.
8. Repeat these steps, but select Weibull and Extreme value instead of Exponential.

Figure 14.19 Nonlinear Fit Results



Lognormal Loss Function

The Locomotive.jmp data can be used to illustrate a lognormal loss. The lognormal distribution is useful when the range of the data is several powers of *e*.

Tip: To view the formula for the loss function, in the Locomotive.jmp data table, right-click the logNormal column and select **Formula**.

The lognormal loss function can be very sensitive to starting values for its parameters. Because the lognormal distribution is similar to the normal distribution, you can create a new variable that is the natural log of Time and use **Distribution** to find the mean and standard deviation of this column. Then, use those values as starting values for the Nonlinear platform. In this example, the mean of the natural log of Time is 4.72 and the standard deviation is 0.35.

1. Select **Help > Sample Data Library** and open Reliability/Locomotive.jmp.

2. Select **Analyze > Specialized Modeling > Nonlinear**.
3. Select logNormal and click **Loss**.
4. Click **OK**.
5. Type 4.72 in the box next to Mu.
6. Type 0.35 in the box next to sigma.
7. Click **Go**.
8. Click **Confidence Limits**.

Figure 14.20 Solution Report

Solution

Loss	DFE	Avg Loss	Sqrt Avg Loss
237.09354534	94	2.5222718	1.5881662

Parameter	Estimate	ApproxStdErr	Lower CL	Upper CL
Mu	5.1169247193	0.10416697	4.93735853	5.35966436
sigma	0.7054940302	0.0932136	0.55446171	0.93294904

Solved By: Numerical NR

Correlation of Estimates

The maximum likelihood estimates of the lognormal parameters are 5.11692 for Mu and 0.7055 for Sigma. The corresponding estimate of the median of the lognormal distribution is the antilog of 5.11692, ($e^{5.11692}$), which is approximately 167. This represents the typical life for a locomotive engine.

Statistical Details for the Fit Parametric Survival Platform

This section contains statistical details for the Fit Parametric Survival platform.

Loss Formulas for Survival Distributions

The following formulas are for the negative log-likelihoods to fit common parametric models. Each formula uses the calculator `if` conditional function with the uncensored case of the conditional first and the right-censored case as the `Else` clause. You can copy these formulas from tables in the Loss Function Templates folder in Sample Data and paste them into your data table.

Exponential Loss Function

In the exponential loss function shown here, sigma represents the mean of the exponential distribution and Time is the age at failure.

$$-\text{If}\left(\begin{array}{ll} \text{Censor} == 0 & \Rightarrow -\text{Log}\left(\text{sigma}\right) - \dfrac{\text{Time}}{\text{sigma}} \\ \text{else} & \Rightarrow -\left(\dfrac{\text{Time}}{\text{sigma}}\right) \end{array}\right)$$

A characteristic of the exponential distribution is that the instantaneous failure rate remains constant over time. This means that the chance of failure for any subject during a given length of time is the same regardless of how long a subject has been in the study.

Weibull Loss Function

The Weibull density function often provides a good model for the lifetime distributions. You can use the Survival platform for an initial investigation of data to determine whether the Weibull loss function is appropriate for your data.

$$-\text{If}\left(\begin{array}{ll} \text{Censor} == 0 & \Rightarrow \dfrac{\text{Model}}{\text{sigma}} - \text{Exp}\left(\dfrac{\text{Model}}{\text{sigma}}\right) - \text{Log}\left(\text{sigma}\right) \\ \text{else} & \Rightarrow -\text{Exp}\left(\dfrac{\text{Model}}{\text{sigma}}\right) \end{array}\right)$$

There are examples of one-parameter, two-parameter, and extreme-value functions in the Loss Function Templates folder.

Lognormal Loss Function

The formula shown below is the lognormal loss function where `Normal Distribution(model/sigma)` is the standard normal distribution function. The hazard function has value 0 at $t = 0$, increases to a maximum, and then decreases. The hazard function approaches zero as t becomes large.

$$-\text{If}\left(\begin{array}{ll} censor==0 & \Rightarrow -0.5*\left(\frac{Model}{\mathbf{sigma}}\right)^2 - 0.5*\text{Log}(2*\text{Pi}) - \text{Log}(\mathbf{sigma}) \\ \text{else} & \Rightarrow \text{Log}\left(1 - \text{Normal Distribution}\left(\frac{Model}{\mathbf{sigma}}\right)\right) \end{array}\right)$$

Loglogistic Loss Function

If Y is distributed as the logistic distribution, Exp(Y) is distributed as the loglogistic distribution.

$$-\text{IfMZ}\left(\begin{array}{ll} \text{censor} == 0 & \Rightarrow \frac{\text{Model}}{\text{sigma}} - 2 * \text{Log}\left(1 + \text{Exp}\left(\frac{\text{Model}}{\text{sigma}}\right)\right) - \text{Log}(\text{sigma}) \\ \text{else} & \Rightarrow -\text{Log}\left(1 + \text{Exp}\left(\frac{\text{Model}}{\text{sigma}}\right)\right) \end{array}\right)$$

Chapter 15

Fit Proportional Hazards

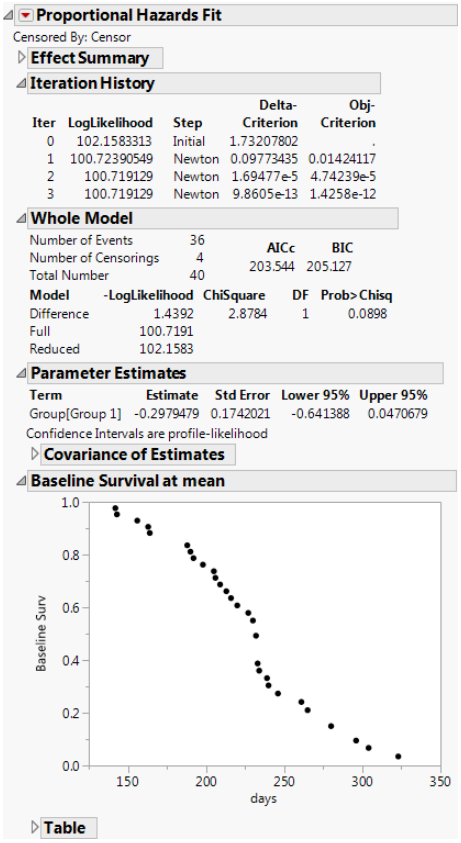
Fit Survival Data Using Semiparametric Regression Models

The Fit Proportional Hazards platform fits the Cox proportional hazards model, which assumes a multiplying relationship between covariates (predictors) and the hazard function.

Proportional hazards models are popular regression models for survival data with covariates. This model is semiparametric. The linear model is estimated, but the form of the hazard function is not. Time-varying covariates are not supported.

Note: The Fit Proportional Hazards platform is a slightly customized version of the Fit Model platform.

Figure 15.1 Example of a Proportional Hazards Fit



Contents

Overview of the Fit Proportional Hazards Platforms 399
Example of the Fit Proportional Hazards Platform 399
Risk Ratios for One Nominal Effect with Two Levels 401
Launch the Fit Proportional Hazards Platform 402
The Fit Proportional Hazards Report 404
Fit Proportional Hazards Options 405
Example Using Multiple Effects and Multiple Levels 406
Risk Ratios for Multiple Effects and Multiple Levels 408

Overview of the Fit Proportional Hazards Platforms

The proportional hazards model is a special semiparametric regression model proposed by D. R. Cox (1972) to examine the effect of explanatory variables on survival times. The survival time of each member of a population is assumed to follow its own hazard function.

The proportional hazards model is nonparametric in the sense that it involves an unspecified arbitrary baseline hazard function. It is parametric because it assumes a parametric form for the covariates. The baseline hazard function is scaled by a function of the model's (time-independent) covariates to give a general hazard function. Unlike the Kaplan-Meier analysis, proportional hazards computes parameter estimates and standard errors for each covariate. The regression parameters (β) associated with the explanatory variables and their standard errors are estimated using the maximum likelihood method. A conditional risk ratio (or hazard ratio) is also computed from the parameter estimates.

The survival estimates in proportional hazards are generated using an empirical method. See Lawless (1982). They represent the empirical cumulative hazard function estimates, $H(t)$, of the survivor function, $S(t)$, and can be written as $S_0 = \exp(-H(t))$. The hazard function is defined as follows:

$$\mathrm{H}(t) = \sum_{j:t_j < t} \frac{d_j}{\sum_{l \in R_j} e^{x_l \beta}}$$

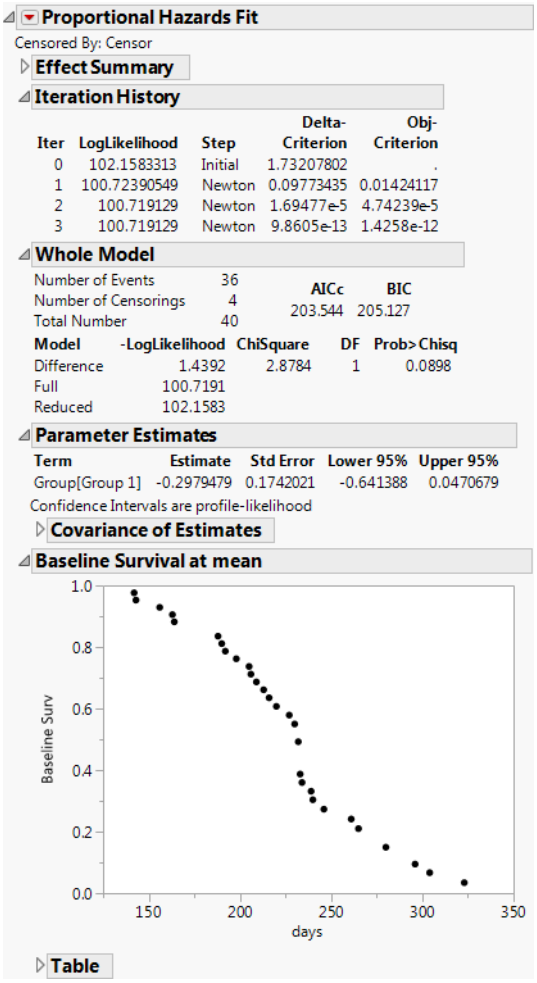
When there are ties in the response, meaning there is more than one failure at a given time event, the Breslow likelihood is used.

Example of the Fit Proportional Hazards Platform

This example illustrates one nominal effect with two levels. For an example with multiple effects and multiple levels, see "Example Using Multiple Effects and Multiple Levels".

1. Select **Help > Sample Data Library** and open Rats.jmp.
2. Select **Analyze > Reliability and Survival > Fit Proportional Hazards**.
3. Select days and click **Time to Event**.
4. Select Censor and click **Censor**.
5. Select Group and click **Add**.
6. Click **Run**.

Figure 15.2 Proportional Hazards Fit Report for Rats.jmp Data



In the Rats.jmp data, there are only two groups. Therefore, in the Parameter Estimates report, a confidence interval that does not include zero indicates an alpha-level significant difference between groups. Also, in the Effect Likelihood Ratio Tests report, the test of the null hypothesis for no difference between the groups shown in the Whole Model Test table is the same as the null hypothesis that the regression coefficient for Group is zero.

Risk Ratios for One Nominal Effect with Two Levels

To show risk ratios for effects, select the **Risk Ratios** option from the red triangle menu. In this example, there is only one effect, and there are only two levels for that effect. The risk ratio for Group 2 is compared with Group 1 and appears in the Risk Ratios for Group report. See Figure 15.3. The risk ratio in this table is determined by computing the exponential of the parameter estimate for Group 2 and dividing it by the exponential of the parameter estimate for Group 1.

Note the following:

- The Group 1 parameter estimate appears in the Parameter Estimates table (Figure 15.2).
- The Group 2 parameter estimate is calculated by taking the negative value for the parameter estimate of Group 1.
- Reciprocal shows the value for 1/Risk Ratio.

Tip: To see the Reciprocal values, right-click in the Risk Ratios report and select **Columns > Reciprocal**.

For this example, the risk ratio for Group2/Group1 is calculated as follows:

exp[-(-0.2979479)] /exp(-0.2979479) = 1.8146558

This risk ratio value suggests that the risk of death for Group 2 is 1.81 times that for Group 1.

Figure 15.3 Risk Ratios for Group Table

Risk Ratios

Risk Ratios for Group

Level1	/Level2	Risk Ratio	Prob>Chisq	Lower 95%	Upper 95%
Group 2	Group 1	1.8146558	0.0872	0.9167104	3.5921661
Group 1	Group 2	0.5510687	0.0872	0.2783836	1.0908571

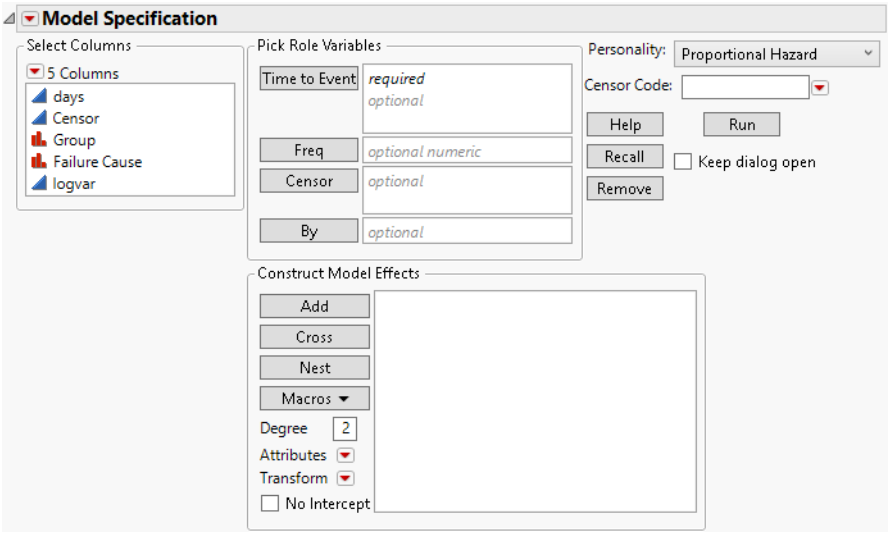
Normal approximations used for ratio confidence limits effects: Group

For information about calculating risk ratios when there are multiple effects, or categorical effects with more than two levels, see "Risk Ratios for Multiple Effects and Multiple Levels".

Launch the Fit Proportional Hazards Platform

Launch the Fit Proportional Hazards platform by selecting **Analyze > Reliability and Survival > Fit Proportional Hazards**.

Figure 15.4 The Fit Proportional Hazards Launch Window



Tip: To change the alpha level, click the Model Specification red triangle and select **Set Alpha Level**.

For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

The Fit Proportional Hazards launch window contains the following options:

Time to Event Contains the time to event or time to censoring.

Censor Specifies a column with indicators to identify right-censored observations. Select the value that identifies right-censored observations from the Censor Code menu.

Freq Column whose values are the frequencies or counts of observations for each row when there are multiple units recorded.

By Performs a separate analysis for each level of a classification or grouping variable.

Construct Model Effects Enters effects into your model. For more information about the Construct Model Effects options, see *Fitting Linear Models*.

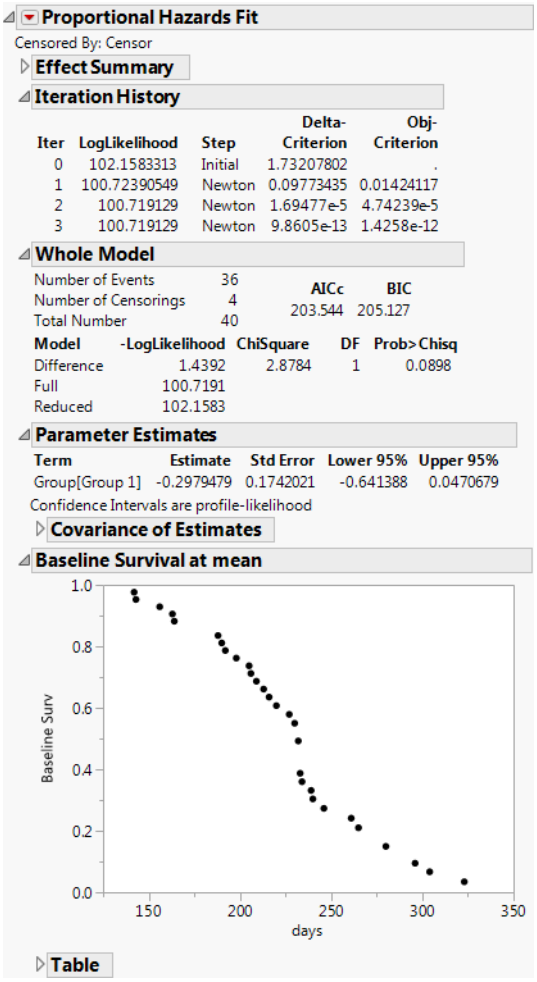
Personality Indicates the fitting method. Proportional Hazard should always be selected.

Censor Code Identifies the value in the Censor column that designates right-censored observations. After a Censor column is selected, JMP attempts to automatically detect the censor code and display it in the box. To change this, you can click the red triangle and select from a list of values. You can also enter a different value in the box. If the Censor column contains a Value Labels column property, the value labels appear in the list of values. Missing values are excluded from the analysis.

The Fit Proportional Hazards Report

Finding parameter estimates for a proportional hazards model is an iterative procedure. When the fitting is complete, the report in Figure 15.5 appears.

Figure 15.5 The Proportional Hazards Fit Report



Iteration History Lists iteration results occurring during the model calculations.

Whole Model Shows the negative of the log-likelihood function (–LogLikelihood) for the model with and without the covariates. Twice the positive difference between them gives a chi-square test of the hypothesis that there is no difference in survival time among the effects. The degrees of freedom (DF) are equal to the change in the number of parameters between the full and reduced models. See *Fitting Linear Models*.

Parameter Estimates Shows the parameter estimates for the covariates, their standard errors, and 95% upper and lower confidence limits. A confidence interval for a continuous column that does not include zero indicates that the effect is significant. A confidence interval for a level in a categorical column that does not include zero indicates that the difference between the level and the average of all levels is significant.

Effect Likelihood Ratio Tests Shows the likelihood ratio chi-square test of the null hypothesis that the parameter estimates for the effects of the covariates is zero.

Baseline Survival at mean Plots the baseline function estimates at each event time in the data. The values in the Table report are plotted here.

Fit Proportional Hazards Options

The Proportional Hazards Fit red triangle menu contains the following options:

Likelihood Ratio Tests Produces tests that compare the log-likelihood from the fitted model to one that removes each term from the model individually.

Wald Tests Produces chi-square test statistics and *p*-values for Wald tests of whether each parameter is zero.

Likelihood Confidence Intervals Specifies the type of confidence intervals shown in the Parameter Estimates table for each parameter. When this option is selected, a profile likelihood confidence interval appears. Otherwise, a Wald interval is shown. In the report, the interval type is noted below the Parameter Estimates table. This option is on by default when the computational time for the profile likelihood confidence intervals is not large.

Note: You can change the α level for the confidence intervals by selecting Set Alpha Level from the red triangle menu in the Fit Model launch window. The default α level is 0.05.

Risk Ratios Shows the risk ratios for the effects. For continuous columns, unit risk ratios and range risk ratios are calculated. The Unit Risk Ratio is Exp(estimate) and the Range Risk Ratio is Exp[estimate*($x_{Max} - x_{Min}$)]. The Unit Risk Ratio shows the risk change over one unit of the regressor, and the Range Risk Ratio shows the change over the whole range of the regressor. For categorical columns, risk ratios are shown in separate reports for each effect. Note that for a categorical variable with *k* levels, only *k* -1 design variables, or levels, are used.

Tip: To see Reciprocal values in the Risk Ratio report, right-click in the report and select **Columns > Reciprocal**.

Model Dialog Shows the completed launch window for the current analysis.

Effect Summary Shows the interactive Effect Summary report that enables you to add or remove effects from the model. See *Fitting Linear Models*.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Example Using Multiple Effects and Multiple Levels

This example uses a proportional hazards model for the sample data, VA Lung Cancer.jmp. The data were collected from a randomized clinical trial, where males with inoperable lung cancer were placed on either a standard or a novel (test) chemotherapy (Treatment). The primary interest of this trial was to assess if the treatment type has an effect on survival time; special interest is given to the type of tumor (Cell Type). See Prentice (1973) and Kalbfleisch and Prentice (2002) for more information about this data set.

For the proportional hazards model, covariates include the following:

- Whether the patient had undergone previous therapy (Prior)
- Age of the patient (Age)
- Time from lung cancer diagnosis to beginning the study (Diag Time)
- A general medical status measure (KPS)

Age, Diag Time, and KPS are continuous measures and Cell Type, Treatment, and Prior are categorical (nominal) variables. The four nominal levels of Cell Type include Adeno, Large, Small, and Squamous.

This example illustrates the results for a model with more than one effect and a nominal effect with more than two levels. Risk ratios are also demonstrated, with example calculations for risk ratios for a continuous effect and risk ratios for an effect that has more than two levels.

1. Select **Help > Sample Data Library** and open VA Lung Cancer.jmp.
2. Select **Analyze > Reliability and Survival > Fit Proportional Hazards**.

3. Select Time as **Time to Event**.
4. Select censor as **Censor**.
5. Select Cell Type, Treatment, Prior, Age, Diag Time, and KPS and click **Add**.
6. Click **Run**.
7. (Optional) Click the disclosure icon on the Baseline Survival at mean title bar to close the plot, and click the disclosure icon on Effect Summary to close the report.

Figure 15.6 Report Window for Proportional Hazards Model with Multiple Effects and Levels

Proportional Hazards Fit

Censored By: censor

Effect Summary

Iteration History

Whole Model

		AICc	BIC
Number of Events	128	967.484	989.719
Number of Censorings	9		
Total Number	137		

Model	-LogLikelihood	ChiSquare	DF	Prob>Chisq
Difference	30.7046	61.4091	8	<.0001*
Full	475.1794			
Reduced	505.8840			

Parameter Estimates

Term	Estimate	Std Error	Lower 95%	Upper 95%
Cell Type[Adeno]	0.57719588	0.1849861	0.2036319	0.9316196
Cell Type[Large]	-0.2114757	0.1741197	-0.56618	0.1191116
Cell Type[Small]	0.24538322	0.1592518	-0.070502	0.5550322
Treatment[Standard]	-0.1449679	0.1036051	-0.348984	0.0578618
Prior[No]	-0.0361633	0.1160663	-0.259353	0.1970409
Age	-0.0085494	0.0093042	-0.026447	0.0100846
Diag Time	-0.000092	0.0091251	-0.020233	0.016114
KPS	-0.0326217	0.0055052	-0.043442	-0.021834

Confidence Intervals are profile-likelihood

Covariance of Estimates

Baseline Survival at mean

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
Cell Type	3	3	18.6003364	0.0003*
Treatment	1	1	1.96251687	0.1612
Prior	1	1	0.09639285	0.7562
Age	1	1	0.82813759	0.3628
Diag Time	1	1	0.00010177	0.9920
KPS	1	1	34.616163	<.0001*

Note the following about the results:

- In the Whole Model report, the low Prob>Chisq value (<.0001) indicates that there is a difference in survival time when at least one of the effects is included in the model.
- In the Effect Likelihood Ratio Tests report, the Prob>ChiSq values indicate that KPS and at least one of the levels of Cell Type are significant; however, the Treatment, Prior, Age, and Diag Time effects are not significant.

8. Click the Proportional Hazards Fit red triangle and select **Risk Ratios**.

Figure 15.7 Risk Ratios Report

Risk Ratios

Unit Risk Ratios

Per unit change in regressor

Term	Risk Ratio	Lower 95%	Upper 95%	Reciprocal
Age	0.991487	0.97357	1.009733	1.0085861
Diag Time	0.999908	0.982184	1.017952	1.000092
KPS	0.967905	0.957517	0.978405	1.0331596

Range Risk Ratios

Per change in regressor over entire range

Term	Risk Ratio	Lower 95%	Upper 95%	Reciprocal
Age	0.669099	0.283964	1.576584	1.4945466
Diag Time	0.992119	0.213096	4.619042	1.0079435
KPS	0.05484	0.020991	0.143271	18.23482

Risk Ratios for Cell Type

Level1	/Level2	Risk Ratio	Prob>Chisq	Lower 95%	Upper 95%
Large	Adeno	0.4544481	0.0092*	0.2511038	0.8224612
Small	Adeno	0.7176217	0.2286	0.4181316	1.2316242
Squamous	Adeno	0.3047391	<.0001*	0.1690124	0.5494621
Small	Large	1.579106	0.0862	0.9370433	2.6611104
Squamous	Large	0.6705696	0.1574	0.3853373	1.166935
Adeno	Large	2.2004712	0.0092*	1.2158629	3.9824176
Squamous	Small	0.4246514	0.0019*	0.2476225	0.7282408
Adeno	Small	1.3934918	0.2286	0.811936	2.3915917
Large	Small	0.6332697	0.0862	0.375783	1.0671865
Adeno	Squamous	3.2814957	<.0001*	1.8199618	5.916725
Large	Squamous	1.4912695	0.1574	0.8569458	2.5951289
Small	Squamous	2.3548727	0.0019*	1.3731721	4.0384051

Risk Ratios for Treatment

Level1	/Level2	Risk Ratio	Prob>Chisq	Lower 95%	Upper 95%
Test	Standard	1.3363418	0.1617	0.8903074	2.0058347
Standard	Test	0.7483115	0.1617	0.4985456	1.1232076

Risk Ratios for Prior

Level1	/Level2	Risk Ratio	Prob>Chisq	Lower 95%	Upper 95%
Yes	No	1.0750063	0.7554	0.6820551	1.6943478
No	Yes	0.9302271	0.7554	0.5901976	1.4661572

Normal approximations used for ratio confidence limits effects: Cell Type Treatment Prior

Risk Ratios for Multiple Effects and Multiple Levels

Figure 15.7 shows the Risk Ratios for the continuous effects (Age, Diag Time, KPS) and the nominal effects (Cell Type, Treatment, Prior). For illustration, focus on the continuous effect, Age, and the nominal effect with four levels (Cell Type) for the VA Lung Cancer.jmp sample data.

For the continuous effect, Age, in the VA Lung Cancer.jmp sample data, the risk ratios are calculated as follows:

Unit Risk Ratios

$$\exp(\beta) = \exp(-0.0085494) = 0.991487$$

Range Risk Ratios

$\exp[\beta(xmax - xmin)] = \exp(-0.0085494 * 47) = 0.669099$

For the nominal effect, Cell Type, all pairs of levels are calculated and are shown in the Risk Ratios for Cell Type table. Note that for a categorical variable with *k* levels, only *k* -1 design variables, or levels, are used. In the Parameter Estimates table, parameter estimates are shown for only three of the four levels for Cell Type (Adeno, Large, and Small). The Squamous level is not shown, but it is calculated as the negative sum of the other estimates. Here are two example Risk Ratios for Cell Type calculations:

$\text{Large/Adeno} = \exp(\beta_{\text{Large}})/\exp(\beta_{\text{Adeno}}) = \exp(-0.2114757)/\exp(0.57719588) = 0.4544481$

$\text{Squamous/Adeno} = \exp[-(\beta_{\text{Adeno}} + \beta_{\text{Large}} + \beta_{\text{Small}})]/\exp(\beta_{\text{Adeno}})$

$= \exp[-(0.57719588 + (-0.2114757) + 0.24538322)]/\exp(0.57719588) = 0.3047391$

Reciprocal shows the value for 1/Risk Ratio.

Appendix A

References

Abernethy, R. B. (1996). *The New Weibull Handbook*. 2nd ed. North Palm Beach, FL: Robert B. Abernethy.

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Andrews, D. F., and Herzberg, A. M. (1985). *A Collection of Problems from Many Fields for the Student and Research Worker*. New York: Springer-Verlag.

Burnham, K. P., and Anderson, D. R. (2002). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. 2nd ed. New York: Springer-Verlag.

Chow, S.-C. (2007). *Statistical Design and Analysis of Stability Studies*. Boca Raton, FL: Chapman & Hall/CRC.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220.

Crow, L. H. (1975). *Reliability Analysis for Complex, Repairable Systems*. Technical Report No. 138, December 1975, US Army Materiel Systems Analysis Activity, Aberdeen Proving Ground, MD.

Crow, L. H. (1982). "Confidence Interval Procedures for the Weibull Process with Applications to Reliability Growth." *Technometrics* 24:67–72.

Escobar, L. A., Meeker, W. Q., Kugler, D. L., and Kramer, L. L. (2003). "Accelerated Destructive Degradation Tests: Data, Models, and Analysis." In *Mathematical and Statistical Methods in Reliability*, edited by B. H. Lindqvist and K. A. Doksum, 319–338. London: World Scientific Publishing Company.

Genschel, U., and Meeker, W. Q. (2010). "A Comparison of Maximum Likelihood and Median-Rank Regression for Weibull Estimation." *Quality Engineering* 22:236–255.

Guo, H., Mettas, A., Sarakakis, G., and Niu, P. (2010). "Piecewise NHPP Models with Maximum Likelihood Estimation for Repairable Systems." In *2010 Proceedings - Annual Reliability and Maintainability Symposium (RAMS).* New York: IEEE Press.

Hosmer, D. W., Jr., and Lemeshow, S. (1999). A*pplied Survival Analysis: Regression Modeling of Time-to-Event Data*. New York: John Wiley & Sons.

ICH Q1E. (2003). *Evaluation for Stability Data*. Tripartite International Conference on Harmonization Guideline Q1E, Geneva. https://database.ich.org/sites/default/files/Q1E_Guideline.pdf.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Kalbfleisch, J. D., and Prentice, R. L. (2002). *The Statistical Analysis of Failure Time Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Kaminskiy, M. P., and Krivtsov, V. V. (2005). "A Simple Procedure for Bayesian Estimation of the Weibull Distribution." *IEEE Transactions on Reliability* 54:612–616.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Lawless, J. F. (1982). *Statistical Models and Methods for Lifetime Data*. New York: John Wiley & Sons.

Lawless, J. F. (2003). *Statistical Models and Methods for Lifetime Data*. 2nd ed. New York: John Wiley & Sons.

Lee, L., and Lee S. K. (1978). "Some Results on Inference for the Weibull Process." *Technometrics* 20:41–45.

Liu, P., and Wang, P. (2013). "Competing Failure Modes Modeling with Limited Wearout Failures." In *2013 Proceedings Annual Reliability and Maintainability Symposium (RAMS)*. New York: IEEE Press.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Nair, V. N. (1984). "Confidence Bands for Survival Functions with Censored Data: A Comparative Study." *Technometrics* 26:265–275.

Nelson, W. B. (1982). *Applied Life Data Analysis*. New York: John Wiley & Sons.

Nelson, W. B. (1985). "Weibull Analysis of Reliability Data with Few or No Failures." *Journal of Quality Technology* 17:140–146.

Nelson, W. B. (1990). *Accelerated Testing: Statistical Models, Test Plans, and Data Analyses*. New York: John Wiley & Sons.

Nelson, W. B. (2003). *Recurrent Events Data Analysis for Product Repairs, Disease Recurrences, and Other Applications*. Philadelphia: Society for Industrial Mathematics.

Nelson, W. B. (2004). *Accelerated Testing: Statistical Models, Test Plans, and Data Analysis*. New York: John Wiley & Sons.

Prentice, R. L. (1973). "Exponential Survivals with Censoring and Explanatory Variables." *Biometrika* 60:279–288.

Rigdon, S. E., and Basu, A. P. (2000). *Statistical Methods for the Reliability of Repairable Systems*. New York: John Wiley & Sons.

Robert, C. P., and Casella, G. (2004). *Monte Carlo Statistical Methods*. 2nd ed. New York: Springer-Verlag.

SAS Institute Inc. (2020). "The LIFETEST Procedure." In *SAS/STAT 15.2 User's Guide*. Cary, NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/lifetest.pdf.

Si, S., Dui, H., Zhao, X., Zhang, S., and Sun, S. (2012). "Integrated Importance Measure of Component States Based on Loss of System Performance." *IEEE Transactions on Reliability*. 61:192–202.

Tobias, P. A., and Trindade, D. C. (1995). *Applied Reliability*. 2nd ed. New York: Van Nostrand Reinhold.

Tobias, P. A., and Trindade, D. C. (2012). *Applied Reliability*. 3rd ed. Boca Raton, FL: Chapman & Hall/CRC.

US Department of Defense (1981). *Military Handbook: Reliability Growth Management* (MIL-HDBK-00189). Washington, DC: US Department of Defense.

Appendix B

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

  - You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - You must cause any modified files to carry prominent notices stating that You changed the files; and
  - You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
  - You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Consumer Research

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Consumer Research*. Cary, NC: SAS Institute Inc.

JMP® 16 Consumer Research

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Consumer Research

1 Learn about JMP 9
Documentation and Additional Resources
Formatting Conventions in JMP Documentation 11
JMP Help 12
JMP Documentation Library 12
Additional Resources for Learning JMP 18
JMP Tutorials 18
Sample Data Tables 18
Learn about Statistical and JSL Terms 19
Learn JMP Tips and Tricks 19
JMP Tooltips 19
JMP User Community 20
Free Online Statistical Thinking Course 20
JMP New User Welcome Kit 20
Statistics Knowledge Portal 20
JMP Training 20
JMP Books by Users 21
The JMP Starter Window 21
JMP Technical Support 21

2 Introduction to Consumer Research 23
Overview of Customer and Behavioral Research Methods

3 Categorical Response Analysis 25
Analyze Survey and Other Counting Data
Example of the Categorical Platform 27
Launch the Categorical Platform 33
Response Roles 33
Columns Roles 38
Other Launch Window Options 38
The Categorical Report 40
Categorical Platform Options 41
Report Options 41

Statistical Testing Options ... 42
Additional Categorical Platform Options ... 44
Crosstab Table Options ... 47
Comparison Letters ... 48
Supercategories ... 49
Supercategories Options ... 50
Additional Examples of the Categorical Platform ... 51
Example of the Test for Response Homogeneity ... 51
Example of the Multiple Response Test ... 52
Example of Supercategories ... 54
Example of the Cell Chisq Test ... 57
Example of Compare Each Sample with Comparison Letters ... 58
Example of Compare Each Cell with Comparison Letters ... 59
Example of User-Specified Comparison with Comparison Letters ... 61
Example of Aligned Responses ... 62
Example of Conditional Association and Relative Risk ... 64
Example of Rater Agreement ... 66
Example of Repeated Measures ... 67
Example of the Multiple Responses ... 68
Example of Mean Score with Comparison Letters ... 73
Example of a Structured Report ... 75
Example of a Multiple Response with Nonresponse ... 76
Set Preferences ... 79
Statistical Details for the Categorical Platform ... 80
Rao-Scott Correction ... 80

4 Choice Models ... 81
Fit Models for Choice Experiments

Overview of the Choice Modeling Platform ... 83
Examples of the Choice Platform ... 85
One Table Format with No Choice ... 85
Multiple Table Format ... 88
Launch the Choice Platform ... 95
Launch Window for One Table, Stacked ... 96
Launch Window for Multiple Tables, Cross-Referenced ... 98
Choice Model Report ... 104
Effect Summary ... 104
Parameter Estimates ... 105
Likelihood Ratio Tests ... 106
Bayesian Parameter Estimates ... 106

Choice Platform Options ... 108
Willingness to Pay ... 111
Additional Examples ... 114
Example of Making Design Decisions ... 114
Example of Segmentation ... 122
Example of Logistic Regression Using the Choice Platform ... 126
Example of Logistic Regression for Matched Case-Control Studies ... 129
Example of Transforming Data to Two Analysis Tables ... 131
Example of Transforming Data to One Analysis Table ... 136
Statistical Details for the Choice Platform ... 138
Special Data Table Rules ... 138
Utility and Probabilities ... 139
Gradients ... 140

5 MaxDiff ... 141
Fit Models for MaxDiff Experiments
Overview of the MaxDiff Modeling Platform ... 143
Examples of the MaxDiff Platform ... 143
One Table Format ... 144
Multiple Table Format ... 147
Launch the MaxDiff Platform ... 150
Launch Window for One Table, Stacked ... 151
Launch Window for Multiple Tables, Cross-Referenced ... 153
MaxDiff Model Report ... 157
Effect Summary ... 157
MaxDiff Results ... 158
Parameter Estimates ... 159
Bayesian Parameter Estimates ... 161
Likelihood Ratio Tests ... 162
MaxDiff Platform Options ... 163
Comparisons Report ... 165
Save Bayes Chain ... 166

6 Uplift ... 167
Model the Incremental Impact of Actions on Consumer Behavior
Overview of the Uplift Platform ... 169
Example of the Uplift Platform ... 170
Launch the Uplift Platform ... 172
The Uplift Model Report ... 173
Uplift Model Graph ... 173
Uplift Report Options ... 176

7 Multiple Factor Analysis 177
Analyze Agreement among Panelists
Overview of the Multiple Factor Analysis Platform 179
Example of Multiple Factor Analysis 180
Launch the Multiple Factor Analysis Platform 183
Data Format 184
The Multiple Factor Analysis Report 186
Summary Plots 186
Consensus Map 187
Multiple Factor Analysis Platform Options 187
Statistical Details for the Multiple Factor Analysis Platform 190
A References 193
B Technology License Notices 195

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 11
JMP Help . . . 12
JMP Documentation Library . . . 12
Additional Resources for Learning JMP . . . 18
JMP Tutorials . . . 18
Sample Data Tables . . . 18
Learn about Statistical and JSL Terms . . . 19
Learn JMP Tips and Tricks . . . 19
JMP Tooltips . . . 19
JMP User Community . . . 20
Free Online Statistical Thinking Course . . . 20
JMP New User Welcome Kit . . . 20
Statistics Knowledge Portal . . . 20
JMP Training . . . 20
JMP Books by Users . . . 21
The JMP Starter Window . . . 21
JMP Technical Support . . . 21

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction to Consumer Research

Overview of Customer and Behavioral Research Methods

- The Categorical platform enables you to tabulate, plot, and compare categorical responses in your data, including multiple response data. You can use this platform to analyze data from surveys and other categorical response data, such as defect records and study participant demographics. Using the Categorical platform, you can analyze responses from data tables that are organized in many different ways. See "Categorical Response Analysis".
- The Choice platform is designed for use in market research experiments, where the ultimate goal is to discover the preference structure of consumers. Then, this information is used to design products or services that have the attributes most desired by consumers. See "Choice Models".
- The MaxDiff platform is an alternative to using standard preference scales to determine the relative importance of items being rated. A MaxDiff experiment forces respondents to report their most and least preferred options, thereby forcing respondents to rank options in terms of preference. See "MaxDiff".
- The Uplift platform enables you to maximize the impact of your marketing budget by sending offers only to individuals who are likely to respond favorably. It can do this even when you have large data sets and many possible behavioral or demographic predictors. You can use uplift models to make such predictions. This method has been developed to help optimize marketing decisions, define personalized medicine protocols, or, more generally, to identify characteristics of individuals who are likely to respond to some action. See "Uplift".
- The Multiple Factor Analysis platform enables you to analyze agreement among panelists in sensory data analysis. You can use MFA to analyze studies where items are measured on the same or different attributes by different instruments, individuals, or under different circumstances. See "Multiple Factor Analysis".

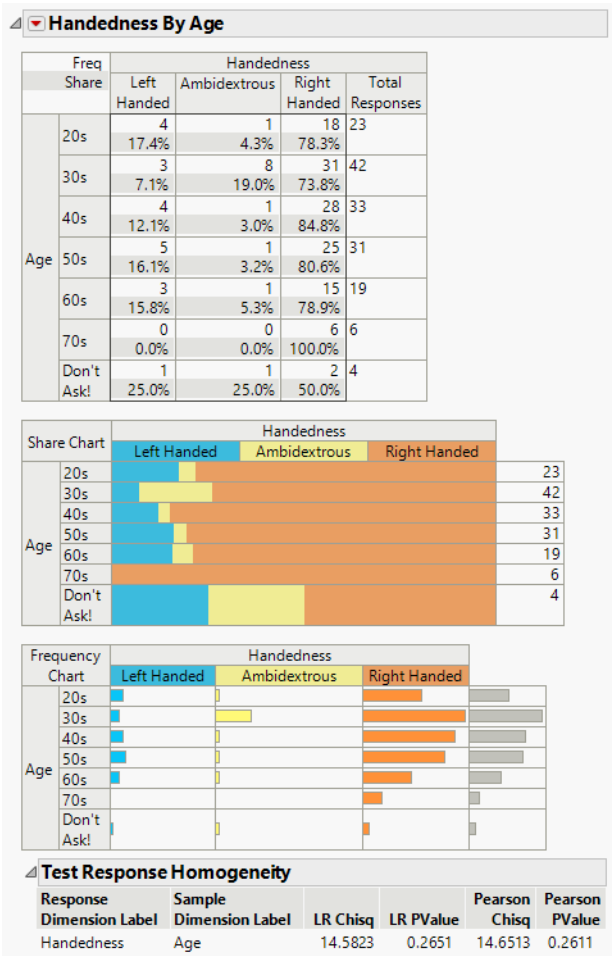
Chapter 3

Categorical Response Analysis

Analyze Survey and Other Counting Data

The Categorical platform enables you to tabulate, chart, and compare categorical response data, including multiple response data. You can analyze data from surveys and other categorical response data, such as defect records and study participant demographics. Many different data types and formats are supported.

Figure 3.1 Categorical Analysis Example



Handedness By Age

Freq / Share		Handedness: Left Handed	Ambidextrous	Right Handed	Total Responses
Age	20s	4 17.4%	1 4.3%	18 78.3%	23
	30s	3 7.1%	8 19.0%	31 73.8%	42
	40s	4 12.1%	1 3.0%	28 84.8%	33
	50s	5 16.1%	1 3.2%	25 80.6%	31
	60s	3 15.8%	1 5.3%	15 78.9%	19
	70s	0 0.0%	0 0.0%	6 100.0%	6
	Don't Ask!	1 25.0%	1 25.0%	2 50.0%	4

Share Chart — Handedness: Left Handed, Ambidextrous, Right Handed

Age: 20s 23; 30s 42; 40s 33; 50s 31; 60s 19; 70s 6; Don't Ask! 4

Frequency Chart — Handedness: Left Handed, Ambidextrous, Right Handed

Age: 20s, 30s, 40s, 50s, 60s, 70s, Don't Ask!

Test Response Homogeneity

Response Dimension Label	Sample Dimension Label	LR Chisq	LR PValue	Pearson Chisq	Pearson PValue
Handedness	Age	14.5823	0.2651	14.6513	0.2611

Contents

Example of the Categorical Platform . . . 27
Launch the Categorical Platform . . . 33
Response Roles . . . 33
Columns Roles . . . 38
Other Launch Window Options . . . 38
The Categorical Report . . . 40
Categorical Platform Options . . . 41
Report Options . . . 41
Statistical Testing Options . . . 42
Additional Categorical Platform Options . . . 44
Crosstab Table Options . . . 47
Comparison Letters . . . 48
Supercategories . . . 49
Supercategories Options . . . 50
Additional Examples of the Categorical Platform . . . 51
Example of the Test for Response Homogeneity . . . 51
Example of the Multiple Response Test . . . 52
Example of Supercategories . . . 54
Example of the Cell Chisq Test . . . 57
Example of Compare Each Sample with Comparison Letters . . . 58
Example of Compare Each Cell with Comparison Letters . . . 59
Example of User-Specified Comparison with Comparison Letters . . . 61
Example of Aligned Responses . . . 62
Example of Conditional Association and Relative Risk . . . 64
Example of Rater Agreement . . . 66
Example of Repeated Measures . . . 67
Example of the Multiple Responses . . . 68
Example of Mean Score with Comparison Letters . . . 73
Example of a Structured Report . . . 75
Example of a Multiple Response with Nonresponse . . . 76
Set Preferences . . . 79
Statistical Details for the Categorical Platform . . . 80
Rao-Scott Correction . . . 80

Example of the Categorical Platform

This example uses the Color Preference Survey.jmp sample data table, which contains survey data on color preferences. You can use the Categorical platform to summarize results from different question types:

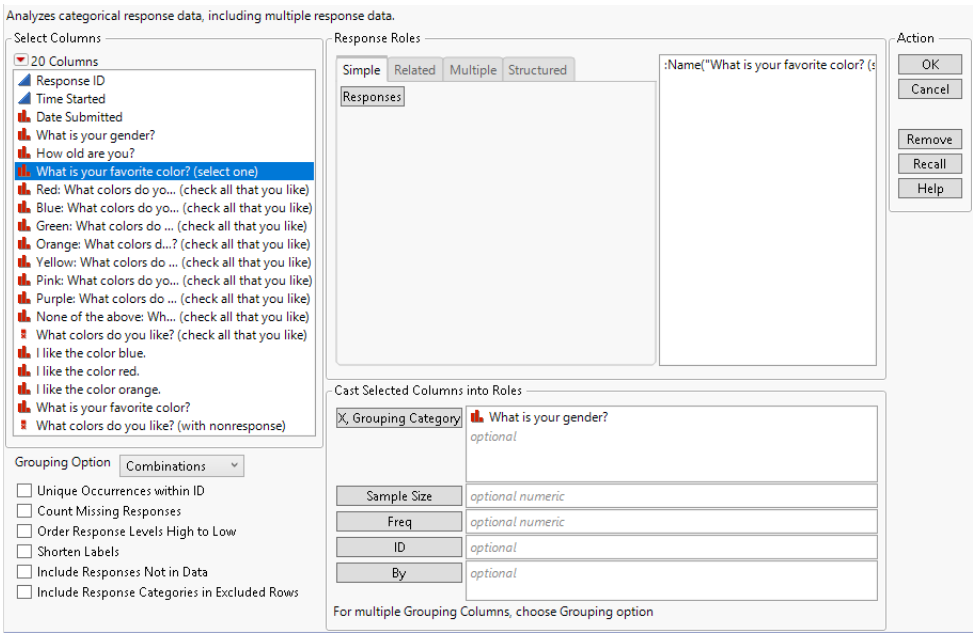
- A single response question
- Three aligned ranking questions
- A multiple response question
- A single response question segmented by a secondary question

You will set up four analyses (one for each type of question) in the Categorical launch window. Alternatively, you can use one launch tab at a time.

1. Select **Help > Sample Data Library** and open Color Preference Survey.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select What is your favorite color? (select one) and click **Responses**.
4. Select What is your gender? and click **X, Grouping Category**.

> **Note:** If you want to run the single analysis of favorite color by gender, click **OK** now.

Figure 3.2 Completed Simple Tab in the Categorical Launch Window

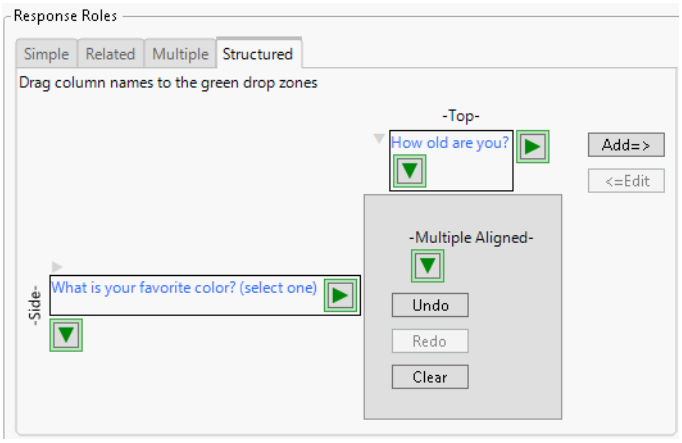


5. Select the **Related** tab.
6. Select I like the color blue. through I like the color orange. and click **Aligned Responses**.

   This enters three questions with rating scales to be analyzed together. The gender grouping category applies to this analysis.
7. Select the **Multiple** tab.
8. Select Red: What colors do you like? (check all that you like) through None of the above: What colors do you like? (check all that you like) and click **Multiple Response**.

   This enters the multiple response question "What colors do you like?" where each response is in an individual column for analysis. The gender grouping category applies to this analysis.
9. Select the **Structured** tab.
10. Drag What is your favorite color? (select one) to the Side green arrow and drag How old are you? to the Top green arrow.

Figure 3.3 Structured Tab in Categorical Launch Window

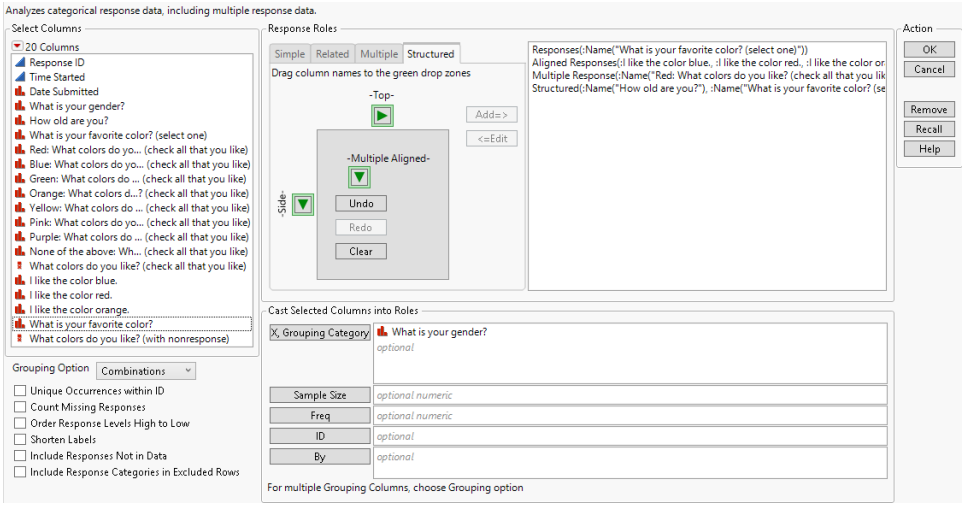


11. Click **Add**.

    The What is your gender? column specified in the X. Grouping Category role does not apply to the structured analysis.

Tip: Click on the green arrows in the structured tab for a column list, where you can click to add a column. You can nest multiple columns by using the down arrow on the top or the right arrow on the side.

Figure 3.4 Completed Launch Window for Full Example



12. Click **OK**.

The results for each analysis are stacked vertically in a single report window.

Simple Tab Report

The first section shows the results for the simple response survey question, “What is your favorite color?”, grouped by gender. Each respondent was asked to select one color from the list: Red, Blue, Green, Orange, Yellow, Pink, Purple, or None of the above.

Figure 3.5 Simple Response: Favorite Color by Gender

Categorical

What is your favorite color? (select one) By What is your gender?

Freq / Share		What is your favorite color? (select one)								
		Blue	Green	None of the above	Orange	Pink	Purple	Red	Yellow	Total Responses
What is your gender?	Female	11 29.7%	4 10.8%	6 16.2%	0 0.0%	1 2.7%	10 27.0%	3 8.1%	2 5.4%	37
	Male	19 44.2%	8 18.6%	5 11.6%	2 4.7%	1 2.3%	3 7.0%	5 11.6%	0 0.0%	43

The analysis shows that blue is the favorite color for both genders. For males, the frequency of the 43 male respondents selecting blue is 19. This corresponds to a share of 44.2% or 19/43. For females, 11 of 37 female respondents (frequency) or 29.7% (share) selected blue.

Tip: You can define the colors in the charts using the Value Colors column property. See *Using JMP*.

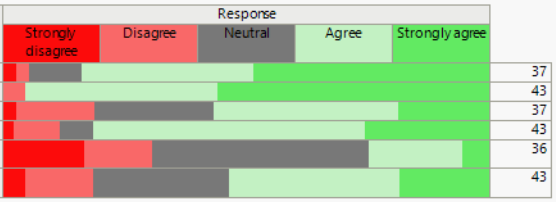
Related Tab Report

The second section of the report shows the results for three related questions that asked respondents to rank how much they liked a color. The responses to these questions are aligned, because the questions used the same rating scale of Strongly agree to Strongly disagree.

Figure 3.6 Aligned Response: Ranking Colors by Gender

Aligned Responses(I like the color blue., I like the color red., I like the color orange.) By What is your gender?

				Freq / Share	Response: Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Total Responses
Response	I like the color blue.	What is your gender?	Female		1 2.7%	1 2.7%	4 10.8%	13 35.1%	18 48.6%	37
			Male		0 0.0%	2 4.7%	0 0.0%	17 39.5%	24 55.8%	43
	I like the color red.	What is your gender?	Female		1 2.7%	6 16.2%	9 24.3%	14 37.8%	7 18.9%	37
			Male		1 2.3%	4 9.3%	3 7.0%	24 55.8%	11 25.6%	43
	I like the color orange.	What is your gender?	Female		6 16.7%	5 13.9%	16 44.4%	7 19.4%	2 5.6%	36
			Male		2 4.7%	6 14.0%	12 27.9%	15 34.9%	8 18.6%	43



The analysis shows that blue is the color with the highest rankings for both males and females. Orange has a high number of neutral results for both genders as compared to red and blue. The value colors column property is used to define the colors used in the share chart.

Note: If you analyze aligned questions individually using the Simple tab, you obtain three individual reports containing the same information as the related report. Using the Related tab aligns the reports so that the results are easier to compare to one another.

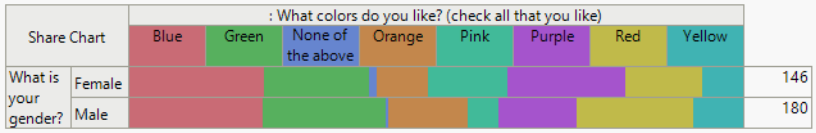
Multiple Tab Report

The third section in the report shows the results from a multiple choice survey question grouped by gender. The question was “What colors do you like? (check all that you like)”. Each possible response was collected in an individual column. If the subject selected a response, there is a value in the corresponding column. Otherwise, the column is empty.

Figure 3.7 Multiple Response: Colors Liked by Gender

Structured

Freq / Share / Rate		: What colors do you like? (check all that you like)										
		Blue	Green	None of the above	Orange	Pink	Purple	Red	Yellow	Total Responses	Total Cases	Total Cases Responding
What is your gender?	Female	32 21.9% 86.5%	25 17.1% 67.6%	2 1.4% 5.4%	12 8.2% 32.4%	19 13.0% 51.4%	28 19.2% 75.7%	18 12.3% 48.6%	10 6.8% 27.0%	146	37	37
	Male	39 21.7% 90.7%	36 20.0% 83.7%	1 0.6% 2.3%	23 12.8% 53.5%	9 5.0% 20.9%	23 12.8% 53.5%	34 18.9% 79.1%	15 8.3% 34.9%	180	43	43



From the results, we observe that blue and green are highly liked by both genders. Females like pink and purple at higher rates than males.

In a multiple response question, more than one answer is allowed. A case represents a single responder, and Total Cases is the total number of responders. In our data, we have 37 female cases and 43 male cases. The total number of cases responding are the number of cases who selected one or more responses (which in our data, was everyone).

The Total Responses column lists the total number of selections made. For females, there were 146 colors selected by the 37 females. For males, there were 180 colors selected by the 43 male responders. The tabulation includes the number of responders selecting each color, the share or the percent of the total responses, and the rate or the percent of the total cases.

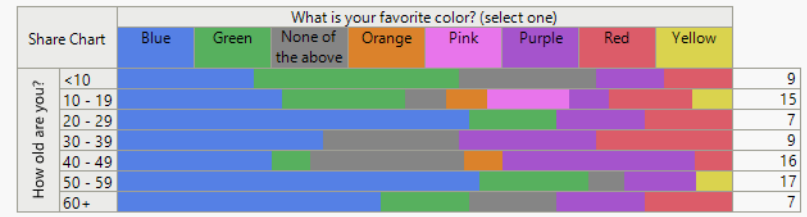
Structured Tab Report

The final section shows the table generated from the structured tab where you set the favorite color question as the response of interest grouped by the age group. In the structured format, the levels of the primary response of interest form the rows of the table and the levels of the grouping variable form the columns.

Figure 3.8 Structured: Favorite Color by Age

What is your favorite color? (select one) By How old are you?

What is your favorite color? (select one)	Freq / Share	<10	10 - 19	20 - 29	30 - 39	40 - 49	50 - 59	60+
	Blue	2 22.2%	4 26.7%	4 57.1%	3 33.3%	4 25.0%	10 58.8%	3 42.9%
	Green	3 33.3%	3 20.0%	1 14.3%	0 0.0%	1 6.3%	3 17.6%	1 14.3%
	None of the above	2 22.2%	1 6.7%	0 0.0%	2 22.2%	4 25.0%	1 5.9%	1 14.3%
	Orange	0 0.0%	1 6.7%	0 0.0%	0 0.0%	1 6.3%	0 0.0%	0 0.0%
	Pink	0 0.0%	2 13.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%
	Purple	1 11.1%	1 6.7%	1 14.3%	2 22.2%	5 31.3%	2 11.8%	1 14.3%
	Red	1 11.1%	2 13.3%	1 14.3%	2 22.2%	1 6.3%	0 0.0%	1 14.3%
	Yellow	0 0.0%	1 6.7%	0 0.0%	0 0.0%	0 0.0%	1 5.9%	0 0.0%
	Total Responses	9	15	7	9	16	17	7

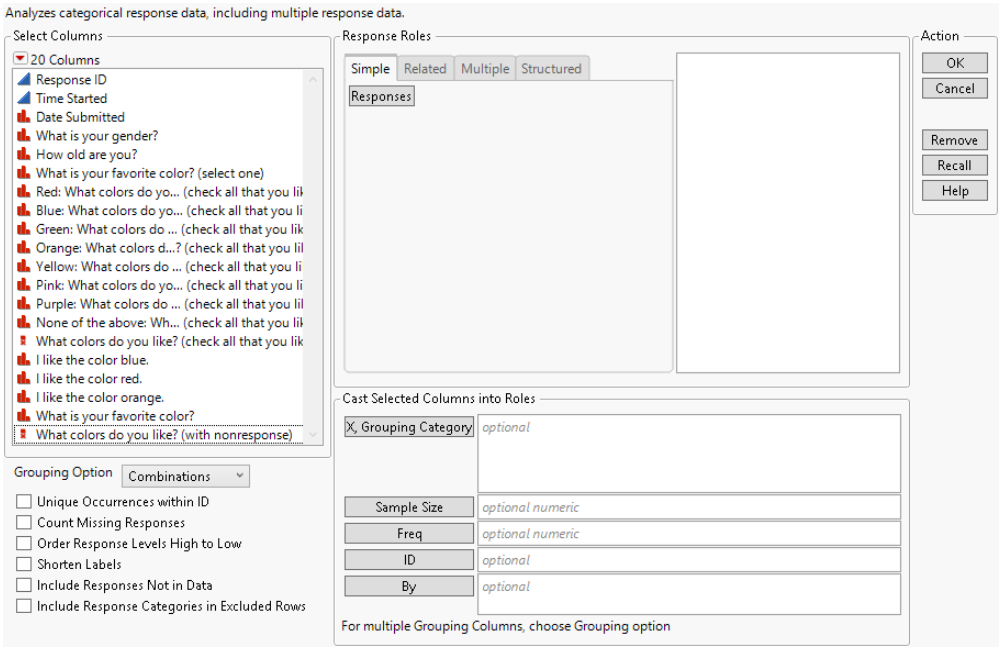


The 10 to 19 age group has two responders with pink as their favorite color. No other age groups have responders with pink as their favorite color. You can see from the bottom row of the table that the number of responders in each age group is small. You could gather more data to draw conclusions about the favorite colors across age groups.

Launch the Categorical Platform

Launch the Categorical platform by selecting **Analyze > Consumer Research > Categorical**.

Figure 3.9 Categorical Platform Launch Window for the Simple Tab



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Response Roles

The launch window includes tabs for three groups of response roles (Simple, Related, and Multiple) and a Structured tab where you can create custom data summaries. The response role corresponds to the type of responses you want to analyze. Options on each tab correspond to how the responses are organized in your data table.

Simple Tab

Use the Simple tab to analyze a single response, such as a survey question where only a single answer is allowed. The data to be analyzed is contained in a single column.

Responses Summarizes data from a single column. If multiple columns are selected, the categorical report contains a separate report for each individual column.

Related Tab

Use the Related tab to analyze a set of related responses across multiple columns.

Figure 3.10 Categorical Platform Launch Window Related Tab

Aligned Responses Summarizes data from multiple columns that have the same response levels in a single report. This option is useful for survey data where you have many questions with the same set of responses. You can quickly summarize and compare response trends for all of the questions at once.

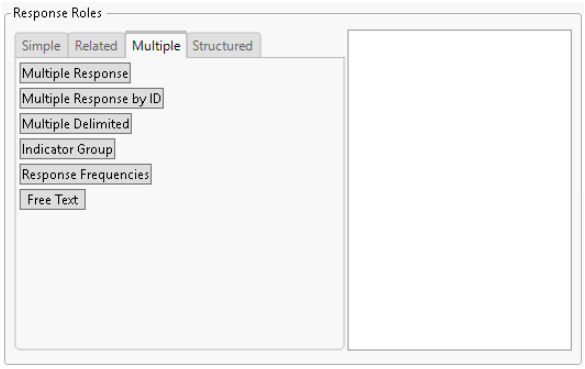
Repeated Measures Summarizes data from multiple columns where each column contains responses to the same question made at different time points. If an individual responds at multiple time points, the samples are called overlapping. When there are overlapping samples, the Kish correction is used. See Kish (1965, sec. 12.4).

Rater Agreement Summarizes data from multiple columns where each column is a rating for the same question or item, but is given by a different individual (rater).

Multiple Tab

Use the Multiple tab to analyze multiple responses where the responses are recorded in one or more columns. The options on the Multiple tab are specific to how the data is organized in your data table. A set of multiple responses could be from a survey where the response set allows for more than one choice (check-all-that apply questions), or from a set of defect data where an item can have multiple defects.

Figure 3.11 Categorical Platform Launch Window Multiple Tab



Multiple Response Summarizes data from multiple responses where each possible response is recorded in its own individual column. Each column can contain blanks, which correspond to the item not being selected.

Figure 3.12 Multiple Response Column Format

	Red: What colors do you like? (check all that you like)	Blue: What colors do you like? ...	Green: What colors do you ...	Orange: What colors do you ...	Yellow: What colors do you ...	Pink: What colors do you like? ...	Purple: What colors do you ...	None of the above: What colors do you like? (check all that you like)
1	Red	Blue				Pink	Purple	
2	Red	Blue	Green					
3	Red	Blue	Green	Orange			Purple	
4								None of the above
5		Blue	Green	Orange			Purple	
6	Red	Blue	Green				Purple	
7	Red	Blue		Orange				
8	Red		Green	Orange				
9	Red	Blue	Green				Purple	
10	Red	Blue	Green	Orange	Yellow	Pink	Purple	

Multiple Response by ID Summarizes data from multiple responses where the data are recorded in a stacked format. There is a single column of responses with a second column containing an ID for the subject.

Figure 3.13 Multiple Response by ID Format

	Response ID	What colors do you like?
1	2	Red
2	2	Blue
3	2	
4	2	
5	2	
6	2	Pink
7	2	Purple
8	2	
9	3	Red
10	3	Blue

Multiple Delimited Summarizes data from multiple responses where the responses are in a single column and each response is separated by a comma, semicolon, or tab.

Figure 3.14 Delimited Multiple Response Format

	What colors do you like? (check all that you like)
1	Red, Blue, , , , Pink, Purple,
2	Red, Blue, Green, , , , ,
3	Red, Blue, Green, Orange, , , Purple,
4	, , , , , , , None of the above
5	, Blue, Green, Orange, , , Purple,
6	Red, Blue, Green, , , , Purple,
7	Red, Blue, , Orange, , , ,
8	Red, , Green, Orange, , , ,
9	Red, Blue, Green, , , , Purple,
10	Red, Blue, Green, Orange, Yellow, Pink, Purple,

Tip: Use the Multiple Responses column property for columns that contain delineated multiple responses. See *Using JMP*.

Indicator Group Summarizes multiple responses that are stored in indicator columns. The data table contains a column for each possible response, and each column is an indicator (for example, 0 and 1). A blank value indicates a missing response. See "Indicator Group".

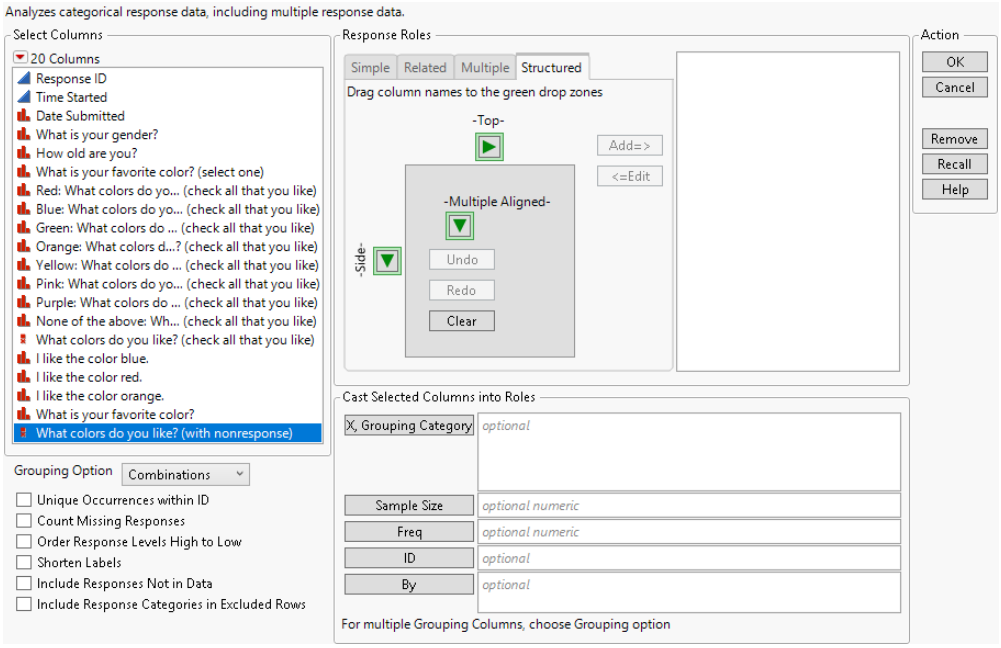
Response Frequencies Summarizes multiple responses that are stored in columns that contain frequency counts. This data format is the summarized version of the Indicator Group format. See "Response Frequencies".

Free Text Summarizes text data. The Free Text option launches a Text Explorer report inside the Categorical report window. See *Basic Analysis*.

Structured Tab

Use the Structured tab to construct custom cross tabulations.

Figure 3.15 Categorical Platform Launch Window for the Structured Tab



The Structured tab has three drop zones for assigning columns to roles. Drag column names to the green drop zone arrows. Alternatively, click the green arrows for a column list to select columns. The resulting structured table considers the innermost terms on the side of the table as responses and all other terms as grouping factors.

Drop Zones

Top Assigns one or more data table columns to the columns of the cross tabulation table. After one column has been assigned, use the down arrows to nest additional columns within already assigned columns or use the right arrows to add additional column groups.

Side Assigns one or more data table columns to the rows of the cross tabulation table. After one column has been assigned, use the right arrows to nest additional columns nested within already assigned rows or use the down arrows to add additional row groups.

Multiple Aligned Assigns two or more columns with aligned response.

Controls

Undo Click to undo column assignments.

Redo Click to redo the most recent assignment that was undone.

Clear Click to remove all assignments in the drop zones.

Add=> Click to add the constructed table to the structured table list.

<=Edit Click to make changes to the selected table from the structured table list.

Columns Roles

The categorical platform launch window has the following column roles:

X, Grouping Category (Not applicable for the Structured tab.) Assigns a column as a grouping category. The responses are summarized for each group. If more than one grouping column is used, then the tabulation is nested by default. Use the Grouping Option in the launch window to change the summarization.

Sample Size Assigns a column whose values define the number of individual units in the group to which that frequency is applicable. The sample size is used for multiple response roles with summarized data. See "Example of the Multiple Responses".

Freq Assigns a column whose values define a frequency to each row for the analysis. The frequency role is used for summarized data.

ID Assigns a column that identifies the respondent. This option is required when Multiple Response by ID is selected, and it is not used for any other response types.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Other Launch Window Options

Additional options are located in the lower left of the launch window. Alternatively, these options can be selected from the Categorical red triangle menu in the platform report window.

Grouping Option Defines how to use grouping variables in the analysis when more than one grouping column is specified.

Combinations Analyzes the response for combinations of the grouping variables. The first column in the grouping list is the outermost group in the cross tabulation table.

Each Individually Analyzes the response for each grouping variable individually.

Both Provides reports for combinations of the grouping variables as well as for each grouping variable individually.

Unique Occurrences within ID Limits the counts to unique response levels within a participant. Specify a column as the ID using the ID role. This option is applicable only when Multiple Response by ID is selected, and it is not used for any other response types.

Count Missing Responses Specifies that missing values be included as a category. Missing values can be either empty cells or a defined missing code in the Missing Value Codes column property. If a column contains only missing values, the missing values are counted regardless of this option. For multiple responses, a response is considered missing if all response categories are missing or for indicators, if all are zero.

Note: If this option is not selected, missing values are excluded from the analysis.

Order Response Levels High to Low Orders the responses from high to low. (The default ordering is low to high.) This option applies only to the response, and does not apply to grouping categories.

Tip: Use the Value Order column property to define a specific category ordering. See *Using JMP*.

Shorten Labels Shortens value labels by removing prefixes and suffixes that are common to all labels.

Note: This option applies only to value labels, and does not apply to column names.

Include Responses Not in Data Specifies that categories with no responses be included in the report. The categories with no responses must be specified in the Value Labels column property. This option applies only to responses. For grouping variables tabulations, include only categories with responses.

Include Response Categories in Excluded Rows Specifies that response categories that appear only in excluded rows be included in the report. The counts for these categories are zero.

The Categorical Report

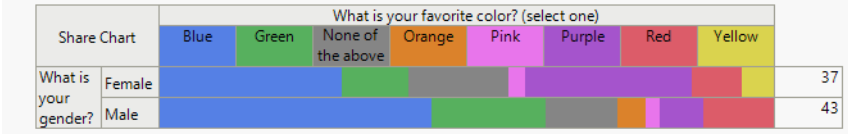
The initial Categorical report shows a cross tabulation and a share chart for each set of selected responses.

Figure 3.16 The Initial Categorical Report

Categorical

What is your favorite color? (select one) By What is your gender?

Freq / Share		What is your favorite color? (select one)								
		Blue	Green	None of the above	Orange	Pink	Purple	Red	Yellow	Total Responses
What is your gender?	Female	11 29.7%	4 10.8%	6 16.2%	0 0.0%	1 2.7%	10 27.0%	3 8.1%	2 5.4%	37
	Male	19 44.2%	8 18.6%	5 11.6%	2 4.7%	1 2.3%	3 7.0%	5 11.6%	0 0.0%	43



The upper left corner of the table lists the quantities (Freq, Share, and Rate when applicable) that are included in each cell of the table. Remove or add these quantities using the options in the Categorical red triangle menu.

- The Frequencies count (labeled Freq) is provided for each category with the total frequency (Total Responses) at the right of the table. When there are multiple responses, the summary columns at the right of the table also include the number of cases or the number of rows (Total Cases) and the number of responders (Total Cases Responding).
- The Share of Responses (Share) is determined by dividing each count (Freq) by the total number of responses.
- The Rate is the frequency of response (Freq) divided by the total number of cases (Total Cases). This quantity appears only for multiple responses and is not shown in Figure 3.16.

In Figure 3.16, the number of responses to the question What is your favorite color? are tabulated by gender. Consider the first row of the table with the results for females:

- The first cell of the table contains 11 responses. This is the count of the female respondents who selected blue as their favorite color.
- There are 37 total responses for females. Of the 37 female responses, 29.7% (11/37) selected blue as their favorite color.

Categorical Platform Options

The Categorical red triangle menu contains options that enable you to customize the report and request various statistical tests. The specific options that are available are determined by the response roles, the use of grouping categories, and the options selected in the launch window.

- "Report Options"
- "Statistical Testing Options"
- "Additional Categorical Platform Options"
- "Crosstab Table Options"

Report Options

The following options enable you to customize the appearance of the report:

Cross Tabulation Options

Frequencies Shows or hides the frequency (labeled Freq) in the Crosstab table. The frequency is the count of the responses in each category.

Share of Responses Shows or hides the share of responses (labeled Share) in the Crosstab table. The share of responses is the percent of responses in each category.

Rate Per Case (Available only for multiple responses.) Shows or hides the rate per case (labeled Rate) in the Crosstab table. The rate per case is the percent of responses in each category based on the total number of cases (regardless of whether they were a respondent).

Rate per Case Responding (Available only for multiple responses.) Shows or hides the Rate per Case Responding in the Crosstab table. The rate per case responding is the percent of responses in each category based on the cases that responded.

Chart Options

Share Chart Shows or hides a divided bar chart. The bar length is proportional to the percentage of responses for each type. The column on the right shows the number of responses in each grouping category. If no grouping category is used, the column on the right shows the total number of responses.

Frequency Chart Shows or hides a Frequency Chart. The bars reflect the frequency of responses within each group. The scale is consistent across the chart. The gray bars at the far right represent the total number of responses in each grouping category.

Transposed Freq Chart Shows or hides a transposed Frequency Chart. The bars reflect the frequency of responses within each group. The responses are the rows, and the grouping levels are the columns in chart. The totals for each grouping level are represented by gray bars in the bottom row of the chart.

Tip: You can change the colors in the share chart using the Value Colors column property. See *Using JMP*.

Crosstab Viewing Options

Crosstab Shows or hides the Crosstab table. The Crosstab table displays the response categories as column headings and the grouping levels (when used) as row labels. The upper left cell of the table shows the labels for the items in each cell of the table (Freq, Share, Rate, and Rate per Case Responding). If the report contains a transposed Crosstab table, this option also removes the transposed Crosstab table from the report.

Crosstab Transposed Shows or hides a transposed Crosstab table. The transposed Crosstab table displays the response categories as row labels and the grouping levels (when used) as column headings. The upper left cell of the table shows the labels for the items in each cell of the table (Freq, Share, Rate, and Rate per Case Responding). If the report contains a Crosstab table, this option also removes the Crosstab table from the report.

Statistical Testing Options

The statistical testing options that are available depend on the response roles and the use of grouping variables in the analysis. Options include tests for multiple responses, response homogeneity, association, relative risk, and agreement. Options depend on both the response data type and the grouping variable (X, Grouping Category) data type.

Test Multiple Response (Available only for multiple response data with one or more grouping categories.) See "Example of the Multiple Response Test". Contains tests for independence of responses across each grouping category:

Count Test, Poisson Shows or hides a test of independence of rates that uses Poisson regression. The frequency per unit is modeled by the sample categorical variable. The result is a likelihood ratio chi-square test of whether the rate of each individual response differs across grouping levels.

Homogeneity Test, Binomial Shows or hides the likelihood ratio chi-square test of independence for each individual response level. Each response category has a binomial distribution (selected or not selected).

Exclude Nonresponses Excludes nonresponses for count and homogeneity tests of multiple response categories. If a row is missing data in all response categories, it is treated as a nonresponse.

Test Response Homogeneity (Available only for response variables that do not have a multiple response modeling type and when one or more groupings variables are specified.) Shows or hides a report that contains tests for response homogeneity that depend on your grouping variable:

- When the grouping variable is not a multiple response, then one tests for independence of the response across the grouping variable levels. The likelihood ratio and Pearson chi-square tests are provided. See "Example of the Test for Response Homogeneity".
- When the grouping variable is a multiple response, then one tests for independence of the response across *each* of the grouping variable levels. A Rao-Scott Chi-square test is provided.

Cell Chisq Shows or hides *p*-values for each cell in the table for a chi-square test of independence. A small *p*-value indicates a cell with an observed value that is larger or smaller than expected under the assumption that the rows are independent of the columns. The *p*-values are colored and shaded according to whether the count is larger or smaller than expected. See "Example of the Cell Chisq Test".

Compare Each Sample (Available only for single responses with one or more grouping variables.) Shows or hides a report that contains pairwise likelihood ratio and Pearson chi-square tests for independence of responses across levels of a grouping variable. See "Example of Compare Each Sample with Comparison Letters".

Compare Each Cell (Available only for single and multiple responses with one or more grouping variables.) Shows or hides pairwise likelihood ratio chi-square, Pearson chi-square, and Fisher's exact tests for independence of each level of the response versus all other levels combined across levels of a grouping variable. See "Example of Compare Each Cell with Comparison Letters".

Relative Risk (Available when the grouping variable has two levels and either the response has two levels or is a multiple response and the Unique occurrences within ID option has been selected.) Shows or hides the relative risks for a two-level grouping variable for each level of the response. See "Example of Conditional Association and Relative Risk".

Conditional Association (Available only when the Unique occurrence within ID option has been selected.) Shows or hides the conditional probability of one response level given a second response level. See "Example of Conditional Association and Relative Risk".

Agreement Statistic (Available only for Rater Agreement responses.) Shows or hides the Kappa coefficient of agreement and the Bowker test of symmetry. See "Example of Rater Agreement".

Transition Report (Available only for Repeated Measures responses.) Shows or hides transition counts and rates matrices for changes in responses across time. See "Example of Repeated Measures".

Test Options Options available in this menu depend on your selected analysis.

ChiSquare Test Choices Specifies which chi-square tests of homogeneity are calculated for single responses. You can choose between Both LR and Pearson, LR Only, or Pearson Only, where LR refers to likelihood ratio.

Show Warnings Shows warnings for small sample sizes in chi-square tests.

Order by Significance Reorders the reports so that the most significant reports are at the top.

Hide Nonsignificant Suppresses reports that are non-significant.

FDR Adjusted PValues Shows or hides false discovery rate *p*-values based on the method of Benjamini and Hochberg (1995).

Additional Categorical Platform Options

The following options enable you to add summary statistics to the report, save reports, and set report formats.

Summary Statistics Options

Total Responses Shows or hides the sum of the frequency counts for the response in crosstab tables and share charts. When a grouping variable is specified, the total is across each grouping category.

Response Levels Shows or hides the categories for the response column in crosstab tables and share charts.

Show Supercategories (Available only when one or more supercategories is defined.) Shows or hides columns for supercategories in the crosstab table and the Frequency Chart. For more information about supercategories, see "Supercategories".

Tip: This option shows or hides the Supercategories. To hide the individual categories within the supercategory, use the Hide option in the Supercategories column property. Alternatively, use the Response Levels option to hide all response levels so that only Supercategories remain unhidden.

Total Cases (Available only for multiple response columns.) Shows or hides a column in the crosstab table that contains the number of cases (participants) in each group.

Total Cases Responding (Available only for multiple response columns.) Shows or hides a column in the crosstab table that contains the number of cases (participants) who responded at least once. Participants who did not respond at all are not included. The total cases responding is less than or equal to the total cases.

Mean Score Shows or hides a column in the crosstab table and share chart that contains the overall mean of the response or the mean for each grouping category. The mean is calculated based on a numerical value assigned to each response category.

- For numeric categories, the numeric value is the actual value.
- For non-numeric categories the value is the value assigned to the categories by the Value Scores column property.
- For categories without value scores, the value is based on a default assignment of 1 to the number of categories.

See "Example of Mean Score with Comparison Letters".

Mean Score Comparisons Shows or hides the Compare Means column in the crosstab table. This column compares the mean scores across grouping categories using the unpooled Satterthwaite *t* test for pairwise comparisons. See the TTEST Procedure chapter in SAS Institute Inc. (2020). The results of the comparison are shown using letters. For more information about comparison letters, see "Comparison Letters". For more information about specifying comparison groups, see "Example of User-Specified Comparison with Comparison Letters".

Std Dev Score Shows or hides a column in the crosstab table that contains the overall standard deviation of the response or the standard deviation of each grouping category.

Order by Mean Score (Appears only when more than one response is specified and there are no grouping variables in the analyses.) Orders the response reports by the mean score.

Save Options

Save Tables Contains options to save specific portions of the reports to a new data table. Each option creates an individual data table for each report. The options available in this menu depend on your selected analysis. The saved tables all include a Source script.

Note: Supercategories are not included in the new tables.

Save Frequencies Saves the frequency counts from the crosstab table to a new data table.

Save Share of Responses Saves the share of responses from the crosstab table to a new data table.

Save Contingency Table Saves the complete crosstab table to a new data table.

Save Rate Per Case Saves the rates per case from the crosstab table to a new data table.

Save Transposed Frequencies Saves the transposed frequency counts from the crosstab table to a new data table.

Save Transposed Share of Responses Saves the transposed share of responses from the crosstab to a new data table.

Save Transposed Rate Per Case Saves the transposed rates per case from the crosstab table to a new data table.

Save Test Rates Saves the results of the Test Multiple Response option to a new data table.

Save Test Homogeneity Saves the results of the Test Response Homogeneity option to a new data table.

Save Mean Scores Saves the mean scores for each sample group to a new data table.

Save tTests and pValues Save *t* tests and *p*-values from the Mean Score Comparisons report to a new data table.

Save Excel File (Available only on Windows.) Creates a Microsoft Excel spreadsheet with the structure of the crosstab table. This option maps all of the tables to one sheet, with the response categories as rows, the sample levels as columns, and sharing the headings for sample levels across multiple tables. When there are multiple elements in each table cell, you have the option to make them multiple or single cells in Microsoft Excel.

Report Format Options

Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Contents Summary Shows or hides a Contents Summary report at the top of the Categorical report. The Contents Summary report contains all of the tests and mean scores in a summary that has links to the associated report.

Show Columns Used in Report (Available only with SPSS or SAS names.) Shows or hides Columns Used in Report information. This option affects only columns that have an SPSS or SAS Name or SPSS or SAS Label column property.

Tip: When you import survey data from SAS or SPSS, the Name and Label column properties are automatically added to your JMP table. You can add a SAS or SPSS Name or Label column property using the Other column property. For example, if you use the SAS or SPSS Name column property to store a survey question, the column name can be a short name.

Format Elements Enables you to specify formats for Frequencies, Shares, Rates, and Means.

Arrange in Rows Enables you to arrange multiple reports across the window. Enter the number of reports that you want to view across the window.

Set Preferences Enables you to set preferences for future launches of the Categorical platform in the current JMP session and in future JMP sessions. See "Set Preferences".

Category Options Contains options (Grouping Option, Count Missing Responses, Order Response Levels High to Low, Shorten Labels, and Include Responses Not in Data) that are also available on the launch window. If these options are selected here, the platform updates with the new setting. For more information about the Category Options, see "Other Launch Window Options".

Force Crosstab Shading Forces shading on Crosstab reports even if the preference is set to no shading. If this option is not selected, the Crosstab reports are shaded according to the current setting of the Shade Alternate Table Rows preference.

Relaunch Dialog Enables you to return to the launch window and edit the specifications for an analysis.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Crosstab Table Options

Show Letters Shows or hides the column letter IDs in the Crosstab table. These letters are used in many of the tests of homogeneity and are displayed automatically for those tests.

Specify Comparison Groups Enables you to specify specific comparison groups for tests of homogeneity. Use group comparison letters separated by a slash to represent each group. Separate multiple groups by commas. For example, to test A with E, B with D, and C with F, specify the groups as "A/E, B/D, C/F". A Compare Each Cell report is provided for the

defined comparison groups. See "Example of User-Specified Comparison with Comparison Letters".

Remove Removes the report from the report window.

Caution: The Remove option cannot be undone.

Comparison Letters

The Compare Each Cell, Compare Each Sample, and Mean Score Comparisons options use comparison letters to identify sample levels. For more than 26 levels, numbers are appended to the letters. The letters are shown to the right of the sample level headings of the crosstab table when a comparison option is turned on. These letters are used to identify levels in the Compare column.

Figure 3.17 Crosstab Table with Comparison Letters

country By type

			Freq / Share	country: American	country: European	country: Japanese	Total Responses	Compare
type	Family	A		74 47.7%	15 9.7%	66 42.6%	155	B
	Sporty	B		23 23.0%	21 21.0%	56 56.0%	100	c
	Work	C		18 37.5%	4 8.3%	26 54.2%	48	*

Default Comparison Groups: A/B/C

Shows letter of the category it is significantly different from at the higher share level

* Base count warning	100	Uppercase Alpha Level	0.05
** Base count minimum	30	Lowercase Alpha Level	0.1

A letter in the Compare column indicates a difference between two levels. The row containing the letter is one level and the letter in the Compare column indicates the second level. When two sample levels are significantly different, the letter of the sample level with a smaller share of responses is placed into the comparison cell of the other level. An uppercase letter indicates a stronger difference between levels than a lowercase letter. The default alpha level (significance level) for an uppercase letter is 0.05 and 0.10 for a lowercase letter. For example, in Figure 3.17, notice the following:

- The first row of the table for Family cars has a B in the Compare column. The letter B is associated with Sporty cars. This indicates that there is a difference in the country of origin for Sporty and Family cars at the 0.05 significance level. The B is in the row for Family cars

because the total responses for Sporty cars (100) is less than the total responses for Family cars (155).

- The c in the Compare column of the Sporty row indicates a significant difference at the 0.10 level between the country of origin when comparing Sporty to Work cars. The c is in the row for Sporty cars because the total responses for Sporty cars (100) is greater than the total responses for Work cars (48).

Warnings for small counts are indicated by asterisks in the comparison cells. One asterisk indicates that the level has fewer than 100 responses. Two asterisks indicate fewer than 30 responses. In Figure 3.17, notice that the row for Work has an asterisk in the Compare column. This warning that the count of responses is small appears because the total responses for Work cars (48) is less than100.

You can change the alpha levels and thresholds for the warning counts in the Categorical platform preferences. For more information about changing preferences, see "Set Preferences".

Tip: If you want only one set of comparison letters in your report, set the **Lowercase Alpha Level** to 0 in the preferences.

The following examples illustrate the use of comparison letters:

- "Example of Compare Each Sample with Comparison Letters"
- "Example of Compare Each Cell with Comparison Letters"
- "Example of Mean Score with Comparison Letters"

Supercategories

The term *supercategories* refers to the aggregation of response categories. For example, when using a five-point rating scale, you might want to know the percent of responses from the top two ratings (top two boxes). Use the Supercategories column property to define groups of responses.

When you add a Supercategories column property to a response column, no additional columns are added to your data table. Instead, the Supercategories column property adds an additional category column to crosstab tables and Frequency Charts in the Categorical platform report. You can create multiple supercategories for a single response column. Share Charts do not show supercategories, and supercategories are not applied to grouping columns.

To create a supercategory:

1. Select a column in your data table that contains categories that you would like to aggregate.
2. Select **Cols > Column Info**.
3. Click **Column Properties** and select **Supercategories**.
4. (Optional) To change the default name of the supercategory, enter a Supercategory Name.
5. Select one or more categories from the Column's Categories list.
6. Click **Add**.
7. (Optional) Select the supercategory and click the Supercategories red triangle for additional options.

Supercategories Options

The following options are available in the Supercategories red triangle menu in the Column Properties window:

Hide Hides categories within a supercategory in the crosstab table and frequency chart.

> **Tip:** If you want the flexibility to show or hide the individual categories in your reports, then do not use the Hide option. Use the Response Level option in the Categorical red triangle menu.

Net (Available only for a multiple response column.) Prevents individual respondents from being counted twice when they appear in more than one supercategory.

Add Mean Includes mean statistics in the report.

Add Std Dev Includes standard deviation statistics in the report.

Add All Includes total responses in the report. By default, the Total Responses column is always included.

Note: Supercategories are supported for all response effects except Repeated Measures and Rater Agreement.

Additional Examples of the Categorical Platform

- "Example of the Test for Response Homogeneity"
- "Example of the Multiple Response Test"
- "Example of Supercategories"
- "Example of the Cell Chisq Test"
- "Example of Compare Each Sample with Comparison Letters"
- "Example of Compare Each Cell with Comparison Letters"
- "Example of User-Specified Comparison with Comparison Letters"
- "Example of Aligned Responses"
- "Example of Conditional Association and Relative Risk"
- "Example of Rater Agreement"
- "Example of Repeated Measures"
- "Example of the Multiple Responses"
- "Example of Mean Score with Comparison Letters"
- "Example of a Structured Report"
- "Example of a Multiple Response with Nonresponse"

Example of the Test for Response Homogeneity

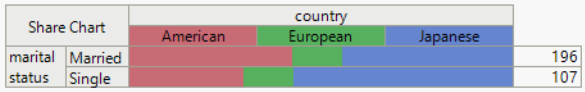
This example uses the Car Poll.jmp sample data table, which contains data collected from a survey about car ownership. The data include demographics about the individuals polled and information about their car. You want to explore the relationship between marital status and the origin of the car. You also want to test for the homogeneity of the responses. That is, you want to test to see whether the distribution of the origin of cars is the same for married and single respondents.

1. Select **Help > Sample Data Library** and open Car Poll.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select country and click **Responses** on the Simple tab.
4. Select marital status and click **X, Grouping Category**.
5. Click **OK**.
6. Click the Categorical red triangle and select **Test Response Homogeneity**.

Figure 3.18 Test Response Homogeneity

country By marital status

Freq / Share		country: American	country: European	country: Japanese	Total Responses
marital status	Married	83 42.3%	26 13.3%	87 44.4%	196
	Single	32 29.9%	14 13.1%	61 57.0%	107



Test Response Homogeneity

Response Dimension Label	Sample Dimension Label	LR Chisq	LR PValue	Pearson Chisq	Pearson PValue
country	marital status	5.14007	0.0765	5.08146	0.0788

The Share Chart indicates that the married group is evenly split between ownership of American and Japanese cars. In the single group, Japanese cars are the most frequently owned.

The test for response homogeneity provides results from two versions of the test. The Pearson Test and the Likelihood Ratio Test both have chi-square test statistics and associated *p*-values. The test for response homogeneity has a *p*-value of about 0.08 for either method.

Example of the Multiple Response Test

This example uses the Consumer Preferences.jmp sample data table. This table contains data from a survey about people's attitudes and opinions, as well as questions concerning oral hygiene. You can use the Test Multiple Response option to test if the response rates for each brushing time (Brush Delimited) are the same across groups (Brush). The groups are defined by the frequency that responders brush their teeth.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Scroll to the right until you see the Brush Delimited column.

Figure 3.19 Consumer Preferences Data Table

	Brush After Waking Up	Brush After Meal	Brush Before Sleep	Brush Another Time	Brush Other	Brush Delimited
1	1	0	0	0		Wake,
2	0	1	0	0		After Meal,
3	1	0	1	0		Wake,Before Sleep,
4	0	1	0	0		After Meal,
5	1	0	1	0		Wake,Before Sleep,
6	1	1	1	0		Wake,After Meal,Before Sleep,
7	1	1	1	0		Wake,After Meal,Before Sleep,

Note that the Brush Delimited column contains the responses to a multiple response question. Each response is separated by a comma. The four columns preceding Brush Delimited contain the same information in a different data format. Each column (Brush after Waking Up, Brush After Meal, Brush Before Sleep, and Brush Another Time) contains one response. If the response was selected the column value is a 1, and otherwise it is 0.

3. Select **Analyze > Consumer Research > Categorical**.
4. Select the **Multiple** tab.
5. Select Brush Delimited and click **Multiple Delimited** on the Multiple tab.

Tip: Alternatively, you can select Brush after Waking Up, Brush After Meal, Brush Before Sleep, and Brush Another Time and click **Indicator Group** on the Multiple tab

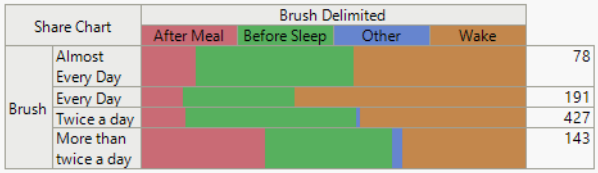
---

6. Select Brush and click **X, Grouping Category**.
7. Click **OK**.
8. Click the Categorical red triangle and select **Test Multiple Response > Count Test, Poisson**.

Figure 3.20 Test Multiple Response, Poisson

Multiple Delimited(Brush Delimited) By Brush

	Freq / Share / Rate	Brush Delimited: After Meal	Before Sleep	Other	Wake	Total Responses	Total Cases	Total Cases Responding
Brush	Almost Every Day	11	32	0	35	78	48	48
		14.1%	41.0%	0.0%	44.9%			
		22.9%	66.7%	0.0%	72.9%			
	Every Day	21	55	0	115	191	142	142
		11.0%	28.8%	0.0%	60.2%			
		14.8%	38.7%	0.0%	81.0%			
	Twice a day	49	189	4	185	427	209	209
		11.5%	44.3%	0.9%	43.3%			
		23.4%	90.4%	1.9%	88.5%			
	More than twice a day	46	47	4	46	143	49	49
		32.2%	32.9%	2.8%	32.2%			
		93.9%	95.9%	8.2%	93.9%			



Test Each Response, Poisson

Brush, Brush Delimited

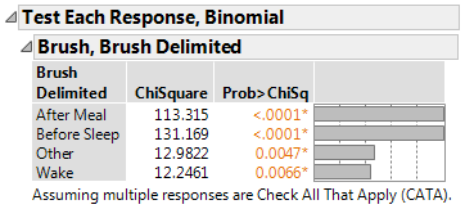
Brush Delimited	ChiSquare	Prob>ChiSq
After Meal	59.5436	<.0001*
Before Sleep	39.1097	<.0001*
Other	12.7131	0.0053*
Wake	1.8793	0.5978

Chi-squared tests use Poisson rates.

The *p*-values show that the response rates for After Meal, Before Sleep, and Other are significantly different across brushing groups.Wake is not significantly different across brushing groups. The bar graph to the right of the Prob>ChiSq column plots the *p*-values on a -Log10(p) scale. From the crosstab table, you can see that most people brush their teeth when they wake up regardless of how frequently they brush their teeth.

9. Click the Categorical red triangle and select **Test Multiple Response > Homogeneity Test, Binomial**.

Figure 3.21 Test Multiple Response, Binomial



The Homogeneity Test, Binomial option always produces a larger test statistic (and therefore a smaller *p*-value) than the Count Test, Poisson option. The binomial distribution compares not only the rate at which the response occurred (the number of people who reported that they brush upon waking) but also the rate at which the response did not occur (the number of people who did not report that they brush upon waking).

In this example, the proportion of responders for each response (After Meal, Before Sleep, Wake, and Other) differs across the groups. The *p*-value for each response is less than 0.05.

Tip: JMP detects a multiple response column by the Multiple Response modeling type or the Multiple Response column property.

Example of Supercategories

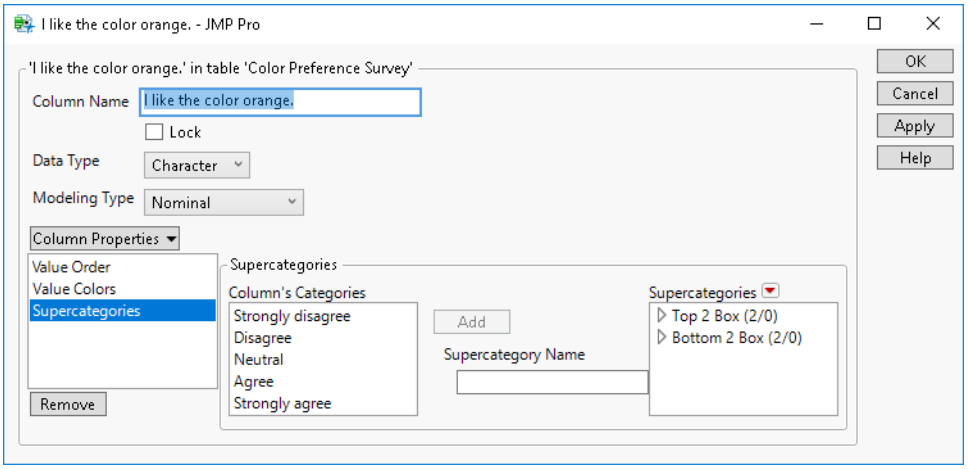
This example uses the Color Preference Survey.jmp sample data table, which contains survey data on people's color preferences. This example illustrates the use of supercategories.

1. Select **Help > Sample Data Library** and open Color Preference Survey.jmp.
2. Select the column I like the color orange. Right-click the column heading and select **Column Properties > Supercategories**.
3. Under Supercategories, select Agree and Strongly agree for the column's categories.
4. Under Supercategory Name, enter Top 2 Box.
5. Click **Add**.

Tip: Click on the triangle to the left of the name to show the categories included in the supercategory.

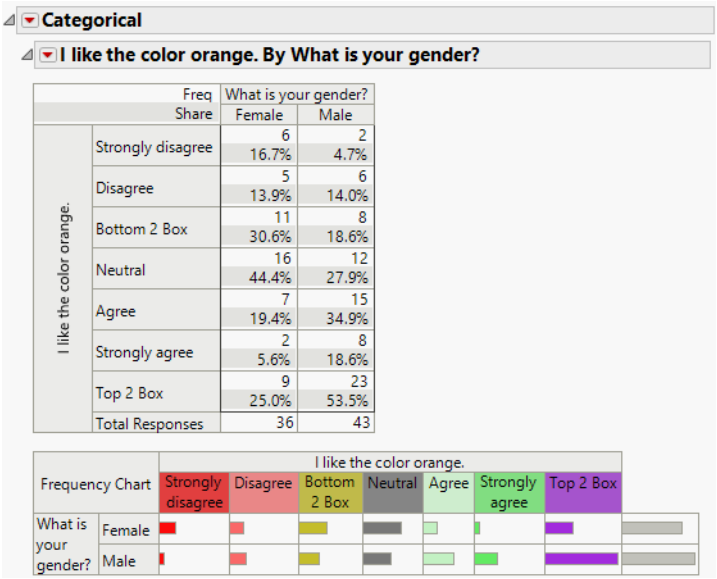
6. Under Supercategories, select Disagree and Strongly disagree for the column's categories.
7. Under Supercategory Name, enter Bottom 2 Box.
8. Click **Add**.

Figure 3.22 Supercategory Column Property



9. Click **OK**.
10. Select **Analyze > Consumer Research > Categorical**.
11. Select the **Structured** tab.
12. Click the Side green triangle and select I like the color orange.
13. Click the Top green triangle and select What is your gender?
14. Click **Add=>** and then click **OK**.
15. Click the Categorical red triangle and select **Transposed Freq Chart**.

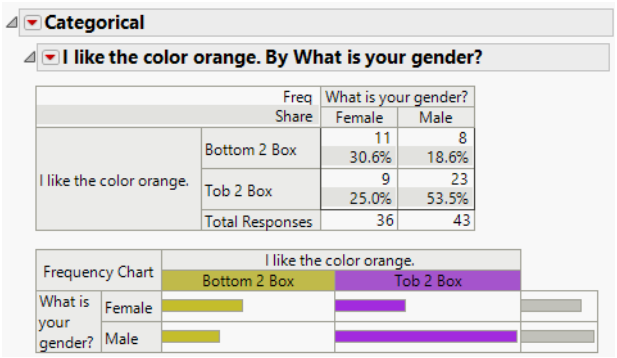
Figure 3.23 Structured Categorical Report with Supercategories



The crosstab table includes two additional rows, one for each supercategory. The supercategories are also included in the Frequency Chart. Note that the frequency counts in the Top 2 Box row are the sums of the counts in the Agree and Strongly agree categories. The frequency counts in the Bottom 2 Box row are the sums of the counts in the Disagree and Strongly disagree categories.

16. Click the Categorical red triangle and deselect **Response Levels**.

Figure 3.24 Structured Categorical Report with Supercategories and No Response Levels



By removing the response levels, your output now contains only the supercategories. Note that the totals are for all levels. In this example, the neutral responses are not included in the supercategories.

Example of the Cell Chisq Test

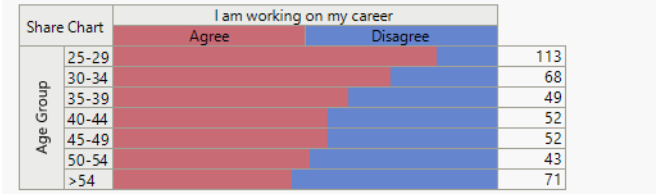
This example uses the Consumer Preferences.jmp sample data table. This table contains survey data about people's attitudes and opinions, as well as questions concerning oral hygiene. You explore the distribution of responses to the statement "I am working on my career" across age groups.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select I am working on my career and click **Responses** on the Simple tab.
4. Select Age Group and click **X, Grouping Category**.
5. Click **OK**.
6. Click the Categorical red triangle and select **Crosstab Transposed**.
7. Click the Categorical red triangle and select **Cell Chisq**.

Figure 3.25 Cell Chisq

I am working on my career By Age Group

Freq / Share / Cell Chisq PVal		Age Group						
		25-29	30-34	35-39	40-44	45-49	50-54	>54
I am working on my career	Agree	95	49	30	29	29	22	33
		84.1%	72.1%	61.2%	55.8%	55.8%	51.2%	46.5%
		0.00788	0.41003	0.80398	0.45495	0.45495	0.29058	0.06415
	Disagree	18	19	19	23	23	21	38
		15.9%	27.9%	38.8%	44.2%	44.2%	48.8%	53.5%
		0.00039	0.27136	0.74035	0.31848	0.31848	0.15823	0.01345
	Total Responses	113	68	49	52	52	43	71



Small *p*-values indicate that there is a significant difference between the observed cell count and the expected cell count. The *p*-values are colored by significance level from dark red for cells with significantly higher counts than expected to dark blue for cells with significantly lower counts than expected. The expected cell count is based on the observed row and column totals. The expected cell count is calculated as the row total times the column total divided by the overall count.

For example, the expected number of responses under the null hypothesis that the rates are equal in the 25 - 29 group who agree is (287*113)/448 = 72.4; the observed value was 95. This observed value, with a *p*-value of 0.00788, is significantly larger than the expected value. The number of responses in the 25 - 29 group who agree with "I am working on my career" is higher than expected if the response to this question was independent of age.

Example of Compare Each Sample with Comparison Letters

This example uses the Consumer Preferences.jmp sample data table. This table contains survey data about people's attitudes and opinions, as well as questions concerning oral hygiene. You explore the distribution of responses to the statement "I am working on my career" between each age group.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select I am working on my career and click **Responses** on the Simple tab.
4. Select Age Group and click **X, Grouping Category**.
5. Click **OK**.
6. Click the Categorical red triangle and select **Compare Each Sample**.

Figure 3.26 Compare Each Sample

I am working on my career By Age Group

			Freq / Share	I am working on my career: Agree	Disagree	Total Responses	Compare
Age Group	25-29	A		95 84.1%	18 15.9%	113	b,C,D,E,F,G
	30-34	B		49 72.1%	19 27.9%	68	d,e,F*
	35-39	C		30 61.2%	19 38.8%	49	*
	40-44	D		29 55.8%	23 44.2%	52	*
	45-49	E		29 55.8%	23 44.2%	52	*
	50-54	F		22 51.2%	21 48.8%	43	*
	>54	G		33 46.5%	38 53.5%	71	B*

Default Comparison Groups: A/B/C/D/E/F/G

Shows letter of the category it is significantly different from at the higher share level
* Base count warning 100 Uppercase Alpha Level 0.05
** Base count minimum 30 Lowercase Alpha Level 0.1

Compare Each Sample

Letter comparisons use Pearson Chisq
Age Group, I am working on my career

LR PValues
LR Chi-square p-value on pairs

	A	B	C	D	E	F	G
A	1.0000	0.0552	0.0020	0.0001	0.0001	<.0001	<.0001
B	0.0552	1.0000	0.2183	0.0641	0.0641	0.0261	0.0020
C	0.0020	0.2183	1.0000	0.5781	0.5781	0.3312	0.1108
D	0.0001	0.0641	0.5781	1.0000	1.0000	0.6540	0.3083
E	0.0001	0.0641	0.5781	1.0000	1.0000	0.6540	0.3083
F	<.0001	0.0261	0.3312	0.6540	0.6540	1.0000	0.6276
G	<.0001	0.0020	0.1108	0.3083	0.3083	0.6276	1.0000

Pearson PValues
Pearson Chi-square p-value on pairs

	A	B	C	D	E	F	G
A	1.0000	0.0523	0.0015	<.0001	<.0001	<.0001	<.0001
B	0.0523	1.0000	0.2170	0.0638	0.0638	0.0255	0.0022
C	0.0015	0.2170	1.0000	0.5783	0.5783	0.3314	0.1119
D	<.0001	0.0638	0.5783	1.0000	1.0000	0.6540	0.3087
E	<.0001	0.0638	0.5783	1.0000	1.0000	0.6540	0.3087
F	<.0001	0.0255	0.3314	0.6540	0.6540	1.0000	0.6276
G	<.0001	0.0022	0.1119	0.3087	0.3087	0.6276	1.0000

The crosstab table summarizes the statement "I am working on my career" across age groups. The cells of the table contain the frequency (count) and share (percent) of those who agree or disagree with the statement for each age group. In addition, the crosstab includes comparison letters. Each group is labeled with a letter in the Compare column, which is to the right of the group label. The letters in the Compare column enable you to interpret the outcomes of the statistical test of independence among groups.

The Compare Each Sample report provides *p*-values from the pairwise Pearson and Chi-square likelihood ratio chi-square tests. The *p*-values are reported in symmetric matrices labeled by the comparison letters.

For this example we make the following observations:

- The comparison column for the 25 - 29 group contains all letters b through g. Thus, the 25 - 29 group has significantly different response rates to the statement "I am working on my career" as compared to all other groups. Because the letter b is lowercase, the difference between the 25 - 29 group and the 30 - 34 group is significant at the 0.10 level. All other letters are uppercase, which indicate differences that are significant at the 0.05 level.
- The >54 group, denoted by letter G, is significantly different from the 30 - 34 group, denoted by B. The letter for the comparison is in the cell for group G because group G has a higher number of responders (71 versus 68) than group B.
- The single asterisks in the comparison cells are small sample size warnings. A single asterisk indicates that a group has more than 30 but fewer than 100 responses.
- A double asterisk, not observed in this example, would indicate a group size of fewer than 30.

Example of Compare Each Cell with Comparison Letters

This example uses the Consumer Preferences.jmp sample data table. This table contains survey data about people's attitudes and opinions, as well as questions concerning oral hygiene. You explore the distribution of the responses about job satisfaction between employee tenure groups.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select Job Satisfaction and click **Responses** on the Simple tab.
4. Select Employee Tenure and click **X, Grouping Category**.
5. Click **OK**.
6. Click the Categorical red triangle and select **Compare Each Cell**.
7. Click the **Compare Each Cell - Details** gray disclosure icon.

Figure 3.27 Compare Each Cell

Job Satisfaction By Employee Tenure

			Job Satisfaction			
Freq / Share / Comparisons			Not at all satisfied	Somewhat satisfied	Extremely satisfied	Total Responses
Employee Tenure	less than 5 years	A	11 6.9% 	97 60.6% d	52 32.5% 	160
	5 to 10 years	B	13 9.4% 	72 51.8% 	54 38.8% 	139
	10 to 20 years	C	4 4.7% *	43 50.0% *	39 45.3% a*	86
	more than 20 years	D	4 6.3% *	30 47.6% *	29 46.0% a*	63

Default Comparison Groups: A/B/C/D

Shows letter of the category it is significantly different from at the higher share level
* Base count warning 100 Uppercase Alpha Level 0.05
** Base count minimum 30 Lowercase Alpha Level 0.1

Share Chart		Job Satisfaction			
		Not at all satisfied	Somewhat satisfied	Extremely satisfied	
Employee Tenure	less than 5 years				160
	5 to 10 years				139
	10 to 20 years				86
	more than 20 years				63

Compare Each Cell - Details

Letter comparisons use Fisher's Exact Test
Employee Tenure, Job Satisfaction

LR Pairs

		AB	BB	AC	BC	CC	AD	BD	CD	DD
Not at all satisfied	1.0000	0.4323	1.0000	0.4781	0.1806	1.0000	0.8872	0.4658	0.6515	1.0000
Somewhat satisfied	1.0000	0.1246	1.0000	0.1092	0.7931	1.0000	0.0783	0.5820	0.7739	1.0000
Extremely satisfied	1.0000	0.2526	1.0000	0.0476	0.3366	1.0000	0.0607	0.3377	0.9341	1.0000

Pearson Pairs

		AB	BB	AC	BC	CC	AD	BD	CD	DD
Not at all satisfied	1.0000	0.4316	1.0000	0.4870	0.1948	1.0000	0.8878	0.4763	0.6496	1.0000
Somewhat satisfied	1.0000	0.1246	1.0000	0.1086	0.7931	1.0000	0.0774	0.5820	0.7740	1.0000
Extremely satisfied	1.0000	0.2524	1.0000	0.0465	0.3360	1.0000	0.0585	0.3364	0.9341	1.0000

Fisher Exact Pairs

		AB	BB	AC	BC	CC	AD	BD	CD	DD
Not at all satisfied	1.0000	0.5235	1.0000	0.5853	0.2990	1.0000	1.0000	0.5912	0.7221	1.0000
Somewhat satisfied	1.0000	0.1302	1.0000	0.1373	0.8909	1.0000	0.0983	0.6493	0.8685	1.0000
Extremely satisfied	1.0000	0.2764	1.0000	0.0531	0.4034	1.0000	0.0649	0.3574	1.0000	1.0000

The *p*-values for pairwise likelihood ratio chi-square, Pearson chi-square, and Fisher's exact tests for independence are provided in tables. The tables are labeled by comparison letters. The comparison letters are shown in the crosstab table to the right of the group labels. Response rates that differ between groups are indicated with a comparison letter in the crosstab table cells.

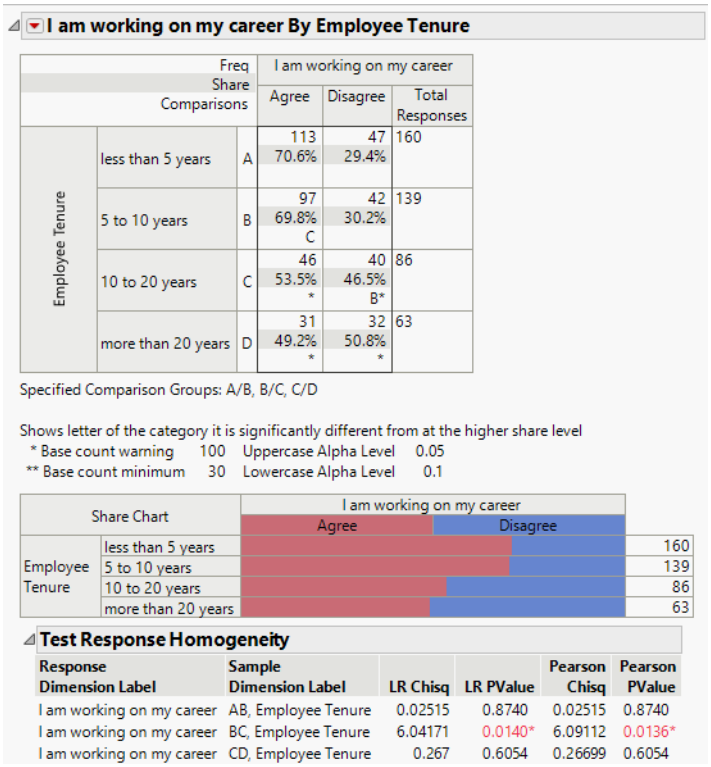
Employees with fewer than 5 years of tenure are somewhat satisfied at a greater rate than those with 20 years of tenure. This finding is noted by the letter d in the Somewhat satisfied cell in the first row of the Crosstab table. In addition, the employees with fewer than 5 years of tenure are Extremely satisfied at a lower rate than the group with 20 years of tenure. This finding is noted by the letter a in the Extremely satisfied cell of the last row of the crosstab table. The letters are placed in the cell with the highest share of responses.

Example of User-Specified Comparison with Comparison Letters

This example uses the Consumer Preferences.jmp sample data table. This table contains survey data about people's attitudes and opinions. You define specific comparison groups across which to compare the responses to the statement "I am working on my career".

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select I am working on my career and click **Responses** on the Simple tab.
4. Select Employee Tenure and click **X, Grouping Category**.
5. Click **OK**.
6. Click the red triangle next to I am working on my career By Employee Tenure and select **Show Letters**.
7. Click the red triangle next to I am working on my career By Employee Tenure and select **Specify Comparison Groups**.
8. Enter A/B, B/C, C/D and click **OK**.
9. Click the Categorical red triangle and select **Test Response Homogeneity**.

Figure 3.28 Specify Comparison Example



I am working on my career By Employee Tenure

			Freq / Share / Comparisons	I am working on my career: Agree	I am working on my career: Disagree	Total Responses
Employee Tenure	less than 5 years	A		113 70.6%	47 29.4%	160
	5 to 10 years	B		97 69.8% C	42 30.2%	139
	10 to 20 years	C		46 53.5% *	40 46.5% B*	86
	more than 20 years	D		31 49.2% *	32 50.8% *	63

Specified Comparison Groups: A/B, B/C, C/D

Shows letter of the category it is significantly different from at the higher share level
* Base count warning 100 Uppercase Alpha Level 0.05
** Base count minimum 30 Lowercase Alpha Level 0.1

Share Chart		I am working on my career: Agree / Disagree	
Employee Tenure	less than 5 years		160
	5 to 10 years		139
	10 to 20 years		86
	more than 20 years		63

Test Response Homogeneity

Response Dimension Label	Sample Dimension Label	LR Chisq	LR PValue	Pearson Chisq	Pearson PValue
I am working on my career	AB, Employee Tenure	0.02515	0.8740	0.02515	0.8740
I am working on my career	BC, Employee Tenure	6.04171	0.0140*	6.09112	0.0136*
I am working on my career	CD, Employee Tenure	0.267	0.6054	0.26699	0.6054

The test of response homogeneity compares Group A to Group B, Group B to Group C, and Group C to Group D. Group B (5 to 10 years) respondents agree with the statement "I am working on my career" more often than those in Group C (10 to 20 years). The Pearson *p*-value for this difference in agreement rates is 0.0136.

Example of Aligned Responses

This example uses the Consumer Preferences.jmp sample data table. This table contains survey data about attitudes and opinions.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Scroll to see the column I am working on my career.

Figure 3.29 Consumer Preferences Data Table (Partial Table)

	I am working on my career	I want to see the world	My home needs some major ...	I have vast interests outside of work	I want to get my debt under control	I come from a large family
1	Agree	Disagree	Disagree	Agree	Disagree	Agree
2	Agree	Agree	Agree	Agree	Disagree	Agree
3	Disagree	Agree	Agree	Agree	Agree	Agree
4	Agree	Agree	Agree	Agree	Agree	Agree
5	Disagree	Agree	Disagree	Agree	Agree	Disagree

Note that there are six columns, all with the same responses: Agree and Disagree. The responses for these six columns are aligned. For this example, we analyze the columns I am working on my career and I want to see the world. First, use standardize attributes to set the value colors, value order, and modeling type for these two columns:

3. Select the column headings I am working on my career and I want to see the world. Right-click and select **Standardize Attributes**.
4. Select **Column Properties > Value Colors.** Right-click the Agree color oval and set it to green and set Disagree to red.
5. Select **Column Properties > Value Order.** Click **Reverse**.
6. Select **Attributes > Modeling Type** and **Modeling Type > Ordinal**.
7. Click **OK**.

   Next, use the Categorical platform to analyze these two columns.
8. Select **Analyze > Consumer Research > Categorical**.
9. Select I am working on my career and I want to see the world.
10. Select the **Related** tab, and then click **Aligned Responses** on the Related tab.

> **Tip:** To do the same thing using the Structured tab, drag the two columns into the Multiple Aligned drop zone (green arrow).

11. Click **OK**.

Figure 3.30 Aligned Response Report

Aligned Responses(I am working on my career, I want to see the world)

		Freq / Share	Response: Disagree	Response: Agree	Total Responses
Response	I am working on my career		161	287	448
			35.9%	64.1%	
Response	I want to see the world		78	370	448
			17.4%	82.6%	

		Share Chart: Response Disagree	Response Agree	
Response	I am working on my career			448
Response	I want to see the world			448

Notice that the Share Chart is a directional bar chart. The type of bar chart you see depends on the modeling type of the columns:

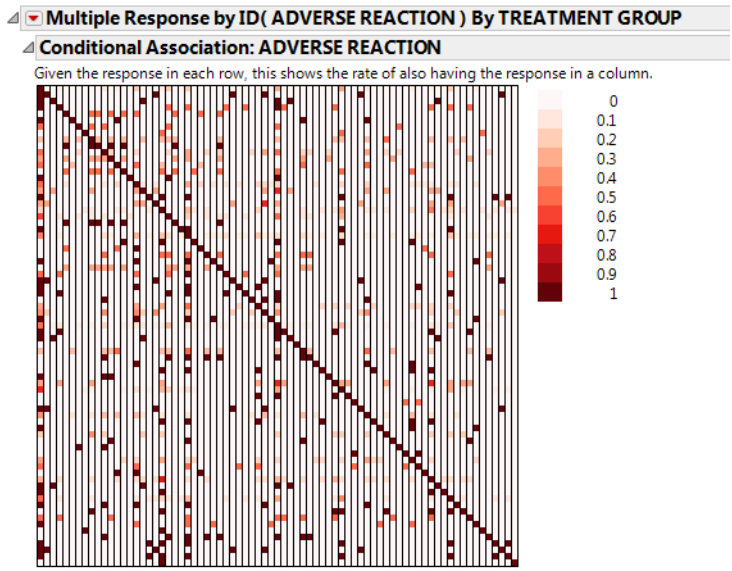
- Ordinal columns result in a directional bar chart. The columns in this example are both ordinal.
- Nominal columns result in a stacked bar chart.

Example of Conditional Association and Relative Risk

This example uses the AdverseR.jmp sample data table, which contains adverse reactions from a clinical trial. Use this data to explore the conditional association of adverse events and then the relative risk of the events in the treatment group as compared to the control.

1. Select **Help > Sample Data Library** and open AdverseR.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Multiple tab.
4. Select ADVERSE REACTION and click **Multiple Response by ID** on the Multiple tab.
5. Select TREATMENT GROUP and click **X, Grouping Category**.
6. Select PATIENT ID and click **ID**.
7. Under the other launch window options, select **Unique Occurrences within ID** and click **OK**.
8. Click the Categorical red triangle and select **Conditional Association**.

Figure 3.31 Conditional Association Report (Partial Report)



Conditional Assoc

	ABDOMINAL PAIN	ABNORMAL VISION	ALOPECIA	AMBLYOPIA	ANEMIA
ABDOMINAL PAIN	1.0000	0.0323	0.0323	0.0645	0.0000
ABNORMAL VISION	1.0000	1.0000	0.0000	0.0000	0.0000
ALOPECIA	1.0000	0.0000	1.0000	0.0000	0.0000
AMBLYOPIA	1.0000	0.0000	0.0000	1.0000	0.0000
ANEMIA	0.0000	0.0000	0.0000	0.0000	1.0000
ANGINA PECTORIS	1.0000	0.0000	0.0000	0.0000	0.0000
ANOREXIA	0.5000	0.0000	0.0000	0.0000	0.0000
ANXIETY	0.0000	0.0000	0.0000	0.0000	0.0000
ASTHENIA	0.2500	0.0000	0.0000	0.0000	0.2500
ASTHMA	0.0000	0.0000	0.0000	0.0000	1.0000
BACK PAIN	0.5000	0.0000	0.0000	0.0000	0.0000
BRONCHITIS	0.3333	0.0000	0.0000	0.0000	0.3333
CHEST PAIN	0.5000	0.0000	0.0000	0.0000	0.0000

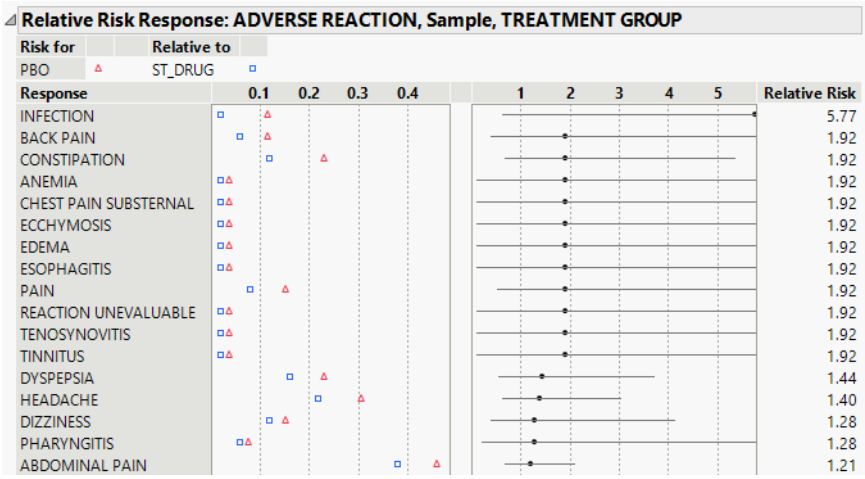
The conditional association matrix provides the conditional probability of one adverse reaction given the presence of another reaction. The probabilities are across all groups. The probability of abnormal vision given that a patient has abdominal pain is 0.0323.

Tip: Hover over the heat map for conditional probabilities.

9. Click the Categorical red triangle and select **Relative Risk.**
10. Select PBO in the window and click **OK**.

    Right-click the Relative Risk Report in the window and select **Sort by Column**.
11. Select Relative Risk and click **OK**.

Figure 3.32 Relative Risk Report (Partial Report)



The Relative Risk option computes relative risks of different responses as the ratio of the risk for each level of the grouping variable. The default Relative Risk report lists the response name, the risk (rate) for each level of the grouping variable, a plot of the relative risk with 95% confidence intervals, and the relative risk estimate. Here you can compare the relative risk of the adverse reactions by treatment group. The relative risk of an infection is 5.7 times greater for PBO relative to ST_DRUG. However, the confidence interval is very wide and includes a relative risk of 1.0. A relative risk of 1.0 occurs when the risk is equal for each level of the grouping variable.

Right-click and select **Columns > Lower 95%** and **Columns > Upper 95%** to add 95% confidence intervals on the relative risk estimates to the report table.

Example of Rater Agreement

This example uses the Attribute Gauge.jmp sample data, which has the ratings (0/1) from three operators rating 50 parts three times.

1. Select **Help > Sample Data Library** and open Attribute Gauge.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Related tab.
4. Select A, B, and C, and click **Rater Agreement** on the Related tab.
5. Click **OK**.

Figure 3.33 Rater Agreement Report

Rater Agreement(A, B, C)

Freq / Share		Response 0	Response 1	Total Responses
Response	A	50 33.3%	100 66.7%	150
	B	47 31.3%	103 68.7%	150
	C	51 34.0%	99 66.0%	150

Share Chart		Response 0	Response 1	
Response	A			150
	B			150
	C			150

Rater Agreement Statistics

Degree of Agreement

Degree of Agreement

Response1	Response2	Kappa	Std Err	Bowker Symmetry	Bowker P-Value
A	B	0.862944	0.044198	1	0.3173
A	C	0.776119	0.054671	0.066667	0.7963
B	C	0.788007	0.053702	1.142857	0.2850

Two by two tables are the same as Bowker's Test

Details

Rater Row	Rater Col	Level	0	1
B	A	0	44	3
B	A	1	6	97
C	A	0	43	8
C	A	1	7	92
C	B	0	42	9
C	B	1	5	94

The rater agreement is strong as shown by the Kappa statistics. The Kappa statistic can take on a value between 0 (no agreement) to 1.0 (perfect agreement). The details section provides 2x2 tables for each pair of raters. The Bowker test of symmetry tests the null hypothesis that cell proportions are symmetric for all pairs of cells ($p_{ij} = p_{ji}$ for all i, j). Here, p-values for the Bowker test are all greater than 0.05, indicating no strong evidence of asymmetry between raters.

Example of Repeated Measures

This example uses the Presidential Elections.jmp sample data table, which contains United States presidential election results for each state from 1980 through 2012. Use this data table to explore repeated measures where we consider the election results as repeated measures.

1. Select **Help > Sample Data Library** and open Presidential Elections.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Related tab.
4. Select 1980 Winner through 2012 Winner, and click **Repeated Measures** on the Related tab.
5. Click **OK**.

6. Near the bottom of the report window, click the gray Transition Report disclosure icon to open the Transition Report.

Figure 3.34 Repeated Measures Transition Report

Transition Report

All From	to	Transition Counts			Transition Rates			
All 1980 Winner	1984 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**1984 Winner**
		Democrat	1	5	Democrat	0.1667	0.8333	
		Republican	0	44	Republican	0.0000	1.0000	
All 1984 Winner	1988 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**1988 Winner**
		Democrat	1	0	Democrat	1.0000	0.0000	
		Republican	9	40	Republican	0.1837	0.8163	
All 1988 Winner	1992 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**1992 Winner**
		Democrat	10	0	Democrat	1.0000	0.0000	
		Republican	22	18	Republican	0.5500	0.4500	
All 1992 Winner	1996 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**1996 Winner**
		Democrat	29	3	Democrat	0.9063	0.0938	
		Republican	2	16	Republican	0.1111	0.8889	
All 1996 Winner	2000 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**2000 Winner**
		Democrat	20	11	Democrat	0.6452	0.3548	
		Republican	0	19	Republican	0.0000	1.0000	
All 2000 Winner	2004 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**2004 Winner**
		Democrat	18	2	Democrat	0.9000	0.1000	
		Republican	1	29	Republican	0.0333	0.9667	
All 2004 Winner	2008 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**2008 Winner**
		Democrat	19	0	Democrat	1.0000	0.0000	
		Republican	9	22	Republican	0.2903	0.7097	
All 2008 Winner	2012 Winner		**Democrat**	**Republican**		**Democrat**	**Republican**	**2012 Winner**
		Democrat	26	2	Democrat	0.9286	0.0714	
		Republican	0	22	Republican	0.0000	1.0000	

The Transition Report is unique to the repeated measures analysis. This report includes counts and rates of differences between subsequent time points. Between 1980 and 1984, there were 5 Democratic states that transitioned to Republican states at a rate of 0.8333 or 5 out of 6 states. In 1980, they voted Democratic but voted Republican in 1984. Between 2008 and 2012, there were 2 out of 28 Democratic states that transitioned to Republican at a rate of 0.0714. All other states voted the same way in both the 2008 and 2012 elections.

Example of the Multiple Responses

The examples in this section use sample data tables that contain the same information organized in five different data table layouts. The data come from testing a fabrication line on three different occasions under two different conditions. Each set of operating conditions (or batch) yielded 50 units for inspection. Inspectors recorded seven types of defects. Each unit could have zero, one, or more than one defect. A unit could have more than one defect of the same kind.

Multiple Response

The Failure3MultipleField.jmp sample data table has a row for each unit and multiple columns for defects, where defects are entered one per column. In this example, there are three columns for defects. Thus, any one unit had at most three defects.

1. Select **Help > Sample Data Library** and open Quality Control/Failure3MultipleField.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Multiple tab.
4. Select Failure1, Failure2, and Failure3, and click **Multiple Response** on the Multiple tab.
5. Select clean and date and click **X, Grouping Category**.
6. Click **OK**.

Figure 3.35 Multiple Response Report

Multiple Response(Failure1, Failure2, Failure3) By clean*date

			Freq / Share / Rate	Failure									
				contamination	corrosion	doping	metallization	miscellaneous	oxide defect	silicon defect	Total Responses	Total Cases	Total Cases Responding
clean	after	date	OCT 1	12	2	0	4	2	1	2	23	50	23
				52.2%	8.7%	0.0%	17.4%	8.7%	4.3%	8.7%			
				24.0%	4.0%	0.0%	8.0%	4.0%	2.0%	4.0%			
			OCT 2	10	1	1	5	1	2	3	23	50	23
				43.5%	4.3%	4.3%	21.7%	4.3%	8.7%	13.0%			
				20.0%	2.0%	2.0%	10.0%	2.0%	4.0%	6.0%			
			OCT 3	8	3	0	5	0	1	0	17	50	17
				47.1%	17.6%	0.0%	29.4%	0.0%	5.9%	0.0%			
				16.0%	6.0%	0.0%	10.0%	0.0%	2.0%	0.0%			
	before	date	OCT 1	14	2	1	2	3	8	1	31	50	28
				45.2%	6.5%	3.2%	6.5%	9.7%	25.8%	3.2%			
				28.0%	4.0%	2.0%	4.0%	6.0%	16.0%	2.0%			
			OCT 2	15	2	2	1	4	6	0	30	50	30
				50.0%	6.7%	6.7%	3.3%	13.3%	20.0%	0.0%			
				30.0%	4.0%	4.0%	2.0%	8.0%	12.0%	0.0%			
			OCT 3	22	2	3	4	0	3	2	36	50	36
				61.1%	5.6%	8.3%	11.1%	0.0%	8.3%	5.6%			
				44.0%	4.0%	6.0%	8.0%	0.0%	6.0%	4.0%			

Share Chart				Failure							
				contamination	corrosion	doping	metallization	miscellaneous	oxide defect	silicon defect	
clean	after	date	OCT 1								23
			OCT 2								23
			OCT 3								17
	before	date	OCT 1								31
			OCT 2								30
			OCT 3								36

The crosstab table has a row for each batch and a column for each defect type. The frequency, share, and rate of each defect within each batch are shown in the table cells. For example, for the batch after cleaning on OCT 1, there were 12 contamination defects representing 12/23 or 52.2% of the defects for that batch. The 12 contamination defects were from 50 units. For each clean and date combination, there were 50 total units. Each unit could have one or more defects. Therefore, the rate of contamination per unit was 24%. For the batch before cleaning on OCT 1, the Total Cases Responding is 28. The Total Responses count is 31, because three of the cases reported two defects.

Multiple Response by ID

The Failure3ID.jmp sample data table has a row for each defect type within each batch, a column for the number of occurrences of each defect type, and an ID column for each batch.

Figure 3.36 Failure3ID Data Table (Partial Table)

	failure	N	clean	date	SampleSize	ID
1	contamination	14	before	OCT 1	50	OCT 1 before
2	corrosion	2	before	OCT 1	50	OCT 1 before
3	doping	1	before	OCT 1	50	OCT 1 before
4	metallization	2	before	OCT 1	50	OCT 1 before
5	miscellaneous	3	before	OCT 1	50	OCT 1 before
6	oxide defect	8	before	OCT 1	50	OCT 1 before
7	silicon defect	1	before	OCT 1	50	OCT 1 before
8	doping	0	after	OCT 1	50	OCT 1 after
9	corrosion	2	after	OCT 1	50	OCT 1 after
10	metallization	4	after	OCT 1	50	OCT 1 after

1. Select **Help > Sample Data Library** and open Quality Control/Failure3ID.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Multiple tab.
4. Select failure and click **Multiple Response by ID** on the Multiple tab.
5. Select clean and date and click **X, Grouping Category**.
6. Select SampleSize and click **Sample Size**.
7. Select N and click **Freq**.
8. Select ID and click **ID**.
9. Click **OK**.

The resulting report is the same as the report shown in Figure 3.35 with the exception of the Total Cases Responding column in the crosstab table. Here, the defect counts were summarized. From the summarized table, there is no record of the number of units with zero defects. Thus, the Total Cases Responding is the full batch size of 50 for each batch.

Multiple Delimited

The Failures3Delimited.jmp sample data table has a row for each unit with a single column in which the defects are recorded, delimited by a comma. Note in the partial data table, shown in Figure 3.37, that some units did not have any observed defects, so the failures column is empty.

Figure 3.37 Failure3Delimited.jmp Data Table (Partial Table)

	failures	clean	date	ID	ID Label
1		before	OCT 1	1	OCT 1 before
2	oxide defect	before	OCT 1	1	OCT 1 before
3	contamination,oxide defect	before	OCT 1	1	OCT 1 before
4		before	OCT 1	1	OCT 1 before
5	contamination	before	OCT 1	1	OCT 1 before
6	oxide defect	before	OCT 1	1	OCT 1 before
7	contamination	before	OCT 1	1	OCT 1 before
8		before	OCT 1	1	OCT 1 before
9		before	OCT 1	1	OCT 1 before
10	metallization,contamination	before	OCT 1	1	OCT 1 before

1. Select **Help > Sample Data Library** and open Quality Control/ Failures3Delimited.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Multiple tab.
4. Select failures and click **Multiple Delimited** on the Multiple tab.
5. Select clean and date and click **X, Grouping Category**.
6. Click **OK**.

When you click **OK**, you also get the report shown in Figure 3.35.

Note: If you specify more than one delimited column, separate analyses are produced for each column.

Indicator Group

The Failures3Indicators.jmp sample data table has a row for each unit and an indicator column for each defect type. The data entry in each defect column is a 0 if that defect was not observed and a 1 if the defect was observed for the unit.

Figure 3.38 Faliure3Indicators.jmp Data Table (Partial Table)

	clean	date	ID	ID Label	contamination	corrosion	doping	metallization	miscellaneous	oxide defect	silicon defect
1	before	OCT 1	1	OCT 1 before	0	0	0	0	0	0	0
2	before	OCT 1	1	OCT 1 before	0	0	0	0	0	1	0
3	before	OCT 1	1	OCT 1 before	1	0	0	0	0	1	0
4	before	OCT 1	1	OCT 1 before	0	0	0	0	0	0	0
5	before	OCT 1	1	OCT 1 before	1	0	0	0	0	0	0

1. Select **Help > Sample Data Library** and open Quality Control/Failures3Indicators.jmp.

2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Multiple tab.
4. Select contamination through silicon defect and click **Indicator Group** on the Multiple tab.
5. Select clean and date and click **X, Grouping Category**.
6. Click **OK**.

When you click **OK**, you get the report shown in Figure 3.35.

Response Frequencies

The Failure3Freq.jmp sample data table has a row for each batch, a column for each defect type, and a column for the batch size. The data entries in the defect columns are the number of occurrences of each defect in the batch.

Figure 3.39 Failure3Freq.jmp Data Table

	clean	date	contamination	corrosion	doping	metallization	miscellaneous	oxide defect	silicon defect	SampleSize
1	after	OCT 1	12	2	0	4	2	1	2	50
2	after	OCT 2	10	1	1	5	1	2	3	50
3	after	OCT 3	8	3	0	5	0	1	0	50
4	before	OCT 1	14	2	1	2	3	8	1	50
5	before	OCT 2	15	2	2	1	4	6	0	50
6	before	OCT 3	22	2	3	4	0	3	2	50

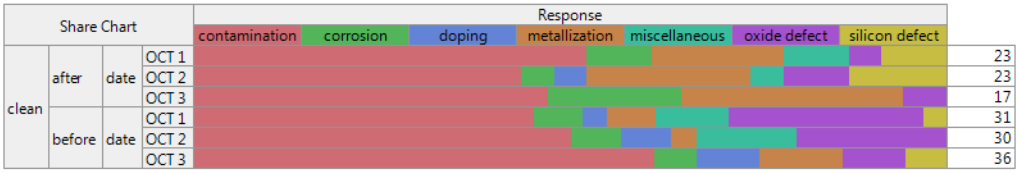
1. Select **Help > Sample Data Library** and open Quality Control/Failure3Freq.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Multiple tab.
4. Select the frequency variables (contamination through silicon defect).
5. On the Multiple tab, click **Response Frequencies**.
6. Select clean and date and click **X, Grouping Category**.
7. Select Sample Size and click **Sample Size**.
8. Click **OK**.

Figure 3.40 Defect Rate Output

Sample Size: SampleSize

Response Frequencies(contamination, corrosion, doping, metallization, miscellaneous, oxide defect, silicon defect) By clean*date

				Freq / Share / Rate	contamination	corrosion	doping	metallization	miscellaneous	oxide defect	silicon defect	Total Responses	Total Cases	Total Cases Responding
clean	after	date	OCT 1	12	2	0	4	2	1	2	23	50	50
				52.2%	8.7%	0.0%	17.4%	8.7%	4.3%	8.7%			
				24.0%	4.0%	0.0%	8.0%	4.0%	2.0%	4.0%			
			OCT 2	10	1	1	5	1	2	3	23	50	50
				43.5%	4.3%	4.3%	21.7%	4.3%	8.7%	13.0%			
				20.0%	2.0%	2.0%	10.0%	2.0%	4.0%	6.0%			
			OCT 3	8	3	0	5	0	1	0	17	50	50
				47.1%	17.6%	0.0%	29.4%	0.0%	5.9%	0.0%			
				16.0%	6.0%	0.0%	10.0%	0.0%	2.0%	0.0%			
	before	date	OCT 1	14	2	1	2	3	8	1	31	50	50
				45.2%	6.5%	3.2%	6.5%	9.7%	25.8%	3.2%			
				28.0%	4.0%	2.0%	4.0%	6.0%	16.0%	2.0%			
			OCT 2	15	2	2	1	4	6	0	30	50	50
				50.0%	6.7%	6.7%	3.3%	13.3%	20.0%	0.0%			
				30.0%	4.0%	4.0%	2.0%	8.0%	12.0%	0.0%			
			OCT 3	22	2	3	4	0	3	2	36	50	50
				61.1%	5.6%	8.3%	11.1%	0.0%	8.3%	5.6%			
				44.0%	4.0%	6.0%	8.0%	0.0%	6.0%	4.0%			



The resulting output is the same as that in Figure 3.35 with the exception of the Total Cases Responding column in the crosstab table. Here, the defect counts were summarized. From the summarized table, there is no record of the number of units with zero defects. Thus, the Total Cases Responding is the full batch size of 50 for each batch.

Example of Mean Score with Comparison Letters

This example uses the Consumer Preferences.jmp sample data table to explore the relationship between employee tenure and having school age children. The Employee Tenure column is a numeric column with values 1, 2, 3, and 4. These values have been assigned Value Labels using the Value Labels column property. To evaluate an average employee tenure using the Mean Score option in the categorical platform, assign Value Scores to the column values. For more information about column properties, see *Using JMP*.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. In the data table, right-click the Employee Tenure column heading and select **Column Properties > Value Scores**.
3. Enter 1 for **Value** and 3 for **Score** and click **Add**.
4. Enter 2 for **Value** and 7.5 for **Score** and click **Add**.

5. Enter 3 for **Value** and 15 for **Score** and click **Add**.
6. Enter 4 for **Value** and 25 for **Score** and click **Add**.
7. Click **OK**.
8. Select **Analyze > Consumer Research > Categorical**.
9. Select Employee Tenure and click **Responses** on the Simple tab.
10. Select School Age Children and click **X, Grouping Category**.
11. Click **OK**.
12. Click the Categorical red triangle and select **Mean Score**.
13. Click the Categorical red triangle and select **Mean Score Comparisons**.

Figure 3.41 Categorical Report with Mean Scores

Employee Tenure By School Age Children

			Freq / Share	Employee Tenure: less than 5 years	5 to 10 years	10 to 20 years	more than 20 years	Total Responses	Mean	Compare Means
School Age Children	Yes	A		53	65	44	23	185	10.170	
				28.6%	35.1%	23.8%	12.4%			
	No	B		107	74	42	40	263	9.529	
				40.7%	28.1%	16.0%	15.2%			

Default Comparison Groups: A/B

Shows letter of the category it is significantly different from at the higher share level
* Base count warning 100 Uppercase Alpha Level 0.05
** Base count minimum 30 Lowercase Alpha Level 0.1

Share Chart		Employee Tenure: less than 5 years	5 to 10 years	10 to 20 years	more than 20 years	
School Age Children	Yes					185
	No					263

The mean employee tenure for those with school age children is 10.17 and 9.53 for those without school age children. Because the means are not statistically different, the Compare Means column in the Crosstab table is empty. If there were a difference, a letter would indicate the difference. If you had not used value scores, then the mean for those with school age children would be 2.20 and 2.057 for those without school age children.

Tip: Be aware of how your data are recorded when using the mean score option. If your data are recorded as coded numeric data with value labels, the mean value calculations are based on the numeric data. If the numeric values do not have meaning, use the Value Score column property to assign meaningful values to the response levels.

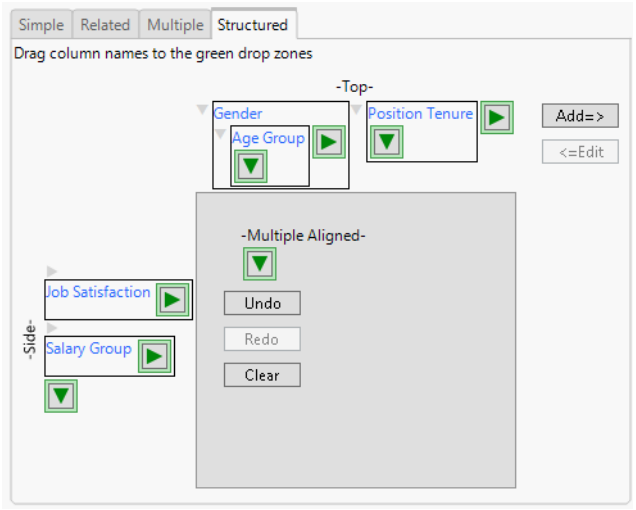
Example of a Structured Report

This example uses the Consumer Preferences.jmp sample data table to compare job satisfaction and salary against gender by age group and position tenure. Use the Structured tab to create the report.

1. Select **Help > Sample Data Library** and open Consumer Preferences.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Select the Structured tab.
4. Drag Gender to the green drop zone at the **Top** of the table on the Structured tab.
5. Drag Age Group to the green drop zone just below Gender.
6. Drag Position Tenure to the green drop zone at the **Top** of the table next to Gender.
7. Drag Job Satisfaction to the green drop zone at the **Side** of the table.
8. Drag Salary Group to the green drop zone at the **Side** of the table under Job Satisfaction.

Tip: Click on the green drop zone arrows to select columns.

Figure 3.42 Structured Tab Report Setup



9. Click **Add=>**.
10. Click **OK**.
11. Click the Categorical red triangle and select **Test Response Homogeneity**.

Figure 3.43 Structured Tab Report Example

Job Satisfaction+Salary Group By Gender*Age Group+Position Tenure

		Freq	Gender																
		Share	M							F									
			Age Group							Age Group							Position Tenure			
			25-29	30-34	35-39	40-44	45-49	50-54	>54	25-29	30-34	35-39	40-44	45-49	50-54	>54	less than 5 years	5 to 10 years	10 to 20 years	more than 20 years
Job Satisfaction	Not at all satisfied		3	3	1	3	3	2	2	3	1	0	5	2	1	3	15	11	6	0
			5.5%	10.3%	3.3%	8.8%	7.9%	8.3%	4.1%	5.2%	2.6%	0.0%	27.8%	14.3%	5.3%	13.6%	7.2%	8.2%	6.8%	0.0%
	Somewhat satisfied		32	14	15	19	19	10	26	34	30	12	6	5	11	9	122	70	40	10
			58.2%	48.3%	50.0%	55.9%	50.0%	41.7%	53.1%	58.6%	76.9%	63.2%	33.3%	35.7%	57.9%	40.9%	58.7%	52.2%	45.5%	55.6%
	Extremely satisfied		20	12	14	12	16	12	21	21	8	7	7	7	7	10	71	53	42	8
			36.4%	41.4%	46.7%	35.3%	42.1%	50.0%	42.9%	36.2%	20.5%	36.8%	38.9%	50.0%	36.8%	45.5%	34.1%	39.6%	47.7%	44.4%
	Total Responses		55	29	30	34	38	24	49	58	39	19	18	14	19	22	208	134	88	18
Salary Group	less than 40000		20	6	4	8	6	5	10	35	19	6	6	2	7	4	80	40	15	3
			36.4%	20.7%	13.3%	23.5%	15.8%	20.8%	20.4%	60.3%	48.7%	31.6%	33.3%	14.3%	36.8%	18.2%	38.5%	29.9%	17.0%	16.7%
	40000 to 60000		16	14	11	12	14	7	13	15	10	8	6	7	6	12	67	38	37	9
			29.1%	48.3%	36.7%	35.3%	36.8%	29.2%	26.5%	25.9%	25.6%	42.1%	33.3%	50.0%	31.6%	54.5%	32.2%	28.4%	42.0%	50.0%
	60000 to 80000		8	5	7	7	8	8	10	5	5	3	3	3	4	5	32	25	19	5
			14.5%	17.2%	23.3%	20.6%	21.1%	33.3%	20.4%	8.6%	12.8%	15.8%	16.7%	21.4%	21.1%	22.7%	15.4%	18.7%	21.6%	27.8%
	80000 to 120000		9	3	6	4	7	3	7	1	2	1	2	1	1	0	19	17	10	1
			16.4%	10.3%	20.0%	11.8%	18.4%	12.5%	14.3%	1.7%	5.1%	5.3%	11.1%	7.1%	5.3%	0.0%	9.1%	12.7%	11.4%	5.6%
	greater than 120000		2	1	2	3	3	1	9	2	3	1	1	1	1	1	10	14	7	0
			3.6%	3.4%	6.7%	8.8%	7.9%	4.2%	18.4%	3.4%	7.7%	5.3%	5.6%	7.1%	5.3%	4.5%	4.8%	10.4%	8.0%	0.0%
	Total Responses		55	29	30	34	38	24	49	58	39	19	18	14	19	22	208	134	88	18

Test Response Homogeneity

Response Dimension Label	Sample Dimension Label	LR Chisq	LR PValue	Pearson Chisq	Pearson PValue
Job Satisfaction	Gender = M, Age Group	4.65556	0.9685	4.59049	0.9703
Job Satisfaction	Gender = F, Age Group	23.8406	0.0214*	24.9917	0.0149*
Salary Group	Gender = M, Age Group	22.0707	0.5750	23.8074	0.4727
Salary Group	Gender = F, Age Group	27.6209	0.2764	26.3992	0.3332
Job Satisfaction	Position Tenure	8.00461	0.2378	6.76025	0.3436
Salary Group	Position Tenure	25.8336	0.0113*	24.0752	0.0199*

The structured tab report contains the table that you specified in the Structured tab. The tests for response homogeneity are for each combination of grouping variables. We see that, for males, there is no difference in job satisfaction across age groups (Pearson p-value = 0.9703). For females, there is a difference in job satisfaction across age groups (Pearson p-value = 0.0149). The middle-aged females tend to be the least satisfied with their jobs. Share and frequency charts can be added to your report to visualize your results.

Example of a Multiple Response with Nonresponse

This example uses the Color Preference Survey.jmp sample data table, which contains survey data about people's color preferences. You can use the Categorical platform to summarize results from a multiple response question with nonresponses. A nonresponse is a participant who did not answer the question. Surveys often use "none of the above" or "not applicable" as a response choice. This provides a method for distinguishing responders who do not see an appropriate response from those who did not answer the question.

1. Select **Help > Sample Data Library** and open Color Preference Survey.jmp.
2. Select **Analyze > Consumer Research > Categorical**.
3. Click the **Structured** tab.
4. Select What colors do you like? (with nonresponse) and drag it to the green arrow drop zone at the **Side** of the table.

Note: This column contains multiple response data where the response levels are delimited with commas. There are three rows that contain no responses. This column was constructed for this example and does not align with the responses in the individual "What colors do you like" columns.

5. Click the **Top** green arrow and select What is your gender?
6. Click **Add=>**.
7. Click **OK**.

Figure 3.44 Initial Cross Tabulation

Categorical

What colors do you like? (with nonresponse) By What is your gender?

		Freq / Share / Rate	What is your gender?	
			Female	Male
What colors do you like? (with nonresponse)	Blue		31 21.5% 83.8%	38 21.5% 88.4%
	Green		24 16.7% 64.9%	36 20.3% 83.7%
	None of the above		2 1.4% 5.4%	0 0.0% 0.0%
	Orange		12 8.3% 32.4%	23 13.0% 53.5%
	Pink		19 13.2% 51.4%	9 5.1% 20.9%
	Purple		28 19.4% 75.7%	23 13.0% 53.5%
	Red		18 12.5% 48.6%	33 18.6% 76.7%
	Yellow		10 6.9% 27.0%	15 8.5% 34.9%
	Total Responses		144	177
	Total Cases		37	43
	Total Cases Responding		36	41

Note that the number of total cases is higher than the number of total cases responding. There are 37 Females who filled out the survey but only 36 responded to this question.

8. Click the Categorical red triangle and select **Test Multiple Responses > Exclude Nonresponses**.
9. Click the Categorical red triangle and select **Test Multiple Responses > Homogeneity Test, Binomial**.

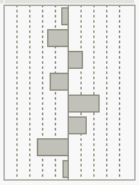
Figure 3.45 Binomial Homogeneity Test Results

Test Each Response, Binomial

Note: Nonresponse rows are excluded.

What is your gender?, What colors do you like? (with nonresponse)

What colors do you like? (with nonresponse)	ChiSquare	Prob>ChiSq
Blue	0.8918	0.3450
Green	5.0612	0.0245*
None of the above	3.1020	0.0782
Orange	4.0516	0.0441*
Pink	7.9930	0.0047*
Purple	4.1132	0.0426*
Red	8.0999	0.0044*
Yellow	0.6819	0.4089



Assuming multiple responses are Check All That Apply (CATA).

The analysis supports that blue and yellow have similar preference levels across genders while other colors tend to be favored more or less across genders. For example, Red appears to be liked more by males than by females. The *p*-value for the test is 0.0044, which supports a conclusion that there is a difference in preference across genders. From the crosstab in Figure 3.44 you find that 76.7% of the males liked the color red as compared to 48.6% of the females.

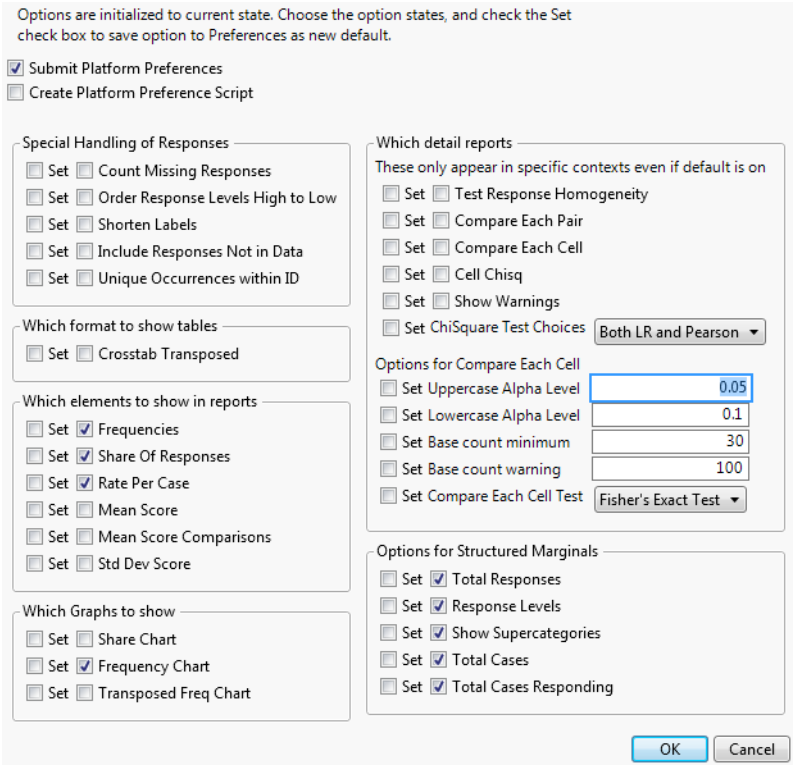
The note at the top of the output indicates that nonresponse rows are excluded from the analysis. The note at the bottom of the output indicates that it is assumed that the multiple response is a check all that applies type question. This means that a respondent can select more than one response to the question. The tests are performed using only those rows with responses. The three rows that did not have responses for the question are not included.

Tip: You can obtain the same output using the Multiple tab. Select What colors do you like? (with nonresponse) as a Multiple Delimited response and What is your Gender? as a grouping variable.

Set Preferences

The Categorical red triangle menu has a Set Preferences option to enable you to specify settings and preferences.

Figure 3.46 Set Preferences Window



Select the **Set** box for the options that you want to set. Select the option box if you want the option to appear by default, or deselect the option box if you do not want the option to appear by default. To submit the changes that you make to the platform preferences, select the **Submit Platform Preferences** box. To save the changes that you make as a preference script, select the **Create Platform Preference Script** box. When the Categorical platform is launched, the preferences associated with the current preference set are used to create the Categorical report.

Running the saved script submits the preferences to the platform preferences. You can use the platform preference script to share a preference set among multiple users, or to save the settings for specific projects.

Statistical Details for the Categorical Platform

This section contains statistical details for the Categorical platform.

Rao-Scott Correction

The Rao-Scott correction is applied to the test of response homogeneity for multiple responses. See Lavassani et al. (2009).

In the case of a multiple response, you can have overlapping samples, meaning a single participant can provide more than one response. The Pearson chi-square test is not appropriate for multiple responses, because the multiple responses violate the Pearson chi-square test assumption of independence. In addition, expected values calculated using the marginal totals are influenced by the multiple responses because the totals are larger than if multiple responses were not allowed.

The Rao-Scott chi-square statistic is defined as follows:

$$\chi_C^2 = \frac{\chi^2}{\bar{\delta}}$$

where

χ^2 is the standard Pearson Chi-squared statistic

$\bar{\delta} = 1 - \frac{m_{++}}{n_+ C}$ is the correction factor

The correction factor contains the following quantities:

m_{++} is the total count of the multiple responses

n_+ is the total number of participants and

C is the number of response levels (number of columns in the Crosstab table).

The degrees of freedom are $(R\text{-}1)C$ or the number of rows minus 1 times the number of columns.

Chapter 4

Choice Models

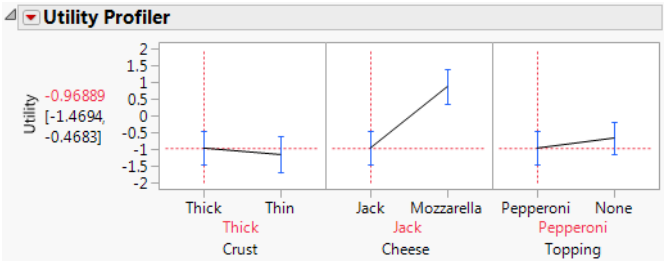
Fit Models for Choice Experiments

Use the Choice platform to analyze the results of choice experiments conducted in the course of market research. Choice experiments are used to help discover which product or service attributes your potential customers prefer. You can use this information to design products or services that have the attributes that your customers most desire.

The Choice platform enables you to do the following:

- Use information about subject (customer) traits as well as product attributes.
- Analyze choice experiments where respondents were allowed to select “none of these”.
- Integrate data from one, two, or three sources.
- Use the integrated profiler to understand, visualize, and optimize the response (utility) surface.
- Obtain subject-level scores for segmenting or clustering your data.
- JMP PRO Estimate subject-specific coefficients using a Bayesian approach.
- Use bias-corrected maximum likelihood estimators (Firth 1993).

Figure 4.1 Choice Platform Utility Profiler



Contents

Overview of the Choice Modeling Platform 83
Examples of the Choice Platform 85
One Table Format with No Choice 85
Multiple Table Format 88
Launch the Choice Platform 95
Launch Window for One Table, Stacked 96
Launch Window for Multiple Tables, Cross-Referenced 98
Choice Model Report 104
Effect Summary 104
Parameter Estimates 105
Likelihood Ratio Tests 106
Bayesian Parameter Estimates 106
Choice Platform Options 108
Willingness to Pay 111
Additional Examples 114
Example of Making Design Decisions 114
Example of Segmentation 122
Example of Logistic Regression Using the Choice Platform 126
Example of Logistic Regression for Matched Case-Control Studies 129
Example of Transforming Data to Two Analysis Tables 131
Example of Transforming Data to One Analysis Table 136
Statistical Details for the Choice Platform 138
Special Data Table Rules 138
Utility and Probabilities 139
Gradients 140

Overview of the Choice Modeling Platform

Choice modeling, pioneered by McFadden (1974), is a powerful analytic method used to estimate the probability of individuals making a particular choice from presented alternatives. Choice modeling is also called conjoint choice modeling, discrete choice analysis, and conditional logistic regression.

A choice experiment studies customer preferences for a set of product or process (in the case of a service) attributes. Respondents are presented sets of product attributes, called *profiles*. Each respondent is shown a small set of profiles, called a *choice set*, and asked to select the preference that he or she most prefers. Each respondent is usually presented with several choice sets. Use the Choice platform to analyze the results of a choice experiment.

Note: You can design your choice experiment using the Choice Design platform. See the *Design of Experiments Guide*.

Because customers vary in how they value attributes, many market researchers view market segmentation as an important step in analyzing choice experiments. Otherwise, you risk designing a product or process that pleases the "average" customer, who does not actually exist, and ignoring the preferences of market segments that *do* exist.

For background on choice modeling, see Louviere et al. (2015), Train (2009), and Rossi et al. (2005).

The Choice Platform

The Choice Modeling platform uses a form of conditional logistic regression to estimate the probability that a configuration is preferred. Unlike simple logistic regression, choice modeling uses a linear model to model choices based on response attributes and not solely upon subject characteristics. In choice modeling, a respondent might choose between two cars that are described by a combination of ten attributes, such as price, passenger load, number of cup holders, color, GPS device, gas mileage, anti-theft system, removable-seats, number of safety features, and insurance cost.

The Choice platform allows respondents to *not* make a choice from among a set of profiles. The *no choice* option is treated as a product with a single attribute ("Select none of these") that respondents are allowed to select. The parameter estimate for the No Choice attribute can then be interpreted in many ways, depending on the assumptions of the model. The Choice platform also enables you to obtain subject-level information, which can be useful in segmenting preference patterns.

You can obtain bias-corrected maximum likelihood estimators as described by Firth (1993). This method has been shown to produce better estimates and tests than MLEs without bias correction. In addition, bias-corrected MLEs improve separation problems that tend to occur in logistic-type models. See Heinze and Schemper (2002) for a discussion of the separation problem in logistic regression.

Note: The Choice platform is not appropriate to use for fitting models that involve ranking, scoring, or nested hierarchical choices. You can use PROC MDC in SAS/ETS for these analyses.

Choice Designs in Developing Products and Services

Although customer satisfaction surveys can disclose what is wrong with a product or service, they fail to identify consumer preferences with regard to specific product attributes. When engineers design a product, they routinely make hundreds or thousands of small design decisions. If customer testing is feasible and research participants (subjects) are available, you can use choice experiments to guide some design decisions.

Decreases in survey deployment, modeling, and prototyping costs facilitate the customer evaluation of many attributes and alternatives as a product is designed. Choice modeling can be used in Six Sigma programs to improve consumer products, or, more generally, to make the products that people want. Choice experiments obtain data on customer preferences, and choice modeling analysis reveals such preferences.

Segmentation

Market researchers sometimes want to analyze the preference structure for each subject separately in order to see whether there are groups of subjects that behave differently. However, there are usually not enough data to do this with ordinary estimates. If there are sufficient data, you can specify the subject identifier as a "By groups" in the Response Data or you could introduce a subject identifier as a subject-side model term. This approach, however, is costly if the number of subjects is large.

If there are not sufficient data to specify "By groups," you can segment in JMP by clustering subjects using the Save Gradients by Subject option. The option creates a new data table containing the average Hessian-scaled gradient on each parameter for each subject. For an example, see "Example of Segmentation". For more information about the gradient values, see "Gradients".

JMP PRO In JMP Pro, you can request that the Choice platform use a Hierarchical Bayes approach in order to facilitate market segmentation. Bayesian modeling provides subject-specific estimates of model parameters (also called part-worths). These parameters i can be analyzed with hierarchical clustering or some other type of cluster analysis to reveal market segments.

Examples of the Choice Platform

In a study of pizza preferences, each respondent is presented with four choice sets, each containing two profiles. The Choice platform can analyze data that is in a one table format or a multiple data format. In the multiple table format, information about responses, choice sets, and subjects is saved in different data tables. In the one table format, that information is contained in a single data table.

- "One Table Format with No Choice" shows how to analyze a subset of the available data in a one table format.
- "Multiple Table Format" shows how to bring together information from different tables into one Choice analysis

One Table Format with No Choice

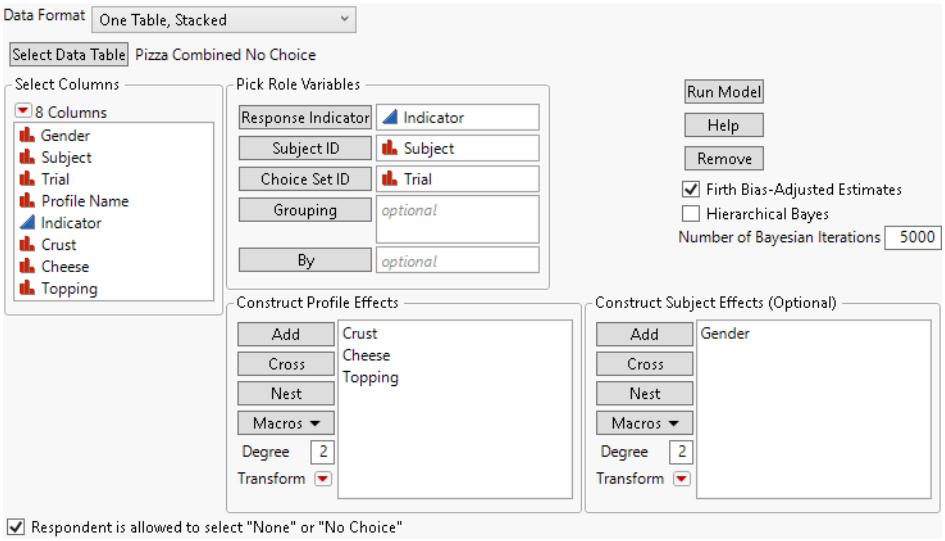
In this example, some respondents do not express a preference for either profile. The respondent makes "no choice". When a respondent does not express a preference, the respondent's choice indicator is entered as missing.

1. Select **Help > Sample Data Library** and open Pizza Combined No Choice.jmp.

   Choice sets are defined by the combination of Subject and Trial. Notice that there are missing values in the Indicator column for some choice sets.
2. Select **Analyze > Consumer Research > Choice**.

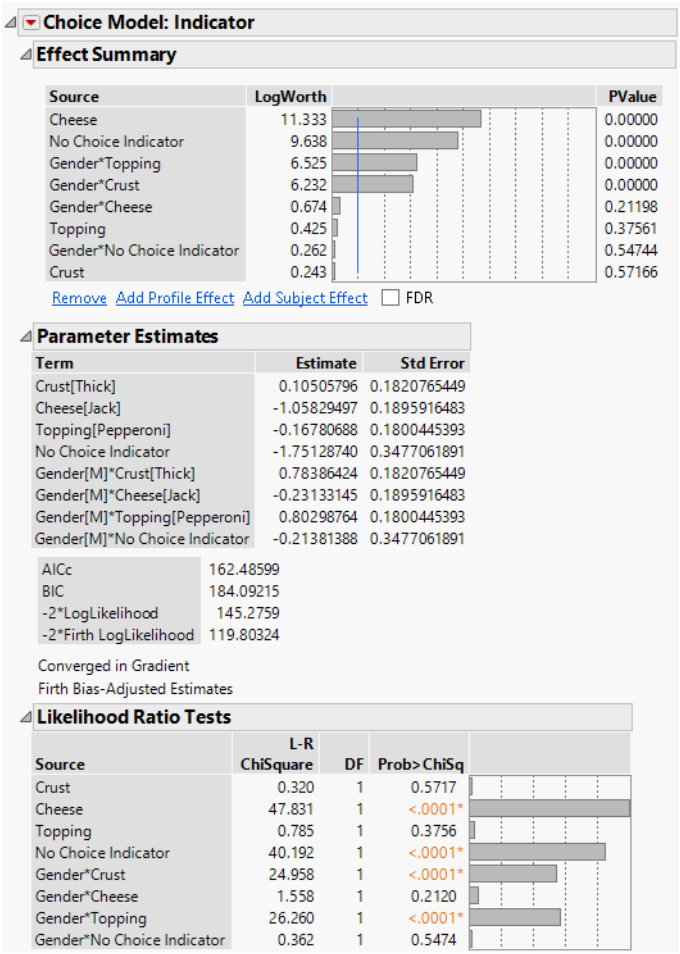
   The One Table, Stacked data format is the default.
3. Click **Select Data Table**.
4. Select **Pizza Combined No Choice** and click **OK**.
5. Complete the launch window:
   - Select Indicator and click **Response Indicator**.
   - Select Subject and click **Subject ID**.
   - Select Trial and click **Choice Set ID**.
   - Select Crust, Cheese, and Topping and click **Add** in the Construct Profile Effects panel.
   - Select Gender and click **Add** in the Construct Subject Effects (Optional) panel.

Figure 4.2 Completed Launch Window



6. Check the box next to **Respondent is allowed to select “None” or “No Choice”**.
7. Click **Run Model**.

Figure 4.3 Report Showing No Choice as an Effect



The Effect Summary report shows the effects in order of significance. Cheese is the most significant effect, followed by the No Choice Indicator, which is treated as a model effect. The subject effect interactions Gender*Topping and Gender*Crust are also significant, indicating that preferences for Topping and Crust depend on Gender market segments.

To get some insight on the nature of the No Choice responses, select and view those choice sets that resulted in No Choice.

8. In the data table, right-click in a cell in the Indicator column where the response is missing and select **Select Matching Cells**.
9. In the Rows panel, right-click **Selected** and select **Data View**.

Figure 4.4 Choice Sets with No Choice Responses

	Gender	Subject	Trial	Profile Name	Indicator	Crust	Cheese	Topping
1	F	2	4	TrimPepperjack	•	Thin	Jack	Pepperoni
2	F	2	4	TrimOni	•	Thin	Mozzarella	Pepperoni
3	M	7	2	Trimella	•	Thin	Mozzarella	None
4	M	7	2	TrimJack	•	Thin	Jack	None
5	M	7	3	Trimella	•	Thin	Mozzarella	None
6	M	7	3	TrimJack	•	Thin	Jack	None
7	F	8	2	ThickElla	•	Thick	Mozzarella	None
8	F	8	2	ThickJack	•	Thick	Jack	None
9	M	11	4	ThickOni	•	Thick	Mozzarella	Pepperoni
10	M	11	4	ThickJackoni	•	Thick	Jack	Pepperoni
11	F	14	2	TrimOni	•	Thin	Mozzarella	Pepperoni
12	F	14	2	TrimPepperjack	•	Thin	Jack	Pepperoni
13	F	18	3	ThickJack	•	Thick	Jack	None
14	F	18	3	ThickElla	•	Thick	Mozzarella	None
15	F	24	3	ThickJack	•	Thick	Jack	None
16	F	24	3	TrimOni	•	Thin	Mozzarella	Pepperoni
17	M	29	1	TrimPepperjack	•	Thin	Jack	Pepperoni
18	M	29	1	ThickJackoni	•	Thick	Jack	Pepperoni

In the table in Figure 4.4, consider the profiles in the first seven choice sets, which are defined by the Subject and Trial combinations in rows 1 to 14. The only difference within each choice set is the Cheese. There is an indication that some respondents might not be able to detect the difference in cheeses. However, the analysis takes the No Choice Indicator into account and concludes that, despite this behavior, Cheese is significant.

To see how to further analyze data of this type, see "Find Optimal Profiles".

Multiple Table Format

In this example, you examine pizza choices where three attributes, with two levels each, are presented to the respondents. The study was designed such that the respondents had to make a choice. The analysis uses three data tables: Pizza Profiles.jmp, Pizza Responses.jmp, and Pizza Subjects.jmp. Although you can always arrange your data into a single table, a multi-table approach can be more convenient than a one table analysis when you have additional profile and subject variables that you want to include in your analysis.

1. Select **Help > Sample Data Library** and open Pizza Profiles.jmp, Pizza Responses.jmp, and Pizza Subjects.jmp.
   - The profile data table, Pizza Profiles.jmp, lists all the pizza choice combinations that you want to present to the subjects. Each choice combination is given an ID.
   - The responses data table, Pizza Responses.jmp, contains the design and results. For the experiment, each subject is given four choice sets, where each choice set consists of two

choice profiles (Choice1 and Choice2). The subject selects a preference (Choice) for each choice set. For information about how to construct a choice design, see the *Design of Experiments Guide*. Notice that each value in the Choice column is an ID value in the Profile data table that contains the attribute information.

– The subjects data table, Pizza Subjects.jmp, includes a Subject ID column and a single characteristic of the subject, Gender. Each value of Subject in the Pizza Subjects.jmp data table corresponds to values in the Subject column in the Pizza Responses.jmp data table.

2. Select **Analyze > Consumer Research > Choice** to open the launch window.

Note: This can be done from any of the three open data tables.

3. From the Data Format menu, select **Multiple Tables, Cross-Referenced**.

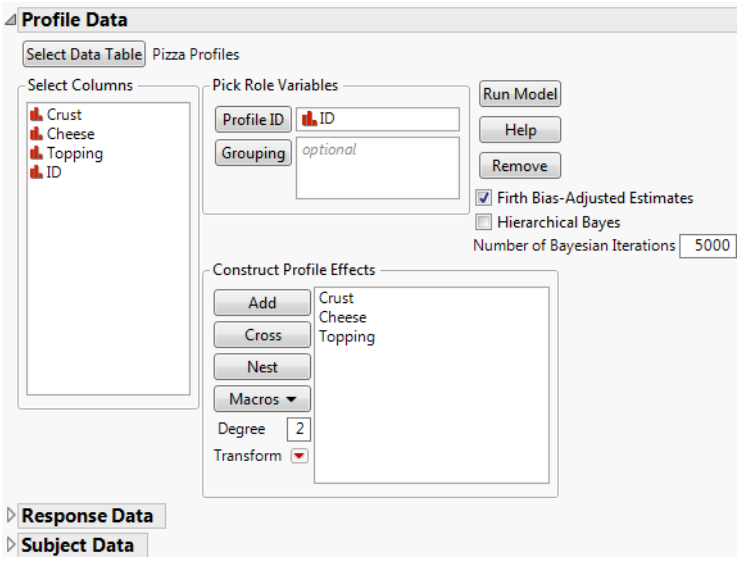
   There are three separate sections, one for each of the data sources.

4. Click **Select Data Table** under Profile Data.

   A Profile Data Table window appears, which prompts you to specify the data table for the profile data.

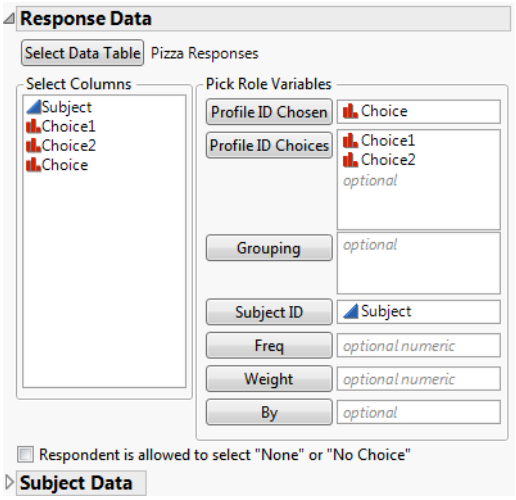
5. Select Pizza Profiles and click **OK**.
6. Select ID and click **Profile ID**.
7. Select Crust, Cheese, and Topping and click **Add**.

Figure 4.5 Profile Data



8. Click the disclosure icon next to Response Data to open the outline and click **Select Data Table**.
9. Select Pizza Responses and click **OK**.
10. Enter Profile ID selections:
    - Select Choice and click **Profile ID Chosen**.
    - Select Choice1 and Choice2 and click **Profile ID Choices**.
    - Select Subject and select **Subject ID**.

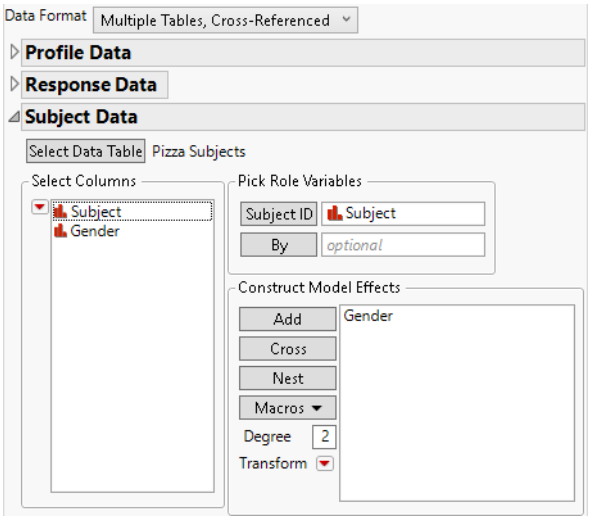
Figure 4.6 Response Data Window



Choice1 and Choice2 are the profiles presented to a subject in each of four choice sets. The Choice column contains the chosen preference between Choice1 and Choice2.

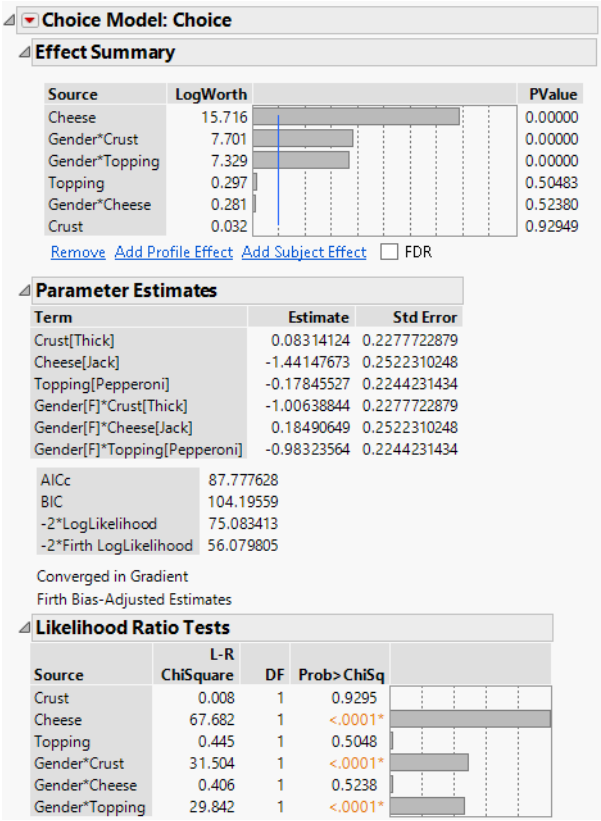
11. Click the disclosure icon next to Subject Data to open the outline and click **Select Data Table**.
12. Select Pizza Subjects and click **OK**.
13. Select Subject and click **Subject ID**.
14. Select Gender and click **Add**.

Figure 4.7 Subject Data Window



15. Click **Run Model**.

Figure 4.8 Choice Model Results



Choice Model: Choice

Effect Summary

Source	LogWorth	PValue
Cheese	15.716	0.00000
Gender*Crust	7.701	0.00000
Gender*Topping	7.329	0.00000
Topping	0.297	0.50483
Gender*Cheese	0.281	0.52380
Crust	0.032	0.92949

Remove Add Profile Effect Add Subject Effect FDR

Parameter Estimates

Term	Estimate	Std Error
Crust[Thick]	0.08314124	0.2277722879
Cheese[Jack]	-1.44147673	0.2522310248
Topping[Pepperoni]	-0.17845527	0.2244231434
Gender[F]*Crust[Thick]	-1.00638844	0.2277722879
Gender[F]*Cheese[Jack]	0.18490649	0.2522310248
Gender[F]*Topping[Pepperoni]	-0.98323564	0.2244231434

AICc	87.777628
BIC	104.19559
-2*LogLikelihood	75.083413
-2*Firth LogLikelihood	56.079805

Converged in Gradient
Firth Bias-Adjusted Estimates

Likelihood Ratio Tests

Source	L-R ChiSquare	DF	Prob>ChiSq
Crust	0.008	1	0.9295
Cheese	67.682	1	<.0001*
Topping	0.445	1	0.5048
Gender*Crust	31.504	1	<.0001*
Gender*Cheese	0.406	1	0.5238
Gender*Topping	29.842	1	<.0001*

Six effects are entered into the model. The effects Crust, Cheese, and Topping are product attributes. The interaction effects, Gender*Crust, Gender*Cheese, and Gender*Topping are subject-effect interactions with the attributes. These interaction effects enable you to construct products that meet market-segment preferences.

Note: For Choice models, subject effects cannot be entered as main effects. They appear only as interaction terms.

The Effect Summary and Likelihood Ratio Tests reports show strong interactions between Gender and Crust and between Gender and Topping. Notice that the main effects of Crust and Topping are not significant. If you had not included subject-level effects, you might have overlooked important information relative to market segmentation.

Find Optimal Profiles

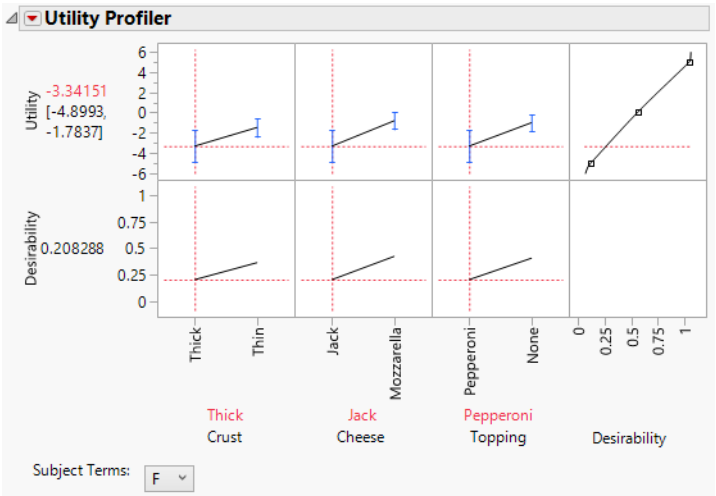
Next, you use the Utility Profiler to explore your results and find optimal settings for the attributes.

1. Click the Choice Model: Choice red triangle and select **Utility Profiler**.

   The Subject Terms menu beneath the profiler indicates that it is showing results for females.

2. Click the Utility Profiler red triangle and select **Optimization and Desirability > Desirability Functions**.

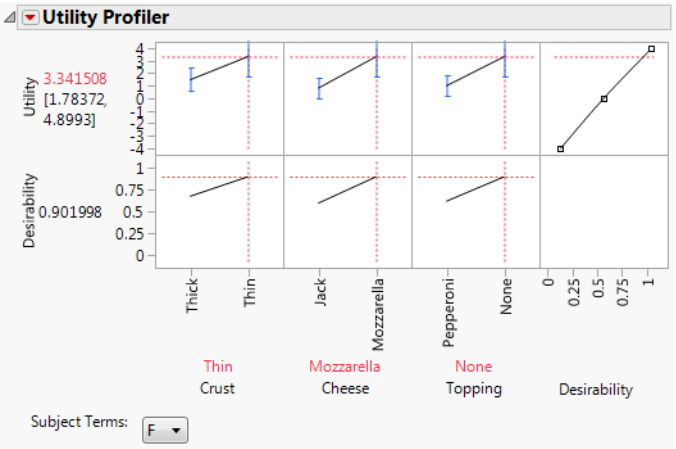
Figure 4.9 Utility Profiler with Desirability Function



   A desirability function that maximizes utility is added to the profiler. See *Profilers*.

3. Click the Utility Profiler red triangle and select **Optimization and Desirability > Maximize Desirability**.

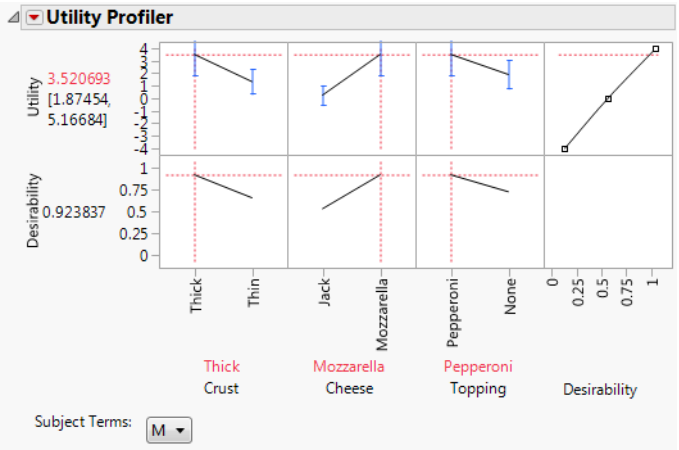
Figure 4.10 Utility Profiler with Optimal Settings for Females



The optimal settings for females are a thin crust, Mozzarella cheese, and no topping.

4. From the Subject Terms menu, select **M**.
5. Click the Utility Profiler red triangle and select **Optimization and Desirability > Maximize Desirability.**

Figure 4.11 Utility Profiler with Male Level Factor Setting



The optimal settings for males are a thick crust, Mozzarella cheese, and a Pepperoni topping.

In this example, understanding the preferences of gender-defined market segments enables you to provide two pizza choices that appeal to two segments of customers.

Launch the Choice Platform

Launch the Choice platform by selecting **Analyze > Consumer Research > Choice**.

Your data for the Choice platform can be combined in a single data table or it can reside in two or three separate data tables. In the Choice launch window specify whether you are using one or multiple data tables in the Data Format list.

One Table, Stacked

For the One Table, Stacked format, the data are in a single data table. There is a row for every profile presented to a subject and an indicator of whether that profile was selected. The Pizza Combined No Choice.jmp sample data table contains the results of a choice experiment in a single table format. See "One Table Format with No Choice".

For more information about the launch window for this format, see "Launch Window for One Table, Stacked".

Multiple Tables, Cross-Referenced

For the Multiple Tables, Cross-Referenced format, the data are in two or three separate data tables. A profile data table and a response data table are required. A subject data table is optional. Note the following:

- The profile data table must contain a column with a unique identifier for each profile and columns for the profile level variables. The profile identifier is used in the response data table to identify the profiles presented and the profile selected.
- The optional subject data table must contain a column with a unique subject identifier for each subject and columns for the subject level variables. The subject identifier is used in the response table to identify the subjects.

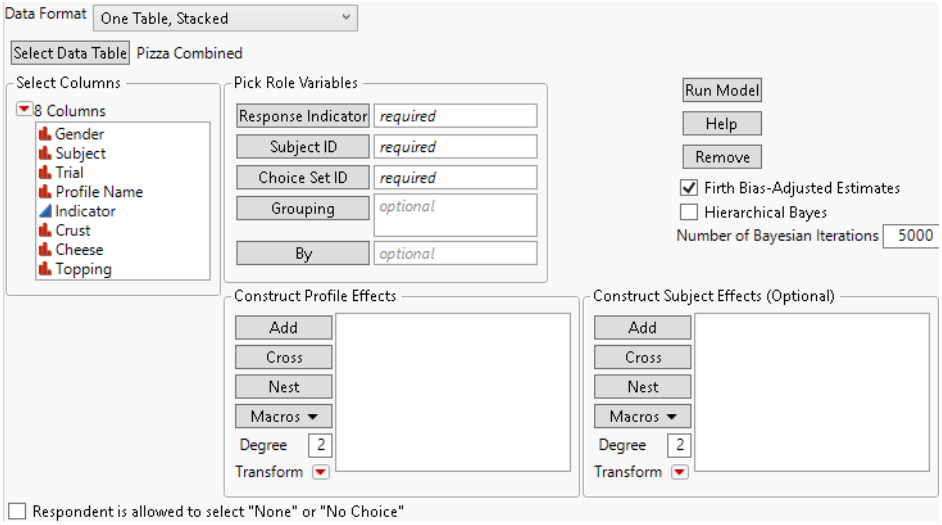
The launch window for this format contains three sections: Profile Data, Response Data, and Subject Data. Each section corresponds to a different data table. You can expand or collapse each section as needed.

The Pizza Profiles.jmp, Pizza Responses.jmp, and Pizza Subjects.jmp sample data tables contain the results of a choice experiment using three tables. There is one table for the profiles, one for the responses, and one for the subject information. See "Multiple Table Format".

For more information about the launch window for this format, see "Launch Window for Multiple Tables, Cross-Referenced".

Launch Window for One Table, Stacked

Figure 4.12 Launch Window for One Table, Stacked Data Format



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Select Data Table Select or open the data table that contains the combined data. Select Other to open a file that is not already open.

Response Indicator A column that contains values that indicate the preferred choice. A 1 indicates the preferred profile and a 0 indicates the other profiles. If respondents are given an option to select no preference, enter missing values for choice sets where no preference is indicated. See "Respondent is allowed to select "None" or "No Choice"".

Subject ID An identifier for the study participant.

Choice Set ID An identifier for the choice set presented to the subject for a given preference determination.

Grouping A column which, when used with the Choice Set ID column, uniquely designates each choice set. For example, if a choice set has Choice Set ID = 1 for Survey = A, and another choice set has Choice Set ID = 1 for Survey = B, then Survey should be used as a Grouping column.

By Produces a separate report for each level of the By Variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Construct Profile Effects Add effects constructed from the attributes in the profiles.

For information about the Construct Profile Effects panel, see *Fitting Linear Models*.

Note: The choice model observes the column coding property of continuous profile and subject effects.

Construct Subject Effects (Optional) Add effects constructed from subject-related factors.

For information about the Construct Subject Effects panel, see *Fitting Linear Models*.

Firth Bias-adjusted Estimates Computes bias-corrected MLEs that produce better estimates and tests than MLEs without bias correction. These estimates also improve separation problems that tend to occur in logistic-type models. See Heinze and Schemper (2002) for a discussion of the separation problem in logistic regression.

JMP PRO **Hierarchical Bayes** Uses a Bayesian approach to estimate subject-specific parameters. See "Bayesian Parameter Estimates".

JMP PRO **Number of Bayesian Iterations** (Applicable only if Hierarchical Bayes is selected.) The total number of iterations of the adaptive Bayes algorithm used to estimate subject-specific parameters. This number includes a burn-in period of iterations that are discarded. The number of burn-in iterations is equal to half of the Number of Bayesian Iterations specified on the launch window.

Respondent is allowed to select "None" or "No Choice" Enters a No Choice Indicator into the model for response rows containing missing values. For the One Table, Stacked data format, the No Choice rows must contain (numeric) missing values in the Response Indicator column. The option appears at the bottom of the launch window.

Launch Window for Multiple Tables, Cross-Referenced

Figure 4.13 Launch Window for Multiple Tables, Cross-Referenced Data Format

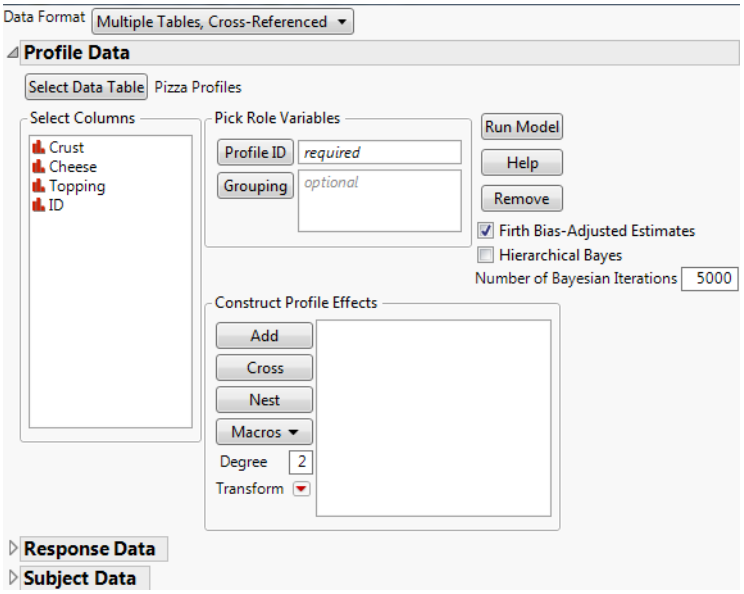


Figure 4.13 shows the launch window for Multiple Tables, using Pizza Profiles.jmp as the Profile table.

In the case of Multiple Tables, Cross-referenced, the launch window has three sections:

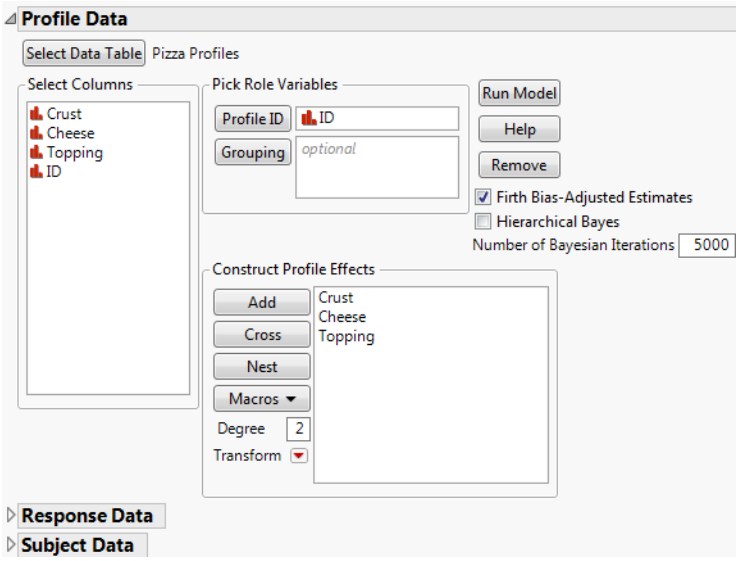
- "Profile Data"
- "Response Data"
- "Subject Data"

Profile Data

The profile data table describes the attributes associated with each choice. Each attribute defines a column in the data table. There is a row for each profile. A column in the table contains a unique identifier for each profile. Figure 4.14 shows the Pizza Profiles.jmp data table and a completed Profile Data panel.

Figure 4.14 Profile Data Table and Completed Profile Data Outline

	Crust	Cheese	Topping	ID
1	Thick	Mozzarella	Pepperoni	ThickOni
2	Thick	Mozzarella	None	ThickElla
3	Thick	Jack	Pepperoni	ThickJackoni
4	Thick	Jack	None	ThickJack
5	Thin	Mozzarella	Pepperoni	TrimOni
6	Thin	Mozzarella	None	Trimella
7	Thin	Jack	Pepperoni	TrimPepperjack
8	Thin	Jack	None	TrimJack



Select Data Table Select or open the data table that contains the profile data. Select Other to open a file that is not already open.

Profile ID Identifier for each row of attribute combinations (profile). If the **Profile ID** column does not uniquely identify each row in the profile data table, you need to add **Grouping** columns. Add **Grouping** columns until the combination of **Grouping** and **Profile ID** columns uniquely identify the row, or profile.

Grouping A column which, when used with the Profile ID column, uniquely designates each choice set. For example, if Profile ID = 1 for Survey = A, and a different Profile ID = 1 for Survey = B, then Survey would be used as a **Grouping** column.

Construct Profile Effects Add effects constructed from the attributes in the profiles.

For information about the Construct Profile Effects panel, see *Fitting Linear Models*.

Note: The choice model observes the column coding property of continuous profile and subject effects.

Firth Bias-adjusted Estimates Computes bias-corrected MLEs that produce better estimates and tests than MLEs without bias correction. These estimates also improve separation problems that tend to occur in logistic-type models. See Heinze and Schemper (2002) for a discussion of the separation problem in logistic regression.

JMP PRO **Hierarchical Bayes** Uses a Bayesian approach to estimate subject-specific parameters. See "Bayesian Parameter Estimates".

JMP PRO **Number of Bayesian Iterations** (Applicable only if Hierarchical Bayes is selected.) The total number of iterations of the adaptive Bayes algorithm used to estimate subject effects. This number includes a burn-in period of iterations that are discarded. The number of burn-in iterations is equal to half of the Number of Bayesian Iterations specified on the launch window.

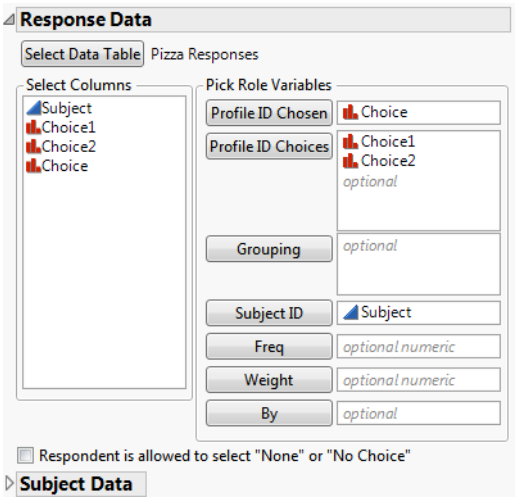
Response Data

The response data table includes a subject identifier column, columns that list the profile identifiers for the profiles in each choice set, and a column containing the preferred profile identifier. There is a row for each subject and choice set. Grouping variables can be used to distinguish choice sets when the data contain more than one group of choice sets. Figure 4.15 shows the Pizza Responses.jmp data table and a completed Response Data panel.

Grouping variables can be used to align choice indices when more than one group is contained within the data.

Figure 4.15 Response Data Table and Completed Responses Data Outline

	Subject	Choice1	Choice2	Choice
1	1	ThickJack	TrimPepperjack	TrimPepperjack
2	1	TrimPepperjack	ThickElla	ThickElla
3	1	TrimOni	Trimella	TrimOni
4	1	ThickElla	ThickJack	ThickElla
5	2	Trimella	ThickJackoni	Trimella
6	2	TrimJack	ThickElla	ThickElla
7	2	Trimella	TrimPepperjack	Trimella
8	2	TrimPepperjack	TrimOni	TrimOni
9	3	TrimOni	ThickJackoni	TrimOni
10	3	TrimPepperjack	ThickElla	ThickElla
11	3	ThickJackoni	TrimPepperjack	ThickJackoni
12	3	ThickOni	Trimella	ThickOni
13	4	ThickElla	ThickOni	ThickElla
14	4	TrimPepperjack	ThickJack	ThickJack



Select Data Table Select or open the data table that contains the response data. Select Other to open a file that is not already open.

Profile ID Chosen The Profile ID from the Profile data table that represents the subject's selected profile.

Grouping A column which, when used with the Profile ID Chosen column, uniquely designates each choice set.

Profile ID Choices The Profile IDs of the set of possible profiles. There must be at least two profiles.

Subject ID An identifier for the study participant.

Freq A column containing frequencies. If *n* is the value of the Freq variable for a given row, then that row is used in computations n times. If it is less than 1 or missing, then JMP does not use it to calculate any analyses.

Weight A column containing a weight for each observation in the data table. The weight is included in analyses only when its value is greater than zero.

By Produces a separate report for each level of the By Variable. If more than one By variable is assigned, a separate analysis is produced for each possible combination of the levels of the By variables.

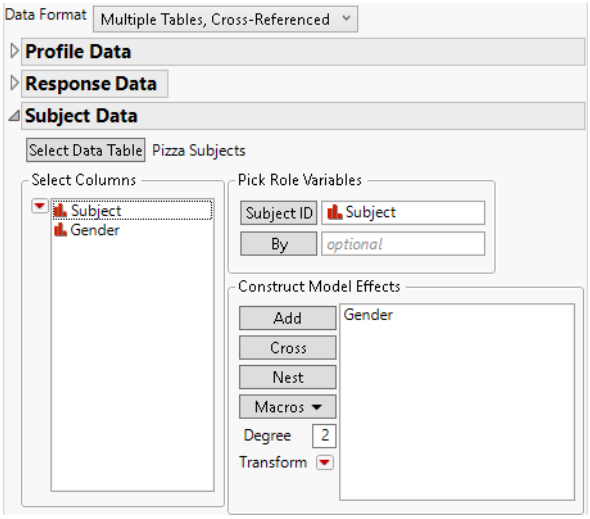
Respondent is allowed to select "None" or "No Choice" Enters a No Choice Indicator into the model for response rows containing missing values. For the Multiple Tables, Cross-Referenced data format, the No Choice rows must contain (categorical) missing values in the Profile ID Chosen column in the Response Data table. The option appears at the bottom of the Response Data panel.

Subject Data

The subject data table is optional and depends on whether you want to model subject effects. The table contains a column with the subject identifier used in the response table, and columns for attributes or characteristics of the subjects. You can put subject data in the response data table, but you should specify the subject effects in the Subject Data outline. Figure 4.16 shows the Pizza Subjects.jmp data table and a completed Subject Data panel.

Figure 4.16 Subject Data Table and Completed Subject Data Outline

	Subject	Gender
1	1	M
2	2	F
3	3	M
4	4	F



Select Data Table Select or open the data table that contains the subject data. Select Other to open a file that is not already open.

Subject ID Unique identifier for the subject.

By Produces a separate report for each level of the By variable. If more than on By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Construct Model Effects Add effects constructed from columns in the subject data table.

For information about the Construct Model Effects panel, see *Fitting Linear Models*.

Choice Model Report

- "Effect Summary"
- "Parameter Estimates"
- "Likelihood Ratio Tests"
- "Bayesian Parameter Estimates"

Effect Summary

The Effect Summary report appears if your model contains more than one effect and if it can be calculated quickly. (If the report does not appear, select Likelihood Ratio Tests from the red triangle menu to make both reports appear.) It lists the effects estimated by the model and gives a plot of the LogWorth (or FDR LogWorth) values for these effects. The report also provides controls that enable you to add or remove effects from the model. The model fit report updates automatically based on the changes made in the Effects Summary report. See *Fitting Linear Models*.

Note: The Effect Summary report is not applicable to models fit with Hierarchical Bayes.

Effect Summary Table Columns

The Effect Summary table contains the following columns:

Source Lists the model effects, sorted by ascending *p*-values.

LogWorth Shows the LogWorth for each model effect, defined as $-\log_{10}(p\text{-value})$. This transformation adjusts *p*-values to provide an appropriate scale for graphing. A value that exceeds 2 is significant at the 0.01 level (because $-\log_{10}(0.01) = 2$).

FDR LogWorth Shows the False Discovery Rate LogWorth for each model effect, defined as $-\log_{10}(\text{FDR PValue})$. This is the best statistic for plotting and assessing significance. Select the **FDR** check box to replace the LogWorth column with the **FDR LogWorth** column.

Bar Chart Shows a bar chart of the LogWorth (or FDR LogWorth) values. The graph has dashed vertical lines at integer values and a blue reference line at 2.

PValue Shows the *p*-value for each model effect. This is the *p*-value corresponding to the significance test displayed in the Likelihood Ratio Tests report.

FDR PValue Shows the False Discovery Rate *p*-value for each model effect calculated using the Benjamini-Hochberg technique. This technique adjusts the *p*-values to control the false

discovery rate for multiple tests. Select the **FDR** check box to replace the **PValue** column with the **FDR PValue** column.

For more information about the FDR correction, see Benjamini and Hochberg (1995). For more information about the false discovery rate, see *Predictive and Specialized Modeling* or Westfall et al. (2011).

Effect Summary Table Options

The options below the summary table enable you to add and remove effects:

Remove Removes the selected effects from the model. To remove one or more effects, select the rows corresponding to the effects and click the Remove button.

Add Profile Effect Opens a panel that contains a list of all columns in the data table for the OneTable, Stacked data format, and for the columns in the Profile Data table for the Multiple Tables, Cross-Referenced data format. Select columns that you want to add to the model, and then click Add below the column selection list to add the columns to the model. Click Close to close the panel.

Add Subject Effect Opens a panel that contains a list of all columns in the data table for the OneTable, Stacked data format, and for the columns in the Subject Data table for the Multiple Tables, Cross-Referenced data format. Select columns that you want to add to the model, and then click Add below the column selection list to add the columns to the model. Click Close to close the panel.

Parameter Estimates

The Parameter Estimates report gives estimates and standard errors of the coefficients of utility associated with the effects listed in the Term column. The coefficients associated with attributes are sometimes referred to as *part-worths*. When the Firth Bias-Adjusted Estimates option is selected in the launch window, the parameter estimates are based on the Firth bias-corrected maximum likelihood estimators. These estimates considered to be more accurate than MLEs without bias correction. For more information about utility, see "Utility and Probabilities".

Comparison Criteria

The AICc (corrected Akaike's Information Criterion), BIC (Bayesian Information Criterion), –2Loglikelihood, and –2Firth Loglikelihood fit statistics are shown as part of the report and can be used to compare models. See *Fitting Linear Models*.

The –2Firth Loglikelihood fit statistic is included in the report when the Firth Bias-Adjusted Estimates option is selected in the launch window. Note that this option is checked by default. The decision to use or not use the Firth Bias-Adjusted Estimates does not affect the AICc score or the –2Loglikelihood results.

Note: For each of these statistics, a smaller value indicates a better fit.

Likelihood Ratio Tests

The Likelihood Ratio Test report appears by default if the model is fit in less than five seconds. If the report does not appear, you can select the Likelihood Ratio Tests option from the Choice Model red triangle menu. The report gives the following:

Source Lists the effects in the model.

L-R ChiSquare The value of the likelihood ratio ChiSquare statistic for a test of the corresponding effect.

DF The degrees of freedom for the ChiSquare test.

Prob>ChiSq The *p*-value for the ChiSquare test.

Bar Graph Shows a bar chart of the L-R ChiSquare values.

JMP PRO Bayesian Parameter Estimates

(Available only for Hierarchical Bayes.) The Bayesian Parameter Estimates report gives results for model effects. The estimates are based on a Hierarchical Bayes fit that integrates the subject-level covariates into the likelihood function and estimates their effects on the parameters directly. The subject-level covariates are estimated using a Bayesian procedure combined with the Metropolis-Hastings algorithm. See Train (2001). Posterior means and variances are calculated for each model effect. The algorithm also provides subject-specific estimates of the model effect parameters. See "Save Subject Estimates".

During the estimation process, each individual is assigned his or her own vector of parameter estimates, essentially treating the estimates as random effects and covariates. The vector of coefficients for an individual is assumed to come from a multivariate normal distribution with arbitrary mean and covariance matrix. The likelihood function for the utility parameters for a given subject is based on a multinominal logit model for each subject's preference within a choice set, given the attributes in the choice set. The prior distribution for a given subject's vector of coefficients is normal with mean equal to zero and a diagonal covariance matrix with the same variance for each subject. The covariance matrix is assumed to come from an inverse Wishart distribution with a scale matrix that is diagonal with equal diagonal entries.

For each subject, a number of burn-in iterations at the beginning of the chain is discarded. By default, this number is equal to half of the Number of Bayesian Iterations specified on the launch window.

Figure 4.17 Bayesian Parameter Estimates Report

Choice Model: Choice

Bayesian Parameter Estimates

Term	Posterior Mean	Posterior Std Dev	Subject Std Dev
Crust[Thick]	0.27529781	0.724891965	2.815958945
Cheese[Jack]	-7.01110292	4.697169974	2.593184268
Topping[Pepperoni]	-1.06702410	0.941602303	2.260232580

Total Iterations	5000
Burn-In Iterations	2500
Number of Respondents	32
Avg Log Likelihood After Burn-In	-14.19342

Term The model term.

Posterior Mean The parameter estimate for the term's coefficient. For each iteration after the burn-in period, the mean of the subject-specific coefficient estimates is computed. The Posterior Mean is the average of these means.

Tip: Select the red triangle option Save Bayes Chain to see the individual estimates for each iteration.

Posterior Std Dev The standard deviation of the means of the subject-specific estimates over the iterations after burn-in.

Subject Std Dev The standard deviation of the subject-specific estimates.

Tip: Select the red triangle option Save Subject Estimates to see the individual estimates.

Total Iterations The total number of iterations performed, including the burn-in period.

Burn-In Iterations The number of burn-in iterations. This number is equal to half of the Number of Bayesian Iterations specified on the launch window.

Number of Respondents The number of subjects.

Avg Log Likelihood After Burn-In The average of the log-likelihood function, computed on values obtained after the burn-in period.

Choice Platform Options

Choice Model red triangle menu contains the following options.

Note: When you use Hierarchical Bayes, the subject-level estimates are based on Monte Carlo sampling. For this reason, results obtained for the options below vary from run to run.

Likelihood Ratio Tests See "Likelihood Ratio Tests".

JMP PRO **Show MLE Parameter Estimates** (Available for Hierarchical Bayes) Shows non-Firth maximum likelihood estimates and standard errors for the coefficients of model terms. These estimates are used as starting values for the Hierarchical Bayes algorithm.

Joint Factor Tests (Not available for Hierarchical Bayes) Tests each factor in the model by constructing a likelihood ratio test for all the effects involving that factor. For more information about Joint Factor Tests, see *Fitting Linear Models*.

Confidence Intervals (Not available for Hierarchical Bayes) Shows or hides a confidence interval for each parameter in the Parameter Estimates report.

Confidence Limits (Available for Hierarchical Bayes) Shows or hides confidence limits for each parameter in the Bayesian Parameter Estimates report. The limits are constructed based on the 2.5 and 97.5 quantiles of the posterior distribution.

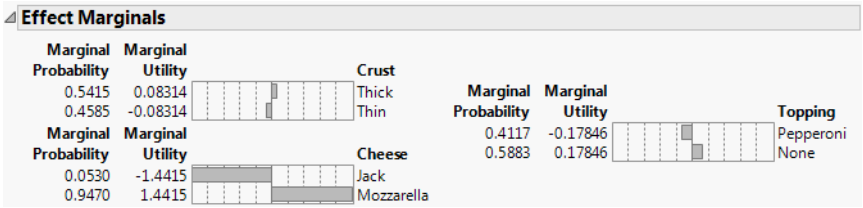
Correlation of Estimates If Hierarchical Bayes was not selected, shows the correlations between the maximum likelihood parameter estimates.

For Hierarchical Bayes, shows the correlation matrix for the posterior means of the parameter estimates. The correlations are calculated from the iterations after burn-in. The posterior means from each iteration after burn-in are treated as if they are columns in a data table. The Correlation of Estimates table is obtained by calculating the correlation matrix for these columns.

Effect Marginals Shows or hides marginal probabilities and marginal utilities for each main effect in the model. The marginal probability is the probability that an individual selects attribute A over B with all other attributes set to their mean or default levels.

In Figure 4.18, the marginal probability of any subject choosing a pizza with mozzarella cheese, thick crust and pepperoni, over that same pizza with Monterey Jack cheese instead of mozzarella, is 0.9470.

Figure 4.18 Example of Marginal Effects



Utility Profiler Shows or hides the predicted utility for different factor settings. The utility is the value predicted by the linear model. See "Find Optimal Profiles" for an example of the Utility Profiler. For more information about utility, see "Utility and Probabilities". For more information about the Utility Profiler options, see *Profilers*.

Probability Profiler Enables you to compare choice probabilities among a number of potential products. This predicted probability is defined as follows:

$$(\exp(U))/(\exp(U) + \exp(U_b))$$

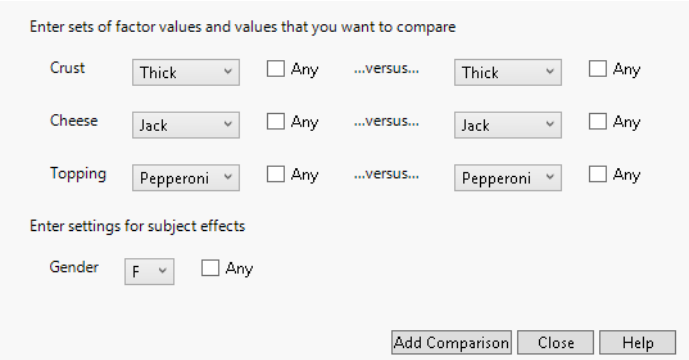
where U is the utility for the current settings and U_b is the utility for the baseline settings. This implies that the probability for the baseline settings is 0.5. See "Utility and Probabilities".

See "Comparisons to Baseline" for an example of using the Probability Profiler. For more information about the Probability Profiler options, see *Profilers*.

Multiple Choice Profiler Provides the number of probability profilers that you specify. This enables you to set each profiler to the settings of a given profile so that you can compare the probabilities of choosing each profile relative to the others. See "Multiple Choice Comparisons" for an example of using the Multiple Choice Profiler. For more information about the Multiple Choice Profiler options, see *Profilers*.

Comparisons Performs comparisons between specific alternative choice profiles. Enables you to select the factors and the values that you want to compare. You can compare specific configurations, including comparing all settings on the left or right by selecting the **Any** check boxes. If you have subject effects, you can select the levels of the subject effects to compare. Using Any does not compare all combinations across features, but rather all combinations of comparisons, one feature at a time, using the left settings as the settings for the other factors.

Figure 4.19 Utility Comparisons Window



Willingness to Pay Requires that your model includes a continuous price column. Calculates the maximum price increase (decrease) that a customer is willing to pay for a new feature over the baseline feature cost. The result is calculated using the Baseline settings for each background setting.

Save Utility Formula When the analysis is on multiple data tables, creates a new data table that contains a formula column for utility. The new data table contains a row for each subject and profile combination, and columns for the profiles and the subject effects. When the analysis is on one data table, a new Utility Formula column is added.

Save Gradients by Subject (Not available for Hierarchical Bayes.) Constructs a new table that has a row for each subject containing the average (Hessian-scaled-gradient) steps for the likelihood function on each parameter. This corresponds to using a Lagrangian multiplier test for separating that subject from the remaining subjects. These values can later be clustered, using the built-in-script, to indicate unique market segments represented in the data. See "Gradients". For an example, see "Example of Segmentation".

JMP PRO **Save Subject Estimates** (Available for Hierarchical Bayes.) Creates a table where each row contains the subject-specific parameter estimates for each effect. The distribution of subject-specific parameter effects for each effect is centered at the estimate for the term given in the Bayesian Parameter Estimates report. The Subject Acceptance Rate gives the rate of acceptance for draws of new parameter estimates during the Metropolis-Hastings step. Generally, an acceptance rate of 0.20 is considered to be good. See "Bayesian Parameter Estimates".

JMP PRO **Save Bayes Chain** (Available for Hierarchical Bayes.) Creates a table that gives information about the chain of iterations used in computing subject-specific Bayesian estimates. See "Save Bayes Chain".

Model Dialog Shows the Choice launch window, which can be used to modify and re-fit the model. You can specify new data sets, new IDs, and new model effects.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Willingness to Pay

The term *willingness to pay* refers to the price that a customer is willing to pay for new features, calculated to match a customer's utility for baseline features. For example, suppose that a customer is willing to pay $1,000 for a computer with a 40 GB hard drive. Willingness to Pay for an 80 GB hard drive is calculated by setting the Hard drive feature to 80 GB and then solving for the price that delivers the same utility as the $1000 40 GB hard drive.

Willingness to Pay Launch Window Options

When you select the Willingness to Pay option, the Willingness to Pay launch window is shown. The launch window in Figure 4.20 is obtained by selecting the Willingness to Pay option in the report that results from running the **Choice** data table script in Laptop Profile.jmp.

Factor The variables from the analysis. These can be product features or subject-specific attributes.

Baseline The baseline setting for each factor. If the factor is categorical, select the baseline value from a list. If the factor is numeric, enter the baseline value.

Role The type of factor.

Feature Factor A product or service feature from the experiment that you want to price.

Price Factor A price factor in the experiment. The price factor must be continuous, and there can be only one specified price factor for each Willingness to Pay analysis.

Background Constant A factor that you want to hold constant in the Willingness to Pay calculation. Generally, these are subject-specific variables.

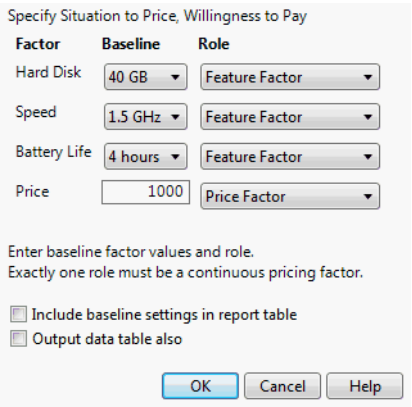
Background Variable A factor that you want to hold constant, at each of its levels, in the Willingness to Pay calculation. Generally, these are subject-level factors. Specifying a subject factor as a Background Variable rather than a Background Constant provides Willingness to Pay estimates for all levels of the variable.

Include baseline settings in report table Adds the baseline settings with a price change of zero to the Willingness to Pay report.

Tip: If you make an output table, use this option to display all the baseline settings as well as the attribute settings.

Output data table also Creates a data table containing the Willingness to Pay report.

Figure 4.20 Willingness to Pay Launch Window



Once you complete your first Willingness to Pay calculation, the platform remembers the baseline values and assigned roles that you selected. This enables you to do multiple Willingness to Pay comparisons without having to re-enter the baseline information. If there is no factor called Price, but there is a continuous factor used in the analysis, the continuous factor is automatically assigned as the Price factor in the Willingness to Pay window. Common cost variables that are not prices in the traditional sense include factors such as travel time or distance.

Willingness to Pay Report

The Willingness to Pay report displays the baseline value for each factor, as well as baseline utility values. For each factor, the report shows the feature setting, estimated price change, and new price. If there are no interaction or second-order effects, standard errors and confidence intervals are also shown. These are calculated using the delta method.

Figure 4.21 Willingness to Pay Report

Willingness to Pay

Factor	Baseline Value
Hard Disk	40 GB
Speed	1.5 GHz
Battery Life	4 hours
Price	1000

Baseline Utility
-3.4736

Factor	Feature Setting	Price Change	Std Error	Lower 95%	Upper 95%	New Price
Hard Disk	80 GB	$959.67	381.743	$211.47	$1,707.87	$1,959.67
Speed	2.0 GHz	$496.86	218.515	$68.58	$925.14	$1,496.86
Battery Life	6 hours	$387.88	191.088	$13.35	$762.40	$1,387.88

Standard deviations for Price Change calculated by Delta method.

JMP PRO Save Bayes Chain

(Available for models fit with Hierarchal Bayes.) You can use the Bayes Chain data table to determine whether your estimates have stabilized. The table that is created has a number of rows equal to the Number of Bayesian Iterations (specified on the launch window) plus one. The first row, Iteration 1, gives the starting values. Subsequent rows show the results of the iterations, in order. The table has a column for the iteration counts, the model Log Likelihood, and columns corresponding to each model effect:

Iteration Gives the iteration number, where the first row shows starting values.

Log Likelihood The log-likelihood of the model for that iteration. You can plot the Log Likelihood against Iteration to view behavior over the burn-in and tuning periods.

Adaptive Sigma for <model effect> Gives the estimate of the square root of the diagonal entries of the inverse Wishart distribution scale matrix for the corresponding effect.

Acceptance for <model effect> Gives the sampling acceptance rate for the corresponding effect.

Mean of <model effect> Gives the estimated mean for the corresponding effect.

Variance of <model effect> Gives the estimated variance for the corresponding effect.

Additional Examples

- "Example of Making Design Decisions"
- "Example of Segmentation"
- "Example of Logistic Regression Using the Choice Platform"
- "Example of Logistic Regression for Matched Case-Control Studies"
- "Example of Transforming Data to Two Analysis Tables"
- "Example of Transforming Data to One Analysis Table"

Example of Making Design Decisions

You can use the Choice Modeling platform to determine the relative importance of product attributes. Even if the attributes of a particular product that are important to the consumer are known, information about preference trade-offs with regard to these attributes might be unknown. By gaining such information, a market researcher or product designer is able to incorporate product features that represent the optimal trade-off from the perspective of the consumer. This example illustrates the advantages of this approach to product design.

It is already known that four attributes are important for laptop design: hard-disk size, processor speed, battery life, and selling price. The data gathered for this study are used to determine which of four laptop attributes (Hard Disk, Speed, Battery Life, and Price) are most important. It also assesses whether there are Gender or Job effects associated with these attributes.

This example is described in four sections:

- "Complete the Launch Window"
- "Analyze the Model"
- "Comparisons to Baseline"
- "Multiple Choice Comparisons"

Complete the Launch Window

1. Select **Help > Sample Data Library** and open Laptop Runs.jmp.

Note: If you prefer not to follow the manual steps in this section, click the green triangle next to the script **Choice with Gender** to run the model, and go to "Analyze the Model".

2. Click the green triangle next to the **Open Profile and Subject Tables** script.

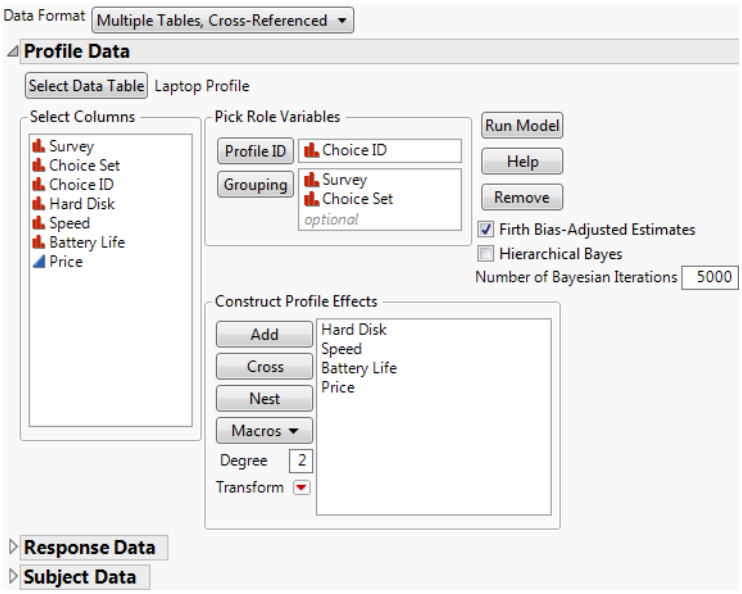
   The script opens the Laptop Profile.jmp and Laptop Subjects.jmp data tables.

3. Select **Analyze** > **Consumer Research** > **Choice**.

Note: This can be done from any of the three open data tables.

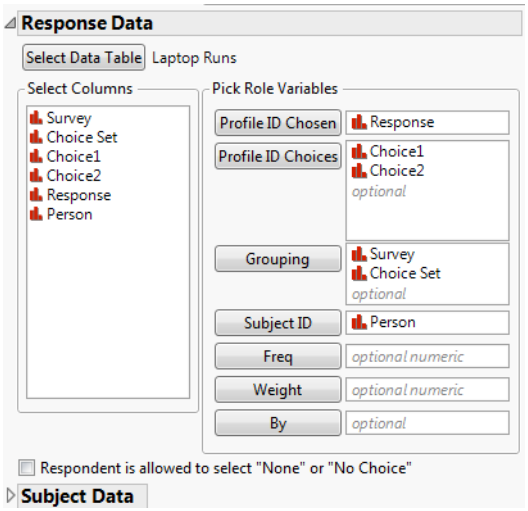
4. From the Data Format list, select **Multiple Tables, Cross-Referenced**.
5. Click **Select Data Table** under Profile Data and select Laptop Profile. Select Choice ID and click Profile ID.
6. Select Hard Disk, Speed, Battery Life, and Price and click **Add**.
7. Select Survey and Choice Set and click **Grouping**.

Figure 4.22 Profile Data Window for Laptop Study



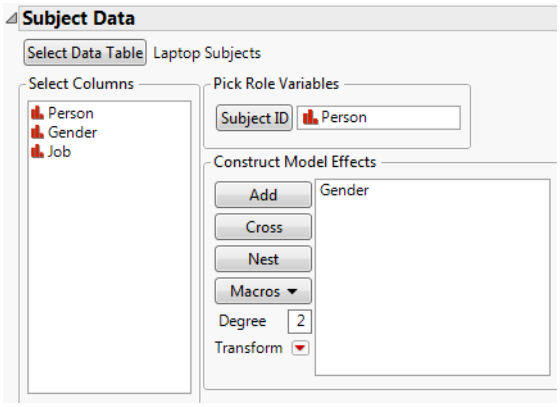
8. Open the **Response Data** outline.
9. From the **Select Data Table** list, select Laptop Runs.
10. Complete the Response Data table:
    - Select Response and click **Profile ID Chosen**.
    - Select Choice1 and Choice2 and click **Profile ID Choices**.
    - Select Survey and Choice Set and click **Grouping**
    - Select Person and click **Subject ID**.

Figure 4.23 Response Data Window for Laptop Study



11. Open the **Subject Data** outline.
12. From the **Select Data Table** list, select Laptop Subjects.
13. Select Person and click **Subject ID**.
14. Select Gender click **Add**.

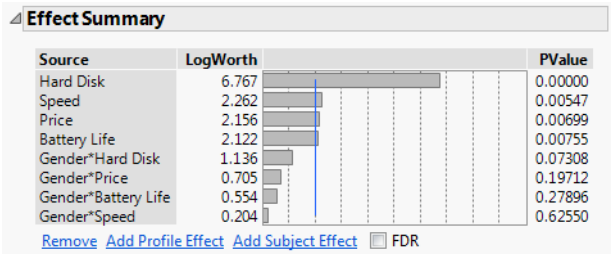
Figure 4.24 Subject Data Window for Laptop Study



Analyze the Model

1. Click **Run Model**.

Figure 4.25 Laptop Effect Summary



Effect Summary

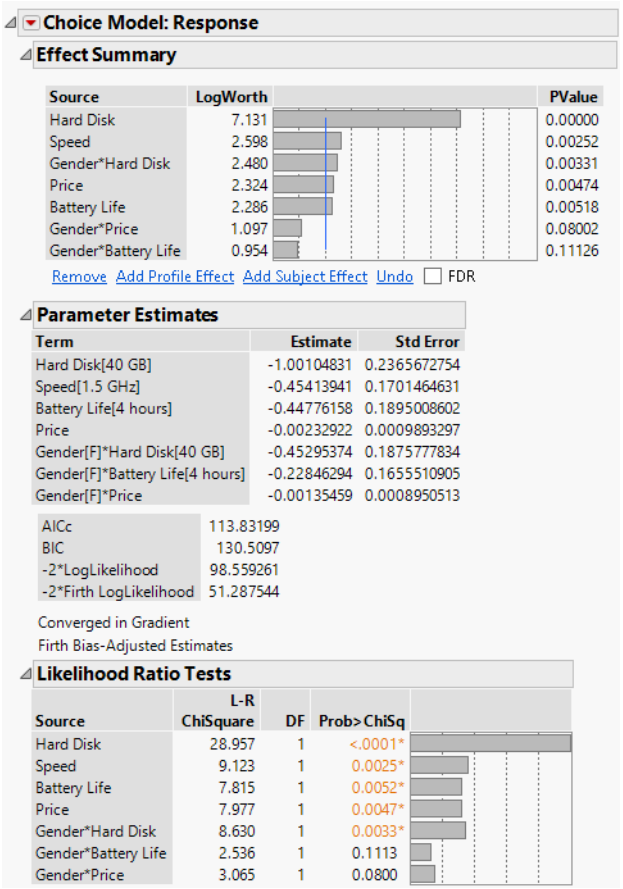
Source	LogWorth	PValue
Hard Disk	6.767	0.00000
Speed	2.262	0.00547
Price	2.156	0.00699
Battery Life	2.122	0.00755
Gender*Hard Disk	1.136	0.07308
Gender*Price	0.705	0.19712
Gender*Battery Life	0.554	0.27896
Gender*Speed	0.204	0.62550

Remove Add Profile Effect Add Subject Effect FDR

The Effect Summary report shows that Hard Disk is the most significant effect. You can reduce the model by removing terms with a *p*-value greater than 0.15. This process should be done one term at a time. Here, Gender*Speed is the least significant effect, with a *p*-value of 0.625.

2. In the Effect Summary report, select Gender*Speed and click **Remove**.

Figure 4.26 Laptop Results



Choice Model: Response

Effect Summary

Source	LogWorth	PValue
Hard Disk	7.131	0.00000
Speed	2.598	0.00252
Gender*Hard Disk	2.480	0.00331
Price	2.324	0.00474
Battery Life	2.286	0.00518
Gender*Price	1.097	0.08002
Gender*Battery Life	0.954	0.11126

Remove Add Profile Effect Add Subject Effect Undo FDR

Parameter Estimates

Term	Estimate	Std Error
Hard Disk[40 GB]	-1.00104831	0.2365672754
Speed[1.5 GHz]	-0.45413941	0.1701464631
Battery Life[4 hours]	-0.44776158	0.1895008602
Price	-0.00232922	0.0009893297
Gender[F]*Hard Disk[40 GB]	-0.45295374	0.1875777834
Gender[F]*Battery Life[4 hours]	-0.22846294	0.1655510905
Gender[F]*Price	-0.00135459	0.0008950513

AICc	113.83199
BIC	130.5097
-2*LogLikelihood	98.559261
-2*Firth LogLikelihood	51.287544

Converged in Gradient
Firth Bias-Adjusted Estimates

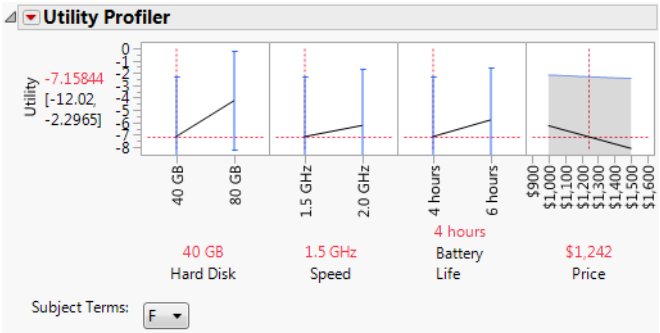
Likelihood Ratio Tests

Source	L-R ChiSquare	DF	Prob>ChiSq
Hard Disk	28.957	1	<.0001*
Speed	9.123	1	0.0025*
Battery Life	7.815	1	0.0052*
Price	7.977	1	0.0047*
Gender*Hard Disk	8.630	1	0.0033*
Gender*Battery Life	2.536	1	0.1113
Gender*Price	3.065	1	0.0800

Once Gender*Speed is removed from the model, all effects have a *p*-value of 0.15 or less. Therefore, you use this as your final model.

3. Click the Choice Model: Response red triangle and select **Utility Profiler**.

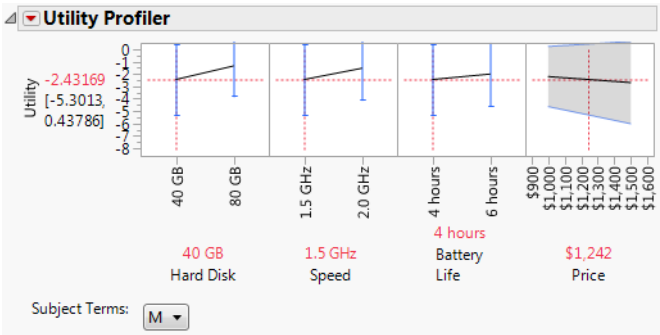
Figure 4.27 Laptop Profiler Results for Females



Tip: If your utility profiler does not look like Figure 4.27, click the Utility Profiler red triangle and select **Appearance > Adapt Y Axis**.

4. From the list next to Subject Terms, select **M**.

Figure 4.28 Laptop Profiler Results for Males in Development



The interaction effect between Gender and Hard Disk is highly significant, with a *p*-value of 0.0033 (Figure 4.26). In the Utility Profilers, check the slope for Hard Disk for both levels of Gender. You see that the slope is steeper for females than for males.

Comparisons to Baseline

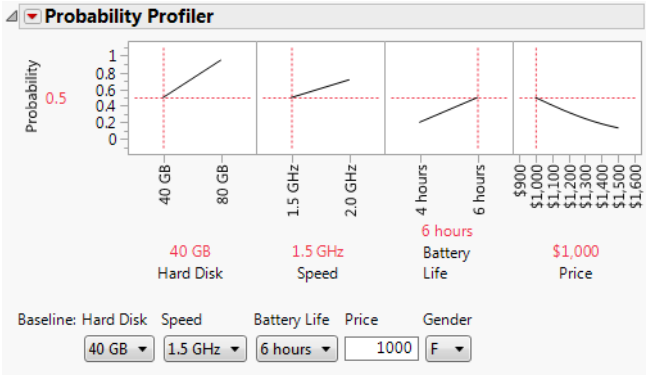
Suppose you are developing a new product. You want to explore the likelihood that a customer selects the new product over the old product, or over a competitor's product. Use the Probability Profiler to compare profiles to a baseline profile.

In this example, your company is currently producing laptops with 40 GB hard drives, 1.5 GHz processors, and 6-hour battery life, that cost $1,000. You are looking for a way to make your product more desirable by changing as few factors as possible. You set the current product configuration as the baseline. JMP adjusts the probabilities so that the probability of preference for the baseline configuration is 0.5. Then you compare the probabilities of other configurations to the baseline probability.

1. Do one of the following:
   - Follow the steps in "Complete the Launch Window". Then complete step 1 and step 2 in "Analyze the Model".
   - In the Laptop Runs.jmp sample data table, click the green triangle next to the **Choice Reduced Model** script.
2. Click the Choice Model: Response red triangle and select **Probability Profiler**.

   Note that the Probability Profiler is for Gender = F. You can change this later.
3. Using the menus and text box below the profiler, in the Baseline area, specify the Baseline settings as 40 GB, 1.5 GHz, 6 hours, and 1000.
4. Now set these as the values in the Probability Profiler. To set the Price at $1000, click $1242 above Price under the rightmost profiler cell, and type 1000. Then click outside the text box.

Figure 4.29 Probability Profiler with Text Entry Area for Price

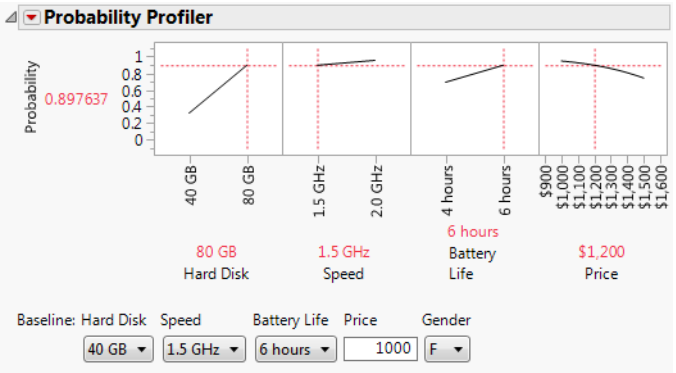


This configuration has probability 0.5.

5. In the Probability Profiler, move the slider for HardDisk to 80 GB.

   Notice that, with this change, the probability is relatively insensitive to increases in Price.
6. Click the $1000 label above the Price cell in the profiler, type **$1,200**, and click outside the text box.

Figure 4.30 Laptop Probability Profiler Results with Baseline Effects



An increase in Hard Disk size from 40 GB to 80 GB and an increase in price to $1200 coincides with an increased probability of preference, from 0.50 to 0.90 for females. Change the Gender effect in the Baseline to **M**. The probability of preference is 0.71.

Multiple Choice Comparisons

Use the Multiple Choice Profiler to compare product profiles.

- You currently produce a low-end laptop with a small hard drive, a slow processor, and low battery life. You charge $1000.
- Company A produces a product with a fast processor speed and high battery life at a reasonable price of $1200.
- Company B makes the biggest hard drives with the fastest speed, but at a high price of $1500 and low battery life.

You want to gain market share by increasing only one area of performance, and price.

1. Do one of the following:
   - Follow the steps in "Complete the Launch Window". Then complete step 1 and step 2 in "Analyze the Model".
   - In the Laptop Runs.jmp sample data table, click the green triangle next to the **Choice Reduced Model** script.
2. Click the Choice Model: Response red triangle and select **Multiple Choice Profiler**.

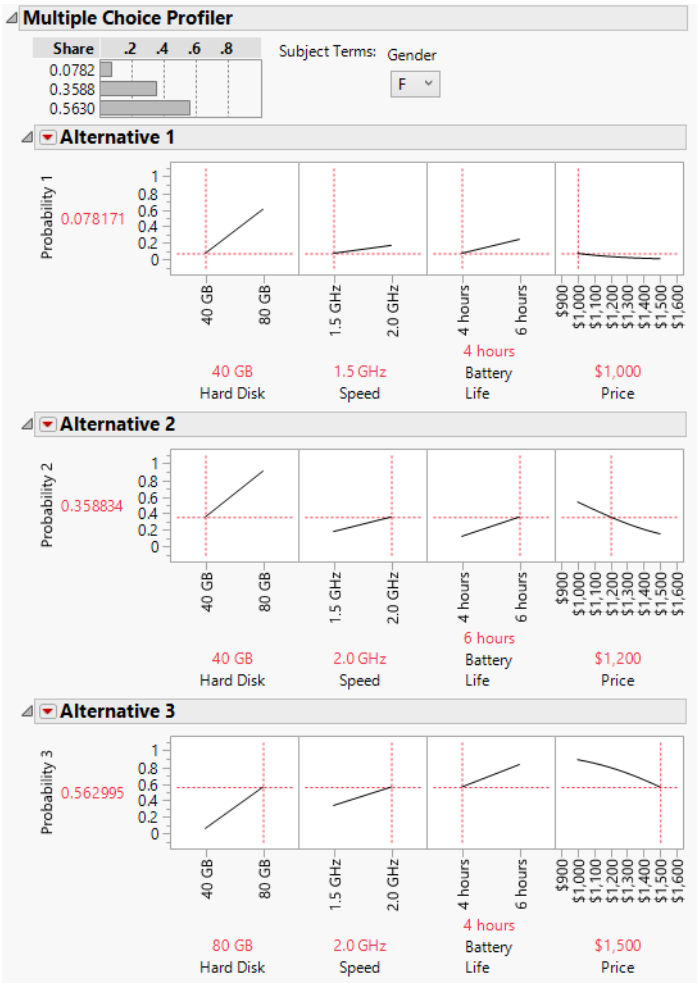
   A window appears, asking for the number of alternative choices to profile. Accept the default number of 3.
3. Click **OK**.

   Three Alternative profilers appear. Notice that the profilers are set for Gender = F.

Each factor in each profiler is set to its default values. Alternative 1 indicates the product that you want to develop. Alternative 2 indicates Company A's product. Alternative 3 indicates Company B's product.

4. For Alternative 1, set Hard Disk to 40 GB, Speed to 1.5 GHz, Battery Life to 4hours, and Price to $1,000.
5. For Alternative 2, set Hard Disk to 40 GB, Speed to 2.0 GHz, Battery Life to 6 hours, and Price to $1,200.
6. For Alternative 3, set Hard Disk to 80 GB, Speed to 2.0 GHz, Battery Life to 4 hours, and Price to $1,500.

Figure 4.31 Multiple Choice Profiler for Females

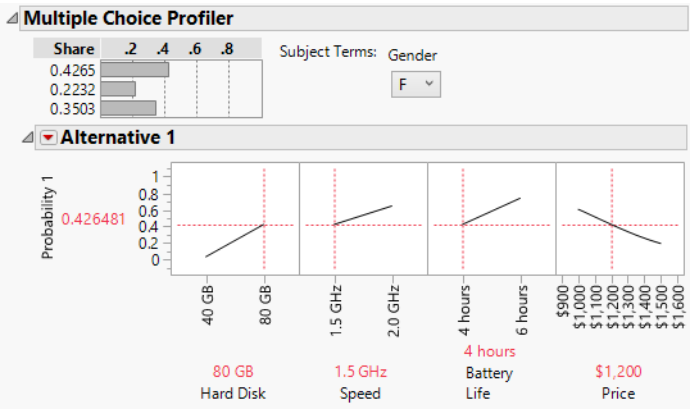


You can see that Company B has the greatest Share of 0.5630. It is obvious that with your company's settings, very few females buy your product.

You want to increase your market share by upgrading your company's laptop in one of the performance areas while increasing price. The slope of the line in Alternative 1's Hard Disk profile suggests increasing hard disk space increases market share the most.

7. For Alternative 1, set Hard Disk to **80 GB** and Price to **$1,200**.

Figure 4.32 Multiple Choice Profiler with Improved Laptop



By increasing hard disk space, you can increase the price of your laptop and expect a market share among females of about 43%. This share exceeds that of Company B's high-performance laptop and is much better than the market share with the initial low-end settings seen in Figure 4.31.

Explore the settings that increase your market share for males. If you increase both Hard Disk size and Speed, you can capture a 44% market share among males.

Example of Segmentation

In this example, you attempt to identify market segments for pizza preferences.

To see how to complete the launch window for this example, see step 1 to step 15 in the example "Multiple Table Format". Otherwise, follow the instructions below.

Define Clusters

1. Select **Help > Sample Data Library** and open Pizza Responses.jmp.
2. Click the green triangle next to the **Choice** script.
3. Click the Choice Model: Choice red triangle and select **Save Gradients by Subject**.

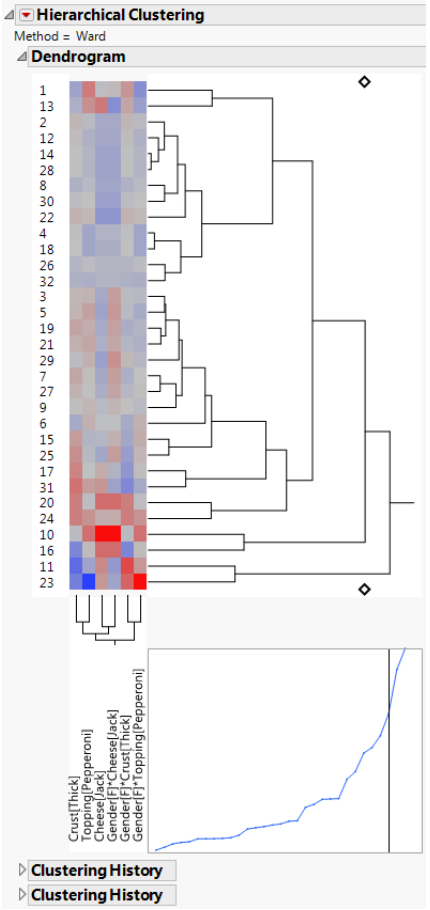
A data table appears with gradient forces saved for each main effect and subject interaction.

Figure 4.33 Gradients by Subject for Pizza Data, Partial View

	Subject	Crust[Thick]	Cheese[Jack]	Topping[Pepperoni]	Gender[F]*Crust[Thick]	Gender[F]*Cheese[Jack]	Gender[F]*Topping[Pepperoni]
1	1	-0.00959	-0.00168	0.014876	0.009585	0.001685	-0.01488
2	2	0.002373	-0.00758	-0.00239	0.002373	-0.00758	-0.00239
3	3	0.002129	-0.0079	0.003031	-0.00213	0.007899	-0.00303
4	4	-0.00106	-0.00485	-0.00901	-0.00106	-0.00485	-0.00901
5	5	0.002828	-0.00945	0.00725	-0.00283	0.009453	-0.00725
6	6	-0.0073	-0.00089	0.003761	-0.0073	-0.00089	0.003761

4. Click the green triangle next to the **Hierarchical Cluster** script.

Figure 4.34 Dendrogram of Subject Clusters for Pizza Data



The script runs a hierarchical cluster analysis on all columns in the gradient table, except for Subject. Click either diamond to see that the rows have been placed into three clusters.

5. Click the Hierarchical Clustering red triangle and select **Save Clusters**.

   A new column called Cluster is added to the data table containing the gradients. Each subject has been assigned a Cluster value that is associated with other subjects having similar gradient forces. See *Multivariate Methods* for a discussion of other Hierarchical Clustering options.

   You can delete the gradient columns because they were used only to obtain the clusters.

6. Select all columns except Subject and Cluster. Right-click the selected columns and select **Delete Columns**.

7. Click the green triangle next to the **Merge Data Back** script (Figure 4.33).

   The cluster information is merged into the Subject data table. The columns in the Subject data table are now Subject, Gender, and Cluster.

Figure 4.35 Subject Data with Cluster Column

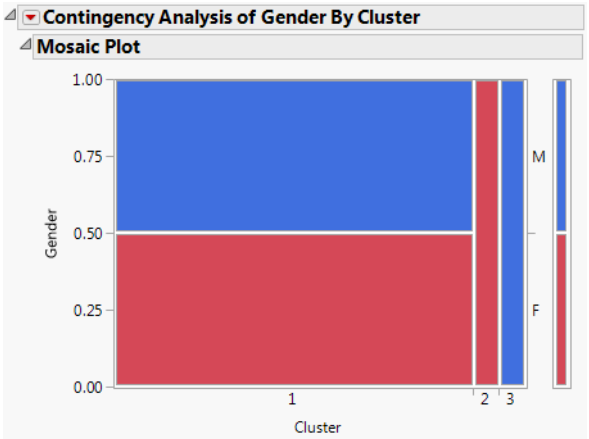
	Subject	Crust[Thick]	Cheese[Jack]	Topping[Pepperoni]	Gender[F]*Crust[Thick]	Gender[F]*Cheese[Jack]	Gender[F]*Topping[Pepperoni]	Cluster
1	1	-0.00959	-0.00168	0.014876	0.009585	0.001685	-0.01488	1
2	2	0.002373	-0.00758	-0.00239	0.002373	-0.00758	-0.00239	1
3	3	0.002129	-0.0079	0.003031	-0.00213	0.007899	-0.00303	1
4	4	-0.00106	-0.00485	-0.00901	-0.00106	-0.00485	-0.00901	1
5	5	0.002828	-0.00945	0.00725	-0.00283	0.009453	-0.00725	1
6	6	-0.0073	-0.00089	0.003761	-0.0073	-0.00089	0.003761	1
7	7	0.006003	-0.00815	0.000308	-0.006	0.008151	-0.00031	1
8	8	-0.0055	-0.00887	-0.00274	-0.0055	-0.00887	-0.00274	1
9	9	0.000438	-0.00271	0.002402	-0.00044	0.00271	-0.0024	1
10	10	-0.00217	0.032641	0.017043	-0.00217	0.032641	0.017043	2

This table can now be used for further analysis.

Explore the Clusters

1. Click the icon to the left of the Cluster variable in the columns panel and select **Nominal**.
2. Select **Analyze** > **Fit Y by X**.
3. Select Gender and click **Y, Response**.
4. Select Cluster and click **X, Factor**.
5. Click **OK**.

Figure 4.36 Contingency Analysis of Gender by Cluster



Contingency Table

Gender			
Count / Total % / Col % / Row %	F	M	
1	14	14	28
	43.75	43.75	87.50
	87.50	87.50	
	50.00	50.00	
2	2	0	2
	6.25	0.00	6.25
	12.50	0.00	
	100.00	0.00	
3	0	2	2
	0.00	6.25	6.25
	0.00	12.50	
	0.00	100.00	
	16	16	32
	50.00	50.00	

Cluster

Tests

N	DF	-LogLike	RSquare (U)
32	2	2.7725887	0.1250

Test	ChiSquare	Prob>ChiSq
Likelihood Ratio	5.545	0.0625
Pearson	4.000	0.1353

Warning: 20% of cells have expected count less than 5, ChiSquare suspect.

Observe patterns in the clusters:

- Cluster 1 is evenly divided between males and females
- Cluster 2 consists of only females
- Cluster 3 consists of only males

If desired, you could now refit and analyze the model with the addition of the Cluster variable.

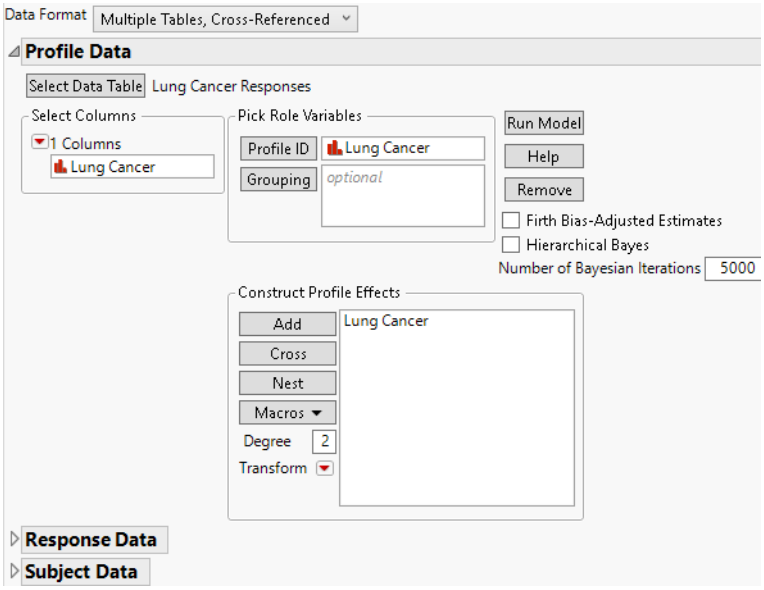
Example of Logistic Regression Using the Choice Platform

Use the Choice Platform

1. Select **Help > Sample Data Library** and open Lung Cancer Responses.jmp and Lung Cancer Choice.jmp.

   Notice Lung Cancer Responses.jmp has only one column (Lung Cancer) with two rows (Cancer and NoCancer).
2. Select **Analyze** > **Consumer Research** > **Choice**
3. Select **Multiple Tables, Cross-Referenced** from the list next to Data Format.
4. Click **Select Data Table**, select Lung Cancer Responses, and click **OK**.
5. Select Lung Cancer and click **Profile** ID.
6. Select Lung Cancer and click **Add**.
7. Uncheck the Firth Bias-Adjusted Estimates box.

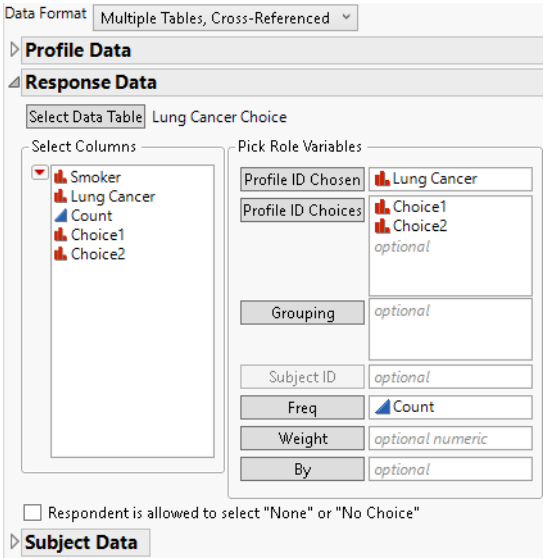
Figure 4.37 Completed Profile Data Panel



8. Open the **Response Data** outline.
9. Click **Select Data Table**, select Lung Cancer Choice, and click **OK**.
10. Complete the **Response Data** outline:
    - Select Lung Cancer and click **Profile ID Chosen**.
    - Select Choice1 and Choice2 and click **Profile ID Choices**.

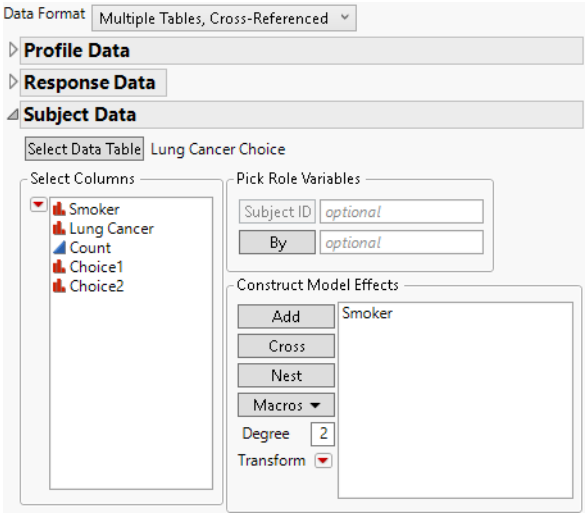
– Select Count and click **Freq**.

Figure 4.38 Completed Response Data Panel



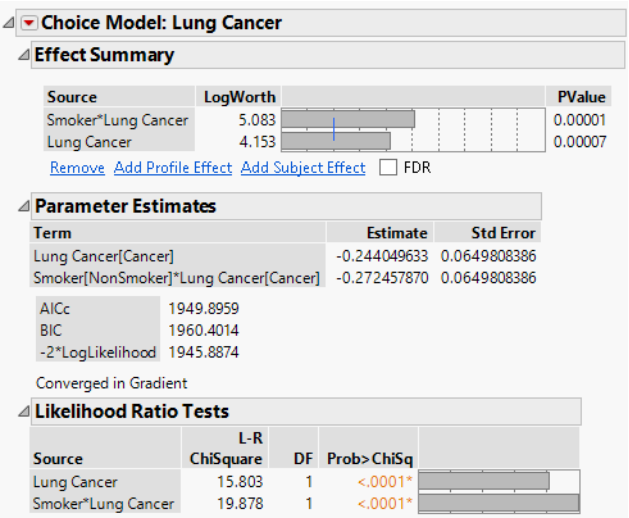
11. Open the **Subject Data** outline.
12. Click **Select Data Table**, select Lung Cancer Choice, and click **OK**.
13. Select Smoker and click **Add**.

Figure 4.39 Completed Subject Data Panel



14. Click **Run Model**.

Figure 4.40 Choice Modeling Logistic Regression Results

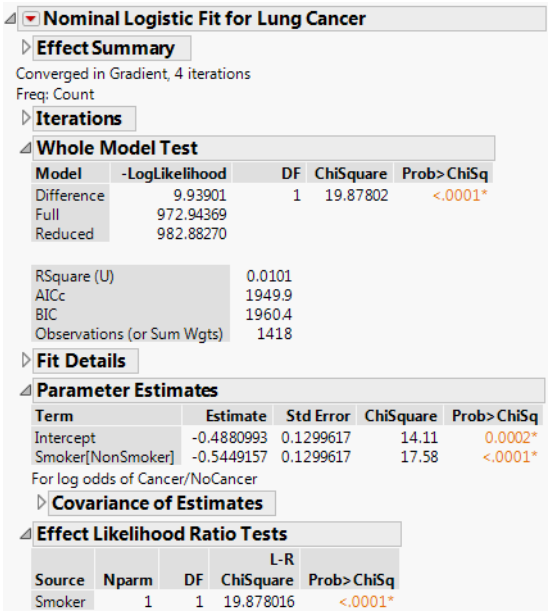


Use the Fit Model Platform

1. Select **Help > Sample Data Library** and open Lung Cancer.jmp.
2. Select **Analyze** > **Fit Model**.

   Because the data table contains a model script, the Model Specification window is automatically completed. The **Nominal Logistic** personality is selected.
3. Set **Target Level** to Cancer.
4. Click **Run**.

Figure 4.41 Fit Model Nominal Logistic Regression Results



Nominal Logistic Fit for Lung Cancer

Effect Summary

Converged in Gradient, 4 iterations
Freq: Count

Iterations

Whole Model Test

Model	-LogLikelihood	DF	ChiSquare	Prob>ChiSq
Difference	9.93901	1	19.87802	<.0001*
Full	972.94369			
Reduced	982.88270			

RSquare (U)	0.0101
AICc	1949.9
BIC	1960.4
Observations (or Sum Wgts)	1418

Fit Details

Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept	-0.4880993	0.1299617	14.11	0.0002*
Smoker[NonSmoker]	-0.5449157	0.1299617	17.58	<.0001*

For log odds of Cancer/NoCancer

Covariance of Estimates

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
Smoker	1	1	19.878016	<.0001*

Notice that the likelihood ratio chi-square test for Smoker*Lung Cancer in the Choice model matches the likelihood ratio chi-square test for Smoker in the Logistic model. The reports shown in Figure 4.40 and Figure 4.41 support the conclusion that smoking has a strong effect on developing lung cancer. See *Fitting Linear Models*.

Example of Logistic Regression for Matched Case-Control Studies

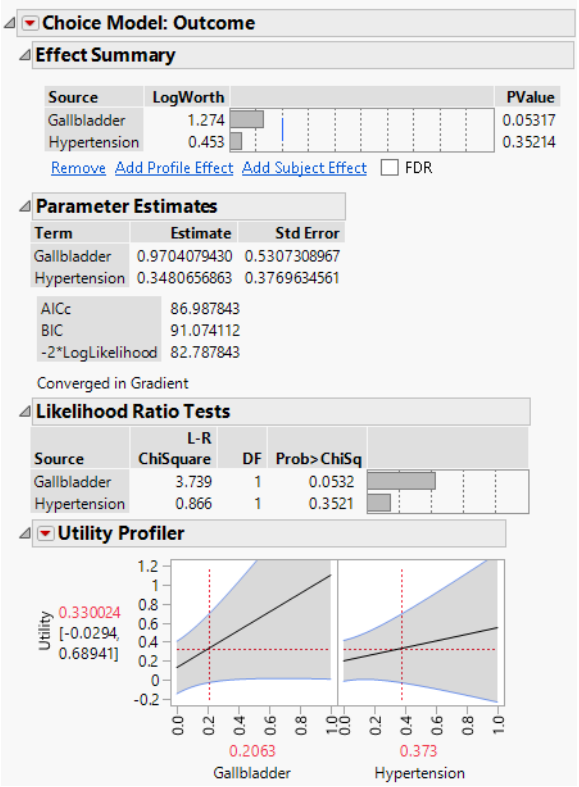
This section provides an example using the Choice platform to perform logistic regression on the results of a study of endometrial cancer with 63 matched pairs. The data are from the Los Angeles Study of the Endometrial Cancer Data reported in Breslow and Day (1980). The goal of the case-control analysis was to determine the relative risk for gallbladder disease, controlling for the effect of hypertension. The Outcome of 1 indicates the presence of endometrial cancer, and 0 indicates the control. Gallbladder and Hypertension data indicators are also 0 or 1.

For more information about performing logistic regression using the Choice platform, see "Logistic Regression".

1. Select **Help > Sample Data Library** and open Endometrial Cancer.jmp.
2. Select **Analyze > Consumer Research > Choice**.
3. Check that the Data Format selected is **One-Table, Stacked**.
4. Click the **Select Data Table** button.

5. Select Endometrial Cancer as the profile data table. Click **OK**.
6. Select Outcome and click **Response Indicator**.
7. Select Pair and click **Grouping**.
8. Select Gallbladder and Hypertension and click **Add** in the Construct Profile Effects window.
9. Deselect the **Firth Bias-Adjusted Estimates** check box.
10. Click **Run Model**.
11. Click the Choice Model: Outcome red triangle and select **Utility Profiler**.

Figure 4.42 Logistic Regression on Endometrial Cancer Data



Likelihood Ratio tests are given for each factor. Note that Gallbladder is nearly significant at the 0.05 level (p-value = 0.0532). Use the Utility Profiler to visualize the impact of the factors on the response.

Example of Transforming Data to Two Analysis Tables

Consider the data from Daganzo, found in Daganzo Trip.jmp. This data set contains the travel time for three transportation alternatives and the preferred transportation alternative for each subject.

Add Choice Mode and Subjects

1. Select **Help > Sample Data Library** and open the Daganzo Trip.jmp data table.

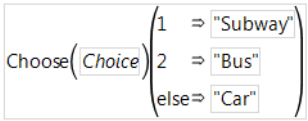
Figure 4.43 Partial Daganzo Trip Table

	Subway	Bus	Car	Choice
1	16.481	16.196	23.89	2
2	15.123	11.373	14.182	2
3	19.469	8.822	20.819	2
4	18.847	15.649	21.28	2
5	12.578	10.671	18.335	2

Each Choice number listed must first be converted to one of the travel mode names. This transformation is easily done by using the **Choose** function in the formula editor:

2. Select **Cols > New Columns**.
3. Specify the Column Name as Choice Mode and the modeling type as **Nominal**.
4. Click the **Column Properties** and select **Formula**.
5. Click **Conditional** in the functions list, select **Choose**, and press comma twice to obtain additional arguments for the function.
6. Click Choice for the Choose expression (expr), and double-click each clause entry box to enter “Subway”, “Bus”, and “Car” (with the quotation marks).

Figure 4.44 Choose Function for Choice Mode Column of Daganzo Data



7. Click **OK** in the Formula Editor window.
8. Click **OK** in the New Column window.

   The new Choice Mode column appears in the data table. Because each row contains a choice made by each subject, another column containing a sequence of numbers should be created to identify the subjects.
9. Select **Cols > New Columns**.

10. Specify the Column Name as Subject.
11. Click **Missing/Empty** next to Initialize Data and select **Sequence Data**.
12. Click **OK**.

Figure 4.45 Partial Daganzo Trip Data with New Choice Mode and Subject Columns

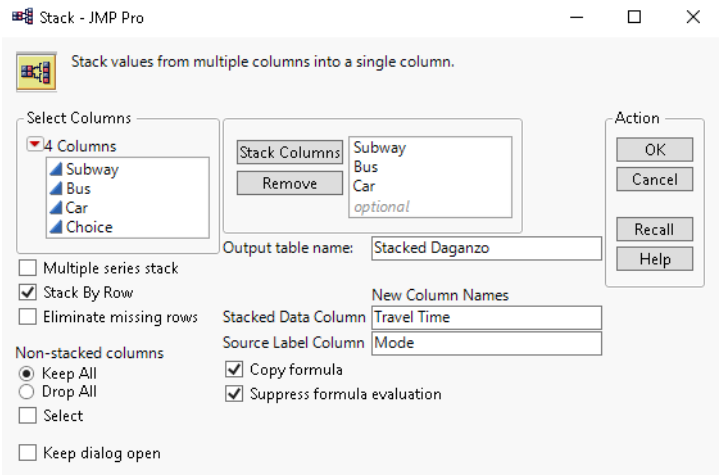
	Subway	Bus	Car	Choice	Choice Mode	Subject
1	16.481	16.196	23.89	2	Bus	1
2	15.123	11.373	14.182	2	Bus	2
3	19.469	8.822	20.819	2	Bus	3
4	18.847	15.649	21.28	2	Bus	4
5	12.578	10.671	18.335	2	Bus	5
6	11.513	20.582	27.838	1	Subway	6
7	10.651	15.537	17.418	1	Subway	7

Stack the Data

In order to construct the Profile data, each alternative needs to be expressed in a separate row.

1. Select **Tables > Stack**.
2. Select Subway, Bus, and Car and click **Stack Columns**.
3. For the Output table name, type Stacked Daganzo. Type Travel Time for the Stacked Data Column and Mode for the Source Label Column.

Figure 4.46 Stack Dialog for Daganzo Data



4. Click **OK**.

Figure 4.47 Partial Stacked Daganzo Table

	Choice	Choice Mode	Subject	Mode	Travel Time
1	2	Bus	1	Subway	16.481
2	2	Bus	1	Bus	16.196
3	2	Bus	1	Car	23.89
4	2	Bus	2	Subway	15.123
5	2	Bus	2	Bus	11.373
6	2	Bus	2	Car	14.182
7	2	Bus	3	Subway	19.469

Make the Profile Data Table

For the Profile Data Table, you need the Subject, Mode, and Travel Time columns.

1. Select the Subject, Mode, and Travel Time columns and select **Tables** > **Subset**.
2. Select **All Rows** and **Selected Columns** and click **OK**.

   A partial data table is shown in Figure 4.48. Note the default table name is Subset of Stacked Daganzo.

Figure 4.48 Partial Subset Table of Stacked Daganzo Data

	Subject	Mode	Travel Time
1	1	Subway	16.481
2	1	Bus	16.196
3	1	Car	23.89
4	2	Subway	15.123
5	2	Bus	11.373
6	2	Car	14.182
7	3	Subway	19.469

Make the Response Data Table

For the Response Data Table, you need the Subject and Choice Mode columns, but you also need a column for each possible choice.

3. From the Daganzo Trip.jmp data, select the Subject and Choice Mode columns.
4. Select **Tables > Subset**.
5. Select **All Rows** and **Selected Columns** and click **OK**.

   Note that the default table name is Subset of Daganzo Trip.
6. Select **Cols > New Columns**.
7. For the Column prefix, type Choice.
8. Select **Data Type>Character**.

9. Enter 3 for the Number of columns to add.
10. Click **OK**.

    The columns Choice 1, Choice 2, and Choice 3 have been added.
11. Type "Bus" (without quotation marks) in the first row of Choice 1. Right-click the cell and select **Fill > Fill to end of table**.
12. Type "Subway" (without quotation marks) in the first row of Choice 2. Right-click the cell and select **Fill > Fill to end of table**.
13. Type "Car" (without quotation marks) in the first row of Choice 3. Right-click the cell and select **Fill > Fill to end of table**.

Figure 4.49 Partial Subset Table of Daganzo Data with Choice Set

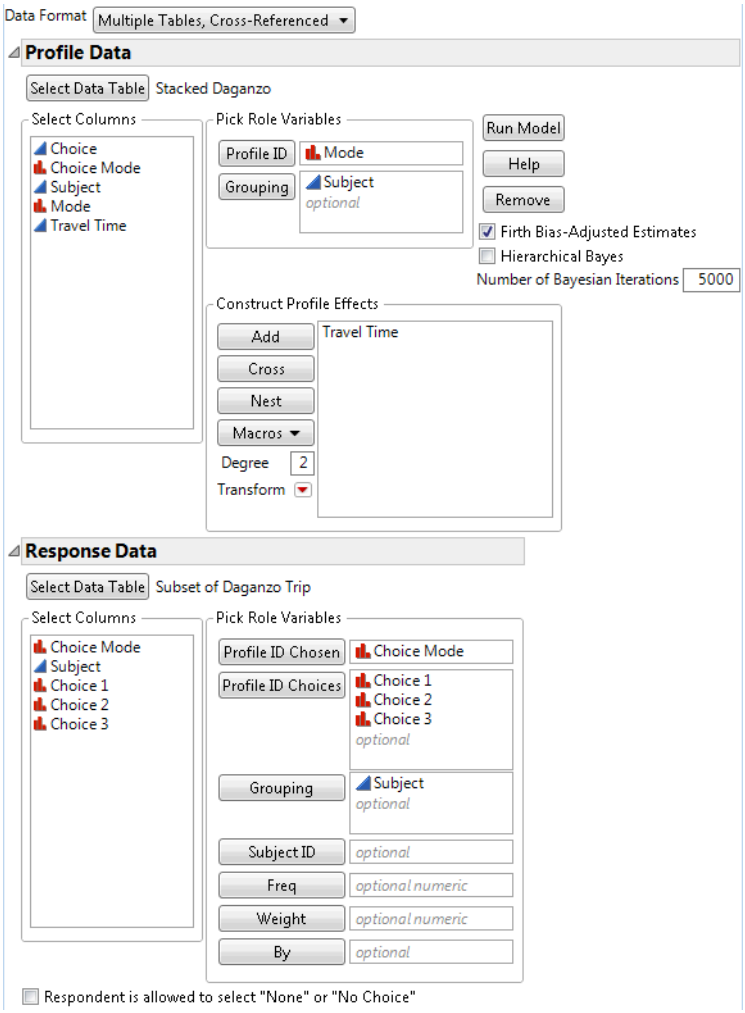
	Choice Mode	Subject	Choice 1	Choice 2	Choice 3
1	Bus	1	Bus	Subway	Car
2	Bus	2	Bus	Subway	Car
3	Bus	3	Bus	Subway	Car
4	Bus	4	Bus	Subway	Car
5	Bus	5	Bus	Subway	Car
6	Subway	6	Bus	Subway	Car
7	Subway	7	Bus	Subway	Car

Fit the Model

Now that you have separated the original Daganzo Trip.jmp table into two separate tables, you can run the Choice Platform.

1. Select **Analyze > Consumer Research > Choice**.
2. From the Data Format list, select **Multiple Tables, Cross-Referenced**.
3. Specify the model, as shown in Figure 4.50.

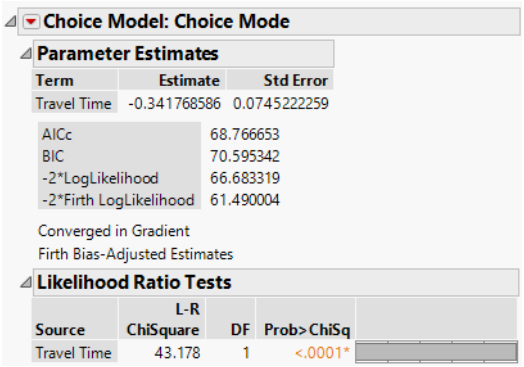
Figure 4.50 Choice Dialog Box for Daganzo Data with Multiple Tables



4. Click **Run Model**.

   The resulting parameter estimate now expresses the utility coefficient for Travel Time.

Figure 4.51 Parameter Estimate for Travel Time of Daganzo Data



The negative coefficient implies that increased travel time has a negative effect on consumer utility or satisfaction. The likelihood ratio test result indicates that the Choice model with the effect of Travel Time is significant.

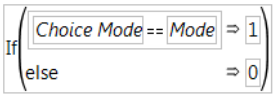
Example of Transforming Data to One Analysis Table

Rather than creating two or three tables, it can be more practical to transform the data so that only one table is used. For the one-table format, the subject effect is added as in the previous example. A response indicator column is added instead of using three different columns for the choice sets (Choice 1, Choice 2, Choice 3). Transform data for use with the one-table scenario:

1. Create or open Stacked Daganzo.jmp from the "Stack the Data" steps shown in "Example of Transforming Data to Two Analysis Tables".
2. Select **Cols > New Columns**.
3. Type Response as the Column Name.
4. Click **Column Properties** and select **Formula**.
5. Select **Conditional** in the functions list and then select **If**.
6. Select the column Choice Mode for the expression (expr).
7. Enter "=" and select Mode.
8. Type 1 for the **Then Clause** and 0 for the **Else Clause**.
9. Click **OK** in the Formula Editor window. Click **OK** in the New Column window.

   The completed formula should look like Figure 4.52.

Figure 4.52 Formula for Response Indicator for Stacked Daganzo Data



10. Select the Subject, Travel Time, and Response columns and then select **Tables** > **Subset**.
11. Select **All Rows** and **Selected Columns** and click **OK**.

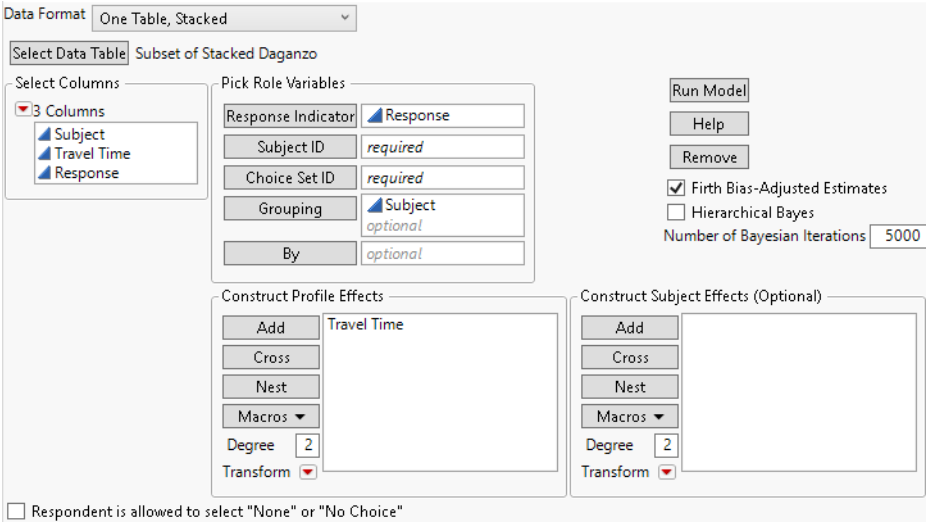
    A partial listing of the new data table is shown in Figure 4.53.

Figure 4.53 Partial Table of Stacked Daganzo Data Subset

	Subject	Travel Time	Response
1	1	16.481	0
2	1	16.196	1
3	1	23.89	0
4	2	15.123	0
5	2	11.373	1
6	2	14.182	0
7	3	19.469	0

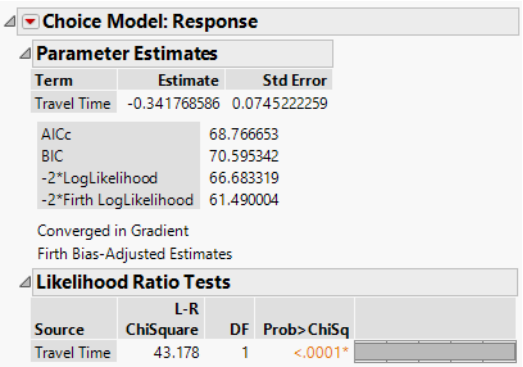
12. Select **Analyze** > **Consumer Research** > **Choice** to open the launch window and specify the model as shown in Figure 4.54.

Figure 4.54 Choice Dialog Box for Subset of Stacked Daganzo Data for One-Table Analysis



13. Click **Run Model.**

Figure 4.55 Parameter Estimate for Travel Time of Daganzo Data from One-Table Analysis



Notice that the result is identical to that obtained for the two-table model, shown earlier in Figure 4.51.

This chapter illustrates the use of the Choice Modeling platform with simple examples. This platform can also be used for more complex models, such as those involving more complicated transformations and interaction terms.

Statistical Details for the Choice Platform

- "Special Data Table Rules"
- "Utility and Probabilities"
- "Gradients"

Special Data Table Rules

Default Choice Set

If in every trial, you can choose any of the response profiles, you can omit the **Profile ID Choices** selection under **Pick Role Variables** in the Response Data section of the Choice launch window. The Choice Model platform then assumes that all choice profiles are available on each run.

Subject Data with Response Data

If you have subject data in the Response data table, select this table as the **Select Data Table** under the Subject Data. In this case, a **Subject ID** column does not need to be specified. In fact, it is not used. It is generally assumed that the subject data repeats consistently in multiple runs for each subject.

Logistic Regression

Ordinary logistic regression can be performed with the Choice Modeling platform.

Note: The Fit Y by X and Fit Model platforms are more convenient to use than the Choice Modeling platform for logistic regression modeling. This section is used only to demonstrate that the Choice Modeling platform can be used for logistic regression, if desired.

If your data are already in the choice-model format, you might want to use the steps given below for logistic regression analysis. However, three steps are needed:

- Create a trivial Profile data table with a row for each response level.
- Put the explanatory variables into the Response data.
- Specify the Response data table, again, for the Subject data table.

For examples of conducting Logistic Regression using the Choice Platform, see "Example of Logistic Regression Using the Choice Platform" and "Example of Logistic Regression for Matched Case-Control Studies".

Utility and Probabilities

Parameter estimates from the choice model identify consumer *utility*, or marginal utilities in the case of a linear utility function. Utility is the level of satisfaction consumers receive from products with specific attributes and is determined from the parameter estimates in the model.

The choice statistical model is expressed as follows:

Let $X[k]$ represent a subject attribute design row, with intercept

Let $Z[j]$ represent a choice attribute design row, without intercept

Then, the probability of a given choice for the $k'th$ subject to the $j'th$ choice of m choices is:

$$P_i[jk] = \frac{\exp(\beta'(X[k] \otimes Z[j]))}{\sum_{l=1}^{m} \exp(\beta'(X[k] \otimes Z[l]))}$$

where:

- $\otimes$ is the Kronecker rowwise product
- the numerator calculates for the $j'th$ alternative actually chosen
- the denominator sums over the m choices presented to the subject for that trial

Gradients

The gradient values that you obtain when you select the Save Gradients by Subject option are the subject-aggregated Newton-Raphson steps from the optimization used to produce the estimates. At the estimates, the total gradient is zero, and $\Delta = H^{-1}g = 0$, where g is the total gradient of the log-likelihood evaluated at the MLE, and H^{-1} is the inverse Hessian function or the inverse of the negative of the second partial derivative of the log-likelihood.

But, the disaggregation of Δ results in the following:

$$\Delta = \Sigma_{ij}\Delta_{ij} = \Sigma H^{-1}g_{ij} = 0,$$

Here i is the subject index, j is the choice response index for each subject, Δ_{ij} are the partial Newton-Raphson steps for each run, and g_{ij} is the gradient of the log-likelihood by run.

The mean gradient step for each subject is then calculated as follows:

$$\bar{\Delta}_i = \Sigma_j \frac{\Delta_{ij}}{n_i} ,$$

where n_i is the number of runs per subject. The $\bar{\Delta}_i$ are related to the force that subject i is applying to the parameters. If groups of subjects have truly different preference structures, these forces are strong, and they can be used to cluster the subjects. The $\bar{\Delta}_i$ are the gradient forces that are saved. You can then cluster these values using the Clustering platform.

Chapter 5

MaxDiff

Fit Models for MaxDiff Experiments

Use MaxDiff (maximum difference scaling) as an alternative to standard preference scales to determine the relative importance of items being rated. MaxDiff forces respondents to report their most and least preferred options. This often results in rankings that are more definitive than rankings obtained using standard preference scales.

The MaxDiff platform enables you to do the following:

- Use information about respondent (subject) traits as well as product attributes.
- Integrate data from one, two, or three sources.
- Obtain subject-level scores for segmenting or clustering your data.
- JMP PRO Estimate subject-specific coefficients using a Bayesian approach.
- Use bias-corrected maximum likelihood estimators (Firth 1993).

Figure 5.1 MaxDiff All Comparisons Report

All Levels Comparison Report

Difference (Row-Column) Standard Error of Difference Wald p-Value	All Dressed	Barbecue	Biscuits and Gravy	Dill Pickle	Gyro	Ketchup	Reuben	Sour Cream and Onion	Southern Barbecue	Truffle Fries
All Dressed	0	-1.4951	-0.3222	-0.0804	0.94665	0.12371	0.24744	-0.228	-1.0068	-0.3615
	0	0.36838	0.29974	0.29447	0.33571	0.29728	0.30148	0.28158	0.3418	0.31136
		9.22e-5	0.28477	0.78538	0.00569	0.67812	0.41355	0.41975	0.00393	0.24809
Barbecue	1.49507	0	1.1729	1.41468	2.44172	1.61877	1.7425	1.26703	0.48824	1.13355
	0.36838	0	0.36267	0.3539	0.41499	0.36911	0.37117	0.37654	0.39164	0.39213
	9.22e-5		0.00161	0.00012	4.32e-8	2.64e-5	7.69e-6	0.00105	0.21514	0.00462
Biscuits and Gravy	0.32217	-1.1729	0	0.24179	1.26882	0.44588	0.56961	0.09413	-0.6847	-0.0393
	0.29974	0.36267	0	0.28249	0.34456	0.29752	0.30639	0.30716	0.34038	0.31345
	0.28477	0.00161		0.39388	0.00036	0.1368	0.06567	0.75983	0.0467	0.90032
Dill Pickle	0.08038	-1.4147	-0.2418	0	1.02703	0.20409	0.32782	-0.1477	-0.9264	-0.2811
	0.29447	0.3539	0.28249	0	0.35251	0.27938	0.30328	0.30707	0.35532	0.30871
	0.78538	0.00012	0.39388		0.00432	0.46661	0.28208	0.63156	0.01038	0.36444
Gyro	-0.9467	-2.4417	-1.2688	-1.027	0	-0.8229	-0.6992	-1.1747	-1.9535	-1.3082
	0.33571	0.41499	0.34456	0.35251	0	0.32802	0.34758	0.33327	0.38939	0.35477
	0.00569	4.32e-8	0.00036	0.00432		0.01356	0.04668	0.00062	2.01e-6	0.00035
Ketchup	-0.1237	-1.6188	-0.4459	-0.2041	0.82294	0	0.12373	-0.3517	-1.1305	-0.4852
	0.29728	0.36911	0.29752	0.27938	0.32802	0	0.30893	0.31915	0.34808	0.31157
	0.67812	2.64e-5	0.1368	0.46661	0.01356		0.68956	0.27279	0.00154	0.12223
Reuben	-0.2474	-1.7425	-0.5696	-0.3278	0.69921	-0.1237	0	-0.4755	-1.2543	-0.609
	0.30148	0.37117	0.30639	0.30328	0.34758	0.30893	0	0.30932	0.35324	0.31817
	0.41355	7.69e-6	0.06567	0.28208	0.04668	0.68956		0.12709	0.00056	0.05821
Sour Cream and Onion	0.22804	-1.267	-0.0941	0.14766	1.17469	0.35175	0.47548	0	-0.7788	-0.1335
	0.28158	0.37654	0.30716	0.30707	0.33327	0.31915	0.30932	0	0.34896	0.31379
	0.41975	0.00105	0.75983	0.63156	0.00062	0.27279	0.12709		0.02764	0.67137
Southern Barbecue	1.00683	-0.4882	0.68465	0.92644	1.95348	1.13053	1.25426	0.77879	0	0.6453
	0.3418	0.39164	0.34038	0.35532	0.38939	0.34808	0.35324	0.34896	0	0.34904
	0.00393	0.21514	0.0467	0.01038	2.01e-6	0.00154	0.00056	0.02764		0.06715
Truffle Fries	0.36152	-1.1335	0.03935	0.28114	1.30817	0.48523	0.60896	0.13348	-0.6453	0
	0.31136	0.39213	0.31345	0.30871	0.35477	0.31157	0.31817	0.31379	0.34904	0
	0.24809	0.00462	0.90032	0.36444	0.00035	0.12223	0.05821	0.67137	0.06715	

Contents

Overview of the MaxDiff Modeling Platform 143
Examples of the MaxDiff Platform 143
One Table Format 144
Multiple Table Format 147
Launch the MaxDiff Platform 150
Launch Window for One Table, Stacked 151
Launch Window for Multiple Tables, Cross-Referenced 152
MaxDiff Model Report 156
Effect Summary 156
MaxDiff Results 157
Parameter Estimates 158
Bayesian Parameter Estimates 160
Likelihood Ratio Tests 161
MaxDiff Platform Options 162
Comparisons Report 164
Save Bayes Chain 165

Overview of the MaxDiff Modeling Platform

MaxDiff, also known as *best-worst scaling* (BWS), is a choice-based measurement method. Rather than asking a respondent to report one favorite choice among several alternative profiles, MaxDiff asks a respondent to report both a *best* and a *worst* choice. The MaxDiff approach can provide more information about preferences than an approach where a respondent reports only a favorite choice. For background on MaxDiff studies, see Louviere et al. (2015). For background on choice modeling, see Louviere et al. (2015), Train (2009), and Rossi et al. (2005).

MaxDiff analysis uses the framework of random utility theory. A choice is assumed to have an underlying value, or *utility*, to respondents. The MaxDiff platform estimates these utilities. The MaxDiff platform also estimates the probabilities that a choice is preferred over other choices. This is done using conditional logistic regression. See McFadden (1974).

Note: One-factor MaxDiff studies can be designed using the MaxDiff Design platform. See the *Design of Experiments Guide*.

Segmentation and Bayesian Subject-Level Effects

Market researchers sometimes want to analyze the preference structure for each subject separately in order to see whether there are groups of subjects that behave differently. If there are sufficient data, you can specify "By groups" in the Response Data or you could introduce a Subject identifier as a subject-side model term. This approach, however, is costly if the number of subjects is large. Other segmentation techniques discussed in the literature include Bayesian and mixture methods.

If there are not sufficient data to specify "By groups," you can segment in JMP by clustering subjects using response data and the **Save Gradients by Subject** option. The option creates a new data table containing the average Hessian-scaled gradient on each parameter for each subject. For an example, see "Example of Segmentation". For more information about the gradient values, see "Gradients".

JMP PRO MaxDiff also provides a Hierarchical Bayesian approach to estimating subject-level effects. This approach can be useful in market segmentation.

Examples of the MaxDiff Platform

Thirty respondents participated in a MaxDiff study to compare seven flavors of potato chips. Each choice set consisted of three profiles (potato chip flavors). For each choice set, a respondent's favorite choice was recorded as 1 and his or her least favorite choice was recorded as -1. Intermediate choices were recorded as 0.

The MaxDiff platform can analyze data that is presented in a one-table format or in a multiple-table format. In the multiple table format, information about responses, choice sets, and subjects is saved in different data tables. In the one-table format, that information is contained in a single data table.

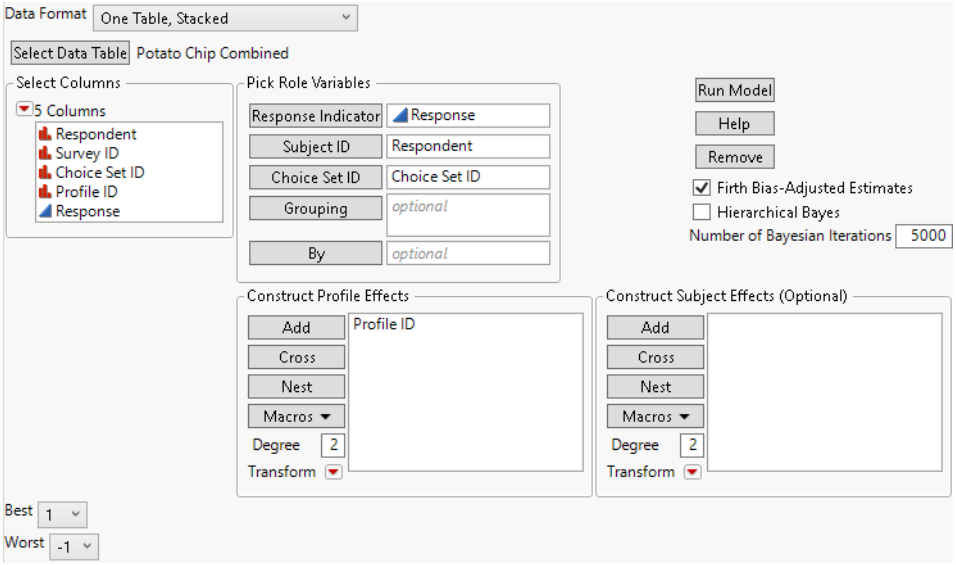
- "One Table Format" shows how to analyze a subset of the available data in a one-table format. Note that you could add additional profile and subject data to the single table for a more complete analysis.
- "Multiple Table Format" shows how to bring together information from different tables into one MaxDiff analysis.

One Table Format

1. Select **Help > Sample Data Library** and open Potato Chip Combined.jmp.
2. Select **Analyze > Consumer Research > MaxDiff**.

   Note that the default Data Format is set to One Table, Stacked.
3. Click **Select Data Table**.
4. Select Potato Chip Combined and click **OK**.
5. Assign roles to columns to complete launch dialog:
   - Select Response and click **Response Indicator**.
   - Select Respondent and click **Subject ID**.
   - Select Choice Set ID and click **Choice Set ID**.
   - Select ProfileID and click **Add** in the Construct Profile Effects panel.

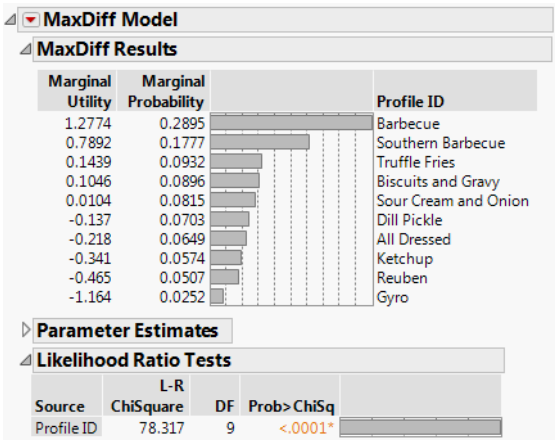
Figure 5.2 Completed MaxDiff Launch Window



Note that the setting for Worst choice changed to -1 when you specified the Response column as the Response Indicator variable.

6. Click **Run Model**.

Figure 5.3 MaxDiff Report for Potato Chip Combined.jmp



The report indicates that Profile ID is significant, indicating that preferences for the various chip types differ significantly. The highest Marginal Utility is for Barbecue chips. The estimated probability that Barbecue chips are preferred to other chip types is 0.2895.

7. Click the MaxDiff Model red triangle and select **All Levels Comparison Report.**

Figure 5.4 All Comparisons Report

All Levels Comparison Report

Difference (Row-Column) Standard Error of Difference Wald p-Value	All Dressed	Barbecue	Biscuits and Gravy	Dill Pickle	Gyro	Ketchup	Reuben	Sour Cream and Onion	Southern Barbecue	Truffle Fries
All Dressed	0	-1.4951	-0.3222	-0.0804	0.94665	0.12371	0.24744	-0.228	-1.0068	-0.3615
	0	0.36838	0.29974	0.29447	0.33571	0.29728	0.30148	0.28158	0.3418	0.31136
		9.22e-5	0.28477	0.78538	0.00569	0.67812	0.41355	0.41975	0.00393	0.24809
Barbecue	1.49507	0	1.1729	1.41468	2.44172	1.61877	1.7425	1.26703	0.48824	1.13355
	0.36838	0	0.36267	0.3539	0.41499	0.36911	0.37117	0.37654	0.39164	0.39213
	9.22e-5		0.00161	0.00012	4.32e-8	2.64e-5	7.69e-6	0.00105	0.21514	0.00462
Biscuits and Gravy	0.32217	-1.1729	0	0.24179	1.26882	0.44588	0.56961	0.09413	-0.6847	-0.0393
	0.29974	0.36267	0	0.28249	0.34456	0.29752	0.30639	0.30716	0.34038	0.31345
	0.28477	0.00161		0.39388	0.00036	0.1368	0.06567	0.75983	0.0467	0.90032
Dill Pickle	0.08038	-1.4147	-0.2418	0	1.02703	0.20409	0.32782	-0.1477	-0.9264	-0.2811
	0.29447	0.3539	0.28249	0	0.35251	0.27938	0.30328	0.30707	0.35532	0.30871
	0.78538	0.00012	0.39388		0.00432	0.46661	0.28208	0.63156	0.01038	0.36444
Gyro	-0.9467	-2.4417	-1.2688	-1.027	0	-0.8229	-0.6992	-1.1747	-1.9535	-1.3082
	0.33571	0.41499	0.34456	0.35251	0	0.32802	0.34758	0.33327	0.38939	0.35477
	0.00569	4.32e-8	0.00036	0.00432		0.01356	0.04668	0.00062	2.01e-6	0.00035
Ketchup	-0.1237	-1.6188	-0.4459	-0.2041	0.82294	0	0.12373	-0.3517	-1.1305	-0.4852
	0.29728	0.36911	0.29752	0.27938	0.32802	0	0.30893	0.31915	0.34808	0.31157
	0.67812	2.64e-5	0.1368	0.46661	0.01356		0.68956	0.27279	0.00154	0.12223
Reuben	-0.2474	-1.7425	-0.5696	-0.3278	0.69921	-0.1237	0	-0.4755	-1.2543	-0.609
	0.30148	0.37117	0.30639	0.30328	0.34758	0.30893	0	0.30932	0.35324	0.31817
	0.41355	7.69e-6	0.06567	0.28208	0.04668	0.68956		0.12709	0.00056	0.05821
Sour Cream and Onion	0.22804	-1.267	-0.0941	0.14766	1.17469	0.35175	0.47548	0	-0.7788	-0.1335
	0.28158	0.37654	0.30716	0.30707	0.33327	0.31915	0.30932	0	0.34896	0.31379
	0.41975	0.00105	0.75983	0.63156	0.00062	0.27279	0.12709		0.02764	0.67137
Southern Barbecue	1.00683	-0.4882	0.68465	0.92644	1.95348	1.13053	1.25426	0.77879	0	0.6453
	0.3418	0.39164	0.34038	0.35532	0.38939	0.34808	0.35324	0.34896	0	0.34904
	0.00393	0.21514	0.0467	0.01038	2.01e-6	0.00154	0.00056	0.02764		0.06715
Truffle Fries	0.36152	-1.1335	0.03935	0.28114	1.30817	0.48523	0.60896	0.13348	-0.6453	0
	0.31136	0.39213	0.31345	0.30871	0.35477	0.31157	0.31817	0.31379	0.34904	0
	0.24809	0.00462	0.90032	0.36444	0.00035	0.12223	0.05821	0.67137	0.06715	

Each comparison is the difference in estimated utilities between the chip type labeling the row and the chip type labeling the column. Small *p*-values are colored with an intense blue or red color, depending on the sign of the difference. For example, based on the blue colors across the Gyro row, you can see that Gyro chips have significantly lower utility than all other chip types. Barbecue chips have higher utility than all other chip types, though they do not differ significantly from Southern Barbecue chips.

Note: Because the All Comparisons Report *p*-values are not corrected for multiple comparisons, use them as a guide.

Multiple Table Format

This version of the potato chip study uses three data tables: Potato Chip Profiles.jmp, Potato Chip Responses.jmp, and Potato Chip Subjects.jmp. Although you can always arrange your data into a single table, a multi-table approach can be more convenient than a one-table analysis when you have additional profile and subject variables that you want to include in your analysis.

Complete the Launch Window

1. Select **Help > Sample Data Library** and open the Potato Chip Responses.jmp sample data table.

Note: If you prefer not to follow the steps for completing the launch window, click the green triangle next to the **MaxDiff for Flavor** script. Then proceed to "Explore the Model".

2. Click the green triangle next to the **Open Profile and Subject Tables** script.
   - The profile data table, Potato Chip Profiles.jmp, lists all the potato chip types in the study (Flavor) along with information about the country of origin (Product Of). Each choice has a Profile ID.
   - The subjects data table, Potato Chip Subjects.jmp, lists the respondents. It also gives additional information about each respondent: Citizenship and Gender.
   - The responses data table, Potato Chip Responses.jmp, lists the respondents. For each respondent, the Survey ID and Choice Set ID for each set of profiles is listed, along with the Profile ID values for each choice set. The table also contains response data in the Best Profile and Worst Profile columns.
3. From any of the three data tables, select **Analyze > Consumer Research > MaxDiff**.
4. From the Data Format list, select Multiple Tables, Cross-Referenced.

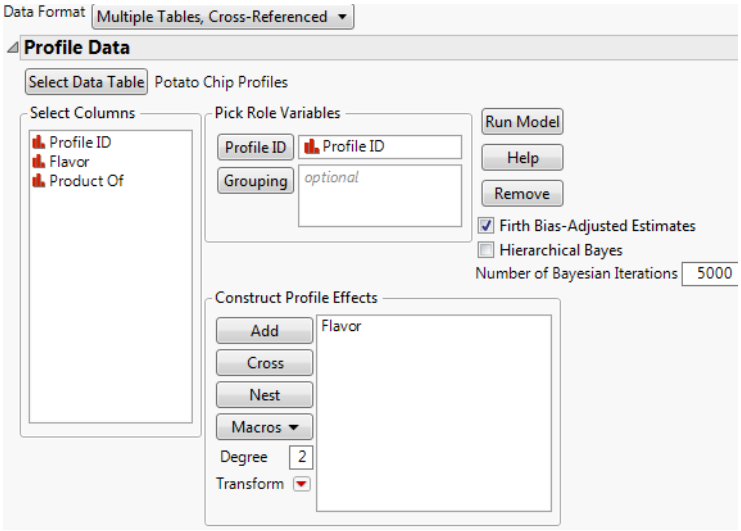
   There are three separate outlines, one for each of the data sources.
5. Click **Select Data Table** under Profile Data.

   A Profile Data Table window appears, which prompts you to specify the data table for the profile data.
6. Select Potato Chip Profiles and click **OK**.

   The columns from this table appear in the **Select Columns**.
7. Select Profile ID from the Select Columns list and click **Profile ID** under **Pick Role Variables**.
8. Select Flavor and click **Add** under **Construct Model Effects.**

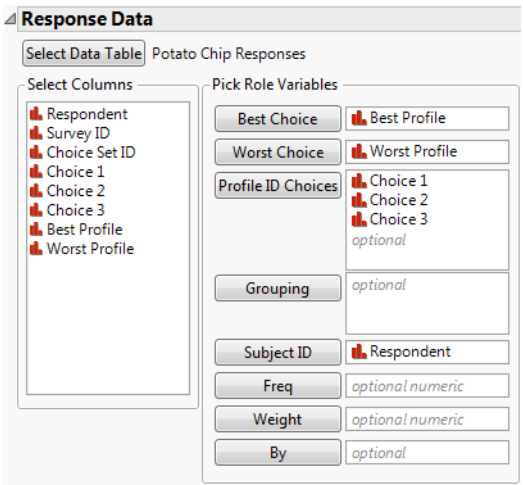
   Note that Product Of is another profile effect that you could add to the effects list.

Figure 5.5 Complete Profile Data Outline



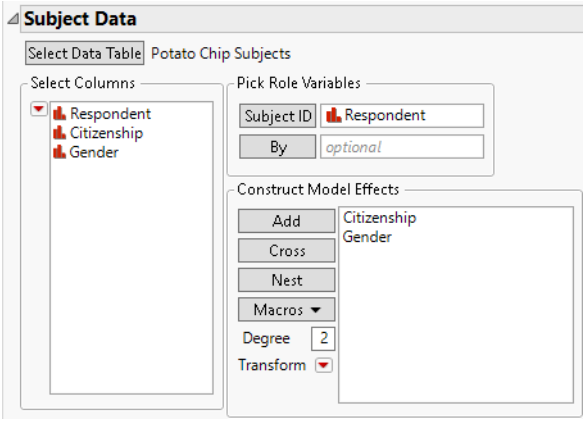
9. Open the Response Data outline. Click **Select Data Table**.
10. Select Potato Chip Responses and click **OK**.
11. Assign roles to columns to complete the launch dialog:
    - Select Best Profile and click **Best Choice**.
    - Select Worst Profile and click **Worst Choice**.
    - Select Choice 1, Choice 2, and Choice 3 and click **Profile ID Choices**.
    - Select Respondent and click **Subject ID**.

Figure 5.6 Completed Response Data Outline



12. Open the Subject Data outline. Click **Select Data Table**.
13. Select Potato Chip Subjects and click **OK**.
14. Select Respondent and click **Subject ID**.
15. Select Citizenship and Gender and click **Add** under **Construct Model Effects**.

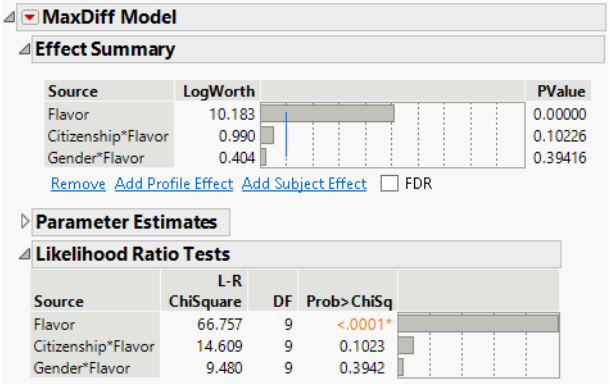
Figure 5.7 Completed Subject Data Outline



Explore the Model

1. Click **Run Model**.

Figure 5.8 MaxDiff Model Report



The Effect Summary report shows the terms in the model and gives *p*-values for their significance. Notice that Flavor is a profile effect, and that each of Citizenship*Flavor and Gender*Flavor is an interaction of a subject and a profile effect.

The Likelihood Ratio Tests report indicates that Flavor is significant.

Launch the MaxDiff Platform

Launch the MaxDiff platform by selecting **Analyze > Consumer Research > MaxDiff**.

Your data for the MaxDiff platform can be combined in a single data table or it can reside in two or three separate data tables. In the MaxDiff launch window, specify whether you are using one or multiple data tables in the Data Format list.

One Table, Stacked

For the One Table, Stacked format, the data are in a single data table. There is a row for every profile presented to a subject within a choice set and an indicator for the best and worst profiles in that choice set. The Potato Chip Combined.jmp sample data table contains the results of a MaxDiff experiment in a single table format. See "One Table Format".

For more information about the launch window for this format, see "Launch Window for One Table, Stacked".

Multiple Tables, Cross-Referenced

For the Multiple Tables, Cross-Referenced format, the data are in two or three separate data tables. A profile data table and a response data table are required. A subject data table is optional. Note the following:

- The profile data table must contain a column with a unique identifier for each profile and columns for the profile level variables. The profile identifier is used in the response data table to identify best and worst profile responses for each choice set.
- The optional subject data table must contain a column with a unique subject identifier for each subject and columns for the subject level variables. The subject identifier is used in the response data table to identify the subjects.

The launch window for this format contains three sections: Profile Data, Response Data, and Subject Data. Each section corresponds to a different data table. You can expand or collapse each section as needed.

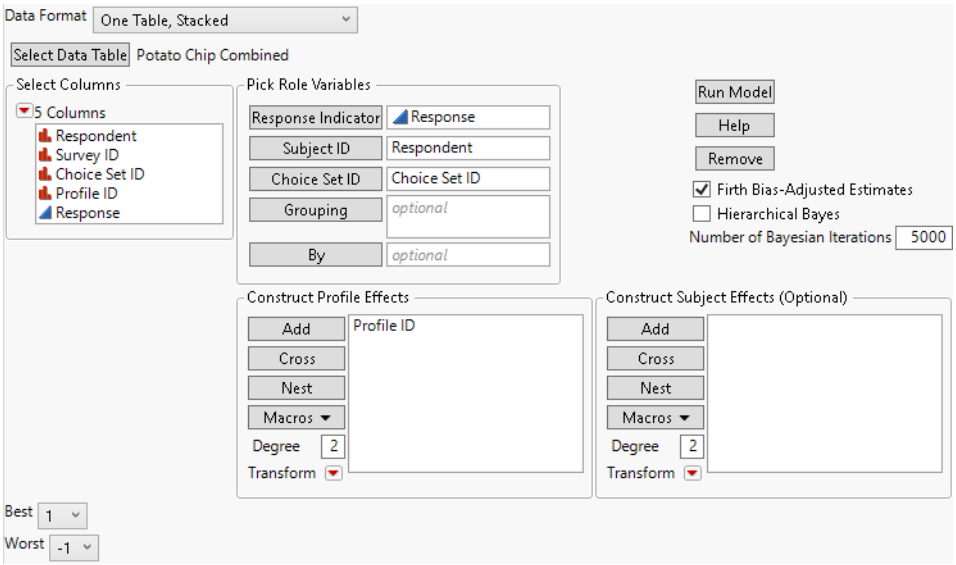
The Potato Chip Profiles.jmp, Potato Chip Responses.jmp, and Potato Chip Subjects.jmp sample data tables contain the results of a MaxDiff experiment using three tables. See "Multiple Table Format".

For more information about the launch window for this format, see "Launch Window for Multiple Tables, Cross-Referenced".

Launch Window for One Table, Stacked

Launch the MaxDiff platform by selecting **Analyze > Consumer Research > MaxDiff**. For one table select **One Table, Stacked** from the Data Format menu.

Figure 5.9 Launch Window for One Table, Stacked Data Format



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Select Data Table Select or open the data table that contains the combined data. Select Other to open a file that is not already open.

Response Indicator A column containing the preference data. Use two of the values 1, -1, or 0 for the Best and Worst choices, and the third value for profiles that are not Best or Worst. The default coding is a 1 to indicate the Best choice and a -1 for the Worst choice.

Subject ID An identifier for the study participant.

Choice Set ID An identifier for the set of profiles presented to the subject for a given preference determination.

Grouping A column which, when used with the Choice Set ID, uniquely designates each choice set. For example, if a choice set has Choice Set ID = 1 for Survey = A, and another choice set has Choice Set ID = 1 for Survey = B, then Survey should be used as a **Grouping** column.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of By variables.

Construct Profile Effects Add effects constructed from the attributes for the profiles.

For information about the Construct Profile Effects panel, see *Fitting Linear Models*.

Construct Subject Effects (Optional) Add effects constructed from subject-related factors.

For information about the Construct Subject Effects panel, see *Fitting Linear Models*.

Firth Bias-adjusted Estimates Computes bias-corrected MLEs that produce better estimates and tests than MLEs without bias correction. These estimates also improve separation problems that tend to occur in logistic-type models. See Heinze and Schemper (2002) for a discussion of the separation problem in logistic regression.

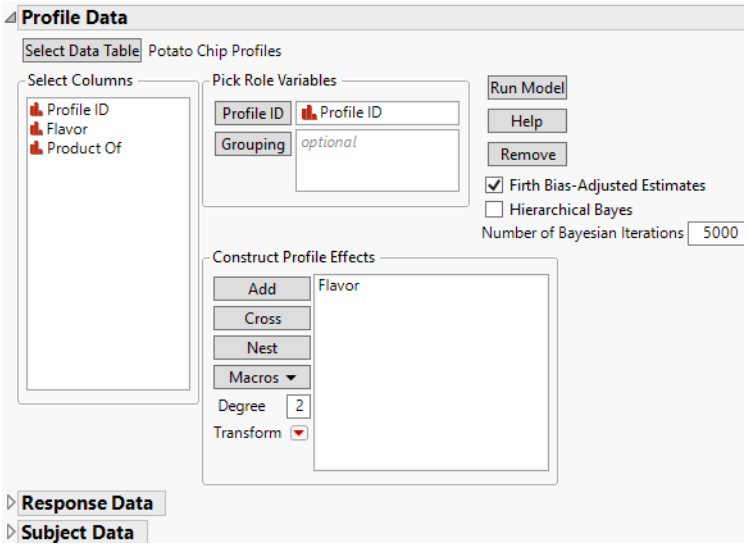
JMP PRO **Hierarchical Bayes** Uses a Bayesian approach to estimate subject-specific parameters. See "Bayesian Parameter Estimates".

JMP PRO **Number of Bayesian Iterations** (Applicable only if Hierarchical Bayes is selected.) The total number of iterations of the adaptive Bayes algorithm used to estimate subject effects. This number includes a burn-in period of iterations that are discarded. The number of burn-in iterations is equal to half of the Number of Bayesian Iterations specified on the launch window.

Launch Window for Multiple Tables, Cross-Referenced

Launch the MaxDiff platform by selecting **Analyze > Consumer Research > MaxDiff**. For multiple tables select **Multiple Tables, Cross-Referenced** form the Data Format menu.

Figure 5.10 Launch Window for Multiple Tables, Cross-Referenced Data Format



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

In the case of Multiple Tables, Cross-Referenced, the launch window has three sections:

- "Profile Data"
- "Response Data"
- "Subject Data"

Profile Data

The profile data table describes the attributes associated with each choice. Each choice can comprise many different attributes, and each attribute is listed as a column in the data table. There is a row for each possible choice, and each possible choice contains a unique ID.

Select Data Table Select or open the data table that contains the profile data. Select Other to open a file that is not already open.

Profile ID Identifier for each row of choice combinations. If the **Profile ID** column does not uniquely identify each row in the profile data table, you need to add **Grouping** columns. Add **Grouping** columns until the combination of **Grouping** and **Profile ID** columns uniquely identifies the row, or profile.

Grouping A column which, when used with the Choice Set ID column, uniquely designates each choice set. For example, if Profile ID = 1 for Survey = A, and a different Profile ID = 1 for Survey = B, then Survey would be used as a **Grouping** column.

Construct Profile Effects Add effects constructed from the attributes in the profiles.

For information about the Construct Profile Effects panel, see *Fitting Linear Models*.

Firth Bias-adjusted Estimates Computes bias-corrected MLEs that produce better estimates and tests than MLEs without bias correction. These estimates also improve separation problems that tend to occur in logistic-type models. See Heinze and Schemper (2002) for a discussion of the separation problem in logistic regression.

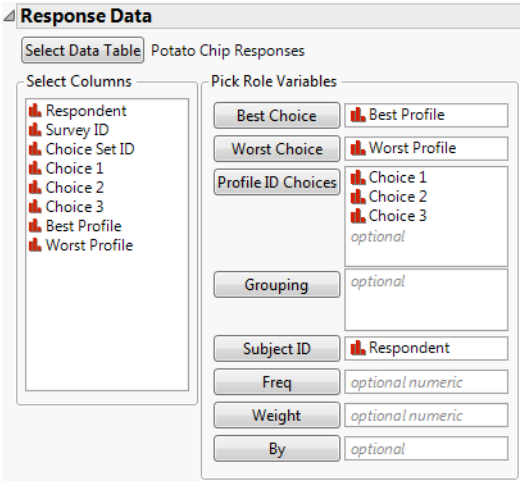
JMP PRO **Hierarchical Bayes** Uses a Bayesian approach to estimate subject-specific parameters. See "Bayesian Parameter Estimates".

JMP PRO **Number of Bayesian Iterations** (Applicable only if Hierarchical Bayes is selected.) The total number of iterations of the adaptive Bayes algorithm used to estimate subject effects. This number includes a burn-in period of iterations that are discarded. The number of burn-in iterations is equal to half of the Number of Bayesian Iterations specified on the launch window.

Response Data

Figure 5.11 shows the Response Data outline populated using Potato Chip Responses.jmp.

Figure 5.11 Response Data Outline



The response data table contains the study results. It gives the choice set IDs for each trial as well as the profiles selected as best and worst by the subject. The Response data are linked to the Profile data through the choice set columns and the choice response column. Grouping variables can be used to align choice indices when more than one group is contained within the data.

Select Data Table Select or open the data table that contains the response data. Select Other to open a file that is not already open.

Best Choice The Response table column containing the Profile ID of the profile that the study participant designated as Best.

Worst Choice The Response table column containing the Profile ID of the profile that the study participant designated as Worst.

Profile ID Choices The columns that contain the **Profile ID**s of the set of possible choices for each choice set. There must be at least three profiles.

Grouping A column which, when used with the Profile ID Chosen column, uniquely designates each choice set.

Subject ID A unique identifier for the study participant.

Freq A column containing frequencies. If n is the value of the Freq variable for a given row, then that row is used in computations n times. If it is less than 1 or missing, then JMP does not use it to calculate any analyses.

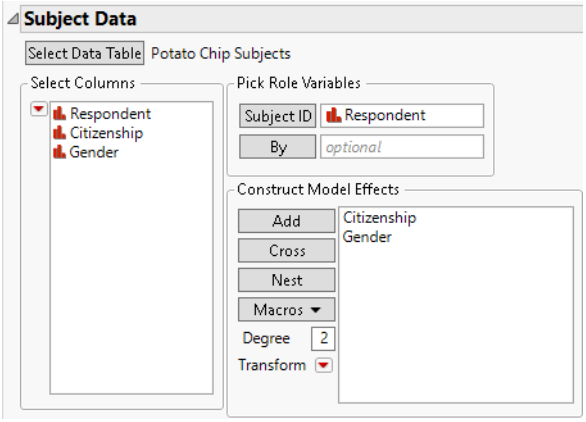
Weight A column containing a weight for each observation in the data table. The weight is included in analyses only when its value is greater than zero.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of By variables.

Subject Data

Figure 5.12 shows the Subject Data outline populated using Potato Chip Subjects.jmp.

Figure 5.12 Subject Data Outline



Note: A subject data table is optional, depending on whether subject effects are to be modeled.

The subject data table contains the Subject ID and one or more columns of attributes or characteristics for each subject. The subject data table contains the same number of rows as subjects and has an identifier column that matches a similar column in the Response data table.

Note: You can include subject data in the response data table, but you need to specify subject effects in the Subject Data outline.

Select Data Table Select or open the data table that contains the subject data. Select Other to open a file that is not already open.

Subject ID Unique identifier for the subject.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of By variables.

Construct Model Effects Add effects constructed from columns in the subject data table.

For information about the Construct Model Effects panel, see *Fitting Linear Models*.

MaxDiff Model Report

The MaxDiff Model window shows some of the following reports by default, depending on your selections in the launch window.

- "Effect Summary"
- "MaxDiff Results"
- "Parameter Estimates"
- "Bayesian Parameter Estimates"
- "Likelihood Ratio Tests"

For descriptions of the platform options, see "MaxDiff Platform Options".

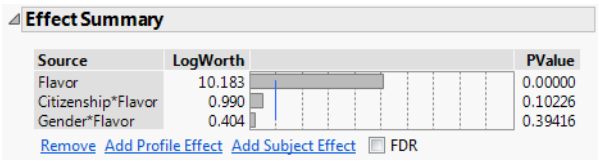
Effect Summary

The Effect Summary report appears if your model contains more than one effect. It lists the effects estimated by the model and gives a plot of the LogWorth (or FDR LogWorth) values for these effects. The report also provides controls that enable you to add or remove effects from the model. The model fit report updates automatically based on the changes made in the Effects Summary report. See *Fitting Linear Models*.

Note: The Effect Summary report is not applicable to models fit with Hierarchical Bayes.

Figure 5.13 shows the Effect Summary report obtained by running the script **MaxDiff for Flavor** in Potato Chip Responses.jmp.

Figure 5.13 Effect Summary Report



Effect Summary Table Columns

The Effect Summary table contains the following columns:

Source Lists the model effects, sorted by ascending *p*-values.

LogWorth Shows the LogWorth for each model effect, defined as $-\log_{10}(p\text{-value})$. This transformation adjusts p-values to provide an appropriate scale for graphing. A value that exceeds 2 is significant at the 0.01 level (because $-\log_{10}(0.01) = 2$).

FDR LogWorth Shows the False Discovery Rate LogWorth for each model effect, defined as $-\log_{10}(\text{FDR PValue})$. This is the best statistic for plotting and assessing significance. Select the **FDR** check box to replace the LogWorth column with the **FDR LogWorth** column.

Bar Chart Shows a bar chart of the LogWorth (or FDR LogWorth) values. The graph has dashed vertical lines at integer values and a blue reference line at 2.

PValue Shows the p-value for each model effect. This is the p-value corresponding to the significance test displayed in the Likelihood Ratio Tests report.

FDR PValue Shows the False Discovery Rate p-value for each model effect calculated using the Benjamini-Hochberg technique. This technique adjusts the p-values to control the false discovery rate for multiple tests. Select the **FDR** check box to replace the **PValue** column with the **FDR PValue** column.

For more information about the FDR correction, see Benjamini and Hochberg (1995). For more information about the false discovery rate, see *Predictive and Specialized Modeling* or Westfall et al. (2011).

Effect Summary Table Options

The options below the summary table enable you to add and remove effects:

Remove Removes the selected effects from the model. To remove one or more effects, select the rows corresponding to the effects and click the Remove button.

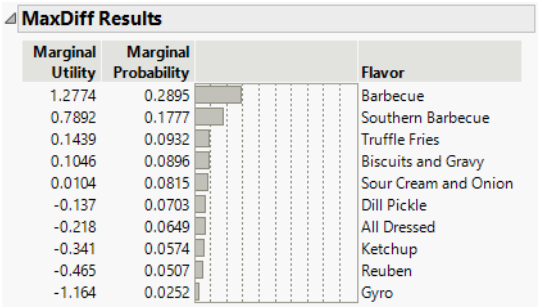
Add Profile Effect Opens a column dialog that contains a list of all columns in the data table for the OneTable, Stacked data format, and for the columns in the Profile Data table for the Multiple Tables, Cross-Referenced data format. Select columns that you want to add to the model, and then click Add below the column selection list to add the columns to the model. Click Close to close the panel.

Add Subject Effect Opens a column dialog that contains a list of all columns in the data table for the OneTable, Stacked data format, and for the columns in the Subject Data table for the Multiple Tables, Cross-Referenced data format. Select columns that you want to add to the model, and then click Add below the column selection list to add the columns to the model. Click Close to close the panel.

MaxDiff Results

Figure 5.14 shows the MaxDiff Results report obtained by running the script **MaxDiff with No Subject Effects** in Potato Chip Responses.jmp.

Figure 5.14 MaxDiff Results Report



MaxDiff Results

Marginal Utility	Marginal Probability	Flavor
1.2774	0.2895	Barbecue
0.7892	0.1777	Southern Barbecue
0.1439	0.0932	Truffle Fries
0.1046	0.0896	Biscuits and Gravy
0.0104	0.0815	Sour Cream and Onion
-0.137	0.0703	Dill Pickle
-0.218	0.0649	All Dressed
-0.341	0.0574	Ketchup
-0.465	0.0507	Reuben
-1.164	0.0252	Gyro

For each Profile effect specified in the launch window, the following are displayed:

Marginal Utility An indicator of the perceived value of the corresponding level of the effect. Larger values suggest that the feature is of greater value.

Marginal Probability The estimated probability that a subject expresses a preference for the corresponding level of the effect over all other levels. For each effect, the marginal probabilities sum to one.

Bar Chart Shows a bar chart of the marginal probabilities.

Effect Column Gives the name of the effect and a list of its levels. The levels define the features to which the Marginal Utility and Marginal Probability estimates apply.

Parameter Estimates

This report gives details about parameter estimates, fit criteria, and the fitting algorithm.

Figure 5.15 shows the Parameter Estimates report obtained by running the script **MaxDiff for Flavor** in Potato Chip Responses.jmp.

Figure 5.15 Parameter Estimates Report

Parameter Estimates

Term	Estimate	Std Error
Flavor[All Dressed]	-0.15616234	0.2227718641
Flavor[Barbecue]	1.22210326	0.2978972284
Flavor[Biscuits and Gravy]	0.16398758	0.2249076601
Flavor[Dill Pickle]	-0.17356828	0.2151668801
Flavor[Gyro]	-1.11927509	0.2827016652
Flavor[Ketchup]	-0.47308393	0.2319900823
Flavor[Reuben]	-0.50927309	0.2294979936
Flavor[Sour Cream and Onion]	0.21115573	0.2450361368
Flavor[Southern Barbecue]	0.70149945	0.2695322771
Citizenship[Canadian]*Flavor[All Dressed]	-0.04368106	0.2239442590
Citizenship[Canadian]*Flavor[Barbecue]	-0.16180196	0.2978036763
Citizenship[Canadian]*Flavor[Biscuits and Gravy]	0.05734172	0.2233312021
Citizenship[Canadian]*Flavor[Dill Pickle]	-0.09824391	0.2188896544
Citizenship[Canadian]*Flavor[Gyro]	0.43257276	0.2907051874
Citizenship[Canadian]*Flavor[Ketchup]	-0.38035261	0.2349720398
Citizenship[Canadian]*Flavor[Reuben]	-0.34677939	0.2342643217
Citizenship[Canadian]*Flavor[Sour Cream and Onion]	0.56678250	0.2355493683
Citizenship[Canadian]*Flavor[Southern Barbecue]	-0.00712532	0.2720518372
Gender[Female]*Flavor[All Dressed]	-0.26955535	0.2106170508
Gender[Female]*Flavor[Barbecue]	0.39881410	0.2961368976
Gender[Female]*Flavor[Biscuits and Gravy]	0.09184462	0.2245258406
Gender[Female]*Flavor[Dill Pickle]	-0.11461679	0.2131955199
Gender[Female]*Flavor[Gyro]	-0.40538827	0.2950786212
Gender[Female]*Flavor[Ketchup]	-0.07317652	0.2135446256
Gender[Female]*Flavor[Reuben]	0.20780054	0.2227994637
Gender[Female]*Flavor[Sour Cream and Onion]	0.27353428	0.2337547848
Gender[Female]*Flavor[Southern Barbecue]	0.07570279	0.2627935035

AICc	397.44423
BIC	456.27172
-2*LogLikelihood	327.00945
-2*Firth LogLikelihood	244.39836

Converged in Gradient
Firth Bias-Adjusted Estimates

Term Lists the terms in the model.

Estimate An estimate of the parameter associated with the corresponding term. In discrete choice experiments, parameter estimates are sometimes referred to as *part-worths*. Each part-worth is the coefficient of utility associated with the given term. By default, these estimates are based on the Firth bias-corrected maximum likelihood estimators and therefore are considered to be more accurate than MLEs without bias correction.

Std Error An estimate of the standard deviation of the parameter estimate.

Comparison Criteria

The AICc (corrected Akaike's Information Criterion), BIC (Bayesian Information Criterion), –2Loglikelihood, and –2Firth Loglikelihood fit statistics are shown as part of the report and can be used to compare models. See *Fitting Linear Models* for more information about the first three of these measures.

The –2Firth Loglikelihood value is included in the report only when the Firth Bias-adjusted Estimates check box is checked in the launch window. This option is checked by default.

For each of these statistics, a smaller value indicates a better fit.

JMP PRO Bayesian Parameter Estimates

(Appears only if Hierarchical Bayes is selected on the launch window.) The Bayesian Parameter Estimates report gives results for model effects. The estimates are based on a Hierarchical Bayes fit that integrates the subject-level covariates into the likelihood function and estimates their effects on the parameters directly. The subject-level covariates are estimated using a version of the algorithm described in Train (2001), which incorporates Adaptive Bayes and Metropolis-Hastings approaches. Posterior means and variances are calculated for each model effect. The algorithm also provides subject-specific estimates of the model effect parameters. See "Save Subject Estimates".

During the estimation process, each individual is assigned his or her own vector of parameter estimates, essentially treating the estimates as random effects and covariates. The vector of coefficients for an individual is assumed to come from a multivariate normal distribution with arbitrary mean and covariance matrix. The likelihood function for the utility parameters for a given subject is based on a multinominal logit model for each subject's preference within a choice set, given the attributes in the choice set. The prior distribution for a given subject's vector of coefficients is normal with mean equal to zero and a diagonal covariance matrix with the same variance for each subject. The covariance matrix is assumed to come from an inverse Wishart distribution with a scale matrix that is diagonal with equal diagonal entries.

For each subject, a number of burn-in iterations at the beginning of the chain is discarded. By default, this number is equal to half of the Number of Bayesian Iterations specified on the launch window.

Figure 5.16 Bayesian Parameter Estimates Report

Bayesian Parameter Estimates

Term	Posterior Mean	Posterior Std Dev	Subject Std Dev
Product Of[Canada]	-0.444803533	0.1857067541	0.2930798802
Citizenship[Canadian]*Product Of[Canada]	-0.103783512	0.2338637067	0.1945040440
Gender[Female]*Product Of[Canada]	-0.258577340	0.2205559716	0.1846953243

Total Iterations	5000
Burn-In Iterations	2500
Number of Respondents	30
Avg Log Likelihood After Burn-In	-196.5618

Term The model term.

Posterior Mean The parameter estimate for the term's coefficient. For each iteration after the burn-in period, the mean of the subject-specific coefficient estimates is computed. The Posterior Mean is the average of these means.

Tip: Select the red triangle option Save Bayes Chain to see the individual estimates for each iteration.

Posterior Std Dev The standard deviation of the means of the subject-specific estimates over the iterations after burn-in.

Subject Std Dev The standard deviation of the subject-specific estimates around the posterior mean.

Tip: Select the red triangle option Save Subject Estimates to see the individual estimates.

Total Iterations The total number of iterations performed, including the burn-in period.

Burn-In Iterations The number of burn-in iterations, which are discarded. This number is equal to half of the Number of Bayesian Iterations specified on the launch window.

Number of Respondents The number of subjects

Avg Log Likelihood After Burn-In The average of the log-likelihood function, computed on values obtained after the burn-in period.

Likelihood Ratio Tests

Figure 5.17 shows the Likelihood Ratio Tests report obtained by running the script **MaxDiff for Flavor** in Potato Chip Responses.jmp.

Figure 5.17 Likelihood Ratio Tests

Likelihood Ratio Tests

Source	L-R ChiSquare	DF	Prob>ChiSq
Flavor	66.757	9	<.0001*
Citizenship*Flavor	14.609	9	0.1023
Gender*Flavor	9.480	9	0.3942

Source Lists the effects in the model.

L-R ChiSquare The value of the likelihood ratio ChiSquare statistic for a test of the corresponding effect.

DF The degrees of freedom for the ChiSquare test.

Prob>ChiSq The *p*-value for the ChiSquare test.

Bar Chart Shows a bar chart of the L-R ChiSquare values.

MaxDiff Platform Options

The MaxDiff Model red triangle menu contains the following options:

JMP PRO **Show MLE Parameter Estimates** (Available for Hierarchical Bayes.) Shows non-Firth maximum likelihood estimates and standard errors for the coefficients of model terms. These estimates are used as starting values for the Hierarchical Bayes algorithm.

Joint Factor Tests (Not available for Hierarchical.) Tests each factor in the model by constructing a likelihood ratio test for all the effects involving that factor. For more information about Joint Factor Tests, see *Fitting Linear Models*.

Confidence Intervals (Not available for Hierarchical Bayes) Shows or hides a confidence interval for each parameter in the Parameter Estimates report.

Confidence Limits (Available for Hierarchical Bayes) Shows or hides confidence limits for each parameter in the Bayesian Parameter Estimates report. The limits are constructed based on the 2.5 and 97.5 quantiles of the posterior distribution.

Correlation of Estimates If Hierarchical Bayes was not selected, shows or hides the correlations between the maximum likelihood parameter estimates.

For Hierarchical Bayes, shows or hides the correlation matrix for the posterior means of the parameter estimates. The correlations are calculated from the iterations after burn-in. The posterior means from each iteration after burn-in are treated as if they are columns in a data table. The Correlation of Estimates table is obtained by calculating the correlation matrix for these columns.

Comparisons Performs comparisons between specific alternative choice profiles. Enables you to select factor values and the values that you want to compare. You can compare specific configurations, including comparing all settings on the left or right by selecting the **Any** check boxes. Using **Any** does not compare all combinations across features, but rather all combinations of comparisons, one feature at a time, using the left settings as the settings for the other factors. See "Comparisons Report".

All Levels Comparison Report Shows the All Levels Comparison Report, which contains a table with information about all pairwise comparisons of profiles. If you are modeling subject effects, you must specify a combination of subject effects and the table is specific to that combination of subject effects. Each cell of the table shows the difference in utilities for the row level and column level, the standard error of the difference, and a Wald *p*-value for a test of no difference.

Caution: The *p*-values are not corrected for multiple comparisons. Use these results as a guide.

The Wald *p*-values are colored. A saturated blue (respectively, red) color indicates that the Difference (Row - Column) is negative (respectively positive). The intensity of the red and blue coloring indicates the degree of significance.

Save Utility Formula When the analysis is on multiple data tables, creates a new data table that contains a formula column for utility. The new data table contains a row for each subject and profile combination, and columns for the profiles and the subject effects. When the analysis is on one data table, a new Utility Formula column is added.

Save Gradients by Subject (Not available for Hierarchical Bayes.) Constructs a new table that has a row for each subject containing the average (Hessian-scaled-gradient) steps for the likelihood function on each parameter. This corresponds to using a Lagrangian multiplier test for separating that subject from the remaining subjects. These values can later be clustered, using the built-in-script, to indicate unique market segments represented in the data. See "Example of Segmentation".

Note: When a subject gradient is non-estimable it is set to missing in the Gradient by Subject table.

JMP PRO **Save Subject Estimates** (Available for Hierarchical Bayes.) Creates a table where each row contains the subject-specific parameter estimates for each effect. The distribution of subject-specific parameter effects for each effect is centered at the estimate for the term given in the Bayesian Parameter Estimates report. The Subject Acceptance Rate gives the rate of acceptance for draws of new parameter estimates during the Metropolis-Hastings step. Generally, an acceptance rate of 0.20 is considered to be good. See "Bayesian Parameter Estimates".

JMP PRO **Save Bayes Chain** (Available for Hierarchical Bayes.) Creates a table that gives information about the chain of iterations used in computing subject-specific Bayesian estimates. See "Save Bayes Chain".

Model Dialog Shows the MaxDiff launch window that resulted in the current analysis, which can be used to modify and re-fit the model. You can specify new data sets, new IDs, and new model effects.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Save By-Group Script Contains options that enable you to save a script that reproduces the platform report for all levels of a By variable to several destinations. Available only when a By variable is specified in the launch window.

Comparisons Report

The Comparisons report is shown when you specify pairwise comparisons. It contains the following columns:

Factor Shows the levels of the subject factors that you specified.

Compared 1 Shows the factor and levels for the profile variables in the first component of the comparison.

Compared 2 Shows the factor and levels for the profile variables in the second component of the comparison.

Utility 1 Shows the estimated utility of the first component for the subjects specified in the Factor column.

Utility 2 Shows the estimated utility of the second component for the subjects specified in the Factor column.

Probability 1 Shows the predicted probability that the first component is preferred to the second for the subjects specified in the Factor column.

Probability 2 Shows the predicted probability that the second component is preferred to the first for the subjects specified in the Factor column.

Odds 1 Probability 1 divided by Probability 2.

Odds 2 Probability 2 divided by Probability 1.

Comparison Difference Utility 1 minus Utility 2.

Standard Deviation The sample standard error of the estimated Comparison Difference.

JMP PRO Save Bayes Chain

(Available for models fit with Hierarchal Bayes.) You can use the Bayes Chain data table to determine whether your estimates have stabilized. The table that is created has a number of rows equal to the Number of Bayesian Iterations (specified on the launch window) plus one. The first row, Iteration 1, gives the starting values. Subsequent rows show the results of the iterations, in order. The table has a column for the iteration counts, the model Log Likelihood, and columns corresponding to each model effect:

Iteration Gives the iteration number, where the first row shows starting values.

Log Likelihood The log-likelihood of the model for that iteration. You can plot the Log Likelihood against Iteration to view behavior over the burn-in and tuning periods.

Adaptive Sigma for <model effect> Gives the estimate of the square root of the diagonal entries of the inverse Wishart distribution scale matrix for the corresponding effect.

Acceptance for <model effect> Gives the sampling acceptance rate for the corresponding effect.

Mean of <model effect> Gives the estimated mean for the corresponding effect.

Variance of <model effect> Gives the estimated variance for the corresponding effect.

Chapter 6

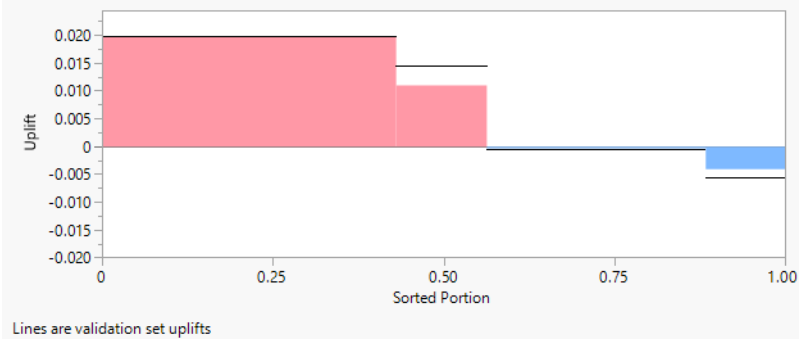
JMP PRO Uplift

Model the Incremental Impact of Actions on Consumer Behavior

The Uplift platform is available only in JMP Pro.

Use uplift modeling to optimize marketing decisions, to define personalized medicine protocols, or, more generally, to identify characteristics of individuals who are likely to respond to an intervention. Also known as incremental modeling, true lift modeling, or net modeling, uplift modeling differs from traditional modeling techniques in that it finds the interactions between a treatment and other variables. It directs focus to individuals who are likely to react positively to an action or treatment.

Figure 6.1 Example of Uplift for a Hair Product Marketing Campaign



Contents

Overview of the Uplift Platform 169
Example of the Uplift Platform 170
Launch the Uplift Platform 172
The Uplift Model Report 173
Uplift Model Graph 173
Uplift Report Options 176

JMP PRO Overview of the Uplift Platform

Use the Uplift platform to model the incremental impact of an action, or *treatment*, on individuals. An uplift model helps identify groups of individuals who are most likely to respond to the action. Identification of these groups leads to efficient and targeted decisions that optimize resource allocation and impact on the individual. See Radcliffe and Surry (2011).

The Uplift platform fits partition models. Although traditional partition models select splits to optimize classification, uplift models select splits to maximize treatment differences.

The uplift partition model accounts for the fact that individuals are grouped by a treatment factor. To determine splits, models are fit to all possible binary splits of each factor. The type of model that is fit is dependent on the type of response. A continuous response is modeled as a linear function of the split, the treatment, and the interaction of the split and treatment. A categorical response is expressed as a logistic function of the split, the treatment, and the interaction of the split and treatment. In either case, the interaction term measures the difference in uplift between the groups of individuals in the two splits. The most significant split is selected and the process repeats.

The Uplift platform selects the most significant split based on the significance of interaction terms in each of the binary split models. However, predictor selection based solely on *p*-values introduces bias in favor of predictors with many levels that result in many models for the single predictor. For this reason, JMP adjusts *p*-values to account for the number of levels or models considered. The correction used is based on Monte Carlo simulation. See Sall (2002). The splits are determined by the minimum adjusted *p*-values for tests of the significance of the interaction effect across models for all possible binary splits across all predictors. The logworth for each adjusted *p*-value, namely $-\log_{10}(\text{adj } p\text{-value})$, is reported.

JMP PRO Example of the Uplift Platform

The Hair Care Product.jmp sample data table results from a marketing campaign designed to increase purchases of a hair coloring product targeting both genders. For purposes of designing the study and tracking purchases, 126,184 "club card" members of a major beauty supply chain were identified. Approximately half of these members were randomly selected and sent a promotional offer for the product. Purchases of the product over a subsequent three-month period by all club card members were tracked.

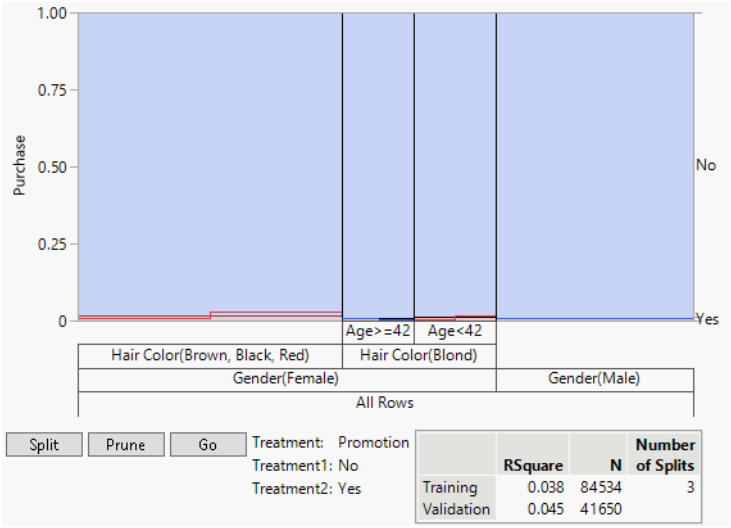
The data table shows a Promotion column, indicating whether the member received promotional material. The column Purchase indicates whether the member purchased the product over the test period. For each member, Gender, Age, Hair Color (natural), U.S. Region, and Residence (whether the member is located in an urban area) was assembled. Also shown is a Validation column consisting of about 33% of the subjects.

For a categorical response, the Uplift platform interprets the first level in its value ordering as the response of interest. This is why the column Purchase has the Value Order column property. This property ensures that "Yes" responses are first in the ordering.

1. Select **Help > Sample Data Library** and open Hair Care Product.jmp.
2. Select **Analyze > Consumer Research > Uplift**.
3. From the Select Columns list:
   - Select Promotion and click **Treatment**.
   - Select Purchase and click **Y, Response**.
   - Select Gender, Age, Hair Color, U.S. Region, and Residence, and click **X, Factor**.
   - Select Validation and click **Validation**.
4. Click **OK**.
5. Below the Graph in the report that appears, click **Go**.

   Based on the validation set, the optimal Number of Splits is determined to be three. Note that the left vertical scale is locked in order to maintain the overall rates of outcomes displayed on the right vertical axis.

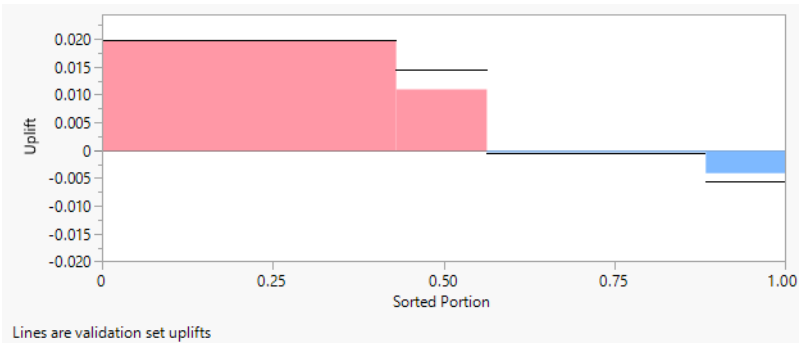
Figure 6.2 Graph after Three Splits



The right hand vertical axis in the graph indicates that the proportion of purchases is small compared to non-purchases. The graph shows that uplift in purchases occurs for females with black, red, or brown hair and for younger females (Age < 42) with blond hair. For older blond-haired women (Age ≥ 42) and males, the promotion has a negative effect.

6. Click the Uplift Model for Purchase red triangle and select **Uplift Graph**.

Figure 6.3 Uplift Graph

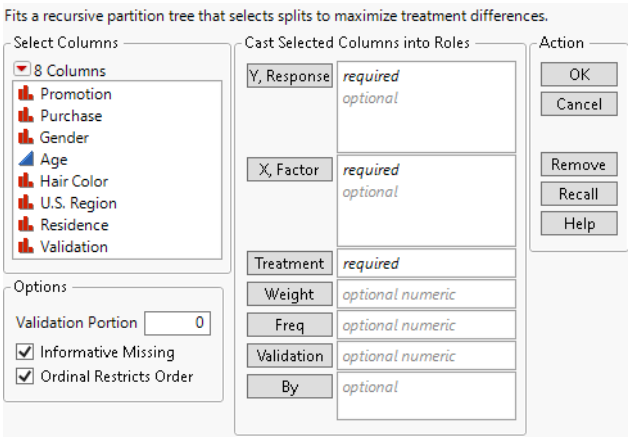


Notice that for two groups of subjects (males and non-blond women in the Age ≥ 42 group), the promotion has a negative effect. The horizontal lines shown on the Uplift Graph delineate the graph for the validation set. Specifically, the decision tree is evaluated for the validation set and the Uplift Graph is constructed from the estimated uplifts.

JMP PRO Launch the Uplift Platform

Launch the Uplift platform by selecting **Analyze > Consumer Research > Uplift**.

Figure 6.4 Uplift Platform Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Y, Response Assigns one or more columns to be analyzed.

X, Factor Assigns one or more columns to be used as factors.

Treatment Assigns a categorical treatment column. If the treatment column contains more than two levels, the first level is treated as one treatment level and the remaining levels are combined into a second treatment level.

Weight Assigns a numeric column that contains a weight for each observation in the data table. A row is included in the analysis only when its weight is greater than zero.

Freq Assigns a frequency variable to this role. This is useful for summarized data.

Validation Assigns a numeric column that defines the validation sets. This column should contain at most three distinct values:

- If there are two values, the smaller value defines the training set and the larger value defines the validation set.
- If there are three values, these values define the training, validation, and test sets in order of increasing size.

– If the validation column has more than three levels, the rows that contain the smallest three values define the validation sets. All other rows are excluded from the analysis.

The Uplift platform uses the validation column to train and tune the model or to train, tune, and evaluate the model. For more information about validation, see *Predictive and Specialized Modeling*.

If you click the Validation button with no columns selected in the Select Columns list, you can add a validation column to your data table. See *Predictive and Specialized Modeling*.

By Produces a separate report for each level of the By variable. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variable.

Validation Portion The portion of the data to be used as a validation set. Enter a value between 0 and 1.

Informative Missing If selected, enables missing value categorization for categorical predictors and informative treatment of missing values for continuous predictors.

Ordinal Restricts Order If selected, restricts consideration of splits to those that preserve the ordering.

JMP PRO The Uplift Model Report

- "Uplift Model Graph"
- "Uplift Report Options"

JMP PRO Uplift Model Graph

The graph represents the response on the vertical axis. The horizontal axis corresponds to observations, arranged by nodes. For each node, a black horizontal line shows the mean response. Within each split, there is a subsplit for treatment shown by a red or blue line. These lines indicate the mean responses for each of the two treatment groups within the split. The value ordering of the treatment column determines the placement order of these lines. As nodes are split, the graph updates to show the splits beneath the horizontal axis. Vertical lines divide the splits.

Beneath the graph are the control buttons: **Split**, **Prune**, and **Go**. The Go button appears only if there is a validation set. Also shown is the name of the Treatment column and its two levels, called Treatment1 and Treatment2. If more than two levels are specified for the Treatment column, all levels except the first are treated as a single level and combined into Treatment2.

To the right of the Treatment column information is a report showing summary values relating to prediction. (Keep in mind that prediction is not the objective in uplift modeling.) The report updates as splitting occurs. If a validation set is used, values are shown for both the training and the validation sets.

RSquare The RSquare for the regression model associated with the tree. Note that the regression model includes interactions with the treatment column. An RSquare closer to 1 indicates a better fit to the data than does an RSquare closer to 0.

Note: A low RSquare value suggests that there might be variables not in the model that account for the unexplained variation. However, if your data are subject to a large amount of inherent variation, even a useful uplift model can have a low RSquare value.

RMSE The root mean square error (RMSE) for the regression model associated with the tree. RMSE is given only for continuous responses. See *Fitting Linear Models*.

N The number of observations.

Number of Splits The number of times splitting has occurred.

AICc The Corrected Akaike Information Criterion (AICc), computed using the associated regression model. AICc is given only for continuous responses. See *Fitting Linear Models*.

JMP PRO Uplift Decision Tree

The decision tree shows the splits used to model uplift. See Figure 6.5 for an example using the Hair Care Product.jmp sample data table. Each node contains the following information:

Treatment The name of the treatment column is shown, with its two levels.

Rate (Appears only for two-level categorical responses.) For each treatment level, the proportion of subjects in this node who responded.

Mean (Appears only for continuous responses.) For each treatment level, the mean response for subjects in this node.

Count The number of subjects in this node in the specified treatment level.

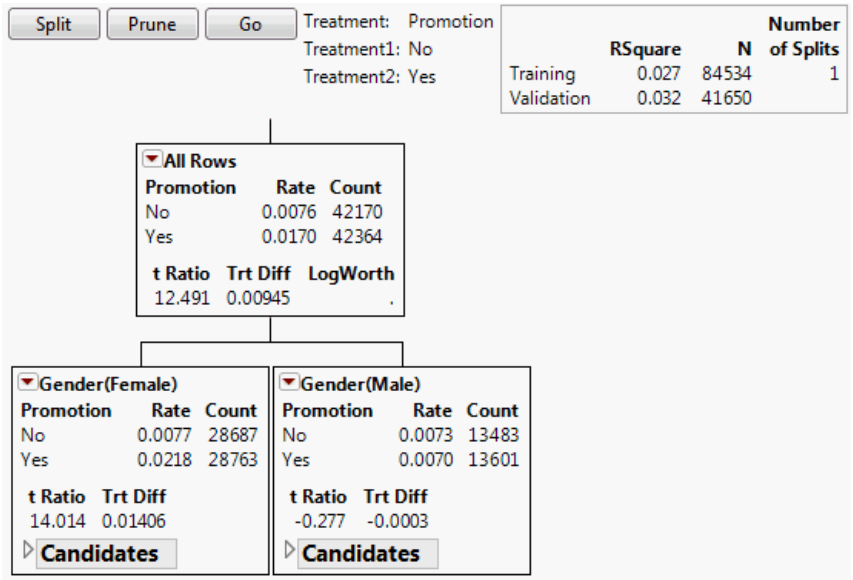
t Ratio The *t* ratio for the test for a difference in response across the levels of Treatment for subjects in this node. If the response is categorical, it is treated as continuous (values 0 and 1) for this test.

Trt Diff The difference in response means across the levels of Treatment. This is the uplift, with the following assumptions:

- The first level in the treatment column's value ordering represents the treatment.
- The response is defined so that larger values reflect greater impact.

LogWorth The value of the logworth for the subsequent split based on the given node.

Figure 6.5 Nodes for First Split



Candidates Report

Each node also contains a Candidates report with additional information:

Term The model term.

LogWorth The maximum logworth over all possible splits for the given term. The logworth corresponding to a split is $-\log_{10}$ of the adjusted p-value.

F Ratio When the response is continuous, this is the F Ratio associated with the interaction term in a linear regression model. The regression model specifies the response as a linear function of the treatment, the binary split, and their interaction. When the response is categorical, this is the ChiSquare value for the interaction term in a nominal logistic model.

Gamma When the response is continuous, this is the coefficient of the interaction term in the linear regression model used in computing the F ratio. When the response is categorical, this is an estimate of the interaction constructed from Firth-adjusted log-odds ratios.

Cut Point If the term is continuous, this is the point that defines the split. If the term is categorical, this describes the first (left) node.

JMP PRO Uplift Report Options

With the exception of the options described below, all of the red triangle options for the Uplift report are described in the documentation for the Partition platform. For more information about these options, see *Predictive and Specialized Modeling*.

JMP PRO Minimum Size Split

This option presents a window where you enter a number or a fractional portion of the total sample size to define the minimum size split allowed. To specify a number, enter a value greater than or equal to 1. To specify a fraction of the sample size, enter a value less than 1. The default value for the Uplift platform is set to 25 or the floor of the number of rows divided by 2,000, whichever value is greater.

JMP PRO Column Uplift Contributions

This table and plot address a column's contribution to the uplift tree structure. A column's contribution is computed as the sum of the F Ratio values associated with its splits. Recall that these values measure the significance of the treatment-by-split interaction term in the regression model.

JMP PRO Uplift Graph

Consider the observations in the training set. Define uplift for an observation as the difference between the predicted probabilities or means across the levels of Treatment for the observation's terminal node. These uplift values are sorted in descending order. On its vertical axis, the Uplift Graph shows the uplift values. On its horizontal axis, the graph shows the proportion of observations with each uplift value.

JMP PRO Save Columns

Save Difference Saves the estimated difference in mean responses across levels of Treatment for the observation's node. This is the estimated uplift.

Save Difference Formula Saves the formula for the Difference, or uplift.

Publish Difference Formula Creates the difference formula and saves it as a formula column script in the Formula Depot platform. If a Formula Depot report is not open, this option creates a Formula Depot report. See *Predictive and Specialized Modeling*.

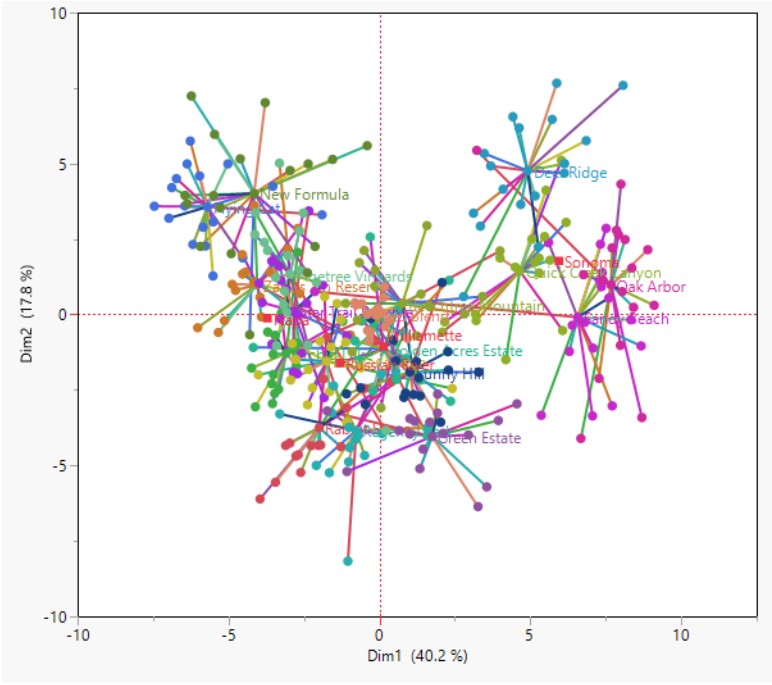
Chapter 7

Multiple Factor Analysis

Analyze Agreement among Panelists

Multiple factor analysis (MFA) is an analytical method closely related to principal components analysis (PCA). MFA uses eigenvalue decomposition to transform multiple measurements on the same items into orthogonal principal components. These components can help you understand how the items are similar and how they are different. MFA uses multiple table or consensus PCA techniques.

Figure 7.1 Consensus Map in Multiple Factor Analysis



Contents

Overview of the Multiple Factor Analysis Platform 179
Example of Multiple Factor Analysis 180
Launch the Multiple Factor Analysis Platform 183
Data Format 184
The Multiple Factor Analysis Report 186
Summary Plots 186
Consensus Map 187
Multiple Factor Analysis Platform Options 187
Statistical Details for the Multiple Factor Analysis Platform 190

Overview of the Multiple Factor Analysis Platform

Multiple factor analysis (MFA) is an analytical method that is closely related to principal components analysis (PCA). However, MFA differs from PCA in that it combines measurements from more than one table. Such tables are sometimes called sub-tables or sub-matrices. Each sub-table has the same number of rows, which represent the items or products being tested. In JMP, sub-tables are represented as groups of columns in a single data table. Each column group is called a block. Note the following about blocks:

- The number of columns in a block can vary. For example, in sensory analysis, a block represents a panelist. Some panelists might rate fewer attributes of a product than other panelists.
- Each block of columns can represent different measurements entirely. MFA scales each block to enable global analysis of all measurements.

The primary goal of MFA is to find groupings of products (rows in a data table) that are similar. A secondary goal is to identify outlier panelists. An outlier panelist results are so different from the rest of the group that they change the study results. Supplementary variables can be used investigate why items group together.

You can use MFA to analyze studies where items are measured on the same or different attributes by different instruments, individuals, or under different circumstances. MFA is frequently used in sensory analysis to account for different measurements among panelists. Traditional sensory analysis can entail hours of up-front training to ensure that panelists' measurements are consistent with each other. For example, consider a juice product with sensory measurements described as "fruity", "sweet", and "refreshing". In traditional sensory analysis, each panelist would have to be trained and tested to make sure reporting on distinct sensory measurements was consistent across panelists. MFA enables the researcher to perform a PCA-like analysis with untrained panelists.

When you use MFA, the same items are measured each time and the measurements can be arranged into internally consistent groups or blocks. For sensory analysis, the rows are the items measured, and the columns are the sensory aspects recorded by each panelist (there is a block for each panelist). Missing observations are replaced by the column mean.

For more information about multiple factor analysis, see Abdi et al. (2013).

Example of Multiple Factor Analysis

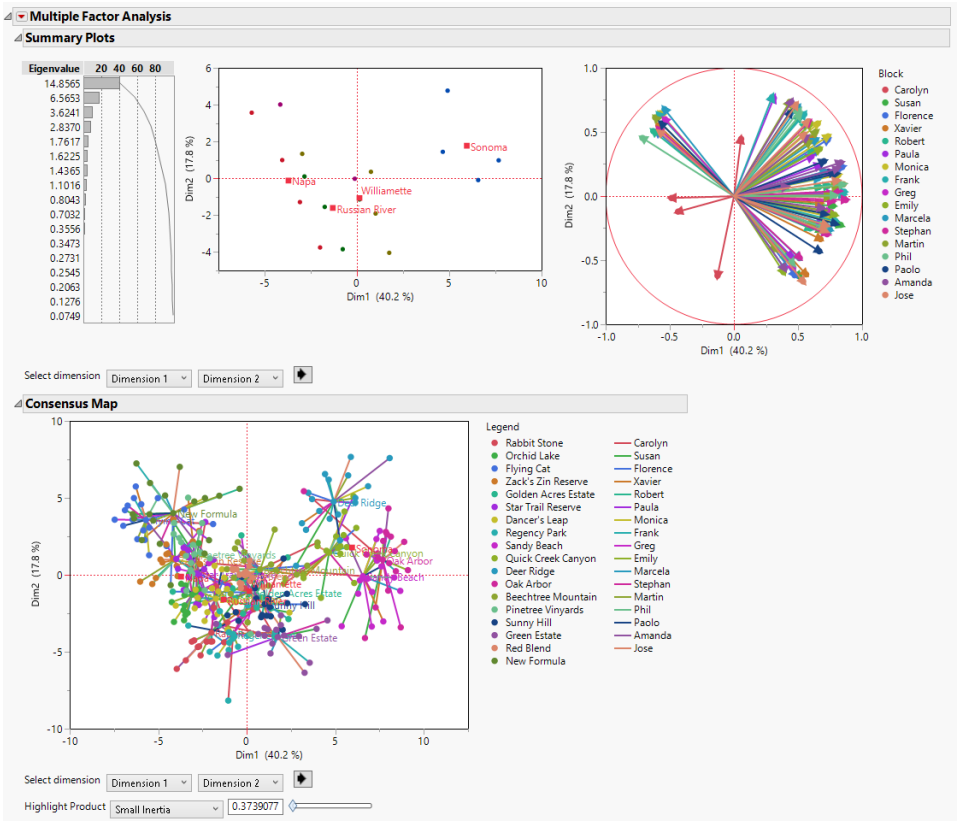
This example uses data from a simulated sensory panel study of wine characteristics. Participants rated 16 wines on a number of characteristics from 1 (no intensity) to 10 (prominent intensity). You want to better understand how the 16 wines are similar or different.

1. Select **Help > Sample Data Library** and open Wine Sensory Data.jmp.
2. Select **Analyze > Consumer Research > Multiple Factor Analysis**.
3. Select Vineyard and click **Product ID**.
4. Select Region and click **Z, Supplementary**.
5. Select all of the column groups from Carolyn to Jose and click **Add Block**.

   Note: The columns in this data table are grouped into one block for each panelist. For ungrouped data, select the columns for a block, click **Add Block**, and repeat for each block.

6. Click **Run Model**.

Figure 7.2 Initial Multiple Factor Analysis Report



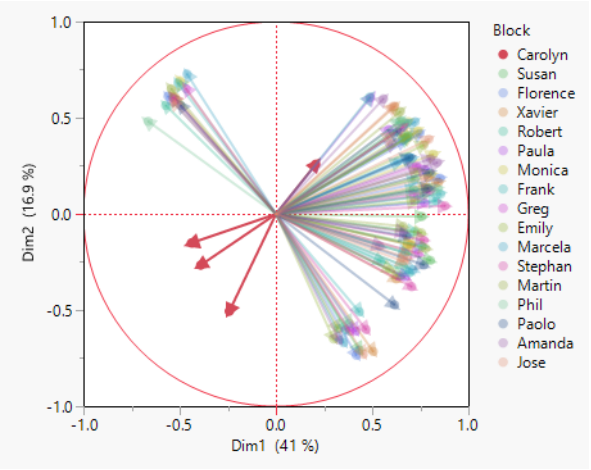
Tip: To set the legend in the Consensus Map to two columns, double-click the legend and set the Item Wrap in the Legend Settings to 18.

Notice the following in the Summary Plots:

- In the plot of the factor scores in the first two dimensions, the wines tend to cluster together according to their regions.
- In the loading plot, the rays in the lower left quadrant correspond to Carolyn. They indicate a difference between Carolyn and the other raters.

7. In the legend next to the loading plot, click **Carolyn** to highlight her results.

Figure 7.3 Loading Plot with Results for Carolyn Highlighted

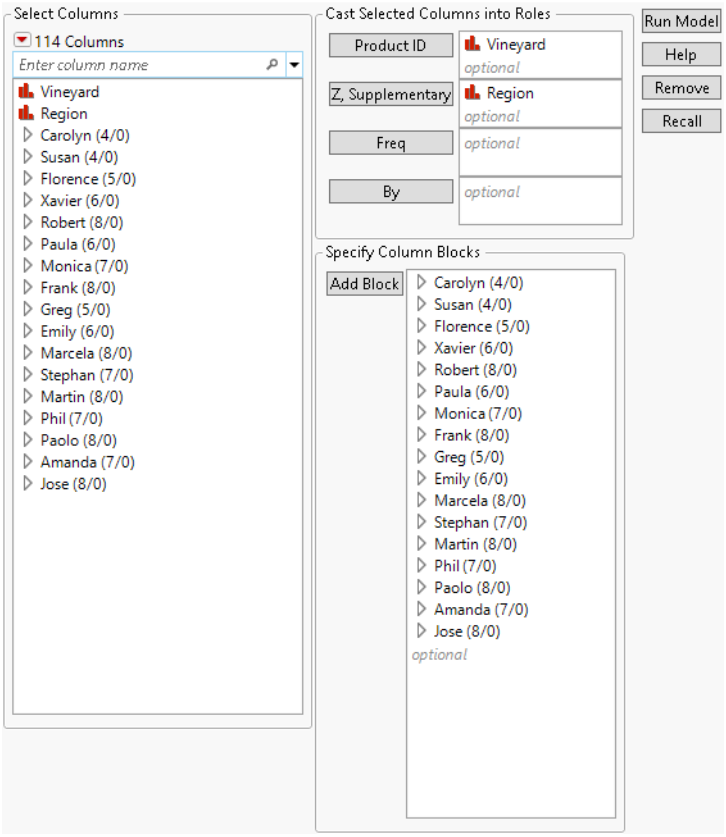


Carolyn's results differ from the other panelists. You might want to re-run the analysis without her results. See Figure 7.6 for the results of the analysis without Carolyn.

Launch the Multiple Factor Analysis Platform

Launch the Multiple Factor Analysis Platform by selecting **Analyze > Consumer Research > Multiple Factor Analysis**.

Figure 7.4 The Multiple Factor Analysis Launch Window



For more information about the options in the Select Columns red triangle menu, see *Using JMP*.

Product ID Specifies columns of items or products to be analyzed.

Z, Supplementary Specifies the columns to be used as supplementary variables. These variables are those with which you are interested in identifying associations, but they are not included in the calculations.

Freq Identifies one column whose numeric values assign a frequency to each row in the analysis.

By A column or columns whose levels define separate analyses. For each level of the specified column, the corresponding rows are analyzed using the other variables that you have specified. The results are presented in separate reports. If more than one By variable is assigned, a separate report is produced for each possible combination of the levels of the By variables.

Add Block Use to add one or more columns to a block:

- Adds individual columns as a single block.
- Adds a column group as a block.
- Adds individual columns to a selected block.

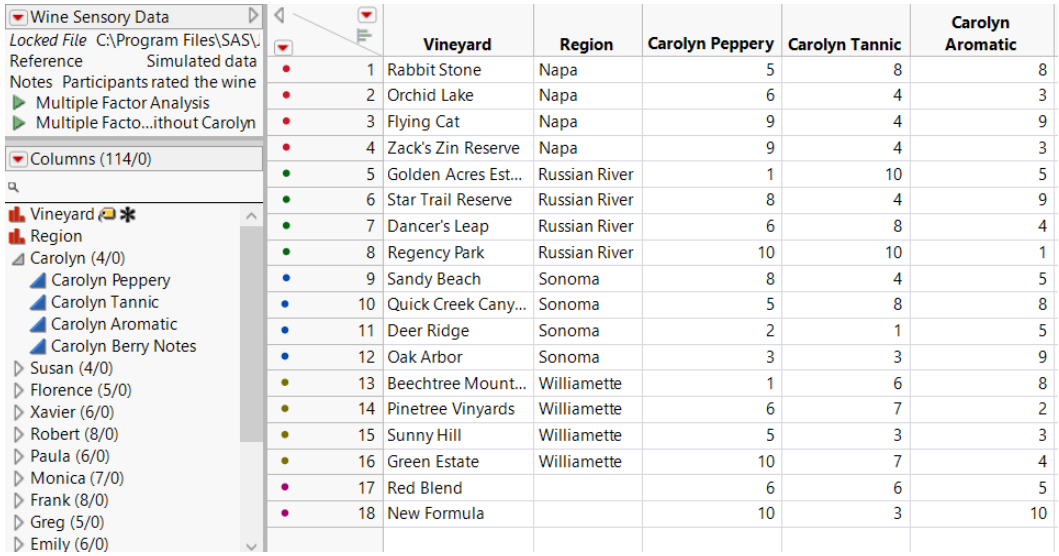
Tip: If you group the columns into blocks before running the platform, you can select multiple column groups and cast them as blocks in a single action. Otherwise, you must select each group of columns for each block and click Add Block, one block at a time. Double-click a block name to change it.

Data Format

The Multiple Factor Analysis platform uses a data table that contains column groups, or sub-tables. Each column group, referred to as a block, can have a different number of columns. Each block of columns can represent different measurements. The columns or blocks do not have to be in JMP column groups. However, the platform is easier to launch when the columns are grouped into blocks in the data table.

The data table rows represent the items that are being measured. Observations for each item must be in a single row. For example, Figure 7.5 shows a table that is measuring attributes of 16 wines from different vineyards. The column panel shows the column groups, or sub-tables, for each panelist. The Vineyards in rows 17 and 18 are not assigned a Region. The analysis could be used to explore which region the vineyards are most aligned to.

Figure 7.5 Partial View of a Data Table for Multiple Factor Analysis



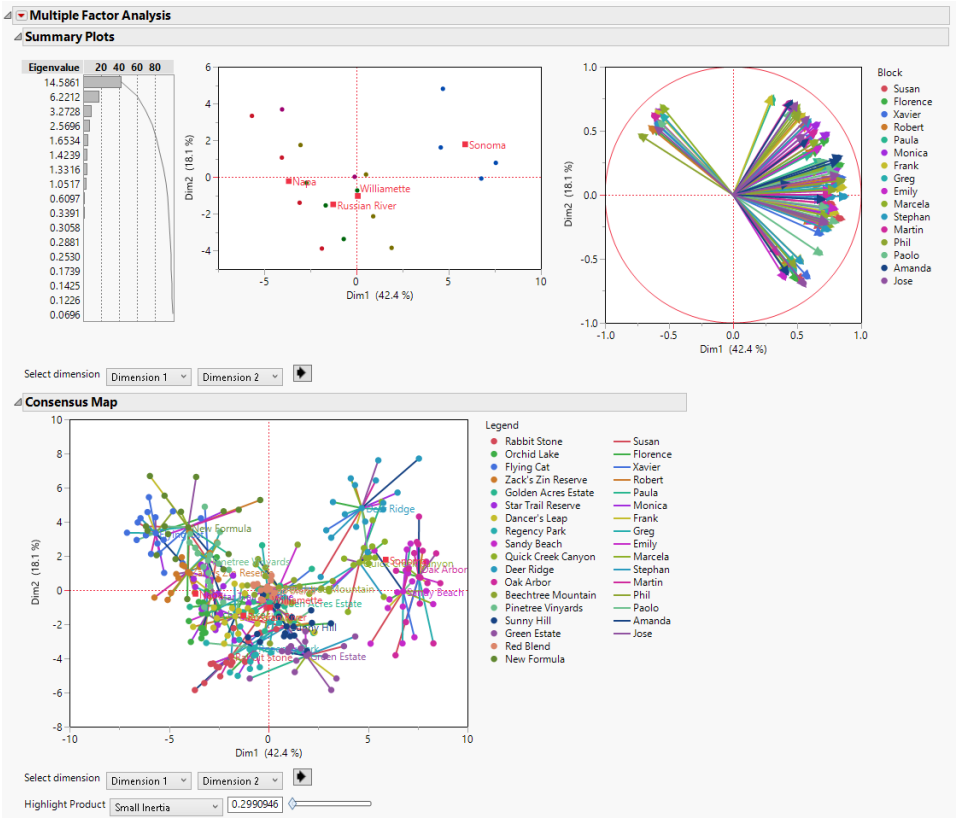
	Vineyard	Region	Carolyn Peppery	Carolyn Tannic	Carolyn Aromatic
1	Rabbit Stone	Napa	5	8	8
2	Orchid Lake	Napa	6	4	3
3	Flying Cat	Napa	9	4	9
4	Zack's Zin Reserve	Napa	9	4	3
5	Golden Acres Est...	Russian River	1	10	5
6	Star Trail Reserve	Russian River	8	4	9
7	Dancer's Leap	Russian River	6	8	4
8	Regency Park	Russian River	10	10	1
9	Sandy Beach	Sonoma	8	4	5
10	Quick Creek Cany...	Sonoma	5	8	8
11	Deer Ridge	Sonoma	2	1	5
12	Oak Arbor	Sonoma	3	3	9
13	Beechtree Mount...	Williamette	1	6	8
14	Pinetree Vinyards	Williamette	6	7	2
15	Sunny Hill	Williamette	5	3	3
16	Green Estate	Williamette	10	7	4
17	Red Blend		6	6	5
18	New Formula		10	3	10

Note: Missing observations are replaced by the column mean. When missing observations result in no variation for a column, the column is excluded from the analysis. Missing rules are applied to all variables, including supplementary variables.

The Multiple Factor Analysis Report

The initial Multiple Factor Analysis report shows a table of eigenvalues, summary plots, and a consensus map.

Figure 7.6 Multiple Factor Analysis Report



Summary Plots

The Summary Plot report has three sections:

- The first section shows the eigenvalues of the consensus PCA with a plot of the cumulative percent of variance explained by each component. A consensus PCA is used to obtain a common representation of the blocks of data. Consensus PCA refers to the principal component solution of the weighted sub-tables and is used to obtain a common representation of the blocks of data.

- The middle section is a plot of factor scores with a marker for each row (item). If supplementary variables are used, there is a labeled marker for each level of the variables. Items that cluster together in this plot are considered to be similar.
- The third section is a loading plot of factor loadings for each block.

Tip: In the loading plot legend, click a block to highlight it in the plot.

Select dimension Controls the dimensions plotted on the score and loading plots. The first control selects the horizontal dimension, and the second control selects the vertical dimension.

Consensus Map

The Consensus Map report contains a plot that overlays the individual panelist responses with the average response among all panelists for each item. You might use this map to investigate response consistency among panelists. For example, if a given panelist's points fall consistently farther from the average of the other panelists, then that panelist might be a candidate to exclude from the analysis. Hover over a data point to view the block label for that point. Click a product ID in the legend to highlight it in the consensus map.

Select dimension Controls the dimensions plotted on the consensus map. The first control selects the horizontal dimension, and the second control selects the vertical dimension.

Highlight Product Controls the transparency of the items according to their inertia score. Small inertia indicates items that panelists have good agreement on. Large inertia indicates items that panelists do not agree on.

Small Inertia Highlights items with inertia less than or equal to the value in the text box. To adjust the cutoff for the inertia value, use the slider or enter a value in the text box.

Large Inertia Highlights items with inertia greater than or equal to the value in the text box. To adjust the cutoff for the inertia value, use the slider or enter a value in the text box.

Min and Max Inertia Highlights the items with the smallest and largest inertia. The text box and slider have no impact on the results when this option is selected.

Multiple Factor Analysis Platform Options

The Multiple Factor Analysis red triangle menu includes the following options.

Block Weights Shows or hides the first eigenvalue for each block as well as the weight of that eigenvalue. The weight is the inverse of the square root of the first eigenvalue.

Eigenvalues Shows or hides a table of eigenvalues that correspond to the consensus dimensions, in order, from largest to smallest.

Eigenvectors Shows or hides a table of the eigenvectors for each of the consensus dimensions, in order, from left to right. Using these coefficients to form a linear combination of the original variables produces the consensus principal component variables.

Variable Loadings Shows or hides the loadings for each column. As in principal components analysis, loadings represent correlations of variables with components. Values near zero indicate the variable has little effect on the consensus dimension.

Variable Partial Contributions Shows or hides a table that contains the partial contributions of variables. The partial contributions represent the percentage of variance that each variable contributes to the consensus dimension.

Variable Squared Cosines Shows or hides a table that contains the squared cosines of variables. The sum of the squared cosine values across consensus dimensions is equal to 1 (100%) for each variable. The squared cosines represent the overlap in variance between variables and dimensions.

Tip: For the variable loadings, variable partial contributions, and variable squared cosines, values near zero indicate the variable is weakly related to the consensus dimension. Values far from zero indicate a strong association. The degree of transparency for the table values highlights these effects.

Summary Plots Shows or hides the summary plots. The summary plots include the plot of the eigenvalues or score plot and the loading plot.

Consensus Map Shows or hides the consensus map. See "Consensus Map".

Biplot Shows or hides a plot that is an overlay of the score and loadings plots. Use the controls to select any two dimensions for the plot.

Partial Axes Plot Shows or hides a partial axes plot. This plot displays correlations between PCA scores from separate block analyses and the consensus principal component. Use the controls to select any two dimensions for the plot. Click a block in the legend to highlight that block in the plot.

Display Options

Arrow Lines Enables you to show or hide arrows on the loading plot, and the partial axes plot. Arrows are shown if the number of variables is 1000 or fewer. If there are more than 1000 variables, the arrows are off by default.

Show Labels Shows or hides block name labels on all points in the consensus map and bi-plot. Shows or hides column name labels on all points in the partial axes plot.

Tip: Use row labels to identify centroids on the consensus map and data points on the loading plot.

RV Correlations Shows or hides a matrix of squared correlation coefficients between blocks.

Lg Coefficients Shows or hides a matrix of similarity measures between blocks.

Block Partial and Consensus Correlations Shows or hides a matrix of correlation coefficients between block partial scores and consensus principal component scores. The matrix is rectangular because only correlations between concordant dimensions are displayed.

Block Partial Contributions Shows or hides the sum of the variable contributions within the block.

Block Partial Inertias Shows or hides the block contribution multiplied by the eigenvalue for the principal component and then divided by 100.

Block Squared Cosines Shows or hides the block inertia squared and divided by the sum of squares used to calculate the eigenvalues. The values have a range between 0 and 1. The Block Squared Cosines can be considered as the percentage of the block variance explained by each principal component.

Save Individual Scores Saves the item consensus principal components to new columns in the data table. If one or more categorical supplementary variables are used, this option also saves individual scores for each level of the supplementary variables to a new data table.

Save Individual Squared Cosines Saves the item squared cosines to new columns in the data table. If one or more categorical supplementary variables are used, this option also saves categorical supplementary variable squared cosines to a new data table.

Save Individual Partial Contributions Saves the item partial contributions to new columns in the data table. If one or more categorical supplementary variables are used, this option also saves categorical supplementary partial contributions to a new data table.

Save Block Partial Scores Saves the block partial scores to a new data table.

Save Partial Axes Coordinates Saves the partial axes coordinates to a new data table.

See *Using JMP* for more information about the following options:

Local Data Filter Shows or hides the local data filter that enables you to filter the data used in a specific report.

Redo Contains options that enable you to repeat or relaunch the analysis. In platforms that support the feature, the Automatic Recalc option immediately reflects the changes that you make to the data table in the corresponding report window.

Save Script Contains options that enable you to save a script that reproduces the report to several destinations.

Statistical Details for the Multiple Factor Analysis Platform

Multiple factor analysis combines information from sub-tables into a set of orthogonal columns that describe the items in the rows of the table. The basic procedure is as follows:

- Perform PCA on each sub-table.
- Record the first eigenvalue of each sub-table to create a matrix of weights.
- Concatenate the sub-tables side-by-side, center and normalize the matrix.
- Perform a generalized PCA on the concatenated table via the singular value decomposition. Generalized PCA is used to constrain the solution using the sub-table weights.

This results in three matrices of generalized right and left singular vectors and singular values. These are then used to derive component scores, eigenvalues, and component loadings for the consensus across sub-tables. These three matrices are the result of decomposing the many columns from the original measurements into a few interpretable dimensions that explain the similarities and differences between the objects being measured.

Calculations

For MFA, a singular value decomposition of the **X** matrix can be defined as follows:

$$X = P\Delta Q^T \quad \text{with the constraint} \quad P^T MP = Q^T AQ = I$$

The matrices use are as follows:

X is an n x p centered and normalized matrix of sub-tables. In consumer research there are n products and p panelists' ratings.

Q is a p x q matrix of right singular vectors, which are weighted by the MFA singular values to obtain the loadings on q principal components.

Δ is a q x q diagonal matrix of singular values from the generalized PCA. As with PCA, the magnitude of the squared singular values, or eigenvalues, represent the importance of each principal component in the combined analysis.

P is an n x q matrix of left singular vectors, which are weighted by the MFA singular values to obtain the q principal components of the compromise.

M is the n x n diagonal matrix of mass weights.

A is the p x p diagonal matrix of block or panelist weights.

For more information about multiple factor analysis, see Abdi et al. (2013).

Mass Weight

JMP calculations use $N - 1$ for mass weight calculations. These calculations affect individual and block partial scores.

Appendix A

References

Abdi, H., Williams, L. J., and Valentin, D. (2013). "Multiple factor analysis: principal component analysis for multitable and multiblock data sets." *WIREs Comp Stat*, 5:149–179. doi: 10.1002/wics.1246.

Benjamini, Y., and Hochberg, Y. (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society, Series B* 57:289–300.

Breslow, N. E., and Day, N. E. (1980). *The Analysis of Case-Control Studies*. Statistical Methods in Cancer Research, IARC Scientific Publications, vol. 1, no. 32. Lyon: International Agency for Research on Cancer.

Firth, D. (1993). "Bias Reduction of Maximum Likelihood Estimates." Biometrika 80:27–38.

Heinze, G., and Schemper, M. (2002). "A Solution to the Problem of Separation in Logistic Regression." *Statistics in Medicine* 21:2409–2419.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Lavassani, K. M., Movahedi, B., and Kumar, V. (2009). "Developments in Analysis of Multiple Response Survey Data in Categorical Data Analysis: The Case of Enterprise System Implementation in Large North American Firms." *Journal of Applied Quantitative Methods* 4:45–53.

Louviere, J. J., Flynn, T. N., and Marley, A. A. (2015). *Best-Worst Scaling: Theory, Methods and Applications*. Cambridge: Cambridge University Press.

McFadden, D. (1974). "Conditional Logit Analysis of Qualitative Choice Behavior." in *Frontiers in Econometrics* edited by P. Zarembka, 105–142. New York: Academic Press. Accessed April 25, 2016. https://eml.berkeley.edu/reprints/mcfadden/zarembka.pdf.

Radcliffe, N. J., and Surry, P. D. (2011). *Real-World Uplift Modeling with Significance-Based Uplift Trees*. Portrait Technical Report TR-2011-1, Stochastic Solutions. Accessed January 13, 2017. http://www.stochasticsolutions.com/pdf/sig-based-up-trees.pdf.

Rossi, P. E., Allenby, G. M., and McCulloch, R. (2005). *Bayesian Statistics and Marketing*. Chichester, UK: John Wiley & Sons.

Sall, J. (2002). "Monte Carlo Calibration of Distributions of Partition Statistics." SAS Institute Inc.,Cary, NC. Accessed July 29, 2015. https://www.jmp.com/content/dam/jmp/documents/en/white-papers/montecarlocal.pdf.

SAS Institute Inc. (2020). "The TTEST Procedure." In *SAS/STAT 15.2 User's Guide.* Cary NC: SAS Institute Inc. https://go.documentation.sas.com/api/docsets/statug/15.2/content/ttest.pdf.

Train, K. E. (2001). "A Comparison of Hierarchical Bayes and Maximum Simulated Likelihood for Mixed Logit." Department of Economics, University of California, Berkley. Accessed January 13, 2017. https://eml.berkeley.edu/~train/compare.pdf.

Train, K. E. (2009). *Discrete Choice Methods and Simulation*. 2nd ed. Cambridge University Press.

Westfall, P. H., Tobias, R. D., and Wolfinger, R. D. (2011). *Multiple Comparisons and Multiple Tests Using SAS*. 2nd ed. Cary, NC: SAS Institute Inc.

Appendix B

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
- You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

Scripting Guide

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 Scripting Guide*. Cary, NC: SAS Institute Inc.

JMP® 16 Scripting Guide

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

Scripting Guide

1 Learn about JMP 23

Documentation and Additional Resources

Formatting Conventions in JMP Documentation 25
JMP Help 26
JMP Documentation Library 26
Additional Resources for Learning JMP 32
JMP Tutorials 32
Sample Data Tables 32
Learn about Statistical and JSL Terms 33
Learn JMP Tips and Tricks 33
JMP Tooltips 33
JMP User Community 34
Free Online Statistical Thinking Course 34
JMP New User Welcome Kit 34
Statistics Knowledge Portal 34
JMP Training 34
JMP Books by Users 35
The JMP Starter Window 35
JMP Technical Support 35

2 Introduction 37

Welcome to the JMP Scripting Language

What JSL Can Do for You 39
Help with Learning JSL 39
The Scripting Guide 39
The Scripting Index 40
Let JMP Teach You JSL 42
Terminology 43
Basic JSL Syntax 46
Book Conventions 47

3 Getting Started 49
Let JMP Write Your Scripts

Capturing a Script for an Analysis Report 51
Capturing a Script for a Data Table 52
Capturing a Script to Import a File 53
Gluing Scripts Together 54

4 Scripting Tools 59
Using the Script Editor, Log Window, Debugger and Profiler

Work with the Script Editor 61
Introduction to the Script Editor 61
Run a Script 62
Stop a Script 63
Edit a Script 63
Color Coding and Themes 63
Autocomplete Functions 64
Tooltips 64
Split a Window 65
Match Parentheses, Brackets, and Braces 66
Select a Rectangular Block of Text 67
Select Fragmented Text 68
Drag and Drop Text 68
Find and Replace 68
Automatic Formatting 69
Errors Are Marked by Special Characters 69
Typing in Multiple Rows at Once 69
Add Code Folding Markers 70
Advanced Options 71
Set Preferences for the Script Editor 72
Work with the Log 72
Enhanced Log in JMP 73
Text Log in JMP 78
Debug or Profile Scripts 79
Debugger and Profiler Window 80
Work with Breakpoints 84
View Variables 87
Work with Watches 87
Modify Preferences in Debugger 88
Persistent Debugger Sessions 89
Examples of Debugging and Profiling Scripts 89

5 JSL Building Blocks 97

Learning the Basics of JSL

JSL Syntax Rules 100
- Value Separators 100
- Numbers 103
- Names 103
- Comments 104

Operators 105
Boolean Values 109
Global and Local Variables 110
- Local Namespaces 110
- Named Namespaces 111
- Show Symbols, Clear Symbols, and Delete Symbols 111
- Lock and Unlock Symbols 112
- Hide a Global Variable 113

Rules for Name Resolution 113
- Variable Names Used as Arguments 114
- Rules for Resolving Names 114
- Scoping Operators 116
- Troubleshoot Variables and Column Names 119
- Troubleshoot Variables and Keywords 120
- Frequently Asked Questions about Name Resolution 120

Alternatives for Gluing Expressions Together 121
Iterate 122
- For 122
- While 123
- Summation 125
- Product 125
- Break and Continue 126

Conditional Functions 128
- If 128
- Match 130
- Choose 132
- Interpolate 133
- Step 135

Compare Incomplete or Mismatched Data 135
- Missing Values 137

Inquiry Functions 138
- General Element Types 139
- Specific Element Types 139

Object Attributes ... 141
Host Information ... 141
Version Information ... 141

6 Types of Data ... 143
Working with Numbers, Strings, Dates, Currency, and More

Numbers and Strings ... 145
Formatting Numbers ... 146
Unicode Characters ... 147
Path Variables ... 147
Create and Customize Path Variables ... 150
Relative Paths ... 151
File Path Separators ... 151
Date-Time Functions and Formats ... 152
Date-Time Values ... 152
Program with Date-Time Functions ... 153
Date-Time Values in Data Tables ... 160
Custom Date-Time Formatting with Format Patterns ... 164
Currency ... 166
Hexadecimal and BLOB Functions ... 167
Work with Character Functions ... 170
Concat ... 170
Munger ... 171
Repeat ... 172
Substitute and Substitute Into ... 173
Regular Expressions ... 174
Regex ... 174
Regex Match ... 176
Special Characters in Regular Expressions ... 178
Escaped Characters in Regular Expressions ... 179
Greedy and Reluctant Regular Expressions ... 181
Backreferences and Capturing Groups ... 183
Lookaround Assertions ... 184
Pattern Matching ... 185
Parsing Strings in Fixed Fields ... 186
Patterns and Case ... 188
Troubleshooting Patterns ... 188

7 Data Structures ... 191
Working with Collections of Data

Lists ... 193

Evaluate Lists ... 193
Assignments with Lists ... 195
Perform Operations in Lists ... 195
Find the Number of Items in a List ... 195
Subscripts ... 195
Locate Items in a List ... 197
List Operators and Functions ... 198
Iterate through a List ... 201
Concatenate Lists ... 202
Insert a Nested List in an Existing List ... 203
Index into a List with Another List ... 203
Send Messages to Lists ... 203
Matrices ... 204
Construct Matrices ... 204
Subscripts ... 206
Inquiry Functions ... 210
Comparisons, Range Checks, and Logical Operators ... 210
Numeric Operations ... 211
Concatenation ... 214
Transpose ... 214
Indexing into a Matrix or List with Another Matrix or List ... 215
Matrices and Data Tables ... 216
Matrices and Reports ... 218
Loc Functions ... 219
Ranking and Sorting ... 221
Special Matrices ... 221
Inverse Matrices and Linear Systems ... 226
Decompositions and Normalizations ... 230
Build Your Own Matrix Functions ... 235
Statistical Examples ... 236
Associative Arrays ... 240
Create Associative Arrays ... 240
Work with Associative Arrays ... 243
Applications for Associative Arrays ... 246
Associative Arrays in Graph Theory ... 248
Associative Arrays in Set Operations ... 251

8 Programming Methods ... 253
Complex Scripting Techniques and Additional Functions

Lists and Expressions ... 255

Expressions 255
Macros 267
Manipulating Lists 267
Manipulating Expressions 270
Advanced Scoping and Namespaces 274
Names Default To Here 274
Scoped Names 278
Namespaces 282
Referencing Namespaces and Scopes 286
Resolving Named Variable References 289
Best Practices for Advanced Scripting 290
Advanced Programming Concepts 291
Throw and Catch Exceptions 291
Functions 293
Recursion 296
Includes 296
Load and Save Text Files 297
Run External Programs 297
Classes 299
File and Directory Operations 302
Select a Directory or File 302
Get a List of Filenames 303
Connect to a Network Directory 304
Scripting By Groups 305
Encrypt and Decrypt Scripts 306
Encryption and Global Variables 308
Encrypting Scripts in Data Tables 308
Working with Expressions in an Encrypted Script 309
Additional Numeric Functions 310
Derivatives 310
Algebraic Manipulations 312
Maximize and Minimize 314
Identifying Differences Between Strings, Lines, or Sequences 315
Scheduling Actions 317
Functions that Communicate with Users 318
Writing to the Log 319
Send Information to the User 320

9 Data Tables ... 323

Working with Data Table Objects

Get Started with Data Tables ... 327
Basic Data Table Scripting ... 330
- Open a Data Table ... 330
- Create a New Data Table ... 332
- Import Data ... 333
- Set the Current Data Table ... 357
- Name a Data Table ... 358
- Save a Data Table ... 358
- Hide a Data Table ... 359
- Print a Data Table ... 360
- Resize a Data Table ... 360
- Close a Data Table ... 360
- Set and Get a Data Table ... 361
- Perform Actions on All Open Data Tables ... 361
- Create Journals ... 362
- Create Excel Workbooks ... 363
- Prevent Changes to a Data Table ... 364

Advanced Data Table Scripting ... 365
- Select Columns in a Data Table ... 365
- Store Summary Statistics in Global Variables ... 366
- Create a Table of Summary Statistics ... 369
- Subset a Data Table ... 371
- Add a Data Filter ... 372
- Sort a Data Table ... 379
- Stack Values in a Data Table ... 379
- Split Values in a Stacked Data Table ... 380
- Transpose a Data Table ... 382
- Vertically Concatenate Data Tables ... 383
- Horizontally Concatenate Data Tables ... 383
- Virtually Join Data Tables ... 384
- Replace Data in Data Tables ... 387
- Create a Table Using Tabulate ... 389
- Copy Data Tables ... 390
- Find Missing Data Patterns ... 390
- Compare Data Tables ... 391
- Create a Summary Table ... 391
- Subscribe to a Data Table ... 392
- Move Data Between Matrices and Data Tables ... 394

Add Hyperlinks to Text Columns . . . 394
Columns . . . 396
Send Messages to Data Column Objects . . . 397
Create Columns . . . 398
Create a New Column by Text Matching . . . 400
Add Several Columns at Once . . . 401
Group Columns . . . 402
Use Set Selected to Select Columns . . . 404
Rearrange and Move Columns . . . 405
Recode Columns . . . 406
Add a Column Switcher . . . 407
Compress Selected Columns . . . 408
Create Custom Functions, Transforms, and Formats . . . 408
Custom Transform Example . . . 411
Delete Columns . . . 415
Get Column Names . . . 415
Column Attributes . . . 416
Column Properties . . . 423
Specify Engineering SI Units . . . 430
Rows . . . 431
Add Rows . . . 431
Delete Rows . . . 433
Select Rows . . . 433
Find Rows . . . 437
Move Rows . . . 437
Assign Colors and Markers to Rows . . . 438
Color Cells . . . 438
Hide, Exclude, and Label Rows . . . 439
Iterate on Rows in a Table . . . 440
Prevent Editing Rows . . . 443
Row States . . . 444
Accessing Data Values . . . 460
Set or Get Values by Column Name . . . 461
Additional Ways to Access Data Values . . . 462
Add Metadata to a Data Table . . . 463
Data Table Variables . . . 463
Table Scripts . . . 463
Formulas . . . 468
Delete Metadata . . . 469
Calculations . . . 469

Pre-Evaluated Statistics . . . 469
Calculator Formulas . . . 471

10 Scripting Platforms . . . 473
Create, Repeat, and Modify Analyses

Example of Scripting a Platform . . . 477
Send Messages to a Platform . . . 480
Conventions for Messages and Arguments . . . 480
Send Multiple Messages . . . 481
Find Messages for Objects . . . 482
Interpret the Show Properties List . . . 483
Send Messages to an Object . . . 484
Specify Which Columns to Analyze . . . 485
Create Column References . . . 485
Specify Multiple Column Names at Once . . . 486
Enable Users to Specify Columns . . . 486
Specify a By Variable . . . 486
Filter by Value using a Where Statement . . . 490
Enable User Input . . . 490
Execute Embedded Red Triangle Options . . . 491
Make Platforms Invisible . . . 491
Specify Report Titles . . . 493
Common Messages for Platform Windows . . . 493
Scripting Considerations by Platform . . . 494
Categorical . . . 494
Control Charts . . . 495
Distribution . . . 498
DOE . . . 499
Fit Model . . . 501
Formula Depot . . . 505
Graph Builder . . . 506
Neural and Neural Net . . . 508
Partial Least Squares and PLS . . . 508
Process Capability . . . 508
Scatterplot 3D . . . 509
Text Explorer . . . 510
Explore Outliers . . . 510
Scripting-Only Messages and Arguments by Platform . . . 511
Bivariate Script-Only Messages . . . 513
Boosted Tree Script-Only Messages . . . 514

Bootstrap Forest Script-Only Messages . . . 515
Bubble Plot Script-Only Messages . . . 516
Choice Script-Only Messages . . . 516
Clustering Script-Only Messages . . . 517
Contingency Script-Only Messages . . . 517
Contour Plot Script-Only Messages . . . 517
Contour Profiler Script-Only Messages . . . 517
Control Chart Script-Only Messages . . . 518
Control Chart Builder Script-Only Messages . . . 518
Cumulative Damage Script-Only Messages . . . 518
Custom Profiler Script-Only Messages . . . 519
Degradation Script-Only Messages . . . 519
Destructive Degradation Script-Only Messages . . . 519
Discriminant Script-Only Messages . . . 519
Distribution Script-Only Messages . . . 520
EMP Measurement Systems Analysis Script-Only Messages . . . 520
Explore Missing Values Script-Only Messages . . . 521
Explore Outliers Script-Only Messages . . . 521
Factor Analysis Script-Only Messages . . . 521
Fit Life by X Script-Only Messages . . . 521
Fit Model Script-Only Messages . . . 522
Fit Parametric Survival Script-Only Messages . . . 524
Functional Data Explorer Script-Only Messages . . . 525
Gaussian Process Script-Only Messages . . . 525
K Means Cluster Script-Only Messages . . . 525
K Nearest Neighbors Script-Only Messages . . . 525
Latent Class Analysis Script-Only Messages . . . 525
Life Distribution Script-Only Messages . . . 526
Make Validation Column Script-Only Messages . . . 527
MaxDiff Script-Only Messages . . . 527
Mixture Profile Script-Only Messages . . . 527
Model Screening Script-Only Messages . . . 528
Multivariate Script-Only Messages . . . 528
Naive Bayes Script-Only Messages . . . 528
Neural Script-Only Messages . . . 528
Nonlinear Script-Only Messages . . . 529
Normal Mixtures Script-Only Messages . . . 530
Oneway Script-Only Messages . . . 530
Parallel Plot Script-Only Messages . . . 530
Pareto Plot Script-Only Messages . . . 531

Partial Least Squares Script-Only Messages 531
Partition Script-Only Messages 531
Principal Components Script-Only Messages 532
Process Capability Script-Only Messages 532
Process Screening Script-Only Messages 532
Profiler Script-Only Messages 533
Reliability Forecast Script-Only Messages 533
Reliability Growth Script-Only Messages 533
Response Screening Script-Only Messages 534
Scatterplot 3D Script-Only Messages 534
Structural Equation Models Script-Only Messages 535
Support Vector Machines Script-Only Messages 536
Surface Plot Script-Only Messages 537
Tabulate Script-Only Messages 540
Ternary Plot Script-Only Messages 540
Text Explorer Script-Only Messages 540
Time Series Script-Only Messages 541
Uplift Script-Only Messages 541
Variability Chart Script-Only Messages 542

11 Display Trees 543
Create and Interact with Windows

Navigate JMP Reports 545
Examples of Common Display Boxes 546
View the Display Tree 547
Show Display Box Properties 549
Display Box Object References 550
Send Messages to Display Boxes 555
Example of Creating a Report 562
Construct Custom Windows 565
Example of Creating a Graph Box 565
Extract Values from a Window 566
Constructors for New Windows 568
Interact with a Closing Window 608
Update an Existing Display 610
Set Function and Set Script 614
Get and Set Selected Values of Display Elements That Return Lists 616
Send Messages to Constructed Displays 618
Construct Display Boxes That Contain Platforms 619
Counting the Number of Child Boxes 621

Examples of Creating a Dashboard from Two Reports . . . 622
Example of Creating a Cluster Platform Launch Window . . . 628
Example of Constructing a Custom Platform . . . 631
Example of Specifying Text and Background Colors . . . 634
Modal Windows . . . 635
Construct a Modal Window . . . 635
Construct a Column Dialog . . . 637
Comparison of Column Dialog and New Window . . . 639
Constructors for Column Dialogs . . . 640
Convert Deprecated Dialog to New Window . . . 643
New Window Example . . . 643
Example of Deprecated Dialog . . . 644
Differences Between New Window and the Deprecated Dialog . . . 646
Optional Scripts in New Window . . . 653
Scripting the Script Editor . . . 654
Technical Details . . . 655
Write Tab Box and Tab Page Box Scripts . . . 655

12 Scripting Graphs . . . 657
Editing and Creating Two-Dimensional Graphs

Add Scripts to Graphs . . . 659
Specify the Order of Graphical Elements . . . 660
Create New Graphs from Scratch and Customize Them . . . 666
Customize Graphs . . . 667
Create Shapes for a Bubble Plot . . . 669
How to Find Shapes . . . 672
Add Lines, Arrows, Points, Shapes, and Text . . . 673
Draw Lines . . . 673
Draw Arrows . . . 675
Draw Markers . . . 677
Draw Pies and Arcs . . . 678
Draw Regular Shapes: Circles, Rectangles, and Ovals . . . 680
Draw Irregular Shapes: Polygons and Contours . . . 682
Add Text . . . 685
Specify Colors . . . 686
Specify Transparency . . . 690
Add Fill Patterns . . . 691
Specify Line Types . . . 694
Draw with Pixels . . . 695
Interactive Graphs . . . 696

Handle() ... 696
Mousetrap() ... 701
Drag Functions ... 703
Troubleshoot Interactive Graphs ... 706
Create a Color Picker ... 706
Clip Display Segs in Graphics ... 707
Create Background Maps ... 709
Graph Elements ... 710
Plot Functions ... 711
Get the Properties of a Graphics Frame ... 717
Add a Legend ... 717
Hover Labels ... 718
Add Rich Text to Hover Labels Using Textlets ... 719
Add Graphs or Images to Hover Labels Using Graphlets ... 722
Update Names and Values in Hover Labels Using Gridlets ... 732
Create Drill-Down Graphs ... 737
Work with the Hover Label Execution Context ... 740

13 Three-Dimensional Scenes ... 749
Scripting in Three Dimensions

About JSL 3-D Scenes ... 751
JSL 3-D Scene Boxes ... 751
Setting the Viewing Space ... 754
Setting Up a Perspective Scene ... 755
Setting up an Orthographic Scene ... 756
Changing the View ... 757
The Translate Command ... 757
The Rotate Command ... 757
The Look At Command ... 759
The ArcBall ... 760
Graphics Primitives ... 761
Primitives Example ... 764
Controlling the Appearance of Primitives ... 765
Other Uses of Begin and End ... 771
Drawing Spheres, Cylinders, and Disks ... 771
Construction ... 771
Lighting ... 772
Drawing Text ... 773
Using Text with Rotate and Translate ... 773
Using the Matrix Stack ... 774

Lighting and Normals ... 778
Creating Light Sources ... 778
Lighting Models ... 780
Normal Vectors ... 781
Shading Model ... 781
Material Properties ... 782
Alpha Blending ... 783
Fog ... 783
Example ... 783
Bézier Curves ... 784
One-Dimensional Evaluators ... 785
Two-Dimensional Evaluators ... 787
Using the Mouse ... 788
Pick Commands ... 789
Three-Dimensional Arguments ... 790
Improve the Display of Graphics ... 790

14 Extending JMP ... 793
External Data Sources, Analytical Tools, and Automation

Real-Time Data Capture on Windows ... 797
Create a Data Feed Object ... 797
Read in Real-Time Data ... 799
Manage a Data Feed with Messages ... 800
Examples of Data Feed ... 802
Dynamic Link Libraries (DLLs) ... 804
Use Sockets in JSL ... 806
Socket-Related Commands ... 807
Messages for Sockets ... 808
Database Access ... 809
Run Query Builder Queries ... 809
Open Database Function ... 810
Creating a Database Connection and Executing SQL ... 812
Write a SQL Query ... 813
Work with SAS ... 814
Make a SAS DATA Step ... 814
Create SAS DATA Step Code for Formula Columns ... 814
SAS Variable Names ... 815
Connect to a SAS Metadata Server ... 815
Get the Values of SAS Macro Variables ... 818
Preferences ... 819

Sample Scripts . . . 819
Work with MATLAB . . . 821
Install MATLAB . . . 821
Work with R . . . 822
Installing R . . . 822
JMP to R Interfaces . . . 824
R JSL Scriptable Object Interfaces . . . 825
Conversion Between JMP Data Types and R Data Types . . . 825
Troubleshooting . . . 828
Examples . . . 829
Work with Python . . . 831
Install Python . . . 831
Troubleshooting the JMP Python Integration . . . 832
JMP-to-Python Interfaces . . . 836
Equivalent Data Types for Python Send() . . . 836
Equivalent Data Types for Python Get() . . . 837
JMP Scoping Operators and Python . . . 838
Python Name() and Python Send() Examples . . . 839
Additional Python Integration Examples . . . 840
Work with Microsoft Excel . . . 842
Parse XML . . . 843
OLE Automation . . . 845
Share Reports on JMP Live . . . 845
JMP Live Prerequisites . . . 845
Get an API Key . . . 845
Publish a Single Report to JMP Live . . . 846
Publish Several Reports to a JMP Live Folder . . . 847
Delete Reports in a JMP Live Folder . . . 847
Replace the Contents of a JMP Live Report . . . 848
Search for a JMP Live Report or Folder ID . . . 848
Update Data in JMP Live Reports . . . 849
Work with Other Content in JMP Live Reports . . . 850
HTTP Requests in JMP Live . . . 850
Communicate with REST Web Service . . . 850
Examples of HTTP Requests . . . 850
OAuth 2.0 for Web APIs . . . 853
Security . . . 853
Using Postman for Redirect URLs . . . 853
Authorization Code Grant . . . 854
Implicit Grant . . . 855

Resource Owner Grant 855
Using a Token 856
OAuth 2.0 Resources 856
Communicate with SAS Cloud Analytic Services 856
About CAS Actions 857
Connect to the CAS Server 858
Create and Submit a CAS Action 859
Create and Submit a CAS DATA Step Action 859
Create a New CAS Server 859
Import Data from the CAS Server 859
View a CAS Dataset 860
Get a List of Available CAS Libraries 860
Export Data from a Data Table to a CAS Server 860
Convert a CAS Table to a Data Table 860
Delete a CAS Table 863
Remove a CAS Table 863
Verify the CAS Server Connection 863
Get Details about the Current Connection 863
Disconnect from the CAS Server 863
Terminate All CAS Sessions 863

15 Creating Projects 865
Organizing Project Files in JMP

Create a Simple Project 867
Create a Project with a Bookmarks Pane and Log 868
Running Scripts in a Project 871
Create a Bookmark Group 872
Close Tabs in Projects 873
Get a Project 873
Reset the Window Layout 874
Run Startup Scripts to Control Workspaces 875

16 Creating Applications 877
Application Builder

Create Applications with Application Builder 879
Example of Creating an Application 879
Application Builder Terminology 881
Design an Application 882
Application Builder Window 883
Application Builder Red Triangle Options 884
Create an Application 886

Edit or Run an Application ... 897
Options for Saving Applications ... 898
Additional Examples of Creating Applications ... 899
Compile Add-Ins with the JMP Add-In Builder ... 915
Compile Scripts into an Add-In Using Add-In Builder ... 915
Edit an Add-In ... 920
Share an Add-In ... 920
Installing Multiple Add-Ins ... 920
Register an Add-In Using JSL ... 921
Create an Add-In Manually ... 921
Manage JMP Add-Ins ... 923

17 Common Tasks ... 925
Getting Started with Sample Scripts

Run a Script at Start Up ... 927
Convert Character Dates to Numeric Dates ... 928
Format Date-Time Values and Subset Data ... 929
Create a Formula Column ... 930
Extract Values from an Analysis into a Report ... 932
Create an Interactive Program ... 936

A Compatibility Notes ... 939
Changes in Scripting in JMP 16

Compatibility Issues in JMP 16.1 ... 939
Compatibility Issues in JMP 16.0 ... 939

B Efficient Scripts ... 941
Tips for Writing Scripts That Run Faster

Using Matrices ... 943
Design of Experiments ... 944
Scripting Data Tables ... 944
Scripting Graphs and Display Boxes ... 945
Scripting Numbers, Strings, Arrays, and Lists ... 946
Programming Tips ... 946
Troubleshooting ... 946

C References ... 947

D Glossary ... 949
Terms, Concepts, and Placeholders

E Technology License Notices ... 953

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 25
JMP Help . . . 26
JMP Documentation Library . . . 26
Additional Resources for Learning JMP . . . 32
JMP Tutorials . . . 32
Sample Data Tables . . . 32
Learn about Statistical and JSL Terms . . . 33
Learn JMP Tips and Tricks . . . 33
JMP Tooltips . . . 33
JMP User Community . . . 34
Free Online Statistical Thinking Course . . . 34
JMP New User Welcome Kit . . . 34
Statistics Knowledge Portal . . . 34
JMP Training . . . 34
JMP Books by Users . . . 35
The JMP Starter Window . . . 35
JMP Technical Support . . . 35

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

Introduction

Welcome to the JMP Scripting Language

The JMP Scripting Language, or *JSL*, lets you write scripts to re-create results in JMP. Power users often develop scripts to extend JMP's functionality and automate a regularly scheduled analysis in production settings. If you do not want to learn JSL, JMP can write the scripts for you.

JSL is used to perform many actions:

- implements column formulas
- launches platforms
- interactively modifies platforms
- creates graphics

Contents

What JSL Can Do for You 39
Help with Learning JSL 39
The Scripting Guide 39
The Scripting Index 40
Let JMP Teach You JSL 42
Terminology 43
Basic JSL Syntax 46
Book Conventions 47

What JSL Can Do for You

JMP can automatically save scripts to reproduce any data table or analysis in its current state. You can pause anytime in your analysis to save a script to a script window (or *script editor*), in a data table, or in an analysis report. You can then modify the script as needed for future projects. When you are finished with your work, you can then save a script to reproduce your final results.

Here are some examples where JSL scripts can be helpful:

- Suppose you need to describe an analysis process in detail, from beginning to end. An example is to create an audit trail for a governing agency, or for peers reviewing your journal article.
- Suppose you have a set of analysis steps that should be followed routinely by your lab technicians.
- Suppose you fit the same model to new data every day, and the steps are always the same.

You can use JMP interactively as usual, save scripts to reproduce your work, and in the future run those scripts to reproduce your results.

There are a few things that JSL is not designed to do:

- JMP cannot record scripts while you are working. Though script-recording is a useful feature in some other scripting languages, it is less important for software like JMP, where the results are what matter. You cannot use script-recording to observe how a sequence of interactive steps is performed.
- JSL is not an alternative command-line interface for using the program.

Help with Learning JSL

There are several places within JMP to get help with writing or understanding a JSL script.

The Scripting Guide

The *Scripting Guide* begins with basic information (such as terminology and syntax) for JMP users who are not familiar with the scripting language. The *Scripting Guide* then progresses to more advanced information.

Chapters 2 through 4	Includes information about learning JSL, producing basic scripts, and introduces you to the JSL scripting environment.
Chapters 5 through 8	Introduces the building blocks of the language; working with basic data types, such as numbers and strings; writings lists, matrices, and associate arrays; namespaces; and the fundamentals of programming in JSL.
Chapters 9 through 13	Covers using JSL with objects in JMP, such as data tables, platforms, windows, and graphics.
Chapter 14	Describes how to write scripts that work with external programs, such as SAS, R, and Microsoft Excel.
Chapter 15	Describes how to organize files that you use in an analysis, perform analyses, and run scripts from one workspace.
Chapter 16	Introduces creating JMP applications in Application Builder, a drag-and-drop environment for visually designing windows with buttons, lists, graphs, and other objects. The chapter also describes how to use Add-In Builder to compile scripts into one easily shared file.
Chapter 17	Contains a collection of recipes, or script examples, that you can copy and modify for your own use.
Appendices A, B, C, and D	Provides information about compatibility issues with the previous version of JMP, provides tips for writing more efficient scripts, gives information about references, and defines JSL concepts and terminology.

The Scripting Index

The Scripting Index on the **Help** menu provides a brief description and the syntax for JSL functions, objects, and display boxes. Each entry includes an example that you can run and modify to test your own code. And an embedded log window lets you see messages as examples are run.

Notes:

- To display the Scripting Index entry for a function in a script, run the script, press Alt, and then double-click the function. You can also run the script, right-click the function, and select **Help Scripting Index.**
- To copy the syntax or description from the Scripting Index window, right-click the area that you want to copy and select **Copy Text**.

The Scripting Index window includes the following buttons:

Click the **Clear** button to clear the search text box to begin a new search.

Click the arrow to set search filter options and parameters.

After you edit the sample script in the Scripting Index, click this button to revert to the original script.

Note: When you edit an example script, the changes persist when you view other entries in the Scripting Index. To revert the script, click **Reset**, which appears above the script after you edit it.

Search Filter Options

Click the down arrow button next to the search box to refine your search.

Contains Terms Returns items that contain a part of the search criteria. A search for “ease oom” returns messages such as “Release Zoom”.

Contains Phrase Returns items that contain the exact search criteria. A search for “text box” returns entries that contain “text” followed directly by “box” (for example, “Context Box” and “Text Box”).

Starts With Phrase Returns items that start with the search criteria.

Ends With Phrase Returns items that end with the search criteria.

Whole Phrase Returns items that consist of the entire string. A search for “text box” returns entries that contain only “text box”.

Regular Expression Enables you to use the wildcard (*) and period (.) in the search box. Searching for “get.*name” looks for items that contain “get” followed by one or more words. It returns “Get Color Theme Names”, “Get Name Info”, and “Get Effect Names”, and so on.

Invert Result Returns items that do not match the search criteria.

Match All Terms Returns items that contain both strings. A search for “t test” returns elements that contain either or both of the search strings: “Pat Test”, “Shortest Edit Script” and “Paired t test”.

Ignore Case Ignores the case in the search criteria.

Match Whole Words Returns items that contain each word in the string based on the Match All Terms setting. If you search for “data filter”, and Match All Terms is selected, entries that contain both “data” and “filter” are returned.

Search Category and Item Names Returns category names (in the left column) and item names (in the middle column) that match the search criteria.

Search Prototypes and Descriptions Returns prototypes (the JSL syntax) and descriptions that match the search criteria.

Search Examples Returns examples that match the search criteria.

Let JMP Teach You JSL

The best JSL writer is JMP. You can work in JMP interactively and then save the results as a script to reuse later. With simple modifications, your script can serve as a template for speeding up routine tasks.

Because JSL is a very flexible language, you can reach your goals in many different ways. Here is an example. Typically, the script that JMP saves for you specifies every detail of your analysis, even if most of the details happen automatically by default. Does that mean that the scripts that you write have to be just as complete and detailed? Not at all. You usually need to specify only those details that you would select in the graphical user interface (GUI). For example, if you open Big Class.jmp from the sample data folder and want to launch Distribution for height, weight, and sex, the following script is all that is necessary:

```
Distribution( Y( :height, :weight, :sex ) );
```

Suppose you run the Distribution platform in the GUI and then select **Save Script > To Script Window** from the red triangle menu for the report. The following script appears:

```
Distribution(
   Continuous Distribution( Column( :height ) ),
   Continuous Distribution( Column( :weight ) ),
   Nominal Distribution( Column( :sex ) ),
);
```

Both scripts give the same result.

Feel free to experiment with JSL. If you think something ought to be possible, it probably is. Give it a try, and see what happens.

Terminology

Before you begin creating scripts, you should become familiar with basic JSL terms used throughout the *Scripting Guide*.

Operators and Functions

An *operator* is one- or two-character symbol (such as + or =) for features such as common arithmetic actions, scoping names, regular expressions, concatenating, and subscripting.

A *function* is a command that might contain additional information for the function to use.

Certain JSL functions work the same as operators but provide access to more complex actions. For example, the following two lines are equivalent:

```
2 + 3; // returns 5
Add( 2, 3 ); // returns 5
```

The first line uses the + operator. The second line uses the `Add()` function equivalent.

Although all JSL operators have function equivalents, not all functions have operator equivalents. For example, `Sqrt(a)` can be represented only by the `Sqrt()` function.

Note: In previous versions of JMP and its documentation, the terms *operators* and *functions* were used interchangeably. Now each term has a specific meaning.

Objects and Messages

An *object* is a dynamic entity in JMP, such as a data table, a data column, a platform results window, a graph, and so on. Most objects can receive messages that instruct the object to perform some action on itself.

A *message* is a JSL expression that is directed to an object. That object knows how to evaluate the message. In the following example, `dt` is the data table object. << indicates that a message follows. In the following example, the message tells JMP to create a summary table with the specified variables.

```
dt << Summary( Group( :age ), Mean( :height ) )
```

In this expression, `dt` is the name of a variable that contains a reference to a data table. You could use any name for this variable. The *Scripting Guide* commonly uses `dt` to represent data table references. Here are some of the more common names used to represent references to certain objects:

Abbreviation	Object
`dt`	data table
`col`	column in a data table
`colname`	the name of a column in a data table
`obj`	an object
`db`	display box

These variables are not pre-assigned references. Each one must be assigned prior to its use. In the following example, the global variable named `A` is assigned the value “Hello, World”. When the `Show( A )` command is processed, the result is the value of `A`.

```
A = "Hello, World";
Show( A );
   A = "Hello, World";
```

Arguments and Parameters

An *argument* is additional information that you can provide to a function or message. For example, in `Root(25)`, `25` is an argument to the `Root()` function. `Root()` acts on the argument that you provide and returns the result: 5.

Programming and scripting books commonly talk about parameters as well. A *parameter* is a description of the argument that a function accepts. For example, the general specification for `Root()` might be `Root( number )`, where *`number`* is the parameter.

Parameter and *argument* express two perspectives of the same concept: information that a function needs.

For simplicity in the *Scripting Guide*, we use the word *argument* in both cases.

A *named argument* is an optional argument that you select from a predetermined set and explicitly define. For example, `title("My Line Graph")` in the `Graph Box()` function is a named argument because the title is explicitly defined as such.

```
Graph Box( title( "My Line Graph" ),
      Frame Size( 300, 500 ),
      Marker( Marker State( 3 ), [11 44 77], [75 25 50] );
      Pen Color( "Blue" );
      Line( [10 30 70], [88 22 44] ));
```

Note that the `Frame Size()` arguments `300` and `500` are not named. The position of these arguments implies meaning; the first argument is always the width, the second argument is always the height.

Optional Arguments

Functions and messages require certain arguments, and other arguments are optional. You can include them, but you do not have to. In specifications, optional arguments are enclosed in angle brackets. For example:

```
Root( x, <n> )
```

The *x* argument is required. The *n* argument is optional.

Optional arguments often have a default value. For example, for `Root()`, the default value of *n* is 2:

Code	Output	Explanation
`Root( 25 )`	`5`	Returns the square root of 25.
`Root( 25, 2 )`	`5`	Returns the square root of 25.
`Root( 25, 3 )`	`2.92401773821287`	Returns the cube root of 25.

Or and the Vertical Bar Symbol

A single vertical bar (`|`) represents a logical `OR`. For brevity, `|` represents the word *or* when referring to alternative values.

For example, a pathname can be either absolute or relative. When you see an argument such as `absolute|relative`, this means that you enter *either* one of the following two options:

- `absolute` indicates an absolute pathname.
- `relative` indicates a relative pathname.

More than two options can also be strung together with a vertical bar in this way.

Script Formatting

Whitespace characters (such as spaces, tabs, and newlines) and capitalization are ignored in JSL. This means that the following two expressions are equivalent:

Expression 1:

```
sum = 0;
For( i = 1, i <= 10, i++,
   sum += i;
   Show( i, sum );
);
```

Expression 2:

```
Sum = 0;
For( i = 1, i <= 10, i++,
   Sum += i;
   Show( i, Sum );
);
```

You can format your script in any way that you like. However, the script editor can also format your script for you. The *Scripting Guide* uses the script editor's default formatting for capitalization, spaces, returns, tabs, and so on. See "Work with the Script Editor" for more information about using the script editor.

Note: The only white space exception is two-character operators (such as <= or ++). The operators cannot be separated by a space.

Basic JSL Syntax

A JSL script is a series of expressions. Each expression is a section of JSL code that accomplishes a task. JSL expressions hold data, manipulate data, and send commands to objects.

Many expressions are nested message names. Message contents are enclosed in parentheses:

```
Message Name ( argument 1, argument 2, ... );
```

The meaning of JSL names depends on the context. The same name might mean one thing in a data table context and something entirely different in a function context. See "Rules for Name Resolution".

Almost anything that follows certain punctuation rules, such as matching parentheses, is a valid JSL expression. For example:

```
win = New Window( "Window Example",
   <<Modal,
   Text Box( "Hello, World" ),
   Text Box( "-----" ),
   Button Box( "OK" )
);
```

Notes:

- Names can have embedded spaces. See "Names".
- Message contents are enclosed in parentheses, which must be balanced. See "Parentheses".
- Items are separated by commas. See "Commas".
- JSL is not case sensitive; you can type "`text box();`" or "`Text Box()`".

- Messages are commonly nested inside other messages.

Book Conventions

In the *Scripting Guide*, the function names that are case sensitive are capitalized as you must use them. Arguments that are placeholders for actual choices are in `lowercase`. For example, `Connect Color` is a function that you need to type as is, and `color` stands for some color choice that you make yourself.

```
Connect Color(color);
```

In this case, the argument in parentheses must be some color value (for example, a JMP color number, or a supported color name like `"red"`, `"blue"`, and so on, or an RGB value given as a list, such as `{.75, .50, .50}`). Sometimes alternatives like these are shown with the vertical bar (`|`) character for "or," like this:

```
Connect Color( number | "color name" | {r,g,b} );
```

Syntax coloring is applied to scripts that you can paste into a script editor and then run.

Chapter 3

Getting Started

Let JMP Write Your Scripts

You often have to produce the same reports for the same data on a regular basis. This chapter shows you how to let JMP write scripts for common tasks like importing text data, opening Microsoft Excel files, and producing reports. A final tutorial shows you how to put it all together into a single script to open an Microsoft Excel file and produce three reports automatically.

The *Scripting Guide* is written for users who are familiar with JMP but might not be familiar with JSL. For more information about performing common tasks, see *Using JMP*. *Discovering JMP* is also a good resource for learning basic concepts and understanding the JMP workflow.

Contents

Capturing a Script for an Analysis Report 51
Capturing a Script for a Data Table 52
Capturing a Script to Import a File 53
Gluing Scripts Together 54

Capturing a Script for an Analysis Report

Here are the basic steps for capturing a script to reproduce an analysis:

1. Launch a platform, such as Distribution.
2. Make any changes or additions that you need. For example, add tests and other graphs.
3. Capture the script to recreate your results.

You can save the script in the data table, so that if you send the data table to others, they can run your script and duplicate your reports.

Example

Follow these steps to produce a distribution report, capture the script to reproduce it, and save it to the data table.

Note: The data tables that you use in examples are located in JMP's Samples/Data folder.

1. Select **Help > Sample Data Library** and open Companies.jmp.
2. Select **Analyze > Distribution** to open the Distribution launch window.
3. Select Profits ($M) in the Select Columns box and click the **Y, Columns** button.
4. Click **OK**.

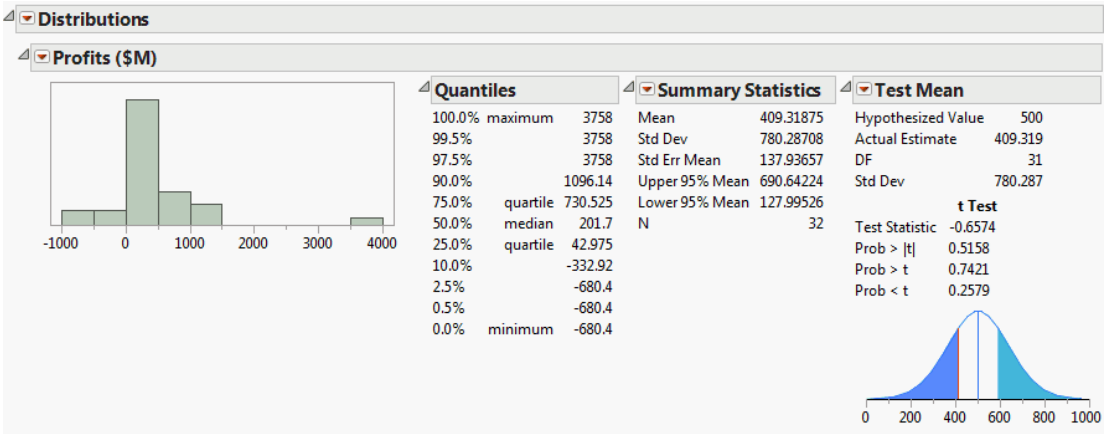
   The Distribution report window appears.
5. Click the Distributions red triangle and select **Stack** to make your report horizontal.
6. Click the red triangle next to Profits ($M) and deselect **Outlier Box Plot** to turn the option off.
7. Click the red triangle next to Profits ($M) and select **Test Mean**.

   The Test Mean window appears.
8. Type 500 in the **Specify Hypothesized Mean** box.
9. Click **OK**.

   The test for the mean is added to the report window.

   Now you have your customized report.

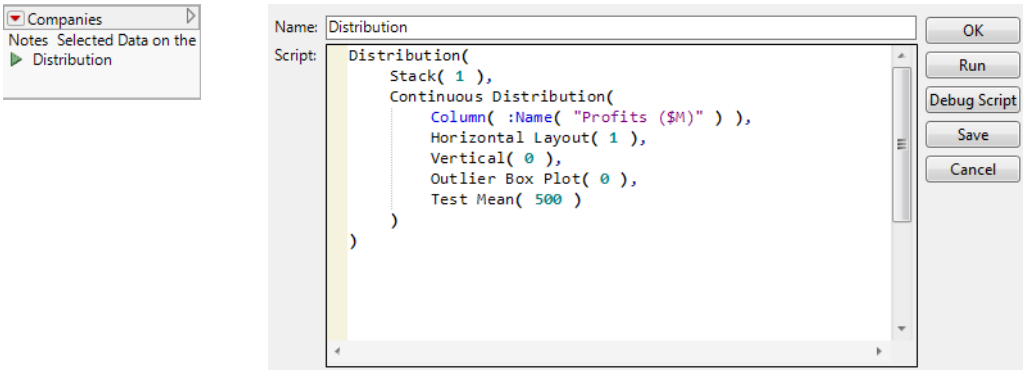
Figure 3.1 Customized Distribution Report



10. Click the Distributions red triangle and select **Save Script > To Data Table**.

    Your data table now has a script named Distribution saved to it. Right-click the Distribution script and select **Edit** to see the script.

Figure 3.2 Distribution Script Saved to the Data Table



11. To run the script and reproduce your final report exactly, click the green triangle next to the script.

Capturing a Script for a Data Table

Here are the basic steps for capturing a script to reproduce a data table:

1. Open the data table.

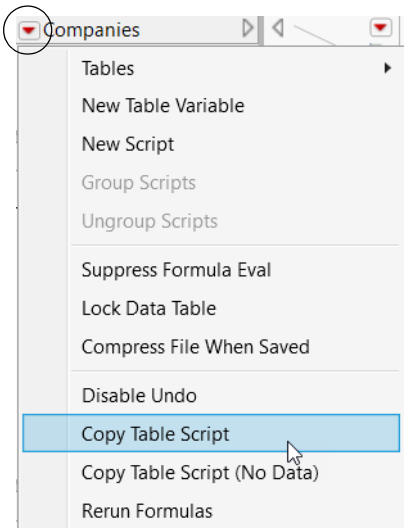
2. Make any changes that you need. For example, add a script, correct values, add new columns.
3. Capture the script to recreate your data table.

Example

Use the data table from the previous example, where you saved a script to it.

1. In the data table, select the red triangle next to the data table's name.
2. Select **Copy Table Script**.

Figure 3.3 Copy the Table Script



3. Open a script window by selecting **File > New > Script**.
4. Select **Edit > Paste**.

You now have a script that duplicates your data table. You can save this script and run it at any time to recreate your data table, with all its scripts attached.

Capturing a Script to Import a File

To capture a script that imports a file, you open the file in JMP. JMP automatically records the steps that occurred when you opened the file.

Import a Text File

1. Select **File > Open**.

The Open Data File window appears.

2. Select **Text Files** from the list next to **File name**.
3. In the Open as section, select **Data (Best Guess).**

   JMP formats the data based on tabs, commas, white space, and other characters in the text file.
4. Browse to select the file, and then select **Open**.

   The file is opened as a data table. The data table includes a script named Source. This JSL script imports your text file with the text import rules that you used.
5. Right-click the Source script and select **Edit**.

   You can copy this script, paste it into a new script window, and save it. Then you can run this script later to reimport the text file.

Tip: The import script is an `Open()` expression that specifies the text file and the import options to correctly import the file into JMP. The first part of this expression is the pathname to the specific file that you imported. If you save this script and want to run it a different place, you might need to edit the pathname so that it points to the text file. Pathnames are discussed in greater detail in "Path Variables".

Gluing Scripts Together

Suppose new data is saved out to an Microsoft Excel file once a week, and you need to produce the same reports every week. You could open the file and perform the same steps every week. However, creating a script that imports the new Microsoft Excel file into JMP and runs all analyses automatically is more efficient. The following example shows you how to set up your script and run it each week.

Import the Microsoft Excel File

1. Open a new script window (**File > New > Script**).
2. In your script window, enter the `Open()` expression to open the Solubil.xlsx sample import data file. The file is located in JMP's Samples/Import Data folder.

   ```
   dt = Open( "$SAMPLE_IMPORT_DATA/Solubil.xlsx" );
   ```

   Be sure to put the semicolon at the end of this expression, because you will add more expressions. The semicolon glues expressions together.
3. Run your script to import the Microsoft Excel file by selecting **Edit > Run Script**.

   The Microsoft Excel file opens as a data table.

Notes:

- You can also include the `Excel Wizard` argument in the `Open()` expression to preview the worksheet before importing it. See "Import Data from a Microsoft Excel File".
- You can specify an absolute or relative path to the file rather than using a path variable. For relative links, the script and file being opened must be in the same relative location each time you run the script. With absolute links, make sure that other users running the script have access to the file's location. See "Path Variables" for more information about using pathnames.

Run Your Reports and Capture Their Scripts

You have three reports to produce: a distribution report, a 3D scatterplot, and a multivariate report. Perform each one using the GUI, and add its script to the script window.

1. With your new data table open, select **Analyze > Distribution**.
2. Select all the columns except Labels and click **Y, Columns**.
3. Click **OK**.
4. Press Ctrl, click the eth red triangle, and then select **Histogram Options > Show Counts**.

   Bar counts are added to all six histograms.
5. In the Distribution window, click the Distributions red triangle and select **Save Script > To Clipboard**.
6. Click a line or two after your `Open()` expression and select **Edit > Paste**.
7. Type a semicolon after the last close parenthesis.
8. Select **Graph > Scatterplot 3D**.
9. Select all the columns except Labels and click **Y, Columns**.
10. Click **OK**.
11. Copy and paste the script for Scatterplot 3D into the script window just like you did for your Distribution report. Be sure to add the semicolon at the end.
12. Select **Analyze > Multivariate Methods > Multivariate**.
13. Select all the columns except Labels and click **Y, Columns**.
14. Click **OK**.
15. Copy and paste the script for Multivariate into the script window just like you did for Distributions and Scatterplot 3D.

    You should see the following script:

```
dt = Open( "$SAMPLE_IMPORT_DATA/Solubil.xlsx" );
Distribution(
  Continuous Distribution( Column( :eth ), Show Counts( 1 ) ),
  Continuous Distribution( Column( :oct ), Show Counts( 1 ) ),
```

```
    Continuous Distribution( Column( :cc14 ), Show Counts( 1 ) ),
    Continuous Distribution( Column( :c6c6 ), Show Counts( 1 ) ),
    Continuous Distribution( Column( :hex ), Show Counts( 1 ) ),
    Continuous Distribution( Column( :chc13 ), Show Counts( 1 ) ),
);
Scatterplot 3D(
     Y( :eth, :oct, :cc14, :c6c6, :hex, :chc13 ),
     Frame3D( Set Grab Handles( 0 ), Set Rotation( -54, 0, 38) )
);
Multivariate(
   Y( :eth, :oct, :cc14, :c6c6, :hex, :chc13 ),
   Estimation Method( "Row-wise" ),
   Scatterplot Matrix( Density Ellipses( 0 ), Shaded Ellipses( 0 ) )
);
```

Save the Script

You now have a script that reproduces all of the steps that you performed manually. Save the script, and close your data table and all its report windows.

1. In the script window that contains your script, select **File > Save** or **File > Save As**.
2. Specify a filename (for example, Weekly Report).
3. Click **Save**.

Run the Script

As long as your weekly updated Microsoft Excel file is saved in the same place and contains the same columns, you can run your script and automatically produce all your reports.

1. Open the script that you saved.
2. Select **Edit > Run Script**.

   Your Microsoft Excel file is opened in JMP, and all three of your reports appear.

You can send this script to others. As long as they have access to the same Microsoft Excel file in the same location, they can also run the script in JMP and see your reports.

Advanced Note: Auto-Submit

If you want a particular script to always be executed instead of opened into the script window, put the following command on the first line of the script:

```
//!
```

If this is not the very first line, with nothing else on the same line, this command does nothing.

You can override this command when opening the file.

1. Select **File > Open**.

2. Press Ctrl, select the JSL file, and then click **Open**.

   The script opens into a script window instead of being executed.

The command is also ignored when you right-click the file in the Home Window and select **Edit Script**.

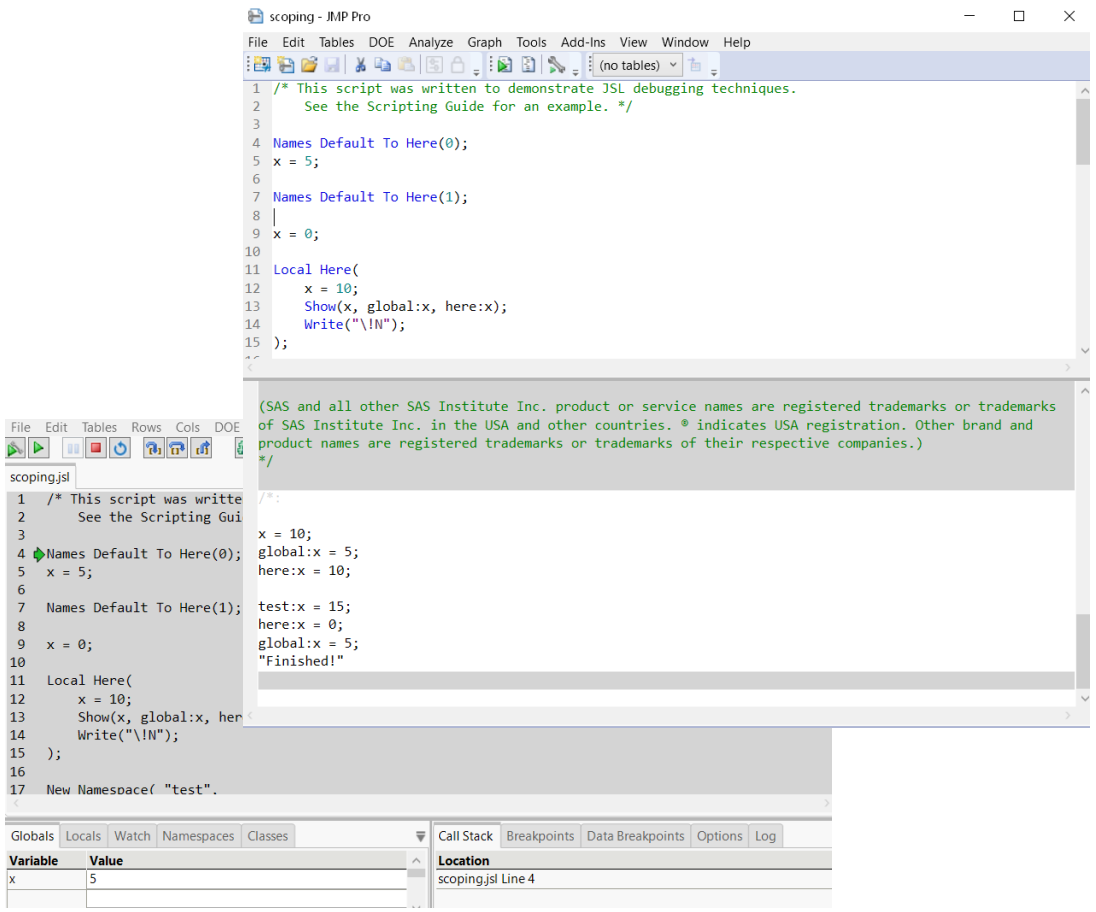
Chapter 4

Scripting Tools

Using the Script Editor, Log Window, Debugger and Profiler

JMP provides several programming tools for script writers. The script editor supports syntax coloring, autocompletes functions as you type, highlights matching braces, allows for code folding, and has additional features to help you develop scripts more quickly. Error messages and output are shown in the log window, which can be displayed inside the script editor. The JMP Scripting Language (JSL) Debugger and Profiler can help you troubleshoot your scripts.

Figure 4.1 Script Editor with Embedded Log and the Debugger



Contents

Work with the Script Editor . . . 61
Introduction to the Script Editor . . . 61
Run a Script . . . 62
Stop a Script . . . 63
Edit a Script . . . 63
Color Coding and Themes . . . 63
Autocomplete Functions . . . 64
Tooltips . . . 64
Split a Window . . . 65
Match Parentheses, Brackets, and Braces . . . 66
Select a Rectangular Block of Text . . . 67
Select Fragmented Text . . . 68
Drag and Drop Text . . . 68
Find and Replace . . . 68
Automatic Formatting . . . 69
Errors Are Marked by Special Characters . . . 69
Typing in Multiple Rows at Once . . . 69
Add Code Folding Markers . . . 70
Advanced Options . . . 71
Set Preferences for the Script Editor . . . 72
Work with the Log . . . 72
Enhanced Log in JMP . . . 73
Text Log in JMP . . . 78
Debug or Profile Scripts . . . 79
Debugger and Profiler Window . . . 80
Work with Breakpoints . . . 84
View Variables . . . 87
Work with Watches . . . 87
Modify Preferences in Debugger . . . 88
Persistent Debugger Sessions . . . 89
Examples of Debugging and Profiling Scripts . . . 89

Work with the Script Editor

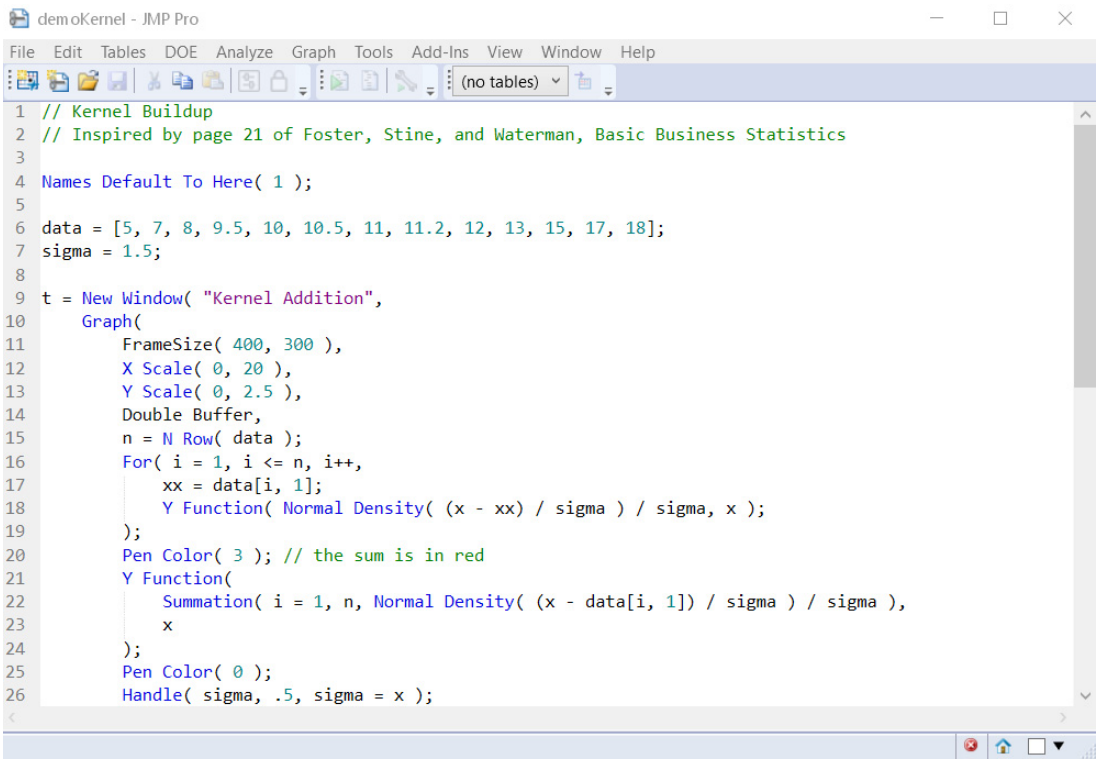
- "Introduction to the Script Editor"
- "Run a Script"
- "Stop a Script"
- "Edit a Script"
- "Color Coding and Themes"
- "Autocomplete Functions"
- "Tooltips"
- "Split a Window"
- "Match Parentheses, Brackets, and Braces"
- "Select a Rectangular Block of Text"
- "Select Fragmented Text"
- "Drag and Drop Text"
- "Find and Replace"
- "Automatic Formatting"
- "Errors Are Marked by Special Characters"
- "Typing in Multiple Rows at Once"
- "Add Code Folding Markers"
- "Advanced Options"
- "Set Preferences for the Script Editor"

Introduction to the Script Editor

The script editor provides a friendly environment for writing and reading JSL scripts. Figure 4.2 shows basic features such as syntax coloring, inline commenting, and automatic formatting. Other common programming options are described later in this section.

Script editor features are also available in the log window and anywhere else that you can edit or write a script (for example, in the Scripting Index or Application Builder).

Figure 4.2 The Script Editor



Tip: Consider setting the *Autosave timeout* value in the General preferences to automatically save open scripts at the specified number of minutes. This autosave value also applies to data tables, journals, scripts, projects, and reports.

Run a Script

To run an entire script, select **Edit > Run Script.**

To run specific lines in a script, select those lines and then select **Edit > Run Script**.

To run specific lines that are not adjacent, press Ctrl, select the lines, and then select **Edit > Run Script**.

On Windows, you can also click in a line or select several lines and press Enter on your numeric keypad.

Run the script automatically when you open it by using one of the following methods:

- Type `//!` on the first line.
- Include `Run JSL( 1 )` in the `Open()` statement:

```
Open( "$SAMPLE_SCRIPTS/scoping.jsl", Run JSL( 1 ) );
```

Stop a Script

To stop the script, press Esc on Windows (or Command-period on macOS). You can also select **Edit > Stop Script**. On macOS, **Edit > Stop Script** is available only when the script is running.

Edit a Script

To edit a script on Windows, press Insert. You can then type directly over (overwrite) any existing JSL code. Note that this feature is not available on macOS.

Color Coding and Themes

On Windows, the default font in the script editor is 10-point Consolas. On macOS, the default script editor font is 13-point Menlo.

JMP applies the following colors in the script window:

- black for text, identifiers (JSL functions), braces, and user macros
- white for the script editor background
- gray for the disabled background and guides
- green for comments
- purple for strings
- maroon for platform names

  Note: If a platform name is used in a message, the JSL messages color is used.

- teal blue for numbers
- dark blue for operator symbols, the first keyword, and JSL messages
- medium blue for function names, second and third keywords, and macros
- red for unknown objects

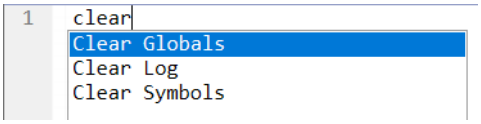
You can customize colors and fonts in the Customize Styles section of the Script Editor preferences. Syntax coloring for JSL, the log, SAS code, JavaScript, Text, JMP Text Analytics, R, Matlab, Python, C, JSON, and SQL are supported. In addition, options for dark and light themes are provided.

Autocomplete Functions

If you do not remember the exact name of a function, use autocompletion to see a list of functions that match what you have typed so far. Type part of the name, and then press Ctrl+Space on Windows (Option+Esc on macOS).

Suppose that you want to clear your JSL variables, but do not remember the command. You can type `clear` and then press Ctrl+Space, to see a list of possible clear commands. Select the command that you want to insert.

Figure 4.3 Autocomplete Example

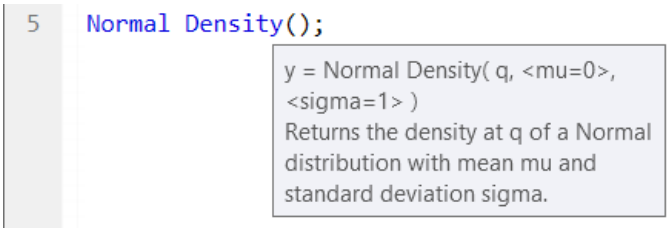


Tooltips

If you are using a function and do not remember the syntax or need more information about it, place the cursor over it to see a brief explanation. This works only with JSL function names, not platform commands, messages, or user-created functions. JSL function names are colored blue in the script editor.

The tooltip shows the syntax, arguments, and a brief explanation of the function (Figure 4.4). The tip also appears in the script editor window status bar.

Figure 4.4 Tooltip for a JSL Function



After running a script, you can also place the cursor over variable names to see their current value. To turn off variable tooltips, deselect **Preferences > Script Editor > Show Variable Value Tips**.

To turn off function tooltips, deselect **Preferences > Script Editor > Show Operator Tips**.

Tip: In addition to showing the tooltip for a function, you can display its Scripting Index entry. Run the script, press Alt, and then double-click the function. A description, sample script, and information about syntax is provided. You can also read help for the topic by running the script, right-clicking the function, and selecting **Help Browser**.

Example of a Tooltip for a JSL Variable

1. Enter and run the following line in a script window:

   ```
   my_variable = 8;
   ```

2. Hover over the variable name after you run the line.

   A tooltip shows the name of the variable and its value: 8.

3. Enter and run the following line:

   ```
   my_variable = "eight";
   ```

4. Hover over the variable name after you run the line.

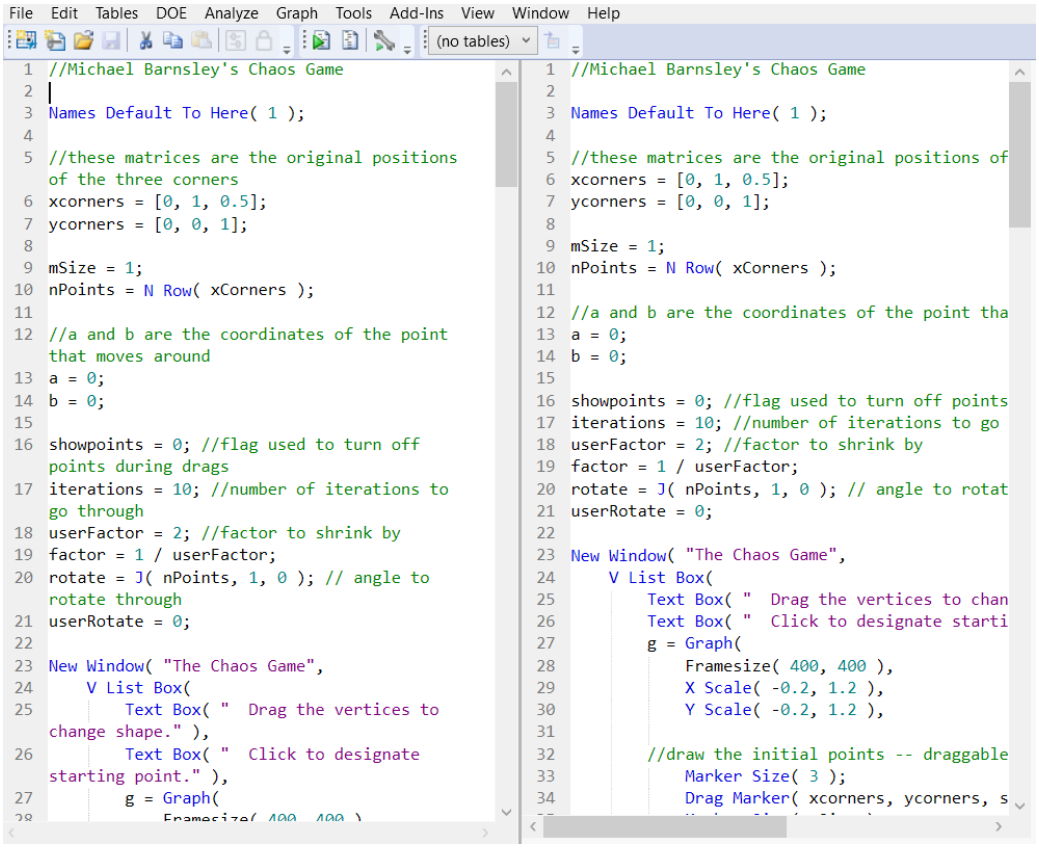
   A tooltip shows the name of the variable and its value: “eight”.

Split a Window

You can split the Script Editor window into two vertical or horizontal windows. This feature enables you to independently scroll through your code in two different places and edit the contents in both. When you make a change in one window, the change is immediately reflected in the other window.

- To split an open Script Editor window, right-click in the window and select **Split > Horizontal** or **Vertical**.
- To revert to a single window, right-click and select **Remove Split**.

Figure 4.5 Example of Splitting a Window Horizontally

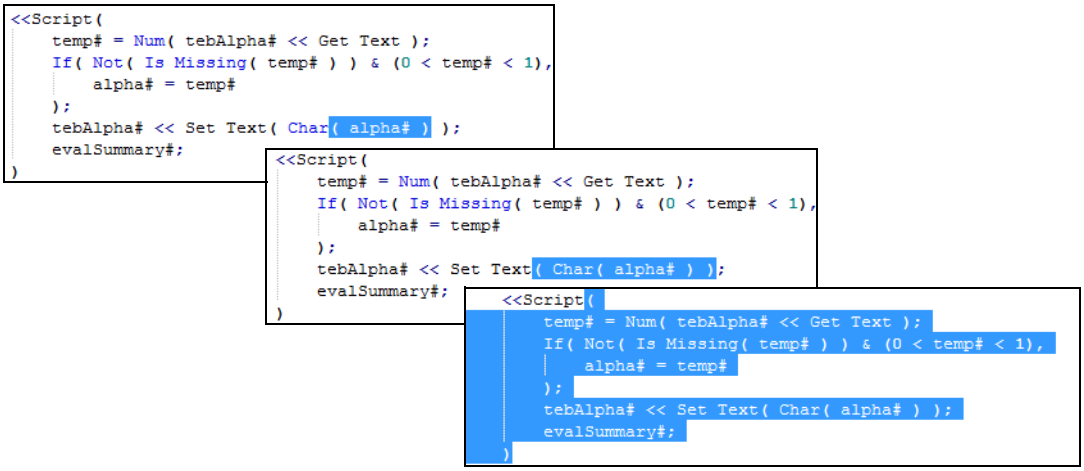


Match Parentheses, Brackets, and Braces

The script editor helps you match *fences* (or parentheses, square brackets, and braces) in the following ways:

- The matching closing fence is added when you type an opening fence.
- When you click next to either an opening or closing fence, the fence and its match are highlighted in blue. If the fence does not have a match, it is highlighted in red.
- If you double-click a fence, everything between the matching fence is selected (including the fences).
- If you put your cursor within an expression and press Ctrl+] on Windows (Command+B on macOS), the entire expression is selected. Fences that enclose the expression are included. Repeat this process to highlight the next-higher expression. Figure 4.6 shows an example.

Figure 4.6 Each Step in Matching Fences



When you type an opening brace, JMP adds the closing brace. Enter code between the braces, type the closing brace, and then your cursor automatically moves after the closing brace that JMP added. This prevents you from accidentally adding an unnecessary closing brace.

You can turn on and off the autocompletion of braces in the JMP preferences. See "Set Preferences for the Script Editor".

Select a Rectangular Block of Text

To select a rectangular block of text, press Alt and drag your cursor from the starting point to the end of the block. You can either copy or cut the text enclosed in the block.

Suppose that you want to the select all of the following code except for the comment marks.

```
// Y( :Y ),
// X( :X ),
```

Select a rectangular portion beginning with Y. When you paste, you get the following code:

```
Y( :Y ),
X( :X ),
```

The rectangular selection inserts returns where needed to maintain the structure of the text. Select `Get Menu Item State` on both lines in the following example.

```
bb << Get Menu Item State(1),
bb << Get Menu Item State(2),
```

When you paste, a return is inserted at the end of each line.

```
Get Menu Item State
Get Menu Item State
```

Select Fragmented Text

To select text that is not contiguous, press Ctrl on Windows or Command on macOS and drag your cursor over the text. Continue this action for any other text that you want to select. You can then copy and paste your selection into a new script or you can run the selected text. Text will be pasted or run in the order it was selected.

Drag and Drop Text

You can drag and drop text within a script editor window or between windows or from a data table into a script editor window. On Windows, pressing Ctrl before dragging and dropping copies the text. On macOS, the text is copied by default.

Drag and drop text in any of these ways:

- Select a row or column, pause, and then drop it into the script editor window.
- Double-click text in a text field and then drop it into the script editor window. Examples are text in a data table cell and any other selectable text.

On Windows, you can also drag and drop text into a minimized window.

1. Drag the text over the Home Window button in the lower right corner of the window.
   The Home Window appears.
2. In the Home Window list, drag the text over the destination window.
   That window appears.
3. Drop the text where you want it.

Find and Replace

Many find and replace options are available in the script editor, including the support of regular expressions. For example, searching with the following regular expression:

```
get.*name
```

returns messages such as “Get Button Name”, and “GetFontName”

Basic regular expressions such as `^` and `$` (which match the start of line and end of line) and `\n` (which matches a carriage return) are also supported.

See *Using JMP* for more information about the Search options.

Automatic Formatting

The script editor can format a script for easier reading. Any generated script (for example, by saving a platform script) is automatically formatted with tabs and returns in appropriate places.

You can also reformat individual scripts that are difficult to read (for example, scripts in which all commands are strung together with no whitespace characters). From the **Edit** menu, select **Reformat Script**.

Tip: This command alerts you if your script is badly formed (for example, if your parentheses are not matched).

Errors Are Marked by Special Characters

After you run a script, the /*###*/ characters in the log indicate the location of an error in the script execution. The following example shows a script in which the height column name is misspelled.

```
Names Default To Here( 1 );
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt:heihgt[3]; // misspelled column name
```

After you run the script, the log reads:

```
Scoped data table access requires a data table column or variable in access or
   evaluation of 'dt:heihgt' , dt:heihgt/*###*/
In the following script, error marked by /*###*/
Names Default To Here( 1 );
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt:heihgt/*###*/[3];
```

/*###*/ directly follows the misspelled height argument.

Typing in Multiple Rows at Once

When you edit a script, you can type in multiple rows at once instead of manually updating the same code on multiple lines. For example, in the following script, you want to change the s variable to aString. You can easily change the first instance of s to aString. To change the other instances at once, hold down Alt and select the column of s variables. Type aString once.

```
s = "";
For( i = 1, i <= n, i++,
   s += "Hello";
   s += "Number";
```

```
    s += Char( i );
);
```

The script then looks like this:

```
aString = "";
For( i = 1, i <= n, i++,
   aString += "Hello";
   aString += "Number";
   aString += Char( i );
);
```

Note: You can type in multiple rows at once, but you can type only in the same column. In the preceding example, the `s` variables are in the same column.

Add Code Folding Markers

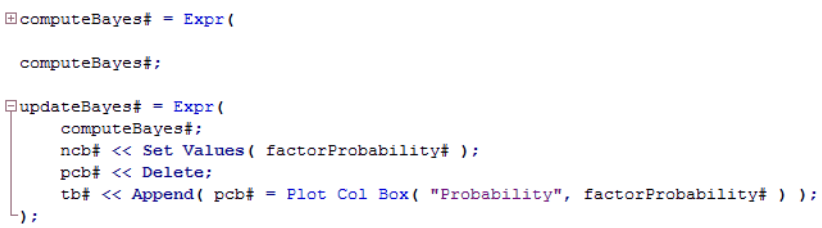
You can add code folding markers that show the beginning and the end of the code block, enabling you to collapse and expand code inside stand-alone functions.

To turn on this feature, select **Code folding** in the Script Editor preferences. Then you can expand and collapse blocks of code by right-clicking on a script and selecting **Advanced > Expand All** or **Collapse All**.

Tip: To fold or expand all code in the entire script, press Shift+Ctrl and then click a code marker.

After you select this preference, `Function` and `Expr` expressions are foldable. See "Add More Folding Keywords" for more information about adding folding markers to other expressions.

Figure 4.7 Code Folding Markers Shown in a Script



By default, code does not remain collapsed after you save the script and restart JMP. To save the state of the folded code, select **Save and restore document state information** in the Script Editor preferences.

Add More Folding Keywords

Custom code folding is supported for other stand-alone functions as shown in the following example:

```
{"If", "For", "For Each Row", "While", "Try", "New Window", "V List Box",
  "H List Box"}
```

JMP supports multiple keyword lists. A system administrator can define a set of keywords in jmpKeywords.jsl and save the script in C:/ProgramData/SAS/JMP/ or designated directory listed below. You save your version of jmpKeywords.jsl in your C:/Users/<user>/Documents/ folder. JMP merges all keyword lists from the designated directories.

On Windows, the following directories are examined in the order listed:

- C:/ProgramData/SAS/JMP/16/
- C:/ProgramData/SAS/JMP/
- C:/Users/<user>/AppData/Roaming/SAS/JMP/16/
- C:/Users/<user>/AppData/Roaming/SAS/JMP/
- C:/Users/<user>/Documents/

On macOS, the following directories are examined in the order listed:

- /Library/Application Support/JMP/16/
- /Library/Application Support/JMP/
- ~/Library/Application Support/JMP/16/
- ~/Library/Application Support/JMP/
- ~/Documents/

Note that jmpKeywords.jsl is stored in the designated JMP directory, even if you are using JMP Pro.

Notes:

- The list in jmpKeywords.jsl is case insensitive.
- Code folding is not supported for messages, platforms, user-defined functions, and comments.
- After you edit and save the list in jmpKeywords.jsl, turn the **Allow additional code folding keywords** preference off and then back on for the changes to take effect. Messages in the log indicate that the keywords were loaded.

Advanced Options

Right-clicking on selected text in the Script Editor provides the following Advanced options:

Expand All (Appears only if JSL code folding is on.) Expands all blocks of code.

Collapse All (Appears only if JSL code folding is on.) Collapses all blocks of code.

Comment Block Makes the selected text comments.

Uncomment Block Uncomments the selected comments.

Make Uppercase Changes all selected text to uppercase.

Make Lowercase Changes all selected text to lowercase.

Set Preferences for the Script Editor

In the JMP preferences, customize the script editor settings such as the font, colors, and spacing options.

Setting the Fonts

1. Select **File > Preferences**.
2. Select the **Fonts** group.
3. Click **Mono** to set the font for the script editor.

For more information about font preferences, see *Using JMP*.

Set Script Editor Preferences

Select **File > Preferences > Script Editor** to further customize the editor. For more information about script editor preferences, see *Using JMP*.

Work with the Log

The Log in JMP has two modes:

- "Enhanced Log in JMP"
- "Text Log in JMP"

You can switch between the two modes by selecting **File > Preferences > Log**.

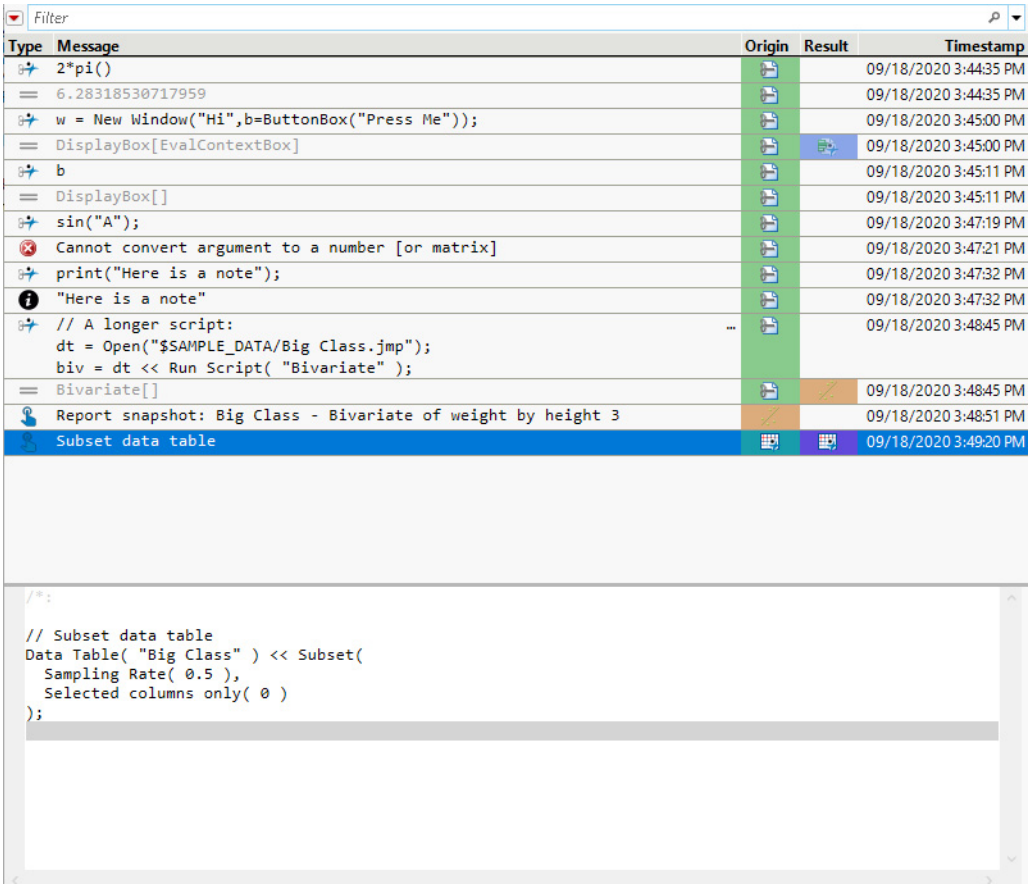
Note: The log shows the steps you take during data cleanup. For example, if you add a column property or import the data into Excel, then the JSL expressions for those actions are captured in the log. This enables you to clean up the data more efficiently when you receive a new data set.

Enhanced Log in JMP

The Enhanced Log is a tabular view of the events in a JMP session. To view the JMP Log, select **View > Log** (**Window > Log** on macOS). By default, the log in JMP appears in Enhanced mode. The log window contains events, or actions, in the rows of the table and the corresponding JSL code and output in a separate text pane. This tabular view of the events is easy to navigate, displays errors and warnings, and can help you create a JSL script of the actions in a JMP session. For an example of using the log to generate a JSL script for recorded actions, see "Example of Recording Actions in the Enhanced Log".

Note: When you used the embedded log in a script window, the embedded log does not use the Enhanced Log mode. Further, any script output from a script window with an embedded log does not appear in the Enhanced Log.

Figure 4.8 Enhanced Log



Each row of the tabular view contains information in the following columns:

Type Contains an icon that represents message type. For more information about the types of messages, see "Filter".

> **Tip:** You can hover over an icon in the Type column to see the type of message that a particular icon represents.

Message Contains a description of the event or result. Among other things, this could be a recorded action, JSL that was submitted, the result of JSL that was submitted, or an error message. Each description of a logged item or result is limited to three lines of text; longer items are denoted with an Ellipsis icon ··· in the upper right corner of the Message section of the row. The full text for longer items appears in the text pane at the bottom of the log window.

Origin Contains an icon that shows the type of JMP window from which the logged item or result was generated. If the Color by Window option is selected in the red triangle menu next to Filter, logged items generated by the same JMP window are specified by a common color. You can click on the icon in the Origin column to bring the corresponding window to the front.

> **Note:** Not all rows in the log table have an icon in the Origin column.

Result Contains an icon that shows the type of JMP window that contains the result. If the Color by Window option is selected, logged items generated by the same JMP window are specified by a common color. You can click on the icon in the Result column to bring the corresponding window to the front.

> **Note:** Not all rows in the log table have an icon in the Result column.

Timestamp Contains a date-time value that indicates when the logged item was generated.

Enhanced Log Options

The red triangle menu in the Enhanced Log contains the following options:

Go To (Available only when one or more rows of the log table are selected.) Brings the window corresponding to the selected row in the log table to the front. If an Origin window and a Result window both appear in the selected row, both windows are brought to the front with the Result window on top of the Origin window.

Run Scripts (Available only when one or more rows of the log table that contain a scriptable action are selected.) Runs the JSL script that corresponds to the selected rows in the log table.

Color by Window Specifies whether the rows in the log table are colored by the JMP window that created the row of the log output.

Clear (Available only when one or more rows of the log table are selected.) Removes the selected row or rows from the log table.

Caution: This action cannot be undone.

Clear Log Removes all rows from the log table.

Caution: This action cannot be undone.

Filter Contains the following options to filter what types of logged events appear in the log table.

Result Shows or hides rows that correspond to the results of running JSL.

Script Shows or hides rows that correspond to submitted JSL.

Log Shows or hides rows that correspond to informational log messages.

Warning Shows or hides rows that correspond to warning messages in the log.

Error Shows or hides rows that correspond to error messages in the log.

Action Shows or hides rows in the log table that correspond to interactive JMP actions. These actions include data table operations, platform launches, and closing report windows. You can customize which actions are shown in the log table by selecting specific options in the Action Recording submenu.

Action Recording Contains the following options to filter what types of interactive JMP actions appear in the log table.

Data Table Operations Shows or hides rows that correspond to manipulations of a data table. These actions can include row and column operations, as well as actions in the Tables, Rows, and Cols menus.

Platform Launch Shows or hides rows that correspond to launching a JMP platform.

Report Snapshot on Close Shows or hides rows that correspond to a JSL script that is saved when a report window is closed.

Save Script Contains the following options that save a script that reproduces all selected rows in the log table. If no rows are selected, these options save a script to reproduce all events recorded in the log table.

To Script Window Saves the script to a script window.

To Clipboard Saves the script to the clipboard.

Make Into Data Table Creates a data table that contains all rows from the log table. If rows are selected in the log table, they are also selected in the data table that is created.

Filter the Results in the Enhanced Log

To the right of the red triangle menu, use the search field to filter the items in the log table view.

Click the down arrow button next to the search box to refine your search.

Contains Terms Returns items that contain a part of the search criteria. A search for "ease oom" returns messages such as "Release Zoom".

Contains Phrase Returns items that contain the exact search criteria. A search for "text box" returns entries that contain "text" followed directly by "box" (for example, "Context Box" and "Text Box").

Starts With Phrase Returns items that start with the search criteria.

Ends With Phrase Returns items that end with the search criteria.

Whole Phrase Returns items that consist of the entire string. A search for "text box" returns entries that contain only "text box".

Regular Expression Enables you to use the wildcard (*) and period (.) in the search box. Searching for "get.*name" looks for items that contain "get" followed by one or more words. It returns "Get Color Theme Names", "Get Name Info", and "Get Effect Names", and so on.

Invert Result Returns items that do not match the search criteria.

Match All Terms Returns items that contain both strings. A search for "t test" returns elements that contain either or both of the search strings: "Pat Test", "Shortest Edit Script" and "Paired t test".

Ignore Case Ignores the case in the search criteria.

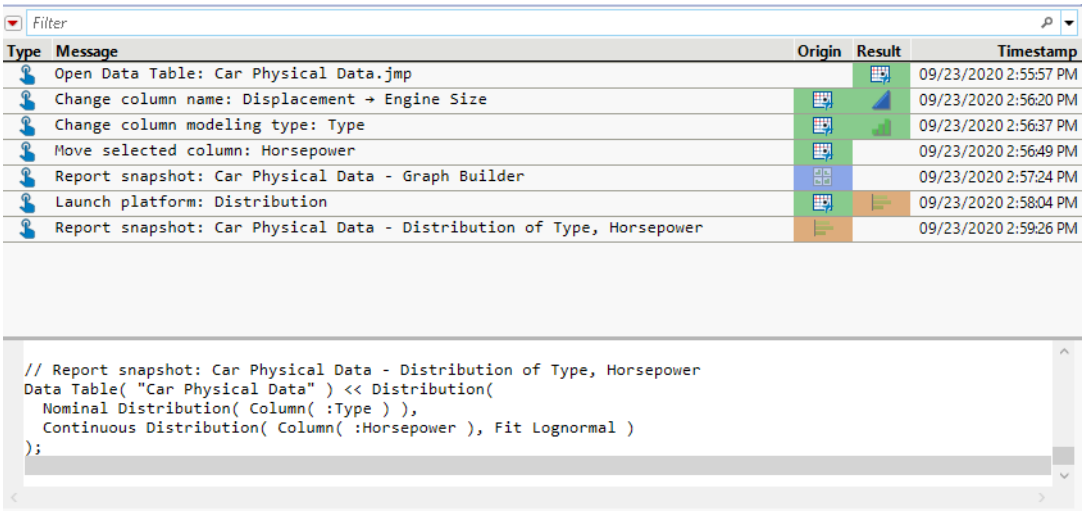
Match Whole Words Returns items that contain each word in the string based on the Match All Terms setting. If you search for "data filter", and Match All Terms is selected, entries that contain both "data" and "filter" are returned.

Example of Recording Actions in the Enhanced Log

You can use the Enhanced Log to record many of the interactive actions that you perform for data cleanup and analysis. In this example, the Enhanced Log is used to generate a JSL script for some data manipulation and analysis actions.

1. Select **View > Log**.
2. Click the red triangle next to the Filter box and select **Clear Log**.
3. Select **Help > Sample Data Library** and open Car Physical Data.jmp.
4. Right-click Displacement in the Columns panel in the data table, and select **Column Info**.
5. Type Engine Size for the Column Name. Click **OK**.
6. Right-click Type in the Columns panel in the data table, and select **Column Info**.
7. Select Ordinal for the Modeling Type. Click **OK**.
8. Click Horsepower in the Columns panel in the data table and drag it above Weight.
9. Select **Graph > Graph Builder**.
10. Drag Horsepower to the **Y** drop zone.
11. Drag Engine Size to the **X** drop zone.
12. Close the Graph Builder window.
13. Select **Analyze > Distribution**.
14. Select Type and Horsepower and click **Y, Columns**.
15. Click **OK**.
16. Click the red triangle next to Horsepower and select **Continuous Fit > Fit Lognormal**.
17. Close the Distribution window.
18. Select **View > Log**.

Figure 4.9 Enhanced Log



19. Press Ctrl and select the first, second, third, and last rows of the log table.
20. Click the red triangle next to the Filter box and select **Save Script > To Script Window**.

The script to reproduce the selected actions appears in a new script window. You can save this script to reuse later or use it as a starting point for a more complex analysis.

Text Log in JMP

The Text Log in JMP is a text-based log of JSL scripts that have been run during a JMP session and the results of those scripts. When the log is open, output appears in the log after you run a script. The script that was run is shaded in the log, and the corresponding output appears beneath it. To view the log in Text mode, select Text for the Mode preference after selecting **File > Preferences > Log**.

Syntax and compatibility errors are reported in the log, including the line number, script filename, and code that JMP could not process. Many interactive JMP actions are also recorded in the log.

Tips:

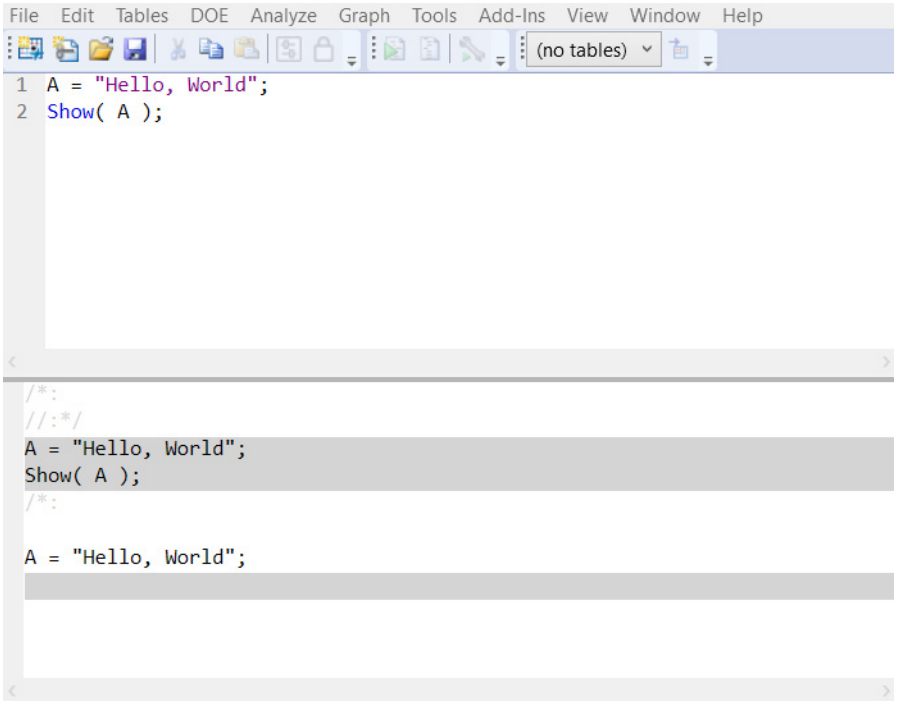
- To omit compatibility warnings from the log, deselect **Show log warnings for JSL compatibility changes in 12** in the JMP General preferences.
- To choose which actions are recorded in the log, you can use the Action Recording preferences after selecting **File > Preferences > Log**.

Show the Text Log in the Script Window

You can view the log inside the script window by right-clicking and selecting **Show Embedded Log**. To view the embedded log by default, select **Show embedded log on script window open** in the Script Editor preferences. This option makes it easy to edit and run a script, quickly see the results of your changes, and then continue to develop the script. The embedded log uses the Text log mode. See "Text Log in JMP".

The embedded log always appears in the Scripting Index script window, but it is not available in Application Builder and the Debugger.

Figure 4.10 The Script Window with an Embedded Text Log



Debug or Profile Scripts

In an open script, click the **Debug Script** button (or select **Edit > Debug Script**) to show the script in the JSL Debugger window. You can also use a keyboard shortcut:

- Press Ctrl+Shift+R (Windows).
- Press Shift+Command+R (macOS).

The JSL Debugger helps identify the point at which a script causes an error or fails. Rather than commenting out portions of the script or adding `Print()` expressions, you can use the Debugger to find the problem.

Once the JSL Debugger appears, you can continue in this mode, or you can click the **Profile JSL Script** button to move into the JSL Profiler mode. The JSL Profiler helps with optimization. You can profile your scripts during execution to see how much time is spent executing particular lines or how many times a particular line is executed.

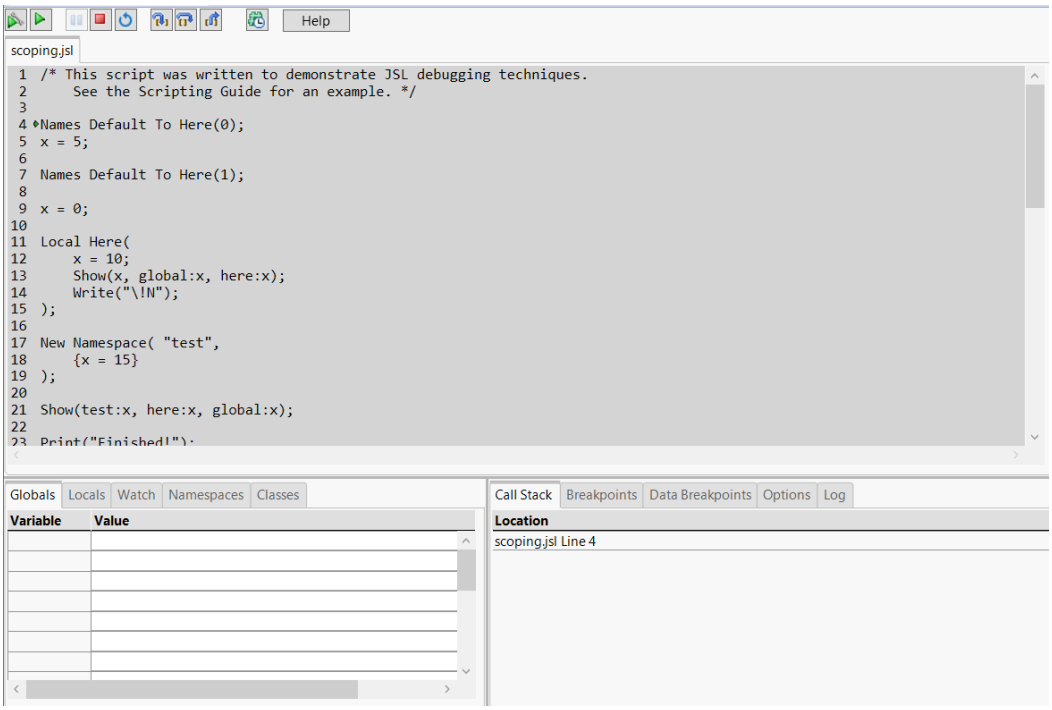
Tip: To debug a script automatically when you open it, include `Debug JSL( 1 )` in the `Open()` statement:

```
Open( "$SAMPLE_SCRIPTS/scoping.jsl", Debug JSL( 1 ) );
```

Debugger and Profiler Window

The Debugger opens in a new instance of JMP (Figure 4.11). The original instance is inoperable until the script produces something that requires interaction. At that point, the Debugger window becomes inoperable until you perform whatever action is required. Then control is returned to the Debugger. Close the Debugger to work again in the original instance of JMP.

Figure 4.11 The Debugger Window



Use the buttons at the top to control the Debugger or the JSL Profiler. One or more scripts that you are debugging or profiling are shown in tabs. If your script includes other scripts, each one opens in a new tab.

Tabs in the bottom portion of the Debugger provide options to view variables, namespaces, the log, and the current execution point; work with breakpoints; and set options.

Tip: To view or edit long values on a tab in the Debugger, right-click the value and select **Edit**. The code opens in a script editor.

Using the Execution Buttons

Use the buttons at the top to control execution of the script within the Debugger or JSL Profiler.

Table 4.1 Description of the Debugger Buttons

Button	Button Name	Action
	Run	Runs the script in the Debugger until it reaches either a breakpoint or the end of the script. **Note:** If there is an error in the script, an abbreviated error message is displayed at the top of the window. Click **More** to see the complete error message.
	Run without breakpoints Run profiler	Runs the script through the end without stopping.
	Break All	If the script is busy, click Break All to stop all action in the script and return to the Debugger or JSL Profiler (for example, if you are in a very long loop). The Debugger or JSL Profiler might not be able to break execution if the executing script is waiting on some interactive user action, such as completing a window or interacting with an opened window.
	Stop	Stops debugging the script and exits the Debugger or the JSL Profiler.
	Restart	Closes the current Debugger session and opens a new session.

Table 4.1 Description of the Debugger Buttons *(Continued)*

Button	Button Name	Action
	Step Into	Lets you step into a function or an included file. Otherwise, it behaves the same as Step Over.
	Step Over	Runs all expressions on a single line, or a complex expression that spans multiple lines, without stepping into a called expression, function, or `Include()` file.
	Step Out	Runs the current script or function to a breakpoint or the end and returns to the calling point. If you are in the main script and the Debugger reaches the end, a message appears: Program execution terminated. The Debugger remains open in order for you to inspect the final program conditions.
	Profile JSL Script	Opens the JSL Profiler. (Press the Run profiler button to start the JSL Profiler.) Use the JSL Profiler to see how much time is spent executing particular lines or how many times a particular line is executed. **Notes:** • You can switch back and forth between the Debugger and JSL Profiler modes only prior to the start of the program. • Some of the debugger buttons are disabled when profiling. • All breakpoints are disabled when running in the JSL Profiler mode.
	Show Profile by Line Count	Shows the number of times each line is executed.
	Show Profile by Time	Shows how much time is spent executing a line.
	Show Profile by Count	For line counts, shows the number of times the line is executed. For time, shows the number of microseconds (or milliseconds or seconds) the line takes to complete.
	Show Profile by Percent	For line counts, shows the individual line count divided by the total line count. For time, shows the percentage of time spent on an individual line (line time/total time*100).
	Time Units	Sets the time unit to microseconds, milliseconds, or seconds. Available in the JSL Profiler after you click the Run profiler button .

Table 4.1 Description of the Debugger Buttons *(Continued)*

Button	Button Name	Action
	Color Theme	Sets the color of the shading for the JSL Profiler. Available in the JSL Profiler after you click the Run profiler button .

Variable Lists

The tabs on the bottom left of the Debugger let you examine global variables, local variables, watch variables, and variables within namespaces.

Globals The Globals tab lists all global variables and updates their values as you step through the script. Each variable is added as it is initialized. If there are already global variables defined from running earlier scripts, they will be listed with their current values when you start the Debugger. See "View Variables".

Locals The Locals tab lists all variables by scope and updates their values as you step through the script. Select a scope in the menu. See "View Variables".

Watch If there is a particular variable or value of an expression whose values you want to watch as you step through the code, you can add them here. This is particularly useful if your script uses many variables that might be difficult to watch in the Globals or Locals lists. See "Work with Watches".

Namespaces As namespaces are defined, they are added to the menu. Select a namespace to view any variables and their values used within the namespace. See "View Variables".

Classes The Classes tab lists classes that are defined in the script. See "Classes".

Debugger Options

The tabs on the bottom right of the Debugger let you view the call stack, work with breakpoints, set options, and view the log.

Call Stack The call stack lists the current execution point in scripts and functions. The main script is always the first script listed. If you call a function, the function is added on top of the calling script. Likewise, any included files are added to the top of the list as you step through them. When you exit any function or script, it is removed from the list and you return to the next one in the list. The current line numbers are updated as you step through.

Double-click a row in the call stack to move the cursor to the specified line.

Breakpoints Add, edit, delete, and disable or enable breakpoints on lines. See "Work with Breakpoints". You can also double-click a row on the Breakpoints tab to move the cursor to the specified line.

Data Breakpoints Add, edit, delete, and disable or enable breakpoints in variables. Data breakpoints watch for changes in variables, unlike breakpoints, which watch for changes in lines.

Options Set the Debugger preferences interactively on this tab. See "Modify Preferences in Debugger".

Log The log from the script that you are debugging is shown on this tab.

Work with Breakpoints

A *breakpoint* interrupts the execution of a script. Although you can step through a script line by line, this can be tedious and lengthy for a long or complex script. You can set breakpoints at places of interest and simply run the script in the Debugger. The script is run normally until a breakpoint is reached. At the breakpoint, the Debugger stops executing the script so that you can look at the values of variables or start stepping line by line.

Breakpoints that appear on the Breakpoints tab in the lower right corner watch for changes in a line. Breakpoints that appear on the Data Breakpoints tab watch for changes in variables.

JMP preserves breakpoints across sessions. So when you close and reopen JMP, the breakpoints still appear.

Tip: Turn on line numbers by right-clicking in the script and selecting **Show Line Numbers**. You can also show line numbers by default in all scripts by modifying the Script Editor preferences.

Create a Breakpoint

When creating a breakpoint, you can specify settings such as conditions and break behavior. To do so, click on the Breakpoints tab or Data Breakpoints tab, and then enter the breakpoint information.

Otherwise, create a quick breakpoint by doing one of the following:

- In the Debugger margin, click the appropriate line (to the right of the line number if displayed).
- In the Debugger margin, right-click in the margin where you want the breakpoint and select **Set Breakpoint**.

The red breakpoint icon appears where you inserted the breakpoint and on the Breakpoints tab.

Delete Breakpoints

Do one of the following:

- In the Debugger margin, click the breakpoint icon.
- In the Debugger margin, right-click the breakpoint icon and select **Clear Breakpoint**.
- On the Breakpoints tab, select the breakpoint and then click .
- On the Breakpoints tab, click to delete all breakpoints (not just the selected breakpoints).

The red breakpoint icon is removed, and the breakpoint no longer appears on the Breakpoints tab.

Disable and Enable Breakpoints

Disabling a breakpoint is helpful when you potentially fix a problem and then want to see whether the script will run correctly past that breakpoint. You can then enable the breakpoint when necessary rather than re-creating it.

Do one of the following:

- In the Debugger margin, right-click the breakpoint icon and select **Enable Breakpoint** or **Disable Breakpoint**.
- On the Breakpoints tab, select or deselect the breakpoint's check box.
- On the Breakpoints tab, click to disable or enable all breakpoints.

A disabled breakpoint turns white; enabled breakpoints are shaded red.

Specify and Clear Conditional Expressions on Breakpoints

Setting a condition on a breakpoint is an alternative to single-stepping through code. Rather than single-step and view the variables for each expression, you specify that the script break only when a condition is met. Then you can step through the code and figure out where the problem arises.

Suppose that a calculation in your script is incorrect, and you suspect the problem occurs when `i==19`. Set a conditional breakpoint for `i==18`. The Debugger will run until that condition is met, then you can step through the code to identify the problem.

Specify a Breakpoint Condition

1. Right-click the breakpoint icon and select **Edit Breakpoint**.
2. On the Condition tab, select **Condition** and enter the conditional expression.
3. Specify whether to break when the expression **Is true** or **Has Changed**.
4. Click **OK**.

Disable or Enable a Condition

1. Right-click the breakpoint icon and select **Edit Breakpoint**.
2. On the Condition tab, deselect or select **Condition**.

Delete a Condition

On the Breakpoints tab, click in the breakpoint's Condition field and press Delete.

Specify Break Options

Right-clicking the breakpoint and selecting **Edit Breakpoint** provides a quick way to manage breakpoint behavior. Alternatively, select the breakpoint on the Breakpoints tab and click . Both methods display the Breakpoint Information window, where you customize settings on the Hit Count and Action tabs.

Change the Hit Count

You can control the number of times a breakpoint must be hit and when the break occurs. For example, to break when the condition is met twice, select **break when the hit count is equal to** and type 2 on the Hit Count tab.

Define an Action

You also have the option of defining a JSL expression or script that the Debugger executes when a breakpoint is hit and execution has stopped. This script is called an *action*. On the Action tab, enter the JSL expression to be executed.

Run the Script to the Cursor

When you right-click and select **Run To Cursor**, all expressions before the location of the cursor are executed. Select this option when you want to see values only up to the current line. To see values when each expression is executed, use the stepping options.

Tips for Setting Breakpoints

- If you do not want to watch for errors in a specific loop, set a breakpoint after the loop ends. The Debugger will hit the next breakpoint rather than stepping through each line of the loop.
- Avoid inserting a breakpoint in lines that do not trigger an action (such as comments, blank lines, and end parentheses). Debugger will not break on these lines.
- When you insert breakpoints, close Debugger, and edit the script, the breakpoints remain on the original line numbers. You might need to delete and then reinsert the breakpoints.

- Breakpoints are remembered across Debugger sessions. This means that your breakpoints list includes breakpoints that have been set in all scripts, not just the script that you are currently debugging.
- Breakpoints are remembered by the Debugger session, not by each script. This means that breakpoints are listed even for scripts that have been moved or deleted.
- On the Breakpoints tab, click to remove all breakpoints in scripts whether they are currently open or not, or for scripts that no longer exist.

View Variables

The variables lists are populated as they appear in the script. Their values are updated every time the script changes them. If you are uncertain why a variable has a particular value when you run your script, you can watch its value at every step to see what happens.

You can also assign the variables whatever value you want. For example, if you are stepping through a `For()` loop but are interested only in what happens starting with a particular iteration, you can assign your iterating variable that value. Step through the first part of the loop that initializes the iteration variable and then assign it the value that you want in the variable list at the bottom. Then when you step through, the loop begins executing at that point.

Tips for Managing Variables

- If you have run several scripts using the global scope, you might want to clear or delete global variables. This makes the list of variables in the Debugger shorter and relevant. Use the `Delete Symbols()` function to do so. You can also close JMP and restart to clear the space.
- If your script uses so many variables that they are difficult to find and watch in the variable lists, add watches for the specific variables in which you are interested.
- To view or edit long values on a tab in the Debugger, right-click the value and select **Edit**. The code opens in a script editor.

Work with Watches

JMP preserves the Watch variables across sessions. So when you close and reopen JMP, the Watch variables are still listed on the Watch tab.

Create a Watch

- On the Watch tab, click and enter the value in the window.

- In the Debugger, right-click the line that you want to watch, select **Add Watch**, and then enter the variable name in the window.
- In the Debugger, place the cursor in or next to a variable name (or select the variable name), right-click, and select **Add Watch**.
- On the Watch tab, enter the variable in an empty **Variable** field.

Modify a Watch

Do one of the following on the Watch tab to enter a new value:

- Select the watch and click .
- Click in the Variable field and enter a new value.

Delete Watches

Do one of the following on the Watch tab to delete watches:

- Select the watch and then click .
- Click to remove all watches.

Modify Preferences in Debugger

The Debugger lets you change preferences as you work in the Debugger. Select the Options tab to find the following settings:

Show Line Numbers Shows or hides the line numbers for the script in the Debugger.

Break on Multiple Statements Per Line Stops executing the script between each expression in a single line.

Break On Throw Breaks when the script executes the `Throw()` function. For example, `Throw()` might be enclosed in a `Try()` expression. The Debugger breaks on `Throw()` instead of continuing through the rest of the expression. This lets you identify where the problem occurred in the script and then return to debugging.

Break On Execution Error Stops executing the script when the error occurs rather than closing the Debugger.

Warn On Assignment In Condition Shows a warning when you enter a breakpoint condition that contains the assignment. For example, if you have a breakpoint on `x = 1` and add the condition `x = 1` to the breakpoint, you are prompted to verify the assignment of `x`.

Enter Debugger Upon Termination Keeps the Debugger open after a JSL program terminates execution. On by default, this option lets you examine attributes of the executed program.

Break For Compatibility Warnings Breaks when a JSL compatibility issue is found.

Persistent Debugger Sessions

JMP saves all breakpoints and watches until you delete them. Other user-specific settings, such as column widths on the tabs and the Debugger window size, persist between sessions of JMP.

These settings are stored in a file named JMP.jdeb, the location of which is defined in the USER_APPDATA variable:

- Windows 7 and higher: "/C:/Users/<username>/AppData/Roaming/SAS/JMP/16/"
- macOS: "/Users/<username>/Library/Application Support/JMP/16/"

As usual, the values of local variables, global variables, and namespaces clear when you close and reopen JMP.

Note: On Windows, the paths differ based on the JMP edition. In JMP Pro, the path refers to "JMPPro".

Examples of Debugging and Profiling Scripts

This section includes examples of setting breakpoints to watch variables; stepping into, over, and out of expressions; watching variables in different scopes and namespaces; debugging interactive scripts; and profiling scripts with the JSL Profiler.

Example scripts are located in the JMP Samples/Scripts folder.

Tip: Make sure that **Show Line Numbers** is selected on the Debugger Options tab before proceeding.

Using Breakpoints and Watching Global Variables

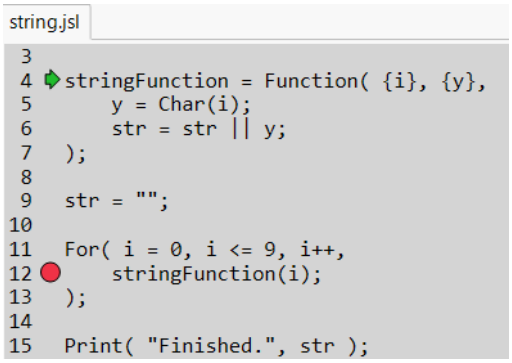
The following example shows how to set a breakpoint in a loop and watch variables change through each iteration of the loop.

1. Open the string.jsl sample script and click the **Debug Script** button.
2. Click in the margin for line 12 to add a breakpoint.

   You should have a breakpoint for the following expression inside the For() loop:

   ```
   stringFunction(i);
   ```

Figure 4.12 Set the Breakpoint



3. Click **Run**.

   The first two expressions are evaluated:

   – `stringFunction` is defined as a function.

   – `str` is defined as an empty string.

   Both variables and their types and values have been added to the Globals list. In addition, the `For()` loop has been evaluated up to the line with the breakpoint, shown in Figure 4.12.

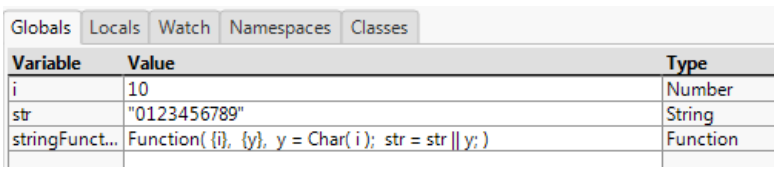
   – `i` has been assigned to 0.

   – `i` and its value and type have been added to the Globals list.

   – `i` has been determined to be less than or equal to 9.

   – `stringFunction()` has not yet been called.

Figure 4.13 View the Initial Global Variables



Tip: The initial value of `i` is 10 because the script was run before entering the Debugger; it's showing the value of `i` after running the script (10). Start a new instance of JMP, open the script without running it, and enter the Debugger immediately.

4. Click **Run** again.

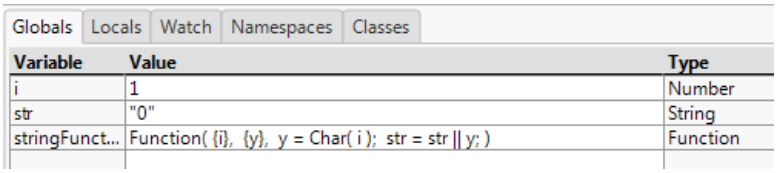
   The script runs until it hits the breakpoint. The results are shown in Figure 4.14.

   – `stringFunction()` is called, evaluated, and returns to the loop.

   – `i` is incremented and determined to be less than or equal to 9.

– In the Globals list, `i` is now 1 and `str` is now "0".

Figure 4.14 Global Variables at First Breakpoint



5. Click **Run** again.

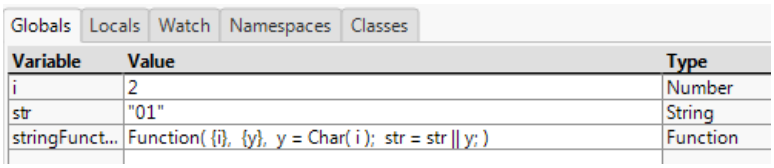
   The script runs until it hits a breakpoint. The results are shown in Figure 4.15.

   – `stringFunction()` is called, evaluated, and returns to the loop.

   – `i` is incremented and determined to be less than or equal to 9.

   In the Globals list, `i` is now 2, and `str` is now "01".

Figure 4.15 Global Variables at Second Breakpoint



You can continue to click **Run** and watch `i` and `str` change with each iteration of the loop. Or, click **Run without breakpoints** to complete running the script and exit the Debugger.

Stepping Into, Over, and Out

Step Into, **Step Over**, and **Step Out** offer flexibility when your script contains expressions, functions, or includes other JSL files.

1. Open the scriptDriver.jsl sample script and click the **Debug Script** button.
2. This script writes information to the log, so select the Log tab at the bottom of the Debugger to view the messages.
3. Click **Step Over**.

   The first line in the script is evaluated.

4. Click **Step Over** again.

   The current expression is evaluated, and the Debugger moves to the following line. In this case, the expression is a few lines long, and it assigns an expression to a variable.

5. Click **Step Over** again.

   This expression is several lines long, and assigns a function to a variable.

Line 30 calls the expression that was created earlier.

6. Click **Step Over**.

   The Debugger steps into the expression, running it line by line.

7. Continue clicking **Step Over** until the expression ends.

   The Debugger returns to the line following the expression call.

   Line 31 calls the function defined earlier.

8. Click **Step Over** to run the function without stepping into it. The Debugger runs the entire function, and returns to the line following the function call.

   Line 33 includes another script.

9. Click **Step Into**.

   The Debugger opens the script in another tab and waits.

10. Click **Step Over**.

    The next line in the included script is run.

11. Click **Step Out**.

    The Debugger runs the rest of the included script and returns to the line following the `Include()` function.

Watching Variables in Different Scopes and Namespaces

Tabs at the bottom of the Debugger window let you watch variables as they are created and changed. This example shows variables in several scopes and a namespace.

1. Open the scoping.jsl sample script and click the **Debug Script** button.
2. Click **Step Over**.

   The fourth line turns off `Names Default To Here`. If you run this script again in the same JMP session, this line resets the scoping so that the first variable that is created is in the global scope.

3. Click **Step Over**.

   A global variable named *x* is created. On the Globals tab, *x* has been added to the list, showing its value as 5 and its type as number.

4. Select the **Locals** tab, and then select **Global** from the list of scopes.

   The global variable *x* is also shown here.

5. Click **Step Over** twice.

   `Names Default To Here` is turned on, which places the rest of the script into a Here scope. Then a new variable *x* is created in that scope.

   Notice that the value of the global variable *x* has not changed.

6. Select **Here** from the list on the Locals tab.

   The local *x* is listed under **Here**, with its value and type.

7. Click **Step Over**.

   A `Local Here` scope is created. A second Here scope is shown in the Locals list.

8. Click **Step Over**.

   A new *x* variable is created in this Here scope. On the Locals tab, select each of the three scopes from the list (**Here**, **Here**, and **Global**) to see three different *x* variables.

9. Click **Step Over**.

   Look in the Debugger's log to see the output. Notice that *here:x* scopes to the local here, not the script window's here.

10. Click **Step Over**.

    After writing an empty line to the log, the script exits the Local Here scope. The second Here, along with its' *x* variable, has disappeared from the Locals list.

11. Click **Step Over**.

    A namespace called "test" is created, with another variable named *x*. Select the **Namespaces** tab to see it.

12. Click **Step Over** and look at the log.

13. Click **Step Over** to exit the Debugger.

Using the Debugger with Interactive Scripts

When your script creates interactive elements, the Debugger hands control back to the main instance of JMP so that you can interact with it. When you are finished, control returns to the Debugger.

1. Open the interactive.jsl sample script and click the **Debug Script** button.
2. Click **Step Over** twice.

   The `New Window` expression is evaluated, and a modal window waiting for input is created. You might need to move the Debugger window to see the new modal window.

3. Enter two numbers in the Assign X and Y window and click **OK**.

   Control is given back to the Debugger.

4. Click **Step Over** three times and look at the log in the Debugger.

   The log shows the two new numbers that you entered in the window.

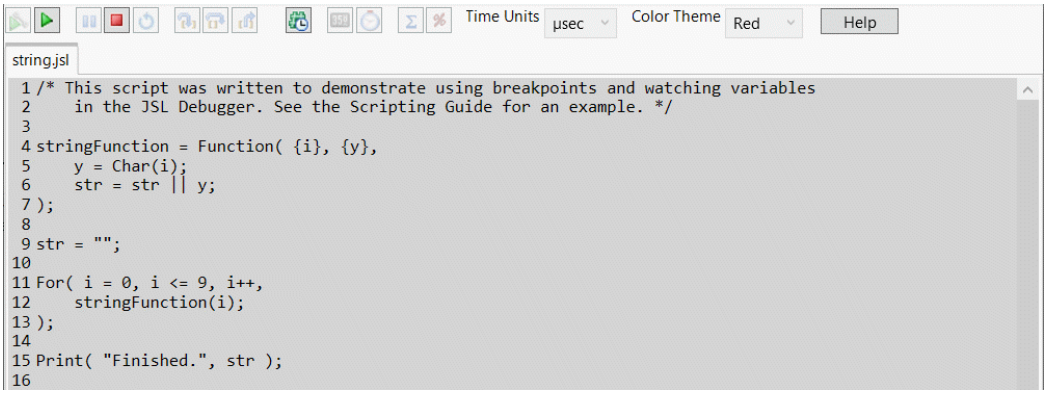
5. Click **Step Into** to exit the Debugger.

Using the JSL Profiler

Use the JSL Profiler to see how much time is spent executing particular lines or how many times a particular line is executed.

1. Open the string.jsl sample script.
2. Click the **Debug Script** button .
3. Click the **Profile JSL Script** button .

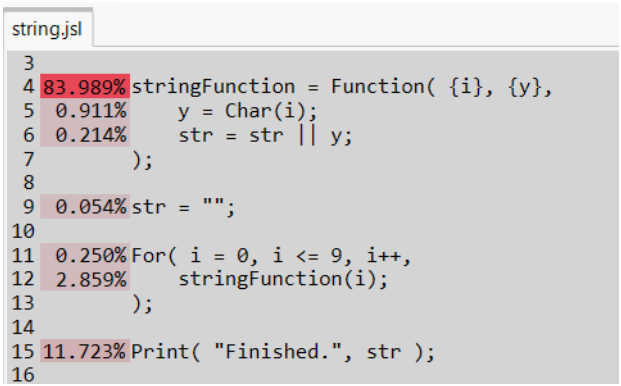
Figure 4.16 Initial JSL Profiler Window



4. Click the **Run** button to start profiling.

   The profiler collects information about the number of times a statement is executed and the time it takes to execute it. Time is cumulative and collected each time a JSL statement is executed.

Figure 4.17 Profiled Script Window



In the left margin, the selected statistics are displayed. Percent of time is displayed by default. Click the **Show Profile by Count** button to switch to percent of statement counts instead. The left margin is color-coded to allow for quick identification of problematic performance areas.

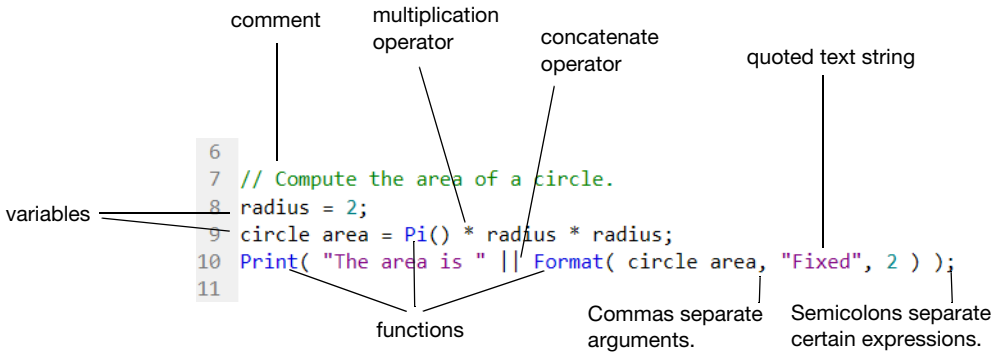
Chapter 5

JSL Building Blocks

Learning the Basics of JSL

Studying the syntax and basics of JSL is crucial, whether you are a beginning or an advanced user. Some concepts (such as loops and variables) are common among many scripting languages, but punctuation rules differ in JSL.

Figure 5.1 Example of a JSL Script



This chapter introduces you to the basic concepts of JSL, from syntax rules and file path conventions to conditions and namespaces.

Contents

JSL Syntax Rules . . . 100
Value Separators . . . 100
Numbers . . . 103
Names . . . 103
Comments . . . 104
Operators . . . 105
Boolean Values . . . 109
Global and Local Variables . . . 110
Local Namespaces . . . 110
Named Namespaces . . . 111
Show Symbols, Clear Symbols, and Delete Symbols . . . 111
Lock and Unlock Symbols . . . 112
Hide a Global Variable . . . 113
Rules for Name Resolution . . . 113
Variable Names Used as Arguments . . . 114
Rules for Resolving Names . . . 114
Scoping Operators . . . 116
Troubleshoot Variables and Column Names . . . 119
Troubleshoot Variables and Keywords . . . 120
Frequently Asked Questions about Name Resolution . . . 120
Alternatives for Gluing Expressions Together . . . 121
Iterate . . . 122
For . . . 122
While . . . 123
Summation . . . 125
Product . . . 125
Break and Continue . . . 126
Conditional Functions . . . 128
If . . . 128
Match . . . 130
Choose . . . 132
Interpolate . . . 133
Step . . . 135
Compare Incomplete or Mismatched Data . . . 135
Missing Values . . . 137
Inquiry Functions . . . 138
General Element Types . . . 139
Specific Element Types . . . 139
Object Attributes . . . 141

Host Information 141
Version Information 141

JSL Syntax Rules

All scripting and programming languages have their own syntax rules. JSL looks familiar if you have programmed in languages such as C and Java. However, rules for punctuation and spaces differ in JSL.

The following sections describe JSL syntax rules for the following basic components of the language.

Value Separators

Words in JSL are separated by parentheses, commas, semicolons, spaces, and various operators (such as +, –, and so on). This section describes the rules for using these separators and delimiters in JSL.

Commas

A comma separates items, such as items in a list, rows in a matrix, or arguments to a function.

```
my list = {1, 2, 3};
your list = List( 4, 5, 6) ;
my matrix = [3 2 1, 0 -1 -2];
If( Y < 20, X = Y );
Table Box( String Col Box( "Age", a ) );
```

Note: To glue a sequence of commands into a single argument inside a function, separate each sequence with a semicolon. See "Semicolons".

Parentheses

Parentheses have several purposes in JSL:

- Parentheses group operations in an expression. The following parentheses group the operations in the `If()` expression.

  ```
  y = 10;
  If(
     Y < 20, X = Y,
     X = 20
  );
  ```

- Parentheses delimit arguments to a function. In the following example, parentheses enclose the argument to the `Open()` function.

  ```
  Open( "$SAMPLE_DATA/Big Class.jmp" );
  ```

- Parentheses also mark the end of a function name, even when arguments are not needed. For example, the `Pi` function has no arguments. However, the parentheses are required so that JMP can identify *Pi* as a function.

  ```
  Pi();
  ```

Note: Be careful that parentheses match. Every `(` needs a `)`, or errors result.

The script editor can match fences (parentheses, brackets, and braces). Press Ctrl+] (Command+B on macOS) with your cursor in any part of a script. The editor searches for fences, highlighting the text between the first set of opening and closing fences that it finds. Repeat this process to highlight the next-higher fence. See "Match Parentheses, Brackets, and Braces" for an example.

Semicolons

Expressions separated by a semicolon are evaluated in succession, returning the result of the last expression. In the following code, 0 is assigned to the variable `i` and then 2 is assigned the variable `j`.

```
i = 0;
j = 2;
```

You can also use semicolons to join arguments that are separated by commas as shown in the following `If()` expression.

```
If( x < 5, y = 3; z++; );
```

The semicolon in other languages is used as an expression terminator character. In JSL, the semicolon is a signal to continue because more commands might follow. For more information about separating expressions with semicolons, see "Alternatives for Gluing Expressions Together".

Semicolons at the end of a script or at the end of a line of arguments are harmless, so you can also think of semicolons as terminating characters. Trailing semicolons are allowed at the end of a script stream and after a closing parenthesis or closing curly brace. In fact, terminating each complete JSL expression with a semicolon helps avoid errors when you copy and paste small scripts into a larger one.

The semicolon is equivalent to the `Glue()` function. See "Operators" for more information about semicolons and `Glue()`.

Double Quotes

Double quotes enclose text strings. Anything inside double quotes is taken literally, including spaces and upper- or lower-case; nothing is evaluated. If you have `"Pi() ^ 2"` (inside double quotes), it is just a sequence of characters, not a value close to ten.

You do have to be a careful with text strings. Extra spaces and punctuation do affect the output, because text strings inside double quotes are used literally, exactly as you enter them.

To have double quotes inside a quoted string, precede each quotation mark with the escape sequence \! (backslash-bang). For example, run the following script and look at the title of the window:

```
New Window( "\!"Hello\!" is a quoted string.",
  Text Box( Char( Repeat( "*", 70 ) ) )
);
```

Table 5.1 Escape Sequences for Quoted Strings

\!b	blank
\!t	tab
\!r	carriage return only
\!n	linefeed (newline) only
\!N	inserts line breaking characters appropriate for the host environment[a]
\!f	formfeed (page break)
\!0	null character **Note:** The null character is dangerous to use, because it is typically treated as the end of the string. Be sure to type the number zero, not the letter O.
\!\	backslash
\!"	double quotation mark

a. On macOS, this escape sequence is CR (carriage return character, hexadecimal '0D'). On Windows, this sequence is CR LF (carriage return followed by a linefeed, hexadecimal '0D0A').

Sometimes, long passages require a lot of escaped characters. In these cases, use the notation \[...]\ and everything between the brackets does not need or support escape sequences. Here is an example where \[...]\ is used inside a double-quoted string.

```
jslPhrase = "The JSL to do this is :\[
  a = "hello";
  b = a|| " world.";
  Show(b);
  ]\ and you use the Submit command to run it.";
```

Spaces

JSL allows whitespace characters inside names; spaces, tabs, returns, and blank lines inside or between JSL words are ignored. This is because most JSL words come from the user interface, and most of those commands have spaces in them. For example, the JSL expression for the Fit Model platform is `Fit Model()` or `FitModel()`.

Spaces inside an operator or between digits in a single number are not allowed. In these cases, the script generates errors. For example, you cannot put spaces between the two plus signs in `i++` (`i+ +`) or in numbers (`4 3` is not the same as `43`).

Note: Why does JSL allow whitespace characters inside names? For one reason, the names of commands and options match the equivalent commands in JMP menus and windows. Another reason is that data table column names often include spaces.

Numbers

Numbers can be written as integers, decimal numbers, dates, times, or date-time values. They can also be included in scientific notations with an `E` preceding the power of ten. For example, these are all numbers:

```
.   1   12   1.234  3E3  0.314159265E+1 1E-20 01JAN98
```

Note: A single period by itself is considered a missing numeric value (sometimes called *NaN* for "not a number").

For more information about dates, times, and date-time values, see "Date-Time Functions and Formats". See "Currency" for more information about combining numbers with currency symbols.

Names

A name is simply something to call an item. When you assign the numeric value 3 to a variable in the expression `a = 3`, `a` is a name.

Commands and functions have names, too. In the expression `Log( 4 )`, `Log` is the name of the logarithmic function.

Names have a few rules:

- Names *must* start with an alphabetic character or underscore and can continue with the following:
  - alphabetic characters (a-z A-Z)
  - numeric digits (0-9)

  - whitespace characters (spaces, tabs, line endings, and page endings)
  - mathematical symbols in Unicode (such as the Greek small alpha letter α)
  - a few punctuation marks or special characters (apostrophes (‘), percent signs (%), periods (.), backslashes (\), and underscores (_))
- When comparing names, JMP ignores whitespace characters (such as spaces, tabs, and line endings). Upper case and lower case characters are not distinguished. For example, the names `Forage` and `for age` are equivalent, despite the differences in white space and case.

You can still have a name that is any other sequence of characters. If the name does not follow the rules above, it needs to be quoted and `followed by "n"`. For example, to use a global variable with the name `taxable income(2011)`, you must use `"taxable income(2011)"n` every time the variable appears in a script:

```
"taxable income( 2011 )"n = 456000;
tax = .25;
Print( tax * "taxable income( 2011 )"n) ;
   114000
```

Note: Previously, the `Name()` parser directive wrapped the column name that contained illegal characters. `Name()` is deprecated.

For more information about how JMP interprets names, see “Rules for Name Resolution”.

Comments

Comments are notes in the code that are ignored by the JSL processor (or *parser*). You include comments to describe sections of the script. Comments are also convenient for removing portions of a script temporarily. For example, you can insert comment symbols around code that might be causing an error and then rerun the script.

Type the comment symbols around code that you want to comment. The following example shows code commented with `/* */` in the middle of a line. When the script is run, JMP considers both expressions to be identical.

```
tax /*percentage*/ = .25;
tax = .25;
```

Table 5.2 describes the comment symbols.

Table 5.2 Comment Symbols

Symbol	Syntax	Explanation
`//`	`// comment`	Begins a comment. The comment does not have to be at the beginning of a line, but everything up to the end of the line is a comment.
`/* */`	`/* comment */`	Begins and ends a comment. This comment can appear in the middle of a line. Script text before and after the comment is parsed normally.
`//!`	`//!`	Add `//!` to the first line of the script, and the script runs automatically when opened in JMP. (In other words, the script editor does not open.)

Operators

An operator is a one- or two-character symbol used to perform common arithmetic actions, compare values, construct lists, subscript into a data element, send messages, concatenate lists, scope names, glue expressions, and end expressions.

Arithmetic operators come in several varieties:

- *infix* (with arguments on either side, such as + in `3 + 4`, or = in `a = 7`)
- *prefix* (with one argument on its right side, such as `!a` for logical negation)
- or *postfix* (with one argument on its left side, such as `a++` for incrementing `a`

To make writing expressions easier, JSL uses certain special character operators that are alternatives to using functions. These operators have the same meaning as if the phrase had been written as a function. For example, the following two expressions are equivalent.

```
Net Income After Taxes = Net Income - Taxes;
Assign( Net Income After Taxes, Subtract( Net Income, Taxes ) );
```

The assignment operation can be written either as the `Assign()` function or as an infix operator =. Similarly, subtraction can be done with the `Subtract()` function or the infix minus sign; they are equivalent inside JMP.

Note: Usually white space is ignored in JSL, so that "netincomeaftertaxes" and "net income after taxes" are the same thing. There is one exception: the two-character operators must *not* have a space between characters. Always enter the following operators without spaces in between:

```
||, |/, <=, >=, !=, ==, +=, -=, *=, /=,  ++, --, <<, ::, :*, :/, /*, */
```

Another common operator is the semicolon (`;`). You use the semicolon to:

- Separate yet join one expression to another in a programming sequence. The semicolon returns the result of its last argument, so `a;b` is the same as `Glue(a,b)`.
- End an expression. Though the semicolon is permitted at the end of an expression, it is not the expression terminator used in other languages.

An expression can contain more than one operator. In these instances, the operators are grouped in order of precedence with decreasing priority. For example, * takes precedence over +:

```
a + b * c
```

So `b * c` is evaluated first, and then the result is added to `a`.

+ takes precedence over -:

```
a + b * c - d
```

So `b * c` is evaluated, and then the result is added to `a`. `d` is then subtracted from the result of `a + b * c`.

Table 5.3 shows operators shaded in order of precedence and each operator's function equivalent.

Table 5.3 Operators and Their Function Equivalents in Order of Precedence

Operator		Function Syntax	Explanation
`{ }`	`List`	`{a,b}` `List(a,b)`	Construct a list.
`[ ]`	`Subscript`	`a[b,c]` `Subscript(a,b,c)`	Subscripts identify specific elements within a data element *a*, where *a* could be a list, a matrix, a data column, a platform object, a display box, and so on.

Table 5.3 Operators and Their Function Equivalents in Order of Precedence *(Continued)*

Operator		Function Syntax	Explanation
++	Post Increment	*a*++ Post Increment(*a*)	Adds one (1) to *a*, in place. JMP does not have a pre-increment operator. Instead, use the Add To() operator, which is (+=).
--	Post Decrement	*a*-- Post Decrement(*a*)	Subtracts one (1) from *a*, in place. JMP does not have a pre-decrement operator. Instead, use the Subtract To() operator, which is (-=).
^	Power	*a*^*b* Power(*a*,*b*) Power(*x*)	Raise *a* to exponent power *b*. With only one argument, 2 is assumed as the power, so Power(*x*) computes x^2.
-	Minus	-*a* Minus(*a*)	Reverses sign of *a*.
!	Not	!*a* Not(*a*)	Logical Not. Maps nonzero (or true) values to 0 (which means false). Maps 0 (or false) values to 1 (which means true).
*	Multiply	*a**b* Multiply(*a*,*b*)	Multiplies *a* by *b*.
:*	EMult	*a*:**b* EMult(*a*,*b*)	Elementwise multiplication for matrices *a* and *b*. (Each element in matrix *a* is multiplied by each element in matrix *b*.)
/	Divide	*a*/*b* Divide(*a*,*b*) Divide(*x*)	Divide(*a*, *b*) divides *a* by *b*. Divide(*x*) interprets the argument as a denominator and implies 1 as the numerator, yielding the reciprocal 1/*x*.
:/	EDiv	*a*:/*b* EDiv(*a*,*b*)	Elementwise division for matrices *a* and *b*. (Each element in matrix *a* is divided by each element in matrix *b*.)

Table 5.3 Operators and Their Function Equivalents in Order of Precedence *(Continued)*

Operator		Function Syntax	Explanation
+	Add	*a*+*b* Add(*a*,*b*)	Adds *a* and *b*.
–	Subtract	*a*–*b* Subtract(*a*,*b*)	Subtracts *b* from *a*.
||	Concat	*a*||*b* Concat(*a*,*b*)	Joins two or more strings; two or more lists; and horizontally concatenates matrices. See "Concatenate Lists" or the *JSL Syntax Reference*.
|/	VConcat	*matrix1*|/*matrix2* VConcat(*matrix1*, *matrix2*)	Vertically concatenate matrices. (Use || or Concat() to horizontally concatenate matrices.)
::	Index	*a*::*b* Index(*a*,*b*)	For matrices, generates the integers from *a* to *b*. (Colons are also used as prefix operators for scoping, where :a means data table column a, and ::a means JSL global variable *a*. See "Scoping Operators".)
<<	Send	object << message Send(*object*, *message*)	Send *message* to *object*.
==	Equal	*a*==*b* Equal(*a*,*b*)...	Boolean values for comparisons. They all return 1 if true, 0 if false. Missing values in either *a* or *b* causes a return value of missing, which evaluates as neither true nor false. See "Missing Values", for treatment of missing values.
!=	Not Equal	*a*!=*b* Not Equal(*a*,*b*)...	
<	Less	*a*<*b* Less(*a*,*b*)...	
<=	Less or Equal	*a*<=*b* Less or Equal(*a*,*b*)	
>	Greater	*a*>*b* Greater(*a*,*b*)	
>=	Greater or Equal	*a*>=*b* Greater or Equal(*a*,*b*)	

Table 5.3 Operators and Their Function Equivalents in Order of Precedence *(Continued)*

Operator		Function Syntax	Explanation
<=, <	Less Equal Less	*a*<=*b*<*c* Less Equal Less(*a*,*b*,*c*)	Range check. Return 1 if true, 0 if false. Missing values in either *a* or *b* propagate missing values.
<, <=	Less Less Equal	*a*<*b*<=*c* Less Less Equal(*a*,*b*,*c*)	
&	And	*a*&*b* And(*a*,*b*)	Logical And. Returns true if both are true. If the value on the left is false, the value on the right is not evaluated. See "Missing Values", for treatment of missing values.
|	Or	*a*|*b* Or(*a*,*b*)	Logical Or. Returns true if either or both are true. See "Missing Values", for treatment of missing values.
=	Assign	*a*=*b* Assign(*a*,*b*)	Put the value of *b* into *a*. Replaces the current value of *a*.
+=	Add To	*a*+=*b* AddTo(*a*,*b*)	Add the value of *b* into *a*.
-=	Subtract To	*a*-=*b* SubtractTo(*a*,*b*)	Subtract *b* from *a*, and put back into *a*.
*=	Multiply To	*a**=*b* MultiplyTo(*a*,*b*)	Multiply *b* with *a*, and put back into *a*.
/=	Divide To	*a*/=*b* DivideTo(*a*,*b*)	Divide *b* into *a*, and put back into *a*.
;	Glue	*a*;*b* Glue(*expr*, *expr*, ...)	First do *a*, and then do *b*.

Boolean Values

A Boolean value enables or disables an option. To enable the option, use `1`, `true`, or `yes` as the argument.

```
obj << Unequal Variances( 1 );
obj << Unequal Variances( "true" );
obj << Unequal Variances( "yes" );
```

Note: Quoting the value prevents it from being evaluated as a variable.

Omitting the argument also enables the option. This is the default for all but row state messages.

```
obj << Unequal Variances;
obj << Unequal Variances();
```

To disable the option, use `0`, `false`, or `no` as the argument.

To toggle an option (turn off an enabled option or turn on a disabled option), use `"toggle".`

Notes:

- The quoted `"present"`, `"absent"`, `"on"`, `"off"`, `"switch"`, and `"flip"` are supported for backwards compatibility.
- For row state messages, an empty argument always toggles the option.

Global and Local Variables

Variables are names that hold values, which you reference later in scripts. There are two types of variables:

- *Global variables* are shared among all scripts that you run in a JMP session.
- *Local variables* apply only to the script context in which you define them. They can also be local to only a piece of a script, as with variables local to a particular function.

To limit the scope of variables, you can define them in a *namespace*, which is a collection of variables, functions, and other unique names. JMP has a single global variable namespace that all scripts use by default. When you use a name plainly, without a qualifying syntax, the name is an *unscoped* variable and therefore in the global namespace.

```
x = 1;
```

Local Namespaces

Putting variables in the global namespace can cause conflicts. When two scripts have variables with the same names, the value of the variable in the last script that you run last overwrites the variable's value in the first script.

To prevent this problem, we recommend that you begin each script with the following line:

```
Names Default To Here( 1 );
```

The `Names Default To Here( 1 );` function makes all unscoped variables in the script local to that script and does not affect the global variable namespace. See "Advanced Scoping and Namespaces".

Notes:

- The `Names Default to Here` setting is true by default for custom menus and toolbar buttons. A script that runs when you select a custom menu item or click a custom toolbar button does not affect global variables.
- The `Names Default to Here` setting is false by default for table scripts. If a table script specifies `Names Default to Here(1)`, this setting lasts only as long as the table script runs. After the script finishes running, the setting is reset to whatever it was before the script ran.

Named Namespaces

You can also create a variable in a specific namespace. In the following example, the `x` variable is created in the `aa` namespace:

```
aa:x = 1;
```

Preceding local variables with the `Local()` function is another option. Both `a` and `b` are local variables in the following expression:

```
Local( {a = 1, b}, ... );
```

Scoping operators also distinguish a global variable from a local variable. See "Rules for Name Resolution".

The following sections describe functions that help you manage variables.

Show Symbols, Clear Symbols, and Delete Symbols

The `Show Symbols()` function lists all variables and namespaces that are defined both globally and in the local script, along with their current values. Here is an example of `Show Symbols()` messages that are shown in the log:

```
Show Symbols();
   // Here
   a = 5;
   b = 6;
   // 2 Here

   // Global
   c = 10;
   // 1 Global
```

Tip: The JSL debugger also shows you the values of variables and namespaces. See "Debug or Profile Scripts".

The `Clear Symbols()` function erases the values set for variables that are defined both globally and in the local script. For example, after you clear and then show symbols, the variables are empty.

```
Clear Symbols();
Show Symbols();
   // Here
   a = Empty;
   b = Empty;
   // 2 Here

   // Global
   c = Empty;
   // 1 Global
```

Note: The older `Show Globals()` and `Clear Globals()` functions are aliases of the newer `Show Symbols()` and `Clear Symbols()` functions.

To remove *all* global variables and namespaces, use the function `Delete Symbols()`. After the last `Show Symbols()` in the following script is run, nothing shows up in the log. All variables have been completely removed from memory.

```
Delete Symbols();
Show Symbols();
```

To list variables in all namespaces, use `Show Namespaces()`. To delete only a specific namespace, use `ns << Delete. Clear Symbols()` and `Delete Symbols()` do not clear or delete variables in each namespace, although they do clear and delete variables that contain references to namespaces. See "Rules for Name Resolution" for more information about unscoped variables.

Note: `Clear Symbols()` and `Delete Symbols()` break all scripts that are currently in use. These functions can be very useful in a programming and debugging environment, but do not include them in any script that you plan to distribute. If you include `Names Default To Here(1)` in your scripts, clearing and deleting global symbols is unnecessary.

Lock and Unlock Symbols

If you want to lock a variable to prevent it from being changed, use the `Lock Symbols()` function. (`Lock Globals()` is an alias.)

```
Lock Symbols( name1, name2, ... );
```

To release the lock and enable the global to be changed, use the `Unlock Symbols()` function. (`Unlock Globals()` is an alias.)

```
Unlock Symbols( name1, name2, ... );
```

The primary use of these two commands is to prevent inadvertent changes to variables. For example, locking a variable prevents `Clear Symbols`() from clearing a variable that is being used by another script.

Note: You cannot use `Lock Symbols()` to lock a namespace. Instead, use `ns <<Lock Namespace.`

Hide a Global Variable

To hide (or protect) a global variable, put two underscores (__) before the name. Protecting a global variable means that it will be hidden and can not be examined or shown. However, the behavior is different depending upon the context.

This example will not print the variable because it is protected:

```
Show(::__xyz);
```

This example returns a NULL value for the protected variable:

```
::a = Name Expr( ::__xyz );
```

This example does return a value for the protected variable:

```
Show( Eval( ::__xyz ) );
```

Rules for Name Resolution

In JMP, you identify the following types of objects by a name:

- Columns and table variables in a data table
- Global variables, which hold values for the remainder of a JMP session
- Scriptable object types
- Arguments and local variables inside formulas

Most of the time, you can just use an object's name directly to refer to the object. Consider the following example:

```
ratio = height / weight;
```

Depending on the complexity of your script, it might be obvious that `ratio` is a variable and `height` and `weight` are data table column names. But what if the meanings are ambiguous? A script might use `ratio` as a global variable and as column names.

Variable Names Used as Arguments

Message arguments are evaluated as a rule. In previous versions of JMP, some messages interpreted a variable name as a string value. For example, the following script results in a linear scale in the current version of JMP and a log scale in JMP 11.

```
Log = "Linear";
axis box << Scale( Log );
```

The syntax `Scale( "Log" )` is preferred for setting a string literal. To catch ambiguous instances, set the "Allow Unquoted Strings in JSL" General preference to "Yes (with a warning)" or "No".

Rules for Resolving Names

JMP interprets object names using *name resolution*. The following rules are applied sequentially to unscoped names:

1. If the variable is followed by a pair of parentheses `( )`, look it up as a function.
2. If the variable is prefixed by `:` scope operator or an explicit data table reference, look it up as a data table column or table variable.
3. If the variable is prefixed by `::` scope operator, look it up as a global variable.
4. If the variable is an explicit scope reference (such as `group:vowel`), look it up in the user-defined `group` namespace.
5. If the variable is in a `Local` or `Parameter` function, look it up as a local variable. If it is nested, repeat until a function call boundary is found.
6. If the variable is in a user-defined function or method, look it up as a function or method argument or local variable.
7. Look the variable up in the current scope and its parent scope. Repeat until the `Here` scope is encountered.
8. Look the variable up as a variable in the `Here` scope.
9. Look the variable up as a global variable.
10. If `Names Default to Here(1)` is at the top of the script, stop looking. The scope is local.
11. Look the variable up as a data table column or table variable. Columns are preferred over table variables.

12. Look the variable up as an operator or a platform launch name (for example, Distribution, Bivariate, Chart, and so on).
13. When the name cannot be found:
    - If the name is used in a reference, print and error to the log.
    - If the name is used as the target of an assignment (as an L-value), test the following:

      If the variable is preceded by `::` scope operator, create and use a global variable.

      If the variable is an explicit scope reference, create and use the variable in the specified namespace or scope.

      If `Names Default to Here(0)` is at the top of the script, create a global variable.

      If `Names Default to Here(1)` is at the top of the script, create a `Here` namespace variable.

Exceptions

- Some names are variables that refer to an object such as a data table, data column, or platform; they are not used for getting or setting a value. These names are passed through (or interpreted literally) rather than resolved.
- For function definitions, column formulas, and Nonlinear platform formulas, the scope is the same for each row in a column.
- If a name is a direct reference to a column in a data table that has been closed, the name is resolved again to that column when the table is reopened.

  ```
  dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
  Show( :weight << Get As Matrix ); // weight  resolves to a column name
  Close( dt, NoSave );
  Show( :weight << Get As Matrix ); // weight  cannot be resolved
  dt = Open( "$SAMPLE_DATA/Big Class.jmp" ); // reopen data table
  Show( :weight << Get As Matrix ); // weight  resolves to a column name
  ```

  However, the following example does not resolve the variable to the second instance of the data table:

  ```
  dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
  col = Column( dt, 5); // col is Column( "weight" );
  Close( dt, NoSave );
  dt = Open( "$SAMPLE_DATA/Big Class.jmp" ); // reopen data table
  Show( col << Get As Matrix ); // the reference to the first data table no
    longer exists
  ```

The following sections describe how JMP resolves the names of data table columns. For more information about name resolution, see “Advanced Scoping and Namespaces”.

Scoping Operators

Using *scoping operators* is an easy way to help JMP resolve ambiguous names (for example, when a name refers to both a variable and a column name).

In the following example, the prefix double-colon operator (`::`) identifies `z` as a global variable. The single-colon prefix operator (`:`) identifies `x` and `y` as column names.

```
::z = :x + :y;
```

Tip: The `Names Default to Here(1)` function also affects name resolution. See "Names Default To Here and Global Variables".

Two JSL functions are interchangeable with scoping operators. Table 5.4 describes the functions and syntax.

Table 5.4 Scoping Operators

Operator and Equivalent Function	Function Syntax	Explanation

`:` `As Column`	`:name`
	`dt:name`
	`As Column(`*`dt`*`,`
	`name``)`
		Forces *`name`* to be evaluated as a data table column. The optional data table reference argument, *`dt`*, sets the current data table. See "Scoped Column Names" for examples.

`::` `As Global`	`::name` `As Global(name)`	Forces *`name`* to be evaluated as a global variable. **Note:** The double-colon is also used as an infix operator to represent ranges.

Scoped Column Names

Scoping column names is the simplest way to prevent conflicts with variable names. Use scoping operators to force names in a script to refer to columns.

1. The prefix colon (`:`) means that the name refers to a table column or table variable only, never a global variable. The prefix colon refers to the current data table context.

   ```
   :age;
   ```

2. The infix colon (`:`) operator extends this notion by using a data table reference to specify which data table has the column. This is particularly important when multiple data tables are referenced in a script.

   In the following example, the `dt` variable sets the data table reference to `Big Class.jmp`. The infix colon separates the data table reference and `age` column.

```
dt = Data Table( "Big Class.jmp" );
dt:age // The colon is an infix operator.
```

`As Column()` achieves the same results:

```
dt = Data Table( "Big Class.jmp" );
As Column( dt, age );
```

Therefore, the following expressions are equivalent when only Big Class.jmp is open:

```
:age;
As Column( dt, age );
dt:age;
```

The `Column` function can also identify a column. For Big Class.jmp, the following expressions all refer to age, the second column in the table:

```
Column( "age" );
Column( 2 );
Column( dt, 2 );
Column( dt, "age" );
```

Prevent Column Name and Variable Name Conflicts

To avoid conflicts, use unique column and variable names or scope the names.

- When a global variable and a column have the same name, the global variable name takes precedence. In this situation, you must scope the column name.

  ```
  ::age = [];
  age = :age << Get As Matrix;
  ```

- To avoid ambiguity between the global variable and column name, scope both variables.

  ```
  ::age = :age << Get As Matrix;
  ```

- If more than one data table might be open, assign data table references to variables. Scope your columns to the appropriate table.

  ```
  dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
  dt2 = Open( "$SAMPLE_DATA/Students.jmp" );
  ::age = dt1:age << Get As Matrix;
  ::height = dt2:height << Get As Matrix;
  ```

Note that JMP evaluates column formulas through each consecutive cell of the column, so scoping the column name is usually unnecessary. However, if a variable assigned in a formula has the same name as a column, you must scope the column name. See "Scoped Names".

Tip: The `Names Default to Here(1)` function also affects name resolution. See "Names Default To Here and Global Variables".

Unscoped Column Names

Sometimes an unscoped name gets or sets a value. JMP resolves it as a column in a data table (rather than a global variable) under these circumstances:

- if no global variable, local variable, or an argument using that name already exists,
- and the data table in context has a column of that name,
- and
  - either the current row is set to a positive value
  - or the name is subscripted (for example, the subscript `[1]` in `weight[1]` selects the first value in the weight column).

Exception

In column formulas and Nonlinear formulas, column names take precedence over global variables.

Set the Current Data Table Row

By default, the current row is 0, an illegal row number. So the following expression assigns a missing value to the *ratio* global variable:

```
ratio = height / weight;
```

Specify the row number with the `Row()` function. In the following example, the row is set to 3. The height in that row is divided by the weight, and the result is assigned to the *ratio* global variable.

```
Row() = 3;
ratio = height / weight;
```

Another possibility is to use subscripts to specify the row number. The following expression divides the height in row 3 by the weight in row 4.

```
ratio = height[3] / weight[4];
```

Specifying the row number is unnecessary when the script iterates a row at a time down the whole column. The following example creates the ratio column. For each row, the height is divided by the weight.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "ratio" );
For Each Row( :ratio = height / weight );
```

JMP evaluates formulas and calculates pre-evaluated statistics iteratively down a column. In these instances, identifying the row number is also unnecessary. (Pre-evaluated statistics are single numbers computed from the data table, as discussed in "Pre-Evaluated Statistics".)

Name Resolution in Automatically Run Scripts

In an `Open()` function, you can specify that the script run automatically when it's opened.

```
Open( "$MY_SCRIPTS/A.jsl", Run JSL( 1 ) );
Open( "$MY_SCRIPTS/B.jsl", Run JSL( 1 ) );
```

You can also type `//!` on the first line of the script to run the script automatically.

In these instances, the only shared names between `A.jsl` and `B.jsl` are globals and namespaces (not here and not local).

To apply the names in `A.jsl` to `B.jsl`, include `B.jsl` in `A.jsl` instead of opening and running both scripts.

```
Include( "$MY_SCRIPTS/B.jsl" );
```

Troubleshoot Variables and Column Names

When you reference a column name using `As Name()`, and `Names Default To Here( 1 )` is set, JMP returns a variable reference. That reference is then processed using the standard reference rules.

In the following example, there is no height variable in the `Here:` scope, so JMP returns an error.

```
Names Default To Here( 1 );
Open( "$SAMPLE_DATA/Big Class.jmp" );
As Name( "height" )[3]; // converts height to a variable reference
  As Name( "height" )[/*###*/3];
```

The `/*###*/` characters in the log indicate the location of an error in the script execution.

To prevent this problem, use one of the following methods:

- Use `As Column()` instead of `As Name()`:

  ```
  Names Default To Here( 1 );
  Open( "$SAMPLE_DATA/Big Class.jmp" );
  As Column( "height" )[3]; // converts height to a data column reference
  ```

- Explicitly scope height with `As Name():`

  ```
  Names Default To Here( 1 );
  dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
  dt:( As Name( "height" ) )[3]; // scopes height as a data column reference
  ```

These scripts return 55, the value of height in the third row of Big Class.jmp.

Troubleshoot Variables and Keywords

Name resolution errors can also occur when a variable and unquoted keyword have the same name. For example, one argument for `<<Preselect Role()` is "Y". Quote this argument if your script also uses Y as a variable.

Frequently Asked Questions about Name Resolution

Should you always scope?

Yes. When in doubt, scope. Scoping is especially important in scripts that might be used by many people on a variety of data tables; you will not necessarily know whether a name is used in two contexts (such as for both a global variable and column name).

If you are writing such scripts, consider using explicit scoping and namespaces. See "Advanced Scoping and Namespaces".

Prefix scope operators do not take run-time overhead after the first resolution. Infix scope operators, which follow data table references, always take run-time overhead.

What is the difference between a column reference and a column referred to by name? If I have a column reference in a global variable, how do I assign a value to a cell in the column?

With a column reference, you can send messages to change specific characteristics of the column or to access its values (for example, coloring cells or setting a formula).

When a column has been assigned to a global variable, assign a value to a cell in the column using a subscript. Suppose that the name of the column height has been assigned to the x variable:

```
x = Column( "height" );
```

Assign a value to the third row in the height column:

```
x[3] = 64 // sets the third row of height to 64
```

Note: The current row in a JSL script is *not* determined by selecting rows or positioning your cursor in the row. The current row is defined to be zero (no row) by default. You can set a current row with `Row()` (for example, `Row() = 3`). Please note that such a setting only lasts for the duration of that script. Then, `Row()` reverts to its default value, zero. This behavior means that submitting a script all at once can produce different results than submitting a script a few lines at a time.

Another way to establish a current row for a script is to enclose it in `For Each Row()`. This method executes the script once for each row of the current data table. For an example, see "If". See "Data Tables" for more information about working with data tables.

Which Has Precedence When Scoping, ":" or "[ ]"?

Scoping occurs before subscripting. This means that these two lines of code are equal:

```
dataTable:colName[i]
(dataTable:colName)[i]
```

Alternatives for Gluing Expressions Together

You can separate expressions with a semicolon, either on the same line or on different lines. JMP then evaluates each expressions in succession, returning the result of the last one. Here is an expression that first sets `a` to 2 and then sets `b` to 3:

```
a = 2;
b = 3;
```

The semicolon joins the two expressions and returns the value of the last one. So if `x = (a = 2; b = 3)`, the value of `x` is 3.

The `Glue()` function returns the result of the last expression. This function is equivalent to using semicolons. The following expressions both return 3:

```
Glue( a = 2, b = 3 );
a = 2; b = 3;
```

The `First()` function also evaluates each argument sequentially but returns the result of the first expression. The following expression returns 2:

```
First( a = 2, b = 3);
```

Example

What does `First()` do in the following script?

```
x = 1000;
```

```
First( x, x = 2000 );
```

The `First()` function returns the value of `x` (1000). 2000 is then assigned to `x`.

Iterate

JSL provides the `For()`, `While()`, `Summation()`, and `Product()` functions to repeat (or *iterate*) actions according to the conditions that you specify.

Note: A similar function called `For Each Row()` is for iterating actions over rows of a data table. See "If" for an example. "Additional Ways to Access Data Values" also describes iterating through table rows.

- "For"
- "While"
- "Summation"
- "Product"
- "Break and Continue"

For

The `For()` function expects four arguments separated by commas. The first three arguments are rules for how many times to repeat the loop, and the fourth is what to do each time the loop is executed.

Here is the basic syntax for `For()`:

```
For( initialization, while, iteration, body );
```

For example, the following script sums the numbers from 0 to 20:

```
s = 0;
For( i = 0, i <= 20, i++, s += i );
```

The script works like this:

`s = 0;`	Sets the `s` variable to 0. This variable holds the sum.
`For(`	Begins the `For()` loop.
`i = 0,`	Sets the initialization variable (`i`) to 0. This expression is performed only once.

`i <= 20,`	Each time the loop begins, compares `i` to 20. As long as `i` is less than or equal to 20, continue evaluating the loop. If `i` is greater than 20, immediately break out of the loop.
`i++,`	At the end of the loop, increments `i` by 1. Note that this step is done after the body of the loop (next line) is evaluated.
`s += i`	Evaluates the body of the loop. Adds the value of `i` to `s`. After the body is finished, *`i`* is incremented (previous line).
`);`	Ends the loop.

Infinite Loops

`For` loops that always evaluate as true create an infinite loop, which never stops. To stop the script, press Esc on Windows (or Command+period on macOS). You can also select **Edit > Stop Script**. On macOS, **Edit > Stop Script** is available only when the script is running.

Comparing For Loops in JSL to C and C++

The JSL `For()` loop works just like it does in the C (and C++) programming language, although the punctuation is different.

Tip: If you know C, watch out for the difference between JSL and C in the use of semicolons and commas. In JSL, `For()` is a function where commas separate arguments and semicolons join expressions. In C, `for` is a special clause where semicolons separate arguments and commas join them.

While

A related function is `While()`, which repeatedly tests the condition and evaluates its body script as long as the condition is true. The syntax is:

```
While( condition, body );
```

For example, here are two different programs that use a `While()` loop to find the least power of 2 that is greater than or equal to `x` (287). The result of both programs is 512.

```
x = 287;

// loop 1:
y = 1;
While( y < x, y *= 2 );
Show( y );
```

```
// loop 2:
k = 0;
While( 2 ^ k < x, k++ );
Show( 2 ^ k );
```

The scripts work like this:

`x = 287;`	Sets `x` to 287.
`// loop 1`	
`y = 1;`	Sets `y` to 1.
`While(`	Begins the `While()` loop.
`y < x,`	As long as `y` is less than `x`, continues evaluating the loop.
`y *= 2`	Multiplies 1 by 2 and then assigns the result to `y`. The loop then repeats while `y` is less than 287.
`);`	Ends the loop.
`Show(y);`	Shows the value of `y` (512).
`// loop 2`	
`k = 0;`	Sets `k` to 0.
`While(`	Begins the `While()` loop.
`2 ^ k < x,`	Raises 2 to the exponent power of `k` and continues evaluating the loop as long as the result is less than 287.
`k++`	Increments `k` to 1. The loop then repeats while `2 ^ k` is less than 287.
`);`	Ends the loop.
`Show(2 ^ k);`	Shows the value of `2 ^ k` (512).

As with `For()` loops, `While()` loops that always evaluate as true create an infinite loop, which never stops. To stop the script, press Esc on Windows (or Command-period on macOS). You can also select **Edit > Stop Script**. On macOS, **Edit > Stop Script** is available only when the script is running.

Summation

The Summation() function adds the body results over all i values. The syntax is:

```
Summation( initialization, limitvalue, body );
```

For example:

```
s = Summation( i = 1, 10, i );
```

returns 55, the result of 1+2+3+4+5+6+7+8+9+10.

The script works like this:

s =	Sets the s variable to the value of the function.
Summation(	Begins the Summation() loop.
i = 1,	Sets i to 1.
10,	Sets the limit of i to 10.
i	All values of i from 1 to 10 are added together, resulting in 55.
);	Ends the loop.

This behavior is similar to Σ in the Formula Editor. The following expression:

```
Summation( i = 1, N Row(), x ^ 2 );
```

is equivalent to the following formula in the Formula Editor:

$$\sum_{i=1}^{\text{NRow}} x^2$$

Product

The Product() function is similar to Summation() except that it multiplies the body results rather than adding them. The syntax is the same as for Summation(). For example:

```
p = Product( i = 1, 5 , i );
```

returns 120, the result of 1*2*3*4*5.

In this example, the initial value of i is 1, the upper limit is 5, then all integer values of i up to 5 are multiplied.

Here is the equivalent in the Formula Editor:

$$\prod_{i=1}^{5} i$$

Break and Continue

The `Break()` and `Continue()` functions give you more control over looping. `Break()` immediately stops the loop and proceeds to the next expression that follows the loop. `Continue()` is a gentler form of `Break()`. It immediately stops the current iteration of the loop and continues with the next iteration of the loop.

Break

`Break()` is typically used inside a conditional expression. For example:

```
For( i = 1, i <= 5, i++,
   If( i == 3, Break() );
   Print( "i=" || Char( i ) );
);
```

results in:

```
"i=1"
"i=2"
```

The script works like this:

`For(`	Begins the `For()` loop.
`i = 1,`	Sets `i` to 1.
`i <= 5,`	As long as `i` is less than or equal to 5, continues evaluating the loop.
`i++,`	Increments `i` by 1. Note that this step is done after the `If` loop is evaluated.
`If(`	Begins the `If()` loop.
`i == 3, Break()`	If `i` is equal to 3, breaks the loop.
`);`	Ends the loop.
`Print(`	When `i` is not equal to 3, opens the `Print()` loop.
`"i="`	Prints the string `"i="` to the log.
`||`	Places `"i="` on the same line as the value that follows.

`Char(i));`	Prints the value of `i` to the log. The `For()` loop then repeats until the value of i is less than or equal to 5, breaking when i equals 3, and breaking and printing only when i is less than 3.
`);`	Ends the loop.

Note that when the `If()` and `Break()` expressions follow `Print()`, the script prints the values of `i` from 1 to 3, because the loop breaks after "i=3" is printed.

```
"i=1"
"i=2"
"i=3"
```

Continue

As with `Break()`, `Continue()` is typically used inside a conditional expression. For example:

```
For( i = 1, i <= 5, i++,
   If( i < 3, Continue() );
   Print( "i=" || Char( i ) );
);
```

results in:

```
"i=3"
"i=4"
"i=5"
```

The script works like this:

`For(`	Begins the `For()` loop.
`i = 1,`	Sets `i` to 1.
`i <= 5,`	Evaluates 1 as less than or equal to 5.
`i++,`	Increments `i` by 1. Note that this step is done after the `If` loop is evaluated.
`If(`	Begins the `If()` loop.
`i < 3, Continue()`	Evaluates `i` as 1 and continues as long as `i` is less than 3.
`);`	Ends the `If()` loop.

`Print(`	When `i` is no longer less than 3, opens the `Print()` loop.
`"i="`	Prints the string `"i="` to the log.
`||`	Places `"i="` on the same line as the value that follows.
`Char(i));`	Prints the value of `i` to the log. The `For()` loop then repeats until the value of `i` is less than or equal to 5, continuing and printing only when i is equal to or greater than 3.
`);`	Ends the loop.

Conditional Functions

JSL provides five functions to evaluate an expression conditionally:

- "If"
- "Match"
- "Choose"
- "Interpolate"
- "Step"

If

The `If()` function evaluates the first result expression when its condition evaluates as true (a nonzero or nonmissing value). Otherwise, it evaluates the second result expression.

The syntax is:

```
If ( condition, result1, result2 );
```

For example, the following script returns "Young" when the age is less than 12. Otherwise, the script returns "Young at Heart".

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "age group", "Character" );
For Each Row(
   :age group = If( :age <= 12,
      "Young",
      "Young at Heart"
   )
);
```

You can also string together multiple conditions and results. The syntax is:

```
If( condition1, result1,
  condition2, result2,
  ...,
  resultElse );
```

In the preceding example, if condition1 is not true, the function continues evaluating until it finds a true condition. Then that condition's result is returned.

The last result is returned when all conditions are false. And when a value is missing, the missing value is returned. For these reasons, it's very important to include a default result at the end of the expression. Consider the following example, which recodes gender abbreviations in Big Class.jmp:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( sex =
  If(
     sex == "F", "Female",
     sex == "M", "Male",
     "Unknown" );
);
```

The script works like this:

`For Each Row( sex =`	For each row in the table, sex is the column that is recoded.
`If(`	Begins the `If()` loop.
`sex == "F", "Female",`	If the value of sex is F, replaces the value with Female.
`sex == "M", "Male",`	If the value of sex is M, replaces the value with Male.
`"Unknown" );`	If neither of the above conditions are true, replaces the value with Unknown. If this result were omitted and the value of sex were missing, the script would return a missing value.
`);`	Ends the loop.

You can also put actions and assignments in the result expression. The following example assigns 20 to `x`, because the first condition (`y < 20`) is false:

```
y = 25;
z = If( y < 20, x = y, x = 20 );
```

Note: Be careful to use two equal signs (==) for equality tests, not one equal sign (=). An `If` with an argument such as *`name=value`* assigns rather than tests the value.

Avoiding Memory Issues with Multiple If Statements

Running a script that contains dozens of If() Statements can cause memory problems. We recommend reducing the If() nesting. Suppose that your script contains 100 `If()` statements in this format:

```
If( val, If( val, If( val, If( val, If( val,
If( val, If( val, If( val, If( val, If( val,
...
) ) ) ) ) ) ) ) ) );
```

Rewrite the script:

```
If( val, val1, If(val, val2, If( val, val3,
If( val, val4, If( val, val5, If( val, val6, ...) ) ) ) ) );
```

You can also try setting the maximum call depth. The `Maximum Call Depth` preference sets the default for the maximum call depth (or stack size) in which JSL built-in functions, user-defined functions, or `Recurse()` function calls can be made. By default, the maximum call depth is set to 256.

In JMP, your JSL script has a limited amount of physical runtime stack memory in which to perform JSL function calls. By default, this size is set to 2MB. Increasing the maximum call depth can cause a physical runtime stack overflow, so incrementally increase this preference in small amounts until you find the best value that works for your JSL script.

```
Preferences[1] << Set( Maximum JMP call depth( 64 ) );
```

Match

You can use the `Match()` function to make several equality comparisons without needing to rewrite the value to be compared. The syntax is:

```
Match( x, value1, result1, value2, result2, ..., resultElse );
```

For example, the following script recodes gender abbreviations in Big Class.jmp:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( sex =
  Match(
     sex,
     "F", "Female",
     "M", "Male",
     "Unknown");
  );
```

The script works like this:

`For Each Row( sex =`	For each row in the table, `sex` is the column that is recoded.
`Match(`	Begins the `Match()` loop.
`sex,`	Specifies `sex` as the match argument.
`"F", "Female",`	If the value matches "F" replace it with “Female”.
`"M", "Male",`	If the value matches "M", replace it with "Male".
`"Unknown" );`	If F or M are not matched, replaces the value with "Unknown".
`);`	Ends the loop.

This `Match()` example is a simplified version of the example in “If”. The advantage of `Match()` is that you define the comparison value once rather than repeat it in each condition. The disadvantage is that you cannot use expressions with operators as you can with `If`; the argument `sex == "F"` returns an error in a `Match()` expression.

With more groups of conditions and results, the value of `Match()` becomes more apparent. The following script would require many additional lines of code with `If()`.

```
dt = Open( "$SAMPLE_DATA/Travel Costs.jmp" );
For Each Row(
   Booking Day of Week = Match( Booking Day of Week,
      "Sunday", "SUN",
      "Monday", "MON",
      "Tuesday", "TUE",
      "Wednesday", "WED",
      "Thursday", "THU",
      "Friday", "FRI",
      "Saturday", "SAT",
      "Not Specified"
   )
);
```

Be careful with the data type of the condition and result. In the preceding example, the conditions and results are both character data. If the data types do not match, JMP automatically changes the column’s data type. The results might not be what you want.

Note: The `Match()` function explicitly checks to see if the compare expression *x* is missing and if the value of *value1* is missing, then it returns the value of `result1`; otherwise it continues to compare the expression *x* to each *valueN* value in each *valueN/resultN* pair, ignoring any missing values. If the expression *x* is equal to any of the *valueN* value, then the corresponding *resultN* value is returned. If no matching *valueN* value is found, then the *resultElse* value is returned.

Choose

The `Choose()` function shortens scripts even more than `Match()`, provided the arguments are tested against integers. The syntax is:

```
Choose( expr, result1, result2, result3, ..., resultElse );
```

Suppose you have a data table with a group column of numeric values from 1 through 7. If the first cell contains the number 1, the following script returns x = "Low".

```
x = ( Choose( group[1], "Low", "Medium", "High", "Unknown" ); );
Show( x );
```

The script works like this:

`x =`	Creates the x variable.
`Choose(`	Begins the `Choose()` loop.
`group[1],`	Evaluates the value of group in the first row.
`"Low",`	If the value of group is 1, return "Low".
`"Medium",`	If the value of group is 2, return "Medium".
`"High",`	If the value of group is 3, return "High".
`"Unknown"`	Otherwise, return "Unknown".
`)`	Ends the loop.
`);`	Closes the x variable.
`Show( x );`	Returns the value of x.

If the expression evaluates to an out-of-range integer (such as 7 when only 4 replacement values are listed), the last result is returned. In the preceding example, "Unknown" is returned.

Notice that `If()` and `Match()` require more code to achieve the same results as the `Choose` function:

```
If(
  group[1] == 1, "Low",
  group[1] == 2, "Medium",
  group[1] == 3, "High",
  "Unknown"
);
Match( group, 1, "Low", 2, "Medium", 3, "High", "Unknown" );
```

Note: If the data types in the expression do not match, JMP automatically changes the column's data type.

Interpolate

The `Interpolate()` function performs linear interpolation for continuous data or bilinear interpolation for two-dimensional continuous data.

In the simplest case, the function finds the `y` value that corresponds to a given `x` value between two sets of points. You might use `Interpolate()` to calculate missing values between data points.

The data points can be specified as alternating ordered arguments:

```
Interpolate( x, x1, y1, x2, y2, ... );
```

or as matrices containing the `x` and `y` values:

```
Interpolate( x, xmatrix, ymatrix );
```

Suppose that your data set includes the height of individuals from age 20 through 25. However, there is no data point for age 23. To estimate the height for 23-year-olds, use interpolation. The following example shows the value that you want to evaluate (age 23), followed by matrices for ages (20 through 25) and heights (59 through 75).

```
Interpolate( 23, [20 21 22 24 25], [59 62 56 69 75] );
```

returns:

```
62.5
```

The value 62.5 is halfway between the `y` values 56 and 69, just as 23 is halfway between the `x` values 22 and 24.

You can also interpolate multiple points by specifying a matrix or list of values in the first argument.

```
Interpolate( [23, 24.5], [20 21 22 24 25], [59 62 56 69 75] );
```

returns:

```
[62.5, 72]
```

Bilinear Interpolation

You can also interpolate in two-dimensions:

```
Interpolate( {x, y}, xvector, yvector, zmatrix );
```

Here, the first argument is a list of two points, the second and third arguments are vectors that define the grid of `x` and `y` values, and the fourth argument is a matrix of data points. The function then finds the interpolated `z` value within the appropriate quadrant of the `zmatrix`. The appropriate quadrant is found by comparing the `x` and `y` values to the `xvector` and `yvector` arguments.

Suppose you have a 2x3 matrix and you want to interpolate a point that is halfway between the first and second rows of points and halfway between the second and third columns of points.

```
Interpolate( {0.5, 0.75}, [0 1], [0 0.5 1], [10 15 20, 12 16 18] );
```

returns:

```
17.25
```

Note that 0.5 is halfway between 0 and 1 for the x values (corresponding to the rows of the matrix) and 0.75 is halfway between 0.5 and 1 for the y values (corresponding to the columns of the matrix). You can find the bilinear interpolation in two steps:

1. Find the halfway points between the second and third columns for the each row of the matrix:
   - Row 1: 17.5 is halfway between 15 and 20.
   - Row 2: 17 is halfway between 16 and 18.
2. Find the halfway point between the two points in the previous step: 17.25.

Notes:

- The `x` values must be in ascending order. For example, `Interpolate(2,1,1,3,3)` returns 2. However, `Interpolate(2,3,3,1,1)` returns a missing value (.).
- `Interpolate()` is best used for continuous data, but `Step()` is designed for discrete data. See "Step".

Step

The `Step()` is like `Interpolate()` except that it finds the corresponding `y` for a given `x` from a step-function fit rather than a linear fit. Use `Step()` with discrete `y` values (that is, when the `y` value's corresponding `x` value can be only y_1 or y_2). However, when the `y` value's corresponding `x` value can fall between y_1 and y_2, use `Interpolate()`.

As with `Interpolate`, the data points can be specified as a list:

```
Step( x, x1, y1, x2, y2, ... );
```

or as matrices containing the `x` and `y` values:

```
Step( x, xmatrix, ymatrix );
```

Suppose that your data table shows the discount percentage for purchases of $25, $50, $75, and $100. You want to create a graph that shows the discount for a $35 purchase, which the data table does not specify. The following example shows the value that you want to evaluate, 35, followed by matrices for purchases from $25 to $100.

```
Step( 35, [25 50 75 100], [5 10 15 25] );
```

returns:

```
5
```

If the discounts were on a sliding scale (in this example, between 5 and 10, you would use `Interpolate()`:

```
Interpolate( 35, [25 50 75 100], [5 10 15 25] );
```

returns:

```
7
```

As with `Interpolate()`, the data points must create a positive slope.

Compare Incomplete or Mismatched Data

Comparing data that contains missing values can return misleading results unless you specify a condition that is always true or use functions such as `Is Missing()` or `Zero Or Missing()`. Comparisons of data with mismatched types (numeric versus character) or data in matrices can also be confusing.

Table 5.5 shows examples of such comparisons and matrices and explanations of the results. For a review of operators used in comparisons, see "Operators". For an overview of comparing data, see the *JSL Syntax Reference*.

Note: Matrices must include the same number of columns or rows.

Table 5.5 Some Special-Case Comparison Tests

Test	Result	Explanation
`m=.; m==1`	.	An equality test with a missing value returns missing.
`m=.; m!=1`	.	An inequality test with a missing value returns missing.
`m=.; m<1; m>1; and so on`	.	A comparison with a missing value returns missing (unless it could not possibly be true, see next).
`m=.; 1<m<0`	0	A comparison involving a missing value that could not possibly be true returns false; false takes precedence over missing for comparisons with more than two arguments (as with logical operators).
`{a, b}==List(a, b)`	1	An equality test of list arguments returns a single result.
`{a, b}<{a, c}`	.	A comparison test of list arguments is not allowed.
`1=="abc"`	0	An equality test with mixed data types returns false.
`1<="abc"`	.	A comparison with mixed data types returns missing.
`[1 2 3]==[2 2 5]`	`[0 1 0]`	An equality test of matrices returns a matrix of elementwise results. When a matrix is compared to a matrix, comparison is done element-by-element and returns a matrix of 1s and 0s.
`[1 2 3]==2`	`[0 1 0]`	An equality test of a matrix and a matrix filled with 2s. If a matrix is compared to a number, the number is treated as a matrix filled with that number.
`[1 2 3] < [2 2 5]`	`[1 0 1]`	A comparison of matrices returns a matrix of elementwise results.
`[1 2 3] < 2`	`[1 0 0]`	A comparison of a matrix and a matrix filled with 2s.
`Is Missing(m)`	1	Returns 1 for a missing value and returns 0 otherwise. For missing character values, you can also use empty quotes for the comparison, as in m == "".
`Zero Or Missing(m)`	1	Returns 1 when the value is 0 or missing. The argument must be numeric or a matrix and not a string.
`All([2 2]==[1 2])`	0	Summarizes elementwise comparisons; returns 1 only if *all* comparisons are true and returns 0 otherwise.

Table 5.5 Some Special-Case Comparison Tests *(Continued)*

Test	Result	Explanation
`Any([2 2]==[1 2])`	1	Summarizes elementwise comparisons; returns 1 if *any* comparison is true and returns 0 otherwise.

Missing Values

In a comparison, missing values typically return missing, not true or false. For this reason, it is very important to include a result that is always true. Suppose that a data table column contains the values 1, 2, 3, and a missing value in column A. A formula in column B sets up the comparison.

```
New Table( "Testing Comparisons",
   Add Rows( 4 ),
   New Column( "A",
      Numeric,
      "Continuous",
      Format( "Best", 10 ),
      Set Values( [1, 2, 3, .] )
   ),
   New Column( "B", Character, "Nominal", Formula( If( :A, "true", 1, "false"
   ) ) )
);
```

The following values are stored in column B:

```
"true"
"true"
"true"
"false"
```

The script works like this:

`If(`	Begins the comparison.
`:A, "true",`	If the value of A is nonmissing and nonzero, the result is "true". This comparison is true for the first three rows.
`1, "false"`	The value of 1 is always true, so the missing value returns "false".
`);`	Closes the comparison.

Notes:

- When a cell contains a non-empty string, the comparison fails with the first comparison. The following script returns a missing value if a is a string. It works for numeric values.

```
a = "a";
If( a, "true", "false" );
```

- When one value is false and another one is missing, `Or()` returns missing. Use `OrMZ()` to return false instead. See the *JSL Syntax Reference*.
- When one value is true and another one is missing, `And()` returns missing. Use `AndMZ()` to return false instead. See the *JSL Syntax Reference*.
- `MatchMZ()` is similar to `Match()` except that missing values are treated as 0.
- `IfMZ()` is similar to `If()` except that missing values are treated as 0.

Is Missing

If you know that some values are missing, you can also compare with `Is Missing()`. The comparison in the preceding example can be rewritten to return "missing" for missing values:

```
If( :A, "true", Is Missing( :A ), "missing", "false" );
```

The preceding expression returns "true" when A is nonmissing and nonzero, "missing" when A is missing, and "false" otherwise.

Zero Or Missing

If the missing value could be 0, use the `Zero Or Missing()` function instead:

```
Zero Or Missing( A );
```

This expression returns 1 when A is 0 or missing.

Tip: You cannot compare a known value with an explicitly defined missing value, only with variables, matrices, or other things that could *contain* missing values.

Inquiry Functions

Inquiry functions identify the type of an element, such as a string, list, or matrix. You can then write a script specific to that element type.

JMP also uses inquiry functions to determine the writability of a directory or file and to identify a computer's operating system and the JMP version.

General Element Types

The Type() function returns a string naming the type of the resulting value. For example:

```
Show( Type( 1 ), Type( "hi" ), Type( {"a", 2} ), Type( [10 24 325] ) );
```

results in:

```
Type(1) = "Integer"
Type("hi") = "String"
Type({"a", 2}) = "List"
Type([10 24 325]) = "Matrix";
```

Specific Element Types

Other inquiry functions (such as Is Matrix(), Is List(), Is Scriptable(), and so on) let you test for specific types of objects. In the following example, Is Matrix() evaluates as true, then the specified calculations are run:

```
a = [2 3];
b = [1, 1];
c = a * b;
If( Is Matrix( c ),
   (c ^ a) / (a * b),
   Print( "c is not a matrix." )
);
   [5 25]
```

Is Scriptable() returns 1 when the object is scriptable. Four variables in the following example refer to a data table, column, platform, and report. All four objects are scriptable, so Is Scriptable() returns 1 for each example.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = Column( "weight" );
plat = Bivariate( Y( :weight ), X( :height ) );
rep = Report( plat );
Show( Is Scriptable( dt ) );
   Is Scriptable(dt) = 1;
Show( Is Scriptable( col ) );
   Is Scriptable(col) = 1;
Show( Is Scriptable( plat ) );
   Is Scriptable(plat) = 1;
Show( Is Scriptable( rep ) );
   Is Scriptable(rep) = 1;
```

Is Empty() tests to see whether a variable has a value, a function, an expression, or a reference to an object. It returns 1 if the variable is undefined or holds the Empty() value.Otherwise, you get errors when referring to something that has not been created or assigned a value yet. Programmers call this an *uninitialized variable*.

Here is an example of a test to see whether a data table is opened and therefore assigned to the `dt` variable. If a data table is not opened, the `Open()` function prompts the user to open the table.

```
If( Is Empty( dt = Current Data Table() ),
   dt = Open()
);
```

You can use `Is Empty()` for any variable (such as global variable, local variable, and columns).

Table 5.6 shows functions that identify object types.

Table 5.6 Inquiry Functions That Identify Object Types

Syntax	Explanation
`Is Associative Array(`*x*`)`	Returns 1 if the evaluated argument is an associative array or 0 otherwise.
`Is Directory(x)`	Returns 1 if the x argument is a directory and 0 otherwise.

`Is Empty(`*global*`)`
`Is Empty(`*dt*`)`
`Is Empty(`*col*`)`
	Returns 1 if the global variable, data table, or data column does not have a value (is uninitialized) or 0 otherwise.

`Is Expr(`*x*`)`	Returns 1 if the evaluated argument is an expression or 0 otherwise.
`Is File(x)`	Returns 1 if the x argument is a file and 0 otherwise.
`Is List(`*x*`)`	Returns 1 if the evaluated argument is a list or 0 otherwise.
`Is Matrix(`*x*`)`	Returns 1 if the evaluated argument is a matrix or 0 otherwise.
`Is Name(`*x*`)`	Returns 1 if the evaluated argument is a name or 0 otherwise. See "Retrieve a Stored Expression Rather than its Result".
`Is Namespace(`*x*`)`	Returns 1 if the evaluated argument is a namespace or 0 otherwise.
`Is Number(`*x*`)`	Returns 1 if the evaluated argument is a number or missing numeric value or 0 otherwise.
`Is Scriptable(`*x*`)`	Returns 1 if the evaluated argument is a scriptable object or 0 otherwise.
`Is String(`*x*`)`	Returns 1 if the evaluated argument is a string or 0 otherwise.
`Type(`*x*`)`	Returns a string naming the type of x.

Object Attributes

JMP provides the following functions to determine whether a file or directory is writable before attempting to write to them. Use these functions in combination with `Is Directory(path)` and `Is File(path)` to verify a script destination and attribute. See the *JSL Syntax Reference* for additional information.

The `Is Directory Writable(path)` function returns 1 if the directory specified in the path argument is writable and 0 otherwise.

The `Is File Writable(path)` function returns 1 if the file specified in the path argument is writable and 0 otherwise.

For example, the following code verifies the path refers to a directory and then checks to ensure the directory is writable:

```
If( Is Directory( "$SAMPLE_DATA/Loss Function Templates" ),
  If( Is Directory Writable( "$SAMPLE_DATA/Loss Function Templates" ),
     "Directory is writable.",
     "Directory is read only!"
  ),
  "Is a read only directory."
);
```

Host Information

The `Host Is()` inquiry function identifies the current operating system. Then actions specific to that operating system can be performed.

The following example specifies text sizes in reports for different operating systems. If you commonly write your scripts on Windows and share them with macOS users, the results can look different from what you intended. For example, the following expression sets the text to a larger size (12) on macOS and a smaller size (10) on Windows:

```
textsize = If( Host is( "Mac" ), 12, 10 );
```

Version Information

The `JMP Version()` inquiry function returns the JMP version as a string. You might use this function to determine the JMP version and then run a script compatible with that version.

```
JMP Version(); // returns "16.0.0" in JMP 16
JMP Version(); // returns " 9.0.0" in JMP 9
```

Notice that a leading blank is inserted before versions less than 10.0.0. This blank helps when comparing version numbers. Without the leading blank, 9.0.0 is interpreted as greater than 10.0.0.

Chapter 6

Types of Data

Working with Numbers, Strings, Dates, Currency, and More

This chapter discusses basic data types:

- numbers and strings
- paths, which are a special type of string
- dates and times, which can be either special numbers or special strings
- currency
- hexadecimal values and blobs

At the end of the chapter are two sections that show more advanced methods of interacting with strings and pattern matching with regular expressions.

Contents

Numbers and Strings 145
Formatting Numbers 146
Unicode Characters 147
Path Variables 147
Create and Customize Path Variables 150
Relative Paths 151
File Path Separators 151
Date-Time Functions and Formats 152
Date-Time Values 152
Program with Date-Time Functions 153
Date-Time Values in Data Tables 160
Custom Date-Time Formatting with Format Patterns 164
Currency 166
Hexadecimal and BLOB Functions 167
Work with Character Functions 170
Concat 170
Munger 171
Repeat 172
Substitute and Substitute Into 173
Regular Expressions 174
Regex 174
Regex Match 176
Special Characters in Regular Expressions 178
Escaped Characters in Regular Expressions 179
Greedy and Reluctant Regular Expressions 181
Backreferences and Capturing Groups 183
Lookaround Assertions 184
Pattern Matching 185
Parsing Strings in Fixed Fields 186
Patterns and Case 188
Troubleshooting Patterns 188

Numbers and Strings

Numbers can be written as integers, decimal numbers, in scientific notation with an E preceding the power of ten, and as date-time values. A single period by itself is the missing numeric value.

For example, these are all numbers:

```
.   1   12   1.234  3E3  0.314159265E+1 1E-20
```

One or more characters placed within double quotation marks constitute a *string*. For example, these are all strings:

```
"Green" "Hello,\NWorld!" "54"
```

Notice that if a number is in quotation marks, it is a string, not a number. There are two functions you can use to change a number into a string or a string into a number.

- Use `Num()` to convert a string into a number. For example:

```
Num( "54" );
   54
```

Note: `Num()` cannot convert non-numeric characters, so it produces a missing value.

```
Num( "Hello" );
   .
```

- Use `Char()` to convert a number into a string. For example:

```
Char( 54 );
   "54"
Char( 3E3 )
   "3000"
```

To preserve locale-specific numeric formatting in `Num()` or `Char()` output, include the `<<Use Locale(1)` option as shown in the following example:

```
Char( 42, 5, 2, << Use Locale( 1 ) );
   // results in the character value "42,00" in the French locale
```

To look at each character in a string, use the `Substr()` function. This example looks for the letter "a" in the string and prints a message to the log:

```
ch = Substr( "alphabetic", 1, 1 ); // start and end with the first character
If( ch == "a",
   Print( "First letter is a." )
);
   "First letter is a."
```

Formatting Numbers

The `Format()` function provides options for formatting numbers with thousands separators, decimal points, geographic symbols, *p*-values, and so on. The following example shows a basic format for a fixed decimal column.

```
Format( x, "Fixed Dec", 10, 2, "Use thousands separator");
```

The format starts with the number to be formatted. Then you can specify the width of the number, the number of decimal places, and whether to use a thousands separator.

Here are other examples:

```
Format( x, "Currency", 20, <<Use Locale(0) );
  // ignores computer locale and uses a period for the decimal separator
Format( x, "Currency", 20, "Use thousands separator" );
Format( x, "m/d/y", 10 );
Format( x, "Precision", 10, 2, "Keep trailing zeroes", "Keep all whole digits"
  );
Format( x, "Latitude DDD", "PUNDIR"); // "PUN" for punctuation, "DIR" for
  direction, PUNDIR for both
Format( x, "Custom", Formula( Abs( value ) ), 15 );
```

The syntax for `Function()` arguments is shown in the *JSL Syntax Reference*. You can also find the argument names and options in the data table Column Info window.

See "Date-Time Functions and Formats" and "Currency" for more information about formatting date-time values and currency.

Note: If the date format is unknown, an error is written to the log.

Matrix Box, Number Col Box, Number Col Edit Box, Number Edit Box

To format numbers in `Matrix Box()`, `Number Col Box()`, `Number Col Edit Box()`, and `Number Edit Box()`, send the `Set Format` message to the display box. You can then specify the same arguments as for the `Format()` function, such as the following:

```
<<Set Format( 10, 2, "Use thousands separator");
<<Set Format( "Currency", "EUR", 20 );
```

The syntax for `Set Format` arguments is shown in the *JSL Syntax Reference*. You can also find the argument names and options in the data table Column Info window.

Note: If you specify the number of decimal places, that number must be preceded with the width.

Unicode Characters

JMP supports both Unicode UTF-8 and UTF-16 standards for encoding and representing text for most of the world languages. See the The Unicode Consortium for code charts and more information on the Unicode standard.

To display Unicode characters in JMP, precede the Unicode code for the character with '\!'. For example:

- Greek letter sigma (σ) in Unicode = U+03C3; in JMP, use \!U03C3
- Greek letter mu (μ) in Unicode = U+03BC; in JMP, use \!U03BC

To use Unicode to express superscripts and subscripts:

- subscript 1 ($_1$) in Unicode = U+2081; in JMP, use \!U2081
- superscript 2 (^{2}) in Unicode = U+00B2; in JMP, use \!U00B2

To express x^2 in Unicode, in JMP, use \!U0078\!U00B2.

To use Unicode surrogate pairs, specify the Unicode escape sequence twice. If you specify an emoji Unicode character, you might need to change the font for it to appear correctly. The following example shows how to include an emoji in a text box.

```
New Window( "Emoji Example",
   V List Box(
      Text Box( "Surrogate Pair with space: \!UD83D \!UDE00", Set Font( "Segoe
   UI Emoji" ) ),
      Text Box( "Surrogate Pair without space: \!UD83D\!UDE00", Set Font(
   "Segoe UI Emoji" ) )
   )
);
```

Path Variables

Path variables are shortcuts to directories or files. Rather than enter the entire path to the directory or file, you use the path variable in a script. A path variable is a special type of string and is always contained within double quotation marks.

One common predefined path variable in JMP is $SAMPLE_DATA. This variable points to the sample data folder in your JMP or JMP Pro installation folder. The following example opens the Big Class.jmp sample data table.

```
Open( "$SAMPLE_DATA/Big Class.jmp" );
```

Several path variables are predefined in JMP. The following table shows the definitions for the current JMP version. Variables in previous versions of JMP might differ.

Table 6.1 Predefined Path Variable Definitions

Variable	Path
ADDIN_HOME(com.your.addin.id) Defines the folder where the specified add-in is stored.	Add-In Builder stores the specified add-in in the following folders, based on your operating system: • Windows: "C:/Users/<username>/AppData/Roaming/SAS/JMP/Addins/" • macOS: "/Users/<username>/Library/Application Support/JMP/Addins/" For example, `Convert File Path( "$ADDIN_HOME(com.your.addin.id)` evaluates to the location of the `com.your.addin.id` add-in. To install your add-ins in a shared network location, use the `Register Addin()` function. See "Register an Add-In Using JSL".
ALL_HOME Defines a folder that can be accessed by all users on a machine.	• Windows (JMP): "/C:/ProgramData/SAS/JMP/16/" • Windows (JMP Pro): "/C:/ProgramData/SAS/JMPPro/16/" • macOS: "/Library/Application Support/JMP/16/" To see if the directory exists, run the script `Is Directory("$ALL_HOME");`. If the folder exists, 1 is returned.
BUILTIN_SCRIPTS	• Windows (JMP): "/C:/Program Files/SAS/JMP/16/Resources/Builtins/" • Windows (JMP Pro): "/C:/Program Files/SAS/JMPPRO/16/Resources/Builtins/" • macOS: "/Applications/JMP 16.app/Contents/Resources/Builtins/"
DESKTOP	• Windows: "/C:/Users/<username>/Desktop/" • macOS "/Users/<username>/Desktop/"
DOCUMENTS	• Windows: "/C:/Users/<username>/Documents/" • macOS: "/Users/<username>/Documents/"
DOWNLOADS	• Windows: "/C:/Users/<username>/Downloads/" • macOS: "/Users/<username>/Downloads/"
GENOMICS_HOME	"/<JMP Genomics installation directory>/"

Table 6.1 Predefined Path Variable Definitions *(Continued)*

Variable	Path
HOME	• Windows (JMP): "/C:/Users/<username>/AppData/Roaming/SAS/JMP/16/" • Windows (JMP Pro): "/C:/Users/<username>/AppData/Roaming/SAS/JMPPro/16/" • macOS: "/Users/<username>/"
JMP_HOME	• Windows: /C:/<JMP installation directory> **Note:** The variable is not available on macOS.
SAMPLE_APPS	• Windows: /C:/<JMP installation directory>/Samples/Apps/" • macOS: "/Library/Application Support/JMP/16/Samples/Apps/"
SAMPLE_DASHBOARDS	• Windows: /C:/<JMP installation directory>/Samples/Dashboards/" • macOS: "/Library/Application Support/JMP/16/Samples/Dashboards/"
SAMPLE_DATA	• Windows: "/C:/<JMP installation directory>/Samples/Data/" • macOS: "/Library/Application Support/JMP/16/Samples/Data/"
SAMPLE_IMAGES	• Windows: "/C:/<JMP installation directory>/Samples/Images/" • macOS "/Library/Application Support/JMP/16/Samples/Images/"
SAMPLE_IMPORT_DATA	• Windows: "/C:/<JMP installation directory>/Samples/Import Data/" • macOS: "/Library/Application Support/JMP/16/Samples/Import Data/"
SAMPLE_PROJECTS	• Windows: "/C:/<JMP installation directory>/Samples/Projects/" • macOS: "/Library/Application Support/JMP/16/Samples/Projects/"
SAMPLE_SCRIPTS	• Windows: "/C:/<JMP installation directory>/Samples/Scripts/" • macOS: "/Library/Application Support/JMP/16/Samples/Scripts/"
TEMP	• Windows: "/C:/Users/<username>/AppData/Local/Temp/" • macOS: "var/folders/..."

Table 6.1 Predefined Path Variable Definitions *(Continued)*

Variable	Path
USER_APPDATA Changes to JMP preferences, menus, and the Home Window are stored here, along with Debugger session settings.	• Windows (JMP): "/C:/Users/<username>/AppData/Roaming/SAS/JMP/16/" • Windows (JMP Pro): "/C:/Users/<username>/AppData/Roaming/SAS/JMPPro/16/" • macOS: "/Users/<username>/Library/Application Support/JMP/16/"

Path variable definitions are updated automatically based on the version of JMP you are using. For example, when you run a JMP 14 script in JMP 16, the JMP 16 path variable definitions are used.

To see the definition of any path variable, use the function `Get Path Variable`:

```
Get Path Variable( "HOME" );
   "/C:/Users/<username>/AppData/Roaming/SAS/JMP/16/"
```

Note that you don't include a dollar sign for `Set Path Variable()` or `Get Path Variable()`. But you must include the dollar sign when using the variable in a script.

Trailing Slashes

Make sure to include a trailing slash after the path variable. In the following example, the root name `"Big Class"` is assigned to the dtName variable. The `Open()` expression evaluates `$SAMPLE_DATA` and the trailing slash and then appends the `dtName` value along with the file extension `.jmp`.

```
dtName = "Big Class";
dt = Open( "$SAMPLE_DATA/" || dtName || ".jmp" );
```

The path is interpreted as:

```
C:/Program Files/SAS/JMP/16/Samples/Data/Big Class.jmp
```

Without the slash that follows `$SAMPLE_DATA`, the path is interpreted as:

```
C:/Program Files/SAS/JMP/16/Samples/DataBig Class.jmp
```

Create and Customize Path Variables

You can create your own path variables or override some of the built-in variables with the `Set Path Variable()`. In the following example, the path variable is called *root*. The variable points to the `c:/` directory.

```
Set Path Variable( "root", "c:/" );
```

To get the value of the new variable, use Get Path Variable().

```
Get Path Variable( "root" );
  "c:/"
```

Use your path variable as you would other variables. The following expression opens the myimportdata.txt file in the c:/ directory.

```
Open( "$root/myimportdata.txt" );
```

As with getting path variables, omit the dollar sign when setting path variables.

Relative Paths

If you plan to use relative paths in variables, you must set the default directory. Then any path not preceded by a drive letter is relative to the default directory. Here is an example:

```
Set Default Directory( "c:/users/smith/data" );
```

To return the value of the default directory, use Get Default Directory().

```
Get Default Directory(); // returns "c:/users/smith/data"
```

So the following expression resolves as C:/users/smith/data/cleansers.jmp:

```
Open("cleansers.jmp");
```

File Path Separators

In JMP, the preferred file path format is the Portable Operating System Interface (POSIX), or UNIX, format with forward slashes (/) as separators. This means that you do not have to identify the current operating system in scripts run on both Windows and macOS. However, each host still accepts its native format for compatibility.

You can a convert file path format from Windows to POSIX (and vice versa) using Convert File Path(). Converting from a POSIX to a Windows path might be useful when you need to output a path to a file or to another application. The syntax is:

```
Convert File Path (path, <absolute|relative>, <POSIX|windows>, <base(path)>);
```

For example, the following script converts a POSIX path to a Windows path:

```
Convert File Path( "c:/users/smith", windows);
  c:\users\smith
```

You can substitute a path variable (such as $HOME) for the path inside quotes.

Date-Time Functions and Formats

A date-time value consists of any portion of a date or time. The value can be seconds (3388594698), a complete date (such as "Wednesday, May 18, 2011"), the date and time ("05/18/2011 8:18:18 PM"), the week number (3), and so on.

JMP lets you convert date-time values to common formats, perform arithmetic on the values, and manipulate the data in a number of ways.

Tip: For descriptions of all date-time functions and their arguments, see the *JSL Syntax Reference*.

Date-Time Values

Date-time values are stored and calculated as the number of seconds since midnight, January 1, 1904. For example:

```
Today(); // returns 3388649872 on May 19, 2011 at 12:00:00 AM
```

As with `Today()`, the `Date DMY()` and `Date MDY()` functions also return month, day, and year arguments as seconds. For example, if it were 12:00:00 a.m. on May 19, 2011, all of the following statements would return the same value:

```
Date DMY( 19, 5, 2011 );
Date MDY( 5, 19, 2011 );
Today();
   3388608000
```

The `As Date()` function takes the number of seconds and displays it as a date or duration.

- Values that represent one year or more are returned as dates:

```
As Date( 3388608000 );
   19May2011
```

- Values that represent less than a year are returned as durations.

```
As Date( 50000 );
   :0:13:53:20
```

You can use date-time values in two ways:

- a literal value, for example `19May2011:10:10`
- a string, for example `"Thursday, May 19, 2011"`

You can perform arithmetic with date-time literals, which use the number of seconds as the base number.

```
As Date( 19May2011 + 1 );
```

19May2011:00:00:01

Program with Date-Time Functions

Table 6.2 shows functions that convert seconds into date-time values and date-time values into seconds.

Table 6.2 Date-Time Functions

Function	Explanation
Abbrev Date(*date*)	Returns a string representation for the *date* supplied. The format is based on your computer's regional setting. So for the English (United States) locale, the date is formatted like "02/29/2004". Even if you are running JMP in English with a different locale, the locale format is applied.

As Date(*expression*)	Formats a number or expression so that it shows as a date or duration in a text window. For example, values that represent one year or more are returned as dates.
	`x = As Date( 8Dec2000 + In Days( 2 ) );`
	shows as:
	10Dec2000
	Values that represent less than a year are returned as durations.
	`As Date( 50000 );`
	shows as:
	:0:13:53:20

Date DMY(*day*, *month*, *year*)	Returns the specified date expressed as the number of seconds since midnight, 1 January 1904. For example, the second Leap Day of the third millennium is DateDMY(29,2,2004), which returns 3160857600.
Date MDY(*month*, *day*, *year*)	Returns the specified date expressed as the number of seconds since midnight, 1 January 1904. For example, the second Leap Day of the third millennium is DateMDY(2,29,2004), which returns 3160857600.
Day Of Week(*date*)	Returns an integer representation for the day of the week of the *date* supplied. Weeks are Sunday–Saturday.

Table 6.2 Date-Time Functions *(Continued)*

Function	Explanation
`Day Of Year(date)`	Returns an integer representation for the day of the year of the *date* supplied.
`Day(date)`	Returns an integer representation for the day of the month of the *date* supplied.
`Format(date, "format")`	Returns the *value* in the *format* specified in the second argument. Most typically used for formatting datetime values from a number of seconds to a formatted date. Format choices are those shown in the **Column Info** dialog box. Also see Table 6.3.
`Hour(datetime)`	Returns an integer representation for the hour part of the *date-time* value supplied.
`In Days(n)`	These functions return the number of seconds per *n* minutes, hours, days, weeks, or years. Divide by these functions to express an interval in seconds as an interval in other units.
`In Hours(n)`	
`In Minutes(n)`	
`In Weeks(n)`	
`In Years(n)`	
`Long Date(date)`	Returns a string representation for the specified *date*. The format is based on your computer's regional setting. So for the English (United States) locale, the date is formatted like `"Sunday, February 29, 2004"`. Even if you are running JMP in English with a different locale, the locale format is applied.
`MDYHMS(date)`	Returns a string representation for the *date* supplied, formatted like `"2/29/2004 00:02:20 AM"`.
`Minute(date-time)`	Returns an integer representation for the minute part of the *date-time* value supplied.
`Month(date)`	Returns an integer representation for the month of the *date* supplied.
`Num(date-time)`	Returns an integer representation for the *date-time* value supplied.
`InFormat(string, "format")` `Parse Date(string, "format")`	Parses a *string* of a given *format* and returns datetime value expressed as if surrounded by `As Date()`, returning the date in `dd`Monyyyy format.

Table 6.2 Date-Time Functions *(Continued)*

Function	Explanation
`Second(`*`date-time`*`)`	Returns an integer representation for the second part of the *`date-time`* value supplied.
`Short Date(`*`date`*`)`	Returns a string representation for the *`date`* supplied, in the format MM/DD/YYYY, regardless of locale (for example, "02/29/2004").
`Time Of Day(`*`date`*`)`	Returns an integer representation for the time of day of the *`date-time`* supplied.
`Today()`	Returns the current date and time expressed as the number of seconds since midnight, 1 January 1904. No arguments are accepted, but the parentheses are still needed.
`Week Of Year(`*`date`*`, <rule_n>)`	Returns the week of the year as a date-time value. Three rules determine when the first week of the year begins. • With rule 1 (the default), weeks start on Sunday, with the first Sunday of the year being week 2. Week 1 is a partial week or empty. • With rule 2, the first Sunday begins with week 1, with previous days being week 0. • With rule 3, the ISO-8601 week number is returned. Weeks start on Monday. Week 1 is the first week of the year with four days in that year. It is possible for the first or last three days of the year to belong to the neighboring year's week number.
`Year(`*`date`*`)`	Returns an integer representation for the year of the specified *`date`*.

Examples of Common Date-Time Functions

You can use any function that returns seconds within a function that returns a date-time.

For example, if today is May 19, 2011 and the time is 11:37:52 AM, `Today()` returns the number of seconds, and the functions that follow show that number of seconds since the base time in different date-time formats:

```
Today()
   3666955847
Short Date( Today() );
   "03/13/2020"
```

```
Long Date( Today() );
   "Friday, March 13, 2020"
Abbrev Date( Today() );
   "3/13/2020"
MDYHMS( Today() );
   "03/13/2020 2:50:17 PM"
```

The date argument in parentheses can be seconds (or any function that returns seconds), or any date-time literal value. For example, both of the following expressions return the same value:

```
Long Date( 3388649872 );
Long Date( 19May2011 );
   "Friday, March 13, 2020"
```

Note: `Long Date()` and `Abbrev Date()` values are formatted according to your computer's regional settings.

Extract Parts of Dates

You can extract parts of date values using the functions `Month()`, `Day()`, `Year()`, `Day Of Week()`, `Day Of Year()`, `Week Of Year()`, `Time Of Day`, `Hour()`, `Minute()`, and `Second()`, which all return integers. If today is May 24th, 2011, each of the following examples returns the 144th day of the year:

```
Day of Year( Today() );
Day of Year( 24May2011 );
Day of Year( Date MDY( 5, 24, 2011 ) );
   144
```

Example

A data table column named Date contains date-time values that are formatted as `"m/d/y"`. You want to create a column that shows only the time. In the following script, the second column's formula extracts the time of day from the Date value in the first column.

```
New Table( "Assembly Tests",
   Add Rows( 1 ),
   New Column( "Date",
      Numeric, Continuous,
      Format( "m/d/y" ),
      Set Values( [3389083557] )
   ),
   New Column( "Time",
      Numeric, Continuous,
      Formula( Format( Time Of Day( :Date ), "h:m:s") )
   )
);
```

Figure 6.1 shows the result. Note that the time of day does not appear in the Date column, because the `Format` function applies the "m/d/y" format.

Figure 6.1 Example of Extracting the Time

	Date	Time
1	05/24/2011	12:05:57 PM

Rules for Determining the Week of the Year

`Week of Year()` returns the week of the year as a date-time value. Three rules determine when the first week of the year begins.

- With rule 1 (the default), weeks start on Sunday, with the first Sunday of the year being week 2. Week 1 is a partial week or empty.

```
Week Of Year( Date DMY( 19, 6, 2013 ), 1 );
  25
```

- With rule 2, the first Sunday begins with week 1, with previous days being week 0.

```
Week Of Year( Date DMY( 19, 6, 2013), 2 ) ;
  24
```

- With rule 3, the ISO-8601 week number is returned. Weeks start on Monday. Week 1 is the first week of the year with four days in that year. It is possible for the first or last three days of the year to belong to the neighboring year's week number.

```
Week Of Year( Date DMY( 19, 6, 2013 ), 3 );
  25
```

Arithmetic on Dates

You can perform the usual arithmetic operations with date-time data as with any other numeric data. One option is simple arithmetic, such as subtracting a number from a date-time value.

Another option is writing a formula to perform the arithmetic.

Example

The Date column in your data table shows when a customer uses his credit card to buy gas. You want to know how many days elapse between purchases. The following script creates a Days elapsed column. The formula in that column subtracts the Date value in the current row from that of the previous row.

```
New Table( "Gas Purchases",
  Add Rows( 3 ),
  New Column( "Date",
```

```
        Numeric,
        "Continuous",
        Format( "m/d/y" ),
        Set Values( [3392323200 3393532800 3394828800] )
    ),
    New Column( "Days elapsed",
        Formula(
        // returns a missing value for the first row
            If( Row() == 1,
                .,
                (:Date[Row()] - :Date[Row() - 1]) / In Days()
            )
        )
    )
);
```

Figure 6.2 shows the result.

Figure 6.2 Example of Calculating Date-Time Values

	Date	Days elapsed
1	07/01/2011	•
2	07/15/2011	14
3	07/30/2011	15

Time Intervals

The `In Minutes`, `In Hours`, `In Days`, `In Weeks`, and `In Years` functions are used to express time intervals in units rather than seconds. Each of these functions returns the number of seconds associated with a particular period of time. For example, the following expression returns the number of weeks between now and July 4, 2012.

```
(Date DMY( 04, 07, 2012 ) - Today()) / In Weeks();
  -208.225444775132
```

When the argument for the interval function is empty, JMP counts by 1. You can enter another number to change the count. For example, `In Years(10)` converts the interval to decades. The following expression returns the number of decades between now and December 31, 2037.

```
(Date DMY( 31, 12, 2037) - Today() ) / In Years( 10 );
  2.18927583529799
```

Two-Digit and Four-Digit Years

JMP applies its own algorithms for interpreting and displaying datetime strings rather than supporting operating system-specific datetime formats. However, JMP uses the date-time separators selected in the Region and Language control panel (Windows) or the Date & Time preferences (macOS) to interpret and display dates.

Two-digit years are interpreted according to the current system clock year and JMP rules. For example, when the year in a script is 11, and you run the script after 1990, the year shows as 2011.

```
Long Date( 25May11 );
  "Wednesday, May 25, 2011"
```

To avoid ambiguity, enter four-digit years. The following expression returns 1911 (rather than 2011) as indicated:

```
Long Date( 25May1911 );
  "Thursday, May 25, 1911"
```

Table 6.3 explains how JMP interprets two-digit years.

Table 6.3 How JMP Interprets Two-Digit Years

Two-Digit Year Value	When it is Evaluated	Result	Examples	Result
00–10	before 1990 (on Windows)	19__	enter 5 in year 1979	1905
	before or during 1990 (on macOS)			
	during or after 1990 (on Windows)	20__	enter 5 in year 1991	2005
	after 1990 (on macOS)			
11–89 (on Windows)	any time	current century	enter 13 in year 1988	1913
11–90 (on macOS)			enter 13 in year 2024	2013
90–99 (on Windows)	before 2011	19__	enter 99 in year 1999	1999
91–99 (on macOS)	during or after 2011	20__	enter 99 in year 2015	2099

Note: JMP always displays four-digit years regardless of the regional settings. If you need to show two-digit years, use character string functions. See "Types of Data".

Date-Time Values in Data Tables

Change Date-Time Input and Display Formats

In data tables, JMP can accept the input of date-time values in one format (the *input format*), store them internally as the number of seconds since the base date, and display them in a different date-time format. The `Informat()` and `Format()` functions give you this control.

- `Informat()` takes a string date-time value, defines the date format used in that string, and returns the date in `ddMonyyyy` format.

```
Informat( "19May2011 11:37:52 AM", "ddMonyyyy h:m:s" );
   19May2011:11:37:52
```

- `Format()` takes the number of seconds since the base date (or a date-time function that returns that number) and returns the date in the specified format.

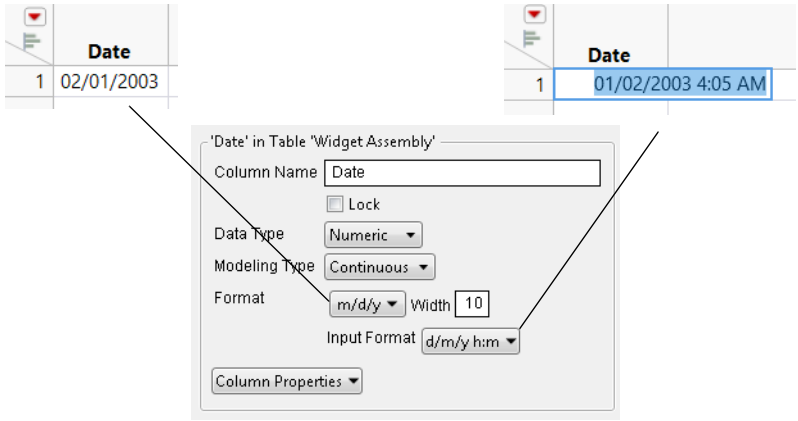
```
Format( 3388649872, "ddMonyyyy h:m:s" );
   "19May2011 11:37:52 AM"
Format( Today(), "ddMonyyyy h:m:s" );
   "19May2011 11:37:52 AM"
```

Suppose that you are entering dates into a column using the `d/m/y h:m` format, but you want to see the dates in the `m/d/y` format. `Input Format` defines the input format, and `Format` defines the display format. For example,

```
New Table( "Widget Assembly",
   Add Rows( 1 ),
   New Column( "Date",
      Numeric,
      "Continuous",
      Format( "m/d/y" ),
      Input Format( "d/m/y h:m" ),
      Set Values( [3126917100] )
   )
);
```

The `Format` and `Input Format` values are shown in the data table's column properties (Figure 6.3). Note that when you click in the cell to edit it, the date-time value appears in the input format. When you edit the value, or add a new value, the format specified in the data table column Format list is used to display the value.

Figure 6.3 Example of Date-Time Display and Input Values



Notes:

- In a script that converts a column from character to numeric, specify `Format()` and `Informat()` to prevent missing values. See "Convert Character Dates to Numeric Dates".
- The date-separator character on your computer might differ from the forward slash (/) character shown in the *Scripting Guide*.
- You can enter time values in 24-hour format (military time) or with AM or PM designators.

Table 6.4 describes the formats used as arguments in date-time functions or as data table formats. You can also use the formats for the *`format`* argument to a `Format` message to a data column. See "Set or Get Formats".

For descriptions of specific date-time functions, see the *JSL Syntax Reference*.

Notes:

- Date-time formats in which a colon or T separate the date and time always use a 24-hour format. Other formats use your computer's regional settings.
- If the format is unknown, an error is written to the log.

Table 6.4 Date-Time Formats

Type	Format argument	Example
Date only	`"m/d/y"`	`"01/02/1999"`
	`"mmddyyyy"`	`"01021999"`
	`"m/y"`	`"01/1999"`
	`"d/m/y"`	`"02/01/1999"`

Table 6.4 Date-Time Formats *(Continued)*

Type	Format argument	Example
	"ddmmyyyy"	"02011999"
	"ddMonyyyy"	"02Jan1999"
	"Monddyyyy"	"Jan021999"
	"y/m/d"	"1999/01/02"
	"yyyymmdd"	"19990102"
	"yyyy-mm-dd"	“1999-01-02“
	"yyyyQq"	1999Q1
Date and time	"m/d/y h:m"	"01/02/1999 13:01" "01/02/1999 1:01 PM"
	"m/d/y h:m:s"	"01/02/1999 13:01:55" "01/02/1999 1:01:55 PM"
	"d/m/y h:m"	"02/01/1999 13:01" "02/01/1999 1:01 PM"
	"d/m/y h:m:s"	"02/01/1999 13:01:55" "02/01/1999 1:01:55 PM"
	"y/m/d h:m"	‘1999/01/02 13:01’ ‘1999/01/02 1:01 PM’
	"y/m/d h:m:s"	‘1999/01/02 13:01:02 ‘1999/01/02 1:01:02 PM’
	"ddMonyyyy h:m"	"02Jan1999 13:01" "02Jan1999 1:01 PM"
	"ddMonyyyy h:m:s"	"02Jan1999 13:01:02" "02Jan1999 1:01:02 PM"
	"ddMonyyyy:h:m"	"02Jan1999:13:01" "02Jan1999:1:01 PM"
	"ddMonyyyy:h:m:s"	"02Jan1999:13:01:02" "02Jan1999:1:01:02 PM"
	"Monddyyyy h:m"	"Jan021999 13:01" "Jan021999 1:01 PM"
	"Monddyyyy h:m:s"	"Jan021999 13:01:02" "Jan021999 1:01:02 PM"

Table 6.4 Date-Time Formats *(Continued)*

Type	Format argument	Example
Day number and time	":day:hr:m"	"34700:13:01" ":33:001:01 PM"
	":day:hr:m:s"	"34700:13:01:02" ":33:001:01:02 PM"
	"h:m:s"	"13:01:02" "01:01:02 PM"
	"h:m"	"13:01" "01:02 PM"
	"yyyy-mm-ddThh:mm"	1999-01-02T13:01
	"yyyy-mm-ddThh:mm:ss"	1999-01-02T13:01:02
Duration	":day:hr:m"	"52:03:01" reads fifty-two days, three hours, and one minute
	":day:hr:m:s"	"52:03:01:30" reads fifty-two days, three hours, one minute, and thirty seconds
	"hr:m"	"17:37" reads seventeen hours and thirty-seven minutes
	"hr:m:s"	"17:37:04" reads seventeen hours, thirty-seven minutes, and 4 seconds
	"min:s"	"37:04" reads thirty-seven minutes and 4 seconds
Note: The following formats display the date-time according to your computer's regional settings. They are available only for the display of dates, not for date input in a data table. Examples are shown for the United States locale.		
Abbreviated date	"Date Abbrev"	(Display only) "01/02/1999"
Long date	"Date Long"	(Display only) "Saturday, January 02, 1999"
Locale date	"Locale Date"	(Display only) "01/02/1999"

Table 6.4 Date-Time Formats *(Continued)*

Type	Format argument	Example
Locale date and time	"Locale Date Time h:m"	(Display only) "01/02/1999 13:01" or "01/02/1999 01:01 PM"
	"Locale Date Time h:m:s"	(Display only) "01/02/1999 13:01:02" or "01/02/1999 01:01:02 PM"

Custom Date-Time Formatting with Format Patterns

Format patterns are strings that define a date-time format. Here's one example:

```
<YYYY></><MM></><DD> <hh><:><mm><:><ss><ampm>
```

The parts of the pattern in angle brackets are called *field descriptors*. They represent a value (such as <YYYY>, which is a four-digit year) or other date-time text (such as </>, which is a locale-specific date separator).

The conversion of a date-time value to a string and a string converted to a date-time value work as follows:

- When a date-time value is converted to a string, the field descriptors are replaced by the appropriate value or symbols.
- When a string is converted to a date-time value, a value is or matching symbols are expected to be found where each field descriptor is. Anything outside a format descriptor is treated as literal text. The literal text is replaced as-is when it is converted *to* a string, and is expected to be found where specified when converting *from* a string.

Table 6.5 Output for Custom Date-Time Formats

Format	Output
<YYYY></><MM></><DD> <hh><:><mm><:><ss><ampm>	2020/02/10 11:54:33 AM
<DayOfWeek>, <Month> <D>, <YYYY>	Wednesday, February 5, 2020
<Day>d <hh24>h <mm>m <ss>s	123d 02h 34m 56s

Note: <Day> is the day count, used as the most significant field of a duration. <Hour> and <Minute> field descriptors are also available.

Example of Outputting the Date-Time Value to the Log

The following script uses `Informat()` to read a string date-time value, defines the format pattern used in that string, and returns the date in `ddMonyyyy` format. The script then outputs the date-time value to the log.

```
dateVal = Informat( "June 16, 2016", "Format Pattern", "<Month> <D>, <YYYY>"
  );
  16Jun2016
```

Here's another way to specify the format pattern:

```
date2 = 16Jun2021;

  dayofwk2 = Format( date2, "Format Pattern", "<DayOfWeek>" );
  monthnew2 = Format( date2, "Format Pattern", "<Month>" );

  YYYYMMDD = Format( date2, "Format Pattern", "<YYYY></><MM></><DD>" );
  YYMMMDD = Format( date2, "Format Pattern", "<YY></><MMM></><DD>" );
  "21/Jun/16"
```

Example of Changing the Input Format of a Column

The following script shows how to change the input format of a column. The Input Format appears when you edit the value in a data table.

```
New Table( "temp_date",
  Add Rows( 2 ),
  New Column( "date1",
     Numeric,
     "Continuous",
     // specify how the value appears in the data table
     Format( "Format Pattern", "<Month> <D>, <YYYY>", 35 ),
     // specify how the value appears when you edit the value
     Input Format( "Format Pattern", "<YYYY>*<MM>*<DD>" ),
     Set Values( {"2018*10*31", "2018*09*09"} )
  ),
  New Column( "date2",
     Numeric,
     "Continuous",
     Format( "Format Pattern", "<YYYY>*<MM>*<DD>", 10 ),
     Set Values( {"2018*02*05", "2018*07*03"} )
  )
);
```

Note: See *Using JMP* for details about creating a format pattern in a data table rather than in a script.

Currency

JMP displays numbers as currency using the `Format()` function, which uses the following syntax:

```
Format(x,"Currency", <"currency code">, <decimal>, < << Use Locale(Boolean));
```

Where:

- *x* is a column or a number
- *"currency code"* is an International Standards Organization (ISO) 4217 code
- *decimal* is the number of decimal places

To illustrate the `Format` function:

```
Format( 12345.6, "Currency", "GBP", 3 );
  "£12,345.600"
```

By default, the operating system locale is used for the currency if the currency code symbol is omitted. For example, running the following script in a Japanese operating system formats the number with the yen symbol.

```
Format( 12345.6, "Currency", 3);
  "¥12,345.600"
```

To ignore the computer's locale and use a period for the decimal separator, include the `Use Locale(0)` argument:

```
Format( 12345.6, "Currency", 3, <<Use Locale( 0 ) );
```

Tip: If you change the region when JMP is open, restart JMP for the `Use Locale` setting to work properly.

If the currency code is not supported by JMP, the currency code string appears before the number.

```
Format( 12345.6, "Currency", "BBD", 3 );
  "BBD 12,345.600"
```

Table 6.6 lists the currencies supported in JMP.

Table 6.6 Currencies Supported in JMP

Code	Currency	Code	Currency	Code	Currency
AUD	Australian dollar	ILS	Israeli new shekel	RUB	Russian ruble
BRL	Brazilian real	INR	Indian rupee	SEK	Swedish krone

Table 6.6 Currencies Supported in JMP *(Continued)*

Code	Currency	Code	Currency	Code	Currency
CAD	Canadian dollar	JPY	Japanese yen	SGD	Singapore dollar
CHF	Swiss franc	KRW	South Korean won	THB	Thai baht
CNY	Chinese yuan	MXN	Mexican peso	TRY	New Turkish lira
COP	Colombian peso	MYR	Malaysian ringgit	TWD	New Taiwan dollar
DKK	Danish krone	NOK	Norwegian krone	USD	US dollar
EUR	Euro	NZD	New Zealand dollar	ZAR	South African rand
GBP	British pound	PHP	Philippine peso		
HKD	Hong Kong dollar	PLN	Polish zloty		

For more information about how to format currency, see "Formatting Numbers".

Hexadecimal and BLOB Functions

JMP can also handle binary (large) objects, commonly called BLOBs. The functions below convert between hexadecimal values, numbers, characters, and BLOBs. Some of the functions are covered in more detail following Table 6.7.

See the *JSL Syntax Reference* for more information.

Table 6.7 Hexadecimal and BLOB Functions

Syntax	Explanation

`Hex("text")`
`Hex("num")`
`Hex("blob")`
	Returns the hexadecimal codes for the characters in *text*, *number*, or *blob*.
	`Char To Hex` is an alias.

`Hex To BLOB("hexstring")`	Returns a BLOB representation of the hexadecimal code supplied as a quoted string.

Table 6.7 Hexadecimal and BLOB Functions *(Continued)*

Syntax	Explanation
`Hex To Char("hexstring", "encoding")`	Returns a character string that corresponds to the hexadecimal code supplied as a quoted string. The default encoding for the character string is `utf-8`. `utf-16le`, `utf-16be`, `us-ascii`, `iso-8859-1`, `ascii~hex`, `shift-jis`, and `euc-jp` are also supported.
`Hex to Number`	Returns the number that corresponds to the hexadecimal code suppled as a quoted string.
`Char To BLOB("string")` `Char To BLOB("string", "encoding")`	Converts a character string into a binary (BLOB). The default encoding for the blob is `utf-8`. `utf-16le`, `utf-16be`, `us-ascii`, `iso-8859-1`, `ascii~hex`, `shift-jis`, and `euc-jp` are also supported.
`BLOB To Char("blob")` `BLOB To Char("blob", "encoding")`	Converts binary data to a character string. The default encoding for the character string is `utf-8`. `utf-16le`, `utf-16be`, `us-ascii`, `iso-8859-1`, `ascii~hex`, `shift-jis`, and `euc-jp` are also supported.
`BLOB Peek("blob", offset, length)`	Returns a new BLOB that is a subset of the given BLOB that is *length* bytes long and begins at the *offset*. Note that the offset is 0-based.
`Matrix to BLOB(matrix, type, bytesEach, endian)`	Makes a BLOB from a matrix by converting the matrix elements to 1-byte, 2-byte, or 4-byte signed or unsigned integers or 4-byte or 8-byte floating point numbers.
`BLOB to Matrix(blob, "type", bytes, "endian", <nCols>)`	Creates a matrix by converting each byte in the `blob` to numbers.

`Hex(string)` returns the hexadecimal codes for each character in the argument. For example,

```
Hex( "Abc" );
```

returns

```
"416263"
```

since 41, 62, and 63 are the hexadecimal codes (in ASCII) for “A”, “b”, and “c”.

`Hex to Char(string)` converts hexadecimal to characters. The resulting character string might not be valid display characters. All the characters must be in pairs, in the ranges 0-9, A-Z, and a-z. Blanks and commas are allowed, and skipped. For example,

```
Hex To Char( "4142" );
```

returns

```
"AB"
```

since 41 and 42 are the hexadecimal equivalents of "A" and "B".

`Hex` and `Hex To Char` are inverses of each other, so

```
Hex To Char( Hex( "Abc" ) );
```

returns

```
"Abc"
```

`Hex To BLOB(string)` takes a string of hexadecimal codes and converts it to a binary object.

```
a = Hex To BLOB( "6A6B6C" );
Show( a );
   a = Char To BLOB("jkl", "ascii~hex")
```

`BLOB Peek(blob,offset,length)` extracts bytes as defined by the arguments from a blob.

```
b = BLOB Peek( a, 1, 2 );
Show( b );
   b = Char To BLOB("kl", "ascii~hex")
b = BLOB Peek( a, 0, 2 );
Show( b );
   b = Char To BLOB("jk", "ascii~hex")
b = BLOB Peek( a, 2 );
Show( b );
   b = Char To BLOB("l", "ascii~hex")
```

`Hex(blob)` converts a blob into hexadecimal.

```
c = Hex( a );
Show( c );
   c = "6A6B6C"
d = Hex To Char( c );
Show( d );
   d = "jkl"
```

`Concat(blob1,blob2)` or `blob1 || blob2` concatenates two blobs.

```
e = Hex To BLOB( "6D6E6F" );
Show( e );
f = a||e;
Show( f );
   e = Char To BLOB("mno", "ascii~hex")
   f = Char To BLOB("jklmno", "ascii~hex")
```

`Length(blob)` returns the number of bytes in a blob.

```
g = Length( f );
```

```
Show( g );
   g = 6
```

Note: When blobs are listed in the log, they are shown with the constructor function `Char To BLOB("...")`.

Any hex code outside the ASCII range (`space` to }, or hex 20 - 7D) is encoded as the three-character sequence [~][hexdigit][hexdigit]. For example,

```
h = Hex To BLOB( "19207D7E" );
Show( h );
i = Hex( h );
Show( i );
   h = Char To BLOB("~19 }~7E", "ascii~hex")
   i = "19207D7E"
```

`Char To BLOB(string)` creates a blob from a string, converting ~hex codes.
`BLOB To Char(blob)` creates a string with ~hex codes to indicate non-visible and non-ASCII codes.

Work with Character Functions

This section shows how to use some of the more complex character functions that are described in the *JSL Syntax Reference*.

Concat

In the `Concat` function, expressions yielding names are treated like character strings, but globals that have the name values are evaluated. The following example demonstrates that if you have a stored name value, you need to either use `Char` before storing it in a global, or `Name Expr` on the global name.

```
n = {abc};
c = n[1] || "def";
Show( c );
   "abcdef"

m = Expr( mno );
c = m || "xyz";
Show( c );
   Name Unresolved: mno in access or evaluation of 'mno' , mno/*###*/
// The /*###*/ characters indicate the location of an error in the script
   execution.

m = Expr( mno );
```

```
c = Name Expr( m ) || "xyz";
Show( c );
   "mnoxyz"

m = Char( Expr( mno ) );
c = m || "xyz";
Show( c );
   "mnoxyz"
```

`Concat Items()` converts a list of string expressions into a single string, with each item separated by a delimiter. If unspecified, the delimiter is a blank. Its syntax is

```
resultString = Concat Items ({list of strings}, <"delimiter string">);
```

For example,

```
a = {"ABC", "DEF", "HIJ"};
result = Concat Items(a, "/");
```

returns

```
"ABC/DEF/HIJ"
```

Alternatively,

```
result = Concat Items( a );
```

returns

```
"ABC DEF HIJ"
```

Munger

`Munger` works many different ways, depending on what you specify for its arguments:

```
Munger(string, offset, find | length, <replace>);
```

Table 6.8 Munger behaviors for various types of arguments

***Find*, *length*, and *replace* arguments**	**Example**
If you specify a string as the *find* and specify no *replace* string, `Munger` returns the position (after *offset*) of the first occurrence *find* string.	`Munger( "the quick brown fox", 1, "quick" );` `5`
If you specify a positive integer as the *length* and specify no *replace* string, `Munger` returns the characters from *offset* to *offset* + *length*.	`Munger( "the quick brown fox", 1, 5 );` `"the q"`

Table 6.8 Munger behaviors for various types of arguments *(Continued)*

***Find*, *length*, and *replace* arguments**	**Example**
If you specify a string as the *find* and specify a *replace* string, Munger replaces the first occurrence after *offset* of *text* with *replace*.	`Munger( "the quick brown fox", 1, "quick", "fast" );` `"the fast brown fox"`
If you specify a positive integer as the *length* and specify a *replace* string, Munger replaces the characters from *offset* to *offset* + *length* with *replace*.	`Munger( "the quick brown fox", 1, 5, "fast");` `"fastuick brown fox"`

If you specify a positive integer as the *length*, and *offset* + *length* exceeds the length of *text*, Munger either returns *text* from *offset* to the end or replaces that portion of *text* with the *replace* string, if it exists.	`Munger( "the quick brown fox", 5, 25);`
	`"quick brown fox"`
	`Munger( "the quick brown fox", 5, 25, "fast" );`
	`"the fast"`

If you specify zero as the *length* and specify no *replace* string, Munger returns a blank string.	`Munger( "the quick brown fox", 1, 0 );` `""`
If you specify zero as the *length* and specify a *replace* string, the string is inserted before the *offset* position.	`Munger( "the quick brown fox", 1, 0, "see " );` `"see the quick brown fox"`
If you specify a negative integer as the *length* value and specify no *replace* string, Munger returns all characters from the offset to the end of the string.	`Munger( "the quick brown fox", 5, -5 );` `"quick brown fox"`
If you specify a negative integer for *length* and specify a *replace* string, Munger replaces all characters from the offset to the end with the *replace* string.	`Munger( "the quick brown fox", 5, -5, "fast" );` `"the fast"`

Repeat

The `Repeat()` function makes copies of its first argument into a result. The second (and sometimes a third) argument is the number of repeats, where 1 means a single copy.

If the first argument evaluates to a character value or list, the result is that many copies.

```
Repeat( "abc", 2 );
```

```
   "abcabc"
Repeat( {"A"}, 2 );
   {"A","A"}
Repeat( {1, 2, 3}, 2 );
   {1,2,3,1,2,3}
```

If the first argument evaluates to a number or matrix, the result is a matrix. The second argument is the number of row repeats, and a third argument can specify the number of column repeats. If only two arguments are specified, the number of column repeats is 1.

```
Repeat( [1 2, 3 4], 2, 3 );
   [  1 2 1 2 1 2,
      3 4 3 4 3 4,
      1 2 1 2 1 2,
      3 4 3 4 3 4]
Repeat( 9, 2, 3 );
   [  9 9 9,
      9 9 9]
```

The `repeat` function is compatible with the function of the same name in the SAS/IML language, but is incompatible with the SAS character DATA step function, which repeats one more time than this function.

Substitute and Substitute Into

The `Substitute()` function returns a copy of a string with a replacement expression. The `Substitute Into()` function changes the original string with a replacement expression.

For example, consider the following script:

```
str1 = str2 = "All things considered";

// str3 holds the result of the Substitute, and str1 is not changed
str3 = Substitute( str1, "All", "Some" );

/* Substitute Into returns nothing, so str4 is missing,
and str2 is changed and now holds the result of Substitute Into */
str4 = Substitute Into( str2, "All", "Some" );
Show( str1, str2, str3, str4 );
```

This script returns the following output:

```
str1 = "All things considered";
str2 = "Some things considered";
str3 = "Some things considered";
str4 = .;
```

Regular Expressions

A regular expression is a specification of a pattern frequently used to clean up or extract pieces of data. You can search for a pattern and replace it with a different string or extract specific parts of the string. Define the pattern in the `Regex()` or `Regex Match()` function.

- "Regex"
- "Regex Match"
- "Special Characters in Regular Expressions"
- "Escaped Characters in Regular Expressions"
- "Greedy and Reluctant Regular Expressions"
- "Backreferences and Capturing Groups"
- "Lookaround Assertions"

Regex

`Regex()` searches for a *`pattern`* within a *`source`* string and returns a string. It simply identifies a pattern in a string or transforms a string into another string.

```
Regex(source, pattern, (<replacement string>, <GLOBALREPLACE>), <format>,
  <IGNORECASE>);
```

`IGNORECASE` disregards case. `GLOBALREPLACE` repeats the match until the entire string is processed. `format` is a backreference to the matched group. `Regex()` returns missing if the match fails.

Example of Matching a String

`bus|car` is the regular expression (in quotation marks because it is a string). The expression means match "bus" or "car".

```
sentence = "I took the bus to work.";
vehicle = Regex( sentence, "bus|car" );
  "bus"
```

Examples of Replacing a String

The third optional argument in `Regex()` is a specification of the result string. The default value, `\0`, is a backreference to everything that was matched by the regular expression. In the preceding example, the word "bus" is matched in `sentence`. The default third argument, `\0`, replaces the entire sentence with "bus".

A more interesting variation uses parentheses to create additional backreferences.

```
sentence = "I took the bus to work.";
Regex( sentence, "(.*) bus (.*)", "\1 car \2" );
   "I took the car to work."
```

The `(.*)` before and after `bus` are part of the regular expression. The parentheses create a capturing group. The `.` matches any character. The * matches zero or more of the previous expression. As a result, the first parenthesis pair matches everything before `bus`, and the second parenthesis pair matches everything after `bus`. The third argument, `\1 car \2`, reassembles the text; it leaves out `bus` and substitutes `car`.

See "Backreferences and Capturing Groups".

Example of Global Replacement

`GLOBALREPLACE` changes the behavior of `Regex()`. If the match succeeds, the entire source string is returned with substitutions made for each place where the pattern matches. If there are no matches, an unchanged source string is returned.

```
sentence = "I took the red bus followed by the blue bus to get to work
   today.";
Regex( sentence, "bus", "car", GLOBALREPLACE);
   "I took the red car followed by the blue car to get to work today."
```

You can also use backreferences. This example starts with a different sentence.

```
sentence = "I took the red bus followed by the blue car to get to work
   today.";
Regex(
   sentence,
   "(\w*) (bus|car)",
   "bicycle (not \2) that was \1",
   GLOBALREPLACE
);
   "I took the bicycle (not bus) that was red followed by the bicycle (not car)
      that was blue to get to work today."
```

The `\w*` matches zero or more word characters and becomes backreference 1 because of the parentheses. `bus|car` becomes backreference 2 because of the parentheses. The third argument, `bicycle (not \2) that was \1`, describes how to build the substitution text for the part of the source text that was matched.

Notice how the backreferences can be used to swap data positions. This might be useful for swapping the position of first names and last names.

Regex Match

`Regex Match()` returns an empty list with zero elements if the match fails. If the match succeeds, the first list is the text of the entire match (backreference `0`). The second list is the text that matches backreference 1, and so on.

```
Regex Match(source, pattern, <replacement>|<MATCHCASE>, <NULL>)
```

Unlike `Regex()`, `Regex Match()` is case insensitive. Include `MATCHCASE` for a case-sensitive match. Include `NULL` if you want to match case but there is no replacement text.

Example of Parsing Name-Value Pairs

The following example parses pairs of names and values.

```
Regex Match(
   "person=Fred id=77 friend= favorite=tea",
   "(\w+)=(\S*) (\w+)=(\S*) (\w+)=(\S*) (\w+)=(\S*)"
);
   {"person=Fred id=77 friend= favorite=tea", "person", "Fred", "id", "77",
      "friend", "", "favorite", "tea"}
```

The `\w+` matches one or more word characters. The `\S*` matches zero or more characters that are not spaces. In the resulting JSL list, the field names (`person, id, friend, favorite`) and their corresponding values (`Fred, 77, "", tea`) are separate strings.

If the first argument to `Regex Match()` is a variable and a third argument specifies the replacement value, the matched text is replaced in the variable.

Comparing Regex and Regex Match

`Regex()` and `Regex Match()` match a pattern in a given string but return different results. To transforms your string into another string, use `Regex()`. To identify the substrings that match specific parts of the pattern, use `Regex Match()`.

This example shows the efficiency of `Regex Match()` compared to `Regex()`. The `source` is a list of six strings. The goal is to extract portions of those six strings into the subject, verb, and object columns of a data table.

Figure 6.4 Final Data Table

	subject	verb	object
1	cat	ate	chicken
2	dog	chased	cat
3	ralph	like	mary
4	girl	pets	dog
5	cat	chased	dog

```
source = {"the cat ate the chicken", "the dog chased the cat", "did ralph like
   mary", "the girl pets the dog",
"these words are strange", "the cat was chased by the dog"};

dt = New Table( "English 101", // create the data table
   New Column( "subject", character ),
   New Column( "verb", character ),
   New Column( "object", character )
);
// iterate through the strings in the list
For( i = 1, i <= N Items( source ), i++,

   // assign the result of each match to matchList
   matchList = Regex Match(
      source[i],

   /* scan each string, match zero or more characters
   and one item in each group */

   ".*?(cat|dog|ralph|girl).*?(ate|chased|like|pets).*?(chicken|cat|mary|dog)
   "
   );

/* If matchList has zero items (string 5), don't add a row
to the table. Put each matched string in separate
data table cells. */
   If( N Items( matchList ) > 0,
      dt << Add Rows( 1 );
      dt:subject = matchList[2]; // match the first open parenthesis
      dt:verb = matchList[3]; // match the second open parenthesis
      dt:object = matchList[4]; // match the third open parenthesis
   );
);
```

`Regex Match()` returns `{"the cat was chased by the dog", "cat", "chased", "dog"}` in a single try with each answer in a separate string. Compare this example to a similar one using `Regex()`, which returns one answer at a time and builds the final string using backreferences.

```
For( i = 1, i <= N Items( source ), i++,
   s = Regex( source[i],
   ".*?(cat|dog|ralph|girl).*?(ate|chased|like|pets).*?(chicken|cat|mary|dog)
   ", "\1" ); // match an item in the first group
   v = Regex( source[i],
   ".*?(cat|dog|ralph|girl).*?(ate|chased|like|pets).*?(chicken|cat|mary|dog)
   ", "\2" ); // match an item in the second group
```

```
      o = Regex( source[i],
      ".*?(cat|dog|ralph|girl).*?(ate|chased|like|pets).*?(chicken|cat|mary|dog)
      ", "\3" ); // match an item in the third group
      If( !Is Missing( s ) & !Is Missing( v ) & !Is Missing( o ),
         dt << Add Rows( 1 );
         dt:subject = s; // return the match for \1
         dt:verb = v; // return the match for \2
         dt:object = o; // return the match for \3
      );
   );
```

Backreferences are discussed in "Backreferences and Capturing Groups".

Special Characters in Regular Expressions

Special characters are commonly used in regular expressions. The period is a special character that matches one instance of the specified character. It must be escaped with a backslash to be interpreted as a period. In the following expression, the period is replaced with an exclamation point.

```
Regex( "Bicycling makes traveling to work fun.", "\.", "!", GLOBALREPLACE );
  "Bicycling makes traveling to work fun!"
```

Table 6.9 describes the special characters and provides examples.

Table 6.9 Special Characters in Regular Expressions

\	• Precedes a literal character. `<\/a>` interprets the forward slash literally in the end HTML anchor tag. • Precedes an escape sequence. `\n` matches a newline character.
^	Matches the beginning of a string, not including the newline character. `^apple` matches "apple" at the beginning of a string.
$	Matches the end of a string, not including the newline character. `apple$` matches "apple" at the end of a string.
.	Matches any single character including a newline character. `.apple` matches any single character and then "apple".
|	Represents a logical OR to separate alternative values. `(apple|orange|banana)` matches "apple", "orange", or "banana".

Table 6.9 Special Characters in Regular Expressions *(Continued)*

?	Matches zero or one instance. `apple (pie)?` matches one or more instances of "pie".
*	Matches zero or more instances.
+	Matches one or more instances.
( )	Encloses a sub-expression. `(apple|orange|banana)` matches "apple", "orange", or "banana". `^(\w+)` matches the beginning of a line and then one or more word characters.
[ ]	Encloses an expression that matches set of characters. `[\s]` matches a whitespace character or a digit. `[a-z0-9]` matches "a" through "z" and numbers "0" through "9".
{ }	Encloses an expression that represents repetition. `apple{3}` repeats three times. `apple{3,}` repeats at least three times as many times as possible. `apple{3, 10}` repeats three times but no more than 10 times. Append a question mark to indicate repeating as few times as possible. For example, `apple{3,}?` repeats at least three times as few times as possible.

Escaped Characters in Regular Expressions

The backslash in a regular expression precedes a literal character. You also escape certain letters that represent common character classes, such as `\w` for a word character or `\s` for a space. The following example matches word characters (alphanumeric and underscores) and spaces.

```
Regex(
   "Are you there, Alice?, asked Jerry.", // source
   "(here|there).+(\w+).+(said|asked)(\s)(\w+)\." ); // regular expression
   "there, Alice?, asked Jerry."
```

`(here|there).+`	Matches "there", a comma, and a space.
`(\w+)`	Matches "Alice".

`.+`	Matches "?, ".
`(said|asked)(\s)`	Matches "asked" followed by a space. Without the space, the match would end here; "asked" is followed by a space in the source string.
`(\w+)\.`	Matches "Jerry" and a period.

Table 6.10 describes the escaped characters supported in JMP. `\C`, `\G`, `\X`, and `\z` are not supported.

Table 6.10 Escaped Characters

`\\`	single backslash
`\A`	start of a string
`\b`	word boundary. The zero-length string between `\w` and `\W` or `\W` and `\w`.
`\B`	not at a word boundary
`\c`*X*	ASCII control character
`\d`	single digit `[0-9]`
`\D`	single character that is NOT a digit `[^0-9]`
`\E`	stop processing escaped characters
`\l`	match a single lowercase letter `[a-z]`
`\L`	single character that is not lowercase `[^a-z]`
`\Q`	ignore escaped characters until `\E` is found
`\r`	carriage return
`\s`	single whitespace character
`\S`	single character that is NOT white space
`\u`	single uppercase character `[A-Z]`
`\U`	single character that is not uppercase `[^A-Z]`
`\w`	word character `[a-zA-Z0-9_]`
`\W`	single character that is NOT a word character `[^a-zA-Z0-9_]`

Table 6.10 Escaped Characters *(Continued)*

\x*00*-\x*FF*	hexadecimal character
\x{*0000*}-\x{*FFFF*}	Unicode code point
\Z	end of a string before the line break

Greedy and Reluctant Regular Expressions

The ?, *, and + operators are greedy by default. They match as many of the preceding character as possible. The ? operator makes them reluctant; ?? matches 0, then 1 if needed; +? matches 1 and then additional characters; *? matches 0 and then additional characters.

The following example starts at the letter n and compares it to the first \d (digits) in the pattern. No digit matches. Because the pattern does not begin with ^ (start of line), the matcher advances to u. The process repeats until the 3 matches the first \d and the 2 matches the second \d.

```
Regex( "number=32.5", "\d\d" );
   "32"
```

Change the pattern to use the greedy + (match one or more).

```
Regex( "number=324.5", "\d+" );
   "324"
```

The preceding example begins much the same, but as soon as the 3 is found and the \d matches, the + greedily matches the 2 and the 4.

Usually, the greedy behavior makes pattern matching faster because the string is consumed sooner. Sometimes a reluctant behavior is better. Adding the ? after the * or + changes them from greedy to reluctant.

```
Regex( "number=324.5", "\d+?" );
   "3"
```

Here, + requires at least one match of a digit character, but ? changes it from "as many as possible" to "as few as possible". It stopped after the 3 because the pattern was satisfied.

Compare the following results.

Greedy:

```
Regex( "number=324.5", "(\d+)(\d+)\.", "first=\1 second=\2" );
   "first=32 second=4"
```

Reluctant:

```
Regex( "number=324.5", "(\d+?)(\d+?)\.", "first=\1 second=\2" );
   "first=3 second=24"
```

In the greedy example above, the matcher greedily matched 3, 2, and 4 for the first \d+. The matcher then had to give back the 4 so that the second \d+ could match something. The reluctant example followed a different path to get a different answer. Initially, the second value was 2, but the pattern could not match the period to the 4, so the second \d+? reluctantly matched the 4 as well.

Use the Reluctant Match for Speed

The greedy and reluctant matches usually produce the same result but not always. See the previous section. One reason you might need the reluctant match is for speed. Suppose that you have a million-character string that begins "The quick fox…" and you want to find the word before "fox". You might write the following expression and expect \1 to contain "quick".

```
The (.+) fox
```

\1 might contain "quick" eventually, after the .+ grabs the million characters to the end of the string and then gives them up, one at a time, until "fox" is found. If there is more than one "fox", it will be the last fox, not this one. To speed it up and make sure we get the first fox, add the ? operator.

```
The (.+?) fox
```

The ? advances one character at a time to get past "quick" and find the first "fox". This method is much faster than going too far.

Typically, the + or * operator is applied to a more restrictive expression such as \d* to match a run of digits, and greedy is faster than reluctant.

Aside from the multiple fox possibility, greedy and reluctant eventually get the same answer. Using the right operator speeds up the match. The right one might be greedy, or it might be reluctant. It depends on what is being matched.

The greedy .* finds the last fox after backing up.

```
Regex(
   "The quick fox saw another fox eating grapes",
   "The (.*) fox",
   "\1"
);
   "quick fox saw another"
```

The reluctant .*? stops on the first fox.

```
Regex(
   "The quick fox saw another fox eating grapes",
   "The (.*?) fox",
   "\1"
);
   "quick"
```

The greedy `.*` has to back up a lot. There is no second fox.

```
Regex(
  "The quick fox saw another animal eating grapes",
  "The (.*) fox",
  "\1"
);
  "quick"
```

The greedy word character match is an even better choice for this problem.

```
Regex(
  "The quick fox saw another fox eating grapes",
  "The (\w*) fox",
  "\1"
);
  "quick"
```

Backreferences and Capturing Groups

A regular expression can consist of patterns grouped in parentheses, also known as capturing groups. In `([a-zA-Z])\s([0-9])`, `([a-zA-Z])` is the first capturing group; `([0-9])` is the second capturing group.

Use a backreference to replace the pattern matched by a capturing group. In Perl, these groups are represented by the special variables `$1`, `$2`, `$3`, and so on. (`$1` indicates text matched by the first parenthetical group.) In JMP, use a backslash followed by the group number (`\1`, `\2`, `\3`).

The following example includes a third argument that specifies the replacement text and backreferences.

```
Regex(
  "   Are you there, Alice?, asked Jerry.", // source
  " (here|there).+ (\w+).+(said|asked) (\w+)\.", // regular expression
"  I am \1, \4, replied \2." ); // optional format argument
  "  I am there, Jerry, replied Alice."
```

`" I am \1,`	Creates the text “I am”, a space, and then the first matched pattern, “there”.
`\4,`	Creates the text “Jerry” with the fourth matched pattern `(\w+)`.
`replied \2."`	Creates the text “replied” and a space. Matches “Alice.” with the second matched pattern `(\w+)`.

Lookaround Assertions

Lookaround assertions check for a pattern but do not return that pattern in the results. *Lookaheads* look forward for a pattern. *Lookbehinds* look back for a pattern.

Negative Lookahead Example

Negative lookaheads check for the absence of a pattern before a specific pattern. `?!` indicates a negative lookahead. The following expression matches a comma *not* followed by a number or space and replaces the pattern with a comma and space:

```
Regex( "one,two 1,234 cat,dog,duck fish, and chips, to go",
   ",(?!\d|\s)", ", ", GLOBALREPLACE );
   "one, two 1,234 cat, dog, duck fish, and chips, to go"
```

Positive Lookahead Example

Positive lookaheads check for the presence of a pattern before a specific pattern. `?=` indicates a positive lookahead. The following expression has the same result as the preceding negative lookahead but matches a comma followed by any lowercase character:

```
Regex( "one,two 1,234 cat,dog,duck fish, and chips, to go",
   ",(?=[a-z])", ", ",GLOBALREPLACE );
   "one, two 1,234 cat, dog, duck fish, and chips, to go"
```

Positive Lookbehind Example

In this example, the positive lookbehind regular expression matches the “ssn=” or “salary=” keywords without including the keyword in the matched text. The matched text is the string of characters that consists of zero or more dollar signs, digits, and hyphens.

```
data = "name=bill salary=$5 ssn=123-45-6789 age=13,name=mary salary=$6
   ssn=987-65-4321 age=14";
redacted = Regex(data, "(?<=(ssn=)|(salary=))[$\d-]*", "###", GLOBALREPLACE);
   "name=bill salary=### ssn=### age=13,name=mary salary=### ssn=### age=14"
```

Here is another way to get the same result using a backreference substitution. `((ssn=)|(salary=))` is the capturing group. `"\1"` is the backreference to that group.

```
data = "name=bill salary=$5 ssn=123-45-6789 age=13,name=mary salary=$6
   ssn=987-65-4321 age=14";
redacted = Regex(data, "((ssn=)|(salary=))[$\d-]*", "\1###", GLOBALREPLACE);
   "name=bill salary=### ssn=### age=13,name=mary salary=### ssn=### age=14"
```

Backreferences are discussed in “Backreferences and Capturing Groups”.

Pattern Matching

Pattern matching in JSL is a flexible method for searching and manipulating strings.

You define and use pattern variables just like any JMP variable:

```
i = 3; // a numeric variable
a = "Ralph"; // a character variable
t = textbox("Madge"); // a display box variable
p = ( "this" | "that" ) + patSpan(" ")
     + ( "car" | "bus" ); // a pattern variable
```

When the above statement executes, `p` is assigned a pattern value. The pattern value can be used either to construct another pattern or to perform a pattern match. The `patSpan` function returns a pattern that matches a span of characters specified in the argument; `patSpan("0123456789")` matches runs of digits.

```
p2 = "Take " + p + "."; // using p to build another pattern
If( Pat Match( "Take this bus.", p2 ), // performing a match
   Print( "matches" ),
   Print( "no match" )
);
```

Sometime all you need to know is that the pattern matched the source text, as above. Other times, you might want to know what matched; for example, was it a bus or a car?

```
p = ("this" | "that") + Pat Span( " " ) + ("car" | "bus") >?
vehicleType; // conditional assignment ONLY if pattern matches
If( Pat Match( "Take this bus.", p ),
   // do not use vehicleType in the ELSE because it is not set
   Show( vehicleType ),
   Print( "no match" )
);
```

You could pre-load `vehicleType` with a default value if you do not want to check the outcome of the match with an `if`. The >? conditional assignment operator has two arguments, the first being a pattern and the second a JSL variable. >? constructs a pattern that matches the pattern (first argument) and stores the result of the match in the JSL variable (second argument) after the pattern succeeds. Similarly, >> does not wait for the pattern to succeed. As soon (and as often) as the >> pattern matches, the assignment is performed.

```
findDelimString = Pat Len( 3 ) >> beginDelim + Pat Arb() >? middlePart
+Expr( beginDelim );
testString = "SomeoneSawTheQuickBrownFoxJumpOverTheLazyDog'sBack";
rc = Pat Match( testString, findDelimString, "<<<" || middlePart || ">>>" );
Show( rc, beginDelim, middlePart, testString );
```

The above example shows a third argument in the `patMatch` function: the replacement string. In this case, the replacement is formed from a concatenation (`||` operator) of three strings. One of the three strings, `middlePart`, was extracted from the `testString` by `>?` because the replacement cannot occur unless the pattern match succeeds (`rc == 1`).

Look at the pattern assigned to `findDelimString`. It is a concatenation of 3 patterns. The first is a `>>` operator that matches 3 characters and assigns them to `beginDelim`. The second is a `>?` operator that matches an arbitrary number of characters and, when the entire match succeeds, assigns them to `middlePart`. The last is an unevaluated expression, consisting of whatever string is in `beginDelim` at the time the pattern is executing, *not* at the time the pattern is built. Just like `expr()`, the evaluation of its argument is postponed. That makes the pattern hunt for two identical three letter delimiters of the middle part.

Other pattern functions might be faster and represent the problem that you are trying to solve better than writing a lot of alternatives; for example, `"a"|"b"|"c"` is the same as `Pat Any("abc")`. The equivalent example for `Pat Not Any("abc")` is much harder. Similar to `Pat Span` (above), `Pat Break("0123456789")` matches up to, but not including, the first number.

Here is a pattern that matches numbers with decimals and exponents and signs. It also matches some degenerate cases with no digits; look at the pattern assigned to digits.

```
digits = Pat Span( "0123456789" ) | "";

number = (Pat Any( "+-" ) | "") >? signPart + (digits) >? wholePart + ("."
+digits | "") >? fractionPart + (Pat Any( "eEdD" ) + (Pat Any( "+-" ) | "")
 + digits | "") >? exponentPart;

If( Pat Match( "-123.456e-78", number ),
   Show( signPart, wholePart, fractionPart, exponentPart )
);
```

Parsing Strings in Fixed Fields

Sometimes data is in fixed fields. The `Pat Tab()`, `Pat R Tab()`, `Pat Len()`, `Pat Pos()`, and `Pat R Pos()` functions make it easy to split out the fields in a fixed field string. `Pat Tab()` and `Pat R Tab()` work from the left and right end of the string and take a number as their argument. They succeed by matching forward to the specified tab position. For example:

```
p = Pat Pos(10) + Pat Tab(15);
```

`Pat Pos(10)` matches the null string if it is in position 10. So at match time, the matcher works its way forward to position 10, then `Pat Tab(15)` matches text from the current position (10) forward to position 15. This pattern is equivalent to `Pat Pos(10)+patLen(5)`. Another example:

```
p = Pat Pos(0) + Pat R Tab(0);
```

This example matches the entire string, from 0 characters from the start to 0 characters from the end. the `Pat Rem()` function takes no argument and is shorthand for `Pat R Tab(0)`; it means the remainder of the string. Pattern matching can also be anchored to the beginning of the string like this:

```
Pat Match( "now is the time", Pat Len(15) + Pat R Pos(0), NULL, ANCHOR );
```

The above pattern uses `NULL` rather than a replacement value, and `ANCHOR` as an option. Both are uppercase, as shown. `NULL` means that no replacement is done. `ANCHOR` means that the match is anchored to the beginning of the string. The default value is `UNANCHORED`.

Patterns can be built up like this, but this is *not* recursive:

```
p = "a" | "b"; // matches one character
p = p + p; // two characters
p = p + p; // four characters
Pat Match( "babb", Pat Pos(0) + p + Pat R Pos(0) );
```

A recursive pattern refers to its current definition using `Expr()`:

```
p = "<" + Expr(p) + "*" + Expr(p) + ">" | "x";
Pat Match( "<<x*<x*x>>*x>", Pat Pos(0) + p + Pat R Pos(0) );
```

Remember, `expr()` is the procrastination function; when the pattern is assigned to the variable `p`, `expr()` delays evaluating its argument (`p`) until later. In the next statement, `patMatch` performs the pattern match operation, and each time it encounters `expr()`, it looks for the current value of the argument. In this example, the value does not change during the match). So, if `p` is defined in terms of itself, how can this possibly work?

`p` consists of two alternatives. The right hand choice is easy: a single letter x. The left side is harder: <p*p> . Each `p` could be a single letter x, since that is one of the choices `p` could match, or it could be <p*p>. The last few example have used `patPos(0) + ... + patRPos(0)` to make sure the pattern matches the entire source text. Sometimes this is what you want, and sometimes you would rather the pattern match a subtext. If you are experimenting with these examples by changing the source text, you probably want to match the entire string to easily tell what was matched. The result from `Pat Match` is 0 or 1.

This example uses “Left” recursion:

```
x =  Expr(x) + "a" | "b"; // + binds tighter than |
```

If the pattern is used in `FULLSCAN` mode, it eventually uses up all memory as it expands. By default, the `patMatch` function does not use `FULLSCAN`, and makes some assumptions that allow the recursion to stop and the match to succeed. The pattern matches either a “b”, or anything the pattern matches followed by an “a”.

```
rc = Pat Match( "baaaaa", x );
```

Patterns and Case

Unlike regular expressions, pattern matching is case insensitive. To force case sensitivity, you can add the named argument MATCHCASE to either `Pat Match()` or `Regex Match()`. For example:

```
string = "abcABC";

result = Regex Match( string, Pat Regex( "[aBc]+" ) );
Show( string, result );
      / *returns string = "abcABC"
   result = {"abcABC"}*/

result = Regex Match( string, Pat Regex( "[cba]+" ), NULL, MATCHCASE );
Show( string, result );
      /* returns string = "abcABC"
   result = {"abc"}*/
```

Troubleshooting Patterns

When a pattern isn't working the way you expect, use `Log()` to send pieces of the match to the JMP log. The following example shows how to debug pattern matching and rewrite a script to account for the white space between words in the phrase "the quick brown fox".

```
// greedy match for one or more characters except for white space
word = Pat Repeat( Pat Not Any( " " ) );

Pat Match(
   "the quick brown fox",
   word >> a +
   word >> b +
   word >> c +
   word >> d
);

Show( a, b, c, d );
   a = "qu";
   b = "i";
   c = "c";
   d = "k";
```

The output is probably not what you want, given the name of the pattern *word*. None of the words in the pattern were matched.

The `Show( a, b, c, d )` expression is a good first step. If it doesn't tell you how JMP got that answer, try inserting `Log()`:

```
word = Pat Repeat( Pat Not Any( " " ) );
```

```
Pat Match(
   "the quick brown fox",
   (word >> a) >> Log( "a" ) +
   (word >> b) >> Log( "b" ) +
   (word >> c) >> Log( "c" ) +
   (word >> d) >> Log( "d" )
);

Show( a, b, c, d );
   0(a) the (the number 0 is the position of the match)

   0(a) th (the "a" is an aribtrary identifier you supplied to the log)

   2(b) e

   ...
   4(a) quick
   4(a) quic
   ...
```

With this extra information, you can now see into the workings of the match, how the word pattern does, in fact, match a word, at least initially. But then it backs up and tries a smaller amount of the word so that the next word pattern can match some non-whitespace characters. However, there's no provision in this pattern to match the white space between words. And that's exactly what it did (find four consecutive words of one or more non-whitespace characters). Maybe the following script is what you need:

```
word = Pat Repeat( Pat Not Any( " " ) );
gap = Pat Repeat( Pat Any( " " ), 1, 999, GREEDY );

Pat Match(
   "the quick brown fox",
   word >> a + gap +
   word >> b + gap +
   word >> c + gap +
   word >> d
);

Show( a, b, c, d );
   a = "the";
   b = "quick";
   c = "brown";
   d = "fox";
```

Chapter 7

Data Structures

Working with Collections of Data

JSL provides these basic data structures that can hold a variety of data in a single variable:

- A list holds a number of other values, including nested lists and expressions.
- A matrix is a row-by-column table of numbers.
- An associative array maps keys to values, which can be almost any other type of data.

Contents

Lists . . . 193
Evaluate Lists . . . 193
Assignments with Lists . . . 195
Perform Operations in Lists . . . 195
Find the Number of Items in a List . . . 195
Subscripts . . . 195
Locate Items in a List . . . 197
List Operators and Functions . . . 198
Iterate through a List . . . 201
Concatenate Lists . . . 202
Insert a Nested List in an Existing List . . . 203
Index into a List with Another List . . . 203
Send Messages to Lists . . . 203
Matrices . . . 204
Construct Matrices . . . 204
Subscripts . . . 206
Inquiry Functions . . . 210
Comparisons, Range Checks, and Logical Operators . . . 210
Numeric Operations . . . 211
Concatenation . . . 214
Transpose . . . 214
Indexing into a Matrix or List with Another Matrix or List . . . 215
Matrices and Data Tables . . . 216
Matrices and Reports . . . 218
Loc Functions . . . 219
Ranking and Sorting . . . 221
Special Matrices . . . 221
Inverse Matrices and Linear Systems . . . 226
Decompositions and Normalizations . . . 230
Build Your Own Matrix Functions . . . 235
Statistical Examples . . . 236
Associative Arrays . . . 240
Create Associative Arrays . . . 240
Work with Associative Arrays . . . 243
Applications for Associative Arrays . . . 246
Associative Arrays in Graph Theory . . . 248
Associative Arrays in Set Operations . . . 251

Lists

Lists are containers to store items, such as the following:

- numbers
- variables
- character strings
- expressions (for example, assignments or function calls)
- matrices
- nested lists

Create a list in one of the following ways:

- use the `List` function
- use `{ }` curly braces

Examples

Use the `List()` function or curly braces to create a list that includes numbers and variables:

```
x = List(1, 2, b);
x = {1, 2, b};
```

A list can contain text strings, nested lists, and function calls:

```
{"Red", "Green", "Blue", {1, "true"}, sqrt( 2 )};
```

You can place a variable into a list and assign it a value at the same time:

```
x = {a = 1, b = 2};
```

Evaluate Lists

When you run a script that contains a list, a copy of the list is returned. The items inside the list are not evaluated.

```
b = 7;
x = {1, 2, b, Sqrt( 3 )};
Show( x );
   x = {1, 2, b, Sqrt(3)};
```

To evaluate items in a list, use the `Eval List()` function.

```
b = 7;
x = {1, 2, b, Sqrt( 3 )};
c = Eval List( x );
   {1, 2, 7, 1.73205080756888}
```

When using a list of variables that reference lists, you will need to use an Eval() function.

Consider the following example, where you have a variable called fullMonth that is a list of 12 items.

```
::fullMonth = {January, February, March, April, May, June, July, August,
  September, October, November, December};
::abbrevMonth = {Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec};
::dow = {Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday};
::levels = {Very Low, Low, Medium Low, Medium, Medium High, High, Very High};
::feel = {Strongly Disagree, Disagree, Neutral, Indifferent, Agree, Strongly
  Agree};
::rating = {Failing, Unacceptable, Very Poor, Poor, Bad, Acceptable, Average,
  Good, Better, Very Good, Excellent, Best};
::mlist = {::fullMonth, ::abbrevMonth, ::dow, ::levels, ::feel, ::rating};
N Items(::fullMonth); // returns 12 because fullMonth has 12 items.
N Items(::mlist[1]); /* returns this message: "N Items() argument must be a
  list" because ::mlist[1] is a variable (fullMonth) and fullMonth is a
  list.*/
```

When you add the Eval() function:

```
N Items( Eval( ::mlist[1] ) );
  12
/* returns 12 because::mlist[1] holds ::fullMonth, and evaluating it
returns ::fullMonth's list.*/
```

In a loop, you need the Eval() function in the nested loop to evaluate the variable's contents. For example:

```
For( g = 1, g <= N Items( ::mlist ), g++,
    For( i = 1, i <= N Items(::mlist[g]), i++,
        Show( ::mlist[g][i] )
    )
);
// returns the message "N Items() argument must be a list."
```

To fix the problem, add the Eval() function:

```
::mlist = {::fullMonth, ::abbrevMonth, ::dow, ::levels, ::feel, ::rating};
For( g = 1, g <= N Items( ::mlist ), g++,
    For( i = 1, i <= N Items( Eval( ::mlist[g] ) ), i++,
        Show( ::mlist[g][i] )
    )
);
```

Assignments with Lists

Create a list to assign values to variables.

Examples

```
{a, b, c} = {1, 2, 3}; // assign 1 to a, 2 to b, and 3 to c
{a, b, c}--; // decrement a, b, and c
{{a}, {b, c}}++; // increment a, b, and c by 1
mylist = {1, log( 2 ), e()^pi(), height[40]}; // store the expressions
```

Perform Operations in Lists

In lists, you can perform operations.

```
a = {{1, 2}, 3, {4, 5}};
b = {{10, 20}, 30, {40, 50}};
c = a + b;
   c = {{11, 22}, 33, {44, 55}}
```

Find the Number of Items in a List

To determine the number of items in a list, use the `N Items()` function.

```
x = {1, 2, y, Sqrt( 3 ), {a, b, 3}};
N = N Items( x );
Show( n );
   n = 5;
```

Subscripts

Subscripts extract specified items from a list. Use a list as a subscript to return multiple items from a list.

Note: JSL starts counting from 1, so the first element in a list is [1], not [0] as in some other languages.

Examples

List a contains four items.

```
a = {"bob", 4, [1, 2, 3], {x, y, z}};
Show( a[1] );
   a[1] = "bob";
Show( a[{1, 3}] );
```

```
   a[{1, 3}] = {"bob", [1, 2, 3]};
a[2] = 5; // assign 5 to the second list item
```

You can also use subscripts to select or change items in a list:

```
Show( a );
   a = {"bob", 5, [1, 2, 3], {x, y, z}};
c = {1, 2, 3};
c[{1, 2}] = {4, 4};
// c[{1, 2}] = 4 produces the same result
Show( c );
   c = {4, 4, 3};
```

When you have assignments or functions in a list, you can use a quoted name for the subscript to extract the value.

```
x={sqrt( 4 ), log( 3 )};
xx= {a = 1, b = 3, c = 5};
x["sqrt"];
   4
xx["b"];
   3
```

The name must be in quotation marks, or else JMP tries to evaluate it and use its value. The following example shows the values of the second item in the list, rather than the value of `a` in the list.

```
a = 2;
Show( xx[a] );
   xx[a] = b = 3;
```

Notes:

- Multiple left-side subscripts (for example, `a[i][j] = value` where `a` contains a list of things that are subscriptable) are allowed in the following circumstances:
  - Each level except the outermost level must be a list. So, in the example above, `a` must be a list but `a[i]` can be anything subscriptable.
  - Each subscript except the last must be a number. So, in the example above, `i` must be a number, but `j` could be a matrix or list of indices.
- Subscripting can be done to any level of nesting, such as the following:

  ```
  a[i][j][k][l][m][n] = 99;
  ```

Note: To get a value in the matrix of an expression column, use double subscripts, as in `:Discrim Data Matrix[row()][i]`.

Locate Items in a List

Use the `Loc()` function or the `Contains()` function to find values in a list:

```
Loc( list, value );
Contains( list, value );
```

`Loc()` and `Contains()` return the positions of the values. `Loc()` returns the results in a matrix, and `Contains()` returns the results as a number.

Notes:

- The `Loc` function returns each occurrence of a repeated value. `Contains()` returns only the first occurrence of a repeated value.
- If the value is not found, the `Loc` function returns an empty matrix and `Contains()` returns a zero.
- To assess whether an item is in a list, use `Loc()` and `Contains()` with `>0`. A returned value of zero means that the item is not in the list. A returned value of 1 means that the item is in the list at least once.

Note: For more information about matrix manipulation and a description of the equivalent `Loc()` command for matrices, see "Matrices".

Examples

```
nameList = {"Katie", "Louise", "Jane", "Jane"};
numList = {2, 4, 6, 8, 8};
```

Search for the value `"Katie"` in the `nameList`:

```
Loc( nameList, "Katie" );
   [1]
Contains( nameList, "Katie" );
   1
```

Search for the value `"Erin"` in the `nameList`:

```
Loc( nameList, "Erin" );
   []
Contains( nameList, "Erin" );
   0
```

Search for the number `8` in the `numList`:

```
Loc( numList, 8 );
   [4, 5]
Contains( numList, 8 );
   4
```

Find out if the number `5` exists in the `numList`:

```
NRow( Loc( numList, 5 ) ) > 0;
   0
Contains( numList, 5 ) > 0;
   0
```

List Operators and Functions

Table 7.1 describes the list operators and their syntax.

Table 7.1 List Operators

Operator and Function		Syntax	Explanation
	As List()	As List(*matrix*)	Returns the matrix as a list. A matrix with multiple columns is returned as a list of lists, one list per row.
= += -= *= /= ++ --	Assign() Add To() SubtractTo() MultiplyTo() DivideTo() Post Increment() Post Decrement()	{*list*} = {*list*} {*list*} += *value* {*list*} -= {*list*} ...	If the target of an assignment operator is a list and the value to be assigned is a list, then it assigns item by item. The ultimate values in the left list must be L-values (in other words, names capable of being assigned values). **Notes:** • If you want to test equality of lists, use ==, not =. • JMP does not have pre-increment or pre-decrement operators. Instead, use the Add To() operator, which is (+=), or the Subtract To() operator, which is (-=).
||=	Concat To()	Concat To(*list1*, *list2*, ...)	Inserts the second and subsequent lists at the end of the first list.
||	Concat()	Concat(*list1*, *list2*, ...);	Returns a copy of the first list with any additional lists inserted at after it.

Table 7.1 List Operators *(Continued)*

Operator and Function	Syntax	Explanation
Eval List()	Eval List(*list*)	Returns a list of the evaluated expressions inside list. See "Evaluate Lists".
Insert Into()	Insert Into(*list*, *x*, <*i*>)	Inserts a new item (*x*) into the *list* at the given position (*i*). If *i* is not given, the item is added to the end of the list. This function does change the original list.
Insert()	*list* = Insert(*list*, *x*, <*i*>)	Returns a copy of the *list* with a new item (*x*) inserted into the *list* at the given position (*i*). If *i* is not given, the item is added to the end of the list. This function does not change the original list.
Is List()	Is List(*arg*)	Returns true (1) if *arg* is a classical list (in other words, one that would result from the construction by List(*items*) or {*items*}) and returns false (0) otherwise. An empty list is still a list, so IsList({ }) returns true. If miss=., then IsList(miss) returns false, not missing.
{ } List	List(*a*, *b*, *c*) {*a*, *b*, *c*}	Constructs a list from a set of items. An item can be any expression, including other lists. Items must be separated by commas. Text should either be enclosed in double quotation marks (" ") or stored in a variable and called as that variable.
N Items	N Items(*list*)	Returns the number of elements in the *list* specified. Can be assigned to a variable.

Table 7.1 List Operators *(Continued)*

Operator and Function	Syntax	Explanation
`Remove From()`	`Remove From(list, <i>, <n>)`	Deletes *n* items from the *list*, starting from the indicated position (*i*). If *n* is omitted, the item at *i* is deleted. If *n* and *i* are omitted, the item at the end is removed. This function does change the original list.
`Remove()`	`Remove(list, <i>, <n>)`	Returns a copy of the *list* with the *n* items deleted, starting from the indicated position (*i*). If *n* is omitted, the item at *i* is deleted. If *n* and *i* are omitted, the item at the end is removed. This function does not change the original list.
`Reverse Into()`	`Reverse Into(list)`	Reverses the order of the items in the *list*. This function does change the original list.
`Reverse()`	`Reverse(list)`	Returns a copy of the *list* with the items in reverse order. This function does not change the original list.
`Shift Into()`	`Shift Into(list, <n>)`	Shifts *n* items from the front of the *list* to the end of the *list*. If *n* is omitted, the first item is moved to the end of the list. This function does change the original list.
`Shift()`	`Shift(list, <n>)`	Returns a copy of the *list* with *n* items shifted from the front of the list to the end of the *list*. If *n* is omitted, the first item is moved to the end of the list. This function does not change the original list.
`Sort Ascending()`	`Sort Ascending(list)`	Returns a copy of the *list* sorted in ascending order. This function does not change the original list.

Table 7.1 List Operators *(Continued)*

Operator and Function		Syntax	Explanation
	`Sort Descending()`	`Sort Descending(`*`list`*`)`	Returns a copy of the *`list`* sorted in descending order. This function does not change the original list.
	`Sort List Into()`	`Sort List Into(`*`list`*`)`	Sorts the *`list`* in ascending order. This function does change the original list.
	`Sort List()`	`Sort List(`*`list`*`)`	Returns a copy of the *`list`* sorted in ascending order. This function does not change the original list.
`[ ]`	`Subscript()`	*`list`*`[`*`i`*`]` *`x`* `=` *`list`*`[`*`i`*`]` *`list`*`[`*`i`*`]` `=` *`value`* *`a`*`[`*`b`*`,` *`c`*`]` `Subscript(`*`a`*`,` *`b`*`,` *`c`*`)`	Subscripts for lists extract the i^{th} item from the *`list`*. Subscripts can in turn be lists or matrices.
	`Substitute()`	`Substitute(`*`list`*`,` *`pattExpr1`*`,` *`replExpr1`*`, ...)`	Returns a copy of a string, list, or expression, replacing instances of each pattern expression with the corresponding replacement expression. See "Substitute and Substitute Into".
	`Substitute Into()`	`Substitute Into(`*`list`*`,` *`pattExpr1`*`,` *`replExpr1`*`, ...)`	Changes a string, list, or expression, replacing instances of each pattern expression with the corresponding replacement expression. **Note:** The *`list`* or expression must be a variable. See "Substitute and Substitute Into".

Iterate through a List

Iterate through a list to do something with each value or look for a particular value. The following script looks at each item in the list. If the item in the list is less than or equal to 10, it is replaced with its square.

```
x = {2, 12, 8, 5, 18, 25};
n = N Items( x );
For( i = 1, i <= n, i++,
```

```
    If( x[i] <= 10,
       x[i] = x[i] ^ 2
    )
);
Show( x );
   x = {4, 12, 64, 25, 18, 25};
```

You can use `Loc()` to locate the items in the new list that are equal to 25:

```
Loc( x, 25 );
   [4, 6] // The fourth and sixth items in the list are equal to 25.
```

Concatenate Lists

Join two or more lists into one list using `Concat()` or the `||` operator. The original lists are not changed.

The following example uses `Concat()` to join lists *a* and *b*:

```
a = {1, 2};
b = {7, 8, 9};
Concat( a, b );
   {1, 2, 7, 8, 9}
```

The following example joins the same lists using the `||` operator:

```
{1, 2} || {7, 8, 9}
   {1, 2, 7, 8, 9}
```

Lists of different types can be concatenated (for example, lists that contain character strings and numbers).

```
d = {"apples", "bananas"};
e = {"oranges", "grapes"};
f = {1, 2, 3};
Concat( d, e, f);
   {"apples", "bananas", "oranges", "grapes", 1, 2, 3}
```

Join two or more lists and replace the first list with the combined list using `Concat to()` or the `||=` operator. The following example performs concatenation in place using `Concat to()`:

```
d = {"apples", "bananas"};
e = {"peaches", "pears"};
Concat to(d,e);
Show( d );
   d = {"apples", "bananas", "peaches", "pears"}
```

The following example joins the same lists using the `||=` operator:

```
d = {"apples", "bananas"};
e = {"peaches", "pears"};
d||=e;
```

```
Show( d );
  d = {"apples", "bananas", "peaches", "pears"}
```

Insert a Nested List in an Existing List

To insert a nested list in an existing list, specify the placement of the list:

```
list2 = {{"a", "b"}, {"c", "d"}};
list2[3] = {"apple", "banana"}; // specify a list as the third item
Show( list2 );
  list2 = {{"a", "b"}, {"c", "d"}, {"apple", "banana"}};
```

If you do not know the number of items in a list, use `N Items()`. In this example, an empty list is inserted at the end of the list.

```
list1 = {"a", "b"};
list1[N Items( list1 ) + 1] = {};
Show( list1 );
  list1 = {"a", "b", {}};
```

Index into a List with Another List

Indexing into a a list with another list is a powerful technique for making scripts fast and small. See “Indexing into a Matrix or List with Another Matrix or List”.

Send Messages to Lists

You can send a single message to a list instead of having to send individual messages. Suppose that you want to create a Bivariate report with a By variable. The script returns a list of the bivariate objects, one for each level of the By variable. Send a single `Fit Line` message to the list to fit a line to all of the Bivariate reports.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
/* return a list of bivariate objects:
{Bivariate[], Bivariate[], Bivariate[], Bivariate[], Bivariate[],
Bivariate[]}; */
biv = dt << Bivariate( X( dt:weight ), Y( dt:height ), By( dt:age ) );

biv << Fit Line; /* send the Fit Line message to
each Bivariate object in the list */
```

Troubleshooting Messages Sent to a List

Any message that is sent to a list is sent to each object in the list. If the message saves a file, it's performing the save for each object in the list to the *same file*. This means that only the last object is actually saved in a file.

Suppose that you have a list of three control charts and you send a message (such as `Save Interactive HTML`) to the list. The message is sent to each member of the list in turn. First, it saves an HTML page with the first control chart; then it saves a second HTML page with the same name that contains the second control chart; and then it saves a third HTML page with the same name that contains only the third control chart. You end up with a page that contains only the third control chart.

The solution is to send the message to a single container higher up the tree that contains all control charts. To do this, rewrite the following line of code:

```
obj << Save Interactive HTML( "Control Charts.html" );
```

`Top Parent` treats the first object as the single container:

```
tp = obj[1] << Top Parent();
tp << Save Interactive HTML( "Control Charts.html" );
```

Matrices

A matrix is a rectangular array of numbers that are arranged in rows and columns. Use matrices to store numbers and perform calculations on those numbers using matrix algebra.

Note the following for this section:

- Matrices are represented with an uppercase bold variable (for example, **A**).
- A matrix with one row or one column is a *vector* (or more specifically, a *row vector* or a *column vector* respectively).
- For clarity, we represent matrices that are vectors with lowercase bold letters (such as **x**).
- A *scalar* is a numeric value that is not in a matrix.

Construct Matrices

Note the following when creating matrices:

- Place matrix literals in square brackets. `[...]`
- Matrix values can contain decimal points, can be positive or negative, and can be in scientific notation.

- Separate items across a column with blank spaces. You can use any number of blank spaces.
- Separate rows with a comma.

For constructing more advanced matrices, see "Special Matrices".

Examples

Create matrix **A** with 3 rows and 2 columns:

```
A = [1 2, 3 4, 5 6];
```

R is a row vector and **C** is a column vector:

```
R = [10 12 14];
C = [11, 13, 15];
```

B is a 1-by-1 matrix, or a matrix with one row and one column:

```
B = [20];
```

E is an empty matrix:

```
E = [];
```

The following examples show two ways of creating empty matrices:

```
J( 0,4 );
  [](0,4)

[]( 5,0 )
  [](5,0)
```

Specifying the number of rows and columns in an empty matrix is optional. JMP creates the matrix as necessary. For example, the following script defines an empty matrix, iterates over each row in the data table, and returns the heights and weights in a matrix.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
data = []( 0, 2 );
For Each Row( data |/= Matrix( {{dt:height, dt:weight}} ) );
Show( data );
  data = [59 95, 61 123, 55 74,...]
```

A script can return an empty matrix. In Big Class.jmp, the following expression looks for rows in which age equals 8, finds none, and returns an empty matrix:

```
a = dt << Get Rows Where( age == 8 );
Show( a );
  a = [](0,1);
```

Construct Matrices from Lists

If you want to convert lists into a matrix, use the `Matrix()` function. A single list is converted into a column vector. Two or more lists are converted into rows.

Create a column vector from a single list:

```
A = Matrix( {1, 2, 3} );
   [1,2,3]
```

Create a matrix from a list of lists. Each list is a row in the matrix.

```
A = Matrix( {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}} );
   [1 2 3,
   4 5 6,
   7 8 9]
```

Construct Matrices from Expressions

To construct matrices from expressions, use `Matrix()`. Elements must be expressions that resolve to numbers.

```
A = Matrix( {4 * 5, 8 ^ 2, Sqrt( 9 )} );
   [20, 64, 3]
```

Subscripts

Use the subscript operator (`[ ]`) to pick out elements or submatrices from matrices. The `Subscript()` function is usually written as a bracket notation after the matrix to be subscripted, with arguments for rows and columns.

Single Element

The expression `A[i,j]` extracts the element in row `i`, column `j`, returning a scalar number. The equivalent functional form is `Subscript(A,i,j)`.

```
P = [1 2 3, 4 5 6, 7 8 9];
P[2, 3]; // returns 6
Subscript( P, 2, 3 ); // returns 6
```

Assign the value that is in the third row and the first column in the matrix **A** (which is 5) to the variable `test`.

```
A = [1 2, 3 4, 5 6];
test = A[3, 1];
Show( test );
   test = 5;
```

Matrix or List Subscripts

To extract a sub-matrix, use matrix or list subscripts. The result is a matrix of the selected rows and columns. The following expressions select the second and third rows with the first and third columns.

```
P = [1 2 3, 4 5 6, 7 8 9];
P[[2 3],[1 3]]; // matrix subscripts
P[{2, 3},{1, 3}]; // list subscripts
```

Both of these methods provide the following output:

```
[4 6,
7 9]
```

Single Subscripts

A single subscript addresses matrices as if all the rows were connected end-to-end in a single row. This makes the double subscript `A[i,j]` the same as the single subscript `A[(i-1)*ncol(A)+j]`.

Examples

```
Q = [2 4 6, 8 10 12, 14 16 18];
Q[8]; // same as Q[3,2]
  16
```

The following examples all return the column vector [10, 14, 18]:

```
Q = [2 4 6, 8 10 12, 14 16 18];
Q[{5, 7, 9}];
Q[[5 7 9]];
ii = [5 7 9];
Q[ii];
ii = {5, 7, 9};
Q[ii];
Subscript( Q, ii );
```

This script returns the values 1 through 9 from the matrix **P** in order:

```
P = [1 2 3, 4 5 6, 7 8 9];
For( i = 1, i <= 3, i++,
  For( j = 1, j <= 3, j++,
    Show( P[i, j] )
  )
);
```

Delete Rows and Columns

Deleting rows and columns is accomplished by assigning an empty matrix to that row or column.

```
A[k, 0] = [];  // deletes the kth row
A[0, k] = [];  // deletes the kth column
```

Select Whole Rows or Columns

A subscript argument of zero selects all rows or columns.

```
P = [1 2 3, 4 5 6, 7 8 9];
```

Select column 2:

```
P[0, 2];
   [2, 5, 8]
```

Select columns 3 and 2:

```
P[0, [3, 2]];
   [3 2, 6 5, 9 8]
```

Select row 3:

```
P[3, 0];
   [7 8 9]
```

Select rows 2 and 3:

```
P[[2, 3], 0];
   [4 5 6, 7 8 9]
```

Select all columns and rows (the whole matrix):

```
P[0, 0];
   [1 2 3, 4 5 6, 7 8 9]
```

Assignment through Subscripts

You can change values in matrices using subscripts. The subscripts can be single indices, matrices or lists of indices, or the zero index representing all rows or columns. The number of selected rows and columns for the insertion must either match the dimension of the inserted argument, or the argument can be inserted repeatedly into the indexed positions.

Examples

Change the value in row 2, column 3 to 99:

```
P = [1 2 3, 4 5 6, 7 8 9];
P[2, 3] = 99;
Show( P );
```

```
P=[1 2 3, 4 5 99, 7 8 9]
```

Change the values in four locations:

```
P = [1 2 3, 4 5 6, 7 8 9];
P[[1 2], [2 3]] = [66 77, 88 99];
Show( P );
   P=[1 66 77, 4 88 99, 7 8 9]
```

Change three values in one column:

```
P = [1 2 3, 4 5 6, 7 8 9];
P[0, 2] = [11, 22, 33];
Show( P );
   P=[1 11 3, 4 22 6, 7 33 9]
```

Change three values in one row:

```
P = [1 2 3, 4 5 6, 7 8 9];
P[3, 0] = [100 102 104];
Show( P );
   P=[1 2 3, 4 5 6, 100 102 104]
```

Change all the values in one row to the same value:

```
P = [1 2 3, 4 5 6, 7 8 9];
P[2, 0] = 99;
Show( P );
   P=[1 2 3, 99 99 99, 7 8 9]
```

Operator Assignment

You can use operator assignments (such as +=) on matrices or subscripts of matrices. For example, the following statement adds 1 to the *i* - *j*th element of the matrix:

```
P = [1 2 3, 4 5 6, 7 8 9];
P[1, 1] += 1;
Show( P );
   P=[2 2 3,
      4 5 6,
      7 8 9];

P[1, 1] += 1;
Show( P );
   P=[3 2 3,
   4 5 6,7 8 9];
```

Ranges of Rows or Columns

If you are working with a range of subscripts, use the `Index()` function or `::` to create matrices of ranges.

```
T1 = 1 :: 3; // create the vector [1 2 3]
T2 = 4 :: 6; // create the vector [4 5 6]
T3 = 7 :: 9; // create the vector [7 8 9]
T = T1 |/ T2 |/ T3; // concatenate the vectors into a single matrix
T[1 :: 3, 2 :: 3]; // refer to rows 1 through 3, columns 2 and 3
   [2 3, 5 6, 8 9]
T[Index( 1, 3 ), Index( 2, 3 )]; // equivalent Index function
   [2 3, 5 6, 8 9]
```

Inquiry Functions

The `NCol()` and `NRow()` functions return the number of columns and rows in a matrix (or data table), respectively:

```
NCol( [1 2 3, 4 5 6] ); // returns 3 (for 3 columns)
NRow( [1 2 3, 4 5 6] ); // returns 2 (for 2 rows)
```

To determine whether a value is a matrix, use the `Is Matrix()` function, which returns a 1 if the argument evaluates to a matrix.

```
A = [20, 64, 3];
B = {20, 64, 3};
Is Matrix( A ); // returns 1 for yes
Is Matrix( B ); // returns 0 for no
```

Comparisons, Range Checks, and Logical Operators

JMP's comparison, range check, and logical operators work with matrices and produce matrices of elementwise Boolean results. You can compare conformable matrices.

```
A < B; // less than
A <= B; // less or equal
A > B; // greater than
A >= B; // greater or equal
A == B; // equal to
A != B; // not equal to
A < B < C; // continued comparison (range check)
A | B; // logical OR
A & B; // logical AND
( A & !B ) | ( !A & B ); // exlusive OR
!( A & B ) & ( A | B ); // exlusive OR
```

You can use the `Any()` or `All()` functions to summarize matrix comparison results. `Any()` returns a 1 if any element is nonzero. `All()` returns a 1 if all elements are nonzero.

```
[2 2] == [1 2]; // returns [0 1], therefore:
All( [2 2] == [1 2] ); // returns 0
```

```
Any( [2 2] == [1 2] ); // returns 1
```

`Min()` or `Max()` return the minimum or maximum element from the matrix or matrices given as arguments.

```
A = [1 2 3, 4 5 6, 7 8 9, 10 11 12];
B = [0 1 2, 2 1 0, 0 1 1, 2 0 0];
Min( A ); // returns 1
Max( A ); // returns 12
Min( A, B ); // returns 0
```

Numeric Operations

You can perform numeric operations (such as subtraction, addition, and multiplication) on matrices. Most statistical methods are expressed in compact matrix notation and can be implemented in JSL.

For example, the following expression uses matrix multiplication and inversion to illustrate least squares regression:

$$b = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$

Implement this equation through the following JSL expression:

```
b = Inv( X'*X )*X'*y;
```

Basic Arithmetic

You can perform the following basic arithmetic on matrices:

- addition
- subtraction
- multiplication
- division (multiplying by the inverse)

Note: The standard multiply operator is a matrix multiplier, not an elementwise multiplier.

To perform matrix multiplication, use one of the following methods:

- `*` operator
- `Multiply()` function
- `Matrix Mult()` function

To perform matrix division, use one of the following methods:

- `/` operator

- `Divide()` function

Note the following about matrix multiplication and division:

- Remember that while multiplication or division of scalars is commutative (ab = ba), multiplication or division of matrices is *not* commutative.
- When one of the two elements is a scalar, elementwise multiplication or division is performed.
- To use elementwise multiplication, use `:*` or the `EMult()` function.
- To use elementwise division, use `:/`, or the equivalent `EDiv()` function.

Examples

```
A = [1 2 3, 4 5 6, 7 8 9, 10 11 12];
B = [0 1 2, 2 1 0, 0 1 1, 2 0 0];
C = [1 2 3 4, 4 3 2 1, 0 1 0 1];
D = [0 1 2, 2 1 0, 1 2 0];
```

Matrix addition:

```
R = A + B;
   [1 3 5,
   6 6 6,
   7 9 10,
   12 11 12]
```

Matrix subtraction:

```
R = A - B;
   [1 1 1,
   2 4 6,
   7 7 8,
   8 11 12]
```

Matrix multiplication (inner product of rows of **A** with columns of **C**):

```
R = A * C;
   [9 11 7 9,
   24 29 22 27,
   39 47 37 45,
   54 65 52 63]
```

Matrix division (equivalent to `A*Inverse(D)`):

```
R = A / D;
   [1.5 0.5 0,
   3 2 0,
   4.5 3.5 0,
   6 5 0]
```

Matrix elementwise multiplication:

```
R = A :* B;
```

```
[0 2 6,
8 5 0,
0 8 9,
20 0 0]
```

Matrix scalar multiplication:

```
R = C * 2;
   [2 4 6 8,
   8 6 4 2,
   0 2 0 2]
```

Matrix scalar division:

```
R = C / 2;
   [0.5 1 1.5 2,
   2 1.5 1 0.5,
   0 0.5 0 0.5]
```

Matrix elementwise division (division by zero results in missing values):

```
R = A :/ B;
   [. 2 1.5,
   2 5 .,
   . 8 9,
   5 . .]
```

Numeric (Scalar) Functions on Matrices

Numeric functions work elementwise on matrices. Most of the pure numeric functions can be applied to matrices, resulting in a matrix of results. You can mix conformable matrix and scalar arguments.

Examples of numeric functions include the following:

- `Sqrt()`, `Root()`, `Log()`, `Exp()`, `^ Power()`, `Log10()`
- `Abs()`, `Mod()`, `Floor()`, `Ceiling()`, `Round()`, `Modulo()`
- `Sine,()`, `Cosine()`, `Tangent()`, `ArcSine()`, and other trigonometry functions.
- `Normal Distribution()`, and other probability functions.

Example

```
A = [1 2 3, 4 5 6, 7 8 9, 10 11 12];
B = Sqrt( A ); // elementwise square root
   [1 1.414213562373095 1.732050807568877,
      2 2.23606797749979 2.449489742783178,
      2.645751311064591 2.82842712474619 3,
      3.16227766016838 3.3166247903554 3.464101615137754]
```

Concatenation

The `Concat()` function combines two matrices side by side to form a larger matrix. The number of rows must agree. A double vertical bar (`||`) is the infix operator, equivalent for horizontal concatenation.

```
Identity( 2 ) || J( 2, 3, 4 );
   [1 0 4 4 4, 0 1 4 4 4]

B = [1, 1];
B || Concat( Identity( 2 ), J( 2, 3, 4 ) );
   [1 1 0 4 4 4, 1 0 1 4 4 4]
```

The `VConcat()` function stacks two matrices on top of each other to form a larger matrix. The number of columns must agree. A vertical-bar-slash (`|/`) is the infix operator, equivalent for vertical concatenation.

```
Identity( 2 ) |/ J( 3, 2, 1 );
// or VConcat( Identity( 2 ),J( 3, 2, 1 ) );
   [1 0, 0 1, 1 1, 1 1, 1 1]
```

Both `Concat()` and `VConcat()` support concatenating to empty matrices, scalars, and lists.

```
a=[];
a || [1]; // returns [1]
a || {2}; // returns [2]
a || [3 4 5]; // returns [3 4 5]
```

There are two in place concatenation operators: `||=` and `|/=` . They are equivalent to the `Concat To()` and `V Concat To()` functions, respectively.

- `a || = b` is equivalent to `a = a || b`
- `a |/ = b` is equivalent to `a = a | /b`

Transpose

The `Transpose()` function transposes the rows and columns of a matrix. A back-quote (`` ` ``) is the postfix operator, equivalent to `Transpose()`. In matrix notation, `Transpose()` is expressed as the common prime or superscript-T notation (A′ or A^T).

```
A = [1 2 3, 4 5 6, 7 8 9, 10 11 12];
A`
   [1 4 7 10,
   2 5 8 11,
   3 6 9 12]
Transpose( [1 2, 3 4] );
   [1 3, 2 4]
```

Indexing into a Matrix or List with Another Matrix or List

Indexing into a matrix (or a list) with another matrix (or list) is a powerful technique for making scripts fast and small. JMP 13 and later handles one edge case differently.

Consider the following example of using a matrix as an index into a data column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
tall rows = Loc( :height << Get Values() >= 67 ); // [25, 27, 30, 37, 39, 40]
:height[tall rows]; // [68, 69, 67, 68, 68, 70]
N Rows( :height[tall rows] ); // 6
```

The second line is the key. `:height << Get Values()` returns the values of the height column as a matrix. The comparison `>= 67` returns a matrix of 0s and 1s corresponding to where the expression is false (0) or true (1). The `Loc()` function returns the locations of the 1s within the previous matrix.

In this case, you get [25, 27, 30, 37, 39, 40] as the row numbers where the `height` column is greater than or equal to 67. That's pretty handy by itself, but the real power comes from being able to use that matrix as the index into a column or other matrix as in the third and fourth lines.

The alternative to this technique is to write a `For()` loop to iterate through every row explicitly. JMP still loops through the rows in either case, but the looping is much faster if done internally rather than in JSL. And the JSL is more compact without the explicit loop.

You can do the same thing with a matrix variable instead of a data column reference:

```
m = :height << Get Values();
tall rows = Loc( m >= 67 ); // [25, 27, 30, 37, 39, 40]
m[tall rows]; // [68, 69, 67, 68, 68, 70]
N Rows( m[tall rows] ); // 6
```

Consider the same examples with a different height cutoff (70 instead of 67):

```
tall rows = Loc( :height << Get Values() >= 70 ); // [40]
:height[tall rows]; // [70]
N Rows( :height[tall rows] ); // 1

m = :height << Get Values();
tall rows = Loc( m >= 70 ); // [40]
m[tall rows]; // [70]
N Rows( m[tall rows] ); /* error in JMP 12 and prior versions
otherwise, returns 1 */
```

Notice that `m[tall rows]` returned 70 instead of [70]. In many cases, a 1x1 matrix and a number are treated the same way, but not always. The `N Rows()` function is one example where it makes a difference. The indexing is too aggressively simplifying the 1x1 matrix into a number.

One work-around is to call `Is Matrix()` before using `N Rows()`. An alternative is to concatenate an empty matrix to the result, which creates a matrix whether the source is a number or a matrix:

```
m = m |/ []
```

In JMP 13 and later, the 1x1 matrix is maintained so that the result of the index operation is a matrix whenever the index itself is a matrix. The same principle applies to list indexing. Data table column indexing has always followed this principle.

Matrices and Data Tables

You can move information between a matrix and a JMP data table. You can use matrix algebra to perform calculations on numbers that are stored in JMP data tables, and you can save the results back to JMP data tables.

Move Data into a Matrix from a Data Table

These sections describe how to move data from a data table into a matrix.

Move All Numeric Values

The `Get As Matrix()` function generates a matrix containing all of the numeric values in a data table or column:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
A = dt << Get As Matrix;
   [12 59 95,
   12 61 123,
   12 55 74, ...]
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = Column( "Height" );
A = col << Get As Matrix;
   [59, 61, 55, 66, 52, ...]
```

Move All Numeric Values and Character Columns

The `Get All Columns As Matrix()` function returns the values from all columns of the data table in a matrix, including character columns. Character columns are numbered according to the alphanumeric order, starting at 1.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
A = dt << Get All Columns As Matrix;
   [21 12 1 59 95,
   28 12 1 61 123, ...]
```

Move Only Certain Columns

To get certain columns of a data table, use column list arguments (names or characters).

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
x = dt << Get As Matrix( {"height", "weight"} );
```

or

```
x = dt << Get As Matrix( {height, weight} );
   [59 95, 61 123, 55 74, ...]
```

Currently Selected Rows

To get a matrix of the currently selected row numbers in the data table:

```
dt << Get Selected Rows;
```

Note: If no rows are selected, the output is an empty matrix.

Find Rows Where

To see a matrix of row numbers where the expression is true:

```
dt << Get Rows Where( expression );
```

For example, the following script returns the row numbers where the sex is male (M):

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
A = dt << Get Rows Where( Sex == "M" );
   [6, 7, 8, 12, 13, 14, ...]
```

Move Data into a Data Table from a Matrix

This section describes how to move data from a matrix into a data table.

Move a Column Vector

The `Set Values()` function copies values from a column vector into an existing data table column:

```
col << Set Values( x );
```

`col` is a reference to the data table column, and **x** is a column vector.

For example, the following script creates a new column called test and copies the values of vector **x** into the test column:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "test" );
col = Column( "test" );
x = 1::40;
```

```
col << Set Values( x );
```

Move All Matrix Values

The `Set Matrix()` function copies values from a matrix into an existing data table, making new rows and new columns as needed to store the values of the matrix. The new columns are named Col1, Col2, and so on.

```
dt = New Table( "B" );
dt << Set Matrix([1 2 3 4 5, 6 7 8 9 10]);
```

This script creates a new data table called B containing two rows and five columns.

To create a new data table from a matrix argument, use the `As Table(matrix)` command. The columns are named Col1, Col2, and so on. For example, the following script creates a new data table containing the values of **A**:

```
A = [1 2 3, 4 5 6, 7 8 9, 10 11 12];
dt = As Table( A );
```

Matrices and Reports

You can extract matrices of values from reports. First, you need to locate the items that you want to extract. This information is in the tree structure of the report.

Run the following script to create a table of parameter estimates in a Bivariate report:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" ) ;
biv = dt << Bivariate( X( height ), Y( weight ), Fit Line );
```

Now, open the tree structure to identify which item contains the parameter estimates:

- Right-click a gray disclosure icon and select **Edit > Show Tree Structure**.

  The parameter estimates are contained in `NumberColBox(13)`. Continue with the script:

  ```
  colBox = Report( biv )[Number Col Box( 13 )];
  beta = colBox << Get As Matrix;
    [-127.145248610915, 3.711354893859555]
  ```

Notes:

- When a variable contains a reference to a table box, `Get As Matrix()` creates a matrix **A** that contains the values from all numeric columns in the table:

  ```
  A = tableBox << Get As Matrix;
  ```

- When a variable contains a reference to a numeric column in a report table, `Get As Matrix()` creates a matrix **A** as a column vector of values from that column.

  ```
  A = colBox << Get As Matrix;
  ```

Loc Functions

The Loc(), Loc Nonmissing(), Loc Min(), Loc Max(), and Loc Sorted() functions all return matrices that show positions of certain values in a matrix.

Loc

The Loc() function creates a matrix of positions that locate where **A** is true (nonzero and nonmissing).

```
A = [0 1 . 3 4 0];
B = [2 0 0 2 5 6];
```

The following example returns the indices for the values of **A** that are nonmissing and nonzero.

```
I = Loc( A );
   [2, 4, 5]
```

The following example returns the indices for the values of **A** that are less than the corresponding values of **B**. Note that the two matrices must have the same number of rows and columns.

```
I = Loc( A < B );
   [1, 5, 6]
```

The following script replaces all values less than 4 in **A** with 0.

```
A = [0 1 0 3 4 0];
A[Loc( A < 4 )] = 0;
Show( A );
   A = [0 0 0 0 4 0];
```

The following script returns an empty matrix:

```
Loc( [2 3 4 5 6] > 7 );
   [](0, 1)
```

Loc Nonmissing

The Loc Nonmissing() function returns a vector of row numbers in a matrix that do not contain any missing values. For example,

```
A = [1 2 3, 4 . 6, 7 8 ., 8 7 6];
Loc Nonmissing( A );
   [1, 4]
```

Loc Min and Loc Max

The `Loc Min()` and `Loc Max()` functions return the position of the first occurrence of the minimum and maximum elements of a matrix. Elements of a matrix are numbered consecutively, starting in the first row, first column, and proceeding left to right.

```
A = [1 2 2, 2 4 4, 1 1 1];
B = [6, 12, 9];
Show( Loc Max( A ) );
Show( Loc Min( B ) );
   Loc Max(A) = 5;
   Loc Min(B) = 1;
```

Loc Sorted

The `Loc Sorted()` function is mainly used to identify the interval that a number lies within. The function returns the position of the highest value in **A** that is less than or equal to the value in **B**. The resulting vector contains an item for each element in **B**.

```
A = [10 20 30 40];
B = [35];
Loc Sorted( A, B );
   [3]
A = [10 20 40];
B = [35 5 45 20];
Loc Sorted( A, B );
   [2, 1, 3, 2]
```

Here's an example that defines a list of strings. `Loc Sorted()` identifies the string at the seventh interval and puts it in a list.

```
{"No Fat", "Some Fat", "Low Fat", "Normal Fat", "Medium Fat", "High Fat"}
[Loc Sorted( [0, 1, 5, 10, 20, 25], 7)]
   {"Low Fat"}
```

To convert the returned value to a string, include the subscript [1] at the end of the expression:

```
{"No Fat", "Some Fat", "Low Fat", "Normal Fat", "Medium Fat", "High Fat"}
[Loc Sorted( [0, 1, 5, 10, 20, 25], 7)][1]
   "Low Fat"
```

Notes:

- **A** must be sorted in ascending order.
- The returned values are always 1 or greater. If the element in **B** is smaller than all of the elements in **A**, then the function returns a value of 1. If the element in **B** is greater than all of the elements in **A**, then the function returns *n*, where *n* is the number of elements in **A**.

Ranking and Sorting

The `Rank()` function returns the positions of the numbers in a vector or list, as if the numbers were sorted from lowest to highest.

```
E = [1 -2 3 -4 0 5 1 8 -7];
R = Rank( E );
   [9, 4, 2, 5, 7, 1, 3, 6, 8]
```

If **E** were sorted from lowest to highest, the first number would be -7. The position of -7 in **E** is 9.

The original matrix **E** can then be sorted using the matrix **R** as subscripts to **E**.

```
sortedE = E[R];
   [-7,- 4, -2, 0, 1, 1, 3, 5, 8]
```

The `Ranking Tie()` function returns ranks for the values in a vector or list, with ranks for ties averaged. Similarly, `Ranking()` returns ranks for the values in a vector or list, but the ties are ranked arbitrarily.

```
E = [1 -2 3 -4 0 5 1 8 -7];
Ranking Tie( E );
   [5.5, 3, 7, 2, 4, 8, 5.5, 9, 1]

E = [1 -2 3 -4 0 5 1 8 -7];
Ranking( E );
   [5, 3, 7, 2, 4, 8, 6, 9, 1]
```

The `Sort Ascending()` and `Sort Descending()` functions sort vectors.

```
E = [1 -2 3 -4 0 5 1 8 -7];
Sort Ascending( E );
   [-7 -4 -2 0 1 1 3 5 8]

E = [1 -2 3 -4 0 5 1 8 -7];
Sort Descending( E );
   [8 5 3 1 1 0 -2 -4 -7]
```

If the argument is not a vector or list, an error message is generated.

Special Matrices

Construct an Identity Matrix

The `Identity()` function constructs an identity matrix of the dimension that you specify. An identity matrix is a square matrix of zeros except for a diagonal of ones. The only argument specifies the dimension.

```
Identity( 3 );
```

```
[1 0 0,
0 1 0,
0 0 1]
```

Construct a Matrix with Specific Values

The `J()` function constructs a matrix with the number of rows and columns that you specify as the first two arguments, whose elements are all the third argument, for example:

```
J( 3, 4, 5 );
   [5 5 5 5,
   5 5 5 5,
   5 5 5 5]
J( 3, 4, Random Normal() ); // your results will differ
   [0.407709113182904 1.67359154091978 1.00412665221308 0.240885679837327,
   -0.557848036549455 -0.620833861982722 0.877166783247633 1.50413740148892,
   -2.09920574748608 -0.154797501010655 0.0463943433032137 0.0640418263933316]
```

Create a Diagonal Matrix

The `Diag()` function creates a diagonal matrix from a square matrix (having an equal number of rows and columns) or a vector. A diagonal matrix is a square matrix whose nondiagonal elements are zero.

```
D = [1 -1 1];
Diag( D );
   [1 0 0,
   0 -1 0,
   0 0 1]
Diag([1, 2, 3, 4]);
   [1 0 0 0,
   0 2 0 0,
   0 0 3 0,
   0 0 0 4]

A = [1 2,3 4];
f = [5];
D = Diag( A, f );
   [1 2 0
   3 4 0
   0 0 5]
```

In the third example, at first glance, not all of the nondiagonal elements are zero. Using matrix notation, the matrix can be expressed as follows:

```
[A 0,
0` f]
```

Where `A` and `f` are the matrices from the example, and **`0`** is a column vector of zeros.

Create a Column Vector from Diagonal Elements

The `VecDiag()` function creates a column vector from the diagonal elements of a matrix.

```
v = Vec Diag(
[1 0 0 1, 5 3 7 1, 9 9 8 8, 1 5 4 3]);
   [1, 3, 8, 3]
```

Calculate Diagonal Quadratic Forms

The `Vec Quadratic()` function calculates the hats in regression that go into the standard errors of prediction or the Mahalanobis or T2 statistics for outlier distances. `Vec Quadratic(Sym, X)` is equivalent to calculating `Vec Diag(X*Sym*X`)`.The first argument is a symmetric matrix, usually an inverse covariance matrix. The second argument is a rectangular matrix with the same number of columns as the symmetric matrix argument.

Return the Sum of Diagonal Elements

The `Trace()` function returns the sum of the diagonal elements for a square matrix.

```
D = [0 1 2, 2 1 0, 1 2 0];
Trace( D ); // returns 1
```

Generate a Row Vector of Integers

The `Index()` function generates a row vector of integers from the first argument to the last argument. A double colon `::` is the equivalent infix operator.

```
6::10;
   [6 7 8 9 10]
Index( 1, 5 );
   [1 2 3 4 5]
```

The optional *`increment`* argument changes the default increment of +1.

```
Index( 0.1, 0.4, 0.1 );
   [0.1, 0.2, 0.3, 0.4]
```

The increment can also be negative.

```
Index( 6, 0, -2 );
   [6, 4, 2, 0]
```

The default value of the increment is 1, or -1 if the first argument is higher than the second.

Reshape a Matrix

The `Shape()` function reshapes an existing matrix across rows to be the specified dimensions. The following example changes the 3x4 matrix `a` into a 12x1 matrix:

```
a = [1 1 1, 2 2 2, 3 3 3, 4 4 4];
Shape( a, 12, 1 );
   [1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4]
```

Summarize Columns of a Matrix

There are several functions that return a row vector based on summary values for each column in a matrix.

```
mymatrix = [11 22, 33 44, 55 66];
V Max( mymatrix ); // returns the maximum of each column
   [55 66]
V Min( mymatrix ); // returns the minimum of each column
   [11 22]
V Mean( mymatrix ); // returns the mean of each column
   [33 44]
V Median( mymatrix ); // returns the median of each column
   [33 44]
V Quantile( mymatrix, 0.75 ); // returns the 0.75 quantile of each column
   [55 66]
V Sum( mymatrix ); // returns the sum of each column
   [99 132]
V Std( mymatrix ); // returns the standard deviations of each column
   [22 22]
```

Create Design Matrices

The `Design()` function creates a matrix of design columns for a vector or list. There is one column for each unique value in the vector or list. The design columns have the elements 0 and 1. For example, **x** below has values 1, 2, and 3, then the design matrix has a column for 1s, a column for 2s, and a column for 3s. Each row of the matrix has a 1 in the column representing that row's value. So, the first row (1) has a 1 in the 1s column (the first column) and 0s elsewhere; the second row (2) has a 1 in the 2's column and 0s elsewhere; and so on.

```
x = [1, 2, 3, 2, 1];
Design( x );
   [1 0 0,
   0 1 0,
   0 0 1,
   0 1 0,
   1 0 0]
```

A variation is the `Design Nom()` or `Design F()` function, which removes the last column and subtracts it from the others. Therefore, the elements of `Design Nom()` or `Design F()` matrices are 0, 1, and –1. And the `Design Nom()` or `Design F()` matrix has one less column than the vector or list has unique values. This function makes full-rank versions of design matrices for effects.

```
x = [1, 2, 3, 2, 1];
Design Nom( x );
   [1 0, 0 1, -1 -1, 0 1, 1 0]
```

`Design Nom()` is further demonstrated in the "ANOVA Example".

To facilitate ordinal factor coding, use the `Design Ord()` function. This function produces a full-rank coding with one less column than the number of unique values in the vector or list. The row for the lowest value in the vector or list is all zeros. Each succeeding value adds an additional 1 to the row of the design matrix.

```
x = [1, 2, 3, 4, 5, 6];
Design Ord( x );
   [0 0 0 0 0,
   1 0 0 0 0,
   1 1 0 0 0,
   1 1 1 0 0,
   1 1 1 1 0,
   1 1 1 1 1]
```

`Design()`, `Design Nom()`, and `Design Ord()` support a second argument that specifies the levels to be looked up and their order. This feature allows design matrices to be created one row at a time.

- `Design( values, levels )` creates a design matrix of indicator columns.
- `DesignNom( values, levels )` creates a full-rank design matrix of indicator columns.

Notes:

- The *values* argument can be a single element or a matrix or list of elements.
- The *levels* argument can be a list or matrix of levels to be looked up.
- The result has the same number of rows as there are elements in the *values* argument.
- The result always has the same number of columns as there are items in the *levels* argument. In the case of `Design Nom()` and `Design Ord()`, there is one less column than the number of items in the *levels* argument.
- If a value is not found, the whole row is zero.

Examples

```
Design( 20, [10 20 30] );
   [0 1 0]
Design( 30, [10 20 30] );
```

```
   [0 0 1]
Design Nom( 20, [10 20 30] );
   [0 1]
Design Nom( 30, [10 20 30] );
   [-1 -1]
Design Ord( 20, [10 20 30] );
   [1 0]
Design( [20, 10, 30, 20], [10 20 30] );
   [0 1 0,
   1 0 0,
   0 0 1,
   0 1 0]
Design({"b", "a", "c", "b"}, {"a", "b", "c"});
   [0 1 0,
   1 0 0,
   0 0 1,
   0 1 0]
```

Find the Direct Product

The `Direct Product()` function finds the direct product (or Kronecker product) of two square matrices. The direct product of a *m* x *m* matrix and a *n* x *n* matrix is the *mn* x *mn* matrix whose elements are the products of numbers, one from **A** and one from **B**.

```
G = [1 2, 3 5];
H = [2 3, 5 7];
Direct Product( G, H );
   [2 3 4 6,
   5 7 10 14,
   6 9 10 15,
   15 21 25 35]
```

The `H Direct Product()` function finds the row-by-row direct product of two matrices with the same number of rows.

```
H Direct Product( G, H );
   [2 3 4 6, 15 21 25 35]
```

`HDirect Product()` is useful for constructing the design matrix columns for interactions.

```
XA = Design Nom( A );
XB = Design Nom( B );
XAB = HDirect Product( XA, XB );
X = J( NRow( A ), 1 ) || XA || XB || XAB;
```

Inverse Matrices and Linear Systems

JMP has the following functions for computing inverse matrices: `Inverse()`, `GInverse()`, and `Sweep()`. The `Solve()` function is used for solving linear systems of equations.

Inverse or Inv

The `Inverse()` function returns the matrix inverse for the square, nonsingular matrix argument. `Inverse()` can be abbreviated `Inv`. For a matrix **A**, the matrix product of **A** and `Inverse`(**A**) (often denoted $\mathbf{A}(\mathbf{A}^{-1})$) returns the identity matrix.

```
A = [5 6, 7 8];
AInv = Inv( A );
A*A Inv;
   [1 0,0 1]

A = [1 2, 3 4];
AInv = Inverse( A );
A*AInv;
   [1 1.110223025e-16,0 1]
```

Note: There can be small discrepancies in the results due to floating point limitations, as illustrated in the second example.

GInverse

The (Moore-Penrose) generalized inverse of a matrix **A** is any matrix **G** that satisfies the following conditions:

AGA = **A**

GAG = **G**

(**AG**)′ = **AG**

(**GA**)′ = **GA**

The `GInverse()` function accepts any matrix, including non-square ones, and uses singular-value decomposition to calculate the Moore-Penrose generalized inverse. The generalized inverse can be useful when inverting a matrix that is not full rank. Consider the following system of equations:

$$\begin{aligned} x + 2y + 2z &= 6 \\ 2x + 4y + 4z &= 12 \\ x + y + z &= 1 \end{aligned}$$

Find the solution to this system using the following script:

```
A = [1 2 2, 2 4 4, 1 1 1];
B = [6, 12, 1];
Show( GInverse( A ) * B );
   GInverse(A) * B = [-4, 2.5, 2.5];
```

Solve

The `Solve()` function solves a system of linear equations. `Solve()` finds the vector **x** so that $x = A^{-1}b$ where *A* equals a square, nonsingular matrix and *b* is a vector. The matrix **A** and the vector **b** must have the same number of rows. `Solve(A,b)` is the same as `Inverse(A)*b`.

```
A = [1 -4 2, 3 3 2, 0 4 -1];
b = [1, 2, 1];
x = Solve( A, b );
   [-16.9999999999999, 4.99999999999998, 18.9999999999999]
A*x;
   [1, 2, 0.999999999999997]
```

Note: There can be small discrepancies in the results due to floating point limitations, as illustrated in the example.

Sweep

The `Sweep()` function inverts parts (or *pivots*) of a square matrix. If you sequence through all of the pivots, you are left with the matrix inverse. Normally the matrix must be positive definite (or negative definite) so that the diagonal pivots never go to zero. `Sweep()` does not check whether the matrix is positive definite. If the matrix is *not* positive definite, then it still works, as long as no zero pivot diagonals are encountered. If zero (or near-zero) diagonal pivots are encountered on a full sweep, then the result is a g2 generalized inverse if the zero pivot row and column are zeroed.

About the Sweep Function

Suppose matrix **E** consists of smaller matrix partitions, **A**, **B**, **C**, and **D**:

$$E = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

The syntax for `Sweep()` appears as follows:

```
Sweep( E, [...] );
```

[...] indicates partition A.

This produces the matrix result equivalent to the following:

$$\begin{bmatrix} A^{-1} & A^{-1}B \\ -CA^{-1} & D - CA^{-1}B \end{bmatrix}$$

Notes:

- The submatrix in the **A** position becomes the inverse.
- The submatrix in the **B** position becomes the solution to **A**x = **B**.
- The submatrix in the **C** position becomes the solution to x**A** = **C**.

Use of the Sweep Function

`Sweep()` is sequential and reversible:

- `A = Sweep( A, {i,j} )` is the same as `A = Sweep( Sweep( A, i ), j )`. It is sequential.
- `A = Sweep( Sweep( A, i ), i )` restores **A** to its original values. It is reversible.

If you have a cross-product matrix partitioned as follows:

$$C = \begin{bmatrix} X'X & X'y \\ y'X & y'y \end{bmatrix}$$

Then after sweeping through the indices of **X′X**, the result appears as follows:

$$\begin{bmatrix} (X'X)^{-1} & (X'X)^{-1}X'y \\ -yX(X'X)^{-1} & y'y - y'X(X'X)^{-1}X'y \end{bmatrix}$$

The partitions are recognizable to statisticians as follows:

- the least squares estimates for the model Y = Xb + e in the upper right
- the sum of squared errors in the lower right
- a matrix proportional to the covariance of the estimates in the upper left

The `Sweep` function is useful in computing the partial solutions needed for stepwise regression.

The *`index`* argument is a vector that lists the rows (or equivalently the columns) on which you want to sweep the matrix. For example, if **E** is a 4x4 matrix, to sweep on all four rows to get $\mathbf{E}^{-1}$ requires these commands:

```
E = [ 5 4 1 1, 4 5 1 1, 1 1 4 2, 1 1 2 4];
Sweep( E, [1, 2, 3, 4] );
   [0.56 -0.44 -0.02 -0.02,
   -0.44 0.56 -0.02 -0.02,
   -0.02 -0.02 0.34 -0.16,
   -0.02 -0.02 -0.16 0.34]
Inverse( E ); // notice that these results are the same
   [0.56 -0.44 -0.02 -0.02,
```

```
-0.44 0.56 -0.02 -0.02,
-0.02 -0.02 0.34 -0.16,
-0.02 -0.02 -0.16 0.34]
```

Note: For a tutorial on `Sweep()`, and its relation to the Gauss-Jordan method of matrix inversion, see Goodnight, J.H. (1979) "A tutorial on the SWEEP operator." *The American Statistician*, August 1979, Vol. 33, No. 3. pp. 149–58.

`Sweep()` is further demonstrated in the "ANOVA Example".

Determinant

The `Det()` function returns the determinant of a square matrix. The determinant of a 2 × 2 matrix is the difference of the diagonal products, as demonstrated below. Determinants for $n \times n$ matrices are defined recursively as a weighted sum of determinants for $(n - 1) \times (n - 1)$ matrices. For a matrix to have an inverse, the determinant must be nonzero.

$$\begin{vmatrix} 1 & 2 \\ 3 & 5 \end{vmatrix} = (1 \cdot 5) - (3 \cdot 2) = -1$$

```
F = [1 2, 3 5];
Det( F );
   -1
```

Decompositions and Normalizations

This section contains functions that calculate eigenvalues and eigenvectors and functions that decompose matrices.

Eigenvalues

The `Eigen()` function performs eigenvalue decomposition of a symmetric matrix. Eigenvalue decompositions are used in many statistical techniques, most notably in principal components and canonical correlation, where the transformation associated with the largest eigenvalues are transformations that maximize variances.

`Eigen()` returns a list of matrices. The first matrix in the returned list is a column vector of eigenvalues; the second matrix contains eigenvectors as the columns.

```
A = [ 5 4 1 1, 4 5 1 1, 1 1 4 2, 1 1 2 4];
Eigen( A );
   {[10, 5, 2, 1][0.632455532033676 - 0.316227766016838 - 2.77555756156289e-16
      -0.707106781186547, 0.632455532033676 - 0.316227766016837 -
      1.66533453693773e-16 0.707106781186547, 0.316227766016838
      0.632455532033676 0.707106781186548 0, 0.316227766016837
      0.632455532033676 - 0.707106781186547 0]}
```

Since the function returns a list of matrices, you might want to assign it to a list of two global variables. That way, the column vector of eigenvalues is assigned to one variable, and the matrix of eigenvectors is assigned to another variable:

```
{evals, evecs} = Eigen( A );
```

For some $n \times n$ matrix **A**, eigenvalue decomposition finds all λ (lambda) and vectors **x**, so that the equation Ax = λx has a nonzero solution **x**. The λ's are called eigenvalues, and the corresponding **x** vectors are called eigenvectors. This is equivalent to solving (A - λI)**x** = 0. You can reconstruct **A** from eigenvalues and eigenvectors by a statement like the following:

```
newA = evecs * Diag( evals ) * evecs`;
  [5.000000000000001 4 1 1,
  4 5 1 1,
  1 1 4 2,
  1 1 2 4]
```

Note the following about eigenvalues and eigenvectors:

- The eigenvector matrices are orthonormal, such that the inverse is the transpose: $E'E = EE' = I$.
- Eigenvectors are uniquely determined only if the eigenvalues are unique.
- Zero eigenvalues correspond to singular matrices.
- Inverses can be obtained by inverting the eigenvalues and reconstituting with the eigenvectors. Moore-Penrose generalized inverses can be formed by inverting the nonzero eigenvalues and reconstituting. (See "GInverse".)

  Note: You must decide whether a very small eigenvalue is effectively zero.

- The eigenvalue decomposition enables you to view any square-matrix multiplication as the following:
  - a rotation (multiplication by an orthonormal matrix)
  - a scaling (multiplication by a diagonal matrix)
  - a reverse rotation (multiplication by the orthonormal inverse, which is the transpose), or in notation:

  ```
  A * x = E` * Diag( M ) * E * x;
  ```

  `E` rotates, `Diag( M )` scales, and `E`` reverse-rotates.

Cholesky Decomposition

The `Cholesky()` function performs Cholesky decomposition. A positive semi-definite matrix **A** is re-expressed as the product of a nonsingular, lower-triangular matrix **L** and its transpose: **L***L' = **A**.

```
E = [ 5 4 1 1, 4 5 1 1, 1 1 4 2, 1 1 2 4];
```

```
L = Cholesky( E );
   [2.23606797749979 0 0 0,
   1.788854381999832 1.341640786499874 0 0,
   0.447213595499958 0.149071198499986 1.9436506316151 0,
   0.447213595499958 0.149071198499986 0.914659120760047 1.71498585142509]
```

To verify the results, enter the following:

```
L*L`;
   [5 4 1 1,
   4 5 1 1,
   1 1 4 2,
   1 1 2 4]
```

About Cholesky Decomposition

`Cholesky()` is useful for reconstituting expressions into a more manageable form. For example, eigenvalues are available only for symmetric matrices in JMP, so the eigenvalues of the product **AB** could not be obtained directly. (Although **A** and **B** can be symmetric, the product is not symmetric.) However, you can usually rephrase the problem in terms of eigenvalues of **L′BL** where **L** is the Cholesky root of **A**, which has the same eigenvalues.

Another use of `Cholesky()` is in reordering matrices within `Trace()` expressions. Expressions such as `Trace(A*B*A`)` might involve huge operations counts if **A** has many rows. However, if **B** is small and can be factored into LL′ by Cholesky, then it can be reformulated to `Trace(A*L*L`*A`)`. The resulting matrix is equal to `Trace(L`A`*AL)`. This expression involves a much smaller number of operations, because it consists of only the sum of squares of the **AL** elements.

Use the function `Chol Update()` to update a Cholesky decomposition. If **L** is the Cholesky root of a $n \times n$ matrix **A**, then after using `cholUpdate(L, C, V)`, **L** will be replaced with the Cholesky root of `A+V*C*V'`. **C** is an $m \times m$ symmetric matrix and **V** is an $n \times m$ matrix.

Examples

Manually update the Cholesky decomposition, as follows:

```
exS = [16 1 0 11 -1 12, 1 11 -1 1 -1 1, 0 -1 12 -1 1 0, 11 1 -1 11 -1 9, -1 -1
  1 -1 9 -1, 12 1 0 9 -1 12];
exAchol = Cholesky( exS ); // conducts the Cholesky decomposition

// add two column vectors to the design matrix
exV = [1 1, 0 0, 0 1, 0 0, 0 0, 0 1];

/* The first column vector is added to one of the rows in the design matrix.
The second column vector is subtracted from one of the rows in the design
  matrix. */
exC = [1 0, 0 -1];
exAnew = exS + exV * exC * exV`;
```

```
// update the Cholesky decomposition manually
exAcholnew = Cholesky( exAnew );
```

Instead of manual updating, use `Chol Update()` to update the Cholesky decomposition, as follows:

```
// update the Cholesky decomposition more efficiently
exAcholnew_test = Chol Update( exAchol, exV, exC );
Show( exAcholnew_test );
   exAcholnew_test =
   [ 4 0 0 0 0 0,
   0.25 3.30718913883074 0 0 0 0, ...]

// results are the same as the manual process
Show( exAcholnew );
   exAcholnew=
   [ 4 0 0 0 0 0,
   0.25 3.30718913883074 0 0 0 0, ...]
```

Singular Value Decomposition

The `SVD()` function finds the singular value decomposition of a matrix. That is, for a matrix **A**, `SVD()` returns a list of three matrices **U**, **M**, and **V**, so that `U*diag(M)*V`=A`.

Notes:

- When **A** is taller than it is wide, **M** is more compact, without extra zero diagonals.
- Singular value decomposition re-expresses **A** in the form USV′, where:
  - **U** and **V** are matrices that contain orthogonal column vectors (perpendicular, statistically independent vectors)
  - **S** is a $n \times n$ diagonal matrix containing the nonnegative square roots of the eigenvalues of **A′A**, the singular values of **A**.
- Singular value decomposition is the basis of correspondence analysis.

Example

```
A = [1 2 1 0, 2 3 0 1, 1 0 1 5, 0 1 5 1];
{U, M, V} = SVD( A );
newA = U * Diag( M ) * V`;
   [1 2 0.999999999999997 -2.99456640040496e-15,
   2 3 -1.17505831453979e-15 1,
   0.999999999999997 -2.16851276518826e-15 0.999999999999999 5,
   2.22586706011274e-15 1 5 0.999999999999997]
```

Orthonormalization

The `Ortho()` function orthogonalizes the columns and then divides the vectors by their magnitudes to normalize them. This function uses the Gram-Schmidt method. The column vectors of orthogonal matrices are unit-length and are mutually perpendicular (their dot products are zero).

```
B = Ortho( [1 -1, 1 0, 0 1] );
   [0.408248290463863 -0.707106781186548, 0.408248290463863 0.707106781186548,
      -0.816496580927726 3.14018491736755e-16]
```

To verify that these vectors are orthogonal, multiply **B** by its transpose, which should yield the identity matrix.

```
C = B` * B;
   [1 -3.119061760824e-16, -3.119061760824e-16 1]
```

By default, vectors are normalized, meaning that they are divided by their magnitudes, which scales them to have length 1. Include the option `Scaled(0)` to turn scaling off:

```
Ortho( [1 -1, 1 0, 0 1], Scaled( 0 ) );
   [0.408248290463863 -0.353553390593274, 0.408248290463863 0.353553390593274,
      -0.816496580927726 1.57009245868377e-16]
```

To create vectors whose elements sum to zero, include the `Centered(1)` option. This option is useful when constructing a matrix of contrasts.

```
result = Ortho( [1 -1, 1 0, 0 1], Centered( 1 ) );
   [0.408248290463863 -0.707106781186548, 0.408248290463863 0.707106781186548,
      -0.816496580927726 3.14018491736755e-16]
```

To verify that the elements of each column sum to zero, premultiply by a vector of ones to sum the columns.

```
J(1, 3) * result;
   [1.11022302462516e-16 2.02996189274239e-16]
```

Orthogonal Polynomials

The `Ortho Poly()` function returns orthogonal polynomials for a vector of indices. The polynomial order is specified as a function argument. Orthogonal polynomials can be useful for fitting polynomial models where some regression coefficients might be highly correlated.

```
Ortho Poly( [1 2 3], 2 );
   [-0.707106781186548 0.408248290463862, 0 -0.816496580927726,
      0.707106781186548 0.408248290463864]
```

The polynomial order must be less than the dimension of the vector. Use the `Scaled(1)` option to produce vectors of unit length, as described in "Orthonormalization".

QR Decomposition

`QR()` factorization is useful for numerically stable matrix work. `QR()` returns a list of two matrices. Here is an example:

```
{Q, R} = QR( [11 22, 33 44] );
```

`Q` and `R` hold the results. For a *m* × *n* matrix, `QR()` creates an orthogonal *m* × *m* matrix `Q` and an upper triangular *m* × *n* matrix `R`, so that `X = Q*R`.

Update Inverse Matrices

To add or drop one or more rows in an inverse of an **M`M** matrix, use the `Inv Update(S, X, w)` function. Updating inverse matrices is helpful in drop-1 influence diagnostics and also in candidate design evaluation.

Notes:

- The first argument, **S**, is the matrix to be updated.
- The second argument, **X**, is the matrix of rows to be added or dropped.
- The third argument, `w`, is either `1` to add or `-1` to delete the row or rows.
- Multiple rows can be added or deleted.

Using the `Inv Update(S, X, w)` function is equivalent to calculating the following:

```
S - w * S * X` * Inv( I + w * X * S * X` ) * X * S
```

Where **I** is an identity matrix and `Inv(A)` is an inverse matrix of **A**.

Build Your Own Matrix Functions

You can store your own operations in macros. See "Maximize and Minimize". Similarly, you can create custom matrix functions. For example, you can make your own matrix operation called `Mag()` to find the magnitude of a vector:

```
mag = Function( {x},
   Sqrt( x` * x )
);
```

Similarly, you could create a function called `Normalize` to divide a vector by its magnitude:

```
normalize = Function( {x},
   x / Sqrt( x` * x )
);
```

Statistical Examples

This section contains statistical examples of using matrices.

Regression Example

Suppose that you want to implement your own regression calculation, rather than use the facilities built into JMP. Because of the compact matrix notation, it might require only a few lines of code:

```
Y = [98, 112.5, 84, 102.5, 102.5, 50.5, 90, 77, 112, 150, 128, 133, 85, 112];
X = [65.3, 69, 56.5, 62.8, 63.5, 51.3, 64.3, 56.3, 66.5, 72, 64.8, 67, 57.5,
  66.5];
X = J( N Row( X ), 1 ) || X; // put in an intercept column of 1s
beta = Inv( X` * X ) * X` * Y; // the least square estimates
resid = Y - X * beta; // the residuals, Y - predicted
sse = resid` * resid; // sum of squared errors
Show( beta, sse );
```

This could be expanded into a script that gets its data from a data table, calculates additional results, and shows the results in a report window:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );

// get data into matrices
x = (Column( "Age" ) << Get Values ) || (Column( "Height" ) << Get Values);
x = J(N Row( x ), 1, 1) || x;
y = Column( "Weight" ) << Get Values;

// regression calculations
xpxi = Inv( x` * x );
beta = xpxi * x` * y; // parameter estimates
resid = y - x * beta; // residuals
sse = resid` * resid; // sum of squared errors
dfe = N Row( x ) - N Col( x );  // degrees of freedom
mse = sse / dfe; // mean square error, error variance estimate

// additional calculations on estimates
stdb = Sqrt( Vec Diag( xpxi ) * mse ); // standard errors of estimates
alpha = .05;
qt = Students t Quantile( 1 - alpha / 2, dfe );
betau95 = beta + qt * stdb; // upper 95% confidence limits
betal95 = beta - qt * stdb; // lower 95% confidence limits
tratio = beta :/ stdb; // Student's T ratios
probt = ( 1 - TDistribution( Abs( tratio ), dfe ) ) * 2; // p-values
```

```
// present results
New Window( "Big Class Regression",
   Table Box(
      String Col Box( "Term",{"Intercept", "Age", "Height"} ),
      Number Col Box( "Estimate", beta ),
      Number Col Box( "Std Error", stdb ),
      Number Col Box( "TRatio", tratio ),
      Number Col Box( "Prob>|t|", probt ),
      Number Col Box( "Lower95%", betal95 ),
      Number Col Box( "Upper95%", betau95 ) ) );
```

ANOVA Example

You can implement your own one-way ANOVA. This example presents a problem involving a three-level factor indicating Low, Medium, and High doses and a response measurement. Therefore, this example solves the general linear model, as follows:

$$Y = a + bX + e$$

Where:

- **Y** is a vector of responses
- *a* is the intercept term
- **b** is a vector of coefficients
- **X** is a design matrix for the factor
- *e* is an error term

```
factor = [1, 2, 3, 1, 2, 3, 1, 2, 3];
y = [1, 2, 3, 4, 3, 2, 5, 4, 3];
```

First, build a design matrix for the factor:

```
Design Nom( factor );
   [1 0, 0 1, -1 -1, 1 0, 0 1, -1 -1, 1 0, 0 1, -1 -1]
```

Next, add a column of 1s to the design matrix for the intercept term. You can do this by concatenating `J` and `Design Nom()`:

```
x = J( 9, 1, 1) || Design Nom( factor );
   [1 1 0,
   1 0 1,
   1 -1 -1,
   1 1 0,
   1 0 1,
   1 -1 -1,
   1 1 0,
   1 0 1,
   1 -1 -1]
```

Now, to solve the normal equation, you need to construct a matrix **M** with partitions:

$$\begin{bmatrix} X'X & X'y \\ y'X & y'y \end{bmatrix}$$

You can construct matrix **M** in one step by concatenating the pieces:

```
M = ( x` * x || x` * y )
        |/
  ( y` * x || y` * y );
   [9 0 0 27,
      0 6 3 2,
      0 3 6 1,
      27 2 1 93]
```

Now, sweep **M** over all the columns in **X′X** for the full fit model, and over the first column only for the intercept-only model:

```
FullFit = Sweep( M, [1, 2, 3] ); // full fit model
InterceptOnly = Sweep( M, [1] ); // model with intercept only
```

Recall that some of the standard ANOVA results are calculated by comparing the results of these two models. This example focuses on the full fit model, which produces this swept matrix:

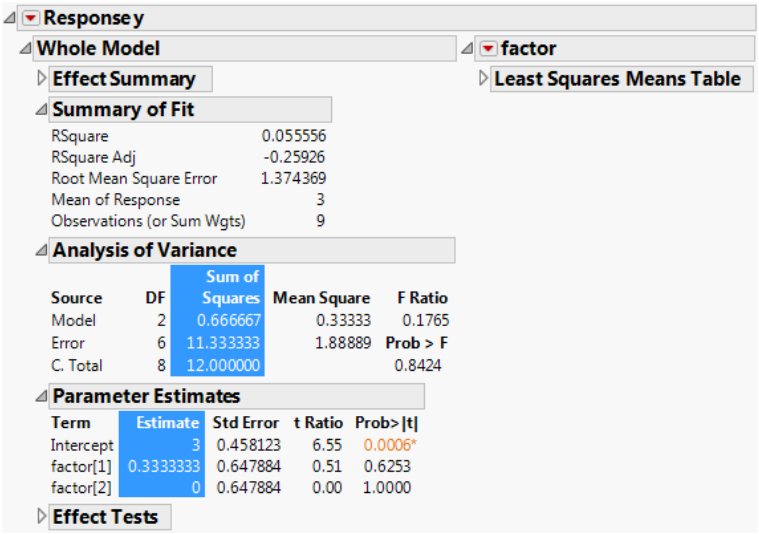
```
[0.111111111111111 0 0 3,
0 0.222222222222222 -0.11111111111111 0.333333333333333,
0 -0.11111111111111 0.222222222222222 0,
-3 -0.33333333333333 0 11.33333333333333]
```

Examine the model coefficients from the upper right partition of the matrix. The lower left partition is the same, except that the signs are reversed: 3, 0.333, 0. The results can be interpreted as follows:

- The coefficient for the intercept term is 3.
- The coefficient for the first level of the factor is 0.333.
- The coefficient for the second level is 0.
- Because of the use of `Design Nom()`, the coefficient for the third level is the difference, –0.333.
- The lower right partition of the matrix holds the sum of squares, 11.333.

You could modify this into a generalized ANOVA script by replacing some of the explicit values in the script with arguments. These results match those from the Fit Model platform.

Figure 7.1 ANOVA Report Within Fit Model



Here is how to construct the report in Figure 7.1:

1. Build a data table (described in "Data Tables"):

```
dt = New Table( "Data" );
dt << New Column( "y", Set Values( [1, 2, 3, 4, 3, 2, 5, 4, 3] ) );
dt << New Column( "factor", "Nominal", Values( [1, 2, 3, 1, 2, 3, 1, 2, 3] )
  );
```

2. Run a model (described in "Launch the Analysis and Save the Script to a Window"):

```
obj = Fit Model(
   Y( :y ),
   Effects( :factor ),
   Personality( "Standard Least Squares" ),
   Run(
      y << {Plot Actual by Predicted( 0 ), Plot Residual by Predicted( 0 ),
      Plot Effect Leverage( 0 )}
   )
);
```

3. Use JSL techniques for navigating displays (described in "Display Box Object References"):

```
ranova = obj << Report;
ranova[Outline Box( 6 )] << Close( 0 );
ranova[Outline Box( 7 )] << Close( 1 );
ranova[Outline Box( 9 )] << Close( 1 );
ranova[Outline Box( 5 ), Number Col Box( 2 )] << Select;
ranova[Outline Box( 6 ), Number Col Box( 1 )] << Select;
```

The result is shown in Figure 7.1.

Associative Arrays

An associative array maps unique keys to values that can be non-unique. An associative array is also called a dictionary, a map, a hash map, or a hash table.

Note the following about keys:

- Keys are placed in quotes.
- The value associated with a key can be a number, date, matrix, list, and so on.
- A key cannot be a floating point number. Only integers are supported.
- You can use matrices as both keys and values, or lists as both keys and values, but you cannot mix the two. In other words, you cannot use a matrix as a key and a list as a value, or the other way around.
- Though associative arrays are not usually ordered, in JMP, keys are returned in alphanumeric order for the purpose of iteration and serialization.

For very large lists, using an associative array is more efficient and faster.

Create Associative Arrays

To create an empty associative array, use the `Associative Array()` function or `[=>]`.

```
cary = Associative Array();
cary = [=> ];
   Associative array( 0 elements ) assigned.
```

Keys and values can be any JSL objects. Items can be added and changed with subscripting:

```
cary = Associative Array();
cary["state"] = "NC";
cary["population"] = 116234;
cary["weather"] = "cloudy";
cary["population"] += 10;
cary["weather"] = "sunny";
cary["high schools"] = {"Cary", "Green Hope", "Panther Creek"};
```

Default Values

A default value determines the value of a key that does not exist in an associative array. If you try to access a key that does not exist in an associative array, an error results. If you define a default value for your associative array, accessing a key that does not exist results in the following:

- adds the key to the associative array
- assigns the default value to the new key
- returns the new key's (default) value instead of an error

If you construct an associative array from a list of strings without assigning values to the keys, then the keys are assigned values of 1. The default value for the associative array is set to 0.

To set the default value:

```
cary = Associative Array();
cary << Set Default Value( "Cary, NC" );
```

To determine whether there is a default value set for an associative array, use the `<<Get Default Value` message.

```
cary << Get Default Value;
   "Cary, NC"
```

If there is no default value, `Empty()` is returned.

Besides the `Set Default Value` message, a default value can be set in the literal constructor using `=>value` without a key.

```
counts = ["a" => 10,
"b" => 3,
=> 0]; // default value of 0
counts["c"] += 1;
Show( counts );
   counts = ["a" => 10, "b" => 3, "c" => 1, => 0];
```

In the first line, the default value is set to 0. In the second line, the key `"c"` does not exist in `counts`. In the output, the key `"c"` is created with the default value of 0 and then incremented by 1.

Note: If a key's value is the default value, then the key is dropped because any key will return the default value.

Associative Array Constructors

Create an empty associative array:

```
map = [=>];
map = Associative Array();
```

Create an empty associative array with a default value:

```
map = [=>0];
map = Associative Array( 0 );
```

Create an associative array with specific values:

```
map = ["yes" => 0, "no" => 1];
```

Create an associative array with specific values with a default value:

```
map = ["yes" => 0, "no" => 1, => 2];
```

Create an associative array with specific values that contains an associative array:

```
map = ["yes" => 0, "no" => 1, "alt" => ["a" => "1", "b" => "2"]];
```

Create an associative array from a list that contains two lists of a key-value pair:

```
map = Associative Array( {{"yes", 0}, {"no", 1}} );
```

Create an associative array from a list that contains two lists of a key-value pair with a default value:

```
map = Associative Array( {{"yes", 0}, {"no", 1}}, 2 );
```

Create an associative array from a list of keys and a list of values:

```
map = Associative Array( {"yes", "no"}, {0, 1} );
```

Create an associative array from a list of keys and a list of values with a default value:

```
map = Associative Array( {"yes", "no"}, {0, 1}, 2 );
```

Create an associative array from two column references. The first column contains the keys and the second contains the values.

```
map = Associative Array( :name, :height );
```

Create an associative array from two column references with a default value:

```
map = Associative Array(:name, :height, .);
```

Create an associative array from a single list of keys or a single column reference of keys with a default value of 0:

```
set = Associative Array( {"yes", "no"} );
set = Associative Array( :name );
```

Work with Associative Arrays

Find the Number of Keys

To determine the number of keys that an associative array contains, use the `N Items()` function.

```
cary = Associative Array();
cary["state"] = "NC";
cary["population"] = 116234;
cary["weather"] = "cloudy";
cary["population"] += 10;
cary["weather"] = "sunny";
cary["high schools"] = {"Cary", "Green Hope", "Panther Creek"};
N Items( cary );
   4
```

Add and Delete Keys and Values

To add or delete key-value pairs from an associative array, use the following functions:

- `Insert()`
- `Insert Into()`
- `Remove()`
- `Remove From()`

Notes:

- `Insert()` and `Remove()` return a named copy of the given associative array with the key-value pair added or removed.
- `Insert Into()` and `Remove From()` add or remove the key-value pairs directly from the given associative array.
- `Insert()` and `Insert Into()` take three arguments: the associative array, a key, and a value.
- `Remove()` and `Remove From()` take two arguments: the associative array and a key.
- If you insert a key with no value provided, the key is assigned a value of 1.

Examples

The following examples illustrate `Insert()` and `Insert Into()`:

```
cary = Associative Array();
cary["state"] = "NC";
cary["population"] = 116234;
cary["weather"] = "cloudy";
cary["population"] += 10;
```

```
cary["weather"] = "sunny";
cary["high schools"] = {"Cary", "Green Hope", "Panther Creek"};
newcary = Insert( cary, "time zone", "Eastern" );
Show( cary, newcary );
   cary = ["high schools" => {"Cary", "Green Hope", "Panther Creek"},
      "population" => 116244, "state" => "NC", "weather" => "sunny"];
   newcary = ["high schools" => {"Cary", "Green Hope", "Panther Creek"},
      "population" => 116244, "state" => "NC", "time zone" => "Eastern",
      "weather" => "sunny"];

cary = Associative Array();
cary["state"] = "NC";
cary["population"] = 116234;
cary["weather"] = "cloudy";
cary["population"] += 10;
cary["weather"] = "sunny";
cary["high schools"] = {"Cary", "Green Hope", "Panther Creek"};
Insert Into(cary, "county", "Wake");
Show( cary );
   cary = ["county" => "Wake", "high schools" => {"Cary", "Green Hope", "Panther
      Creek"}, "population" => 116244, "state" => "NC", "weather" => "sunny"];
```

Note that `cary <<Insert` is a message sent to an associative array that does the same thing as the function `Insert Into()`. For example, these two statements are equivalent:

```
cary << Insert( "county", "Wake" );
Insert Into( cary, "county", "Wake" );
```

The following examples illustrate `Remove()` and `Remove From()`:

```
cary = Associative Array();
cary["state"] = "NC";
cary["population"] = 116234;
cary["weather"] = "cloudy";
cary["population"] += 10;
cary["weather"] = "sunny";
cary["high schools"] = {"Cary", "Green Hope", "Panther Creek"};
newcary = Remove( cary, "high schools" );
Show( cary, newcary );
   cary = ["high schools" => {"Cary", "Green Hope", "Panther Creek"},
      "population" => 116244, "state" => "NC", "weather" => "sunny"];
   newcary = ["population" => 116244, "state" => "NC", "weather" => "sunny"];
Remove From( cary, "weather" );
Show( cary );
   cary = ["high schools" => {"Cary", "Green Hope", "Panther Creek"},
      "population" => 116244, "state" => "NC", "weather" => "sunny"];
   newcary = ["population" => 116244, "state" => "NC", "weather" => "sunny"];
   cary = ["high schools" => {"Cary", "Green Hope", "Panther Creek"},
      "population" => 116244, "state" => "NC"];
```

Note that `aa << Remove` is a message sent to an associative array that does the same thing as the function `Remove From()`. For example, these two statements are equivalent:

```
cary << Remove( "weather" );
Remove From( cary, "weather" );
```

Find Keys or Values in an Associative Array

To determine whether a certain key is contained within an associative array, use the `Contains()` function.

```
cary = Associative Array();
cary["state"] = "NC";
cary["population"] = 116234;
cary["weather"] = "cloudy";
cary["population"] += 10;
cary["weather"] = "sunny";
cary["high schools"] = {"Cary", "Green Hope", "Panther Creek"};
Contains(cary, "high schools");
  1
Contains(cary, "lakes");
  0
```

To obtain a list of all keys contained in an associative array, use the `<< Get Keys` message.

```
cary << Get Keys;
  {"high schools", "population", "state", "weather"}
```

To obtain a list of all values contained in an associative array, use the `<<Get Values` message.

```
cary << Get Values;
  {{"Cary", "Green Hope", "Panther Creek"}, 116244, "NC", "sunny"}
```

If you want to see only the values for certain keys, you can specify them as arguments. The keys must be given as a list.

```
cary << Get Values({"state", "population"});
  {"NC", 116244}
```

To see a value for a single key, use the `<<Get Value` message. Specify only one key and do not place it in a list.

```
cary << Get Value("weather");
  "sunny"
```

To obtain a list of all key-value pairs in an associative array, use the `<<Get Contents` message.

```
cary << Get Contents;
  {{"high schools", {"Cary", "Green Hope", "Panther Creek"}},
  {"population", 116244},
  {"state", "NC"},
  {"weather", "sunny"}}
```

Note: Using the `<<Get Contents` message, the returned list does not include the default value. Keys are listed alphabetically.

Iterate through an Associative Array

To iterate through as associative array, use the `<<First` and `<<Next` messages. `<<First` returns the first key in the associative array. `<<Next(key)` returns the key that follows the *key* that is given as the argument.

The following example removes all key-value pairs from the associative array `cary`, leaving an empty associative array:

```
currentkey = cary << First;
total = N Items( cary );
For( i = 1, i <= total, i++,
   nextkey = cary << Next( currentkey );
   Remove From( cary, currentkey );
   currentkey = nextkey;
);
Show( cary );
   cary = [=>];
```

Applications for Associative Arrays

You can use associative arrays to quickly and efficiently perform other tasks.

Get the Unique Values from a Data Table Column

Note: The strategy described here is not useful for columns with floating point numbers. Use Summarize instead. See "Store Summary Statistics in Global Variables".

A key can exist only once in an associative array, so putting a column's values into one automatically results in the unique values. For example, the Big Class.jmp sample data table contains 40 rows. To see how many unique values are in the column height, run this script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
unique heights = Associative Array( dt:height );
nitems( unique heights );
   17
```

There are only 17 unique values for height. You can use those unique values by getting the keys:

```
unique heights << Get Keys;
   {51, 52, 55, 56, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70}
```

Note: This is possible because you can use any JMP data type as keys in an associative array, not only strings.

Using an associative array to discover unique values in a column is efficient and fast. The following script takes some time to create a data table with 100,000 rows. Finding the 39 unique values takes very little time.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
nms = dt:name << Get Values;
dtbig = New Table( "Really Big Class",
   New Column( "name",
      Character,
      Set Values( nms[J( 100000, 1, Random Integer( N Items( nms ) ) )] )
   )
);
Wait( 0 );
t1 = Tick Seconds();
Write(
   "\!N# names from Really Big Class = ",
   N Items( Associative Array( dtbig:name ) ),
   ", elapsed time=",
   Tick Seconds() - t1
);
   # names from Really Big Class = 39, elapsed time=0.116666666639503
```

Sort a Column's Values in Lexicographic Order

Because keys are ordered lexicographically, putting the values into an associative array also sorts them. For example, the `<<Get Keys` message returns the keys (unique values of the names column) in ascending order:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
unique names = Associative Array( dt:name );
unique names << Get Keys;
   {"ALFRED", "ALICE", "AMY", "BARBARA", "CAROL", "CHRIS", "CLAY", "DANNY",
      "DAVID", "EDWARD", "ELIZABETH", "FREDERICK", "HENRY", "JACLYN", "JAMES",
      "JANE", "JEFFREY", "JOE", "JOHN", "JUDY", "KATIE", "KIRK", "LAWRENCE",
      "LESLIE", "LEWIS", "LILLIE", "LINDA", "LOUISE", "MARION", "MARK",
      "MARTHA", "MARY", "MICHAEL", "PATTY", "PHILLIP", "ROBERT", "SUSAN",
      "TIM", "WILLIAM"}
```

Compare Columns in Two Different Data Tables

Using associative arrays, determining which values in one column are not in another column (or determining which values are in both columns) is fast. For example, given two data tables with information about countries, which countries are in both data tables?

Place the columns of each data table that contain country names into associative arrays:

```
dt1 = Open( "$SAMPLE_DATA/BirthDeathYear.jmp" );
dt2 = Open( "$SAMPLE_DATA/World Demographics.jmp" );
aa1 = Associative Array( dt1:Country );
aa2 = Associative Array( dt2:Territory );
```

Use `N Items()` to see how many countries appear in each data table:

```
N Items(aa1);
   23
N Items(aa2);
   239
```

Use the `<<Intersect` message to find the common values:

```
aa1 = Associative Array( dt1:Country );
aa1 << Intersect( aa2 );
```

Look at the results:

```
Show(N Items(aa1), aa1 << Get Keys);
   N Items(aa1) = 21;
   aa1 << get keys = {"Australia", "Austria", "Belgium", "France", "Greece",
      "Ireland", "Israel", "Italy", "Japan", "Mauritius", "Netherlands", "New
      Zealand", "Norway", "Panama", "Poland", "Portugal", "Romania",
      "Switzerland", "Tunisia", "United Kingdom", "United States"};
```

This example uses a set operation called intersection. For more examples of using set operations with associative arrays to compare values, see "Associative Arrays in Set Operations".

Associative Arrays in Graph Theory

You can use associative arrays for graph theory data structures, such as the following directed graph example:

```
g = Associative Array();
g[1] = Associative Array({1, 2, 4});
g[2] = Associative Array({1, 3});
g[3] = Associative Array({4, 5});
g[4] = Associative Array({4, 5});
g[5] = Associative Array({1, 2});
```

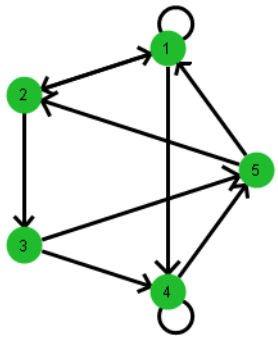
This is a two-level associative array. The associative array `g` contains five associative arrays (`1`, `2`, `3`, `4`, and `5`). In the containing array `g`, both the keys (1-5) and the values (the arrays that define the map) are important. In the inner associative arrays, the values do not matter. Only the keys are important.

The associative array represents the graph shown in Figure 7.2:

- node 1 is incident on nodes 1, 2, and 4
- node 2 is incident on nodes 1 and 3

- node 3 is incident on nodes 4 and 5
- node 4 is incident on nodes 4 and 5
- node 5 is incident on nodes 1 and 2

Figure 7.2 Example of a Directed Graph



The following depth-first search function can be used to traverse this graph, or any other directed graph represented as an associative array:

```
dfs = Function( {ref, node, visited},
    {chnode, tmp},
    Write( "\!NNode: ", node, ", ", ref[node] << Get Keys );
    visited[node] = 1;
    tmp = ref[node];
    chnode = tmp << first;
    While( !Is Missing( chnode ),
        If( !visited[chnode],
            visited = Recurse( ref, chnode, visited )
        );
        chnode = tmp << Next( chnode );
    );
    visited;
);
```

Notes:

- The first argument is the associative array that contains the map structure.
- The second argument is the node that you want to use as the starting point.
- The third argument is a vector that the function uses to keep track of nodes visited.

To see how the preceding example works, add the following expression to the end of the script:

```
dfs( g, 2, J( N Items( g << Get Keys ), 1, 0 ) );
```

Here is the output:

```
Node 2: {1, 3}
Node 1: {1, 2, 4}
Node 4: {4, 5}
Node 5: {1, 2}
Node 3: {4, 5}
[1, 1, 1, 1, 1]
```

The first five output lines show that starting from node 2, you can reach all the other nodes in the order in which they are listed. Each node also lists the nodes it is incident on (the keys). The value for each key is 1. The final line is a matrix that shows that you can reach each node from 2. If there were nodes that could not be reached from node 2, their values in the matrix would be 0.

Here is how to read the traversal of the nodes:

1. Start at node 2 and go to node 1.
2. From node 1, go to node 4.
3. From node 4, go to node 5.
4. From node 5, go back to node 2, and then to node 3.

Map Script

Here is the script that produced the map shown in Figure 7.2.

```
New Window( "Directed Graph",
   Graph Box(
      Frame Size( 300, 300 ),
      X Scale( -1.5, 1.5 ),
      Y Scale( -1.5, 1.5 ),
      Local( {n = N Items( g ), k = 2 * Pi() / n, r, i, pt, from, to,
         edge, v, d},
         Fill Color( "green" );
         Pen Size( 3 );
         r = 1 / (n + 2);
         For( i = 1, i <= n, i++,
            pt = Eval List( {Cos( k * i ), Sin( k * i )} );
            edges = g[i];
            For( edge = edges << First, !Is Empty( edge ),
               edge = edges << Next( edge ),
               to = Eval List( {Cos( k * edge ), Sin( k * edge )} );
               If( i == edge,
                  Circle( Eval List( 1.2 * pt ), 0.9 * r ), // else
                  v = pt - to;
                  d = Sqrt( Sum( v * v ) );
                  {from, to} = Eval List(
                     {pt * (d - r) / d + to * r / d, pt * r / d + to *
```

```
                        (d - r) / d}
                    );
                    Arrow( from, to );
                );
            );
            Circle( pt, r, "fill" );
            Text( Center Justified, pt - {0, 0.05}, Char( i ) );
        );
    )
  )
);
```

Associative Arrays in Set Operations

You can also use associative arrays to perform set operations. The following examples show how to take a union of two sets, a difference of two sets, and an intersection of two sets.

First, create three sets and view them in the log:

```
set_y = Associative Array( {"chair", "person", "relay", "snake", "tripod"} );
set_z = Associative Array( {"person", "snake"} );
set_w = Associative Array( {"apple", "orange"} );

// write the sets to the log
Write(
  "\!NExample:\!N\!tset_y = ",
  set_y << Get Keys,
  "\!N\!tset_z = ",
  set_z << Get Keys,
  "\!N\!tset_w = ",
  set_w << Get Keys
);
  Example:
    set_y = {"chair", "person", "relay", "snake", "tripod"}
    set_z = {"person", "snake"}
    set_w = {"apple", "orange"}
```

Union Operation

To find the union of two sets, insert one set into the other:

```
set_z << Insert( set_w );
Write( "\!N\!NUnion operation (set_w, set_z):,\!N\!tset_z = ", set_z << Get
  Keys );
  Union operation (set_w, set_z):,
    set_z = {"apple", "orange", "person", "snake"}
```

Difference Operation

To find the difference of two sets, remove one set from the other:

```
set_y << Remove( set_z );
Write( "\!N\!NDifference operation (set_z from set_y):\!N\!tset_y = ", set_y
   << Get Keys );
   Difference operation (set_z from set_y):
      set_y = {"chair", "relay", "tripod"}
```

Intersect Operation

To find the intersection of two sets, use the `aa << Intersect` message.

```
set_w << Intersect( set_z );
Write( "\!N\!NIntersect operation (set_w, set_z):\!N\!tset_w = ", set_w << Get
   Keys );
   Intersect operation (set_w, set_z):
      set_w = {"apple", "orange"}
```

Example of Using Set Operations

Given a list of names, which of them are not contained in Big Class.jmp? You can find the answer by taking the difference of the two sets of names.

1. Get the list of names and open the data table:

   ```
   names list = {"ROBERT", "JEFF", "CHRIS", "HARRY"};
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   ```

2. Put the list of names into an associative array:

   ```
   names = Associative Array( names list );
   ```

3. Perform a difference operation by removing the column values from your list:

   ```
   names << Remove( Associative Array( dt:name ) );
   ```

4. Look at the result:

   ```
   Write( "\!NWhich of {ROBERT, JEFF, CHRIS, HARRY}, is not in Big Class = ",
      names << Get Keys );
      Which of {ROBERT, JEFF, CHRIS, HARRY}, is not in Big Class = {"HARRY", "JEFF"}
   ```

Chapter 8

Programming Methods

Complex Scripting Techniques and Additional Functions

This chapter includes advanced techniques, such as throwing and catching exceptions, encrypting scripts, and using complex expressions.

Contents

Lists and Expressions 255
Expressions 255
Macros 267
Manipulating Lists 267
Manipulating Expressions 270
Advanced Scoping and Namespaces 274
Names Default To Here 274
Scoped Names 278
Namespaces 282
Referencing Namespaces and Scopes 286
Resolving Named Variable References 289
Best Practices for Advanced Scripting 290
Advanced Programming Concepts 291
Throw and Catch Exceptions 291
Functions 293
Recursion 296
Includes 296
Load and Save Text Files 297
Run External Programs 297
Classes 299
File and Directory Operations 302
Select a Directory or File 302
Get a List of Filenames 303
Connect to a Network Directory 304
Scripting By Groups 305
Encrypt and Decrypt Scripts 306
Encryption and Global Variables 308
Encrypting Scripts in Data Tables 308
Working with Expressions in an Encrypted Script 309
Additional Numeric Functions 310
Derivatives 310
Algebraic Manipulations 312
Maximize and Minimize 314
Identifying Differences Between Strings, Lines, or Sequences 315
Scheduling Actions 317
Functions that Communicate with Users 318
Writing to the Log 319
Send Information to the User 320

Lists and Expressions

Expressions

This section delves into JSL expressions and introduces several JSL functions that can be used to manipulate JSL expressions. These functions provide JSL programmers with a rich, powerful, meta programming toolkit. One of the benefits of such a toolkit is that it provides JSL programmers with a powerful macro system capability that is similar in functionality to macro systems available in other modern programming languages.

In order to understand JSL expressions, it is useful to keep in mind that everything that happens in JSL is a result of a function call. Function calls come in three varieties:

prefix The function name comes before its arguments (for example, `Foo( m, n )`). Most function calls in JSL are of this form.

infix The function name is between its arguments (for example, `m + n`). This form is common for mathematical operators.

special Functions like `[ ]` (the subscript operator) fall into this category. Since there are only a handful of special functions, they are not discussed further in this section.

Although function calls come in three forms, all function calls can be written in prefix form. It turns out that JMP actually uses the prefix form internally and so, in general, it is very useful to think of JSL expressions in this way.

To illustrate this idea, begin by examining the syntactic structure of JSL expressions a bit more closely. All JSL expressions must conform to a very simple syntactic structure. Each expression consists of a *head* expression and a sequence of zero or more argument expressions. Consider the following JSL expressions, written first in infix form and then in prefix form, to see how this syntactic structure applies:

```
x = 100.1
m & n
1 <= z <= 20
```

Using the prefix form, these expressions are written equivalently:

```
Assign( x, 100.1 )
And( m, n )
Less or Equal( 1, z, 20 )
```

The head of the first example is `Assign` and its arguments are `x` and 100.1. For the second example, the head is `And` and its arguments are `m` and `n`, whereas for the third example, the head is `Less or Equal` and its arguments are 1, `z`, and 20. For these examples, the "=", "&", and "<=" operators are just human friendly ways of representing `Assign`, `And`, and `Less or Equal`.

Examine a few additional simple examples.

```
Assign( 100.1, x )
And( )
Less or Equal( z )
```

These examples also conform to the required syntactic structure, but they differ from the previous examples in that they all produce an error message when they are evaluated. This point leads to an important distinction that you should make when thinking about JSL expressions. That is, you should separate the idea of the syntactic structure of a JSL expression from what happens when the expression is evaluated.

Consider the following examples that present both the human friendly form and the prefix form in each case:

```
x = 100.1 + Log( z )                    // human friendly form
Assign( x, Add( 100.1, Log( z ) ) )     // prefix form
```

This example illustrates that the arguments of a head expression can also be expressions.

```
// human friendly form
x = 0.8;
x ||= Random Uniform( 0.2, 0.6 );

// prefix form
Glue(
   Assign( x, 0.8 ),
   Concat To( x, Random Uniform( 0.2, 0.6 ) )
)
```

This example would typically be referred to as a JSL script, but it is just another case of a JSL expression. Here, the *head* expression is `Glue`, which has two arguments: the expressions `Assign( x, 0.8 )` and `Concat To( x, Random Uniform( 0.2, 0.6 ) )`. This example illustrates two important points:

1. The ";" symbol is another instance of a human friendly operator. In this case, it is for the `Glue` function.
2. There is no distinction between what are typically referred to as a JSL script and a JSL expression.

The remainder of this section covers how to assign an expression to a variable and then how to manipulate, or execute, that expression when needed. For information about how JMP resolves names when evaluating JSL expressions, see the section "Rules for Name Resolution".

Quoting and Unquoting Expressions

The operators to control when expressions are evaluated are `Expr` and `Eval`, which you should think of as the procrastination and eager operators. `Expr` just copies its argument as an expression, rather than evaluating it. `Eval` does the opposite: it evaluates its argument, and then takes that result and evaluates it again.

`Expr` and `Eval` can be thought of as quoting and unquoting operators, telling JMP when you mean the expression itself, and when you mean the result of evaluating the expression.

The following examples all assume these two assignments:

```
x = 1; y = 20;
```

If you assign the expression `x+y` to *a*, quoting it as an expression with `Expr`, then whenever *a* is evaluated, it evaluates the expression using the current values of *x* and *y* and returns the result. (Exceptions are the utilities `Show`, `Write`, and `Print`, which do not evaluate expressions for pure name arguments.)

```
x = 1; y = 20;
a = Expr( x + y );
a;
   21
```

If you want the expression that is stored in the name, rather than the result of evaluating the expression, use the `NameExpr` function. See "Retrieve a Stored Expression Rather than its Result".

```
x = 1; y = 20;
Show( Name Expr( a ) );
   NameExpr(a) = x + y
```

If you assign an extra level of expression-quoting, then when *a* is evaluated, it unpacks one layer and the result is the expression `x+y`.

```
x = 1; y = 20;
a = Expr( Expr( x + y ) );
Show( a );
   a = Expr(x + y)
```

If you want the value of the expression, then use `Eval` to unpack all layers:

```
x = 1; y = 20;
Show( Eval( a ) );
   Eval(a) = 21
```

You can do this to any level, for example:

```
x = 1; y = 20;
a = Expr( Expr( Expr( Expr( x + y ) ) ) );
b = a;
   Expr( Expr( x + y ) )
c = Eval( a );
   Expr( x + y )
d = Eval( Eval( a ) );
   x+y
e = Eval( Eval( Eval( a ) ) );
   21
```

For more information about expression handling, see Morgan (2010).

Quote an expression as a string

The `JSL Quote()` function returns the contents of an expression as a quoted string. Comments and white space in the string are preserved.

The following expression is an example:

```
x = JSL Quote( /* Begin quote. */
For (i = 1, i <= 5, i++,
// Print the value of i.
Print( i );
);
// End expression.
);
Show( x );
```

In the output, the contents of the `JSL Quote()` function are enclosed in quotes.

```
x = " /* Begin quote. */
For (i = 1, i <= 5, i++,
   // Print the value of i.
Print(i);
);
   // End expression.
";
```

Quoting Expressions

A common use of the `Expr()` function is to store an expression in a JSL variable. This could be considered as a macro. Consider the following script:

```
dist = Expr( Distribution( Column( height ) ) );
```

To execute the above expression, just use the following:

```
dist;
```

You can use expressions in a loop to execute it:

```
For( i = 0, i < 3, i = i + 1, dist );
```

This loop results in three Distribution platform reports for height.

Use `Eval()` to evaluate an expression explicitly:

```
Eval( dist );
```

Note, however, that in column formulas, `Eval()` only works if it is outermost in the formula. So, for example,

```
Formula( Log( Eval( Column Name( i ) ) ) );
```

would generate an error. Instead, use:

```
Formula( Eval( Substitute( Expr( Log( xxx ) ), Expr( xxx ), Column Name( i ) )
  ) );
```

As another example,

```
Formula( Eval( Column Name( i ) ) + 10 );
```

generates an error, since `Eval()` is actually under the `Add` function. Instead, use:

```
Formula(Eval(Substitute(Expr(xxx+10), Expr(xxx), column name(i))))
```

Retrieve a Stored Expression Rather than its Result

What if you wanted the symbolic value of a variable (such as the expression `Distribution(Column(height))` stored in `dist` above), rather than the evaluation of it (the actual launched platform)? The `Name Expr` function does this. `Name Expr` retrieves its argument as an expression without evaluating it.

`Expr` returns its argument exactly, whereas `Name Expr` looks up the expression stored in its argument. `Name Expr` "unpacks" just one layer to get the expression, but does not keep unpacking to get the result.

For example, you would need to use this if you had an expression stored in a name and you wanted to edit the expression:

```
popVar = Expr( Summation( i = 1, N Row(), (y[i] - Col Mean( y )) ^ 2 / N Row()
  ) );
  Summation( i = 1, N Row(), (y[i] - Col Mean( y )) ^ 2 / N Row() )

unbiasedPopVar = Substitute( Name Expr( popVar ), Expr( Wild()/N Row() ),
  Expr( (y[i] - Col Mean( y )) ^ 2 / ( N Row() - 1 ) ) );
  Summation( i = 1, N Row(), (y[i] - Col Mean( y )) ^ 2 / (N Row() - 1) )
```

Compare `x`, `Expr(x)`, `NameExpr(x)`, and `Eval(x)` after submitting this script:

```
a = 1; b = 2; c = 3;
x = Expr( a + b + c );
```

Table 8.1 Compare Eval, Name Expr, and Expr

Command and result	Explanation
`x;` *6*	Evaluates *x* to the expression `a+b+c`, and then evaluates the expression, returning the result, 6 (unpacks *all* layers).
`Eval( x );` *6*	Equivalent to simply calling *x*. Evaluates *x* to the expression `a+b+c`, and then evaluates the expression, returning the result, 6 (unpacks *all* layers).
`NameExpr( x );` *a+b+c*	Returns the expression that was stored in *x*, which is `a+b+c` (unpacks the *outside* layer).
`Expr( x );` *x*	Returns the expression `x` (packs *one* layer).

JSL also supports functions to access and traverse expressions, all of them either a name or a literal expression as an argument. In the following, *expressionArg* is either a single name or a compound expression to be taken literally.

`NArg(`*expressionArg*`)` finds the number of arguments in *expressionArg*.

The *expressionArg* can be a name holding an expression, an expression evaluated to an expression, or a literal expression quoted by `Expr()`.

`NArg(`*name*`)` obtains the expression held in `name` (it is not evaluated) and returns the number of arguments

`NArg(`*expression*`)` evaluates `expression` and returns the number of arguments

`NArg(Expr(`*expression*`))` returns the number of arguments to literal expression.

For example, if `aExpr = {a+b,c,d,e+f+g};`

- `NArg(aExpr)` results in 4.
- `NArg(Arg(aExpr,4))` results in 3.
- `NArg(Expr({1,2,3,4}))` results in 4.

`Head(`*expressionArg*`)` returns the head of the expression without any arguments. If the expression is an infix, prefix, or postfix special character operator, then it is returned as the functional equivalent.

The *expressionArg* can be a name holding an expression, an expression evaluated to an expression, or a literal expression quoted by `Expr()`.

For example, if `aExpr = expr(a+b);`

- `r = Head(aExpr)` results in `Add()`.
- `r = Head (Expr(sqrt(r)))` results in `Sqrt()`.
- `r = Head({1,2,3})` results in `{}` .

`Arg(`*`expressionArg`*`,`*`indexArg`*`)` extracts the specified argument of the symbolic expression, resulting in an expression.

For example, `Arg(`*`expressionArg`*`,i)` extracts the i^{th} argument of *`expressionArg`*.

The *`expressionArg`* can be a name holding an expression, an expression evaluated to an expression, or a literal expression quoted by `Expr()`.

- `Arg(`*`name`*`,i)` obtains the expression held in *`name`* (it is not evaluated) and finds the `i`th argument.
- `Arg(`*`expression`*`,i)` evaluates *`expression`* and finds the `i`th argument.
- `Arg(Expr(`*`expression`*`),i)` finds the i^{th} argument of *`expression`*.

As another example, if `aExpr = Expr(12+13*sqrt(14-15));`

- `Arg(aExpr,1)` yields `12`.
- `Arg(aExpr,2)` yields `13*sqrt(14-15)`.
- `Arg(Expr(12+13*sqrt(14-15)),2)` yields `13*sqrt(14-15)`.

To extract an argument of an argument inside an expression, you can nest Arg commands:

- `Arg(Arg(aExpr,2),1)` yields the first argument within the second argument of `aExpr`, or 13.
- `Arg(Arg(aExpr,2),2)` yields `Sqrt(14-15)`.
- `Arg(Arg(Arg(aExpr,2),2),1)` yields `14-15`.
- `Arg(Arg(Arg(aExpr,2),2),3)` yields `Empty()`

Here is a description of how the last example line unwraps itself:

1. The inner `Arg` statement is evaluated.

   ```
   Arg(aExpr,2)
     13 * Sqrt( 14 - 15 )
   ```

2. Then the next one is evaluated.

   ```
   Arg(Arg(aExpr,2),2)
   // this is equivalent to Arg(Expr (13 * Sqrt( 14 - 15 ) ), 2)
     Sqrt( 14 - 15 )
   ```

3. Finally, the outer Arg is evaluated.

   ```
   Arg(Arg(Arg(aExpr,2),2),3)
   // this is equivalent to Arg (Expr (Sqrt( 14 - 15 ) ), 3)
   ```

```
Empty()
```

There is only one element to the `Sqrt` expression, so a request for the third argument yields `Empty()`. To access the two arguments inside the `Sqrt` expression, try this:

```
Arg(Arg(Arg(Arg(aExpr,2),2),1),2);
  15
```

`HeadName(`*`expressionArg`*`)` returns the name of the head of the expression as a string. If the expression is an infix, prefix, postfix, or other special character operator, then it is returned as the functional equivalent.

The *`expressionArg`* can be a name holding an expression, an expression evaluated to an expression, or a literal expression quoted by `Expr()`.

For example, if `aExpr = expr(a+b);`

- `r = HeadName (aExpr)` results in `"Add"`.
- `r = HeadName (Expr(sqrt(r)))` results in `"Sqrt"`.
- `r = HeadName ({1,2,3})` results in `"List"`.

In previous versions of JMP, other versions of `Arg`, `Narg`, `Head`, and `HeadName` were implemented, called `ArgExpr`, `NArgExpr`, `HeadExpr`, and `HeadNameExpr`, respectively. These did the same thing, but did not evaluate their argument. These forms are now deprecated and will not be documented in future versions.

Making Substitutions

`Eval Insert` is for the situation where you want to make substitutions, by evaluating expressions inside a character string.

With `Eval Insert`, you specify characters that delimit the start and end of an expression, and everything in between is evaluated and expanded.

There are two functions, one to return the result, the other to do it in-place.

```
resultString = EvalInsert( string with embedded
  expressions,startDelimiter,endDelimiter )
EvalInsertInto( string l-value with embedded
  expressions,startDelimiter,endDelimiter )
```

The delimiter is optional. The default start delimiter is `"^"`. The default end delimiter is the start delimiter.

```
xstring = "def";
r = Eval Insert( "abc^xstring^ghi" );  // returns "abcdefghi";

r = "abc^xstring^ghi"; // in-place evaluation
Eval Insert Into( r ); // r now has "abcdefghi";
```

```
// with specified delimiter
r = Eval Insert( "abc%xstring%ghi","%" );  // returns "abcdefghi";

// with different start and end delimiters
r = Eval Insert( "abc[xstring]ghi","[","]" );  // returns "abcdefghi";
```

When a numeric value contains locale-specific formatting, include the <<Use Locale(1) option. The following example substitutes a comma for the decimal point based on the computer's locale setting.

```
Eval Insert( "^1.2^", <<Use Locale( 1 ) );
  1,2
```

Evaluate expressions inside lists

`Eval List` evaluates expressions inside a list and returns a list with the results:

```
x = { 1 + 2, 3 + 4 };
y = Eval List( x );   // returns y is {3,7}
```

`Eval List` is useful for loading the list of user choices returned by `Column Dialog` or `New Window` with the `Modal` argument.

Evaluate expressions inside expressions

`Eval Expr()` evaluates only the inner expressions and returns an expression that contains the results of the evaluation. By comparison, `Eval` evaluates the inner expressions, takes the results, and then evaluates it again.

Suppose that a data table contains a column named X3. Here is an example of using `Eval Expr()` to evaluate the inner expression first:

```
x = Expr( Distribution( Column( Expr("X"||Char( i ) ) ) ) );
i = 3;
y = Eval Expr( x );   // returns Distribution( Column( "X3" ) )
```

To evaluate further, you need to either call the result in a subsequent step, or else put `Eval()` around the `Eval Expr()`. The following examples create a Distribution report.

```
// two-step method
x = Expr( Distribution( Column( Expr( "X" || Char( i ) ) ) ) );
i = 3;
y = Eval Expr( x );
y;

// one-step method
x = Expr( Distribution( Column( Expr( "X" || Char( i ) ) ) ) );
i = 3;
Eval( Eval Expr( x ) );
```

See Table 8.3 to learn what would happen if you tried to use Eval directly on *x* without first doing Eval Expr.

Parsing Strings into Expressions, and Vice Versa

Parsing is the syntactic scanning of character strings into language expressions. Suppose that you have read in a valid JSL expression into a character string, and now want to evaluate it. The Parse function returns the expression. To evaluate it, use the Eval function.

```
x = Parse( "a=1" ) ;  // x now has the expression a=1
Eval( Parse( "a=1" ) ); // a now has the value 1
```

To go in the reverse, use the Char function, which converts an expression into a character string. Usually the argument to a Char function is an Expr function (or a NameExpr of a variable), since Char evaluates its argument before deparsing it.

```
y = Char( Expr( a = 1 ) ); // results in y having the character value "a=1"
z = Char( 42 ); // returns "42"
```

The Char function allows arguments for field width and decimal places if the argument is a number. The default is 18 for width and 99 for decimal (Best format).

```
Char( 42, 5, 2 ); // returns the character value "42.00"
```

To preserve locale-specific formatting in the numeric value, include the <<Use Locale(1) option as shown in the following example:

```
Char( 42, 5, 2, <<Use Locale(1) ); // returns the character value "42,00" in
  the French locale
```

The reverse of Char is not quite as simple. To convert a character string into an expression, you use Parse, but to convert a character string into a number value, you use Num.

```
Parse( y );
Num( z );
```

Table 8.2 Functions to Store or Evaluate Expressions

Function	Syntax	Explanation
Char	Char(Expr(*expression*)) Char(*name*)	Converts an *expression* into a character string. The expression must be quoted with Expr; otherwise its evaluation is converted to a string.
	string = char(*number*, *width*, *decimal*)	Converts a *number* into its character representation. *Width* and *decimal* are optional arguments to specify formatting; the default is 18 for width and 99 for decimal.

Table 8.2 Functions to Store or Evaluate Expressions *(Continued)*

Function	Syntax	Explanation
Eval	Eval(*x*)	Evaluates *x*, and then evaluates the result of *x* (unquoting).
Eval Expr	Eval Expr(*x*)	Returns an expression with all the expressions inside *x* evaluated.
Eval List	Eval List(*list*)	Returns a list of the evaluated expressions inside *list*.
Expr	Expr(*x*)	Returns the argument unevaluated (expression-quoting).
NameExpr	NameExpr(*x*)	Returns the unevaluated expression of *x* rather than the evaluation of *x*. NameExpr is like Expr except that if *x* is a name, NameExpr returns the unevaluated expression stored in the name rather than the unevaluated name *x*.
Num	Num("*string*")	Converts a character string into a number.
Parse	Parse("*string*")	Converts a character string into a JSL expression.

Summary

Table 8.3 compares various ways that you can use the evaluation-control functions with *x*. Assume that a data table contains a column named X3, and *x* and *i* have been assigned:

```
x = Expr( Distribution(Column( Expr("X"||Char( i ) ) ) ) );
i = 3;
```

Table 8.3 Functions for Controlling Evaluation

Commands and results	Explanation

x; // or
Eval(x);
Not Found in access or evaluation of 'distribution' , Bad Argument({"X" || Char(i)})
	Eval(x) and simply calling x are equivalent.
	Evaluates the expression distribution(column(expr("X" || Char(i)))). This results in errors. The column name is recognized as "X"||Char(i) because it is packed by the Expr() function.

Expr(x); *x*	Returns the expression x (packs *an additional* layer).

Table 8.3 Functions for Controlling Evaluation *(Continued)*

Commands and results	Explanation
`Name Expr(x);` *`Distribution(Column(Expr("X" || Char(i))))`*	Returns the expression stored in *x* exactly as is: `Distribution(Column(Expr("X" || Char(i))))`.
`y=Eval Expr(x);` *`Distribution(Column("X3"))`*	Evaluates the inner expression but leaves the outer expression unevaluated, so that *y* is `Distribution(Column("X3"))`.

`y; //or`
`Eval(Eval Expr(x));`
`Distribution[]`
	`Eval(eval expr(x))` and simply calling `y` are equivalent.
	Evaluates `Distribution(Column("X3"))` to launch the platform.

`z = Char(nameexpr(x));` *`"Distribution(Column(Expr (\!"X\!" || Char(i))))"`*	Quotes the entire expression as a text string, adding `\!"` escape characters as needed. Note that `Char(x)` would first attempt to evaluate *x*, producing an error and ultimately returning a quoted missing value: `"."`
`Parse(z);` *`Distribution(Column(Expr("X" || Char(i))))`*	Unquotes the text string and returns an expression.

`a = Parse(Char( NameExpr(x)));`
`Eval(EvalExpr(a));`
`Distribution[]`
	Evaluation control taken to its logical extreme.
	Note that you must break this into at least two steps as shown. Combining it into one giant step produces different results because the `Eval Expr` layer causes the `Parse` layer to be copied literally, not executed.
	`Eval(`
	`EvalExpr(`
	`Parse(`
	`Char(`
	`NameExpr(x)))));`
	`Distribution(Column(Expr("X" || Char(i))))`

Macros

Stored expressions can serve as a macro feature. You can store a generalized action as an expression in a JSL variable, and then call that variable wherever you need that action to be performed. This example has four macros as the arguments to `If`:

```
lastStdzdThickness=expr(
  (thickness[nrow()]-col mean(thickness)) / col std dev(thickness));
continue=expr(...<script to read in more data>...);
log=expr(print("In control at "||char( long date(today()))));
break=expr(...<script to shut down process>...); limitvalue=1;

if(lastStdzdThickness<limitvalue,log;continue,break);
```

Storing the expression with `Expr` delays its evaluation until the variable is evaluated. Any variables, data points, or expressions included in that expression are evaluated on the fly when the expression is evaluated. See "Expressions" for detailed rules for storing expressions and later evaluating them.

Manipulating Lists

The following functions manipulate lists. They can also be used to manipulate expressions, as shown in the next section, "Manipulating Expressions". A summary of commands with explanations is in Table 8.4.

Most of the functions have two variants, one that produces a new value, and one that works in-place directly on its arguments. Here are some examples:

```
// In-place example
A = {2, 3, 4, 2, 1, 2, 1};
A = Remove( A, 3 ); // delete the third item in the list A, storing result in A
Remove From( A, 3 ); // delete the third item in the list A, in place
Show( A ); // A = {2, 3, 2, 1, 2, 1};

// New value example
B = {2, 3, 4, 2, 1, 2, 1};
onetwo = Insert( {1}, 2 );  // {1, 2}
Insert Into( B, {1, 2}, 4 ); // put 1, 2 before the current 4th item
Show( B ); // B = {2, 3, 4, 1, 2, 2, 1, 2, 1};
```

Note: If the position is omitted in the `Insert Into` command, items are placed at the end of the list.

```
a = Shift( {1,2,3,4}, 1 ); // store the list {2,3,4,1} in a
Shift Into( a, -1 );
Show( a ); // a = {1, 2, 3, 4};
```

```
b = Reverse( a ); // b is now {4,3,2,1}
Reverse Into( a ); // a is now {4,3,2,1}
Show( a ); // a = {4, 3, 2, 1};

s = Sort List( {1,4,2,5, -7.2, Pi(), -11, cat, apple, cake} );
Show( s ); // s = {-11, -7.2, 1, 2, 4, 5, apple, cake, cat, Pi()};

c = {5, pie, 2, Pi(), -2};
Sort List Into( c );
Show( c ); // c = {-2, 2, 5, Pi(), pie};
```

In-place functions

In-place functions are those that operate on lists or expressions directly. They have `From` or `Into` in their names (for example, `Remove From` and `Insert Into`). They do not return a result; you have to show the list to see the result. The first argument for an in-place function must be an L-value. An *L-value* is an entity such as a variable whose value can be set.

```
myList = {a, b, c, d};
Insert Into( myList, 2, 3 );
Show( myList );
   myList = {a, b, 2, c, d}
```

These examples show how to use `Insert Into` and `Remove From` with nested lists:

```
a = {{1, 2, 3}, {"A", "B", "C"}};
Show( a );
   a = {{1, 2, 3}, {"A", "B", "C"}}

Insert Into( a[1], 99, 1 );
Show( a );
   a = {{99, 1, 2, 3}, {"A", "B", "C"}}

Remove From( a[1], 1 );
Show( a );
   a = {{1, 2, 3}, {"A", "B", "C"}}
```

Not in-place functions

For the *not-in-place functions,* you must either state the list directly or else quote a name that evaluates to a list. Such functions do *not* have `From` or `Into` in their names. They *return* manipulated lists or expressions without changing the original list or expression given in the first argument.

```
myNewList = Insert( {a, b, c, d}, 2, 3 );
   {a, b, 2, c, d}

oldList = {a, b, c, d};
newList = Insert( oldList, 2, 3 );
```

```
{a, b, 2, c, d}
```

Substituting

`Substitute()` and `Substitute Into()` merit further discussion. Both functions find all matches to a pattern in a list (or expression) and replace them with another expression. Each pattern must be a name. The arguments are evaluated before they are applied, so most of the time you must quote them with an `Expr()` function.

```
Substitute( {a,b,c}, Expr( a ), 23); // returns {23,b,c}
Substitute( Expr( Sine( x ) ), Expr( x ), Expr( y ) ); // returns Sine(y)
```

To delay evaluating an argument, use `NameExpr` instead of `Expr`:

```
a = {quick, brown, fox, jumped, over, lazy, dogs};
b = Substitute( a, Expr( dogs ), Expr( cat ) );
canine = Expr( dogs );
equine = Expr( horse );
c = Substitute( a, Name Expr( canine ), Name Expr( equine ) );
Show( a, b, c );
   a = {quick,brown,fox,jumped,over,lazy,dogs}
   b = {quick,brown,fox,jumped,over,lazy,cat}
   c = {quick,brown,fox,jumped,over,lazy,horse}
```

`Substitute Into` does the same work, in place:

```
Substitute Into( a, Expr( dogs ), Expr( horse ) );
```

You can list multiple pattern and replacement arguments to do more than one replacement in a single step:

```
d = Substitute( a,
   Name Expr( quick ), Name Expr( fast ),
   Name Expr( brown ), Name Expr( black ),
   Name Expr( fox ), Name Expr( wolf )
);
   {fast,black,wolf,jumped,over,lazy,dogs}
```

Note that substitutions are done repeatedly over multiple instances of the expression pattern. For example:

```
Substitute( Expr( a + a ), Expr( a ), Expr( aaa ) );
```

results in:

```
aaa + aaa
```

Manipulating Expressions

The functions for manipulating lists can also operate on most expressions. Be sure to quote the expression with `Expr()`. For example:

```
Remove( Expr( A + B + C + D ), 2 ); // results in the expression A + C + D
b = Substitute( Expr( Log( 2 ) ^ 2 / 2 ), 2, 3 ); /* returns the expression
  Log( 3 ) ^ 3 / 3*/
```

As with lists, remember that the first argument for in-place functions must be an L-value. An L-value is an entity such as a variable whose value can be set. In-place functions are those that operate on lists or expressions directly. They have `From` or `Into` in their names (for example, `Remove From` and `Insert Into`). They do not return a result; you have to show the expression to see the result.

```
polynomial = Expr( a * x ^ 2 + b * x + c );
Insert Into( polynomial, Expr( d * x ^ 3 ), 1 );
Show( polynomial );
  polynomial = d * x ^ 3 + a * x ^ 2 + b * x + c
```

For the not-in-place functions, you must either state the expression directly or else quote a name that evaluates to an expression using `NameExpr`. Such functions do *not* have `From` or `Into` in their names. They *return* manipulated lists or expressions without changing the original list or expression given in the first argument.

```
cubic = Insert( Expr( a * x ^ 2 + b * x + c ), Expr( d * x ^ 3 ), 1 );
  d * x ^ 3 + a * x ^ 2 + b * x + c

quadratic = Expr( a * x ^ 2 + b * x + c );
cubic = Insert( Name Expr( quadratic ), Expr( d * x ^ 3 ), 1 );
  d * x ^ 3 + a * x ^ 2 + b * x + c
```

Substituting

Substituting is extremely powerful; please review the earlier discussion "Substituting". Here are a few notes regarding substituting for expressions.

`Substitute(`*`pattern`*`,`*`name`*`,`*`replacement`*`)` substitutes for names in expressions

`NameExpr()` looks through the name but copies instead of evaluates:

```
a = Expr(
  Distribution( Column( x ), Normal Quantile Plot )
);
Show( Name Expr( a ) );
  Name Expr(a) = Distribution(Column(x), Normal Quantile Plot);
```

`Substitute()` evaluates all its arguments, so they must be quoted correctly:

```
b = Substitute( Name Expr( a ), Expr( x ), Expr( :weight ) );
```

```
Show( Name Expr( b ) );
  Name Expr(b) = Distribution(Column(:weight), Normal Quantile Plot);
```

SubstituteInto() needs an L-value, so the first argument is not quoted:

```
Substitute Into( a, Expr( x ), Expr( :weight ) );
Show( Name Expr( a ) );
  Name Expr(a) = Distribution(Column(:weight), Normal Quantile Plot);
```

Substitute() is useful for changing parts of expressions, such as in the following example that tests the Is functions:

```
data = {1, {1, 2, 3}, [1 2 3], "abc", x, x( y )};
ops = {is number, is list, is matrix, is string, is name, is expr};
m = J( N Items( data ), N Items( ops ), 0 );
test = Expr(
  m[r, c] = _op( data[r] )
);
For( r = 1, r <= N Items( data ), r++,
  For( c = 1, c <= N Items( ops ), c++,
    Eval( Substitute( Name Expr( test ), Expr( _op ), ops[c] ) )
  )
);
Show( m );
  m =
  [1 0 0 0 0 0,
  0 1 0 0 0 1,
  0 0 1 0 0 0,
  0 0 0 1 0 0,
  0 0 0 0 1 1,
  0 0 0 0 0 1];
```

You can use SubstituteInto() to have JMP solve quadratic equations. The following example solves $4x^2 - 9 = 0$:

```
/* FIND THE ROOTS FOR THE EQUATION:
       a*x^2 + b*x + c = 0
The quadratic formula is x=(-b + - sqrt(b^2 - 4ac))/2a.
Use a list to store both the + and - results of the +- operation */
x = {Expr(
  (-b + Sqrt( b ^ 2 - 4 * a * c )) / (2 * a)
), Expr(
  (-b - Sqrt( b ^ 2 - 4 * a * c )) / (2 * a)
)};
Substitute Into( x, Expr( a ), 4, Expr( b ), 0, Expr( c ), -9 );

// plug in the coefficients
Show( x ); // see the result of substitution
Show( Eval Expr( x ) ); // see the solution
  x = {Expr(((-0) + Sqrt(0 ^ 2 - 4 * 4 * -9)) / (2 * 4)), Expr(((-0) - Sqrt(0
    ^ 2 - 4 * 4 * -9)) / (2 * 4))};
```

```
Eval Expr(x) = {1.5, -1.5};
```

The functions for manipulating lists and expressions are discussed in the previous section, "Manipulating Lists", and summarized in Table 8.4.

Table 8.4 Functions for Manipulating Lists or Expressions

Function	Syntax	Explanation

`Remove`	`x = Remove(list|expr)`
	`x = Remove(list|expr, position)`
	`x = Remove(list|expr, {positions})`
	`x = Remove(list|expr, position, n)`
		Copies the *list* or *expression*, deleting the item(s) at the indicated *position*. If *position* is omitted, items are deleted from the end. *Position* can be a list of positions. An extra argument, *n*, deletes *n* items instead of just 1.

`Remove From`	`Remove From(list|expr, position)`
	`Remove From(list|expr)`
	`Remove From(list|expr, position, n)`
		Remove items in place. The function returns the removed item(s), but you do not have to assign them to anything. The first argument must be an L-value.

`Insert`	`x = Insert(list|expr, item, position)` `x = Insert(list|expr, item)`	Inserts a new item into the *list* or *expression* at the given position. If position is not given, it is inserted at the end.
`Insert Into`	`Insert Into(list|expr, item, position)` `Insert Into(list|expr, item)`	Same as `Insert`, but does it in place. *List* or *expression* must be an L-value.
`Shift`	`x = Shift(list|expr)` `x = Shift(list|expr, n)`	Shift an item or *n* items from the front to the back of the *list* or *expression*. Shift items from back to front if *n* is negative.

Table 8.4 Functions for Manipulating Lists or Expressions *(Continued)*

Function	Syntax	Explanation
Shift Into	Shift Into(*list*|*expr*) Shift Into(*list*|*expr*, *n*)	Shift items in place.
Reverse	*x*=Reverse(*list*|*expr*)	Reverse the order of elements of a *list* or terms of an *expression*.
Reverse Into	Reverse Into(*list*|*expr*)	Reverse the order of elements of a *list* or terms of an *expression* in place.
Sort List	*x*=Sort List(*list*|*expr*)	Sort the elements of a *list* or the terms of an *expression*. Numbers sort low, followed by the name value of names, strings, or operators. For example 1+2 is lower than 1-2 because the name value Add sorts lower than the name value Subtract. {1,2} sorts lower than {1,3}, which sorts lower than {1,3,0}. {1000} sorts lower than {“a”}, but {a} and {“a”} sort as equal.
Sort List Into	Sort List Into(*list*|*expr*)	Sort the elements of a *list* or terms of an *expression* in place.
Sort Ascending	Sort Ascending(*list*|*matrix*)	Returns a copy of a *list* or *matrix* with the items in ascending order.
Sort Descending	Sort Descending(*list*|*matrix*)	Returns a copy of a *list* or *matrix* with the items in descending order.
Loc Sorted	Loc Sorted(*A*, *B*)	Creates a matrix of subscript positions where the values in matrix **A** match the values in matrix **B**. **A** must be a matrix sorted in ascending order.
Substitute	R = Substitute(*list*|*expr*, Expr(*pattern*), Expr(*replacement*), ...)	Finds all matches to the pattern in the *list* or *expression*, and replaces them with the replacement expression. Each pattern must be a name. The second and third arguments are evaluated before they are applied, so most of the time you must quote them with an Expr function. To delay evaluating an argument, use Name Expr instead of Expr. You can list multiple pattern-replacement pairs for multiple substitutions in one statement.

Table 8.4 Functions for Manipulating Lists or Expressions *(Continued)*

Function	Syntax	Explanation
Substitute Into	Substitute Into(*list|expr*, Expr(*pattern*), Expr(*replacement*), ...)	Substitute in place.

Advanced Scoping and Namespaces

Scripts that are used in production environments need to use more advanced scoping techniques to avoid collisions between scripts. JMP provides three progressively more advanced techniques:

- The `Names Default To Here()` function. If you have simple scripting needs, this single command might be sufficient. See "Names Default To Here".
- Scopes that are pre-defined by JMP. See "Scoped Names".
- Namespaces that you can create for your scripts. See "Namespaces".

Names Default To Here

If you write production scripts, you need to insulate the script from the current user environment. Otherwise, the variables that you use might interact with variables used by the user and by other scripts. The way to do this is to keep your names in a local environment, which you can do by setting an execution mode with the statement:

```
Names Default To Here( 1 );
```

Unqualified names in a script with the *Names Default To Here* mode turned on are private to that script. However, the names persist as long as the script persists, or as long as objects created by or holding the script are still active. It is recommend that all production scripts start with `Names Default To Here(1)` unless there is a specific reason not to do so. When the script uses an unqualified name in this mode, that name is resolved in the local namespace.

To refer to global variables, scope the name specifically as a global variable (for example, `::global_name`). To refer to columns in a data table, scope with name specifically as a data table column (for example, `:column_name`).

Note: `Names Default To Here( 1 )` defines a mode for a particular script. It is not a global function. One script can have this mode turned on, while another script can have it turned off. The default setting is off.

In JMP 8 and earlier, the only method to the insulate scripts was to use lengthy names that were less likely to collide with names in other scripts. Using `Names Default To Here(1)` makes this technique unnecessary.

If you have simple scripting needs, `Names Default To Here(1)` might be sufficient.

Local()

`Local()` creates local scopes only in specific contexts within a script and cannot enclose a longer script with interacting functions, while `Names Default To Here(1)` creates a local scope for an entire script. In `Local()`, you list the variables that are local to the local block. Anything not explicitly listed is not local to that block.

Local Here()

`Local Here()` provides a namespace block with `Names Default to Here(1)`. Use `Local Here()` to prevent name collisions when multiple scripts are executed from the same root namespace (for example, when a script executes two button scripts that have the same variables or an included script is run from the main script).

`Local()` does not always work due to the lifetime of the local block, but `Local Here()` is persistent across the call.

Here's an example that runs an included script from a main script. The scripts define x and y variables as different values. Variables in the included script are in the Local Here namespace.

Main.jsl:

```
Names Default To Here( 1 );

x = 5;
y = 0;

Print( "Main, Before: x=" || Char( x ) || ", y=" || Char( y ) );
Include( "Inc1.jsl" );
Print( "Main, After: x=" || Char( x ) || ", y=" || Char( y ) );
```

Inc1.jsl:

```
Names Default To Here( 1 );

Local Here( // variables are local to this script
  x = 100;
  y = 100;

Print( "Inc1: x=" || Char( x ) || ", y=" || Char( y ) );
);
Print( "Inc1 outside: x=" || Char( x ) || ", y=" || Char( y ) );
```

Return:

```
"Main, Before: x=5, y=0" // defined in Main.jsl, before running Inc1.jsl
"Inc1: x=100, y=100" // defined in Inc1.jsl
"Inc1 outside: x=5, y=0" // from Inc1.jsl, referring to Main.jsl
"Main, After: x=5, y=0" // from Main.jsl, after running Inc1.jsl
```

Notice that the x and y variables in inc1.jsl don't change because they're in the Local Here namespace.

Handling Unqualified Named Variable References

The `Names Default To Here()` function determines how *unqualified* named variable references are resolved. Explicitly scoping a variable using `here:var_name` always works, whether `Names Default To Here()` is on or off. See "Scoped Names" for more information about `here` and other scopes.

Enabling the *Names Default To Here* mode associates a scope called `Here` with an executing script. The `Here` scope contains all of the unqualified named variables that are created when they are the target of an assignment (as an L-value). In JMP 8 and earlier, these variables normally would have been placed in the `Global` scope. Using a `Here` scope keeps variables in multiple executing scripts separate from each other, avoiding name collisions and simplifying the scripting and management of variable name collisions. You can still share information using the `Global` scope.

Names Default To Here and Global Variables

Run this example script one line at a time to see how the `Names Default To Here()` function changes the resolution of variable names.

Example Script

```
a = 1;
Names Default To Here( 1 );
a = 5;
Show( global:a, a, here:a );
   global:a = 1;
   a = 5;
   here:a = 5;
```

1. Run the first line to create a global variable named *a* that holds the value 1.
2. Run the second line to turn on the *Names Default To Here* mode.
3. Run the third line to create a new variable named *a* in the local space that holds the value 5. This line does *not* change the value assigned to the global variable *a*.
4. Run the fourth line to see how scoped and unscoped variables are resolved.

The unqualified *a* is resolved to `here:a`. If `Names Default To Here()` were not on, *a* would be resolved to the global variable named *a*.

Note that if you use `::a` instead of `global:a` in the `Show()` function, your output is a little different:

```
Show(::a, a, here:a);
   a = 1;
   a = 5;
   here:a = 5;
```

Example of Using the Names Default To Here() Function

You have two scripts with the following definitions, and`Names Default To Here()` is turned *off* (the default condition) in both scripts.

Note: Both scripts must be in separate script windows for this example.

```
a = 1; // script 1
Show( a );

a = 3; // script 2
Show( a );
```

1. Run Script 1. Here is the result:

   ```
   a = 1
   ```

2. Run Script 2. Here is the result:

   ```
   a = 3
   ```

3. Run only the `show(a);` line in Script 1. Here is the result:

   ```
   a = 3
   ```

The log shows `a = 3` because variable *a* is global, and was last modified by Script 2. This is the default behavior in JMP 9 and later, and it is the only possible behavior in JMP 8 and earlier.

4. Now turn on `Names Default To Here()` in *both* scripts.

   ```
   Names Default To Here(1);
   ```

   Note: `Names Default To Here()` is local to a particular script. It is *not* a global setting.

5. Run Script 1. Here is the result:

   ```
   a = 1
   ```

6. Run Script 2. Here is the result:

   ```
   a = 3
   ```

7. Run only the `Show( a );` line in Script 1. Here is the result:

   ```
   a = 1
   ```

The log shows `a = 1`, because a copy of variable *a* is maintained for each script.

Note: Problems using this function are generally due to the mixing of unqualified and qualified references to global variables. Always explicitly scoping a name prevents accessing an unintended variable.

Scoped Names

Specify where a name is to be resolved by using a scope in the form `scope:name` where `scope` indicates how to resolve the name. For example, `here:name` indicates that the name should be resolved locally. Using the *Names Default To Here* mode, `here:name` is equivalent to `name`. The scope instructs how to look up the name.

The syntax is to use the colon scope operator:

```
scope:name
```

There are several types of scopes:

- Scope can be a resolution rule. For example, `here:x` means that `x` should be resolved to a name that is local to the script. `Global:x` means that `x` should be resolved to a global name.
- Scope can be a namespace reference variable. For example. `ref:a` means that `a` should be resolved within the namespace that `ref` refers to.
- Scope can be a data table reference to look up names as column names. For example, `dt:height` means that `height` should be resolved as a column in the data table that `dt` references.
- Scope can be the name of a namespace that you created. For example, `myNamespace:b` where `myNamespace` is a namespace that you created. `"myNamespace":b` is equivalent. See "Namespaces".

Note: If you create a function, make sure you scope it to avoid conflicts with built-in functions.

Examples of Scoping Column Formulas

The following examples demonstrate how to scope columns that contain formulas. In both scripts, *x* is a global variable, local variable, and column name.

In the first script, the column name x is unscoped. the formula in the second column multiplies the value in column `x` by 100. In this case, The result is a column with the values 100, 200, and 300.

```
::x = 5;
```

```
New Table( "Test",
   New Column( "x", Values( [1, 2, 3] ) ),
   New Column( "y", Formula( 100 * x ) ),
);
```

In the following script, the formula in column y assigns 500 to x and then adds 50 to x. Each cell in the column contains the value 550.

```
::x = 5;
New Table( "Test",
   New Column( "x", Values( [1, 2, 3] ) ),
   New Column( "y", Formula( Local( {x = 500}, x + 50 ) ) ),
);
```

Predefined Scopes

JMP provides predefined that cannot be removed or replaced. Each of these scopes has specific roles, depending on its associated object.

Table 8.5 Predefined Scopes

Scope	Description
`Global`	Global names are shared throughout the JMP environment.
`Here`	Scope of the executing script.
`Builtin`	JMP built-in functions. For example, `Builtin:Sqrt()`. These names are shared throughout the JMP environment. If you over-ride a JSL function with a custom function, you can still access the built-in JSL function by using this scope.
`Local`	Nearest local scope. Can be nested within the user-defined functions, `Local` and `Parameter`.
`Local Here`	Provides a namespace block inside `Names Default to Here(1)`. Use `Local Here()` to prevent name collisions when multiple scripts are executed from the same root namespace (for example, when a script executes two button scripts that have the same variables). The argument can be any valid JSL expression. `Local( {Default Local}, )` does not always work due to the lifetime of the local block, but `Local Here()` is persistent across the call.
`Window`	Scope of the containing user-defined window. (Rare.)
`Platform`	Scope of the current platform. (Rare.)

Table 8.5 Predefined Scopes *(Continued)*

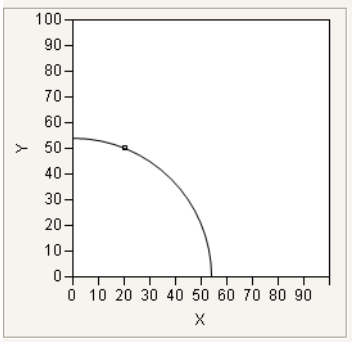
Scope	Description
Box	Scope of the containing context box. A context box is nested within a user-defined window. (Rare.)

Example of Using the Window Scope

This example uses the `Window` scope to pass information during execution. Explicitly scoping the variables `x` and `y` to this window ensures that JMP does not try to scope `x` and `y` in other contexts, such as a data table. The variables `x` and `y` are created and used solely inside the `Window` environment. The `Window` scope is similar to using `Local()`, but more useful because `Local()` is limited in the places that it can be used.

```
New Window( "Example",
   window:gx = 20;
   window:gy = 50;
   Graph Box(
      Frame Size( 200, 200 ),
      Handle(
         window:gx,
         window:gy,
         Function( {x, y},
            window:gx = x;
            window:gy = y;
         )
      );
      Circle( {0, 0}, Sqrt( window:gx * window:gx + window:gy * window:gy ) );
   );
);
```

Figure 8.1 Example of Current Window Namespace



Example of Using the Here Scope

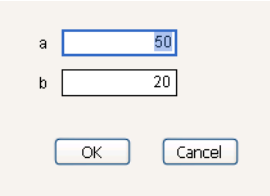
This example uses the `Here` scope to pass information between windows that are created by the same script. Scoping a variable using `Here:` is not dependent on turning `Names Default To Here()` on. The `Here:` scope is always available.

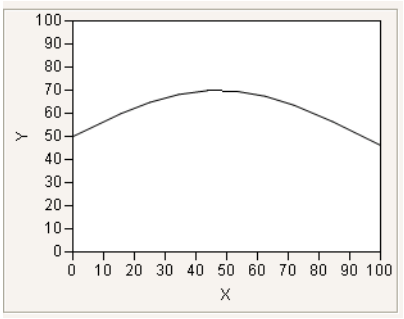
This script produces two windows and uses two different scopes.

The Launcher window asks the user for two values. Those two values are passed to the Output window, which uses them to graph a function. The Launcher window scopes `aBox` and `bBox` to the window: essentially, those two variables (pointers to Number Edit Boxes) exist only in the Launcher window and are not available to the Output window. The values from those two boxes are then copied into variables that are scoped to `Here`, and so are available to both windows that are produced by this script.

```
launchWin = New Window( "Launcher",
   <<Modal,
   V List Box(
      Lineup Box( N Col( 2 ), Spacing( 10 ),
         Text Box( "a" ),
         window:aBox = Number Edit Box( 50 ),
         Text Box( "b" ),
         window:bBox = Number Edit Box( 20 ),

      ),
      Lineup Box( N Col( 2 ), Spacing( 20 ),
         Button Box( "OK",
            // copy values before window goes away
            here:a = window:aBox << Get;
            here:b = window:bBox << Get;
         ),
         Button Box( "Cancel", Throw( 1 ) )
      ),

   )
);

New Window( "Output",
   Graph Box( Y Function( here:a + here:b * Sin( x / 30 ), x ) )
);
```

Figure 8.2 Launcher and Output





Namespaces

A *namespace* is a collection of unique names and corresponding values. You can store references to namespaces in variables. Namespace names are global, because JMP has only one namespace map. Namespace references are variables like any other variable that references an object, so they must be unique within their scope or namespace. The members of a namespace are referenced with the `:` scoping operator, such as `my_namespace:x` to refer to the object that is named `x` within the namespace called `my_namespace`. See "User-Defined Namespace Functions" for more information about creating and managing your own namespaces. Namespaces are especially useful for avoiding name collisions between different scripts.

User-Defined Namespace Functions

Create your own namespaces to hold related sets of variables and function definitions. There are several functions that you can use to manage namespaces.

New Namespace

```
nsref = New Namespace( <"nsname">, <{ name = expr, ... }> );
```

Creates a new namespace called `nsname` (a string expression) and returns a reference to the namespace. All arguments are optional.

`Nsname` is the name of the namespace in the internal global list of namespace names. `Nsname` can be used as the prefix in a scoped variable. The function returns a reference to the namespace, and can also be used as the prefix in a scoped variable reference. If `nsname` is absent, the namespace is anonymous and is given a unique name created by JMP. `Show Namespace()` shows all namespaces and their names, whether assigned or anonymous.

Important: If you already have a namespace named `nsname`, it is replaced. This behavior means that while you are developing your script, you can make your changes and re-run the script without having to clear or delete your namespace. To avoid unintentional replacement, you can either use anonymous namespaces, or test to see whether a particular namespace already exists:

```
If( !Namespace Exists( "nsname" ), New Namespace( "nsname" ) );
```

A list of named expressions is optional. The names are JMP variables that exist only within the namespace.

Note: The named expressions must be a comma-separated list. Separating the expressions with semi-colons causes the list to be ignored.

These namespaces must be uniquely named to prevent collisions in situations where multiple user-defined namespaces are being used. Using anonymous namespace names prevents collisions.

Namespace

```
nsref = Namespace( "nsname" | nsref);
```

Returns a namespace reference. The argument might be either of the following:

- a quoted string that contains a namespace
- a reference to a namespace

Note: `Namespace()` returns a reference to a namespace that already exists. It does not create a new namespace.

Is Namespace

```
b = Is Namespace( nsref );
```

Returns `1` (`true`) if *`nsref`* is a namespace or `0` (`false`) otherwise.

As Scoped

```
b = As Scoped( "nsname", var_name);
nsname:var_name;
```

`As Scoped()` is the function form of a scoped reference. The function returns a reference to the specified variable in the specified scope.

Namespace Exists

```
b = Namespace Exists( "nsname" );
```

Returns `1` (`true`) if *`nsname`* exists in the list of global namespaces, or `0` (`false`) otherwise.

Show Namespaces

```
Show Namespaces();
```

Shows the contents of all namespaces contained in the list of global namespaces. Namespaces are not visible unless a reference is made to one, using either the `New Namespace` or `Namespace` functions.

Namespace Messages

In addition to the namespace management functions, a namespace also supports a set of messages to access and manipulate its contents.

Note that these messages, as with all message, must be sent to a scriptable object. A namespace name is not a defined scriptable object and cannot be used in a Send operation. However, you can use the name of a namespace in variable references. For example, `nsmane::var` is equivalent to `nsref::var`.

See the *JSL Syntax Reference* for descriptions of the messages that are supported by user-defined namespace references.

Using Namespace References

The following are all equivalent references to a variable that is named `b` in the namespace that is named `nsname` that has a reference `nsref`:

```
nsref:b
nsname:b
"nsname":b
nsref["b"]
nsref<<Get Value("b") // used as an r-value
```

Namespaces and Included Scripts

An included script runs in the namespace of the parent script. If the included script has its own namespace definitions, you need to do one of the following:

- manage the namespace names to avoid name collisions
- use anonymous names created by the `New Namespace` function

In either case, you still need to manage variable references to namespaces.

There is also an option for the `Include` function (`New Context`) that creates a namespace that the included script runs in. This namespace is anonymous and it is independent from the parent script's namespace. For example:

```
Include("file.jsl", <<New Context);
```

This anonymous namespace can be referenced using `Here`.

If both the parent and included scripts use the global namespace, add `Names Default To Here` to the `New Context` option to avoid name collisions. For example:

```
Include("file.jsl", <<New Context, <<Names Default To Here);
```

See "Includes" for more information about the `Include` function.

Examples of User-Defined Namespaces

Creating and Using a Basic Namespace with Expressions

This example shows creating an anonymous namespace and using functions and variables within it.

```
new_emp = New Namespace(
   {name_string = "Hello, *NAME*!",

   print_greeting = Function( {a},
      Print( Substitute( new_emp:name_string, "*NAME*", Char( a ) ) )
   )}
);
```

Note that you must use the fully qualified name for variables defined within the namespace.

```
new_emp:print_greeting( 6 );
   "Hello, 6!"
```

Complex Number Operations

This example creates a namespace that contains functions to support using two-element lists to represent complex numbers, and then locks the namespace.

```
If( !Namespace Exists( "complex" ),
   New Namespace(
      "complex"
   );
   complex:makec = Function( {a, b},
      Eval List( {a, b} )
   );
   complex:addc = Function( {a, b}, a + b );
   complex:subtractc = Function( {a, b}, a - b );
   complex:multiplyc = Function( {a, b},
```

```
      Eval List( {a[1] :* b[1] - a[2] :* b[2], a[1] :* b[2] + a[2] :* b[1]} )
   );
   complex:dividec = Function( {a, b},
      d = b[1] ^ 2 + b[2] ^ 2;
      Eval List(
         {a[1] :* b[1] - a[2] :* b[2] / d, a[2] :* b[1] - a[1] :* b[2] / d}
      );
   );
   complex:charc = Function( {a},
      Char( a[1] ) || "+" || Char( a[2] ) || "i"
   );
);
Namespace( "complex" ) << Lock;
```

Here are examples using functions that are within the above user-defined namespace.

```
c1 = complex:makec( 3, 4 );
c2 = complex:makec( 5, 6 );
cadd = complex:addc( c1, c2 ); // returns {8, 10}
csum = complex:subtractc( c1, c2 ); // returns {-2, -2}
cmul = complex:multiplyc( c1, c2 ); // returns {-9, 38}
cdiv = complex:dividec( c1, c2 ); // returns {14.6065573770492,
  19.7049180327869}
Show( complex:charc( c1 ) ); // returns complex:char(c1) = "3+4i";
cm1 = complex:makec( [1, 2, 3], [4, 5, 6] ); // returns {[1, 2, 3], [4, 5, 6]}
```

Referencing Namespaces and Scopes

There are a number of factors in resolving a named variable reference. Table 8.6 describes the named variable references that are resolved for specific situations.

Notes:

- In JMP 9 and later, `a`, `:a`, and `::a` have the same meaning with the *Names Default To Here* mode turned off.
- If the current point of execution is in a user-defined function, or a Local or Parameter JSL function body, then the Local namespace is used for creation when `Default Local` is specified. However, the reference rule is unchanged.

Table 8.6 Namespace References

Form	Reference Type	Reference Rule	Creation Rule
`a`	Unqualified	If the *Names Default To Here* mode is on, JMP looks for the variable in these locations: • `Local` namespace • `Here` namespace • current data table If the *Names Default To Here* mode is off, JMP looks for the variable in these locations: • `Local` namespace • `Here` namespace • `Global` namespace • current data table	• If the *Names Default To Here* mode is on, then JMP creates the variable in the `Local` namespace or in the `Here` namespace. • If the *Names Default To Here* mode is off, then JMP creates the variable in the `Local` namespace or in the `Global` namespace.
`:a`	Current data table	JMP looks for the variable in the current data table.	(Not applicable)
`::a` `Global:a`	Global	JMP looks for the variable in the `Global` namespace.	JMP creates the variable in the `Global` namespace.

`ns:a`
`dt:a`
`Here:a`
`"name":a`
`expr:a`
	Qualified	JMP looks for the variable in the specified namespace. If the variable is not found, an error results.	JMP creates the variable in the specified namespace. Any previous values are replaced.

`ns["a"]` `ns[expr]`	Subscript	JMP looks for the variable in the specified namespace. If the variable is not found, an error results.	JMP creates the variable in the specified namespace. Any previous values are replaced.
`Platform:a`	Qualified	JMP looks for the variable in the encapsulating platform.	JMP creates the variable in the encapsulating platform.

Table 8.6 Namespace References *(Continued)*

Form	Reference Type	Reference Rule	Creation Rule
`Local:a`	Qualified	JMP looks for the variable within any nested local function, up to and including a function call boundary. See "Example of Local:a".	JMP creates the variable in the innermost nested local function or function call boundary.
`Window:a`	Qualified	JMP looks for the variable in the encapsulating `New Window` window namespace.	JMP creates the variable in the encapsulating `New Window` window namespace.
`Box:a`	Qualified	JMP looks for the variable in the encapsulating `Context Box` namespace contained in a `New Window` window.	JMP creates the variable in the encapsulating `Context Box` namespace contained in a `New Window` window.

Example of Local:a

Sample Script	Log Output

`Delete Symbols();`
`Local( {d1l1 = 12},`
`  local:f1f1 = Function( {fa1, fa2},`
`  {fl1 = 99},`
`  local:fa12 = fa1 + fa2;`
`  Local( {d2l1 = 56},`
`  local:l2l2 = 78;`
`  Show( fa12 );`
`  Show( fl1 );`
`  Try( Show( d1l1 ), Write(`
`"\!n\!n***Error=" || Char( exception_msg`
`) || "\!n" ) );`
`  Show Symbols();`
`  );`
`  local:fa12;`
`  );`
`  f1f1( 2, 3 );`
`);`
	fa12 = 5;
	fl1 = 99;
	dil1 = 12;
	
	// Local
	
	d2l1 = 56;
	l2l2 = 78;
	
	// 2 Local
	
	// Local
	
	fa1 = 2;
	fa12 = 5;
	fa2 = 3;
	fl1 = 99;
	
	// 4 Local
	
	// Local
	
	d1l1 = 12;
	
	// 1 Local
	
	5

Resolving Named Variable References

When variables are referenced within a JMP script, JMP resolves the storage location of the variable using a specific set of rules. If the variable is referenced by a qualified name, then the resolution is based on the specific qualification specification. If the variable is referenced by an unqualified name, the situation is a bit more complex. JMP looks through a hierarchy of scopes representing the point of execution with the executing script. This section describes the rules that are used to resolve named variable references.

By default, variable name resolution in JMP 9 and later worked the same way as in JMP 8 and earlier, allowing your current JSL scripts to be executed unchanged. For JMP 9 and later, the difference between qualified and unqualified variable named references is important to understand.

Qualified Named References

A qualified named reference uses the `:` and `::` operators to provide specific information about where a referenced variable resides, or where it is created. Examples of qualified named references include the following:

```
:var
::globalvar
datatable:var
nsref:var
"nsname":var
```

Unqualified Named References

An unqualified named reference provides no explicit information to completely identify where a variable resides or where it is created. No scoping operator (`:` or `::`) is specified in the reference. To change the behavior of JMP when resolving unqualified named variable references, use the `Names Default To Here(1)` function. For more information about variable name resolution, see the "Rules for Name Resolution".

Rules for Resolving Variable References

JMP applies rules in sequence to resolve variable references. See "Rules for Resolving Names".

Best Practices for Advanced Scripting

Minimize Polluting the Global Namespace and Prevent Scripts from Interacting

Always start your script with this line:

```
Names Default To Here(1);
```

Share Variables Across Scripts

Use named namespaces. Namespace names are placed in the global scope.

Use Anonymous Namespaces

Using namespace references to anonymous namespaces avoids possible conflicts with other namespaces.

Advanced Programming Concepts

This section covers some more advanced programming techniques that can be useful for developing scripts.

- "Throw and Catch Exceptions"
- "Functions"
- "Recursion"
- "Includes"
- "Load and Save Text Files"
- "Run External Programs"
- "Classes"

Throw and Catch Exceptions

A script can stop itself by executing the `Throw()` function. If you want to escape from part of a script when it is in an error condition, you can enclose it in a `Try()` expression.

If you include a character-valued expression as a `Throw()` argument, one of the following occurs:

- Throwing stores the string in a global variable named `exception_msg`. This is illustrated in the first example below.
- Throwing stores the string in a local variable named `exception_msg` if you define `exception_msg` in the Here namespace using `Names Default to Here(1)`.

`Try()` takes two expression arguments. It starts by evaluating the first expression, and if or when the first expression throws an exception by evaluating `Throw`, it does the following:

1. Immediately stops evaluating that first expression.
2. Returns nothing
3. Evaluates the second expression.

Examples

You can use `Try()` and `Throw()` to catch an exception that JMP itself throws.

```
Try( Throw( "Hello." ), Show( exception_msg ) );
  exception_msg = "Hello.";
```

The following example prints a message to the log when the `Try()` expression cannot be executed:

```
Try(
```

```
    dt = Open( "Mydata.jmp" ); // a file that cannot be opened
    Summarize( a = by( age ), c = count, meanHt = Mean( Height ) );
    Show( a, c, meanHt );
,
    Print( "This script does not work without the data set." );
    Throw();
);
```

If the Throw() string begins with "!" and is inside a Try() expression, throwing creates an error message about where the exception was caught.

```
Try(
    Throw( "!This message has an exclamation point." );
,
    Write( "\!Nexception_msg: ", exception_msg );
    Print( "Hello from the catch block (WITH exclamation)." );
);
Print( "This AFTER message will NOT print due to error." );
```

When writing the Try() error handler code, consider wrapping the exception variable exception_msg in a Char() function. If your script calls Concat() without wrapping it in a Char() call, you might get an error depending on what was actually returned in this variable. Though some commands return a string in exception_msg, other commands (such as Open()) return a list or another data structure. This would cause an error in the error handler if it were passed directly as an argument to Concat().

```
Try(
    Open( "no file" ),
    If( Is Empty( exception_msg ),
        // use Write to print "no error"if exception_msg is empty
        Write( "\!Nexception_msg (0): no error" ),
        // use Print to print the exception message
        Print( "\!Nexception_msg (1): ", exception_msg );
        /* or first make sure that exception message is a string
        and then use Concat to build the message that is passed to Print(). */
        Print( "\!Nexception_msg (2): " || Char( exception_msg ) );
    )
);
    exception_msg (1): "
    {"Cannot open table"(1, 2, "Open", Open /*###*/("no file"))}
    "
    exception_msg (2): {\!"Cannot open table\!"(1, 2, \!"Open\!", Open
        /*###*/(\!"no file\!"))}"
```

You can also use Try() and Throw() to escape from deep inside For loops.

```
a = [1 2 3, 4 5 ., 7 8 9];
b = a;
nr = N Row( a );
```

```
nc = N Col( a );
// a[2, 3] = 2; // uncomment this line to see the "Missing b" outcome

Try(
   sum = 0;
   For( i = 1, i <= nr, i++,
      For( j = 1, j <= nc, j++,
         za = a[i, j];
         If( Is Missing( za ),
            Throw( "Missing a" )
         );
         zb = b[j, i];
         If( Is Missing( zb ),
            Throw( "Missing b" )
         );
         sum += za * zb;
      )
   );
,
   Show( i, j, exception_msg );
   Throw();
);
   i = 2;
   j = 3;
   exception_msg = "Missing a";
```

You do not have to use `Try()` to make use of `Throw()`. In this example, `Throw()` is not caught by `Try()` but still stops a script that cannot proceed:

```
dt = New Table(); // to get an empty data table
If( N Row( dt ) == 0,
   Throw( "!Empty Data Table" )
);
```

Functions

JSL also has a function called `Function` to extend the macro concept with a local context arguments. Suppose that you want to create a function that takes the square root but tolerates negative arguments, returning zero rather than errors. You first specify the local arguments in a list with braces { } and then state the expression directly. You do not need to enclose the expression in `Expr` because `Function` stores it as an expression implicitly.

```
myRoot = Function( {x},
   If( x > 0, Sqrt( x ), 0 )
);
a = myRoot( 4 );  // returns 2
b = myRoot( -1 ); // returns 0
```

Functions are stored in variables, the same as values. This means that you cannot have both a root function and a root value. It also means that you can redefine a function anytime except when you are inside the function itself.

When a function is called, its arguments are evaluated and given to the local variables specified in the list forming the first argument. Then the body of the function, the second argument, is evaluated.

The values of the arguments are for the temporary use of the function. When the function is exited, the values are discarded. The only value returned is the return value. If you want to return several values, then return a list instead of a single value.

In defined functions, the stored function is not accessible directly, even by the `Name Expr` command. If you need to access the function expression in your script, you have to create the function within an `expr()` clause. For example,

```
makeFunction = Expr(
   myRoot = Function( {x},
      If( x > 0, Sqrt( x ), 0 )
   )
);
d = Substitute( Name Expr( MakeFunction ), Expr( x ), Expr( y ) );
Show( d );
makeFunction;
```

Optional Arguments

You can create optional arguments. For example:

```
f1 = Function( {x, y, z}); // all arguments are required
f2 = Function( {x, y=2, z=4}); // x is required, y and z are optional
```

Notes:

- To make an argument optional, give it a default value in the function's definition (x=1 rather than just x).
- Optional arguments always must follow a required argument. For example, this is not allowed:

  ```
  Function( {x = 1, y}, ... )
  ```

  It will work when defined, but you will always have to pass in a value for x and y, because the default value will never be used.
- When you call the function, you cannot skip optional arguments. In this example, you have to supply a value for y to supply a value for z when you call the function:

  ```
  ex = Function( {x, y = 2, z = 3},
    Return( x + y + z )
  );
  ex( 1, 4 ); // passes in 1 for x and 4 for y; z will be 3. returns 8.
  ```

Local Symbols

You can declare variables as local to a function so that they do not affect the global symbol space. This is particularly useful for recursive functions, which need to keep separate the values of the local variables at each level of function call evaluation.

As shown above, a function definition looks like this:

```
functionName=Function({arg1, ...}, body);
```

You can also have the function definition default all the unscoped names to be local.

```
functionName=Function({arg1, ...}, {Default Local}, body);
```

For the purposes of creation, the use of `Default Local` localizes all the names that meet the following qualifications:

- Are not scoped as globals (for example, `::name`)
- Are not scoped as data table column names (for example, `:name`)
- Occur without parentheses after them (for example, are not of the form `name(...)`)

For example, the following function sums three numbers.

```
add3 = Function( {a, b, c},
   {temp},
   temp = a + b;
   temp + c;
);
X = add3( 1, 5, 9 );
   15
```

The following function does the same thing, automatically finding locals.

```
add3 = Function( {a, b, c},
   {Default Local},
   temp = a + b;
   temp + c;
);
X = add3( 1, 5, 9 );
   15
```

In both cases, the variable `temp` is not a global, or, if it is already a global, remains untouched by evaluating the functions.

Using `Default Local` in user-defined functions can cause some confusion because it is context-sensitive. That is, the same function may behave differently in different contexts (depending on whether same-named outer variables are in scope) because the reference rules for unqualified names are unchanged when using `Default Local`. To make a function less context-sensitive, enumerate each and every variable that you want to be local. This reduces the confusion and the potential incorrect use of outer scope variable values.

Recursion

The `Recurse()` function makes a recursive call of the defining function. For example, you can make a function to calculate factorials. A factorial is the product of a number, the number minus 1, the number minus 2, and so on, down to 1.

```
myfactorial = Function( {a},
   If( a == 1,
      1,
      a * Recurse( a - 1 )
   )
);
myfactorial( 5 );
   120
```

You can define recursive calculations without using `Recurse()`. For example, you could replace `Recurse()` by `myfactorial`, and the script would still work. However, `Recurse()` offers these advantages:

- It avoids name conflicts when a local variable has the same name as the function.
- You can recurse even if the function itself has not been named (for example, assigned to a global variable, such as *myfactorial* above).

Includes

The `Include()` function opens a script file, parses the script in it, and executes the JSL in the specified file.

```
Include( "pathname" );
```

For example,

```
Include( "$SAMPLE_SCRIPTS/myStartupScript.jsl" );
```

There is an option to obtain the parsed expression from the file, rather than evaluating the file.

```
Include( "pathname", <<Parse Only );
```

Another named option creates a namespace that the included script runs in. This namespace is an anonymous namespace and it is independent from the parent script's namespace.

```
Include( "file.jsl", <<New Context );
```

See "Advanced Scoping and Namespaces" for information about using namespaces with your scripts.

Note the following about included files:

- JMP files aside from JSL cannot be used.

- Other recognized file types, such as image files, SAS data sets, and Microsoft Excel files cannot be used.
- Unrecognized file types are treated as a JSL file.
- Files with the .txt extension are treated as a JSL file. A text file that contains data can be included, however an error will appear since this is not valid JSL.

Load and Save Text Files

The `Load Text File()` and `Save Text File()` commands allow manipulation of text files from JSL. Note that the paths in the following code are strings.

```
text = Load Text File( "path" );
Save Text File( "path", text );
```

You can load a text file from a Web site:

```
Load Text File( "URL", <blob> );
```

The URL is a quoted string that contains the URL for the text file. The text file is returned as a string. If you add the optional named argument *blob*, a blob is returned instead.

Run External Programs

`Run Program()` enables you to control a program that uses standard input and standard output. Line-mode programs read commands from stdin and write answers to stdout. cmd.exe, the Windows command prompt, is one example. You can also use it to launch GUI programs, such as Windows Notepad.

```
Run Program(
   Executable( "cmd.exe" ),
   Options( {"/c", "TaskList\!n"} ),
   Read Function( "text" ) // returns all text
);
```

The preceding script writes the Windows task list and details to the log.

Figure 8.3 Log Output

```
"
Image Name                     PID Session Name        Session#    Mem Usage
========================= ======== ================ =========== ============
System Idle Process              0 Services                   0         24 K
System                           4 Services                   0     64,340 K
smss.exe                       516 Services                   0      1,568 K
```

You can also print the task list directly without using cmd.exe and then import the task list into a data table.

```
blob = Run Program(
   Executable( "tasklist.exe" ),
   Read Function( "blob" )
);

Open(
   blob,
   Columns(
      Column( "Image Name", Character, "Nominal" ),
      Column(
         "PID",
         Numeric,
         "Continuous",
         Format( "Best", 10 )
      ),
      Column( "Session Name", Character, "Nominal" ),
      Column(
         "Session#",
         Numeric,
         "Continuous",
         Format( "Best", 10 )
      ),
      Column( "Mem Usage", Character, "Nominal" ),
      Omitted Column( . )
   ),
   Import Settings(
      Fixed Column Widths( 26, 9, 17, 12, 12, 63 ),
      Labels( 1 ),
      Column Names Start( 2 ),
      Data Starts( 4 )
   )
);
```

Figure 8.4 Imported Data from Run Program()

	Image Name	PID	Session Name	Session#	Mem Usage
1	System Idle Proc...	0	Services	0	8 K
2	System	4	Services	0	42,560 K
3	smss.exe	456	Services	0	1,228 K

See the *JSL Syntax Reference*.

Note: If you use `Run Program()` to launch a GUI program, JMP is inoperable until you close the program.

Classes

Classes enable you to create objects based on templates. You can use JSL to create a definition of a class. A class definition is a template, which contains definitions for member variables, methods, and functions. After you have defined a class in JSL, you can instantiate an object that has all of the characteristics defined in the class definition.

Use the JSL function `Define Class()` to create a JSL class definition. Use the JSL function `New Object()` to instantiate an instance of a JSL class.

The JSL function `Define Class()` has a number of arguments that define the contents of the class:

```
Define Class( "class name",
   <Base Class( "base class name", <"base class name", ...> ),>
   <Show( All( Boolean ) ) | Show( <Members( Boolean ),> <Methods( Boolean ),>
   <Functions( Boolean )> ),>
   <Assignment Statements>
);
```

Where:

- `"class name"` is the name of the class being defined.
- `"base class name"` is one or more class names that make the set of base classes that are used as the foundation of the class being created.
- `Show` defines the value that appears when a class object is written out using the `Show()`, `Print()`, or `Write()` functions. Options include:
  - `All( Boolean )` shows all members, methods, and functions defined in a class object instance. The default value for All is true. If you specify the `All` option, that setting overrides any other arguments in the `Show` option.
  - `Members( Boolean )` shows all member variables contained in a class object instance.
  - `Methods( Boolean )` shows all methods contained in a class object instance.
  - `Functions( Boolean )` shows all functions contained in a class object instance.

  Note: You can specify either the `All` option or any number of the `Members`, `Methods`, and `Functions` options, but you cannot specify both options at the same time.

  Note: If a class contains a `_show_` method, that method overrides the `Show` option.

- `Assignment Statements` are zero or more of the following assignment statements, separated by semicolons:
  - `Member = JSL Expression`
  - `Method Name = Method( {<argument, ...>}, body )`

- `Function Name = Function( {<argument, ...}, {<local variable, ...>}, body )`

Note: The Assignment Statements can alternatively be specified as a list of JSL expressions.

There are three specialized methods that you can use in a class definition:

init Initializes the object at the time it is instantiated. This is considered the constructor for the class. You can have only one constructor per class.

show Defines the output of the `Show()`, `Print()`, and `Write()` JSL functions when an instance of the class is used as the argument.

to string Defines the output of the `Char()` JSL function when an instance of the class is used as the argument.

Notes:

- All methods, member variables, and functions are public.
- Functions defined within a class cannot access method variables or methods within the class.

Example of Defining a Class

The following example defines a simple class for a pet, containing the following information: name, breed, and color.

In this example, you define the constructor to require all information to create a new Pet object. Also, you define a set of accessor methods to obtain each of the attributes of a Pet object.

```
Define Class( "Pet",
   name = ""; breed = ""; color = "";
   _init_ = Method( { name, breed, color },
      this:name = name;
      this:color = color;
      this:breed = breed;
   );
   _show_ = Method( {},
      Return( name || " is a " || color || " " || breed || "." )
   );
   Get Pet Name = Method( {}, Return( name ) )
);
```

Instantiating a Class

To instantiate an instance of a JSL class object, use `New Object()` JSL function with any of the following functions and arguments:

```
New Object( "Class Name"( constructor arguments ) );
New Object( Class Name( constructor arguments ) );
New Object( Class Reference( constructor arguments ) );
New Object( "Class Name", { constructor arguments } );
New Object( Class Name, { constructor arguments } );
New Object( Class Reference, { constructor arguments } );
```

Examples of Instantiating and Using a Class

The following example shows how to instantiate a class to create and access JSL class objects.

```
class = "Pet";
xa = New Object( Pet( "Oliver", "Schnauzer", "Black" ) );
xb = New Object( class, { "Flynn", "Cockapoo", "Brown" } );
xc = New Object( class( "Mandi", "King Charles Cavalier", "Ruby" ) );
xd = New Object( "Pet"( "Charlie", "Asian", "White" ) );
xd1 = New Object( xd( "Fred", "Siamese", "Silver" ) );
xd2 = New Object( xd, { "Sam", "German Shepard", "Black and White" } );

Show( xa );
Show( xb );
Show( xc );
Show( xd );
Show( xd1 );
Show( xd2 );
xd2 = xd;                  // returns a second reference to xd
xdclone = xd << Clone;     // copy xd to a new instance, named xdclone
xd2:name = "Jerry";
Show( xd:Get Pet Name(), xd2:Get Pet Name(), xdclone:Get Pet Name() );
```

This example creates multiple instances of the Pet class. The output of the `Show()` JSL function is determined by the `_show_` method that is defined in the Pet class definition. This example also illustrates the difference between creating a second JSL reference to a class instance and cloning a class instance. When using the `Clone` message, you are creating a new class instance that is independent of the original. However, the new instance starts out as a copy of the original.

File and Directory Operations

`Pick Directory()` and `Pick File()` enable the user to select a file or directory when the script is run. To get a list of files in a directory, use `Files In Directory()`.

- "Select a Directory or File"
- "Get a List of Filenames"
- "Connect to a Network Directory"

Select a Directory or File

You can prompt the user to select a directory using the `Pick Directory()` function. The command displays a platform-specific window in which the user selects a folder. On Windows, the optional prompt string appears at the top of the Browse window (Windows) or below "Open" on the Finder window (macOS). You can also specify the path for an initial directory that appears, and specify whether files should appear in the Pick Directory window.

```
Pick Directory ( "Select a directory." );
```

To let the user select a file, use the `Pick File()` function:

```
Pick File(
  <"prompt message">, <"initial directory"> <{filter list}>,
  <first filter>, <save flag>, <"default file">,
  <multiple>)
```

The `"prompt message"` is used as the window title. The `"initial directory"` defines which folder initially appears. If a directory is defined as an empty string, the default directory is used.

You can also define the `{filter list}` used for the `Open()` window, forcing it to show only certain file types. This list must use the following syntax:

```
{"Label1|suffix1;suffix2;suffix3", "Label2|suffix4;suffix5"}
```

Each quoted string adds an entry to the **File name** list in the `Open()` window. `Label` defines the text that is displayed for each menu option. The following list of suffixes defines the file types that are displayed if its corresponding label is selected. Note the use of `"*"` to list all files in the window.

```
Pick File(
  "Select JMP File", // prompt message
  "$SAMPLE_DATA", // initial directory
  {"JMP Files|jmp;jsl;jrn", "All Files|*"}, // file filter list
  1, // intially selected item
  0, // doesn't prompt the user to save the file
```

```
    "Analgesics.jmp" // file that is selected by default
);
```

Tip: All arguments are optional, however, they are also positional. This means that you can leave out arguments only at the end of the script. Use empty strings for the arguments that you want to omit from the beginning of the script.

The script below does not set the default directory or the default file:

```
Pick File(
   "Select JMP File",
   "", // no default directory
   {"JMP Files|jmp;jsl;jrn", "All Files|*"},
   1,
   0,
   "" // no default file
);
```

The <first filter> argument sets the default selection where n is the index for the list item. In the script above, the <first filter> is the first item in the list: "JMP Files|jmp;jsl;jrn".

If <Save Flag> is false, the Multiple argument can be added to allow the user to select multiple files using the one window:

```
Pick File(
   "Select JMP File",
   "",
   {"JMP Files|jmp;jsl;jrn", "All Files|*"},
   1,
   0,
   "",
   multiple // save flag is 0, allows multiple file selection
);
```

Get a List of Filenames

To obtain a list of filenames in a specific directory, use the Files In Directory command.

```
Files In Directory( path, <recursive( 0 | 1 )> );
```

Both filenames and subdirectory names are returned as shown in the following example:

```
Files In Directory( "$SAMPLE_DATA" );
   { "2D Gaussian Process Example.jmp", "Abrasion.jmp", ... "Design
     Experiment", "Detergent.jmp", ... }
```

Notice that the files within the Design Experiment subdirectory are not included. And only files in the root $SAMPLE_DATA directory are listed.

To return a list of all file names, add the optional `recursive(Boolean)` argument to `Files In Directory`:

```
Files In Directory( "$SAMPLE_DATA", recursive( 1 ) );
   { "2D Gaussian Process Example.jmp", "Abrasion.jmp", ... "Design
     Experiment/2x3x4 Factorial.jmp", "Design Experiment/Algorithm Data.jmp",
     ... }
```

To get the full pathnames, recurse the directories and concatenate the file paths and file names. The following example loops through each file in the $SAMPLE_DATA directory and subdirectories. The file path is concatenated to each file name.

```
names = Files In Directory( "$SAMPLE_DATA", recursive( 1 ) );
For( i = 1, i <= N Items( names ), i++,
   names[i] = Convert File Path( "$SAMPLE_DATA" ) || names[i]
);
names;
   {"/C:/Program Files/SAS/JMP/16/Samples/Data/2D Gaussian Process
     Example.jmp",
   "/C:/Program Files/SAS/JMP/16/Samples/Data/Abrasion.jmp", ... }
```

The `Files in Directory` command accepts native and POSIX paths, as well as paths using path variables. See "Path Variables" for more information about working with paths.

Connect to a Network Directory

On Windows, you can connect to a network directory without connecting to the shared drive first.

```
Files in Directory( "\\myserver.company.com/source/Users/Smith" );
```

On macOS, you have to mount the volume first and then connect to the directory using a combination of JSL and AppleScript. Note that you can also modify this script to work on Windows. On macOS only, the user is prompted for a password.

```
mount = Function( {server},
   {Default Local},

   If( Host is( "Mac" ),

      /* Use osascript to mount the volume.
      The user will be prompted for credentials. */
      Run Program(
         Executable( "/usr/bin/osascript" ),
         Options( {"-e", Eval Insert( "mount volume \!"smb://^server^\!"" )} ),
         Read Function( "text" )
      );
      server = Concat Items( Remove( Words( server, "/" ), 1 ), "/" );
      Eval Insert( "/Volumes/^server^" );
```

```
    ,
       // Windows can use UNC to auto-mount the server.
       Eval Insert( "\!\\!\^server^" )
    )
);


users = Mount( "myserver.company.com/Users" );
Files In Directory( Eval Insert( "^users^/Smith" ) );
```

Scripting By Groups

By group arguments are supported for these functions: ColMean(), ColStdDev(), ColNumber(), ColNMissing(), ColMinimum(), ColMaximum().

Any number of By arguments can be specified, and you can use expressions for the By arguments. By arguments must be used in a column formula, or in the context of ForEachRow(). The first argument can also be a general numeric expression.

Here is an example:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Mean of height by sex", Numeric, Formula( Col Mean(
   :height, :sex ) ) );

dt << New Column( "Minimum of height by sex and age",
   Numeric,
   Formula( Col Minimum( :height, :sex, :age ) )
);

dt << Distribution( Continuous Distribution( Column( :height ) ), By( :sex )
   );

dt << Tabulate(
   Show Control Panel( 0 ),
   Add Table(
      Column Table(
         Analysis Columns( :height ),
         Statistics( Mean, N, Std Dev, Min, Max, N Missing )
      ),
      Row Table( Grouping Columns( :age, :sex ) )
   )
);
```

Encrypt and Decrypt Scripts

To add a basic level of protection to scripts, you can encrypt it so only someone who knows the password can view it; you can also require a password to run it. This is useful in situations when you want to implement controlled sharing of a script.

To encrypt a script:

1. Open the script that you want to encrypt.
2. Select **Edit > Encrypt Script**.
3. Enter a decrypt password so that the user needs a password to view the script.
4. (Optional) Enter a run password to require the user to enter a password before running the encrypted script.

   Note: The passwords must consist of single-byte characters; using a text Input Method Editor (IME) does not work.

5. Click **OK**.
6. If you entered only a decrypt password, click **Yes** to confirm that you do not want to assign a run password.

   The encrypted script opens in a new window. For example:

```
//-e6.0.2
S@FTQ;VGMUTF?J<;LS;B<=IRLXCU=BV;@NS<TW;LR<ZFOP=JJS>NNDA@T<V><DZA>SU@MG;LR<ZFO
   P=JJS>NNDA@T<V><DZA>SU@MG;LR<ZFOP=JJS>NNDA@T<V><DZA>SU@MG;LR<ZFOP=JJS>NNDA
   @T<HNIZ;WDN?RMJ;FR>KYAXTEPPF?;XFJJOP=RQGBIAGXOYNNZ>PLIF>SW>L>ACL<KGP;=QQTC
   EG??U<PUXLV?TRBO?J>QGWTJCFJA@BNHWLVORNNGQYPIKL<IM<>JX>@G?LJ>=;RBODH@PTKK@S
   IUE;IJOR<TUTRMTGSYRSVGOR<XK<F=IWQYE=LVZFP;AUHA?YJLL;EIT?ZJZC;*
```

7. Save the encrypted script.

To encrypt a column formula in a script:

Column formulas created by an encrypted script are visible by default. If you want a formula to be encrypted, follow these steps:

1. Paste just the formula into a new script window.
2. Encrypt the script by clicking **Edit > Encrypt Script**.
3. Provide a password only.
4. Copy the encrypted script.
5. Paste the encrypted script in a new script as a string argument for `JSL Encrypted()` as shown below.

```
dt = Open( "$SAMPLE_DATA\Big Class.jmp" );

dt << New Column( "Encrypted",
   Numeric,
   Continuous,
   Format( "Best", 12 ),
   Formula(
      JSL Encrypted(
         "//-e6.0.2
0,
136,4112eJwFwYkRhCAMAMB+roUQcITwDcoP4fovxN3RiOB+MFqvqXB8ZOhZdFC+dgqclJo0dwrlZ
   bFjgyPlZRuR2hp399ObhpxhQXP8
GPtftOMN6Rwz8pDnjmK1KOqq1bEG+HOxvyBb
,0,"
      )
   )
);
```

To decrypt a JSL script:

1. Open the encrypted script in JMP.
2. Select **Edit > Decrypt Script**.
3. Enter the decrypt password and click **OK**.

   The decrypted script opens in a new window.

To run an encrypted JSL script:

Note: You must know which data table the script runs on before running an encrypted script. If you do not know the name of the data table, you must decrypt the script before running it.

1. Open the encrypted script in JMP.
2. Select **Edit > Run Script**.
3. Enter the run password and click **OK**.

   The script runs:

   – If the script references a data table, you are prompted to open the data table, and then the script runs.

   – If the script requires an empty data table, you must create the table and then run the encrypted script.

Note that entering the run password runs the script, but does not show the script: you must supply the decrypt password to actually view the script.

Encryption and Global Variables

Encryption alone does not hide global variables and their values. A `Show Globals()` command displays them normally. If you want to hide global variables in an encrypted script, you can give them special names.

Any global variable whose name begins with two underscore characters (__) is hidden, and `Show Globals()` displays neither its name nor its value. For example:

```
myvar = 2;
__myvar = 5;
Show Symbols();
   // Globals
   myvar = 2;
   // 2 Global (1 Hidden)
```

This strategy works whether your script is encrypted or not.

Encrypting Scripts in Data Tables

You can also encrypt a script that is saved to a data table using the `JSL Encrypted()` or `Include( Char to Blob() )` functions.

- `JSL Encrypted()` is more straightforward, because it involves one function. You can include comments inside the encrypted script.
- `Include( Char to Blob() )` lets you include comments, but not inside the script.

Note: Column formulas in encrypted scripts are not encrypted. To encrypt them, include them in the `JSL Encrypted()` function.

Follow these steps to encrypt a data table script:

1. Place the script in a script window.

   You cannot directly encrypt a script that is already saved to a data table.
2. In the script window, select **Edit > Encrypt**.
3. Enter a decrypt password.
4. (Optional) Enter a run password to require the user to enter a password before running the script.
5. If you entered only a decrypt password, click **Yes** to confirm that you do not want to assign a run password.

   The encrypted script opens in a new script window.
6. Copy the entire encrypted script.
7. Create a new data table script or open an existing script.

8. In the script portion of the window, type one of the following functions:

   ```
   JSL Encrypted( "" );
   ```

   ```
   Include( Char to Blob( "" ) );
   ```

9. Paste the encrypted script inside the quotation marks in the function.
10. Click **OK**.

Figure 8.5 Example of an Encrypted Data Table Script

```
JSL Encrypted(
    "//-e6.0.2
WE@GSACGT<?CKEG=NG;B<=IRLXCU=BV;@NS<TW;LR<ZFOP=JJS>NNDA@T<V><DZA
>SU@MG;LR<ZFOP=JJS>NNDA@T<V><DZA>SU@MG;LR<ZFOP=JJS>NNDA@T<V><DZA
>SU@MG;LR<ZFOP=JJS>NNDA@T<FFHG=R;GYNBNME>ZBMIB?O;G<>?;UTG@=HVYCC
@MFYCMVQEOTNWASNOEV=Y<TCELCQHW>AFY>;PGSSF<IBIU?H<YLQABWEGSQBMLHW
BJE?HUC=BTP?;?;LVTT>ZQI=IXTJ<P=IHYJIPAC?PGZH;OCCMXPOJHLIRHERSW;V
;@GZDHDNF><YCN@WLBTOFZRAILT?L;NT<OOCLLLWYAIVCSPBMVL???KK=TBZJ@KU
BD>=@;EOUFMUUFLSP<HMZWL?VLHXKK?S@WAFXC>EZODQOHREMNYKRBKAWS=PSXSQ
CRYFOPVJRBNHIVLE?BNPVNM=LO>BAT<?CQF?JNNQAUEHMM;BTU><=WFVOO<;WVKB
;RSBEXTS;QGBEKNDCANPFP;DYDHTSMQNAGVESB;TCWJKGWNBG>JBESVNKOVOBPBE
;?ZPYS;X?=VE=>YFHD;;;*"
)
```

Working with Expressions in an Encrypted Script

An encrypted script might not run the same as the unencrypted script if the script attempts to convert parts of itself from expressions back to character strings. This is necessary to keep the script encrypted. You can avoid this problem by using the expression manipulation functions such as `Arg()`, `Head()`, `Substitute()`, and so on. You can also use `Parse()` to make expressions from character strings that are part of the encrypted script.

For example, here is a script that modifies an expression the wrong way for encryption:

```
ex = Expr(
   aaa = 20;
   Show( aaa );
);
// Wrong, don't do this.
temp = (Parse( Substitute( Char( Name Expr( ex ) ), "aaa", "bbb" ) ));
temp;
```

The script fails when encrypted because `Char()` is unable to break the encryption of the expression stored in `ex`. A second example follows:

```
ex = Expr(
   aaa = 20;
   Show( aaa );
);
// Best choice if it works for you.
temp = Substitute( Name Expr( ex ), Expr( aaa ), Expr( bbb ) );
temp;
```

A third example, though not optimal, follows:

```
ch = "aaa = 20; Show(aaa);";
/* Leaves the ch variable holding unencrypted text
and Temp holds an unencrypted expression.*/
temp = Parse( Substitute( ch, "aaa", "bbb" ) );
temp;
```

You might prefer the third choice over the second choice because it looks easier to understand; be aware that you might want to clear variables like `ch` and `Temp` when you are done with them. You probably encrypted the script to keep other folks from reading it, and they will definitely be able to see the content of variables when you are done.

Additional Numeric Functions

JSL also offers several categories of operations that do not make much sense in the context of the formula editor: matrix operations and numeric derivatives of functions. Algebraic derivatives are also available.

The basic arithmetic operators can also be used with matrix arguments for matrix-wise addition, subtraction, and so on. Matrices also have a few special operators for elementwise multiplication and division, concatenation, and indexing. See "Matrices".

Derivatives

JSL has three internal functions (not all available in the calculator) for taking derivatives.

`Derivative()` takes the first derivative of an expression with respect to names you specify in the second argument. A single name might be entered as this second argument; or multiple values can be specified in a list, in other words, surrounded by braces.

Note: `Derivative()` is also available as an editing command inside the formula editor (calculator), located on the drop-down list in the top center of the formula editor (above the keypad). To use it, highlight a single variable in the expression (to designate which variable the derivative should be taken with respect to), then select the Derivative command from the menu. The whole formula is replaced by its derivative with respect to the highlighted name.

In scripts, the easiest way to use the function is with a single name. In this example, the mathematical notation is shown first followed by the JSL equivalent.

For $f(x) = x^3$, the first derivative is $f'(x)$ or $\frac{d}{dx}x^3 = 3x^2$.

```
result = Derivative( x ^ 3, x );
```

```
Show( result );
  result = 3 * x ^ 2
```

If you want an efficient expression to take the derivative with respect to several variables, then the variables are specified in a list. The result is a list containing a *threaded* version of the original expression, followed by expressions for the derivatives. The expression is *threaded* by inserting assignments to temporary variables of expressions that are needed in several places for the derivatives.

Here is an example involving an expression involving three variables. Listing all three variables returns the first derivatives with respect to each. The result is a list with the original expression and then the derivatives in the order requested. However, note here that JMP creates a temporary variable `T#1` for storing the subexpression `x^2`, and then uses that subexpression subsequently to save calculations.

```
result2 = Derivative( 3 * y * x ^ 2 + z ^ 3, {x, y, z} );
Show( result2 );
  result2 = {3 * y * (T#1 = x ^ 2) + z ^ 3, 6 * x * y, 3 * T#1, 3 * z ^ 2}
```

To take second derivatives, specify the variable as a third argument. Both the second and third arguments must be lists. JMP returns a list with the original expression, the first derivative(s), and then the second derivative(s) in the order requested.

```
second = Derivative( 3 * y * x ^ 2, {x}, {x} );
Show( second );
  second = {3 * y * x ^ 2, 6 * x * y, 6 * y}

second = Derivative( 3 * y * x ^ 2, {y}, {y} );
Show( second );
  second = {3 * y * (T#1 = x ^ 2), 3 * T#1, 0}

second = Derivative( 3 * y * x ^ 2, {y}, {x} );
Show( second );
  second = {3 * y * (T#2 = x ^ 2), 3 * T#2, 6 * x}
```

`Num Deriv()` takes the first numeric derivative of a function with respect to the value of the first argument by calculating the function at that value and at that value plus a small delta (Δ) and dividing the difference by the delta. `Num Deriv2()` computes the second numeric derivative in a similar fashion. These are used internally for nonlinear modeling but are not frequently useful in JSL. Note that these functions do not differentiate using a variable, but only with respect to arguments to a function. In order to differentiate with respect to x, you have to make x one of the immediate arguments, not a symbol buried deep into the expression.

Suppose to differentiate $y = 3x^2$ at the value of $x = 3$. The *incorrect* way would be to submit the following script:

```
x = 3;
n = Num Deriv( 3 * x ^ 2 );
```

The *correct* way is to make x an argument in the function.

```
x = 3;
f = Function( {x}, 3 * x ^ 2 );
n = Num Deriv( f( x ), 1 );
   18.000029999854
```

Consider both the mathematical notation and the JSL equivalent for another example:

For $f(x) = x^2$, it calculates $\frac{d}{dx}x^2 = \frac{(x+\Delta)^2 - x^2}{\Delta}$. At $x_0 = 3$, $\frac{d}{dx}x^2 = 6.00001$.

```
x = 3;
y = Num Deriv( x ^ 2 ); // or equivalently: y = Num Deriv( 3 ^ 2 );
   6.0000099999513
```

And here are a few more examples:

```
x = Num Deriv( Sqrt( 7 ) ); // returns 0.188982168980445
y = Num Deriv( Normal Distribution( 1 ) ); // returns 0.241969514669371
z = Num Deriv2( Normal Distribution( 1 ) ); // returns -0.241969777547979
```

Table 8.7 Derivative functions

Function	Syntax	Explanation
Derivative	Derivative(*expr*, {*name*, ...})	Returns the derivative of the *expr* with respect to *name*. Note that the second argument can be specified in a list with braces { } or simply as a variable if there is only one. Give two lists of names to take second derivatives.
NumDeriv	NumDeriv(*expr*)	Returns the first numeric derivative of the *expr* with respect to the first argument in the expression.
NumDeriv2	NumDeriv2(*expr*)	Returns the second numeric derivative of the *expr* with respect to the first argument in the expression.

Algebraic Manipulations

JSL provides a way of algebraically unwinding an expression (essentially, solving for a variable). It is accomplished through the `Invert Expr()` function.

`Invert Expr(`*expression*`, `*name*`, `*y*`)`

where

- *expression* is the expression to be inverted, or the name of a global containing the expression
- *name* is the name inside expression to unwind the expression around

- *y* is what the expression was originally equal to

For example,

```
Invert Expr( Sqrt( log( x ) ), x, y );
```

is wound around the name x (which should appear in the expression only once), and results in

```
Exp( y ^ 2 )
```

It is performed exactly as you would when doing the algebra by hand.

$y = \text{Sqrt}(\log(x));$

$y^2 = \text{Log}(x);$

$\text{Exp}(y^2) = x;$

`Invert Expr` supports most basic operations that are invertible, and makes assumptions as necessary, such as assuming you are interested only in the positive roots, and that the trigonometric functions are in invertible areas so that the inverse functions are legal.

F, Beta, Chi-square, *t*, Gamma, and Weibull distributions are supported for the first arguments in their Distribution and Quantile functions. If it encounters an expression that it cannot convert, `Invert Expr()` returns `Empty()`.

JSL provides a `Simplify Expr` command that takes a messy, complex formula and tries to simplify it using various algebraic rules. To use it, submit

```
result = Simplify Expr(expr(expression));
```

or

```
result = Simplify Expr(nameExpr(global));
```

For example,

```
Simplify Expr( Expr( 2 * 3 * a + b * (a + 3 - c) - a * b ) );
```

results in

```
6*a + 3*b + -1*b*c
```

`Simplify Expr()` also unwinds nested `If` expressions. For example:

```
r = Simplify Expr( Expr( If( cond1, result1, If( cond2, result2, If( cond3,
  result3, resultElse ) ) ) ) );
```

results in

```
If(cond1, result1, cond2, result2, cond3, result3, resultElse);
```

Maximize and Minimize

The `Maximize()` and `Minimize()` functions find the factor values that optimize an expression. The expression is assumed to be a continuous function of the factor values.

Here is the form of the call:

```
result = Maximize(objectiveExpression,{list of factor names},
  <<option(value))
result = Minimize(objectiveExpression,{list of factor names},
  <<option(value))
```

objectiveExpression is the expression whose value is to be optimized, and can either be the expression itself, or the name of a global containing a stored expression.

{*list of factor names*} is an expression yielding a list of names involved in *objectiveExpression*.

The name can be followed by limits that bound the permitted values, for example name(`lowerBound`,`upperBound`).

If you want to limit the values on one side, make the other side a missing value, for example:

```
{beta} // unconstrained
{beta (0,1)} // constrained between 0 and 1
{beta (.,1)} // upper limit of 1
{beta (0,.)} or {beta (0)} // lower limit of 0
```

Factor values can be either numbers or matrices.

Options available, shown with their default value, include:

```
<< Tolerance(.00000001) // convergence criterion
<< Max Iter( 250) // maximum number of iterations
<< Limits() //
```

Initial values are assumed to be already supplied the factor values before calling the function.

These functions are not expected to find global optima for functions that have multiple local optima; they are useful only for taking an initial value and moving it to either a local or global optimum.

The return value is currently the value of the objective function, if the optimization was successful, or `Empty()` if not.

Least Squares Example

The following example uses `Minimize` to find the least squares estimates of this exponential model, with data taken from the Nonlinear Example/US Population.jmp sample data table.

```
xx = [1790, 1800, 1810, 1820, 1830, 1840, 1850, 1860, 1870, 1880, 1890, 1900,
  1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990];
yy = [3.929, 5.308, 7.239, 9.638, 12.866, 17.069, 23.191, 31.443, 39.818,
  50.155, 62.947, 75.994, 91.972, 105.71, 122.775, 131.669, 151.325, 179.323,
  203.211, 226.5, 248.7];
b0 = 3.9;
b1 = .022;
sseExpr = Expr(
  Sum( (yy - (b0 * Exp( b1 * (xx - 1790) ))) ^ 2 )
);
sse = Minimize( sseExpr, {b0, b1}, <<Tolerance( .00001 ) );
Show( b0, b1, sse );
  b0 = 13.9991388055261;
  b1 = 0.0147104409355048;
  sse = 1862.14141218875;
```

Identifying Differences Between Strings, Lines, or Sequences

The `Shortest Edit Script()` function compares two strings, lines, or sequences and returns a list of changes or a matrix that describes the changes. You might use `Shortest Edit Script()` to identify differences between two columns, lists, or matrices. The function describes *a* (not *the*) shortest list of instructions to convert sequence A into sequence B. There might be more than one shortest script.

Example of Sequences

```
t1 = New Table( "t1",
  New Column( "Column 1",
     Numeric,
     Continuous,
     Format( "Best", 12 ),
     Set Values( [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6, 7] )
  )
);

t2 = New Table( "t2",
  New Column( "Column 1",
     Numeric,
```

```
      Continuous,
      Format( "Best", 12 ),
      Set Values( [2, 3, 4, 5, 2, 3, 4, 5, 2, 3, 4, 5, 6] )
   )
);
EditScript = Shortest Edit Script( // compares column 1 in each table
   sequences(
      N Rows( t1 ),
      N Rows( t2 ),
      Function( {a, b}, // subscripts the columns
         t1:column 1[a] == t2:column 1[b] // in data tables t1 and t2
      )
   )
);
   [-1 1 . 1, // delete 1 item at position 1 in string a
   0 2 1 4, // keep 4 items at position 2 in string a, position 1 in string b
   -1 6 . 2, // delete 2 items at position 6 in string a
   1 . 5 4, // add 4 items at position 5 in string b
   0 8 9 5, // keep 5 items at position 8 in string a, position 9 in string b
   -1 13 . 1] // delete 1 item at position 13 in string a
```

Here are the columns in the matrix:

- Column 1: delete the items (-1), keep the common items (0), or add the items (1)
- Column 2: the position in string a
- Column 3: the position in string b
- Column 4: the number of items the instruction uses

A missing value indicates that the position isn't used in the comparison.

Example of Separators

The following example considers "@" and "$" to be separators.

```
aa = "this is$a test of@shortest$edit script lines$with several words";
bb = "this is a$test of$shortest$edit script lines with several@words";

// @ and $ separators
Shortest Edit Script( lines( aa, bb, separators( "@$" ) ) );
   {{"Remove", "this is$a test of@"}, {"Insert", "this is a$test of$"},
      {"Common",
   "shortest$"}, {"Remove", "edit script lines$with several words"}, {"Insert",
   "edit script lines with several@words"}}
```

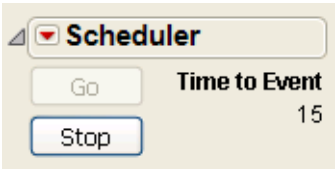
See the *JSL Syntax Reference*.

Scheduling Actions

A `Schedule()` function lets you set up a script to be executed some number of seconds later.

```
Schedule( 15, Print( "hello" ) );
```

Figure 8.6 JMP Scheduler



A Scheduler window shows the time until the next event and has buttons for restarting (Go) or stopping (Stop) the schedule. Its pop-up menu has a command Show Schedule, which echoes the current schedule to the log. For example, if you checked the schedule several times during the "hello" example, you would see something like this:

```
Scheduled at 11.55000000000018 :Print("hello")
Scheduled at 4.716666666666697 :Print("hello")
Scheduled at 3.083333333333485 :Print("hello")
```

The script might also be a name referring to a stored expression. For example, try submitting this script, which calls itself:

```
quickieScript = Expr(
   Show( "Hi there" );
   Schedule( 15, quickieScript );
);
quickieScript;
```

This script should show the string "Hi there" in a log window after 15 seconds, and reschedule itself for another 15 seconds, continuing until the Stop button is clicked.

More typically, in a production setting you might want to set up a schedule like this:

```
FifteenMinuteCheck = Expr(
   Show( "Checking data" );
   Open( "my file", options... );
   distribution( Column( column1 ), capability( spec limits ) );
   Schedule( 15 * 60, FifteenMinuteCheck );
);
FifteenMinuteCheck;
```

`Schedule()` initiates an event queue, but once it has the event queued, JMP proceeds with the next statement in the script. For example, the following has results that might surprise you:

```
Schedule( 3, Print( "one" ) );
```

```
Print( "two" );
   "two"
   "one"
```

If you want the script to wait until the scheduled events are finished before proceeding, one solution would be to use `Wait( )` with a suitable pause. Another is to embed the subsequent actions into the schedule queue. `Schedule()` accepts a series of arguments to queue many events in sequence. Each event is a separate call to the schedule. Each event time is an absolute time relative to "now" (or the instant that **Go** is clicked). Therefore, the following sequence finishes in five seconds, not in twelve:

```
Schedule( 3, Print( "hello" ) );
Schedule( 4, Print( ", world" ) );
Schedule( 5, Print( "--bye" ) );
```

To cancel all events in a schedule queue, use `Clear Schedule`.

```
scheduler[1] << Clear Schedule( );
```

Note: It is not possible to create multiple threads using `Schedule`.

Table 8.8 Schedule commands

Message	Syntax	Explanation
`Schedule`	*sc*=Schedule(*n*, *script*)	Queues an event to run the *script* after *n* seconds.
`Clear Schedule`	*sc*<<Clear Schedule()	Cancels all events in a schedule queue.

Functions that Communicate with Users

`Show()`, `Print()`, and `Write()` put messages in the log window. `Speak()`, `Caption()`, `Beep()`, and `StatusMsg()` provide ways to say something to a viewer. `Mail()` can send an e-mail alert to a process operator.

The JMP scripting language has methods for constructing dialog boxes to ask for data column choices and other types of information. See "Modal Windows".

Tip: To preserve locale-specific numeric formatting in `Show()`, `Print()`, or `Write()` output, include `<<Use Locale(1)`.

Writing to the Log

Show

`Show()` displays the items that you specify in the log. Notice that when you show variables, the resulting message is the variable name, an equal sign, and its current value.

```
X = 1;
A = "Hello, World";
Show( X, A, "foo" );
   x = 1
   a = "Hello, World"
   "foo"
```

Print

`Print()` sends the message that you specify to the log. `Print()` is the same as `Show()` except that it prints only the value of each variable without the variable name and an equal sign.

```
X = 1;
A = "Hello, World";
Print( X, A, "foo" );
   1
   "Hello, World"
   "foo"
```

Write

`Write()` sends the message that you specify to the log. `Write()` is the same as `Print()` except that it suppresses the quotation marks around the text string, and it does not start on a new line unless you include a return character yourself with the `\!N` escape sequence.

```
myText = "Here is a message.";
Write( "Here is a message." );
   Here is a message.

Write( myText || " Do not forget to buy milk." ); // use || to concatenate
Write( "\!NAnd bread." ); // use \!N for return
   Here is a message. Do not forget to buy milk.
   And bread.
```

The sequence `\!N` inserts the line breaking characters that are appropriate for the host environment. For an explanation of the three line breaking escape sequences, see "Double Quotes".

Send Information to the User

Beep

`Beep()` causes the user's computer to make an alert sound.

Speak

`Speak()` reads text aloud. On macOS, `Speak()` has one Boolean option, `Wait()`, to specify whether JMP should wait for speaking to finish before proceeding with the next step. The default is not to wait, and you need to issue `Wait(1)` each time. For example, here is a script certain to drive anybody crazy. With `Wait(1)`, you probably want to interrupt execution before too long. If you change it to `Wait(0)`, the iterations proceed faster than the speaking possibly can and the result sounds strange. On Windows, you can use a `Wait(n)` command to accomplish the same effect.

```
For( i = 99, i > 0, i--,
   Speak(
      Wait( 1 ),
      Char( i ) || " bottles of beer on the wall, " || Char( i ) || " bottles
   of beer; " ||
      "If one of those bottles should happen to fall, " || Char( i - 1 ) || "
   bottles of beer on the wall. "
   ) );
```

A more practical example has JMP announce the time every sixty seconds:

```
script = Expr(
   tod = Mod( Today(), In Days( 1 ) );
   hr = Floor( tod / In Hours( 1 ) );
   min = Floor( Mod( tod, In Hours( 1 ) ) / 60 );
   timeText = "time, " || Char( hr ) || " " || Char( min );
   text = Long Date( Today() ) || ", " || timeText;
   Speak( text );
   Show( text );
   Schedule( 60, script );    // seconds before next script
);
script;
```

You might use a similar technique to have JMP alert an operator that a process has gone out of control.

Caption

`Caption()` brings up a small window with a message to the viewer. Captions are a way to annotate demonstrations without adding superfluous objects to results windows. The first argument is an optional `{h,v}` screen location given in pixels from the upper left; the second argument is the text for the window. If the location argument is omitted, windows appear in the upper left corner.

You can include pauses in the playback by including the named argument `Delayed` and a time in seconds. Such a setting causes that caption *and all subsequent* caption windows to be delayed by that number of seconds, until a different `Delayed` setting is issued in a `Caption` statement. Use `Delayed(0)` to stop delaying altogether.

Specify the font type, font size, text color, or background color with the following arguments:

```
Font( font );
Font Size( size );
Text Color("color");
Back Color("color");
```

The `Spoken` option causes captions to be read aloud by the operating system's speech system (if available). `Spoken` takes a Boolean argument, and the current setting (on or off) remains in effect until switched by another `Caption` statement that includes a `Spoken` setting.

This script turns speaking on and leaves it on until the last caption. In the first caption, the font type, color, and background color is specified. Run the script and notice that the font and color settings apply only to the first caption.

```
Caption(
   {10, 30},
   "A Tour of the JMP Analyses",
   Font( "Arial Black" ),
   Font Size( 16 ),
   Text Color( "blue" ),
   Back Color( "yellow" ),
   Spoken( 1 ),
   Delayed( 5 )
);
Caption( "Open a data table." );
bigClass = Open( "$SAMPLE_DATA/Big Class.jmp" );
Caption( "A data table consists of rows and columns of data." );
Caption( "The rows are numbered and the columns are named." );
Caption( {250, 50}, "The data itself is in the grid on the right" );
Caption(
   {5, 30},
   Spoken( 0 ),
   "A panel along the left side shows columns and other attributes."
);
```

Each new `Caption` hides the previous one. In other words, there is only one caption window available at a time. To close a caption without displaying a new one, use the named argument `Remove`.

```
Caption( remove );
```

StatusMsg

This command sends a message to the status bar.

```
StatusMsg( "string" );
```

Mail

`Mail()` sends an e-mail message to a user. For example, a process control manager might include a test alert script in a control chart to trigger an e-mail warning to her pager:

```
Mail(
   "JaneDoe@company.com",
   "out of control",
   "Process 12A out of control at " || Format( Today(), "d/m/y h:m:s" )
);
```

`Mail()` can also send an attachment with the e-mail. An optional fourth argument specifies the attachment. The attachment is transferred in binary format after its existence on the disk is verified. For example, to attach the Big Class.jmp data table, submit

```
Mail(
   "JohnDoe@company.com",
   "Interesting Data Set",
   "Have a look at this class data.",
   "$SAMPLE_DATA\Big Class.jmp" );
```

Note: On macOS, `Mail()` works on Mountain Lion and later operating systems. On Mountain Lion, you must enter the e-mail address and subject in the e-mail due to operating system limitations. Click the **Send message** button to send the e-mail.

Chapter 9

Data Tables

Working with Data Table Objects

Before you can work with a data table or with objects in a data table, you must first open the data table and assign a *reference* to it. In the *Scripting Guide*, *dt* represents a reference to a data table object.

To manipulate an object in JSL, you send a message to the reference that represents the object, asking the object to perform one of its tasks. *Messages* are commands that can be understood only in context by a particular type of object. For example, messages for data table objects include Save, New Column, Sort, and so on. Most of these messages would not make sense for another object, such as a platform object.

In this chapter, you can learn the following:

- How to assign a reference to a data table
- How to send messages to the reference, resulting in specific actions
- About the different types of messages that data table objects can understand

Most of this chapter focuses on the different types of messages that you can send to data table objects, such as data tables, columns, rows, and values.

Tip: This chapter contains most, but not all of the JSL commands that you can use with data tables. For an exhaustive list, see the JMP Scripting Index. Select **Help > Scripting Index**. Select **Objects** from the menu, and then select **Data Table**.

Contents

Get Started with Data Tables . . . 327
Basic Data Table Scripting . . . 330
Open a Data Table . . . 330
Create a New Data Table . . . 332
Import Data . . . 333
Set the Current Data Table . . . 357
Name a Data Table . . . 358
Save a Data Table . . . 358
Hide a Data Table . . . 359
Print a Data Table . . . 360
Resize a Data Table . . . 360
Close a Data Table . . . 360
Set and Get a Data Table . . . 361
Perform Actions on All Open Data Tables . . . 361
Create Journals . . . 362
Create Excel Workbooks . . . 363
Prevent Changes to a Data Table . . . 364
Advanced Data Table Scripting . . . 365
Select Columns in a Data Table . . . 365
Store Summary Statistics in Global Variables . . . 366
Create a Table of Summary Statistics . . . 369
Subset a Data Table . . . 371
Add a Data Filter . . . 372
Sort a Data Table . . . 379
Stack Values in a Data Table . . . 379
Split Values in a Stacked Data Table . . . 380
Transpose a Data Table . . . 382
Vertically Concatenate Data Tables . . . 383
Horizontally Concatenate Data Tables . . . 383
Virtually Join Data Tables . . . 384
Replace Data in Data Tables . . . 387
Create a Table Using Tabulate . . . 389
Copy Data Tables . . . 390
Find Missing Data Patterns . . . 390
Compare Data Tables . . . 391
Create a Summary Table . . . 391
Subscribe to a Data Table . . . 392
Move Data Between Matrices and Data Tables . . . 394
Add Hyperlinks to Text Columns . . . 394

Columns 396
Send Messages to Data Column Objects 397
Create Columns 398
Create a New Column by Text Matching 400
Add Several Columns at Once 401
Group Columns 402
Use Set Selected to Select Columns 404
Rearrange and Move Columns 405
Recode Columns 406
Add a Column Switcher 407
Compress Selected Columns 408
Create Custom Functions, Transforms, and Formats 408
Custom Transform Example 411
Delete Columns 415
Get Column Names 415
Column Attributes 416
Column Properties 423
Specify Engineering SI Units 430
Rows 431
Add Rows 431
Delete Rows 432
Select Rows 433
Find Rows 436
Move Rows 437
Assign Colors and Markers to Rows 437
Color Cells 438
Hide, Exclude, and Label Rows 439
Iterate on Rows in a Table 440
Prevent Editing Rows 443
Row States 443
Accessing Data Values 460
Set or Get Values by Column Name 461
Additional Ways to Access Data Values 462
Add Metadata to a Data Table 463
Data Table Variables 463
Table Scripts 463
Formulas 468
Delete Metadata 469
Calculations 469
Pre-Evaluated Statistics 469

Calculator Formulas 471

Get Started with Data Tables

Tip: Keep the log window open to see the output of each script that you run. Select **View > Log** to open the Log window. See "Work with the Log".

The typical way to work with values in a data table is to follow these steps:

1. Set up the data table whose values you want to access as the current data table. Or, if you already have a data table reference, you can simply use that reference.
2. Specify the row or rows whose values you want to access and specify the column name that contains the values that you want to access.

The following example opens the Big Class.jmp sample data table (making it the current data table), and then specifies row 2 in the weight column. A value of 123 is returned in the log, which is the weight for Louise in row 2.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt:weight[2];
   123
```

If the data table that you want to work with is already open, proceed using one of the following examples:

```
dt = Data Table( "My Table" ); // the open table named My Table
dt = Current Data Table(); // the table in the active window
dt = Data Table( 3 ); // the third open table
```

Once you have an open data table with a reference, you can send it messages using either the << operator or the Send function. The following example illustrates both methods:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Save();
Send( dt, Save() );
```

Note the following about messages:

- You can send messages to any data table object (a data table, column, row, and so on).
- Messages sent to data table objects always have the same pattern:

  ```
  Reference << Message( Arguments );
  ```

- Usually, creating a reference and sending many messages to the reference is the easiest approach. However, you can also use the direct route instead of using a reference if you have only one message to send. The following example saves the current data table:

  ```
  Current Data Table() << Save();
  ```

- You can stack up a series of messages in one statement. Commands are evaluated from left to right, and each returns a reference to the affected object. The following example creates a new data table called My Table, adds two columns to it, and prompts you to save it:

  ```
  dt = New Table( "My Table" );
  dt << New Column( "Column 1" ) << New Column( "Column 2" ) << Save( "" );
  ```

  In this example, each message returns a reference to the data table (`dt`).
- If you specify too few arguments for a message, JMP presents a window to get the necessary information from you. JMP often presents windows when your script is incomplete, a behavior that you can use to your advantage when writing scripts that need to query users for their choices.
- Some messages come in pairs; one to "set" or assign each attribute, and one to "get" or query the current setting of each attribute.

Why are Some Commands Sent to Objects and Others Used Directly?

`New Table` and `Open` are commands to create objects that do not exist yet. Once created, you send them messages requesting changes. To close such objects, you must close the objects' *container*, because the objects cannot delete themselves.

The following example creates a table and assigns the data table reference to the *dt* variable. New Column messages are sent to the data table reference. To delete one of those columns, the Delete Columns message is sent to the data table reference, not to the column itself.

```
dt = New Table( "Airline Data" );
dt << New Column( "Date" );
dt << New Column( "Airline" );
dt << Delete Columns( "Date" );
```

How Can I See All of the Messages that Can be Sent to a Data Table Object?

To see all of the messages that can be sent to a data table object, see the Scripting Index:

1. Select **Help > Scripting Index**.
2. Select **All Categories** from the list.
3. Select **Data Table** in the list.

You can also use the `Show Properties()` command. This is a good approach if you want to print or copy the list of messages. The `Show Properties()` command lists the messages that you can send to a data table in the Log window. `Show Properties()` is a command that takes any scriptable object (such as a data table or column) as its argument. To show properties for a data table:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Show Properties( dt );
```

The resulting message list is hierarchical.

`[Subtable]` refers to a set of JMP menu commands. For example, the Tables subtable represents the JMP Tables menu, and the indented messages in this subtable correspond to commands in the Tables menu. Many messages include a short description.

Messages labeled as `[Action]` all result in some action being taken. Some messages are available for `[Scripting Only]`. Some messages take a Boolean argument, labeled as `[Boolean]`.

The platforms in the Analyze and Graph menus appear in the properties list because you can send a platform name as a message to a data table. This launches the platform for the data table through the usual launch window. For more information about writing platform-specific scripts, see "Scripting Platforms".

```
dt << Distribution( Y( height ) );
```

Note: In addition to data tables, `Show Properties` also works with platforms and display boxes.

The JMP Scripting Index provides more information about these properties. You can also run and modify sample scripts from the Scripting Index. Select **Help > Scripting Index** and search for the property in the Objects list.

Basic Data Table Scripting

Before you can work with data table objects, you must open or create a data table, and assign a reference to the data table.

- "Open a Data Table"
- "Create a New Data Table"
- "Import Data"
- "Set the Current Data Table"
- "Name a Data Table"
- "Save a Data Table"
- "Hide a Data Table"
- "Print a Data Table"
- "Resize a Data Table"
- "Close a Data Table"
- "Set and Get a Data Table"
- "Perform Actions on All Open Data Tables"
- "Create Journals"
- "Create Excel Workbooks"
- "Prevent Changes to a Data Table"

Open a Data Table

Use the `Open()` function to open a data table.

- To simply open a data table without returning a reference to it:

```
Open( "$SAMPLE_DATA/Big Class.jmp" ); // open the data table
```

- To open a data table and retain a reference to it:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" ); // open and store a reference
```

The path to the data table can be a quoted literal path (absolute or relative) or an unquoted expression that yields a pathname. Relative paths are interpreted relative to the location of the .jsl file (for a saved script). For unsaved scripts, the path is relative to your primary partition (Windows) or your <*username*>/Documents folder (macOS).

```
Open( "../My Data/Repairs.jmp" ); // relative path on Windows and macOS
Open( "::My Data:Repairs.jmp" ); // relative path on macOS
Open( "C:/My Data/Repairs.jmp" ); // absolute path
```

JMP provides shortcuts (*path variables*) to directories or files. Instead of entering the entire path to the directory or file, you include a path variable in the `Open()` expression. For example, JMP sample scripts typically use the $SAMPLE_DATA path variable to open files in the Samples/Data folder. For more information about path variables, see "Path Variables".

If you do not want to specify the entire path every time you open a data table, define a filepath string and concatenate the path with the filename:

```
myPath = "C:/My Data/Store25/Maintenance/Expenses/";
Open( myPath || "Repairs.jmp" );
```

Upon opening a data table, JMP stores the data table in memory. This means that if a script attempts to open a data table that is already open, the opened version appears. JMP does not read the version that is saved on your computer.

Test for an Open Data Table

In a script that depends on an opened data table, you can test to see whether the table is open using `Is Empty()` or `Is Scriptable()`. In the following example, the script performs a Bivariate analysis on Big Class.jmp and then closes the data table. Before proceeding to the Oneway analysis later in the script, `Is Scriptable()` tests for the open data table. If `1` (true) is returned, the table opens, and the script continues.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
obj = dt << Bivariate( Y( :height ), X( :age ), Fit Line );
Close( dt );
If( Not( Is Scriptable( dt ) ),
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" ),
);
obj = dt << Oneway( Y( :height ), X( :age ), Means( 1 ), Mean Diamonds( 1 ) );
```

You might want to see if the data table is open and then close it to discard any changes before running another script on the data table. To test for and close Big Class.jmp, run the following script:

```
Try( Close( Data Table( "Big Class" ) ) );
```

Prompt Users to Open a Data Table

You can use an `If()` expression to prompt a user to open a data table, if no open data table is found. And if they do not select a table, the script should end. The following script shows an example:

```
dt = Current Data Table();
If( Is Empty( dt ),
   Try( dt = Open(), Throw( "No data table found" ) )
);
```

The user is prompted to open a data table. If the user clicks Cancel instead of opening a data table, an error appears in the log.

Show Only Specific Columns

To open a data table and show only a specific set of columns, identify those columns in the `Open()` expression. This is particularly helpful with a large data table in which only a few columns are necessary.

The following example opens Big Class.jmp and includes only the age, height, and weight columns.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp", Select Columns( "age", "height",
  "weight" ) );
```

You can also specify the columns to leave out of the open data table:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp", Ignore Columns( "name", "sex" ) );
```

Create a New Data Table

You can start a new data table, or start a new data table and store its reference in a global variable. In either case, specify the table name as an argument.

```
New Table( "My Table" );
```

or

```
dt = New Table( "My Table" );
```

The following sections describe the optional arguments for the `New Table()` function.

Private, Invisible, and Visible

The optional `visibility` argument specifies whether the new data table is shown. `visibility( "invisible" )` hides the data table from view; it appears only in the JMP Home Window and the Window menu. `visibility( "private" )` avoids opening the data table. `visibility( "visible" )` shows the data table. `"visible"` is the default value.

```
New Table( "My Table", visibility( "private" ) );
```

A private data table enables you to avoid exposing the table to general use. Sometimes a script creates a private data table to store private data generated by a process. It is not for saving memory.

When you do not need a view of the data, or when you do not need to interact with the data, use an invisible data table.

Notes:

- The advantages of using private data tables is based on the size of the data in the table. If the data table has only five columns and 20 rows, the savings for a private data table is substantial. However, for a table with 100 columns and 1,000,000 rows, the overhead of the actual window is small in comparison to the actual data.
- Private tables are not counted when `N Table()` returns the number of open data tables.
- Private tables cannot be made current with `Current Data Table()`.

Actions

An optional argument that can define the new table. For example, the following script creates a new data table named *Little Class*. It adds three rows and two named columns as well as entering values for each cell in the table:

```
dt = New Table( "Little Class",
   Add Rows( 3 ),
   New Column( "name",
      Character,
      "Nominal",
      Set Values( {"KATIE", "LOUISE", "JANE"} )
   ),
   New Column( "height",
      "Continuous",
      Set Values( [59, 61, 55] )
   )
);
```

Import Data

Upon importing, JMP converts file types such as text files and Microsoft Excel files to the data table format. On Windows, JMP relies on filename extensions to identify the type of file and how to interpret its contents. Here are some examples of filename extensions identified by JMP:

```
Open( "$SAMPLE_IMPORT_DATA/Bigclass.xlsx" ); // Microsoft Excel file
Open( "$SAMPLE_IMPORT_DATA/Bigclass.txt" ); // text file
Open( "$SAMPLE_IMPORT_DATA/Carpoll.xpt" ); // SAS transport file
```

On macOS, JMP relies on the macOS Type and Creator codes (if present) and secondarily on three-letter filename extensions. Be sure to add the file extension before importing the file on macOS.

- Type and Creator codes are invisible data that enable the Finder to display a file with the correct icon corresponding to the application that created it.

- Files with generic icons should have the filename extensions (as with files created on other operating system).

```
Open( "$SAMPLE_IMPORT_DATA/Bigclass.txt", text );
```

Additional supported formats include .csv, .jsl, .dat, .tsv, and .jrn.

For more detailed information about import options, see the *JSL Syntax Reference*.

Import Multiple Files

When you save a script from the Multiple File Import (MFI) window into a script window, you get a compact form of the script that includes all of the parameters, default or otherwise, as messages. You could pare down the script as follows:

```
mfi = Multiple File Import(
   <<Set Folder( "$SAMPLE_IMPORT_DATA" ),
   <<Set Name Filter( "UN*.csv" ), // import files with this name
   <<Set Name Enable( 1 ),
   <<Set Add File Name Column( 1 ) // display the file names in a column
)
<<Import Data();
```

Note the slight distinction between the following expressions:

```
mfi = Multiple File Import();
```

`mfi` holds the Multiple File Import object.

```
dtlist = Multiple File Import(…) << Import Data;
```

`dtlist` holds the list of files produced by sending the `Import Data` message to the MFI object. The MFI object is gone because it was not stored in a variable or window. You can keep the MFI object by sending `Create Window` to it or storing the object in a variable, such as `mfi`, and by sending messages such as `Import Data` to that variable. Sending `Import Data` to an MFI object always returns a list, which could affect tasks such as creating reports.

`Create Window` can be useful for debugging. The window is visible *and* the MFI object is available. Run the following lines one at a time to see the results:

```
mfi = Multiple File Import();
mfi <<Create Window;
mfi <<Set Folder( "$DESKTOP" );
```

Here's an example of using a callback function to get a list of imported files after the user clicks Import on the Multiple File Import window.

```
Delete Directory( "$TEMP/test" );
Create Directory( "$TEMP/test" );
Save Text File( "$TEMP/test/argylesocks1.txt", "a1\!n1" ); // make some
Save Text File( "$TEMP/test/argylesocks2.txt", "a2\!n1" ); // data
```

```
global:importedFiles={};

mfi = Multiple File Import( // make an MFI object
     <<Set Folder( "$TEMP/test/" ),
     <<Set Name Filter( "argylesocks?.txt;" ),
     <<Set Name Enable( 1 ),
     <<Set Add File Name Column( 1 ),
     <<Set Import Callback(
           Function( {this, files}, // the callback function
                global:importedFiles = files;
/* optional: omit Close Window to keep the Multiple File Import window open so
   that the user can modify the settings. */
                win << Close Window;
           )
     )
);
win = mfi << Create Window; // open the window
While(
   Try(
      win << Parent;
      1;
   ,
      0
   ),
   Wait( 1 )
); // wait for the dialog to close
Show( global:importedFiles ); // proceed
```

Notes:

- See *Using JMP* for more information about MFI.
- See "Import Images from a Directory" for more information about using MFI to import images from a web page into a data table.

Import a PDF File

In a PDF file, data is laid out in tables. JMP auto-detects these tables to determine where the rows and columns begin and end.

You write scripts that specify the dimensions of each table on each page, the number of column headers, whether the tables should be concatenated or individual, and more.

Note: A PDF file consists of tables that contain data. In the PDF preview of the PDF Import Wizard, the tables are outlined and selected on each page.

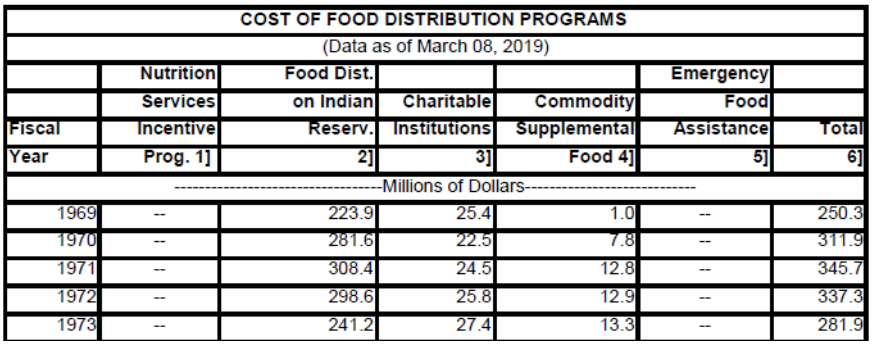
The following options are available for importing a PDF file:

```
Open( "PDF file path" ),
   Tables(
      Table( "data table name",
      Add Rows( Header Rows( number ),
      (Rect( left, right, top, bottom ),
      Page( number ), Row Borders(), Column Borders())) |
      Tables( Combine( All | Matching Headers | None),
         Minimum Rows( number ), Minimum Columns(#))))
   | PDF wizard );
```

Concatenate Tables Into One Data Table

This section shows how to import and concatenate the tables into one data table. The script that you create appears in a Source variable in the final data table, so you can run it again anytime.For this example, Figure 9.1 shows the sample import data file called Food Distribution.pdf, which contains one table per page.

Figure 9.1 Initial PDF File



COST OF FOOD DISTRIBUTION PROGRAMS

(Data as of March 08, 2019)

Fiscal Year	Nutrition Services Incentive Prog. 1]	Food Dist. on Indian Reserv. 2]	Charitable Institutions 3]	Commodity Supplemental Food 4]	Emergency Food Assistance 5]	Total 6]
----------Millions of Dollars----------						
1969	--	223.9	25.4	1.0	--	250.3
1970	--	281.6	22.5	7.8	--	311.9
1971	--	308.4	24.5	12.8	--	345.7
1972	--	298.6	25.8	12.9	--	337.3
1973	--	241.2	27.4	13.3	--	281.9

```
Open(
   "$SAMPLE_IMPORT_DATA/Food Distribution.pdf",
   PDF Tables(
      Table( // import tables into one table
         Table Name( "Food Distribution" ), // data table name
         Add Rows( // rows on page 1
            Header Rows( 7 ), // number of header rows
            Page( 1 ),
            Rect( 0.6922, 0.7155, 7.2842, 10.1124 ) // size of the table
         ), // rows and specifications on page 2
         Add Rows( Page( 2 ), Rect( 0.6922, 0.7156, 7.2842, 10.0708 ) )
      )
   )
);
```

Tip: Import a PDF file in the PDF Import Wizard and copy the script from the Source variable into a standalone script. This is a quick way to get the settings and avoid having to guess at the `Rect` settings.

Save as Individual Data Tables

The following syntax shows how to save tables in a PDF file into individual data tables. Each resulting data table contains a Source script like the one below.

```
Open(
   "PDF file path",
   PDF Tables(
      Table(
         Table Name( "data table name" ),
         Add Rows( Page( number ), Rect( left, right, top, bottom ) )
      )
   )
);
```

Concatenate Data Based on Column Headers

The following example shows how to concatenate data when column names on each page match.

Figure 9.2 Data with Matching Column Headers on each Page

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa

```
pdftable = Open(
   "$SAMPLE_IMPORT_DATA/Iris.pdf",
   PDF All Tables( Combine( Matching Headers ) )
);
```

Figure 9.3 Final Data Table

Iris Page(1) T...
Source

Columns (7/0)
Page
Table
Sepal length
Sepal width
Petal length
Petal width
Species

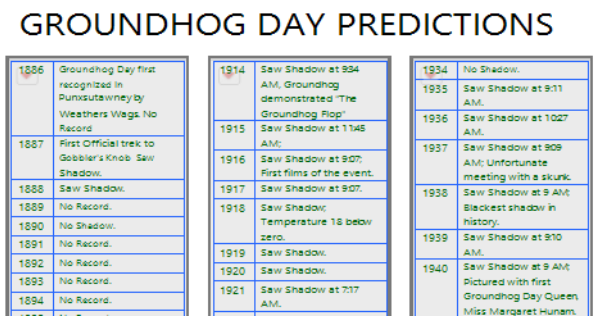
Rows
All rows 150
Selected 0
Excluded 0
Hidden 0
Labelled 0

	Page	Table	Sepal length	Sepal width	Petal length	Petal width	Species
1	1	1	5.1	3.5	1.4	0.2	setosa
2	1	1	4.9	3	1.4	0.2	setosa
3	1	1	4.7	3.2	1.3	0.2	setosa
4	1	1	4.6	3.1	1.5	0.2	setosa
5	1	1	5	3.6	1.4	0.2	setosa
6	1	1	5.4	3.9	1.7	0.4	setosa
7	1	1	4.6	3.4	1.4	0.3	setosa
8	1	1	5	3.4	1.5	0.2	setosa
9	1	1	4.4	2.9	1.4	0.2	setosa
10	1	1	4.9	3.1	1.5	0.1	setosa
11	1	1	5.4	3.7	1.5	0.2	setosa
12	1	1	4.8	3.4	1.6	0.2	setosa
13	1	1	4.8	3	1.4	0.1	setosa
14	1	1	4.3	3	1.1	0.1	setosa
15	1	1	5.8	4	1.2	0.2	setosa
16	1	1	5.7	4.4	1.5	0.4	setosa
17	1	1	5.4	3.9	1.3	0.4	setosa
18	1	1	5.1	3.5	1.4	0.3	setosa

Import PDF Pages that Contain Multiple Tables

A PDF file might contain data divided into multiple columns on each page. Figure 9.4 shows the sample import data PDF file called Groundhog Day Predictions.pdf, which contains multiple tables per page.

Figure 9.4 Multiple Columns on a Page


GROUNDHOG DAY PREDICTIONS

1886	Groundhog Day first recognized in Punxsutawney by Weathers Wags. No Record
1887	First Official trek to Gobbler's Knob. Saw Shadow.
1888	Saw Shadow.
1889	No Record.
1890	No Shadow.
1891	No Record.
1892	No Record.
1893	No Record.
1894	No Record.

1914	Saw Shadow at 9:34 AM. Groundhog demonstrated "The Groundhog Flop"
1915	Saw Shadow at 11:45 AM;
1916	Saw Shadow at 9:07; First films of the event.
1917	Saw Shadow at 9:07.
1918	Saw Shadow; Temperature 18 below zero.
1919	Saw Shadow.
1920	Saw Shadow.
1921	Saw Shadow at 7:17 AM.

1934	No Shadow.
1935	Saw Shadow at 9:11 A.M.
1936	Saw Shadow at 10:27 A.M.
1937	Saw Shadow at 9:09 AM; Unfortunate meeting with a skunk.
1938	Saw Shadow at 9 AM; Blackest shadow in history.
1939	Saw Shadow at 9:10 A.M.
1940	Saw Shadow at 9 AM; Pictured with first Groundhog Day Queen, Miss Margaret Hunam.

The following script imports all columns into a single data table.

```
Open(
  "$SAMPLE_IMPORT_DATA/Groundhog Day Predictions.pdf",
  PDF Tables(
    Table(
      Table Name( "Groundhog Day Predictions" ),
      // specifications for page 1
      Add Rows( Page( 1 ), Rect( 0.9726, 1.4325, 3.0351, 9.9974 ) ),
      Add Rows( Page( 1 ), Rect( 3.3063, 1.4325, 5.3688, 9.9608 ) ),
```

```
            Add Rows( Page( 1 ), Rect( 5.6399, 1.4325, 7.7025, 8.7925 ) ),
            // specifications for page 2
            Add Rows( Page( 2 ), Rect( 0.9726, 1.4325, 3.0351, 9.9441 ) ),
            Add Rows( Page( 2 ), Rect( 3.3063, 1.4325, 5.3688, 9.9274 ) ),
            Add Rows( Page( 2 ), Rect( 5.6399, 1.4325, 7.7025, 8.8225 ) ),
            // specifications for page 3
            Add Rows( Page( 3 ), Rect( 0.9726, 1.4325, 3.0351, 7.6404 ) ),
            Add Rows( Page( 3 ), Rect( 3.3063, 1.4325, 5.3688, 7.1254 ) ),
            Add Rows( Page( 3 ), Rect( 5.6399, 1.4325, 7.7025, 7.4721 ) )
        )
    )
);
```

Figure 9.5 Final Data Table

Groundhog Day P...
Source
Columns (4/0)
Page
Table
Column 3
Column 4
Rows
All rows 132

	Page	Table	Column 3	Column 4
1	1	1	1886	Groundhog Day first
2	1	1	1887	First Official trek to
3	1	1	1888	Saw Shadow.
4	1	1	1889	No Record.
5	1	1	1890	No Shadow.
6	1	1	1891	No Record.
7	1	1	1892	No Record.
8	1	1	1893	No Record.
9	1	1	1894	No Record.
10	1	1	1895	No Record.
11	1	1	1896	No Record.
12	1	1	1897	No Record.
13	1	1	1898	Saw Shadow.
14	1	1	1899	No Record. Groundh...
15	1	1	1900	Saw Shadow.

Export Data to a Text String

You can export data from a PDF file into a text string. You might want to do this to extract a pattern from the data.

Here is a basic example that imports all text from the Iris.pdf sample import data.

```
Open( "$SAMPLE_IMPORT_DATA/Iris.pdf", PDF text );
```

The following script finds the word “Figure” followed by a space, a digit, a period, and another digit and then prints the data to the log.

```
txt = Open( "$JMP_HOME/Documentation/JMP Documentation Library.pdf",
  PDFText() );
figures = {};
For( page = 1, page < N Items( txt ), page++,
  // search for the string
  next = Regex Match( txt[page], "Figure\s+\d+\.\d+", "" );
  While( N Items( next ) > 0,
```

```
        figures = Insert( figures, next );
        // extract the string
        next = Regex Match( txt[page], "Figure\s+\d+\.\d+", "" );
    );
);
Print( figures );
    {"Figure 2.1", "Figure 2.2", "Figure 2.3", "Figure 2.4", "Figure 2.4",
        "Figure 2.4", "Figure 2.4", "Figure 2.5", "Figure 2.6", "Figure 2.7",
        "Figure 2.8", "Figure 2.9", "Figure 2.10", "Figure 2.10", ...}
```

Import Data from a Text File

The Import Settings in the Text Data Files preferences determine how text files are imported. For example, column names begin on line one and data begin on line two by default. To use different settings, specify the import settings as `Open()` options in your script.

The default Import Settings and your custom import settings are saved in the data table Source script, so you can reimport the data using the same settings. However, the default Import Settings are optional in the script.

The following `Open()` options are available:

```
/* Charset options: "Best Guess", "utf-8", "utf-16", "us-ascii",
   "windows-1252", "x-max-roman", "x-mac-japanese", "shift-jis", "euc-jp",
   "utf-16be", "gb2312" */
CharSet("option")
Number of Columns(Number)

/* colType is Character|Numeric
colWidth is an integer specifying the width of the column */
Columns(colName=colType(colWidth),... )
Treat Empty Columns as Numeric(Boolean)
Scan Whole File(Boolean)
End Of Field(Tab|Space|Comma|Semicolon|Other|None)
EOF Other("char")
End Of Line(CRLF|CR|LF|Semicolon|Other)
EOL Other("Char")
Strip Quotes|Strip Enclosing Quotes(Boolean)
Labels | Table Contains Column Headers(Boolean)
Year Rule | Two digit year rule ("Decade Start")
Column Names Start | Column Names are on line(Number)
Data Starts | Data Starts on Line(Number)
Lines to Read(Number)
Use Apostrophe as Quotation Mark
CompressNumericColumns(Boolean)
CompressCharacterColumns(Boolean)
CompressAllowListCheck(Boolean)
```

The following script opens a text file of comma-delimited text, which includes no column names. The script defines the column names and the column widths.

```
Open(
   "$SAMPLE_IMPORT_DATA/EOF_comma.txt",
   End of Field( comma ),
   Labels( 0 ),
   Columns(
      name = Character( 12 ),
      age = Numeric( 5 ),
      sex = Character( 5 ),
      height = Numeric( 3 ),
      weight = Numeric( 3 )
   )
);
```

Here is an example of opening a text file in which the field separator is a space and the text file does not contain column headings:

```
Open(
   "$SAMPLE_IMPORT_DATA/EOF_space.txt",
   Labels( 0 ),
   End of Field( Space )
);
```

In the preceding example of a text file with space field separators, JMP can select the best file format so that you do not have to specify the labels or separator:

```
Open(
   "$SAMPLE_IMPORT_DATA/EOF_space.txt",
   Import Settings( Guess File Format() )
);
```

To set the import options interactively, include the `Text Wizard` argument. A preview of the text file opens in the text import window.

```
Open( "$SAMPLE_IMPORT_DATA/EOF_space.txt", "Text Wizard" );
```

The following sections describe each argument in more detail. For more information about import options, see the *JSL Syntax Reference*.

Number of Columns

Specifies the total number of columns in the source file. This option is important if data is not clearly delimited.

Columns

Identifies column names, column types, and column widths with a `Columns` argument as shown in the preceding examples.

If you specify settings for a column other than the first column in the file, you must also specify settings for all the columns that precede it. Suppose that you want to open a text file that has four columns (name, sex, and age, and ID, in that order). age is a numeric column, and the width should be 5. You must also set the name and sex column types and widths, and list them in the same order:

```
Columns(
   name = Character( 15 ),
   sex = Character( 5 ),
   age = Numeric( 5 )
);
```

You are not required to provide settings for any columns that follow the one that you want to set (in this example, ID).

After the data is imported, you use the modeling type for a column. See "Set or Get Data and Modeling Types".

Note: Most of the following arguments are defined in the JMP preferences. To override the preference, include the corresponding argument described below in your import scripts.

Treat Empty Columns as Numeric

Imports columns of missing data as numeric rather than character data. A period, Unicode dot, NaN, or a blank string are possible missing value indicators. This is a Boolean value. The default value is false.

Scan Whole File

Specifies how long JMP scans the file to determine data types for the columns. This is a Boolean value. The default value is true; the entire file is scanned until the data type is determined. To import large files, consider setting the value to false, which scans the file for five seconds.

Strip Quotes | Strip Enclosing Quotes

Specifies whether to include or remove the double quotation marks (") that surround string values. This is a Boolean value. The default value is `true`.

For example, suppose that the field delimiter is a space:

- *John Doe* is interpreted as two separate strings (*John* and *Doe)*.
- "*John Doe*" is interpreted as a single string. Most programs (including JMP) read a quotation mark and ignore other field delimiters until the second quotation occurs.
- If you include `Strip Quotes(1),` "*John Doe*" is interpreted as *John Doe* (one string without quotation marks).

Note that many word processors have a “smart quotation marks” feature that automatically converts double quotation marks (") into left and right curled quotation marks (“ ”). Smart quotation marks are interpreted literally as characters when the text file is imported, even when JMP strips double quotation marks.

End of Line(CR | LF | CRLF | Semicolon | Other)

Specifies the character or characters that separate rows:

- CR for carriage returns (typical for text files created on macOS up to version 9)
- LF for linefeeds (typical for UNIX and Mac OS X text files)
- CRLF for both a carriage return followed by a linefeed (typical for Windows text files).

All three characters are line delimiters by default.

Use the `Other` option to use an additional character for the row separator, which you must specify in the `EOLOther` argument. JMP interprets either this character or the default character as a row separator.

End of Field(Tab | Space | Spaces | Comma | Semicolon | Other | None)

Specifies the character or characters used to separate fields.

Notes:

- The default field delimiter is `Tab`.
- Use the `Other` option to use a different character for the field separator, which you must specify in the `EOFOther` argument.
- The `Space` option uses a single space as a delimiter.
- The `Spaces` option uses two or more spaces.

EOFOther, EOLOther

Specifies the character or characters used to separate fields or rows. For example, `EOLOther("*")` indicates that an asterisk separates rows in the text file.

Labels | Table Contains Column Headers

Indicates whether the first line of the text file contains column names. This is a Boolean value. The default value is true.

Year Rule | Two Digit Year Rule

Specifies how to import two-digit year values. If the earliest date is 1979, specify "1970". If the earliest date is 2001, specify "20xx".

Column Names Start | Column Names Are on Line

Specifies the starting line for column names. The following example specifies that the column names in the text file start on line three.

```
Open(
    "$SAMPLE_IMPORT_DATA/Animals_line3.txt",
    Columns(
        Column( "species", Character, "Nominal" ),
        Column( "subject", Numeric, "Continuous", Format( "Best", 10 ) ),
        Column( "miles", Numeric, "Continuous", Format( "Best", 10 ) ),
        Column( "season", Character, "Nominal" )
    ),
   Column Names Start( 3 )
);
```

Data Starts | Data Starts on Line

Specifies the starting line for data.

The following example specifies that the data in the text file start on line five.

```
Open(
   "$SAMPLE_IMPORT_DATA/Bigclass_L.txt",
   Columns(
      Column( "name", Character, "Nominal" ),
      Column( "age", Numeric, "Continuous", Format( "Best", 10 ) ),
      Column( "sex", Character, "Nominal" ),
      Column( "height", Numeric, "Continuous", Format( "Best", 10 ) ),
      Column( "weight", Numeric, "Continuous", Format( "Best", 10 ) )
   ),
   Data Starts( 5 )
);
```

Lines to Read

Specifies the number of lines to include in the data table. JMP starts counting after column names are read.

The following example includes only the first 10 lines in the data table.

```
Open(
    "$SAMPLE_IMPORT_DATA/Bigclass_L.txt",
    Columns(
        Column( "name", Character, "Nominal"),
        Column( "age", Numeric, "Continuous", Format( "Best", 10 ) ),
        Column( "sex", Character, "Nominal" ),
        Column( "height", Numeric, "Continuous", Format( "Best", 10 ) ),
        Column( "weight", Numeric, "Continuous", Format( "Best", 10 ) )
```

```
    ),
  Lines To Read( 10 )
);
```

Use Apostrophe as Quotation Mark

For data that are enclosed in apostrophes, this option treats apostrophes as quotation marks and omits them. For example, '2010' is imported as 2. This is a Boolean value. The default value is false.

This option is not recommended unless your data comes from a nonstandard source that places apostrophes around data fields rather than quotation marks.

Set Text Import Preferences

If you want to set preferences for importing text, it can be helpful to first see a list of all preferences. To do so, use the `Show Preferences (All)` function.

Then, copy and paste only the section you want into a Preferences function. For example, to specify import settings, write the following expression:

```
Preferences(
  Import Settings(
    End Of Field( Tab, Spaces, Comma )
  )
);
```

Import Data from a Microsoft Excel File

When you open a Microsoft Excel workbook in JMP, the file is converted to a data table. JMP supports .xls, .xlsm, and .xlsx formats. See *Using JMP* for more information about Microsoft Excel support.

Excel Preferences

In the JMP preferences, settings in the Third Party Data group can help determine how worksheets are imported:

Excel Open Method Specifies how a Microsoft Excel file should be opened by default, when using a non-specific open statement.

Use Excel Wizard Opens the Excel Import Wizard to import the file. This is the default setting.

Open All Sheets Opens all worksheets in the Microsoft Excel file.

Select Individual Worksheets Prompts users to select one or more worksheets when they open the file.

Use Excel Labels as Headings Determines whether text in the first row of the worksheet is converted to column headings in the data table.

By default, JMP takes the best guess. If names have been defined for all cells in the first row, the text in those cells is converted to column heading. Otherwise, columns are named *Column 1*, *Column 2*, and so on.

To override a preference, include the corresponding argument described later in this section.

Note: See "Create Excel Workbooks" for information about exporting data tables to an Excel workbook.

Open() Function

Using the `Open()` function without additional arguments to open an Excel file has different behaviors depending on the context:

- If the `Open()` function is a direct part of the script, the Excel files open into data tables using your Excel preferences. The following example opens both worksheets into data tables without using the wizard:

```
Path = Convert File Path( "$SAMPLE_IMPORT_DATA/Team Results.xlsx", absolute );
dt = Open( Path );
```

Note: To use the Excel Wizard, you must specify the `Excel Wizard` option in the argument as in `Open( "$SAMPLE_IMPORT_DATA/Team Results.xlsx", "Excel Wizard" );`.

- However, if the `Open()` function is part of a script that is run from clicking a button, the Preview window opens and requires the user to interact with it. Run the following example and click the button to see the Excel Import Wizard:

```
Path = Convert File Path( "$SAMPLE_IMPORT_DATA/Team Results.xlsx", absolute );
New Window( "button", Button Box( "Open", dt = Open( Path ) ) );
```

- To prevent the button script from opening the Preview window and importing the Excel file directly, provide additional arguments to the `Open()` function. Run the example and click the button. Both worksheets are opened into data tables without using the Excel Import Wizard.

```
Path = Convert File Path( "$SAMPLE_IMPORT_DATA/Team Results.xlsx", absolute );
New Window( "button", Button Box( "Open", dt = Open( Path, Use for all
   sheets(1) ) ) );
```

Alternatively, you can set the Excel Open Method preference to open all sheets:

```
Preference( Excel Open Method( "Open All Sheets" ) );
Path = Convert File Path( "$SAMPLE_IMPORT_DATA/Team Results.xlsx", absolute );
New Window( "button", Button Box( "Open", dt = Open( Path ) ) );
```

Open a Workbook in the Excel Import Wizard

The Excel Import Wizard shows a preview of the data and lets you modify the settings before importing the data. Specify `"Excel Wizard"` as the argument.

```
dt = Open( "$SAMPLE_IMPORT_DATA/Team Results.xlsx", Excel Wizard );
```

Customize Import Settings

Customize settings such as the number of rows in the column headers.

```
Open(
   "$SAMPLE_IMPORT_DATA/Team Results.xlsx",
   Worksheets( "Grouped Team Results" ),
   Use for all sheets( 1 ),
   Concatenate Worksheets( 0 ),
   Create Concatenation Column( 0 ),
   Worksheet Settings(
      1,
      Has Column Headers( 1 ),
      Number of Rows in Headers( 1 ),
      // customization based on Team Results structure
      Headers Start on Row( 3 )
   )
);
```

When you click a custom button to open a workbook, the Excel Open Method preference applies. To force the worksheet to open directly, specify the Excel Open Method in your script:

```
Preference( Excel Open Method( "Open All Sheets" ) );
```

Import Specific Worksheets

Suppose that you want to import data from specific worksheets in your workbook. Specify those worksheets using the `Worksheets` argument. In the following example, the worksheet named *small* is imported into JMP.

```
Open( "C:\My Data\cars.xlsx", Worksheets( "small" ) );
```

Or specify the number of the worksheet, the third worksheet in the following example:

```
Open( "C:\My Data\cars.xlsx", Worksheets( "3" ) );
```

Import multiple or all worksheets by including the worksheet names in a list:

```
Open( "C:\My Data\cars.xlsx", Worksheets( {"small", "medium", "large"} ) );
```

Import XML and JSON Files

The XML Import Wizard imports two types of nested text data: XML and JSON files. Both kinds of files contain structured text that can be nested to represent hierarchical relationships. This wizard helps you pick out the elements of the hierarchy that hold values and elements that determine rows to build a data table.

A useful way to imagine a nested text file: the document is a book of short stories. Some of the short stories might have chapters, others only have paragraphs. Chapters have paragraphs. Paragraphs have sentences. Here's an example of the full tree for such a book. This content is also available in the JMP Samples/Import Data directory as Book.xml.

```
<book>
  <story name="car poll">
    <wheels>4</wheels>
    <para><name>chev</name></para>
    <para><name>ford</name></para>
    <para><name>volk</name></para>
  </story>
  <story name="big class">
    <para><name>ralph</name><height>6</height></para>
    <para><name>billy</name><height>5</height></para>
  </story>
  <story name="cheese" location="NC">
    <chapter name="american">
      <para>1<price>2.00</price><quantity>2</quantity></para>
      <para>2<price>2.10</price><quantity>4</quantity></para>
      <para>3<price>2.20</price><quantity>3</quantity></para>
    </chapter>
    <chapter name="swiss">
      <para>1<price>3.00</price><quantity>3</quantity></para>
      <para>2<price>3.10</price><quantity>2</quantity></para>
    </chapter>
  </story>
</book>
```

This XML document (the book) contains three short stories; each short story is a data table, all different. This particular XML file uses paragraphs to represent rows in a table in all of the tables. That is not required, nor are any of the names special. If you wanted to import the cheese table, which is slightly more complicated than the other two, you'd probably want a variety column, containing "american" or "swiss" for each row. You would also want price and quantity columns. And you might want a column that doesn't have a clearly defined name for the 1, 2, 3, 1, 2 values of <para>. You'd expect five rows. Finally, you might want a table variable or a column for the location NC value.

The following example shows how to import the preceding data into a data table.

```
Open(
```

```
    "$SAMPLE_IMPORT_DATA/Book.xml",
    XML Settings(
       Row( "/book/story/chapter/para" ),
       // create a row for each value in <para>
       Col(
          "/book/story/@location",
          // create a column for the location attribute
          Column Name( "location" ),
          // rename the column
          Fill( "Use Forever" ),
          // fill an empty cell with the preceding value
          Type( "Character" ), // data type
          Format( "best" ) // column format
       ),
       Col(
          "/book/story/chapter/@name",
          Column Name( "variety" ),
          Fill( "Use Once" ),
          Type( "Character" ),
          Format( "best" )
       ),
       Col(
          "/book/story/chapter/para",
          Column Name( "interesting" ),
          Fill( "Use Once" ),
          Type( "Numeric" ),
          Format( "best" )
       ),
       Col(
          "/book/story/chapter/para/price",
          Column Name( "price" ),
          Fill( "Use Once" ),
          Type( "Character" ),
          Format( "best" )
       ),
       Col(
          "/book/story/chapter/para/quantity",
          Column Name( "quantity" ),
          Fill( "Use Once" ),
          Type( "Character" ),
          Format( "best" )
       )
    ),
    XML Wizard( 0 ) /* omit or change to 1 to preview the data
)                      in the XML Import Wizard */
```

Import from and Export to Google Sheets

A Google Sheet is a spreadsheet (which consists of sheets, like tabs) that is saved in your Google account. This enables you to continue to edit the spreadsheet and to share it with others.

To write a script to import data from a Google spreadsheet, log in to your Google account and get the URL for the spreadsheet. Also notice how the data is structured. For example, you must know which row the data begin on if there are column headers.

Give Google access to JMP when running the script for the first time.

To import data from a Google spreadsheet

```
dt = Google Sheet Import(
   Email( "yourgoogleaccount@gmail.com" ), // modify the email address
   // modify the sheet URL
   Spreadsheet(
   "https://docs.google.com/spreadsheets/d/1JO1M4ScGsAHb-bCUaOi575WWvQjXciQIV
   NOwVTrXDSe/edit#gid=2303744702" ),
   Sheets( "May","June" ), // specify the sheet name(s)
   Sheet Settings( // specify the structure of the data
      Has Column Headers( 1 ),
      Headers Start on Row ( 1 ),
      Data Starts on Row( 2 ),
      Cell Range( "" ),
      Import Cell Colors( 1 ),
      Suppress Empty Columns( 1 )
   )
);
```

Note: See *Using JMP* for more information about security, country restrictions, and more.

To export a JMP data table to a Google spreadsheet

Export a JMP data table to a new spreadsheet in Google Sheets by following this example:

```
email = "yourgoogleaccount@gmail.com"; // modify the Google account
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Google Sheet Export(
       dt,
       Email( email ),
      New Spreadsheet( "JSL Example" ), // specify the name of the spreadsheet
       Sheet Name( "Example 1" ) // specify the sheet name
);
```

“1” is returned in the log when the export is successful.

Export a JMP data table to an existing spreadsheet in Google Sheets by following this example. You'll need the URL of the existing spreadsheet.

```
id =
  "https://docs.google.com/spreadsheets/d/1J01M4ScGsAHb-bCUa0i575WWvQjXciQIV
  N0wVTrXDSe/edit#gid=2303744702"; // modify the URL
email = "yourgoogleaccount@gmail.com"; // modify the Google account
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Google Sheet Export(
       dt,
       Email( email ),
       Spreadsheet( id ),
       Sheet Name( "Example 1" )
);
```

"1" is returned in the log when the export is successful.

Note: See *Using JMP* for more information about security, country restrictions, and more.

Import SAS Data Sets

Open a SAS file as a data table without connecting to a SAS server.

```
sasxpt = Open( "$SAMPLE_IMPORT_DATA/carpoll.xpt" );
```

To convert the labels to column headings, include the `Use Labels for Var Names` argument.

```
sasdbf = Open( "$SAMPLE_IMPORT_DATA/Bigclass.sas7bdat", Use Labels for Var
  Names( 1 ));
```

.xpt and .stx file formats are also supported.

On Windows, you can also open SAS data sets from a SAS server. See "Connect to a SAS Metadata Server".

Import Web Pages and Remote Files

You can import data from websites or from other computers. The data might be a JMP data table, a table defined in a web page, or another file type that JMP supports.

Open or Import Data from a Website

In the `Open()` command, specify the quoted URL to open a file from a website. You can open JMP data tables or other supported file types this way.

```
Open( "http://company1.com/Repairs.jmp" );
Open( "http://company1.com/My Data.txt", text); // specify text on macOS
```

Import a Web Page

A web page can include data in tabular format. Import the table as a JMP data table:

```
Open( "http://company1.com", HTML Table( n ) );
```

n identifies which table you want to import. For example, to import the fourth table on the page, specify `HTML Table(4)`. If you omit the value, only the first table on the page is imported.

JMP attempts to preserve the table header defined in a `<th>` HTML tag. The table header is converted to column headings in the data table. If the `<th>` tag is wrong or missing, use `ColumnNames(n)` to specify the *n*th row. By default, `DataStarts(n)` will be the next row, or you can specify the `DataStarts` row.

Import Images from a Web Page

You can import images on the web into a data table, using the Expression data type. The following example creates a new data table and adds two images to it.

```
dt = New Table( "test", New Column( "Image", Expression ) );
dt << Add Rows( 1 );
dt:Image = New Image( Open(
  "https://www.jmp.com/support/help/en/16.0/jmp/images/iris_first.gif" )
  );
dt << Add Rows( 1 );
dt:Image = New Image( Open(
  "https://www.jmp.com/support/help/en/16.0/jmp/images/trimeth.png" ) );
```

Import Images from a Directory

You can use `Multiple File Import()` and an Event Handler script to create a data table of thumbnail images that link to a directory of full-size pictures. This example sets the cell height so that it fits the height of the tallest thumbnail.

Note: Typically, the Multiple File Import option is for importing multiple text files. However, this example shows how to use the option to import images.

```
path = "$SAMPLE_IMAGES/"; // images are in this directory

{dtx} = Multiple File Import(
  <<Set Folder( path ),
  <<Set Name Filter( "*.jpg; " ),
  <<Set Name Enable( 1 ), /* enable the ability to filter by name,
  in this example, files with the .jpg extension */
  <<Set Add File Name Column( 1 ), // must be included but can be hidden
  <<Set Add File Size Column( 1 ) // for Show Sizes at end
```

```
) << Import Data;

/* FileName is the column that contains the image file name.
The column is required for the image links to work, but you don't need
to see it. */
dtx:FileName << Hide;
maxheight = 0;

For Each Row( // recreate each image to minimize size of data table
   x = Eval( dtx:Picture );
   x << Scale( .1 ); // scale the image by 10%
   m = x << Get Pixels; // compress the image by recreating it
   dtx:Picture = New Image( m );
   maxheight = Max( maxheight, (dtx:Picture[] << Get Size)[2] );
   // width x height is returned, [2] gets the height
);

Eval( /* insert the value in the path variable into the Click handler. The
   path variable won't be available later, but the path value is needed to
   concatenate with the file name column's value to build a link to the image
   on the computer. */
   Eval Expr(
      dtx:picture << Set Property(
         "Event Handler",
         Event Handler( // Event Handler column property
            Click( JSL Quote( Function( {thisTable, thisColumn, iRow},
            // insert links to images
            Open( Expr( path )||Char( thisTable:FileName[ iRow ] ) ); ); ) ),

            Tip( JSL Quote( Function( {thisTable, thisColumn, iRow}, "Open " ||
   Char( thisTable:FileName[ iRow ] ) || " in your viewer."; ); ) ),

            // color of the link -- blue
            Color( JSL Quote( Function( {thisTable, thisColumn, iRow}, 5; );) )
         )          )
   )
);

// add two pixels to the maximum height
dtx << Set Cell Height( maxheight + 2 );

/* save the file to show that the data table is much smaller than the
original file */
dtx << Save( "$TEMP/Sample Images.jmp" );
Show( File Size( "$TEMP/Sample Images.jmp" ), Col Sum( dtx:FileSize ) );
```

Import a File from a Shared Computer

JMP can import files stored on a shared computer, such as another computer or a network drive. The following example shows how to open a file from Windows:

```
Open( "\\Data\Repairs.jmp" ); // UNC name
Open( "z:/Data/Repairs.jmp" ); // mapped letter drive
Open( "Repairs.jmp" ); // relative path in which the script is in the same
  directory as the file that you are importing
```

macOS doesn't have a concept like UNC naming or drive letters. Instead, you need to mount the volume through the user interface. It is possible to do it from a script, but you would need to have connected interactively at least once and stored the password in your keychain.

Here is an example that connects to a remote volume from either macOS or Windows:

```
mount = Function( {server},
  {Default Local},

  If( Host is( Mac ),
    Run Program(
      Executable( "/usr/bin/osascript" ),
      // replace ^server^ with your server name throughout the script
      Options( {"-e", Eval Insert( "mount volume \!"smb://^server^\!"" )} ),
      Read Function( "text" )
    );
    server = Concat Items( Remove( Words( server, "/" ), 1 ), "/" );
    Eval Insert( "/Volumes/^server^" );
  ,
    Eval Insert( "\!\\!\^server^" )
  )
);


users = mount( "data.company.com/Users" );
Files In Directory( Eval Insert( "^users^/Smith" ) );
```

Import JSON Data

JSON (JavaScript Object Notation) is a data interchange format that is based on JavaScript. It shares many of the same principles as languages based on C and is easy to read and write. Because of these characteristics, JSON is highly portable and can be imported into JMP.

Table 9.1 Representations of JSON in JSL

JSON	JSL
array	list (sequence of values)
object	associative array (set of named values)
number	number
string	string
true	1.0
false	0.0
null	. (missing value)

JSON consists of name-value pairs in an associative array. For example, here is JSON code that shows name-value pairs:

```
"[ { \!"key1\!": \!"a\!", \!"key2\!": \!"b\!"}, {\!"key1\!": \!"c\!",
  \!"key2\!": \!"d\!" }, {\!"key2\!": \!"f\!", \!"key1\!":\!"e\!",
  \!"key3\!": \!"g\!"}]";
```

Each list is imported as a column in the data table. “key1” is the name of the first column. The first value in the column is “a”. The second value is “c”, and the third value is “e”.

Use `JSON To Data Table()` to convert the JSON into a data table.

```
js = "[ { \!"key1\!": \!"a\!", \!"key2\!": \!"b\!"}, {\!"key1\!": \!"c\!",
  \!"key2\!": \!"d\!" }, {\!"key2\!": \!"f\!", \!"key1\!":\!"e\!",
  \!"key3\!": \!"g\!"}]";
d = JSON To Data Table( js ); // convert JSON to a data table
d << New Data View; // make a view of the data table
```

Figure 9.6 Imported JSON Data

	key1	key2	key3
1	a	b	
2	c	d	
3	e	f	g

You can also use `JSON To List()` to convert a string containing JSON into a list.

```
list = JSON To List(
```

```
    "[ { \!"name\!": \!"KATIE\!", \!"age\!": 12, \!"sex\!": \!"F\!",
    \!"height\!": 59, \!"weight\!": 95 }, { \!"name\!": \!"LOUISE\!",
    \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!": 61, \!"weight\!": 123 }, {
    \!"name\!": \!"JANE\!", \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!":
    55, \!"weight\!": 74 } ]"
);
Show( list );
   list = {{{"name", "KATIE"}, {"age", 12}, {"sex", "F"}, {"height", 59},
      {"weight", 95}}, ...}};
```

Exporting JSL to JSON

`As JSON Expr()` converts an associative array to a JSON string. You can then export the data into a JSON document.

```
list_of_associative_arrays = {["key1" => "a", "key2" => "b"], ["key1" => "c",
"key2" => "d"], ["key2" => "f", "key1" => "e", "key3" => "g"]};
jx = As JSON Expr( list_of_associative_arrays );
   "[{\!"key1\!":\!"a\!",\!"key2\!":\!"b\!"}..."[{\!"key1\!":\!"a\!",\!"key2\
      !":
   \!"b\!"}
```

Note: To import the data table as invisible, include the argument `Invisible(1)` in `JSON To Data Table()` and omit `New Data View`.

Import HDF5 Files

Hierarchical Data Format, Version 5 (HDF5) is a portable file format for storing data. An HDF5 file consists of groups and datasets. When you import the file, JMP opens a group to present the names of the inner datasets.

JMP handles only tables with numeric (integer, float, double) and string types, and compound files with three or fewer dimensions that contain only simple types.

You can import up to 1,000,000 columns and an unbounded number of rows.

Here is the syntax:

```
Open( "filename.h5", {"list_of", "dataset_names"});
```

Errors are written to the log when invalid data set names are passed.

Import ESRI Shapefiles

An ESRI shapefile is a geospatial vector data format used to create maps. JMP imports shapefiles as data tables. A .shp shapefile consists of coordinates for each shape. A .dbf shapefile includes values that refer to regions. To create maps in JMP, you modify the structure of the data and save the files with specific suffixes.

The following example imports a .shp file and saves it with the -XY suffix.

```
dt = Open( "$SAMPLE_IMPORT_DATA/Parishes.shp"),
:X << Format( "Longitude DDD", 14, 4 );
:Y << Format( "Latitude DDD", 14, 4 ) );
dt << Save( "c:/Parishes-XY.jmp" );
```

Save the .dbf file with the -Name suffix.

```
dt = Open( "$SAMPLE_IMPORT_DATA/Parishes.dbf" );
dt << Save( "c:/Parishes-Name.jmp" );
```

Restructuring the data requires several steps, including adding a Map Role column property to names in the -Name.jmp file. See *Essential Graphing*.

Import a Database

`Open Database()` opens a database using Open Database Connectivity (ODBC) and extracts data into a JMP data table. See the "Database Access".

JMP also converts DataBase Files (.dbf) files to data table format.

```
sasdbf = Open( "$SAMPLE_IMPORT_DATA/Bigclass.dbf",
  Use Labels for Var Names( 1 )
);
```

Set the Current Data Table

Tip: Be careful about assuming that the data table that you want is the current data table, even if you set it to be earlier in your script, because interim actions could change that.

A data table becomes the current data table in the following instances:

- When you open it like this:

  ```
  dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
  ```

- When you create a new table like this:

  ```
  dt2 = New Table( "Cities" );
  ```

To switch to another open data table, specify the name of the data table in `Current Data Table()`:

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt2 = New Table( "Cities" );
Current Data Table( dt1 ); // make Big Class.jmp the current data table
```

`Current Data Table()` can also take a scriptable object reference as an argument. The following expression would make the data table in use by the second Bivariate object the current data table:

```
Current Data Table( Bivariate[2] );
```

For more information about using references to analysis platform objects, see "Send Messages to a Platform".

Name a Data Table

Assign a name to a data table by sending it the `Set Name` message. The argument is a filename in quotation marks, or something that evaluates to a filename.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Set Name( "New Big Class.jmp" );

s = "New Big Class";
dt << Set Name( s );
```

To retrieve the name, send a `Get Name` message to the data table:

```
dt << Get Name();
   "New Big Class"
```

Save a Data Table

To save a data table, send a `Save` message to the data table. Here are some examples:

```
dt << Save(); // save using the current name
dt << Save( "Newest Big Class.jmp" ) // save as a new file
dt << Save( "c:/My Data/New Big Class.jmp" );  // save as a new file
dt << Save( "My Table", JMP( 5 ) ); // save as JMP 5 table
dt << Save("") // prompt to select the directory and save in the desired
   format
dt << Save( "Big Class.xlsx" ); // save as a Microsoft Excel file
Create Excel Workbook( "c:/MyData/data.xlsx", {"Abrasion", "Big Class"},
{"Abrasive", "Class"} ); // save several data table as a workbook
```

Note: If you specify the filename with no path and have not set the default directory, the file is saved on your primary partition (Windows) or in your *<username>*/Documents folder (macOS). For more information about setting the default directory, see "Relative Paths".

On Windows, saving with a `.txt` extension exports according to the Text Export preferences. On macOS, add `Text` as a second argument to the `Save` function:

```
dt << Save( "New Big Class.txt", Text );
```

If you plan to set the name of a data table and later send the `Save` message, you can just specify the name in a `Save` message.

```
dt << Set Name( "New Big Class.jmp" );
```

```
dt << Save();
```

works the same as

```
dt << Save( "New Big Class.jmp" );
```

Including Save and the pathname is also an alternative to using Save As along with the pathname.

Revert to a Saved Data Table

To return to the most recently saved data table, send a Revert message to the data table.

```
dt << Revert();
```

Hide a Data Table

There are two ways to hide a data table if the user does not need to see the table: by opening it as invisible or private.

Invisible Data Tables

An *invisible* data table is hidden from view but linked to analyses that you run on it. Open a data table as an invisible file:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp", "invisible" );
```

Here is an example of creating an invisible table, called Abc, that has ten rows. One column is named X.

```
dt = New Table( "Abc", "invisible", New Column( "X" ), Add Rows( 10 ) );
```

To find out if an open data table is invisible, pass the Boolean Has Data View message to the data table object. The following expression returns 0 (false) if the data table is invisible:

```
dt << Has Data View();
```

After you are finished with an invisible data table, do not forget to close it with the Close() function. Otherwise, the data table remains in memory until you quit JMP.

Showing an Invisible Data Table

The Show Window() function shows an invisible data table.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp", "invisible" );
dt << Show Window( 1 );
```

The user has the option of opening the table from the JMP interface. On Windows, an invisible data table appears in the Window List on the Home Window and the **Window > Unhide** menu. On macOS, the data table appears in the JMP Home window and the **Window > Hidden** menu.

Private Data Tables

Completely hide the data table from view by including the `private` argument:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp", "private" );
```

A private data table enables you to avoid exposing the table to general use. Sometimes a script creates a private data table to store private data generated by a process. It is not for saving memory.

See "Private, Invisible, and Visible".

Print a Data Table

Print a data table by sending it a `Print Window` message. JMP uses your computer's default printer settings.

```
dt << Print Window();
```

This message also applies to other display boxes.

Resize a Data Table

dt `<< Optimize Display` forces the data table to remeasure all of its columns and zoom to the best-sized window.

```
dt << Optimize Display;
```

Close a Data Table

To close a data table, use a `Close` command with the data table's reference as the argument. Data tables with unsaved changes are saved automatically, including any linked tables. Reports and graphs generated from the data tables are also closed.

To save and close the data table:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Close( dt );
```

To close the data table without saving changes:

```
Close( dt, No Save );
```

To save a copy with a new name and then close the original version:

```
Close( dt, Save("c:/My File.jmp" ) );
```

You can also close all data tables at once and either save or discard changes:

```
Close All( Data Tables );
```

```
Close All( Data Tables, No Save );
```

To close all private or invisible data tables, specify the appropriate argument:

```
Close All( Data Tables, private );
Close All( Data Tables, invisible );
```

Tip: `Close All()` also works with journals. Specify "Journals" in the argument.

Set and Get a Data Table

To specify the data table in a display box, for example, in `New Data Box()`, use `Set Data Table()` and `Get Data Table()` to set and return a data table.

The following example uses the Big Class.jmp and Cities.jmp sample data tables. The sample data tables are opened invisibly. In a new window, a new data box is created containing Big Class.jmp. After waiting 1 second, the data box contents change to display cities.jmp.

```
dtC = Open( "$SAMPLE_DATA/Cities.jmp", "invisible" );
dtA = Open( "$SAMPLE_DATA/Big Class.jmp", "invisible" );
nw = New Window( "My Table", H List Box( dtBox = dtA << New Data Box() ) );
Wait( 1.0 );
dtBox << Set Data Table( Data Table( "Cities.jmp" ) );
```

Perform Actions on All Open Data Tables

If you want to perform an action on all of the data tables that are currently open (except private tables), use `N Table()` to get a list of references to each one:

```
dt1 = Open( "$SAMPLE_DATA/Cities.jmp" );
dt2 = Open( "$SAMPLE_DATA/Big Class.jmp" );
openDTs = List();
For( i = 1, i <= N Table(), i++,
  Insert Into( openDTs, Data Table( i ) );
);
```

`openDTs` now is a list of references to all open data tables. You can send messages to any one by using `openDTs(n)`. You can use a for loop to send messages to all of the open data tables. This loop adds a new column named My Column to each open data table.

```
For( i = 1, i <= N Items( openDTs ), i++,
  openDTs[i] << New Column( "My Column" );
);
```

If you just want a list of table names and not references, use the `Get Name` message:

```
For( i = 1, i <= N Table(), i++,
  Insert Into( openDTs, Data Table( i ) << Get Name());
```

```
);
```

You can then use the list of table names to put the names of all of the open data tables in a list box (window) so that the user can choose and perform tables operations. Or, you can write the names out to a file.

Create Journals

Journals consist of JMP graphs and reports, graphics, text, and links to items such as web pages and files. They make it easy to reuse content in presentations and import into other documents.

Here is the basic syntax for journals:

```
dt << Journal;
```

You can also create a new journal window and immediately add to it within the `New Window()` command. The following examples create an empty untitled journal:

```
New Window( <<  Journal);
```

The following examples create an empty journal named “Sales”:

```
New Window( "Sales", << Journal );
```

Here is an example of creating a journal named “Test Buttons” that contains two buttons.

```
New Window( "Test Buttons", << Journal,
   Button Box( "Test One", New Window( "Hi there1", << Modal ) ),
   Button Box( "Test Two", New Window( "Hi there2", << Modal ) ) );
```

Capturing Journals

There is an optional argument in the journal and journal window commands. This argument captures (or freezes) the current display. That is, it converts the display tree with just a picture.

Here are the four options for arguments:

- Makes a bitmap “snapshot” of the display when it is sent to the journal, rather than sending a clone of the more editable display box structure to the journal. By freezing the display into a bitmap, references to variables from scripts in graphs are less problematic.

  ```
  << Journal("Freeze All");
  ```

- Similar to `Freeze All` but only within areas of a report called pictures.

  ```
  << Journal("Freeze Pictures");
  ```

- Similar to `Freeze Pictures` but only within frame boxes (which are within pictures).

  ```
  << Journal("Freeze Frames");
  ```

- Similar to `Freeze Frames` but only for the frame boxes that have scripts to draw something on them.

```
<< Journal("Freeze Frames with Scripts");
```

Current Journal

`Current Journal()` returns a reference to the display box at the top of the current journal display window. If no journal is open, one is created. There are no arguments.

You can add to the journal using the Append command. The following example adds a text box to the bottom of the current journal:

```
Current Journal() <<  Append( Text Box( "Hello World" ) );
```

You can also find items in an existing journal by enclosing the search specification in square brackets. Suppose that the current journal includes the string "Parameter Estimates". The following script appends a text box to the bottom of that journal:

```
Current Journal()["Parameter Estimates"] << Append( Text Box( "Asterisks show
  items significant at 0.05" ) );
```

Create Excel Workbooks

You can save open data tables in a new or existing Excel workbook by using the `Create Excel Workbook()` function. The first example adds all open data tables without changing the worksheet names:

```
dt1 = Open( "$SAMPLE_DATA/Big Class Families.jmp" );
dt2 = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
Create Excel Workbook( "c:/MyWorkbook1.xlsx" );
```

The following example creates MyWorkbook2.xlsx by combining the open Big Class Families.jmp and San Francisco Crime.jmp sample data tables.

```
dt1 = Open( "$SAMPLE_DATA/Big Class Families.jmp" );
dt2 = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );

// specify the data tables
tableList = {"Big Class Families", "San Francisco Crime"};

// specify the worksheets (optional)
sheetList = {"Families", "Crime"};
Create Excel Workbook( "c:/MyWorkbook2.xlsx", tableList, sheetList );
```

Use data table references to create the workbook.

```
dt1 = Open( "$SAMPLE_DATA/Big Class Families.jmp" );
dt2 = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
Create Excel Workbook(
  "c:/MyWorkbook3.xlsx",
  {dt1, dt2}, // data table references
```

```
    {"Families", "Crime"}
);
```

`Create Excel Workbook()` also accepts a mix of data table references and data table names in the list. For example, `{dt1, dt2}` above can also be expressed as `{dt1, "San Francisco Crime"}`.

Notes:

- The saved file is an .xlsx file; the .xls format is not supported.
- The JMP data tables must conform to the maximum Excel limit of 1 million rows and 16,535 columns, or the data will be truncated in Excel.
- Data table references or data table names must always be in a list, even if the list has only one item.

Prevent Changes to a Data Table

To prevent any changes to the data table (such as changing column names, changing the data type, deleting rows, and so on), use `Lock Data(1)`.

```
dt << Lock Data Table( 1 );
```

To prevent modifying adding or deleting columns and rows and modifying cells, use `Set Edit Lock`:

```
dt << Set Edit Lock( "Add Columns" );
dt << Set Edit Lock( "Delete Columns" );
dt << Set Edit Lock( "Add Rows" );
dt << Set Edit Lock( "Delete Rows" );
dt << Set Edit Lock( "Modify Cells" );
```

To get a list of data table operations that are locked, use `Get Edit Lock`:

```
dt << Get Edit Lock();
   {"Modify Cells", "Add Rows", "Delete Rows", "Add Columns", "Delete Columns"}
```

To allow a specific data table operation again, use `Clear Edit Lock`:

```
dt << Clear Edit Lock( "Modify Cells" );
```

Advanced Data Table Scripting

This section covers more advanced actions that you can perform on a data table.

- "Select Columns in a Data Table"
- "Store Summary Statistics in Global Variables"
- "Create a Table of Summary Statistics"
- "Subset a Data Table"
- "Add a Data Filter"
- "Sort a Data Table"
- "Stack Values in a Data Table"
- "Split Values in a Stacked Data Table"
- "Transpose a Data Table"
- "Vertically Concatenate Data Tables"
- "Horizontally Concatenate Data Tables"
- "Virtually Join Data Tables"
- "Replace Data in Data Tables"
- "Create a Table Using Tabulate"
- "Copy Data Tables"
- "Find Missing Data Patterns"
- "Compare Data Tables"
- "Create a Summary Table"
- "Subscribe to a Data Table"
- "Move Data Between Matrices and Data Tables"
- "Add Hyperlinks to Text Columns"

Select Columns in a Data Table

To select one or more columns, send the `Select Columns` message to the data table.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
clist = {:Height, :Weight};
dt << Select Columns( clist );
```

To select all columns, send the Select Columns message with the All argument to the data table.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Columns( all );
```

Store Summary Statistics in Global Variables

The `Summarize` command collects summary statistics for a data table and stores them in global variables. The `Summarize` command is different from the `Summary` command, which also calculates summary statistics, but presents them in a new data table.

The first argument is an optional data table reference. Include it if more than one data table might be open.

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt2 = Open( "$SAMPLE_DATA/Animals.jmp" );
Summarize( dt1,
   exg = By( :sex ),
   exm = Mean( :height )
);
Show( exg );
Show( exm );
```

Named arguments include the following: `Count`, `Sum`, `Mean`, `Min`, `Max`, `StdDev`, `First`, `Corr`, and `Quantile`. These statistics can be calculated only for numeric columns. Each argument takes a data column argument.

Notes:

- If a *name*=`By(`*groupvar*`)` statement is included, then a list of subgroup statistics is assigned to each *name*.
- `Count` does not require a column argument, but it is often useful to specify a column to count the number of nonmissing values.
- `Quantile` also takes a second argument for specifying which quantile, such as 0.1 for the 10th percentile.
- When you summarize data that contain row states, the summary statistics are not calculated. An error appears in the log instead.

Note: Excluded rows are excluded from `Summarize` calculations. If all data are excluded, Summarize returns lists of missing values. If all data have been deleted (there are no rows), Summarize returns empty lists.

The following example uses the Big Class sample data table:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Summarize(
   a = By( :age ),
   c = Count,
   sumHt = Sum( :height ),
   meanHt = Mean( :height ),
   minHt = Min( :height ),
```

```
    maxHt = Max( :height ),
    sdHt = Std Dev( :height ),
    q10Ht = Quantile( :height, .10 )
);
Show( a, c, sumHt, meanHt, minHt, maxHt, sdHt, q10Ht );
```

Because the script included a By group, the results are a list and six matrices:

```
a = {"12", "13", "14", "15", "16", "17"}
c = [8, 7, 12, 7, 3, 3]
sumHt = [465, 422, 770, 452, 193, 200]
meanHt = [58.125, 60.28571428571428, 64.16666666666667, 64.57142857142857,
   64.33333333333333, 66.66666666666667]
minHt = [51, 56, 61, 62, 60, 62]
maxHt = [66, 65, 69, 67, 68, 70]
sdHt = [5.083235752381126, 3.039423504234876, 2.367712103711172,
   1.988059594776032, 4.041451884327343, 4.163331998932229]
q10Ht = [51, 56, 61.3, 62, 60, 62]
```

You can format the results using `Table Box`.

```
New Window( "Summary Results",
  Table Box(
     String Col Box( "Age", a ),
     Number Col Box( "Count", c ),
     Number Col Box( "Sum", sumHt ),
     Number Col Box( "Mean", meanHt ),
     Number Col Box( "Min", minHt ),
     Number Col Box( "Max", maxHt ),
     Number Col Box( "SD", sdHt ),
     Number Col Box( "Q10", q10Ht )
  )
);
```

Figure 9.7 Results from Summarize

Age	Count	Sum	Mean	Min	Max	SD	Q10
12	8	465	58.125	51	66	5.08324	51
13	7	422	60.2857	56	65	3.03942	56
14	12	770	64.1667	61	69	2.36771	61.3
15	7	452	64.5714	62	67	1.98806	62
16	3	193	64.3333	60	68	4.04145	60
17	3	200	66.6667	62	70	4.16333	62

You can add totals to the window:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );

Summarize(
  a = By( :age ),
  c = Count,
  sumHt = Sum( :height ),
```

```
    meanHt = Mean( :height ),
    minHt = Min( :height ),
    maxHt = Max( :height ),
    sdHt = Std Dev( :height ),
    q10Ht = Quantile( :height, .10 )
);

Summarize(
    tc = Count,
    tsumHt = Sum( :height ),
    tmeanHt = Mean( :height ),
    tminHt = Min( :height ),
    tmaxHt = Max( :height ),
    tsdHt = Std Dev( :height ),
    tq10Ht = Quantile( :height, .10 )
);

Insert Into( a, "Total" );
c = c |/ tc;
sumHt = sumHt |/ tsumHt;
meanHt = meanHt |/ tmeanHt;
minHt = minHt |/ tminHt;
maxHt = maxHt |/ tmaxHt;
sdHt = sdHt |/ tsdHt;
q10Ht = q10Ht |/ tq10Ht;

New Window( "Summary Results",
    Table Box(
        String Col Box( "Age", a ),
        Number Col Box( "Count", c ),
        Number Col Box( "Sum", sumHt ),
        Number Col Box( "Mean", meanHt ),
        Number Col Box( "Min", minHt ),
        Number Col Box( "Max", maxHt ),
        Number Col Box( "SD", sdHt ),
        Number Col Box( "Q10", q10Ht )
    )
);
```

Figure 9.8 Summarize with Total

Age	Count	Sum	Mean	Min	Max	SD	Q10
12	8	465	58.125	51	66	5.08324	51
13	7	422	60.2857	56	65	3.03942	56
14	12	770	64.1667	61	69	2.36771	61.3
15	7	452	64.5714	62	67	1.98806	62
16	3	193	64.3333	60	68	4.04145	60
17	3	200	66.6667	62	70	4.16333	62
Total	40	2502	62.55	51	70	4.24234	56.2

If you do *not* specify a By group, the result in each name is a single value:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Summarize(
// a = By( :age ),
   c = Count,
   sumHt = Sum( :height ),
   meanHt = Mean( :height ),
   minHt = Min( :height ),
   maxHt = Max( :height ),
   sdHt = Std Dev( :height ),
   q10Ht = Quantile( :height, .10 )
);
Show( c, sumHt, meanHt, minHt, maxHt, sdHt, q10Ht );
   c = 40;
   sumHt = 2502;
   meanHt = 62.55;
   minHt = 51;
   maxHt = 70;
   sdHt = 4.24233849397192;
   q10Ht = 56.2;
```

Summarize supports multiple By groups. For example, in Big Class.jmp, proceed as follows:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Summarize( g = By( :age, :sex ), c = Count() );
Show( g, c );
   g = {"12", "12", "13", "13", "14", "14", "15", "15", "16", "16", "17", "17"},
      {"F", "M", "F", "M", "F", "M", "F", "M", "F", "M", "F", "M"}}
   c = [5,3,3,4,5,7,2,5,2,1,1,2]
```

If you specify a By group, the results are always matrices. Otherwise, the results are scalars.

Create a Table of Summary Statistics

The Summary command creates a new table of summary statistics according to the grouping columns that you specify. Do not confuse Summary with Summarize, which collects summary statistics for a data table and stores them in global variables. See "Store Summary Statistics in Global Variables".

```
summDt = dt << Summary(
   Group( groupingColumns ),
   Subgroup( subGroupColumn ),
   Statistic( columns ),// where statistic is Mean, Min, Max, Std Dev, and so
   on.
   Output Table Name( newName ) );
```

The following example creates a new table with columns for the mean of height and weight by age, and the maximum height and minimum weight by age:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
summDt = dt << Summary(
  Group( :age ),
  Mean( :height, :weight ), Max( :height ), Min( :weight ),
  Output Table Name( "Height-Weight Chart" ) );
```

Tip: `Output Table Name` can take a quoted string or a variable that is a string.

By default, a summary table is linked to the original data table. If you want to produce a summary that is not linked to the original data table, add this option to your `Summary` message:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
summDt = dt << Summary( Group( :age ), Mean( :height ),
  Link to Original Data Table( 0 )
);
```

You can choose to add marginal statistics for grouping variables to the output columns of your data table. JMP also adds a row to the end of the data table that summarizes each level of the grouping variables. To add marginal statistics, add this option to your `Summary` message:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
summDt = dt << Summary( Group( :age ), Mean( :height ),
  Include marginal statistics
);
```

You can specify the format of the statistics column name using `statistics column name format()`. Table 9.2 shows the option and examples:

Table 9.2 Statistics Column Name Format Options and Examples

Option	Example
stat (column)	Mean (Profits ($M))
column	Profits ($M)
stat of column	Mean of Profits ($M)
column stat	Profits ($M) Mean

For example, add an option like this to your `Summary` message:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
summDt = dt << Summary( Group( :age ), Mean( :height ),
  statistics column name format( "stat of column" )
);
```

The height column is named Mean of height.

Subset a Data Table

Subset() creates a new data table from rows that you specify. If you specify no rows, Subset uses the selected rows. If no rows are selected or specified, it uses all rows. If no columns are selected or specified, it uses all columns. And if Subset has no arguments, the Subset window appears.

```
dt << Subset((<"Private">|<"Invisible">), <Selected Columns>, <Columns(column
   list)>, <Selected Rows>, <Rows([number, number, ...])>, <By(column
   list)>, <Sampling Rate(fraction)>, <Sample Size(integer)>,
   <Stratify(column list)>, <Link to Original Data Table(Boolean)>, <Copy
   Formula(Boolean)>, <Suppress Formula Evaluation(Boolean)>, <Keep by
   Column>);
```

Note: For more arguments, see the Scripting Index in the Help menu.

For example, using Big Class.jmp, to select the columns for all rows in which the age is 12:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( Selected( Row State() ) = (:age == 12) );
subdt = dt << Subset( Output Table Name( "Subset" ) );
```

To select three columns and all rows, run this script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
subDt1 = dt << Subset(
   Columns( :name, :age, :height ),
   Output Table Name( "Big Class 2" )
);
```

To select specified rows of two columns and link the two data tables, run this script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
subDt2 = dt << Subset(
   Columns( :name, :weight ),
   Rows( [2, 4, 6, 8] ),
   Linked
);
```

To select the columns for all rows in which the age is 12, run this script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age == 12 );
dt << Subset( ( Selected Rows ), Output Table Name( "Subset" ) );
```

To verify that rows were selected and then conditionally proceed to subset the current data table or open another data table, run this script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age == 12 );

/* If the condition selected no rows, open Fitness.jmp.
   Else, generate the desired subset table of Big Class.jmp.*/
If( N Rows( dt << Get Selected Rows ) == 0,

newdt = Open( "$SAMPLE_DATA/Fitness.jmp" ),
   newdt = dt << Subset( All Columns( 1 ), Selected Rows( 1 ) )
);
```

Add a Data Filter

The Data Filter provides a variety of ways to identify subsets of data. Using Data Filter commands and options, you can select complex subsets of data, hide these subsets in plots, or exclude them from analyses.

Here is the basic syntax:

```
dt << Data Filter(<Columns>, <Mode(...)>, <Add Filter(...)>, <Show Columns(),
   <Selector>, <Animation>, <Animate>, <Clear>, <Delete All>, <Delete>,
   <Filter Columns>, <Filter Column>, <Filter Group>, <Add Filter>, <Add>,
   <Match>, <Mode>, <Display>, <Where>, <Make Subset>, <Display>, <Show
   Histograms and Bars>)
```

You can send an empty `Data Filter` message to a data table, and the initial Data Filter window appears, showing the Add Filter Columns panel that lists all the variables in the data table.

`Mode` takes three arguments, all of which are optional: `Select(`*`Boolean`*`) Show(`*`Boolean`*`), Include(`*`Boolean`*`)`. These arguments turn on or off the corresponding options. The default value for `Select` is true (1). The default value for `Show` and `Include` is false (0).

`Add Filter()` adds rows and builds the WHERE clauses that describe a subset of the data table. Here is the basic syntax:

```
Add Filter( Columns( col, ... ), Where( ... ), ... )
```

To add columns to the data filter, list the columns names separated by commas. Note that this is not a list data structure.

You can define one or more WHERE clauses to specify the filtered columns:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
df = dt << Data Filter(
   Mode( Show( 1 ) ),
```

```
    Add Filter(
       Columns( :age, :sex, :height ),
       Where( :age == {13, 14, 15} ),
       Where( :sex == "M" ),
       Where( :height >= 52 & :height <= 65 )
    )
);
```

This script selects 13, 14, and 15-year-old males with a height greater than or equal to 52 and less than or equal to 65. In this filtered data, you can instead select the excluded ages by adding the `Invert Selection` message to the preceding script.

```
df << ( Filter Column( :age ) << Invert Selection );
```

In this example, ages other than 13, 14, and 15 in the filtered data are selected.

You can also use `Add Filter()` to select matching strings from columns with the Multiple Response property or Multiple Response modeling type.

```
dt = Open( "$SAMPLE_DATA/Consumer Preferences.jmp" );
df = dt << Data Filter(
   Location( {437, 194} ),
   Add Filter(
      Columns( :Brush Delimited ),
      Match None( Where( :Brush Delimited == {"Before Sleep", "Wake"} ) ),

      /* Define the display options. "Brush Delimited" appears above
      the list of values. Size is the size of the filter list. The
      values appear next to check boxes. */
      Display( :Brush Delimited, Size( 121, 70 ), Check Box Display )
      )
);
```

This script selects rows with values in the Brush Delimited column that do not match either of the specified values ("Before Sleep", "Wake"). Other available scripting options include `Match Any`, `Match All`, `Match Exactly`, and `Match Only`. See *Using JMP* for more information about the Multiple Response property and the Multiple Response modeling type.

You can also send messages to an existing Data Filter object:

```
Clear(), Display( ... ), Animate(), Mode(), ...
```

`Clear` takes no arguments and clears the data filter.

To prevent the data filter from appearing when the script is run, set `Show Window` to 0:

```
obj  = ( dt << Data Filter( Show Window( 0 ) ) );
```

To specify which values to exclude, use the `!=` operator. The following example excludes ages 16 and 17 from the filtered values.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
```

```
df = dt << Data Filter(
   Add Filter( Columns( :age, :sex ) ),
   Match( Columns( :age, :sex ),
      Where( :sex = "M" ), Where( :age != {16, 17} )
   )
);
```

To omit the Data Filter outline and red triangle menu, specify `No Outline Box`.

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
obj = dt << Data Filter(
   Add Filter( Columns( :Region, :POP ), Where( :Region == {"C", "N"} ) ),
   No Outline Box( 1 )
);
```

Add a Local Data Filter

A local data filter enables you to filter data in an individual report window without affecting other reports. When the local data filter is added to the platform, the reference to the data filter is returned. This provides an easy way to get access to the filter.

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
biv = dt << Bivariate(
   Y( :x ),
   X( :y ),
   Automatic Recalc( 1 ),
   Fit Line( {Line Color( {213, 72, 87} )} )
);

dfObj = biv << Local Data Filter(
// Include Show Histograms and Bars(0) to hide them in the filter.
// Enable Include to include rows selected by the local data filter in
// calculations that are required to build graphs controlled by the local data
// filter.  Deselected rows are excluded from calculations.
   Location( {566, 47} ),
   Mode( Include( 1 ) )
);

dfObj << Add Filter(
   Columns( :Region, :POP ),
   Where( :Region == {"C", "N"} )
);
```

Track Data Filter Changes

Make Filter Change Handler enables you to track changes to the data filter.

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
```

```
dist = Distribution(
  Automatic Recalc( 1 ),
  Continuous Distribution( Column( :POP ) )
);
filter = dist << Local Data Filter(Add Filter( columns( :Region ) ));
f = Function( {a}, Print( a ) );
rs = filter << Make Filter Change Handler( f );
```

Define the Context of a Data Filter

A data filter lets you interactively select complex subsets of data, hide the subsets in plots, or exclude them from analyses. Your selections affect all analyses of the data table.

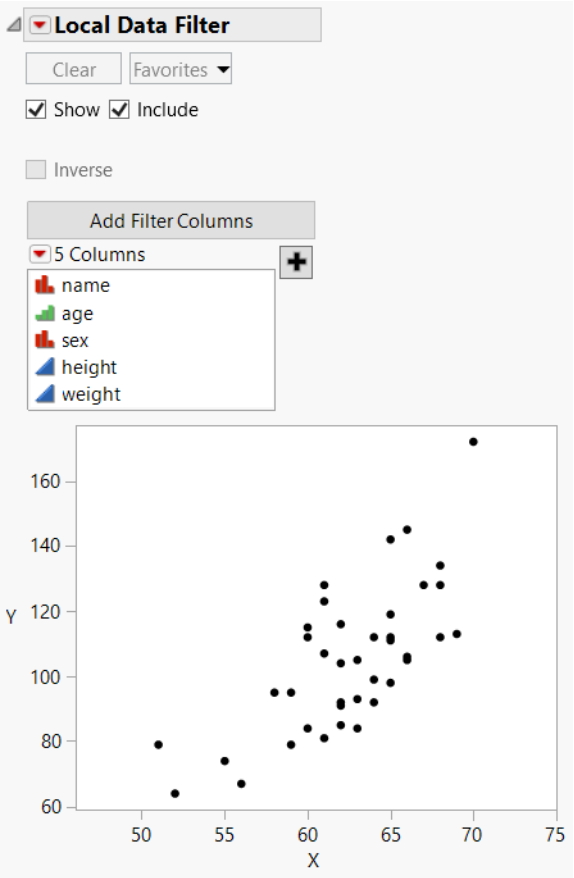
Another option is to filter data from specific platforms or display boxes. Create a local data filter inside the `Data Filter Context Box()` function. This defines the context as the current platform or display box rather than the data table.

The following example creates a local data filter for a Graph Box. See Figure 9.9 for the output.

```
New Window( "Marker Seg Example",
  Data Filter Context Box(
    V List Box(
      dt << Data Filter( Local ),
      g = Graph Box(
        Frame Size( 300, 240 ),
        X Scale( Min( xx ) - 5, Max( xx ) + 5 ),
        Y Scale( Min( yy ) - 5, Max( yy ) + 5 ),
        Marker Seg( xx, yy, Row States( dt, rows ) )
      )
    )
  )
);
```

Tip: The preceding script is part of a larger script that first builds the arrays required for a marker seg. To experiment with this script, open the Local Data Filter for Custom Graph.jsl sample script.

Figure 9.9 Local Data Filter and Graph



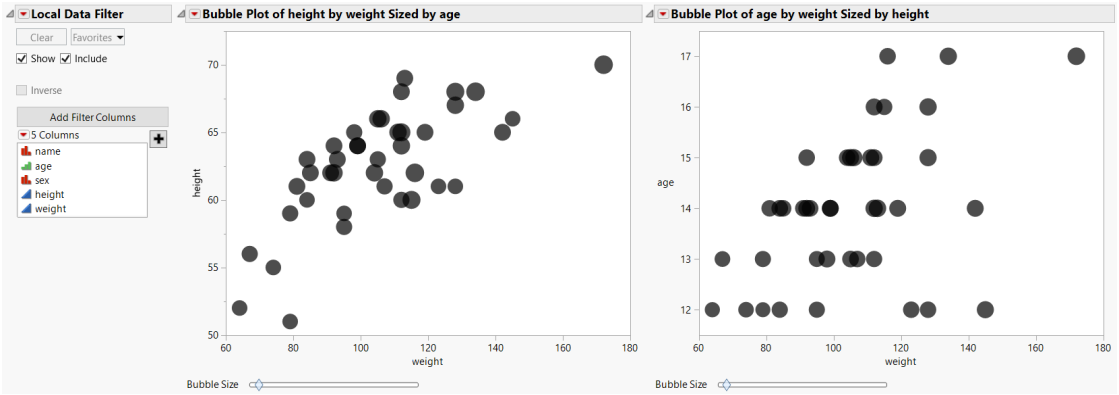
A context box can also contain several graphs with one local filter. The filtering then applies to both graphs. The following script creates two bubble plots and one data filter in one context box. See Figure 9.10 for the output.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
New Window( "Shared Local Filter",
   Data Filter Context Box(
      H List Box(
         dt << Data Filter( Local ),
         Platform(
            dt,
            Bubble Plot( X( :weight ), Y( :height ), Sizes( :age ) )
         ),
         Platform(
            dt,
            Bubble Plot( X( :weight ), Y( :age ), Sizes( :height ) )
         )
```

```
      )
   )
);
```

Tip: To experiment with this script, open the Local Data Filter Shared.jsl sample script.

Figure 9.10 Local Filter with Two Bubble Plots



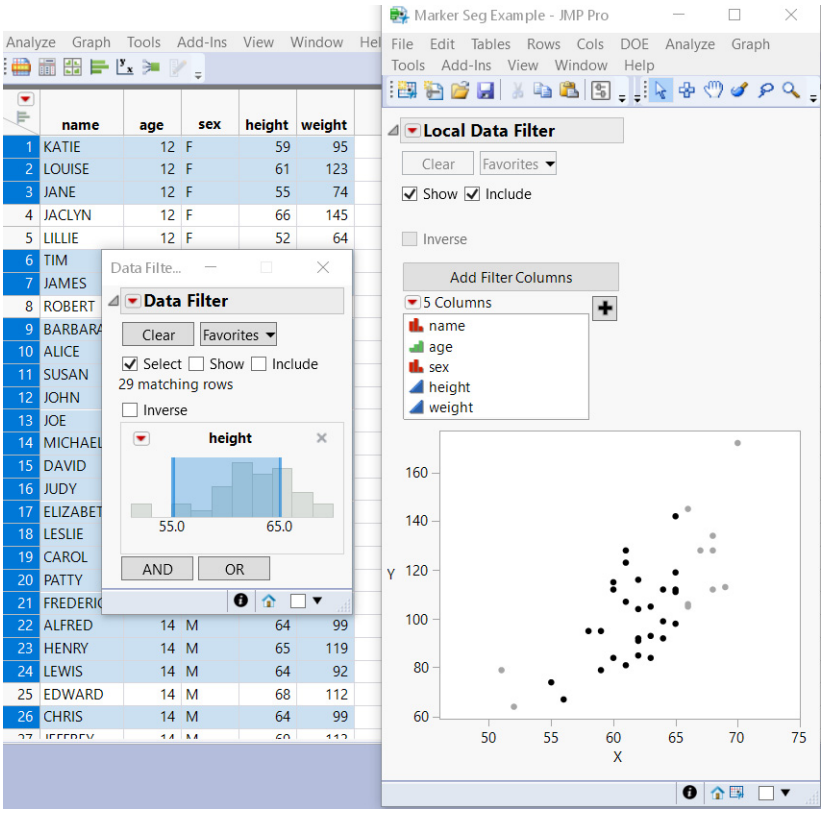
Data filters can be hierarchical. One script might generate a data filter and a local data filter. The outer data filter for the data table determines which data is available for the local data filter. To illustrate this point, the following script produces both types of data filters as shown in Figure 9.11.

Tip: The following script is part of a larger script that first builds the arrays required for a marker seg. See the Local Data Filter for Custom Graph.jsl sample script for code that builds those arrays.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
...
New Window( "Marker Seg Example",
   Data Filter Context Box(
      V List Box(
         dt << Data Filter( Local ),
         g = Graph Box(
            Frame Size( 300, 240 ),
            X Scale( Min( xx ) - 5, Max( xx ) + 5 ),
            Y Scale( Min( yy ) - 5, Max( yy ) + 5 ),
            Marker Seg( xx, yy, Row States( dt, rows ) )
         )
      )
   )
);
```

Heights greater than or equal to 55 and less than or equal to 65 are initially filtered. Then the user can work with the local data filter to filter columns on the graph.

Figure 9.11 Data Filter Hierarchy



Export a Report as Interactive HTML with a Live Data Filter

The preceding examples of adding a data filter enabled the `Include` mode. This mode includes rows selected by the local data filter in calculations that are required to build graphs controlled by the local data filter. Deselected rows are excluded from calculations.

For an Interactive HTML export, disable the `Include` mode, because Interactive HTML doesn't exclude rows from calculations.

```
dt = Open( "$SAMPLE_DATA/Quality Control/Coating.jmp" );
dt << Graph Builder(
   Size( 532, 455 ),
   Show Control Panel( 0 ),
   Variables( X( :Sample ), Y( :Weight ) ),
   Elements( Points( X, Y, Legend( 3 ) ), Line( X, Y, Legend( 5 ) ) ),
   Local Data Filter( Mode( Include( 0 ) ), Add Filter( columns( :Weight ) ) )
```

```
);
Current Report() << Save Interactive HTML( "$DESKTOP/example1.html" );
Web( "$DESKTOP/example1.html" );
```

Sort a Data Table

Sort() rearranges the rows of a table according to the values of one or more columns, either replacing the current table or creating a new table with the results. Specify ascending or descending sort for each By column.

```
dt << Sort(("<Private>"|"<Invisible>"), <Replace Table>, By(columns),
  Order(Descending | Ascending), <Output Table Name(name)>)>
```

The following example creates a new data table based on Big Class.jmp that sorts the data in descending order by age and by name:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
sortedDt = dt << Sort(
  By( :age, :name ),
  Order( Descending, Ascending ),
  Output Table Name( "Sorted age name" ) );
```

You can also use an associative array to sort values in a column. The associative array gives you a quick look at unique values in a column. See "Sort a Column's Values in Lexicographic Order".

Stack Values in a Data Table

Stack() combines values from several columns into one column.

```
dt << Stack(
  Columns ( columns ),  // the columns to stack together
  Source Label Column ( "name" ), // to identify source columns
  Stacked Data Column ( "name" ), // name for the new stacked column
  Number of Series( n ), // stacks selected columns into two or more columns
  Contiguous, // specifies that the series consists of adjacent columns
  Keep ( columns ), // the columns to keep in the data table
  Drop ( columns ), // the columns to drop in the data table
  Output Table ( "name" ), // the name for the new data table
  Columns( columns ) ), // specify which columns to include in the stacked
  table
  Copy Formula( Boolean ), // copy the formula into the stacked column
  Drop All Other Columns( Boolean ), // omit the non-stacked columns from the
  new table
  "non-stacked columns"n )( Keep( col1, ... ) // the non-stacked columns to
  keep
```

```
"non-stacked columns"n )( Drop( col1, ... ) // the non-stacked columns to
omit
Suppress Formula Evaluation( Boolean ) // do not evaluate the formula
```

For example, the following script stacks the weight and height columns in Big Class.jmp:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
StackedDt = dt << Stack(
    Columns( :weight, :height ),
    Source Label Column( "ID" ),
    Stacked Data Column( "Y" ),
   "Non-stacked columns"n )( Keep( :age, :sex ) ),
    Output Table( "Stacked Table" )
);
```

The `Columns(columns)` argument can take a list of columns, or an expression that evaluates to a list.

You can also stack multiple sets of columns into multiple columns.

```
dt = Open( "$SAMPLE_DATA/Blood Pressure.jmp" );
dt << Stack(
  Columns(
     :BP 8M,
     :BP 12M,
     :BP 6M,
     :BP 8W,
     :BP 12W,
     :BP 6W,
     :BP 8F,
     :BP 12F,
     :BP 6F
  ),
  Stacked Data Column( "BP" ),
  Source Label Column( "Day" ),
  Number of Series( 3 ), // number of sets in each series
  Contiguous, /* If you want to stack vertically. Otherwise, omit the
  argument. */

);
```

Split Values in a Stacked Data Table

`Split()` breaks a stacked column into several columns.

```
dt << Split(
  Split( columns ),
  // the column to split (required)
```

```
Split by( column ),
// the column to split by (required)

Group(column),
// split data within groups

<Private>|<Invisible>,
// resulting table is private or invisible

Remaining Columns( Keep All | Drop All | Select( columns ) ),
/* specify what to do with the remaining columns in the resulting table
(Keep All by default) */

<Copy formula( 0|1 )>,
/* include column formulas from the source table in the resulting table
(default is 1, true) */

<Suppress formula evaluation( 0|1 )>,
// stop any copied formulas from being evaluated (default is 1, true)

Sort by Column Property,
/* Sort the order of the output columns by the sort column property that is
defined for the Split by column. */

Output Table( "name" ));
// generate the output to the table name specified
```

The following example reverses the previous example for Stack(), returning essentially the original table, except that the height and weight columns now appear in alphabetic order:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Stack(
    Columns( :weight, :height ),
    Source Label Column( "ID" ),
    Stacked Data Column( "Y" ),
   "Non-stacked columns"n( Keep( :age, :sex ) ),
    Output Table( "Stacked Table" )
);
dt2 = Data Table( "Stacked Table" );
dt2 << Split(
/* Split adds the column or columns that you want to split into several
new columns. */
  Split( Y ),
/* Split By adds the column whose values you want to use as the new column
   names, and as the basis for splitting the column. */
  Split By( ID ),
```

```
  Output Table( "Split" ) );
```

When multiple rows are mapped to the same row, only one value is kept.

```
dtabc = New Table( "abc",
 New Column( "Column 1", Character, Set Values( {"a", "a", "b", "b", "c"} )),
 New Column( "Column 2", Character, Set Values( {"x", "y", "x", "y", ""} )),
 New Column( "Column 3", Set Values( [1, 2, 3, 4, 5] ))
);
/* Split adds Column 3. You want to split the values (1, 2, 3..) into several
   new cols.
Split By adds Column 2. You want to use the values (x, y) as the new column
   names. */
dtabc << Split( Split By( :Column 2 ), Split( :Column 3 ));
```

Transpose a Data Table

`Transpose` creates a new data table by flipping a data table on its side, interchanging rows for columns and columns for rows. If you specify no rows, `Transpose()` uses the selected rows. If no rows are selected, it uses all rows.

```
dt << Transpose(
   "private", "invisible",
   columns( columns ),
   Rows( row matrix ),
   By ( column ),
   Label column name( "name" ),
   Output Table( "name" )
);
dt << Transpose(
   Columns( columns ),
   Rows( row matrix ),
   Output Table Name( "name" )
);
```

The following example transposes the height and weight columns in the Big Class.jmp sample data table:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
tranDt = dt << Transpose( Columns( :height, :weight ),
   Output Table Name( "Transposed Columns" ) );
```

Note: The simple transpose command `dt << Transpose` brings up the Transpose window. If you do not want the window to appear, invoke the transpose as `dt << Transpose(no option)`.

Vertically Concatenate Data Tables

Concatenate, also known as a vertical join, combines rows of several data tables top to bottom.

```
dt << Concatenate( DataTableReferences,...,Keep Formulas,
  Output Table Name( "name" ) );
```

For example, if you have subsetted tables for males and females, you can put them back together using Concatenate:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :sex == "M" );
m = dt << Subset( Output Table Name( "M" ) );
dt << Invert Row Selection;
f = dt << Subset( Output Table Name( "F" ) );
both = m << Concatenate( f, Output Table Name( "Both" ) );
```

Or, instead of creating a new table containing all the concatenated data, you can append all the data to the current data table:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :sex == "M" );
m = dt << Subset( Output Table Name( "M" ) );
dt << Invert Row Selection;
f = dt << Subset( Output Table Name( "F" ) );
both = dt << Concatenate( m, f, "Append to First Table" );
```

Horizontally Concatenate Data Tables

Join, also known as horizontal join or concatenate, combines data tables side to side.

```
dt << Join( // message to the first table
  With( dataTable ), // the secondary data table
  Select( columns ), /* select columns from the main table
                      to add to the output table */
  Select With( columns ), /* select columns from the secondary table to
                           add to the output table */
  // join type; alternatives are
  Cartesian join, By Row Number, By matching columns(col1=col2, ...)
  Merge Same Name Columns, // merge columns with the same name
  Copy Formula( 0 ), // on by default; 0 turns it off
  Suppress Formula Evaluation( 0 ), // on by default; 0 turns it off
  Match Flag( 1 ), /* add the Match Flag column from the joined data table
  when you are matching by column */
  Update, /* replace the data in the main table with the corresponding data
  from the secondary table */
  // options for each table:
```

```
Drop Multiples( Boolean, Boolean ), // include all rows from the data table
Include Non Matches( Boolean, Boolean ), // include non-matching rows
Preserve Main Table Order(), // maintain the order of the original data
Output Table Name( "name" )); // the resulting table
```

To try this, first break Big Class.jmp into two parts:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
part1 = dt << Subset(
  Columns( :name, :age, :height ),
  Output Table Name( "Big Class 1" )
);
part2 = dt << Subset(
  Columns( :name, :sex, :weight ),
  Output Table Name( "Big Class 2" )
);
```

To make it a realistic experiment, rearrange the rows in part 2:

```
sortedPart2 = part2 << Sort( By( :name ), Output Table Name( "Sorted_Big
  Class" ) );
```

Now you have a data set in two separate chunks, and the rows are not in the same order, but you can join them together by matching on the column that the two chunks have in common.

```
part1 << Join(
  With( sortedPart2 ),
  By Matching Columns( :name == :name ),
  Preserve Main Table Order();
  Output Table Name( "Joined Parts" );
);
```

The resulting table has two copies of the name variable, one from each part, and you can inspect these to see how `Join` worked. Notice that you now have four Robert rows, because each part had two Robert rows (there were two Roberts in the original table) and `Join` formed all possible combinations.

Tip: To maintain the order of the original data table in the joined table (instead of sorting by the matching columns), include `Preserve Main Table Order()`. This function speeds up the joining process.

See the *JSL Syntax Reference* for more information about the Join arguments.

Virtually Join Data Tables

Virtual Join links a main data table to one or more auxiliary data tables. The feature enables the main data table to access data from the auxiliary data tables without physically joining the tables. See *Using JMP*.

The following example shows how to virtually join data tables and create distributions from the joined data:

```
dt1 = Open( "$SAMPLE_DATA/Pizza Profiles.jmp" );
dt1:ID << Set Property( "Link ID", 1 );// add Link ID and turn it on

dt2 = Open( "$SAMPLE_DATA/Pizza Responses.jmp" );

// add Link Reference to the Choice1, Choice2, and Choice columns
dt2:Choice1 << Set Property( "Link Reference", Reference Table(
  "$SAMPLE_DATA/Pizza Profiles.jmp" ) );
dt2:Choice2 << Set Property( "Link Reference", Reference Table(
  "$SAMPLE_DATA/Pizza Profiles.jmp" ) );
dt2:Choice << Set Property( "Link Reference", Reference Table(
  "$SAMPLE_DATA/Pizza Profiles.jmp" ) );

obj = dt2 << Distribution( // create a distribution of Crust[Choice1]
  Weight( :Subject ),
  Nominal Distribution(
    Column(
      Referenced Column(
        "Crust[Choice1]", // specify the joined column name
        Reference( Column( :Choice1 ), Reference( Column( :Crust ) ) )
      )
    )
  ),
  Nominal Distribution( Column( :Choice1 ) )
);
```

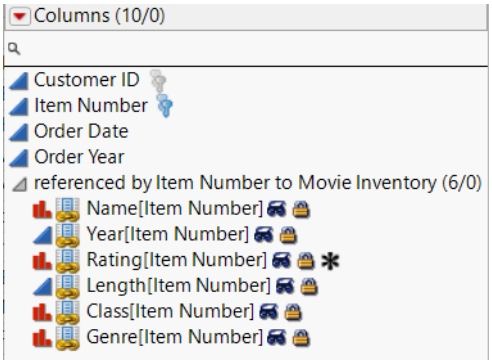
For another example of virtual join, search the Help > Scripting Index for “virtual join”.

Specifying Linked Column Names

The Columns panel in a data table shows you the names of the linked columns. For example, open the Movie Inventory.jmp and Movie Rentals.jmp sample data tables to see the linked columns shown in Figure 9.12.

The column name of the source column is followed by the name of the referencing column in brackets. For example, Figure 9.12 shows the Name source column followed by the referencing column [Item Number] in brackets.

Figure 9.12 Linked Columns in Movie Rentals.jmp



You change the name of the virtually linked column name with `Set Title`. For example, the following script changes the name of the virtually linked column Rating[Item Number] to Movie Rating in the Distribution report.

```
dt1 = Open( "$SAMPLE_DATA/Movie Inventory.jmp" );
dt2 = Open( "$SAMPLE_DATA/Movie Rentals.jmp" );
dt2 << Distribution(
   Nominal Distribution(
      Column(
         Referenced Column( // specify the joined column name
            "Rating[Item Number]",
            Reference( Column( :Item Number ), Reference( Column( :Rating ) ) )
         )
      )
   ),
   SendToReport( Dispatch( {}, "Rating[Item Number]", OutlineBox,
      {Set Title( "Movie Rating" )} ) ) /* change the name of the
      virtually linked column */
);
```

Note that running the script automatically opens the referenced table (the one with the Link ID property), just like running any other platform script, before the referenced table is opened.

You can also specify that the referenced column name be used in linked columns. This option lets you define a shorter column name (for example, Rating instead of Rating[Item Number]). Include the `Use Linked Column Name` option:

```
dt1 = Open( "$SAMPLE_DATA/Movie Inventory.jmp" );
dt2 = Open( "$SAMPLE_DATA/Movie Rentals.jmp" );

Column( dt2, "Item Number" ) << Set Property(
   "Link Reference",
   {Reference Table( "Movie Inventory.jmp" ), Options( "Use Linked Column
   Name"( 1 ) )}
```

```
);
dt2 << Distribution(
  Nominal Distribution(
     Column(
        Referenced Column( // specify the joined column name
           "Rating",
           Reference( Column( :Item Number ), Reference( Column( :Rating ) ) )
        )
     )
  )
);
```

Merging Referenced Data into One Data Table

To share the data from virtually joined data tables with another user, you might want to merge the data so that the data tables are permanently joined. This option is helpful because you provide one data table, not the main and auxiliary data tables.

When you merge referenced data, the data are replicated from the auxiliary data table to the main data table. When you save the main data table, the actual data are also saved.

The following example shows how to merge two referenced columns:

```
dt1 = Open( "$SAMPLE_DATA/Pizza Profiles.jmp" );
dt2 = Open( "$SAMPLE_DATA/Pizza Responses.jmp" );

dt1:ID << Set Property( "Link ID", 1 );

dt2:Choice << Set Property( "Link Reference", Reference Table( dt1 ) );
dt2:Choice1 << Set Property( "Link Reference", Reference Table( dt1 ) );
dt2:Choice2 << Set Property( "Link Reference", Reference Table( dt1 ) );

// select the columns to merge
dt2 << Select Columns( :Choice1, :Choice2 );

dt2 << Merge Referenced Data();
/* linked data from the referenced columns are now embedded in
Pizza Responses.jmp */
```

Note: If you don't select the columns to merge, all columns are merged.

Replace Data in Data Tables

Note: `Merge Update()` is an alias for `Update()`.

`Update()` replaces data in one table with data from a second table.

```
dt << Update( // message to first table
  With( dataTable ), // the other data table
  By Row Number, /* default join type; alternative is
                  By Matching Columns( col1 == col2 ) */
  Ignore Missing, // optional, does not replace values with missing values
  );
```

To try this, make a subset of Big Class.jmp:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
NewHt = dt << Subset( Columns( :name, :height), Output Table Name( "hts" ) );
```

Next, add 0-6 inches to each student's height:

```
diff = Random Uniform( 0, 6 );
For Each Row( NewHt, :height = :height + diff );
```

Finally, update the heights of students in Big Class.jmp with the new heights from the subset table:

```
dt << Update(
  With( NewHt ),
  By Matching Columns( :name == :name),
  );
```

Controlling the Columns Added to an Updated Table

Your updated table might contain more columns than your original table. You can select which columns are included in your updated table using the option `Add Column from Update Table()`.

To add no additional columns:

```
Data Table( "table" ) << Update(
  With( Data Table( "update data" ) ),
  Match Columns( :ID = :ID ),
  Add Columns from Update Table( None )
);
```

To add some columns:

```
Data Table( "table" ) << Update(
  With( Data Table( "update data" ) ),
  Match Columns( :ID = :ID ),
  Add Columns from Update table( :col1, :col2, :col3 )
);
```

Create a Table Using Tabulate

Tabulate constructs tables of descriptive statistics. The tables are built from grouping columns, analysis columns, and statistics keywords. The following example creates a table containing the standard deviation and mean for the height and weight of male and female students:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Tabulate( // message to data table
  Add Table( // start a new table
     Column Table( Grouping Columns( :sex ) ), // group using the column sex
     Row Table( // add rows to the table

        // use the height and weight columns for the analysis
        Analysis Columns( :height, :weight ),

        // show the standard deviation and mean
        Statistics( Std Dev, Mean )
     )
  )
);
```

You can apply a transformation to a column in the Tabulate table and set the format of the column at the same time. Use the `Transform Column()` function inside `Analysis Column()`. For example, the following script applies the Log transformation to height and sets the column format for the Mean and % of Total columns.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Tabulate(
  Set Format(
     Mean(
        :height( 10, 1 ),
        Analysis Column(
           Transform Column(
              "Log[height]",
              Formula( Log( :height ) )
           ),
           Format( 10, "Best" )
        )
     ),
     "% of Total"n(:height( 12, 2 ),
     Analysis Column(
        Transform Column(
           "Log[height]",
           Formula( Log( :height ) )
        ),
        Format( 12, 2 )
     ))
```

```
    ),
    Add Table(
       Column Table(
          Analysis Columns(
             :height,
             Transform Column(
                "Log[height]",
                Formula( Log( :height ) )
             )
          ),
          Statistics( Mean, "% of Total"n )
       ),
       Row Table( Grouping Columns( :sex ) )
    )
);
```

Note: You can also change the width of columns and perform additional operations within Tabulate. See the JMP Scripting Index in the Help menu.

Copy Data Tables

The most efficient way to copy a data table is with the `Subset()` function. The following example subsets all rows and columns into New Table.jmp.

```
sourceDT = Open( "$SAMPLE_DATA/Big Class.jmp" );
destDT = sourceDT << Subset( Columns(), Rows(), Output Table Name( "New
  Table.jmp" ) );
```

You can also use `Copy File()`.

```
Copy File( "$SAMPLE_DATA/Big Class.jmp", "c:/MyData/New Table.jmp" );
```

`Copy File()` is equivalent to copying the file on your computer. However, you have to consider whether the destination data table with the specified name already exists. And the source file has to be saved. If the source file is not saved, or the destination file already exists, the copy fails. It's safer to use the `Subset()` method shown above.

Find Missing Data Patterns

If your data table contains a lot of missing data in different columns, you might want to see whether there is a pattern in the distribution of the missing data relative to the different columns.

```
dt << Missing Data Pattern( // message to data table
  Columns( :miss ), // find missing data in this column
  Output Table( "Missing Data Pattern" ) // name the output table
```

```
);
```

Compare Data Tables

JMP can compare two open data tables and report the differences between data, scripts, table variables, column names, column properties, and column attributes. To compare data tables using JSL:

```
obj = dt << Compare Data Tables(
Compare With( Data Table ( "Data Table Name" ) );
```

For example, to compare the data tables Students1.jmp and Students2.jmp, proceed as follows:

```
dt = Open( "$SAMPLE_DATA/Students1.jmp" );
dt2 = Open( "$SAMPLE_DATA/Students2.jmp" );
obj = dt << Compare Data Tables( Compare With( Data Table( "Students2" ) ) );
```

To view the difference summary matrix for the comparison results, use this function:

```
mtx = ( obj << Get Difference Summary Matrix );
```

The resulting matrix appears in the log window:

```
[-1 1 2 2,
-1 2 4 3,
-1 1 7 4,
1 3 8 4,
1 1 10 9,
0 1 11 11,
-1 1 14 14,
-1 1 16 15,
1 1 18 16,
-1 1 19 18,
0 1 22 20,
0 1 26 24,
1 1 29 27,
1 1 34 33]
```

In the comparison summary matrix in the example, the first column represents the action performed on a column: -1 for a Delete, 0 for a Replace, and 1 for an Add. The second column represents the number of rows affected by the action. The third and fourth columns represent the row numbers affected in the two data tables in their respective order (Students1.jmp, Students2.jmp).

Create a Summary Table

A Summary table includes summary statistics for a data table, such as the mean, median, standard deviation, minimum value, and maximum value. By default, the summary table is linked to its source table. When you select rows in the summary table, the corresponding rows are highlighted in its source table.

```
dt << Summary( <private>, <invisible>, <Group( column )>, <Weight( column )>,
   <Freq( column )>, <N>, <Mean( column )>, <Std Dev( column )>, <Min( column
   )>, <Max( column )>...
```

Include columns for any statistic listed in the Summary launch window.

The following example creates a summary table from the Fitness.jmp sample data table. The N Rows column, which specifies the number of rows for each level, is included in the Summary table by default. This script creates a column of the mean for Runtime by Sex.

```
dt = Open( "$SAMPLE_DATA/Fitness.jmp" );
dt << Summary(
       Group( :Sex ),
       Mean( :Runtime ),
);
```

You can override the preselected Freq or Weight columns in the data table by specifying the Freq or Weight columns in the script. To omit the preselected Freq or Weight, use the keyword "none".

```
dt = Open( "$SAMPLE_DATA/Fitness.jmp" );
dt << Summary(
       Group( :Sex ),
       Mean( :Runtime ),
       Weight( "none" )
);
```

To unlink the Summary table, include `Link to Original Data Table( 0 )` in the script.

```
dt = Open( "$SAMPLE_DATA/Fitness.jmp" );
dt << Summary(
     Group( :Sex ),
     Mean( :Runtime ),
     Link to Original Data Table( 0 )
);
```

Subscribe to a Data Table

If you want to receive a message when a data table changes, use the `Subscribe` message. For example, you might want a message sent to the log when columns are added or deleted.

Here is the basic syntax:

```
dt << Subscribe( keyname(<client>), On Delete Columns|On Add Columns|On Add
   Rows|On Delete Rows|On Rename Column|On Close|On Save|On Rename(function)
   );
```

The first argument is a name for the subscription (or “client”), so that it can also be removed.

Note: The subscription is to the data table. The subscriber subscribes to the broadcast from the data table; it does not subscribe to a program or to script.

The application can also subscribe as a client to the data table (for example, most built-in platforms such as Distribution). If a data table has clients when the data table is closed, the user is warned that there are applications open that might need the data table.

Each subscription remains in effect until you unsubscribe:

```
dt << Unsubscribe("keyname", On Delete Columns | On Add Columns | On Add Rows
   | On Delete Rows | On Close | On Col Rename | All);
```

The following example sends messages to the log when a row is added or deleted:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
delRowsFn = Function( {a, b, rows},
   dtname = ( a << Get Name());
   Print( dtname );
   Print( b );
   Print Matrix( rows );
);
addRowsFn = Function( {a, b, insert},
   dtname = ( a << Get Name() );
   Print( dtname );
   Print( b );
   Print( insert );
);
dt << Subscribe( "Test Delete", onDeleteRows( delRowsFn, 3 ) );
dt << Subscribe( "Test Add", onAddRows( addRowsFn, 3 ) );
```

`Subscribe to Data Table List()` subscribes to a list of data tables to be notified when a new data table has been added, closed, or renamed. The following example shows how to subscribe and unsubscribe:

```
f2 = Function( {x}, Show( x ) );
Show( Subscribe to Data Table List( "My Data", OnClose( f2 ) ) );
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Wait( 2 );
Close( dt );
Show( Unsubscribe to Data Table List( "My Data" ) );
   Subscribe to Data Table List( "My Data", OnClose( f2 ) ) = "My Data";
   x = Data Table( "Big Class" );
   Unsubscribe to Data Table List( "My Data" ) = Empty();
```

Empty Application Names

If `Subscribe` is called with an empty application name, JMP generates a unique name that is returned to the caller. In the following example, `appname2` is subscribed to the data table as a client.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
appname1 = dt << Subscribe( "", On Close( Print( "Closing Data Table" ) ) );
appname2 = dt << Subscribe(
   ""( "client" ),
   On Close(
      Function( {dtab},
         dtname = ( dtab << Get Name() );
         Print( dtname );
      )
   )
);
dt << Unsubscribe( appname1, On Close );
dt << Run Script( "Distribution" )
```

Move Data Between Matrices and Data Tables

For information about moving information between a matrix and a data table, see "Matrices and Data Tables".

Add Hyperlinks to Text Columns

You can define a text column as a column of hyperlinks using the Event Handler column property. For example, you can create a column of geographical data as hyperlinks to a Google map, or you might want to include links to a Wikipedia web pages.

Note: To learn more about the Event Handler column property, add the column property to a column that contains a hyperlink and study the property in the JMP interface. Tips about the property are provided. To see the script that creates the property, select the data table red triangle menu; select **Copy Script (No Data)**; and copy the script to an empty script.

Use the Event Handler column property to build hyperlinks based on one or more columns in your data table. This example shows how the hyperlinks in the SAS Offices.jmp sample data table were created.

1. Select **Help > Sample Data Library** and open SAS Offices.jmp.
2. Right-click the City column and select **Column Info**.

   Notice that the Event Handler column property has been added.
3. The script on the Click tab inserts the code for the URL.

```
Function( {thisTable, thisColumn, iRow},
Web(
   "https://www.google.com/maps/@"||Char(thisTable:latitude[irow])||","||Char
   (thisTable:longitude[irow])||",12z" ) );
```

- `thisTable` is the data table object.
- `thisColumn` is the column object.
- `iRow` is the index of the row in `thisColumn`.
- `Web` opens the URL in the default browser.
- The `@` sign in "https://www.google.com/maps/@" is specific to the Google API for maps. Google expects coordinates to follow it.
- `||Char(thisTable:latitude[irow])` concatenates the latitude defined in the Latitude column.
- `||","||Char(thisTable:longitude[irow])||` concatenates the longitude defined in the Longitude column.
- The Google API uses `",12z"` to specify a zoom factor.

- The script on the Tip tab defines the tooltip that appears when you hover over the hyperlink. The tip accurately describes what happens after the link is clicked. Consider using this function to set the tooltip to the URL the Click script uses or a description of the platform.

  ```
  Function( {thisTable, thisColumn, iRow},
  "Open " ||
     "https://www.google.com/maps/@"||char(thisTable:latitude[irow])||","||char
     (thisTable:longitude[irow])||",12z" || " in your browser.";);
  ```

  - Returns the tool tip string.
- The script on the Color tab defines the color of the hyperlink (in this example, blue).

  ```
  Function( {thisTable, thisColumn, iRow}, 5;);
  ```

  See "JMP Colors" for a diagram of JMP colors.

You might also want to embed links to other files in a data table column. The following example shows how to use Event Handler scripts and set the values to the audio and video locations on your computer.

```
New Table( "Wildlife",
   New Column( "Column 1", Character, "Nominal",
      Set Property( "Event Handler",
         Event Handler(
            Click( JSL Quote(Function( {thisTable, thisColumn, iRow}, Open(
   Char(
thisTable:thisColumn[ iRow ] ) ); );) ),
            Tip( JSL Quote(Function( {thisTable, thisColumn, iRow}, "Open " ||
   Char(
```

```
thisTable:thisColumn[ iRow ] ) || " in a media player."; );) ),
           Color( JSL Quote(Function( {thisTable, thisColumn, iRow}, 5; );) )
        )
     ),
     Set Selected,
     Set Values( {
"C:\Users\Public\Videos\Sample Videos\Wildlife.wmv",
"C:\Users\Public\Music\Sample Music\Kalimba.mp3"
     } )
  )
)
```

The SAS Country Office and Photo columns in SAS Offices.jmp also contain examples of Event Handler scripts.

Columns

This section covers actions that you can perform on columns in a data table, such as creating, grouping, setting and getting attributes and properties, and so on.

- "Send Messages to Data Column Objects"
- "Create Columns"
- "Create a New Column by Text Matching"
- "Add Several Columns at Once"
- "Group Columns"
- "Use Set Selected to Select Columns"
- "Rearrange and Move Columns"
- "Recode Columns"
- "Add a Column Switcher"
- "Compress Selected Columns"
- "Create Custom Functions, Transforms, and Formats"
- "Custom Transform Example"
- "Delete Columns"
- "Get Column Names"
- "Column Attributes"
- "Column Properties"

Note: JMP can display math symbols and Greek letters (controlled in Preferences in the Font category). This means that if you save a column (such as T square limits), the column name could either be "T Square Limits" (no special characters) or "T^2 Limits" (with special characters). Any reference using the column name must match the name exactly, or it fails.

Send Messages to Data Column Objects

Just as you send data table messages to a data table reference, you can send column messages to a reference to a data column object. The `Column` function returns a data column reference. Its argument is either a name in quotation marks, something that evaluates to a name in quotation marks, or a number.

```
Column( "age" );       // a reference to the age column
col = Column( 2 );     // assign a reference to the second column
```

Note: You can also use the syntax `"Profits ($M)"n` to evaluate to a name that contains special characters.

The *Scripting Guide* uses *col* to represent data column references. To see the messages that you can send to data column objects, see the Scripting Index (**Help > Scripting Index > Objects > Data Tables > Column Scripting**). Alternatively, you can use the `Show Properties()` command:

```
Show Properties( col );
```

Note: With column references, you must include a subscript to refer to the individual data values. Without a subscript, the reference is to the column object as a whole.

Once you have stored a data column reference in a global variable, to modify columns, you send messages to the data column reference.

Sending messages to columns is comparable to sending messages to data tables. Either state the object, a double-angle operator <<, and then the message with its arguments in parentheses, or use the `Send()` function with the object and then the message. In some cases the messages themselves need no arguments, so the trailing parentheses are optional.

```
col << message( arg, arg2, ... );
Send( col, message(arg, arg2, ... ) );
```

As with data tables and other types of objects, you can stack or list messages:

```
col << message << message2 << ...
col << {message, message2, ...};
```

Tip: To delete a column, you must send the message to a data *table* reference, because objects cannot delete themselves, only their containers can delete them.

Access Cell Values through Column References

Always use a subscript on a column reference to access the values in the cells of columns. Without a subscript, the reference is to the column object as a whole.

```
x = col[irow]    // specific row
x = col[]        // current row
col[irow] = 2;   // as an l-value for assignment
dt << Select Where( col[] < 14 );  // in a WHERE clause
```

Access Cell Values that have Value Labels

If the column whose values you want to access has a Value Label property, you might want to access the value label (formatted value) instead of the actual data value. To do this, use the formatted option. Consider the following example:

```
dt = Open( "$SAMPLE_DATA\Big Class.jmp" );
dt << Run Script( "Set Age Value Labels" );
x = Column( dt, "age", formatted )[1];
     Show( x );
y = Column( dt, "age" )[1];
     Show( y );
```

The log shows the formatted value for *x*, which is `Twelve`, and the actual data value for *y*, which is `12`.

Create Columns

To add a new column to a data table, send a `New Column` message to a data table reference or use the `New Column()` function. The first argument, the column's name, is required. Either enclose the name in quotation marks or specify an expression evaluating to the name.

```
dt = New Table( "MyData1.jmp" );
dt << New Column( "wafer" );
```

or

```
dt = New Table( "MyData2.jmp" );
a = "wafer";
dt << New Column( a );
```

or

```
dt = New Table( "MyData3.jmp" );
col = New Column( "X", Formula( Random Uniform() ) );
```

If the table already includes a column by the same name, a sequential numeric value is appended to the new column name (wafer, wafer 2, wafer 3, and so on).

You can specify the following properties when creating a new column:

- Data type (numeric, character, row state, or expression)
- Modeling type (continuous, nominal, ordinal, multiple response, unstructured text, or vector)
- Column width (only for numeric columns)
- Numeric format

Unless otherwise specified, columns are numeric, continuous, and 12 characters wide.

The following example creates a new column whose data type is numeric, modeling type is continuous, width is set to 5, and numeric format is set to Best:

```
dt << New Column( "wafer", Numeric, "Continuous", Format( "Best", 5 ) );
```

The following example creates a character column and automatically assigns the nominal modeling type.

```
dt << New Column( "Last Name", Character );
```

You can also add formulas and other script messages appropriate to the column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Ratio", Numeric, "Continuous", Formula( :height/:weight )
  );
dt << New Column( "myMarkers",
   Row State,
   Set Formula( Marker State( age - 12 ) )
);
```

You can also create new columns that have the same properties as an existing column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Like Name", Like( :name ) );
```

If you plan to work with specific columns later (for example, grouping or changing data types), create a column reference:

```
myCol = dt << New Column( "Birth Date" );
```

Fill a Column with Data

To fill the column with data, use `Values` or its equivalent, `Set Values`. Include the value for each cell in a list.

The following example adds a new column called Last Name to a new data table. Three values are added: Smith, Jones, and Anderson.

```
dt = New Table( "My Data");
dt << New Column( "Last Name", Character, Values( {"Smith", "Jones",
   "Anderson"} ) );
```

The column can also be filled with numeric values. The following example adds a new column called Row Number to a new data table. The function N Row returns the number of rows in the table, and numeric values populate all of the rows in the column, beginning with 1.

```
dt = New Table( "My Data");
dt << New Column( "Last Name", Character, Values( {"Smith", "Jones",
   "Anderson"} ) );
dt << New Column( "Row Number",
   Numeric,
   Values( 1 :: N Row() ),
   Format( "Best", 5 )
);
```

To add a column of random numbers, insert a formula along with the random function.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Random", Numeric, Formula( Random Uniform() ) );
```

Another option is to add a constant numeric value to each cell. In the following example, the number 5 is added to each cell.

```
dt << New Column( "Number" );
:Number << Set Each Value( 5 );
```

New Column() can also be used as a built-in function, in other words, without the << (Send) command applied to the data table reference. When used in this way, the column is added to the current data table.

```
dt << New Column( "Address" ); // command is sent to the data table reference
New Column( "Address" ); // column is added to the current data table
```

Create a New Column by Text Matching

The New Column by Text Matching option enables you to use regular expressions (which identify patterns in text) to match text in a column and then create a new column from the matched text. This feature is an alternative to using the Text Explorer platform to write regular expressions and then save the results to a column.

The following example shows how to select *cat*, *cats*, *dog*, and *dogs* in a column and put the values in new columns.

```
dt = Open( "$SAMPLE_DATA/Pet Survey.jmp" );
dt << New Column by Text Matching(
```

```
   Column( :Survey Response ),
   Set Regex(
      Custom(
         Title( "Cat" ),
         /* finds "cat", optionally followed by "s",
            with either a space or period afterward. */
         Regex( "(cat(?:s?))([\s\.])" ),
         Result( "\[\1]\" ), // results in the value matched by (cat...)
      ),
   ),
   Output Column Name( "Instances of Cats" )
);
dt << New Column by Text Matching(
   Column( :Survey Response ),
   Set Regex(
      Custom(
         Title( "Dog" ),
         Regex( "(dog(?:s?))([\s\.])" ),
         Result( "\[\1]\" ),
      ),
   ),
   Output Column Name( "Instances of Dogs" )
);
```

For more information about using regular expressions in JSL, see "Regular Expressions". To learn more about regular expressions, visit Regular-Expressions.info.

Add Several Columns at Once

The `Add Multiple Columns` message creates several columns at once. Arguments include the column name prefix, the number of columns, where to insert them (`Before First`, `After Last`, or `After(col)`), and the data type (`Numeric`, `Row State`, or `Character(width)`). An additional argument, field width, is optional for a numeric column.

```
// add two row state columns named "beginning <n>" before the first column
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Add Multiple Columns( "beginning", 2, Before First, Row State );

// add three numeric columns named "middle <n>" after the height column
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Add Multiple Columns( "middle", 3, After( :height ), Numeric );

// add four character columns named "end <n>" after the last column
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Add Multiple Columns( "end", 4, After Last, Character( 4 ) );
```

The column name is a prefix. If multiple columns with the same name are added, a sequential number is appended to each column name (beginning 1, beginning 2, and so on).

Group Columns

You can group columns by sending the data table the `Group Columns` message, which takes a list of columns to group as an argument. For example, the following code opens the Big Class.jmp sample data table and groups the age and sex columns.

```
dt = Open( "$SAMPLE_DATA\Big Class.jmp" );
dt << Group Columns( {:age, :sex} );
   "age etc."
```

You can also send a column name followed by the number of columns to include in the group. The group includes the first column named and the *n*-1 columns that follow it. This line is equivalent to the line above, which groups age and sex:

```
dt << Group Columns( :age, 2 );
```

The group name is based on the first column specified in the argument. So in the preceding example, the group is automatically named age etc. To customize the name, include the group name as the first argument.

```
dt << Group Columns( "My group", :age, 2 );
```

To ungroup grouped columns, use the `Ungroup Columns` message, which takes a list of columns in a group as an argument. For example, the following line ungroups the two columns grouped in the previous example.

```
dt << Ungroup Columns( {:age, :sex} );
```

Note the following about grouping and ungrouping columns:

- Both messages take a single list as an argument. The list must be enclosed in braces.
- You cannot create more than one group in a single message (for example, by giving the `Group Columns` message two lists of columns). Instead, you must send the data table two separate `Group Columns` messages.
- The `Ungroup Columns` message takes a list of columns to ungroup, not the name of a group of columns. You can remove a partial list of columns from a group. For example, this line creates a group of four columns:

  ```
  dt << Group Columns( {:age, :sex, :height, :weight} );
  ```

  And this line removes two of the columns, while leaving the other two in the group:

  ```
  dt << Ungroup Columns( {:age, :sex} );
  ```

  Notice that the column group's name does not change automatically. You have to change the name.

Get Column Groups or Names

To get column groups or names, use `Get Column Group` or `Get Column Groups Names`. The first part of this example creates two sets of grouped columns as shown in the previous section. `Get Column Group()` returns the column names in the specific group. `Get Column Groups Names` returns the names of the column groups.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Group Columns( {:age, :sex} );
   "age etc."
dt << Group Columns( {:weight, :height} );
   "height etc."
dt << Get Column Group( "age etc." );
   {:age, :sex}
dt << Get Column Groups Names();
   {"age etc.", "height etc."}
```

Select a Column Group

To select or deselect a column group, use the following messages:

```
dt << Select Column Group( "name" ); // select the columns in the group
dt << Deselect Column Group( "name" ); // deselect the columns in the group
```

The following example groups columns X and Y and groups columns Ozone through Lead, and then selects those groups in the data table:

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
dt << Group Columns( "xy", {:X, :Y} );
dt << Group Columns(
   "pollutants",
   :Ozone :: :Lead
);
dt << Select Column Group( "xy", "pollutants" );
```

Rename a Column Group

To rename a column group, use the following message:

```
dt << Rename Column Group( "name", "toname" );
```

The following example renames the column group from xy to coordinates:

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
dt << Group Columns( "xy", {:X, :Y} );
dt << Group Columns(
   "pollutants",
   :Ozone :: :Lead
);
Wait( 3 );
```

```
dt << Rename Column Group( "xy", "coordinates" );
```

Move a Column Group

To move a column group, use the following message:

```
dt << Move Column Group (To First | To Last | After(column) | After(group));
```

The following example moves the columns in the pollutants group to the end:

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
dt << Group Columns( "xy", {:X, :Y} );
dt << Group Columns(
   "pollutants",
   :Ozone :: :Lead
);
dt << Move Column Group( To Last, "pollutants" );
```

Use Set Selected to Select Columns

Send the `Set Selected` message to select a column.

```
col << Set Selected( 1 );
```

To select all columns with a specific string in the name, run the following script:

```
dt = Open( "$SAMPLE_DATA/Semiconductor Capability.jmp" );

colList = dt << Get Column Names( String );

For( i = 1, i <= N Cols( dt ), i++,
   If( Contains( colList[i], "NPN" ), /* select only columns with NPN
   in the name */
      Column( colList[i] ) << Set Selected( 1 )
   )
);
```

If you want to select many columns, consider sending the `Select Columns` message to the data table. See the *Scripting Guide*.

Get Selected Columns

To get a list of currently selected columns, use the `Get Selected Columns` message.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
:age << Set Selected( 1 );
:sex << Set Selected( 1 );
dt << Get Selected Columns();
   {:age, :sex}
```

Return the list of selected columns as a string using the `string` argument:

```
dt << Get Selected Columns( string );
   {"age", "sex", "height"}
```

Once you know what columns are selected, you can then write a script that acts upon these columns. Or your script can iteratively select columns and act upon them one at a time.

To actually select the columns before getting the columns, send the `Set Selected` message to a column. See "Column Attributes".

Go To Column

To select and move to a specific column, use the `Go To` message.

```
dt << Go To( column name | column number );
```

For data tables with many columns, you can use this message to scroll the data table all the way to the left, so that the first column comes into view and is selected:

```
dt = Open( "$SAMPLE_DATA/Tiretread.jmp" );
dt << Go To( 1 );
```

Invert Column Selections

To select any column that is currently deselected and deselect any column that is currently selected, use the `Invert Column Selection` message.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt:age << Set Selected( 1 );
dt:height << Set Selected( 1 );
Wait( 1 );
b = dt << Invert Column Selection;
```

If a list of columns is provided, the columns that are not in the list are selected.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
a = {:height, :weight};
b = dt << Invert Column Selection( a );
```

Rearrange and Move Columns

These messages enable you to rearrange columns in a data table:

```
dt << Reorder By Name;           // alphanumeric order
dt << Reverse Order;             // reverse current order
dt << Reorder By Data Type;      // row state, character, and then numeric
dt << Reorder By Modeling Type;  // continuous, ordinal, nominal
dt << Original Order;            // saved order
```

These commands move the currently selected column or columns to the indicated destination point.

```
dt << Move Selected Column( to first );
dt << Move Selected Column( to last );
dt << Move Selected Column( after( "name" ) );
```

Note that `Move Selected Column` is an alias for `Move Selected Columns`.

You can also move columns without first selecting them in the data table by using the following syntax.

```
dt << Move Selected Columns( {"name"}, to first );
dt << Move Selected Columns( {"name"}, to last );
dt << Move Selected Columns( {"name"}, after( "name" ) );
```

You may use a list to specify the column names.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
list = {"name", "sex"};
dt << Move Selected Columns( list, To Last );
```

Recode Columns

Use `Recode Column` to change all of the values in a column at once. Specify the original column, or reference column, a list of transformations, and a target column. The function applies the transformations to the reference column, updating the target column. If the `Update Properties` argument is turned on, the following column properties are updated when a column is recoded: Value Labels, Value Scores, Value Order, Value Colors, Supercategories, Coding, and Missing Value Codes.

```
obj << Recode Column(<column reference>, {<transform>, ...}, <Update
  Properties(0|1)>, Target Column(<column reference> | <column name>))
```

For example, in Consumer Preferences.jmp, change the coding of the Gender column to 0|1 instead of 1|2, but keep the value labels the same.

```
dt = Open( "$SAMPLE_DATA/Consumer Preferences.jmp" );
col = New Column( :Gender );
dt << Recode Column(
   :Gender,
   {if(
      _rcNow == 1, 0,
      1,
   )},
   Update Properties( 1 ),
   Target Column( col )
);
```

You can also make the transformations to the original column. Recode the age column to age groups in Big Class.jmp.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Recode Column(
   :age,
   {if(
      _rcNow >= 17, "Older",
      _rcNow >= 15, "Middle",
      "Younger"
   )},
   Target Column( :age)
);
```

Add a Column Switcher

Use JSL to add a Column Switcher panel to a report window. The Column Switcher lets you quickly analyze different variables without having to re-create your analysis.

```
// add a Column Switcher to the report
obj << Column Switcher(<default_col>(<col1>, <col2>, ...);

// remove the Column Switcher from a report
obj << Remove Column Switcher();
```

For example, add the Column Switcher to a Contingency report for the Car Poll.jmp data:

```
dt = Open( "$SAMPLE_DATA/Car Poll.jmp" );
obj = dt << Contingency(
   Y( :size ),
   X( :marital status )
);
ColumnSwitcherObject = obj << Column Switcher(
   :marital status,
   {:sex, :country, :marital status}
);
// number of pixels for the switcher width
ColumnSwitcherObject << Set Size( 200 );

// number of columns to display in the switcher
ColumnSwitcherObject << Set NLines( 6 );
```

You can also share a Column Switcher between two platform reports:

```
dt = Open( "$SAMPLE_DATA/Nicardipine.jmp" );
New Window( "Column Switcher Example",
   H List Box(
      sexCS = dt << Column Switcher( :Sex, {:Sex, :Causality, :Race} ),
```

```
        nDist = Distribution( Nominal Distribution( Column( :Sex ) ) ),
        bar = Graph Builder(
           Show Control Panel( 0 ),
           Variables( X( :Sex ), Y( :Age ) ),
           Elements( Box Plot( X, Y, Legend( 5 ) ) )
        )
     )
  );

  sexCS << Link Platform( nDist );
  sexCS << Link Platform( bar );
```

Note: If you are publishing reports that share a Column Switcher to JMP Public or JMP Live, add them to a dashboard instead of using JSL. See *Using JMP*.

For more information about the Column Switcher, see *Using JMP*.

Compress Selected Columns

To minimize the size of a large data table, use the `Compress Selected Columns` message. Each column is compressed into the most compact form.

```
dt << Compress Selected Columns( {column name, column name} );
```

For example, compress the age, sex, height, and weight columns in Big Class.jmp:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Compress Selected Columns(
{:age, :sex, :height, :weight}
);
```

For more information about this feature, see *Using JMP*.

Create Custom Functions, Transforms, and Formats

You can create user-defined custom functions and then use them in the Formula Editor's function list, in JSL scripting, as custom formats, and as custom transforms. Note that these custom functions can only be used in the context of the current JMP session. Restarting JMP removes previously defined custom functions. They must be defined and added for each session. An easy way to do this is to define your functions in an add-in or a startup script that runs each time you start JMP.

To define a custom function, use `New Custom Function()`. After a function is defined, you can add or activate it in the current running instance of JMP using `Add Custom Functions()`. Similarly, you can remove or deactivate a function using `Remove Custom Functions()`.

`New Custom Function()` requires three parameters: namespace, functionName, functionDefinition. Additional messages may be passed to define the following:

- Formula Editor values: Formula category, Result type, Parameters
- Custom transform values: Transform category
- Custom format values: Custom format category
- Scripting Index values: Description, Prototype, Scripting Index category, Examples

Tip: If you receive a data table that uses a custom transform, function, or format, you'll see "###" in the column. The data table author must give you a script to add that custom item to your computer. Run the script and then reopen the data table.

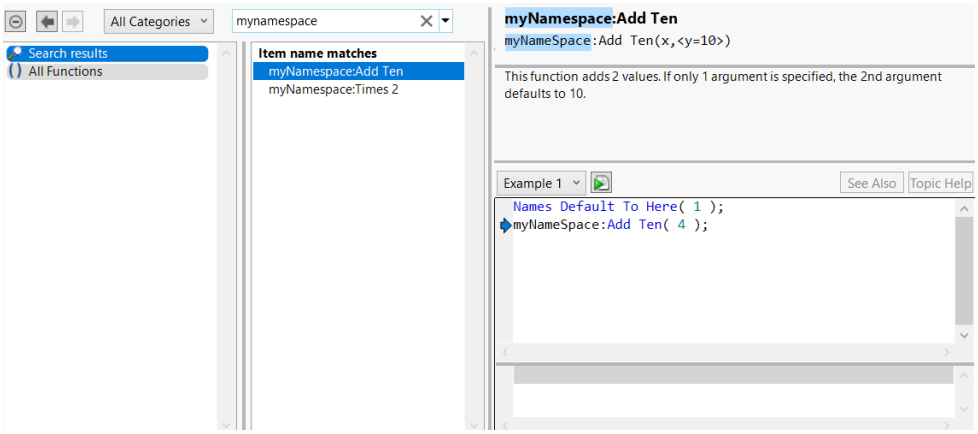
Custom Function Example

Let's create a completely specified function (including Scripting Index information and hint text). We will define the function first, then activate it.

```
funcDef = Function( {x, y = 10}, x + y );
description = "This function adds 2 values. If only 1 argument is specified,
   the 2nd argument defaults to 10.";

// examples can be multiple statements inside expr()
ex2 = Expr(
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   myNameSpace:Add Ten( :age[4], 20 );
);
parmHint1 = "x value";
parmHint2 = "<y=10>";

/* create function using variables
the commands can also be accepted as arguments in New Custom Function */
newAdd = New Custom Function( "myNamespace", "Add Ten", funcDef );
newAdd << Description( description );
newAdd << Prototype( "myNameSpace:Add Ten(x,<y=10>)" );
newAdd << Example( "myNameSpace:Add Ten(4)" );

// expr() Examples need to be passed inside NameExpr()
newAdd << Example( Name Expr( ex2 ) );
newAdd << Parameter( "Number", parmHint1 );
newAdd << Parameter( "Number", parmHint2 );
newAdd << Formula Category( "NumberStuff" );

// add myNamespace:Add Ten function to system
Add Custom Functions( newAdd );
```

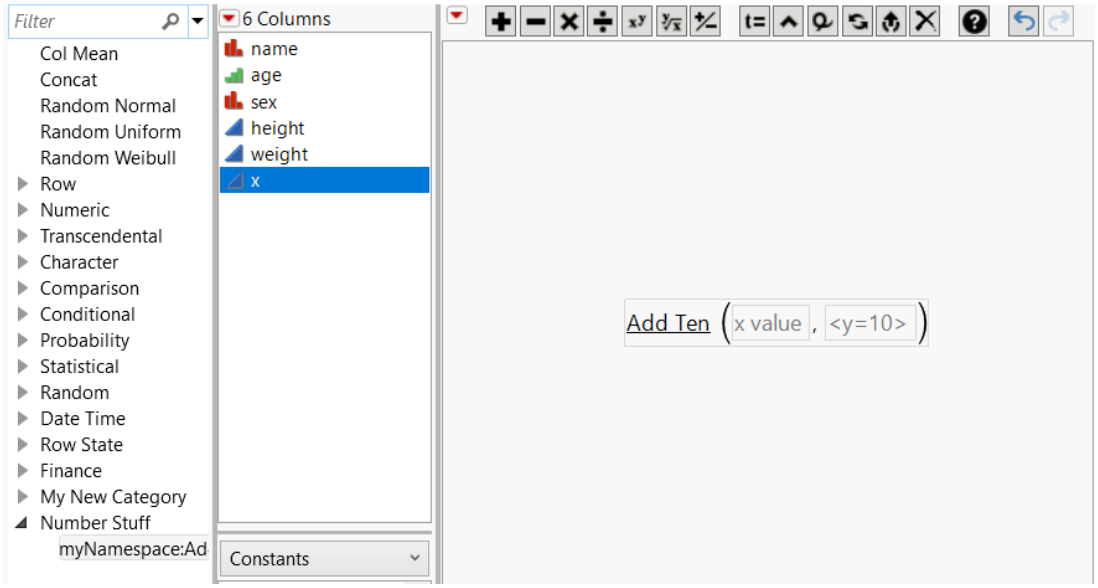
Running the above example will write “Deploying function: myNamespace:Add Ten” in the JMP log. Search for this function in the Scripting Index to see the results of using the Description, Prototype, and Example messages.

Figure 9.13 Custom Function in the Scripting Index



The `Formula Category` message results in a new category in the Formula Editor’s list of formula categories. Custom functions are shown with an underline in the Formula Editor. Also see the hint text from the `Parameter` messages.

Figure 9.14 Custom Function in the Formula Editor



Delete a Custom Function Example

To delete a custom function, use `Remove Custom Functions( {"name"} )`.

```
Remove Custom Functions( {"myNamespace:Add Ten"} );
```

Custom Transform Example

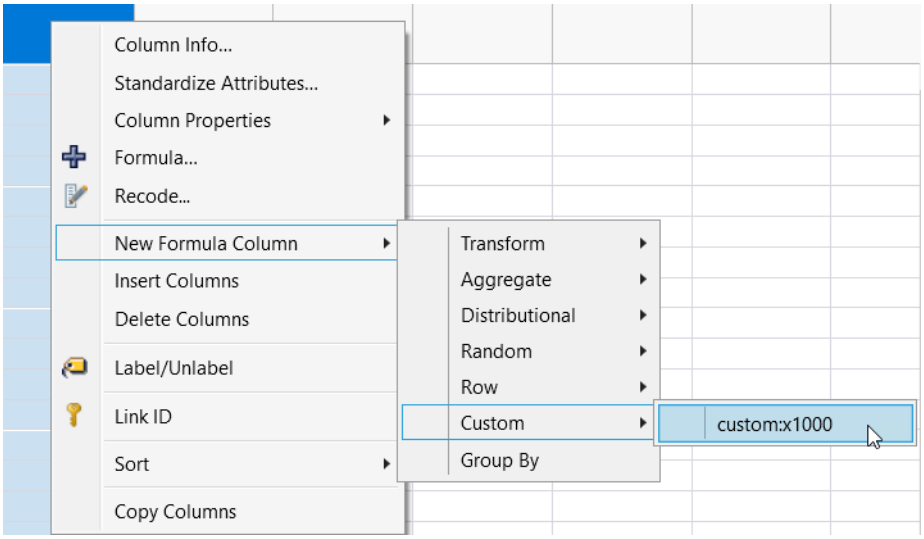
Let's create a simple Custom Transform to multiply a value by 1,000. We embed the function definition inside `Add Custom Functions()`.

```
Add Custom Functions(
  New Custom Function(
    "custom", // namespace in which new function resides
    "x1000", /* function name. The completely scoped name is
      "custom:x1000".*/

    Function( {x}, x * 1000 ),// function definition

    // optional message to enables this as a custom transform
    <<Transform Category( 1 )
    )
);
```

Running the above example writes "Deploying function: custom:x1000" in the JMP log. Search for this function in the Scripting Index to show that `custom:x1000` is an existing function.

When you create a new formula column in a data table, the custom transform appears as shown in Figure 9.15.

Figure 9.15 Custom Transform in a Data Table



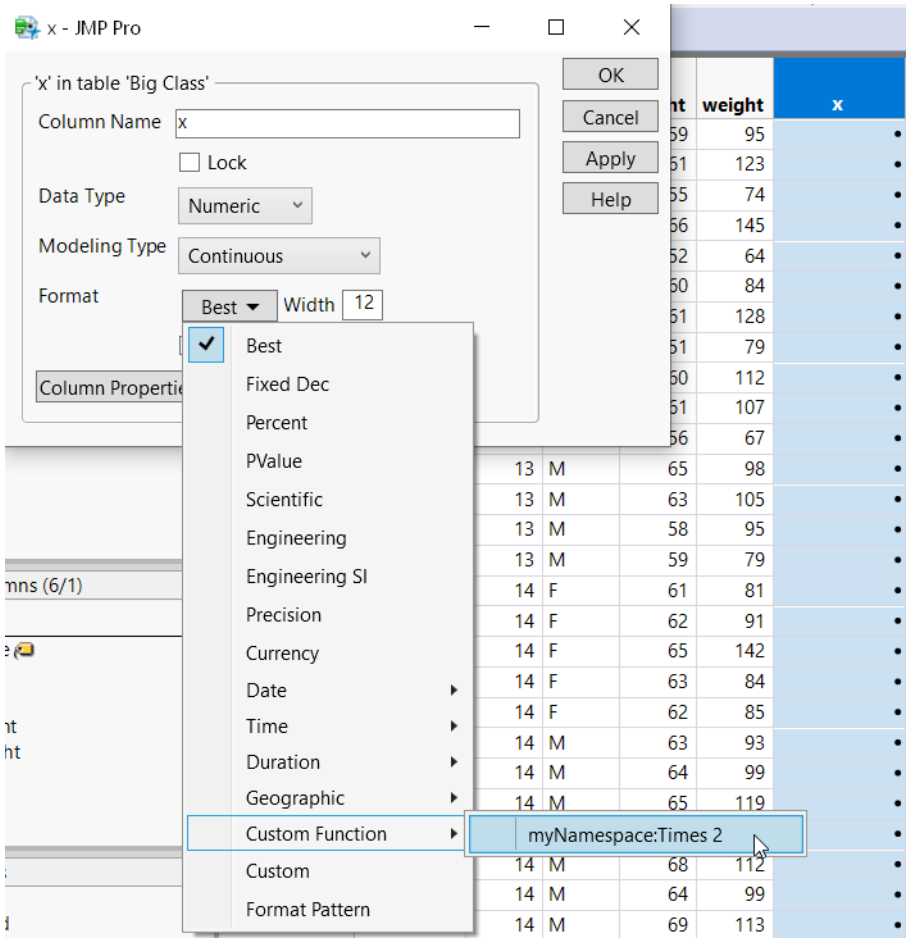
Custom Format Example: Multiplication

The last type of custom function is a custom format. You create custom formats using `New Custom Function()` and by specifying the `Custom Format Category(1)` message. Here's an example that multiplies an input variable by 2 and also displays `(x2)` after the value. Note that, in this example, this function persists as a custom format and in the formula category "My New Category". This allows you to use the format in a column formula or as a custom format.

```
/* add a New Custom Function, specifying it as a Custom Format
show "(x2)" after the value in the data table */
Add Custom Functions(
   New Custom Function(
      "myNamespace",
      "Times 2",
      Function( {inputVar},
         {Default Local},
         Char( inputVar * 2 ) || " (x2)"
      ),
      <<Description( "Multiply input by 2. Display (x2) after the new value."
   ),
      <<Formula Category( "My New Category" ),
      <<Custom Format Category( 1 ) // enable this as a custom format
   )
);
```

By specifying the `Custom Format Category(1)` message, this format now persists under the Custom Function category in the Format menu. Here's what this looks like when applying this new Times 2 custom format in the Column Info window:

Figure 9.16 Custom Format in the Col Info Window



Custom Format Example: Dates

Suppose that you want to display dates in a format that isn't provided in JMP. The following script formats a data to show the month, year, day, and day of week.

```
Add Custom Functions(
  New Custom Function(
    "myNamespace",
    "DateYMD_Day",
    Function( {inputVar},
      {Default Local},
```

```
            Char(
               Match( Month( inputVar ),
                  1, "January ",
                  2, "February ",
                  3, "March ",
                  4, "April ",
                  5, "May ",
                  6, "June ",
                  7, "July ",
                  8, "August ",
                  9, "September ",
                  10, "October ",
                  11, "November ",
                  12, "December "
               )
            ) || Char( Year( inputVar ) ) || ", Day#" || Char( Day( inputVar ) )
      || "-" ||
            Char(
               Match( Day Of Week( inputVar ),
                  1, "Sunday",
                  2, "Monday",
                  3, "Tuesday",
                  4, "Wednesday",
                  5, "Thursday",
                  6, "Friday",
                  7, "Saturday"
               )
            )

         ),
         <<Description(
            "Date format showing Month, Year, Day and Day of Week \!nfor example,
      April 1930, Day#29-Tuesday"
         ),
         <<Custom Format Category( 1 )
      )
   );
```

After you run the script, you can assign the format to a column of date values. In the Cols > Column Info window, the new date format is in the Format > Custom Function menu. You need to widen the data table column to be wide enough to display all of the characters.

Tip: See "Display Numbers as Four Digits" for another example of creating a custom format.

Delete Columns

To delete columns, send a `Delete Columns` message and specify which column or columns to delete. To delete more than one column, list the columns as multiple arguments or as a list.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Delete Columns( "weight" );
dt << Delete Columns( "weight", "age", "sex" );
dt << Delete Columns( {"weight", "age", "sex"} );
```

Without an argument, `Delete Columns` deletes columns that were previously selected. See "Column Attributes".

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column("age") << Set Selected;
dt << Delete Columns();
```

Get Column Names

`Column Name(`*n*`)` returns the name of the *n*th column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column Name( 2 );
   age
```

The returned value is a name value, not a quoted string. What this means is you can use it anywhere you would normally use the actual name in a script. For example, you could subscript it:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column Name( 2 )[1];
   12
```

If you want the name as a text string, quote it with `Char`:

```
Char( Column Name( 2 ) );
   "age"
```

To retrieve a list of the names of all columns in a data table, submit `Get Column Names`.

```
dt << Get Column Names( argument );
```

where the optional *`argument`* controls the output of the `Get Column Names` function, as follows:

- Specify `Numeric`, `Character`, `Row State`, or `Expression` to include only those column data types.
- Specify `Continuous`, `Ordinal`, `Nominal`, `Multiple Response`, `Unstructured Text`, `None`, and `Vector` to include only those modeling types.
- Specify `String` to return a list of strings rather than column names.

For example, if you want to get numeric and continuous columns in the Big Class.jmp sample data table, proceed as follows:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
names = dt << Get Column Names(Numeric, "Continuous")
   {height, weight}
```

Column Attributes

Use a collection of message pairs for data table columns to control all of the various attributes or characteristics of a column, including its name, data, states, and metadata. The messages come in pairs, one to "set" or assign each attribute and one to "get" or query the current setting of each attribute.

For example, you can hide, exclude, label, and turn on or off the scroll lock for a column through scripting. The value is Boolean; enter a one to turn the column attribute on, and a zero to turn it off.

In the following examples, the name column is unhidden, unexcluded, labeled, and locked from horizontal scrolling.

```
Column( "name" ) << Hide( 0 );
Column( "name" ) << Exclude( 0 );
Column( "name" ) << Label( 1 );
Column( "name" ) << Set Scroll Locked( 1 );
```

Note: All the messages to set various arguments (for example, `Set Name`, `Set Values`, `Set Formula`) start with `Set`. The word `Set` is optional for all messages except `Set Name` (recall that `Name` is already used for something else, the command that lets you use unusual characters in a name). Use whichever form you prefer or find easier to remember. The corresponding messages to retrieve the current value of an argument (for example, `Get Formula`) are the same, except that they start with `Get` instead of `Set`, and the word `Get` is *not* optional.

To deselect all selected columns, send a `Clear Column Selection` message to the data table object.

```
dt << Clear Column Selection;
```

Set or Get a Column Name

`Set Name` lets you name or rename a column, and `Get Name` returns the name for a column. The following example changes the name of the second column from age to ratio. It then returns the current column name to the log.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = Column( 2 );
```

```
col << Set Name( "ratio" );
col << Get Name;
  "ratio"
```

Set or Get Column Values

Similarly, `Set Values` sets values for a column. If the variable is character, the argument should be a list. If the variable is numeric, the argument should be a matrix (vector). If the number of values is greater than the current number of rows, the necessary rows are added to the data table. `Get Values` returns the values in list or matrix form. `Get As Matrix` is similar to `Get Values` but returns values in the numeric columns.

```
col << Set Values( myMatrix ); // for a numeric variable
col << Set Values( myList ); // for a character variable
col << Get Values; // returns a matrix, or list if character
col << Get Matrix(<list of column names>|<list of column numbers>|<column
  range>; // returns the specified columns as a matrix
```

The following example returns a list and a matrix of values:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column( "name" ) << Values( {Fred, Wilma, Fred, Ethel, Fred, Lamont} );
myList = :name << Get Values;
  {"Fred", "Wilma", "Fred", "Ethel", "Fred", "Lamont", "JAMES", "ROBERT",
    "BARBARA", ...}

Column( "age" ) << Values( [28, 27, 51, 48, 60, 30] );
myVector = :age << Get Values;
  [28, 27, 51, ... ]
myMatrix = :weight << Get as Matrix;
  [95, 123, 74, ... ]
```

Set or Get Value Labels

Note: For more information about value labels, see *Using JMP*.

Value labels provide a method of displaying a descriptive label for abbreviated data. For example, you might have a column of 0 and 1 values, where 0 represents a male and 1 represents a female. The value label “male” for 0 and “female” for 1 are more readable.

You can specify value labels in any one of the following three ways. Using the Big Class.jmp sample data table, assume that M maps to Male, and F maps to Female.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
:sex << Value Labels( {"F", "M"}, {"Female", "Male"} ); // use two lists
:sex << Value Labels( {"F", "Female", "M", "Male"} ); // use a list of pairs
:sex << Value Labels( {"F" = "Female", "M" = "Male"} ); // use a list of
  assignments
```

You can activate value labels by sending `Use Value Labels` as a message to the column.

```
:sex << Use Value Labels( 1 );
```

To revert back to showing the column's actual values:

```
:sex << Use Value Labels( 0 );
```

The same message can be used for the data table to turn value labels on and off for all columns.

```
dt << Use Value Labels( 1 );
```

Set or Get Data and Modeling Types

You can set or get the data type of a column using JSL. The choices are character, numeric, and row state. The following example adds a new column that has a character data type to the Big Class.jmp sample data table.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "New" );
Column( "New" ) << Data Type( Character );
Column( "New" ) << Get Data Type;
   "Character"
```

To set or get the modeling type of a column:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = New Column( "New" );
col << Modeling Type( "Continuous" );
col << Get Modeling Type;
   "Continuous"
```

You can specify the format of a column when changing its data type:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = New Column( "Date" );
Column( "Date" ) << Data Type( Numeric, Format( "ddMonYYYY" ) );
```

Set or Get Formats

The `Format` message controls numeric and date/time formatting. The first argument is a quoted string from the list of format choices shown in the Column Info window. Subsequent arguments depend on the format choice. You can also set the field width by itself.

```
col << Format( "Best", 5 ); // width is 5
col << Format( "Fixed Dec", 9, 3 ); // width is 9, with 3 decimal places
col << Format( "PValue", 6 );
col << Format( "d/m/y", 10 );
col << Set Field Width( 30 );
```

For date formats, the `Format` message sets how dates appear in a data table column. To set the format that you use for entering data, or for displaying the current cell when you have it selected for entry or editing, use the `Input Format` message.

```
col << Format( "d/m/y", 10 ); // display the date in day-month-year order
col << Input Format( "m/d/y" ); // enter the date in month-day-year order
```

For more information about date/time formatting choices, see "Date-Time Functions and Formats".

Note: Do not confuse the `Format` *message* for columns with the `Format` *function* for converting numeric values to strings according to the format specified (typically used for date/time notation as described in "Date-Time Functions and Formats"). Sending a message to an object has a very different effect from using a function that might happen to have the same name.

To get the current format of a column, submit a `Get Format` message:

```
col << Get Format;
```

Display Numbers as Four Digits

The `Format()` function and the `Format` message have a custom format capability that enables you to provide your own script to do the formatting. The name of the format is "Custom" (rather than "Best", "Fixed", and so on) and the argument is a script that works with the variable named "value" to return a string to represent that value. To make positive or negative integers always have at least four digits by adding zeros, you could do this:

```
New Table( "Untitled",
  New Column( "Column 1",
    Numeric,
    "Continuous",
    Format(
      "Custom", // make a custom format
      Formula( // using a JSL formula
        Local( {sign, length = 4, result}, // local variables
          sign = If( value < 0, -1, 1 ); // handle negative numbers
          value = Char( Abs( value ) );
          If( Length( value ) > length, // handle too big
            value = Repeat( "#", length )
          );
          result = Right( value, length, "0" ); // add zeros
          If( sign < 0,
            result = "-" || result // re-attach sign
          );
          result; // return the modified value
        )
```

```
        )
      ),
      // sample data to test positive, negative, zero
      Set Values( [0, 3, -5, 33, -55, 333, -555, 3333, -5555, 33333, -55555] )
    )
);
```

See "Create Custom Functions, Transforms, and Formats" for more information about creating custom formats.

Specify the Significant Digits in a Numeric Column

Significant digits are the first nonzero digit and all succeeding digits. For example, 3.14159 has six significant digits, and 0.00314 has three significant digits.

Set the number of significant digits with the Precision column format:

```
Format( "Precision", 12, 3 ); // 12 wide with 3 significant digits
```

Set, Get, or Evaluate a Formula

The following examples show how to set, get, and evaluate a formula:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = New Column( "Ratio" ); // create column and stores its reference
col << Set Formula( :height/:weight ); // set the formula
col << Eval Formula; // evaluate the formula
col << Get Formula; // return the expression :height/:weight
```

To use the values from columns in scripts, be sure to add commands to evaluate the formula. Formula evaluation timing can differ between different versions of JMP.

Notes:

- When formulas are added, they are scheduled to be evaluated in a background task. This can be a problem for scripts if they depend on the column having the values while the script is running.
- To force a single column to evaluate, you can send an `Eval Formula` command to the column. You can do this inside the command to create the column, right after the Formula clause:

  ```
  dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
  dt << New Column( "Ratio",
     Numeric,
     Formula( :height / :weight ),
     Eval Formula
  );
  ```

  where `Formula()` is an alias for `Set Formula()`.

However, it is best to wait until you are finished adding a set of formulas, and then use the command `Run Formulas` to evaluate all of the formulas in their proper order, as follows:

```
dt << Run Formulas;
```

- The `Run Formulas` command is preferable to the `Eval Formula` command, because while it is evaluating the formulas, `Eval Formula` does not suppress the background task from evaluating them again. The formula dependency system background task takes great care to evaluate the formulas in the right order, and `RunFormulas` simply calls this task until all the formulas are finished evaluating.
- If you use random numbers and use the `Random Reset(`*`seed`*`)` feature to make a reproducible sequence, then you have another reason to use `Run Formulas`, in order to avoid a second evaluation in the background.
- If formula suppression is enabled with `Supress Eval`, turn it off to enable `Eval Formula` to run.
- The `Add Custom Function()` function enables you to create your own functions and display them in the Formula Editor's Functions list. See "Create Custom Functions, Transforms, and Formats".

Set and Get Range and List Checks

You can manipulate list and range check properties using JSL. The following examples use the Big Class.jmp sample data table.

Set and clear the list check property in the sex column:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column( "sex" ) << List Check( {"M", "F"} ); // set the property
Column( "sex" ) << List Check(); // clear the property
```

Range checks require the specification of a range using the syntax in Table 9.3.

Table 9.3 Range Check Syntax

To specify this range	Use this function
$a \leq x \leq b$	`LELE(a, b)`
$a \leq x < b$	`LELT(a,b)`
$a < x \leq b$	`LTLE(a,b)`
$a < x < b$	`LTLT(a,b)`

The following example specifies that the values in the age column must be greater than zero and less than 120:

```
Column( "age" ) << Range Check( LTLT( 0, 120 ) );
```

All of the functions can be preceded by `Not` and one of them can be missing. The following example specifies that the values in the age column should be greater than or equal to 12:

```
Column( "age" ) << Range Check( not( LT( 12 ) ) );
```

To clear a range check state, submit an empty `Range Check()`:

```
Column( "age" ) << Range Check();
```

To retrieve the list or range check assigned to a column, send a `Get List Check` or `Get Range Check` message to the column:

```
Column( "sex" ) << Get List Check;
Column( "age" ) << Get Range Check;
```

Here is an example of `Get Range Check` for the age column in Big Class.jmp:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column( "age" ) << Range Check( LTLT( 0, 120 ) );
Column( "age" ) << Get Range Check;
   Range Check( LTLT( 0, 120 ) )
```

Note that you can also use `Set Property`, `Get Property`, and `Delete Property` to set, retrieve, and remove list checks and range checks. See "Column Properties".

Note: Operations sent through JSL that involve range check columns show any warnings in the log rather than in interactive windows.

Get a Column Script

`Get Script` returns a script to create the column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
New Column( "Ratio", Set Formula( :height/ :weight) );
Column( "Ratio" ) << Get Script;
   New Column( "Ratio",
      Numeric,
      "Continuous",
      Format( "Best", 10 ),
      Formula( :height / :weight )
   );
```

Preselect Roles

To preselect a role on a column, use the `Preselect Role` message. Choices include No Role, X, Y, Weight, Freq, and Validation. The `Get Role` message returns the current setting.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = New Column( "New" );
col << Preselect Role( X );
col << Get Role;
```

Lock a Column or Cells

To lock or unlock a column, use `Lock` or `Set Lock` with a Boolean argument. `Get Lock` returns the current setting.

```
col << Lock( 1 ); // lock
col << Set Lock( 0 ); // unlock
col << Get Lock; // show current state
```

Column Properties

Tip: JSL choices for properties are the same as those in the **Column Properties** menu in the Column Info window. The arguments for each property correspond to the settings in the Column Info window. An easy way to learn the syntax is to establish the property that you want in the Column Info window first, and then use `Get Property` to view the JSL. See *Using JMP* for more information about each property.

Data columns have numerous optional metadata attributes that can be set, queried, or cleared using the messages `Get Property`, `Set Property`, and `Delete Property`.

```
col << Set Property( "propertyName", {argument list} );
col << Get Property( "propertyName" );
col << Delete Property( "propertyName" );
```

The name of the property in question is always the first argument for `Set Property`, and what is expected for subsequent arguments depends on which property you set:

- `Get Property` and `Delete Property` always take a single argument, which is the name of the property.
- `Get Property` returns the property's settings. `Delete Property` completely removes the property from the column.
- To get a list of all column property names for a column, specify the column and then use `Get Properties List`.

If you want to set several properties, you need to send several separate `Set Property` messages. You can stack several messages in a single JSL statement if you want.

```
col << Set Property( "Axis",{Min(50), Max(180)} )<< Set Property( "Notes", "to
  get proportions" );
```

To get a property's value, send a `Get Property` message whose argument is the name of the property that you want:

```
Column("ratio") << Get Property( "axis" ); // return axis settings
```

To set columns as label columns:

```
dt << Set Label Columns( col1, col2, col3 );
```

To clear all label columns:

```
dt << Set Label Columns();
```

The same syntax works for `Set Scroll Lock Columns`, and `Scroll Lock`.

Set Column Properties with Variables

When writing JSL to script a column property, it is important to understand that the column is a repository for all column properties. The column stores the column property with the values supplied without evaluation or validation of the supplied arguments. In order to add column properties that contains variables using JSL code, the script must convert any variables to their actual values.

One way to do this is to use expressions. In the following example, the `Eval Expr()` function evaluates and replaces any variable wrapped in the `Expr()` function with its value. The outer `Eval()` function evaluates the entire statement after the replacement has occurred.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
// assign desired spec limits
lLimit = 55;
uLimit = 70;
tLimit = 62.5;

Eval(
   Eval Expr(

      // store the limits in the Spec Limits column property
      :height << Set Property("Spec Limits",
         {LSL( Expr( lLimit ) ), USL( Expr( uLimit ) ), Target( Expr( tLimit )
   ),
         Show Limits( 0 )}
      )
   )
);
```

Set Value Order Column Properties

The Value Order column property specifies the order in which you want the data to appear in reports. Data can be sorted by numerical order, custom order, or by the order in which the data appear in the data table.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );

// numerical ordering sorts character column values by splitting
the data into string and numeric segments. The following example turns off
numerical order, which you might specify if the column contains character
data. Hex data ("A1FE2", "A13E4", "BEEEE") sort differently with numerical
ordering on or off).
:age << Set Property( "Value Order", {Numerical Order( 0 )} );

// specifies a custom order for the age column
:age << Set Property( "Value Order", {Custom Order( {15, 17, 12, 13, 14, 16}
  )} );

// sorts "ELIZABETH" first, the remaining items are sorted in alphabetical
  order
:name << Set Property( "Value Order", {Custom Order( {"ELIZABETH"} )} )

// specifies that if all values have a predefined sort order
// (for example, the days of the week), the remaining items are
// sorted on that basis. This option is available only for character data.
:age << Set Property( "Value Order", {Common Order( 1 )} )

// sorts the column values by their occurrence in the data table,
// rather than sorting by value
:name << Set Property( "Value Order", {Row Order Levels( 1 )} );
```

When you set the column property, you are setting the entire property. Suppose that you specify the row order levels. If the data happen to match a Common Order (for example, the days of the week), then Common Order takes precedence. In that case, you need to also include `Common Order(0)` in the property; you cannot send separate `Set Property` messages.

Examples of Setting and Getting Popular Column Properties

Table 9.4 shows examples for setting and getting popular column properties. See *Using JMP* for more information about each property.

Table 9.4 Common Properties for Data Table Columns

Property	Description	Example Using Arguments

Notes	Stores notes about a column. Is a quoted text string.	`col<<Set Property( "Notes", "Extracted from Fisher iris data" );`
		
		`col<<Get Property( "Notes" );`

List Check
Range Check
	Prescribes the possible values that can be entered in a column.	`col<<Set Property( List Check,{"F","M"} );`
		
		`col<<Set Property( "Range Check",LTLT(0, 120) );`
		
		`col<<Get Property( "List Check" );`
		
		`col<<Delete Property( "Range Check" );`

Missing Value Codes	Specifies column values that should be treated as missing.	`col<<Set Property( "Missing Value Codes", {0, 1} );`

Value Labels	Specifies labels to be displayed in place of the values.	`col<<Value Labels( {0 = "Male", 1 = "Female"} );`
		
		Use Boolean values to turn value labels on or off:
		
		`col<<Use Value Labels( 1 );.`

Value Order	Specifies the order in which you want the data to appear in reports.	`col<<Set Property( "Value Order", {"Spring", "Summer", "Fall", "Winter"} );`

Table 9.4 Common Properties for Data Table Columns *(Continued)*

Property	Description	Example Using Arguments

`Value Colors and Color Gradient`	Specifies colors for either categorical or continuous data, respectively.	Specify the color by assigning a JMP number to each value:
		`col<<Set Property( "Value Colors", {"Female" = 3, "Male" = 5} );`
		Specify a color theme:
		`col<<Set Property( "Value Colors", Color Theme( "White to Blue" ) );`
		If you omit the color theme, JMP uses the color theme preferences.
		Specify the gradient and the range and midpoint for Value Gradient:
		`col<<Set Property( "Color Gradient", {"White to Blue", Range( {18, 60, 25} )} );`

`Axis`	Most platforms use this (if it exists) when constructing axes. Mostly Boolean.	`col<<Set Property( "Axis",{Min(50), Max(180), Inc(0), Minor Ticks(10), Show Major Ticks(1), Show Minor Ticks(1), Show Major Grid(0), Show Labels(1), Scale(Linear)} );`
`Units`	Provided for custom uses. Specify the units of measure.	`col<<Set Property( "units", "grams" );` `col<<Get Property( "units" );`
`Coding`	Used for DOE and fitting. List with low and high values.	`col<<Set Property( "Coding", {59,172} );` `col<<Get Property( "Coding" );`
`Mixture`	Used for DOE, fitting, and profiling. Specifies the Mixture column properties in a list.	`col<<Set Property( "Mixture", {0.2, 0.8, 1, L PseudoComponent Coding} );` `col<<Get Property( "Mixture" );`
`Row Order Levels`	Specify to sort levels by their occurrence in the data instead of by value.	`col<<Set Property( "Row Order Levels", 1 );`
`Spec Limits`	Used for capability analysis and variability charts.	`col<<Set Property( "Spec Limits", {LSL(-1), USL(1), Target(0)} );` `col<<Get Property( "Spec Limits" );`

Table 9.4 Common Properties for Data Table Columns *(Continued)*

Property	Description	Example Using Arguments

`Control Limits`	Used for control charts.	`col<<Set Property( "Control Limits", {XBar(Avg(44), LCL(29), UCL(69))} );`
		
		`col<<Get Property( "Control Limits" );`

`Detection Limits`	Set in DOE and used in generalized regression.	`col<<Set Property( "Detection Limits", {10, 50} );`

`Response Limits`	Set in DOE and used in desirability profiling.	`col<<Set Property( "Response Limits", {Goal("Match Target"), Lower(1,1), Middle(2,2), Upper(3,3)} );`
		
		`col<<Get Property("Response Limits");`
		
		Choices for `Goal` are `Maximize`, `Match Target`, `Minimize`, `None`. Other arguments take numeric value and desirability arguments.

`Design Role`	Used for DOE. Specify a single role.	`col<<Set Property( "Design Role", "Covariate" );`
		
		`col<<Get Property( "Design Role" );`
		
		Choices for role are `Continuous`, `Discrete Numeric`, `Categorical`, `Blocking`, `Covariate`, `Mixture`, `Constant`, `Uncontrolled`, `Random Block`, `Signal`, and `Noise`.

`Factor Changes`	Sets the difficulty of changing a factor (Easy, Hard, or Very Hard).	`col<<Set Property( "Factor Changes", Hard) ;`
		
		`col<<Get Property( "Factor Changes" );`

`Sigma`	Used for control charts. Specify known sigma value.	`col<<Set Property( "Sigma",1.332 );`
		
		`col<<Get Property( "Sigma" );`
		
		Each type of chart uses a different sigma calculation.

`Distribution`	Set the distribution type to fit to the column.	`col<<Set Property( "Distribution", Distribution( GLog ) );`
		
		`col<<Get Property( "Distribution" );`

Table 9.4 Common Properties for Data Table Columns *(Continued)*

Property	Description	Example Using Arguments

`Time Frequency`	Set the type of time frequency.	`col<<Set Property( "Time Frequency", Time Frequency( "Annual" ) );`
		
		`col<<Get Property( "Time Frequency" );`

`Map Role`	Set how the column is used to connect map shape data with name data. Specify the role and other information as necessary.	`col<<Set Property( "Map Role", Map Role( Shape Name Use( "filepath to data table", "column name" ) ) );`
		
		`col<<Get Property( "Map Role" );`

`Supercategories`	Groups specific categories into one category. Supported only in the Categorical platform.	`col<<Set Property( "Supercategories", {Group( "Genders", {"F", "M"} )} );`
		
		`col<<Get Property( "Supercategories" );`

`Multiple Response`	Specifies the character that separates the responses within a cell.	`col<<Set Property( "Multiple Response", Multiple Response( Separator( "." ) ) );`
		
		`col<<Get Property( "Multiple Response" ) );`
		
		Note: We recommend assigning the Multiple Response modeling type if the delimiter is a comma. Then you don't have to specify the delimiter in the column property. See *Using JMP*.

Table 9.4 Common Properties for Data Table Columns *(Continued)*

Property	Description	Example Using Arguments

`Profit Matrix`	Assigns weights that define a decision model.
	
	Specify a list that contains the weights in a matrix and categories in another list. In the matrix, each row contains weights that reflect the profit for predicting the category indicated by the column of the matrix.
	
	Caution: The matrix argument is the transpose of the profit matrix as displayed in the Column Properties window.
		`col<<Set Property( "Profit Matrix", {[5 -1, -1 4, -2 -2], {"F", "M", "Undecided"}} );`
		
		`col<<Get Property( "Profit Matrix" );`

`Expression Role`	Applies to columns that contain expressions. Specifies whether the expression should be interpreted as a Picture, a Matrix, or an Expression.	`col<<Set Property( "Expression Role", Expression Role( "Picture", MaxSize( 640, 480 ), StretchToMaxSize( 1 ), PreserveAspectRatio( 1 ), Frame( 0 ) )`

`[Custom property]`	Provided for custom uses. Corresponds to **Column Properties > Other** in the Column Info window.
	
	The first argument is a name for the custom property, and the second argument is an expression.
		`col<<Set Property( "Date recorded",12Dec1999 );`
		
		`long date( col<<Get Property( "Date recorded") );`

Specify Engineering SI Units

The Engineering SI format displays the value with an appropriate SI suffix. For example, 3,100 is displayed as 3.1k. 0.0055 is displayed as 5.5m.

Magnitude	Suffix	Magnitude	Suffix	Magnitude	Suffix	Magnitude	Suffix
103	k	1015	P	10-3	m	10-15	f
106	M	1018	E	10-6	µ	10-18	a
109	G	1021	Z	10-9	n	10-21	z
1012	T	1024	Y	10-12	p	10-24	y

Rows

This section describes the messages for adding and manipulating the rows in a data table. Row messages are directed to a data table reference, and most act on the currently selected rows. A number of row messages might not be practical in scripting (for example, `Move Rows`).

- "Add Rows"
- "Delete Rows"
- "Select Rows"
- "Find Rows"
- "Move Rows"
- "Assign Colors and Markers to Rows"
- "Color Cells"
- "Hide, Exclude, and Label Rows"
- "Iterate on Rows in a Table"
- "Prevent Editing Rows"
- "Row States"

Add Rows

To add rows, send an `Add Rows` message and specify how many rows. You can also specify after which row to insert the new rows. The arguments can either be numbers or expressions that evaluate to numbers.

```
dt << Add Rows( 3 ); // add 3 rows to bottom of data table
dt << Add Rows( 3, 10 ); /* add 3 rows after the 10th row, moving the 11th and
   lower rows farther down */
```

A variation of `Add Rows` lets you specify an argument yielding a list of assignments. Assignments can be separated with commas or semicolons.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Add Rows(
   {:name = "Peter", :age = 14, :sex = "M", :height = 61, :weight = 124}
);
```

or

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Add Rows(
   {:name = "Peter"; :age = 14; :sex = "M"; :height = 61; :weight = 124}
);
```

You can send several arguments yielding lists, or even a list of lists. The following script creates a data table with `Add Rows` commands of each variety:

```
dt = New Table( "Cities" );
dt << New Column( "xx", Numeric );
dt << New Column( "cc", Character, width( 12 ) );

dt << Add Rows( {xx = 12, cc = "Chicago"} ); // single list
dt << Add Rows( // several lists
   {xx = 13, cc = "New York"}, {xx = 14, cc = "Newark"} );
dt << Add Rows( // list of lists
   {{xx = 15, cc = "San Francisco"}, {xx = Sqrt( 256 ), cc = "Oakland"}}
);
a = {xx = 20, cc = "Miami"};
dt << Add Rows( a ); // evaluate as single list

b={{xx = 17, cc = "San Antonio"},{xx = 18, cc = "Houston"}, {xx = 19, cc =
   "Dallas"}};
dt << Add Rows( b ); // evaluate as list of lists
```

Further information for rows can be specified with messages described under "Row States".

Delete Rows

To delete rows, send a `Delete Rows` message and specify which row or rows to delete. To delete more than one row, give a list or matrix as the *rownum* argument, or combine `Delete Rows` with other commands such as `For`. The *rownum* argument can be a number, list of numbers, range of numbers, matrix, or an expression that yields one of these. Without an argument, `Delete Rows` returns the number of selected rows and deletes those rows. When an argument is not specified or rows are not selected, `Delete Rows` only returns the number of selected rows: 0.

```
dt << Delete Rows( 10 ); // delete row 10
```

```
dt << Delete Rows( {11, 12, 13} ); // delete rows 11-13
myList = {11, 12, 13};
dt << Delete Rows( mylist ); // delete rows 11-13
dt << Delete Rows( 1 :: 20 ); // delete first 20 rows
dt << Delete Rows( [1 2 3] );  // delete first 3 rows
```

For example, the following script opens Big Class.jmp and deletes row 10:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Selected( Row State( 10 ) ) = 1; // select row 10
dt << Delete Rows;  // delete row 10
```

You can list duplicate rows, and you can list rows in any order with no consequence.

Here is a general way to remove the bottom *x* rows of a data table of any size. The following example removes five rows from the bottom of the data table:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
x = 5;
n = N Row( dt );
For( i = n, i > n - x, i--,
   dt << Delete Rows( i )
);
```

`NRow` counts the rows in the table. See "Iterate a Script on Each Row".

Select Rows

`Select All Rows` selects (or highlights) all of the rows in a data table.

```
dt << Select All Rows;
```

If all rows are selected, you can deselect them all by using `Invert Row Selection`. This command reverses the selection state for each row, so that any selected rows are deselected, and any deselected rows are selected.

```
dt << Invert Row Selection;
```

Note: With the exception of `Invert Row Selection`, whose result depends on the current selection, any new selection message starts over with a new selection. If you already have certain rows selected and you then send a new message to select rows, all rows are first deselected.

To select a specific row, use `Go To Row`:

```
dt << Go To Row( 9 );
```

To select specific rows in a data table based on their row number, use the `Select Rows` command. The argument to the command is a list of row numbers. For example, to select rows 1, 3, 5, and 7 of a data table:

```
dt << Select Rows( {1, 3, 5, 7} );
```

To select a range of rows, specify one of the following messages:

```
dt << Select Rows( Index( 7, 10 ) );
dt << Select Where( Any( Row() == Index( 7, 10 ) ) );
```

Both of these examples select rows seven through 10 in the current data table.

Select Where

To select rows according to data values, use `Select Where`, specifying a logical test inside the parentheses.

Tip: For a description of the functions and operators that you can use within a `Select Where` message, see "Operators".

For example, using the Big Class.jmp sample data table, select the rows where the students' age is greater than 13:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age > 13 );
```

Or, select the rows where the students' ages are less than 14:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
col = Column( dt, 2 );
dt << Select Where( col[] < 14 );
```

The following example selects the rows where the student's ages are less than 15 and the sex is "F":

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age < 15 & :sex == "F" );
```

To select a row without deselecting a previously selected row, combine `<< Select Where` with `<< Select Where` and the `Current Selection("extend")` argument. This is an alternative to using an `OR` statement.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age == 14 );
dt << Select Where( :sex == "F", Current Selection( "extend" ) );
```

To select rows that contain specific strings, follow this example:

```
dt = Open( "$SAMPLE_DATA/Hollywood Movies.jmp" );
my_genres = {"Romance", "Comedy"};
dt << Select Where( Contains( my_genres, :Genre ) );
```

You can also create more complex conditions by using regular expressions.

```
// select rows where :name begins with "j" or "J".
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
/* \0 is a group reference that causes the expression to return the entire
   matched text. !Is Missing is needed because the regex returns the matched
   text if it successfully matched or missing if it failed to match. So !Is
   Missing means the string successfully matched */
dt << Select Where( !Is Missing( Regex( :name, "^j", "\0", IGNORECASE ) ) );

// select rows where :name ends with "N" or "E".
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( !Is Missing( Regex( :name, "(N|E)$", "\0" ) ) );
```

Select Excluded, Hidden, or Labeled Rows

To select rows that are currently excluded, hidden, or labeled:

```
dt << Select Excluded;
dt << Select Hidden;
dt << Select Labeled;
```

To select rows that are *not* excluded, hidden, or labeled, stack a select message and an invert selection message together in the same statement, or send the two messages sequentially:

```
dt << Select Hidden << Invert Row Selection;
dt << Select Hidden;
dt << Invert Row Selection;
```

Select Duplicate Rows

To select duplicate rows in a data table, use the `Select Duplicate Rows` message:

```
dt = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
dt << Select Duplicate Rows(); // selects rows 301 and 8864
```

The second and subsequent duplicate rows are selected. Note that the duplicate values are case sensitive.

Note: By default, if no matching column is specified, all of the columns are matched.

In the preceding example, the duplicate rows might have been data input errors. Exclude the duplicate rows to remove them from the analysis:

```
dt = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
dt << Select Duplicate Rows();
dt << Exclude();
```

To select duplicate values in the rows of selected columns, specify the column names. The following example finds duplicate values in the rows of the Incident Number and Time columns.

```
dt = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
dt << Select Duplicate Rows( Match( Column( "Incident Number" ), :Time ) );
```

To select duplicate rows according to data values, select the duplicate rows, specify the data values, and then restrict the duplicate rows to the current selection.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Duplicate Rows( Match( :age ) );
dt << Select Where( :age > 15, Current Selection( "restrict" ) );
```

Refer to a Specific Cell in the Row

To refer to a specific cell, assign a subscript to the cell's row number. In the following example, the subscript `[1]` is used with the weight column. The formula then calculates the ratio between each height and the first value in the weight column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
New Column( "ratio", Formula( height / weight[1] ) );
```

Select Random Rows

To obtain a random selection, use this syntax:

```
dt << Select Randomly( number );
dt << Select Randomly( Sample Size( number ) );
dt << Select Randomly( probability );
dt << Select Randomly( Sampling Rate( probability ) );
```

These commands use a conditional probability to obtain the exact count requested.

Select Matching Cells

The row menu command `Select Matching Cells` is also implemented in JSL.

```
// select matching cells in the current data table
dt << Select Matching Cells;
// select matching cells in all open data tables
dt << Select All Matching Cells;
```

For more complicated selections, or to store selections permanently as row state data, see "Row States".

Find Rows

`Get Rows` returns a matrix of rows that match the specified condition. The following example select rows in which the age is greater than or equal to 16:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
```

```
dt << Get Rows Where( :age >= 16 );
   [35, 36, 37, 38, 39, 40]
```

`Get Selected Rows` returns a matrix of the currently selected rows. The following example selects rows 1, 3, 5, and 7 and then returns the row numbers in a matrix:

```
dt << Select Rows( {1, 3, 5, 7} );
dt << Get Selected Rows;
   [1, 3, 5, 7]
```

Find Selected Rows

`Next Selected` and `Previous Selected` scroll the data table window up or down so that the next selected row that is not already in view moves into view. The table wraps, so `Next Selected` jumps from the bottom-most selected row to the top-most, and *vice versa* for `Previous Selected`.

```
dt << Next Selected;
dt << Previous Selected;
```

Clear Selected Rows

To cancel a selection, leaving no rows selected, use `Clear Select`:

```
dt << Clear Select;
```

Move Rows

These commands move the currently selected rows to the indicated destination point.

```
dt << Move Rows( At Start );
dt << Move Rows( At End );
dt << Move Rows( After( rowNumber ) );
```

Assign Colors and Markers to Rows

You can use the `Colors` and `Markers` messages to assign (or change) colors and markers used for rows. These settings mostly affect graphs produced from the data table. Both messages expect numeric arguments to choose which color or marker to use. For more information about how numbers correspond to colors and markers, see "Colors and Markers".

```
dt << Colors( 3 );  // set selected rows to red
dt << Markers( 2 ); // pick the X marker for selected rows
```

As with other row messages, you can stack selection and other messages together:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age == 12 ) // select the youngest subjects
   << Colors( 8 ) << Markers( 8 ); // and use purple open circles for them
```

`Color by Column` sets colors according to the values of a column that you specify, and `Marker by Column` works similarly:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Color by Column( :age );
dt << Marker by Column( :age );
```

Here are additional, named arguments:

- `Continuous Scale` (`Color by Column` only) Assigns colors in a chromatic sequential fashion based on the values in the highlighted column.
- `Reverse Scale` Reverses the color scheme in use.
- `Make Window with Legend` Creates a separate window with a legend.
- `Excluded Rows` Applies the row states to excluded rows.
- `Marker Theme` Specifies the marker type.
- `Color Theme` Specifies the color theme.

Color Cells

You can color individual cells in the data grid. For example, this line uses the row state color to color the cells:

```
dt << Color Rows by Row State;
```

You can also specify a color theme for either categorical or continuous columns:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
:height << Set Property(
  "Color Gradient",
  {"White to Blue", Range( 40, 80 )}
);
:height << Color Cell By Value( 1 ); // turn on the cell coloring

:age << Set Property(
  "Value Colors", // assign the value colors
  {12 = Red, 13 = Yellow, 14 = Green, 15 = Blue, 16 = Magenta, 17 =
  Gray}
);

:age << Color Cell By Value( 0 ); // turn off the cell coloring
```

You can also color specific cells. The following example sets rows 1, 5, 8 of the “name” column to red:

```
:name << Color Cells( red, {1, 5, 8} );
```

To remove the color from specific cells, set the color to black. The following example removes the color for row 1 of the “name” column.

```
:name << Color Cells( black, {1} );
```

You can color cells based on specific values. The following example colors the cells in the height column. All cells containing a value greater than 60 are blue. All cells containing a value equal to or less than 60 are purple.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For( i = 1, i <= N Row( dt ), i++,
   If( Column( dt, "height" )[i] > 60,
      Column( dt, "height" ) << Color Cells( 5, {i} ),
      Column( dt, "height" ) << Color Cells( 8, {i} )
   )
);
```

Note: The first argument for `Color Cells` represents the color value. The second argument contains the row numbers.

Hide, Exclude, and Label Rows

Note: For more information about hiding, excluding, and labeling rows using row state functions, see “Row States”.

Use the `Hide`, `Exclude`, and `Label` messages to hide, exclude, and label rows. These messages are toggles, meaning that to turn the messages on you send them once, and to turn them off you send the message a second time. They also accept Boolean arguments to explicitly turn them off and on.

For example, to hide all rows in Big Class where age is greater than 13, you could do the following:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age > 13 );
dt << Hide( 1 );
```

Messages to the same object can be stacked together in a single statement, so you could simplify the preceding script.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age > 13 ) << Hide( 1 );
```

Iterate on Rows in a Table

In addition to built-in programming functions for iterating, JSL provides functions for iterating through data table rows, groups, or conditional selections of rows.

Generally, an expression is executed on the current row of the data table only. Some exceptions are the expressions inside formula columns, `Summarize` and the pre-evaluated statistics functions, and any use of data table columns by analysis platforms.

Set the Current Row

Note: The current row for scripting is *not* related to rows being selected (or highlighted) in the data table or to the current cursor position in the data table window. The current row for scripting is defined to be zero (no row) by default.

You can set the current row for a script using `For Each Row` or `Row() = X`.

```
Row() = 3; ...
For Each Row( ... );
```

`For Each Row` executes the script once for each row of the current data table. Note that `Row()=1` only lasts for the duration of the script, then `Row()` reverts to its default value, which is zero. This means that submitting a script all at once can produce different results than submitting a script a few lines at a time.

Throughout this chapter, examples without an explicit current row should be assumed to take place within a context that establishes a current row. See "What is the Current Row?".

What is the Current Row?

By default, the current row number is 0. The first row in a table is row 1, so row 0 is essentially not a row. In other words, *by default, an operation is done on no rows*. Unless you take action to set a current row or to specify some set of rows, you get missing values due to the lack of data. For example, a column name returns the value of that column on the current row. Scope the column name with the prefix `:` operator to avoid ambiguity (to force the name to be interpreted as a column name).

```
:sex; // returns ""
:age; // returns .
```

Scoping names prevents you from getting a result that might look reasonable for the whole data table but is actually based on only one row. It also protects you from accidentally overwriting data values when making assignments to ambiguous names under most circumstances. You can have even more complete protection by using the prefix or infix `:` operator to refer specifically to a data column and the prefix :: operator to refer specifically to a global script variable. See "Advanced Scoping and Namespaces".

You can use the Row() function to get or set the current row number. Row() is an example of an *L-value* expression in JSL: a function that returns its value unless you place it before an assignment operator (=, +=, and so on.) to set its value.

```
Row(); // returns the number of the current row (0 by default)
x = Row(); // store the current row number in x
Row() = 7; // make the 7th row current
Row() = 7; :age; // make the 7th row current and returns 12
```

Note that the current row setting only lasts for the portion of a script that you select and submit. After the script executes, the current row setting resets to the default (row 0, or no row). Therefore, a script submitted all at once can produce different results from the same script submitted a few lines at a time.

How Many Rows and Columns?

The N Rows() and N Cols() functions return the rows and columns in a data table.

```
N Rows( dt ); // number of rows
N Cols( dt ); // number of columns
```

N Rows and N Cols also count the number of rows in matrices. Note that NRow and NCol are synonyms. See the "Inquiry Functions".

Iterate a Script on Each Row

To iterate a script on each row of the current data table, put For Each Row around the script.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( If( :age > 15, Show( :age ) ) );
```

To specify the open data table, include a data table reference as the first argument.

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt2 = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
For Each Row( dt1, If( :age > 15, Show( :age ) ) );
```

You can use For Each Row to set row states instead of creating a new formula column in the data table. The scripts below are similar, except that the first one creates a row state column, and the For Each Row script simply sets the row state without creating a column.

```
New Column( "My Row State", Row State, Formula( Color State( :age - 9 ) ) );
For Each Row( Color of( Row State() ) = :age - 9 );
```

To iterate a script on each row that meets a specified condition, combine For Each Row and If:

```
For Each Row( Marker of( Row State() ) = If( :sex == "F", 2, 6 ) );
```

You can use Break and Continue to control the execution of a For Each Row loop. See "Break and Continue".

Return Row Values

`Dif()` and `Lag()` are special functions that can be useful for statistical computations, particularly when working with time series or cumulative data.

- `Lag()` returns the value of a column *n* rows before the current row.
- `Dif()` returns the difference between the value in the current row and the value *n* rows previous.

The following lines are equivalent:

```
dt  <<  New Column( "htDelta" );
For Each Row( :htDelta = :height - Lag( :height, 1 ) );
For Each Row( :htDelta = Dif( :height, 1 ) );
```

Add Sequence Data

`Sequence()` corresponds to the Sequence function in the Formula Editor and is used to fill the cells in a data table column. It takes four arguments and the last two are optional:

```
Sequence( from, to, stepsize, repeat );
```

`From` and `to` are not optional. They specify the range of values to place into the cells. If `from = 4` and `to = 8`, the cells are filled with the values 4, 5, 6, 7, 8, 4, ...

`Stepsize` is optional. If you do not specify a stepsize, the default value is 1. Stepsize increments the values in the range. If `stepsize = 2` with the above `from` and `to` values, the cells are filled with the values 4, 6, 8, 4, 6, ...

`Repeat` is optional. If you do not specify a Repeat, the default value is 1. Repeat specifies how many times each value is repeated before incrementing to the next value. If `repeat = 3` with the above `from`, `to`, and `stepsize` values, the cells are filled with the values 4, 4, 4, 6, 6, 6, 8, 8, 8, 4, ... If you specify a `Repeat` value, you must also specify a `Stepsize` value.

The sequence is always repeated until each cell in the column is filled.

```
dt = New Table( "Sequence Example" ); // create a new data table
dt << New Column( "Count to Five" ); // add two columns
dt << New Column( "Count to Seventeen by Fours" );
dt << Add Rows( 50 ); // add fifty rows

/* fill the first column with the data sequence 1, 2, 3, 4, 5, ...
fill the second column with the data sequence 1, 1, 5, 5, 9, 9, 13, 13, 17,
  17, ... */
For Each Row (
  Column( 1 )[ ] = Sequence( 1, 5 );
  Column( 2 )[ ] = Sequence( 1, 17, 4, 2 );
);
```

Because `Sequence()` is a formula function, you can also set a column's formula to use `Sequence()` to fill the column. The following example creates a new column named Formula Sequence and adds a formula to it. The formula is a sequence that fills the column with values between 25 and 29, incremented by 1, and repeated twice (25, 25, 26, 26, 27, 27, 28, 28, 29, 29, 25, ...).

```
dt = New Table( "Formula Sequence Example" );
dt << Add Rows( 20 );
dt << New Column( "Formula Sequence", Formula( Sequence( 25, 29, 1, 2 ) ) );
```

The following are more examples of `Sequence()` results:

- `Sequence(1,5)` produces 1,2,3,4,5,1,2,3,4,5,1, ...
- `Sequence(1,5,1,2)` produces 1,1,2,2,3,3,4,4,5,5,1,1, ...
- `Sequence(10,50,10)` produces 10,20,30,40,50,10, ...
- `10*Sequence(1,5,1)` also produces 10,20,30,40,50,10, ...
- `Sequence(1,6,2)` produces: 1,3,5,1,3,5, ... The limit is never reached exactly.

Note: If you want a matrix of values, then use the `Index` function, not `Sequence`.

Prevent Editing Rows

To prevent adding or deleting rows, use `Set Edit Lock`:

```
dt << Set Edit Lock( "add rows", "delete rows");
```

See "Prevent Changes to a Data Table" for more information about preventing other data table operations.

Row States

There is a special data element type called a row state that stores various attributes in the data table. Row states can indicate the following:

- Whether a row is selected, excluded, hidden, or labeled
- Which marker type, color, shade, and hue to use for graphs

In JSL, you can use row state functions to manipulate row states.

About Row States

Row states change how JMP works with your data. Table 9.5 explains each row state. Remember that you can use several row states at once to get the combination of effects that you want.

Table 9.5 Row States

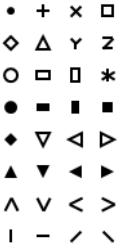
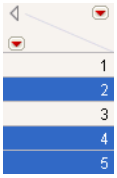
Row states	How they affect results
Excluded	If rows are excluded, JMP omits them from calculations for statistical analyses (text reports and charts). Results are the same as if the data was not entered. However, points are still included in *plots*. (To omit points from plots, use `Hide`. To omit points from all results, use both `Exclude` and `Hide`.)
Hidden	If rows are hidden, JMP does not show them in plots. However, the rows *are* still included in text reports and *charts*. (To omit points from reports and charts, use `Exclude`. To omit points from all results, use both `Exclude` and `Hide`.)
Labeled	If rows are labeled, JMP places row number labels, or the values from a designated Label column, on points in scatterplots.
Color	If rows have colors, JMP uses those colors to distinguish the points in scatterplots.
Marker 	If rows have markers, JMP uses those markers to distinguish the points in scatterplots.

Table 9.5 Row States *(Continued)*

Row states	How they affect results
Selected 	If rows are selected, JMP highlights the corresponding points and bars in plots and charts.

Note: For row state messages, omitting the argument *toggles* the option: if the option is off, the message turns it on, and if the option is on, the message turns it off. You can also use the `"toggle"`, `"switch"`, or `"flip"` argument, as in `r << Exclude( "toggle" )`.

About Row State Functions

When you give a row state function a number as its argument (or an expression that evaluates to a number), the function interprets that number as an index to its possible values.

Table 9.6 shows a comparison chart of the different row state functions, so that you can see which functions convert row states to numbers and numbers to row states. It also includes the numbers that you can use with each function.

Table 9.6 Row State Functions

Numbers	Convert from numbers to row states →	Row states	Convert from row states to numbers →	Numbers
1 or 0	`Excluded State(`*`n`*`)`	Excluded	`Excluded(`*`rowstate`*`)`	1 or 0
1 or 0	`Hidden State(`*`n`*`)`	Hidden	`Hidden(`*`rowstate`*`)`	1 or 0

Table 9.6 Row State Functions *(Continued)*

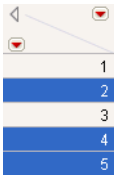
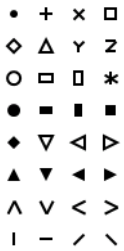
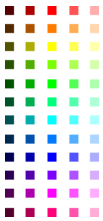
Numbers	Convert from numbers to row states →	Row states	Convert from row states to numbers →	Numbers
1 or 0	`Labeled State(n)`	Labeled	`Labeled(rowstate)`	1 or 0
1 or 0	`Selected State(n)`	Selected 	`Selected(rowstate)`	1 or 0
0 to 31	`Marker State(n)`	Marker 	`Marker Of(rowstate)`	0 to 31
0 to 84 (0–15 basics, 16–31 dark, 32–47 light, 48–63 very dark, 64–79 very light, 80–84 grays)	`Color State(n)`	Color	`Color Of(rowstate)`	0 to 84 (0–15 basics, 16–31 dark, 32–47 light, 48–63 very dark, 64–79 very light, 80–84 grays)
0–11 (rainbow order)	`Hue State(n)`	Hue		

Table 9.6 Row State Functions *(Continued)*

Numbers	Convert from numbers to row states →	Row states	Convert from row states to numbers →	Numbers
–2 to 2 (dark to light)	`Shade State(n)`	Shade 		

Assign Row States

To assign a row state using JSL, use `Select Where` to indicate which rows are affected, and then specify which row state to assign to the rows. In the following example, marker type 5 (a triangle) is assigned to rows in which sex is “F”.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :sex == "F" ) << Markers( 5 );
```

You can also use a formula to assign row states. Here is how you would do it in the data table:

1. Create a row state column.
2. Add a formula to the column that assigns a marker to each row in which sex is “F”.
3. In the Column panel, right-click the star next to the row state column and select **Copy to Row States**.

The JSL equivalent would be something like this:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Row State",
   Row State,
   Set Formula( If( :sex == "F", Marker State( 4 ) ) )
);
Column( "Row State" ) << Copy To Row States();
```

Store Row State Information

A row state column is a dedicated column that *stores* row state information, but does not put the information into effect. You can then use `For Each Row` to put the row state column into effect. For more information about row state columns, see *Using JMP*.

The following example creates a row state column then puts the row states into effect:

1. Submit this line to start a new data table:

```
dt = New Table( "Row State Testing" );
```

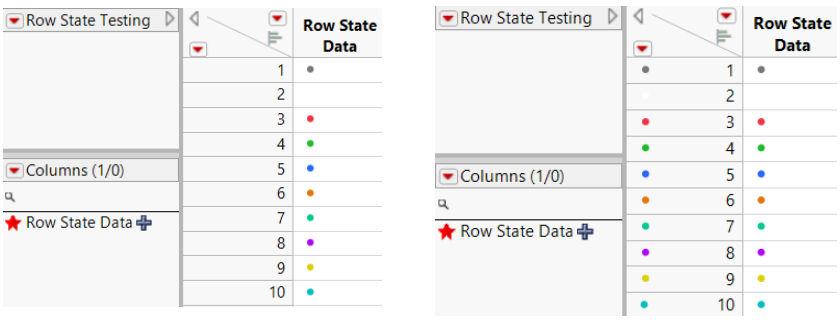
2. Submit these lines to add row states to a column and add ten rows:

```
dt << New Column( "Row State Data", Row State, Set Formula( Color State( Row()
   ) ) );
dt << Add Rows( 10 );
```

3. Submit this line to put the row states into effect:

```
For Each Row( Row State() = :Row State Data );
```

Figure 9.17 Table with No Row States (left) and Table with Row States (right)



This action replaces any row states in effect with the row state combination from the row state column. For more information about how to change selected attributes of a row state without changing or canceling others, see "Set One Characteristic and Cancel Others".

Set or Get Row States

From JSL, you can set or get row states directly using the `Row State` function. You can set or get the state for row *n* with `Row State(n)`. If you do not supply an argument, you set or get the current row. To work with all rows, either use a `For Each Row` loop or work with formula columns.

To set a row state, place the row state expression on the left of an assignment (L-value):

```
Row State( 1 ) = Color State( 3 ); // make row 1 red
Row() = 8; Row State() = Color State( 3 ); // make the 8th row red
For Each Row( Row State() = Color State( 3 ) ); // make each row red
```

To specify the data table, include a data table reference as the first argument:

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt2 = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
Row State( dt1, 1 ) = Color State( 3 ); // make row 1 red;
```

To get a row state, place the row state expression on the right side of an assignment:

```
x = Row State( 5 );  // put the row state of row 5 into x
x = Row State();   // row state of current row
```

Notes on Setting Row States

Be careful whether you set every aspect of `Row State()` or just one aspect of it, such as `Color Of( Row State() )`. To see how this works, first color and mark all the rows:

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row(
  Row State() = Combine States( Color State( Row() ), Marker State( Row() ) )
  );
```

And now observe the difference between setting one attribute of a row state:

```
Color Of( Row State( 1 ) ) = 3; // make row 1 red without changing marker
```

And setting every aspect of a row state to a single state:

```
Row State( 1 ) = Color State( 5 ); // make row 1 blue and remove its marker
```

To copy all the current row states into a row state column:

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row(
       dt1,
       Row State() = Combine States(
              Color State( Row() ),
              Marker State( Row() )
       )
);
dt1 << New Column( "rscol", "Row State", Set Formula( Row State() ) );
```

To copy several but not all of the current row states into a row state column, use a script like the following (commenting out or omitting any states that you do not want):

```
New Column( "rscol2",
  Set Formula(
     Combine States(
        Color State( Color Of() ),
        Excluded State( Excluded() ),
        Hidden State( Hidden() ),
        Labeled State( Labeled() ),
        Marker State( Marker Of() ),
```

```
            Selected State( Selected() )
        )
    )
);
```

To set a component of a row state:

```
Color Of( Row State( i ) ) = 3; // change color to red for row i
Selected( Row State( i ) ) = 1; // select the ith row state and set to 1
```

You can see from the example above that some of the functions convert numbers into states, and others convert states into numbers. Here are some helpful hints for remembering which are which:

Number-to-state functions have the word "State" Functions that take number arguments and either return states or accept state assignments all have the word "State" in their names: `Row State`, `As Row State`, `Color State`, `Combine States`, `Excluded State`, `Hidden State`, `Hue State`, `Labeled State`, `Marker State`, `Selected State`, `Shade State`.

State-to-number functions are one word or have the word "Of" Functions that take row state arguments (and assume that the argument `Row State()` if none is given) and functions that return or are set to numbers are either one word, or their second word is "Of": `Color Of`, `Excluded`, `Hidden`, `Labeled`, `Marker Of`, `Selected`.

Table 9.6 is a helpful comparison chart for these functions.

The following lines are equivalent to their interactive commands:

```
Copy From Row States
Add From Row States
Copy To Row States
Add To Row States
```

Examples of Getting Row States

For example, create the following table containing row states:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Row State Data", Row State, Set Formula( Color State( Row()
  ) ) );
dt << Add Rows( 10 );
For Each Row( Row State() = :Row State Data );
```

To get row states, submit the following lines:

```
Row State( 1 ); // returns row state for row 1
Row() = 8; Row State();  // returns row state for current (8th) row
For Each Row( Print( Row State()) ); // returns the row states for each row
```

To get a row state and store it in a global, place `Row State()` on the right side of an assignment. Add the following lines to the preceding script:

```
::x = Row State( 1 ); // put the row state of row 1 in x, a global
Row() = 8; ::x = Row State();  // put the row state of the 8th row in x
Show( x ); // returns x = Color State( 8 )
dt << New Column( "rscol", Row State) ;
// put row states in rscol (a row state column)
For Each Row( :rscol = Row State() );
```

To get a component of a row state, place the row state expression on the right side of an assignment and also use one of the L-value operators:

```
x = Selected( Row State() ); // selection index of current row selected
```

Get and Set Multiple Characteristics at Once

You can get or set many characteristics at once by combining state settings inside `Combine States()`. You can also get or set each characteristic one at a time, the ultimate row state being the accumulation of characteristics. The following example sets green Y markers for males, but hides them in plots for now. It also sets red X markers for females but shows them in plots.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row(
  If( sex == "M",
     /* then */ Row State() = Combine States(
        Color State( 4 ), Marker State( 6 ), Hidden State( 1 ) ),
     /* else */ Row State() = Combine States(
        Color State( 3 ), Marker State( 2 ), Hidden State( 0 ) ) ) );
```

Get the row state for one row, such as the 6th:

```
Row State( 6 );
  Combine States(Hidden State(1), Color State(4), Marker State(6))
```

Notice that JMP returns a `Combine State()` combination. This is because a row state datum is not just the state of one characteristic, such as color, but the cumulative state of *all* the characteristics that have been set: exclusion, hiding, labeling, selection, markers, colors, hues, and shades. A list of such characteristics is called a *row state combination*.

Just as there can be many row state characteristics in effect, a row state *column* can have multiple characteristic row states as its values.

Set One Characteristic and Cancel Others

In addition to the overall `Row State()` function for getting or setting all the characteristics of a row state, there are separate functions to get or set one characteristic at a time preemptively. That is, to give a row one characteristic, canceling any other characteristics that might be in effect. The functions that set one characteristic and cancel others are `Color State`, `Combine States`, `Excluded State`, `Hidden State`, `Hue State`, `Labeled State`, `Marker State`, `Selected State`, `Shade State`.

For example, to make row 4 be hidden only:

```
Row State( 4 ) = Hidden State( 1 );
```

Set or Get One Characteristic at a Time

A row state is not just one characteristic, but many. To work with just one characteristic at a time, use one of the L-value operators with `Row State` on either side of the equal sign. The side of the equal sign depends on whether you want to get or set a characteristic. There is an L-value operator for each of the following characteristics: `Color Of`, `Excluded`, `Hidden`, `Labeled`, `Marker Of`, `Selected`.

This example hides row 4 without affecting any other characteristics:

```
Hidden( Row State( 4 ) ) = 1
```

This example stores the color of row 3 without getting any other characteristics:

```
::color = Color Of( Row State( 3 ) );
```

Identify Row State Changes

The `MakeRowStateHandler` message (sent to a data table object) obtains a callback when the row states change. For example,

```
f = Function( {X}, Show( x ) );
obj = dt << Make Row State Handler( f );
```

Then when you select a group of rows, the row numbers of any row whose row state changed are sent to the log. For example:

```
x:[3, 4, 28, 40, 41]
```

When a group is highlighted, it might call the handler twice, once for rows whose selection is cleared, then again for the new selection.

Exclude, Hide, Label, and Select

This section discusses conditions that have Boolean states (either on or off). These conditions include: excluding, hiding, labeling, and selecting rows.

- `Excluded` gets or sets an excluded index. The index is 1 for true or 0 for false, indicating whether each row is excluded.
- `Hidden` gets or sets a hidden index, which is 1 for hidden or 0 for not hidden.
- `Labeled` gets or sets a labeled index, which is 1 for labeled or 0 for not labeled.
- `Selected` gets or sets a selected index, which is 1 for selected or 0 for not selected.

The following examples illustrate these conditions:

```
Excluded( Row State() ); // returns 1 if current row is excluded, 0 if not
Hidden();                // returns 1 if current row is hidden, 0 if not
Labeled( Row State() );  // returns 1 if current row is labeled, 0 if not
Selected();              // returns 1 if current row is selected, 0 if not

Excluded( Row State() ) = 1; // exclude current row
Hidden() = 0;                // unhide current row
Labeled( Row State() ) = 1;  // label current row
Selected() = 0;              // deselect current row
```

Remember that these functions assume the argument `Row State()` if none is given.

`Excluded State`, `Hidden State`, `Labeled State`, and `Selected State` do the reverse; they get or set a row state condition as true or false according to the argument. Nonzero values set the row state to true, and zero values set it to false. Missing values result in no change of state.

```
Row State() = Excluded State( 1 ); // change current row state to excluded
Row State() = Hidden State( 0 );   // change current row state to not hidden
Row State() = Labeled State( 1 );  // change current row state to labeled
Row State() = Selected State( 0 );
// change current row state to not selected
```

Notice that the first two expressions above replace the row state with just the exclusion or just the unhiding, so that any preexisting row state characteristics are lost. More commonly, you would issue the `state` commands for all of the characteristics that you want inside a `Combine States`. The following example changes the third row to hidden and a green square marker:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Row State( 3 ) = Combine States(
  Color State( 4 ),
  Marker State( 3 ),
  Hidden State( 1 ) );
```

Another common way to use a `-State` command would be in a row state data column whose values could be added to the row state (for cumulative characteristics). The following example excludes each odd numbered row:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "myExcl",
  Row State,
```

```
    Set Formula( Excluded State( Modulo( Row(), 2 ) ) )
);
For Each Row( Row State() = Combine States( :myExcl, Row State() ) );
```

Clear Row States

`Clear Row States()` removes row states from the data table. The following script assigns a row state to each row and then removes the row state.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( Row State() = Color State( Row() ) );
Wait( 3 ); // pause for demonstration purposes
dt << Clear Row States();
```

To clear row states for selected rows, select the rows first and then clear the selected row states.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( Row State() = Color State( Row() ) );
dt << Select Rows( [ 5 ] );
Wait( 3 ); // pause for demonstration purposes
dt << Clear Selected Row States();
```

Colors and Markers

This section discusses conditions that have many choices. These conditions include: coloring, adding markers, different hues, or different shades to rows.

`Color Of()` returns or sets the color index. The color index is a number from the JMP color map that corresponds to the *`row state`*, or a missing value if there is no assigned color.

```
Color Of( Row State() );   // returns color index for current row
Color Of() = 4;           // set current row to Color 4
```

Similarly, `Marker Of` returns or sets the marker index. The marker index is a number from the JMP marker map that corresponds to the active marker, or a missing value if there is no assigned marker.

```
Marker Of();                  // returns marker index for current row
Marker Of( Row State() ) = 4; // set current row to Marker 4
```

Both `Color Of()` and `Marker Of()` accept any row state expression or column or `Row State()` as arguments. They also assume the argument `Row State()` if none is given (some examples are shown with, and some without).

`Color State()` and `Marker State()` are similar to `Color Of()` and `Marker Of()`, except they work in the opposite direction. Where the `-Of` functions turn actual states into indices, the `-State` functions turn indices into states.

```
Row State() = Color State( 4 );  // change current row to green
```

```
Row State() = Marker State( 4 ); // change current row to the diamond marker
```

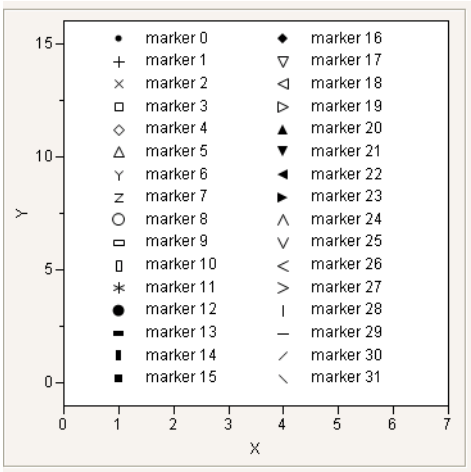
Notice that the last two commands replace the row state with just the color or just the marker, so that any preexisting row state characteristics are lost. More commonly you would issue the `-State` commands for all the characteristics that you want inside a `Combine States()`. The following example puts a green square marker in the third row and hides the row:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Row State( 3 ) = Combine States(
   Color State( 4 ),
   Marker State( 3 ),
   Hidden State( 1 ) );
```

The following script shows the standard JMP markers, which are numbered 0–31. Indices outside the range 0–31 have undefined behavior.

```
New Window( "Markers",
   Graph Box(
      FrameSize( 300, 300 ),
      Y Scale( -1, 16 ),
      X Scale( 0, 7 ),
      For(
         i = 0;
         jj = 15;,
         i < 16;
         jj >= 0;,
         i++;
         jj--;, // 16 rows, 2 columns
         Marker Size( 3 );
         Marker( i, {1, jj + .2} ); // markers 0-15
         Marker( i + 16, {4, jj + .2} ); // markers 16-31
         Text( {1.5, jj}, "marker ", i ); // marker labels 0-15
         Text( {4.5, jj}, "marker ", i + 16 ); // marker labels 16-31
      )
   )
);
```

Figure 9.18 JMP Markers



Tip: For more information about this script, see "Display Trees".

Note the following about colors:

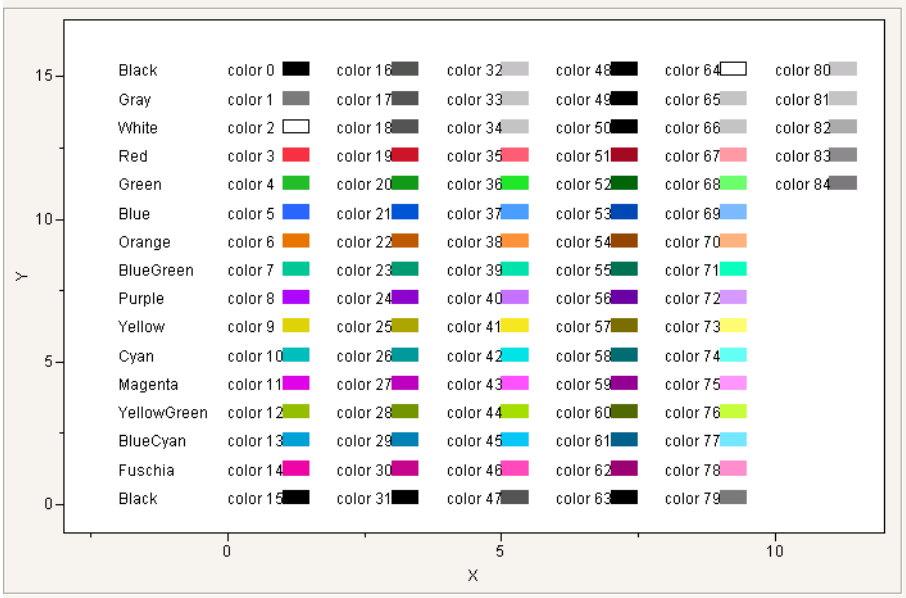
- JMP colors are numbered 0 through 84.
- The first 16 named colors are the basic colors. See the script below.
- Numbers higher than 16 are darker or lighter shades of the basic colors.
- Indices outside the range 0–84 have undefined behavior.
- For more information about using JMP colors, see "Specify Colors".

The following script illustrates the standard JMP colors:

```
Text Color( 0 );
New Window( "Colors",
   Graph Box(
      FrameSize( 640, 400 ),
      Y Scale( -1, 17 ),
      X Scale( -3, 12 ),
      k = 0;
      For( jj = 1, jj <= 12, jj += 2,
         l = 15;
         For( i = 0, i <= 15 & k < 85, i++,
            thiscolor = Color To RGB( k );
            Fill Color( k );
            thisfill = 1;
            If( thiscolor == {1, 1, 1},
               Pen Color( 0 );
               thisfill = 0;
```

```
            ,
               Pen Color( k )
            );
            Rect( jj, l + .5, jj + .5, l, thisfill );
            Text( {jj - 1, l}, "color ", k );
            k++;
            l--;
         );
      );
      jj = -2;
      color = {"Black", "Gray", "White", "Red", "Green", "Blue", "Orange",
"BlueGreen",
      "Purple", "Yellow", "Cyan", "Magenta", "YellowGreen", "BlueCyan",
"Fuschia", "Black"};
      For(
         i = 0;
         l = 15;, i <= 15 & l >= 0,
         i++;
         l--;,
         Text( {jj, l}, color[i + 1] )
      );
) );
```

Figure 9.19 JMP Colors



If you prefer to use RGB values, each color should be a list with percentages for each color in red, green, blue order. For example, the following percentages produce a teal color:

```
Pen Color( {.38,.84,.67} ); // a lovely teal
```

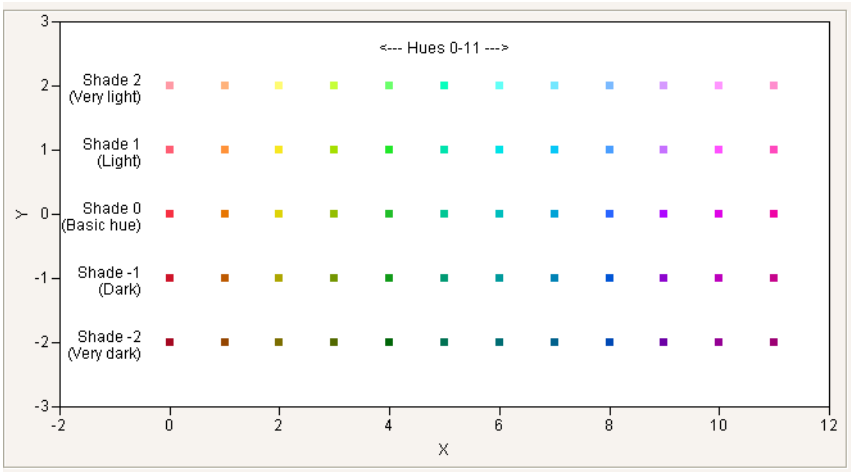
Hue and Shade Example

`Hue State` and `Shade State` together are an alternative to `Color State` for choosing colors. You cannot select black, white, or the shades of gray when you use `Hue State`. For these, you must use `Shade State` alone, or `Color State`.

The following script demonstrates how hue and shade values relate to colors:

```
New Window( "Hues and Shades",
   Graph Box(
      FrameSize( 600, 300 ),
      Y Scale( -3, 3 ),
      X Scale( -2, 12 ),
      k = 0;
      For( h = 0, h < 12, h++,
         For( s = -2, s < 3, s++,
            myMk = Combine States( Hue State( h ), Shade State( s ), Marker
   State( 15 ) );
            Marker Size( 3 );
            Marker( myMk, {h, s} );
         )
      );
      Text( Center Justified, {5, 2.5}, " <--- Hues 0-11 ---> " );
      Text( Right Justified,
         {-.5, -2}, "Shade -2", {-.5, -2.25}, "(Very dark)",
         {-.5, -1}, "Shade -1", {-.5, -1.25}, "(Dark)",
         {-.5, 0}, "Shade 0", {-.5, -.25}, "(Basic hue)",
         {-.5, 1}, "Shade 1", {-.5, .75}, "(Light)",
         {-.5, 2}, "Shade 2", {-.5, 1.75}, "(Very light)"
      );
   )
);
```

Figure 9.20 Hues and Shades



There are no `-Of` functions for `Hue` and `Shade`. `Color Of` returns the equivalent `Color State` index for a color row state that has been set with `Hue State` or `Shade State`. For example, the following example gives rows 4 and 5 the same dark red marker:

```
Row State( 4 ) = Combine States( Hue State( 0 ), Shade State( -1 ), Marker
   State( 12 ) );
Row State( 5 ) = Combine States(
   Color State( Color Of( Row State( 4 ) ) ),
   Marker State( Marker Of( Row State( 4 ) ) )
);
```

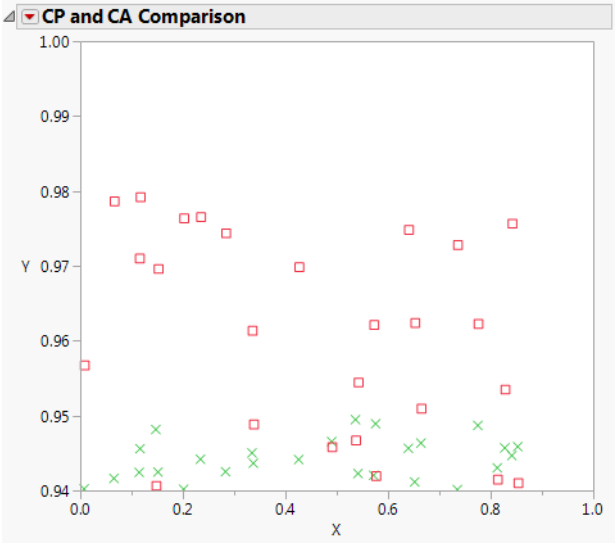
Row State and Matrices Example

In the following example, row state values are prepared ahead and passed to the `Marker` routine, along with matrices of coordinates.

```
dt = New Table( "Artificial CP and CA data",
   Add Rows( 26 ),
   New Column( "cover_cp",
      Numeric,
      "Continuous",
      Formula( Random Uniform() / 100 + 0.94 )
   ),
   New Column( "cover_ca",
      Numeric,
      "Continuous",
      Formula( Random Uniform() * 0.04 + 0.94 )
   ),
   New Column( "p", Numeric, "Continuous", Formula( Random Uniform() ) )
);
```

```
dt << Run Formulas;
greenMark = Combine States( Marker State( 2 ), Color State( 4 ) );
redDiamond = Combine States( Marker State( 3 ), Color State( 3 ) );
New Window( "CP and CA Comparisons",
   Graph Box(
      Title( "CP and CA Comparison" ),
      FrameSize( 400, 350 ),
      X Scale( 0, 1 ),
      Y Scale( 0.94, 1 ),
      For Each Row(
         Marker( greenMark, {p, cover_cp} );
         Marker( redDiamond, {p, cover_ca} );
      )
   )
);
```

Figure 9.21 Example of Row States Values and Matrices



Accessing Data Values

The typical way to work with values in a data table is to follow these steps:

1. Set up the data table whose values you want to access as the current data table. Or, if you already have a data table reference, you can simply use that reference. See "Set the Current Data Table".

2. Specify the row or rows whose values you want to access and specify the column name that contains the values that you want to access. See "Set or Get Values by Column Name".

The following example opens the Big Class.jmp sample data table (making it the current data table), and then specifies row 2 in the weight column. A value of 123 is returned in the log, which is the weight for Louise in row 2.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt:weight[2];
   123
```

Set or Get Values by Column Name

The easiest way to refer to a column is by its name. If you have a global variable and a column with the same name, to prevent ambiguity, scope the column name with the `:` prefix operator.

To set the value of a cell in the current row, provide the column name and the new value.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Row() = 5; // select row 5
dt:age = 19; // set the age in row 5 to 19
dt:name = "Sam"; // set the name in row 5 to Sam
```

To set the value of a cell in row 10, specify the row number in a subscript.

```
dt:age[10] = 20; // set the value of age in row 10 to 20
```

An empty subscript refers to the current row, so :age[ ] is the same as :age.

To get the value of a cell in row 16, specify the column name.

```
Row() = 16; // select row 16
myGlobal = :age; // store the age as a variable
:age; // returns 14, the age in row 16
Show( :age ); // returns age = 14;
```

To get the value of a cell in a specific row, include the column name and row number. Both of the following examples return 13, the value of age in row 12 of Big Class.jmp:

```
:age[12];
myGlobal = :age[12];
```

To get a value in a data column reference, use `Column()` and `As Column()` to get the value in a data column reference. See "Access Cell Values through Column References".

Notes:

- If you do not specify the row number, the current row is selected.
- An empty subscript, such as `:age[ ]`, refers to the current row.

- Be careful that you are subscripting to a table row that exists. The default row number is zero, so statements like `:name` that refer to row zero generate an `Invalid Row Number` error.

Additional Ways to Access Data Values

There are other ways to specify a data table, row, and column. You can specify all three items in one expression by using an infix operator and a subscript:

```
dt:age[2] = 12; // table, column, and row
```

If you want to target multiple rows, you can use subscripts with a list or matrix of row numbers.

```
age[i] = 3;
age[{3, 12, 32}] = 14;
list = age[{3, 12, 32}]; // results in a list
vector = age[1 :: 20]; // results in a matrix
dt[1,1] = dt[2,1]; // in Big Class.jmp, “KATIE” (row 1, column 1) is now
  “LOUISE” (from row 2, column 1)
dt[0,{height}] = dt[0,{weight}]; // everybody’s height is their weight
dt[1,{height,weight}] = dt[2,4::5]; // row 1 height and weight is from Louise
dt[[5 3 1], 0] = .; /* set rows 5, 3, and 1 to missing. Using out-of-order
  subscripts makes more sense if the right hand side is a matrix or a data
  table with subscripts */
```

Note about Changing Values

Whenever you change values in a data table, messages are sent to the displays to keep them up-to-date. However, if you have thousands of changes in a script, this increases the time it takes to complete the updates.

In order to speed up changes, use `Begin Data Update` before the changes to block these update messages. Use `End Data Update` after the changes have been completed to release the messages and update the displays.

```
dt << Begin Data Update;
...<many changes>...
dt << End Data Update;
```

Be sure to always send the `End Data Update` message, otherwise the display is not updated until forced to do so in some other way.

Note: `Begin Data Update` does not affect the data refresh due to some other table manipulations. For example, when you delete or add columns, the data table is updated and then the data update begins.

Add Metadata to a Data Table

Data tables store observation data, or measurements of various variables on a specific set of subjects. However, JMP data tables can also store *metadata*, or data about the data.

- "Data Table Variables"
- "Table Scripts"
- "Formulas"
- "Delete Metadata"

Data Table Variables

Table variables are for storing a single text string value, such as "Notes". To understand how variables work, first get its existing value by sending a `Get Table Variable` message:

```
dt = Open( "$SAMPLE_DATA/Solubility.jmp" );
dt << Get Table Variable( "Notes" );
   "Chemical compounds were measured for solubility in different solvents. This
      table shows the log of the partition coefficient (logP) of 72 organic
      solutes in 6 aqueous/nonpolar systems."
```

Now change the existing value of the string using `Set Table Variable` and then use `Get Table Variable` again to check that the string has been updated:

```
dt << Set Table Variable( "Notes", "Solubility of chemical compounds" );
dt << Get Table Variable( "Notes" );
   "Solubility of chemical compounds"
```

The following example adds two new table variables to a data table:

```
dt = Open( "$DOCUMENTS/Big Class.jmp" );
myvar = "This is my version of the JMP Big Class sample data.";
dt << Set Table Variable( "key1", myvar );
dt << Set Table Variable( "key2", myvar );
```

Notice that setting the value creates a new variable only if one by the given name does not already exist. If you add two table variables with the same name, only one variable is created.

Table Scripts

You can add a new script, run a script, or get a script using JSL.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Script( "Bivariate Example", Bivariate( Y( :weight), X( :height ) )
   );
dt << Get Table Variable( "Bivariate Example" );
```

```
    // returns Bivariate( Y( weight ), X( height ) )
dt << Run Script( "Bivariate Example" )
```

The following example replaces the contents of the table script that was created by the preceding script:

```
dt << Set Property( "Bivariate Example", Bivariate( Y( :weight ), X( :height
  ), Fit Line ) ); // include Fit Line in the Bivariate Example script
```

Suppose you want a text representation of a data table, perhaps to e-mail to a colleague or to use as part of a script. You can obtain a script that reconstructs the information in a data table with `Get Script`. The following example opens Big Class.jmp and prints the data, table variables, and column properties to the log. A portion of the output is shown here:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Get Script;
   New Table( "Big Class",
      Add Rows( 40 ),
      New Script(
         ["en" => "Distribution",...],
         Distribution(
            Continuous Distribution( Column( :weight ) ),
            Nominal Distribution( Column( :age ) )
         ) ),
```

`Delete Scripts` deletes one or more scripts.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Delete Scripts( "Distribution" );
```

In JMP versions prior to 14, use `Delete Property` to delete a script.

Note: The `Names Default to Here` setting is false by default for table scripts. If a table script specifies `Names Default to Here(1)`, this setting lasts only as long as the table script runs. After the script finishes running, the setting is reset to whatever it was before the script ran.

Options for Downloaded Data Tables that Contain Scripts

You are prompted to open or examine downloaded data tables that contain scripts. This help you prevent potentially damaging scripts from running on your computer. In the `Open()` command, you can control this behavior.

To open the data table and enable scripts to run:

```
Open( "$DOWNLOADS/file.jmp", Quarantine Action( "Allow Scripts" ) );
```

To open the data table and block scripts:

```
Open( "$DOWNLOADS/file.jmp", Quarantine Action( "Block Scripts" ) );
```

To display the default prompt window to open or examine the file:

```
Open( "$DOWNLOADS/file.jmp", Quarantine Action( "Show Dialog" ) )
```

To enable a `Try()` expression to capture the error:

```
Try( Open( "$DOWNLOADS/file.jmp", Quarantine Action( "Do Not Open" ) ),
Show( exception_msg ) );
```

If the user selects Examine, the scripts are disabled.

Notes:

- The Block Scripts and Show Dialog options don't work if the domain that you downloaded the data table from is on your Windows Internet Options Trusted sites list.
- After you select Open for a quarantined data table, the data table is trusted until the next time you open JMP.

Prompting to Run a Script upon Opening a Data Table

You can prompt to open a table script named `On Open` or `OnOpen` when the data table opens. The choice is remembered each time you open the data table in the current JMP session.

To create an `On Open` script, perform one of the following actions:

- Create the script using the **Save Script > To Data Table** option, double-click the script name, and change the name to `On Open`.
- Store the script using a `New Script` message.

In this example, you create an OnOpen script in Big Class.jmp.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Script(
   "OnOpen", // create the script
   sortedDt = Data Table( "Big Class" ) << Sort( By( :name ), Output Table
   Name( "Sorted Big Class" ) ) // sort the data and put it in a new data table
);
```

The JMP preference called Evaluate OnOpen Scripts determines when the script runs. By default, the user is prompted to run the script. You can set the preference to always run the `On Open` script or to prevent it from running:

```
Preference( Evaluate OnOpen Scripts( "always" ) );
Preference( Evaluate OnOpen Scripts( "never" ) );
Preference( Evaluate OnOpen Scripts( "prompt" ) ); // default setting
```

`On Open` scripts that execute other programs are never run. As a safety precaution, you might consider suppressing automatic execution when opening data tables that you receive from others.

Notes:

- If the OnOpen script is in a downloaded data table, you must specify `Quarantine Action( "Allow Scripts" )` in the `Open()` expression to run the OnOpen script automatically. See "Options for Downloaded Data Tables that Contain Scripts".
- When you create a new data table in a script and include the `On Open()` function, `On Open()` is called after the data table is created.
- Some operations in an OnOpen script are considered unsafe, such as saving the current data table. Run an external script to save that table instead of opening a table to run the OnOpen script.

Work with Groups of Table Scripts

Suppose that you want to put related table scripts in a group to save room in the list of data table scripts. Use `Group Scripts`, specify the group name, and then specify the script names to put in the group.

```
dt = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
dt << Group Scripts(
   "Maps",
   {"Bubble Plot Street Map", "Graph Builder Street Map: Traffic Incidents",
   "Graph Builder Street Map Zoomed"}
);
dt << Group Scripts(
   "Tabulate",
   {"Tabulate: Category and Description", "Tabulate: Category, Summary,
   Percentage",
   "Tabulate: Category and Resolution", "Tabulate: Resolution and
   Description"}
);
```

Move a Script Group

`Move Script Group` rearranges the table script groups using the `to first`, `to last`, `after(script)`, and `after(group)` arguments:

```
dt << Move Script Group( "Tabulate", after( "Maps" ) );
dt << Move Script Group( "Tabulate", to last);
```

Get Groups and Group Names

To get the list of table scripts in the group, use `Get Script Group`. Then you can perform an operation such as running a script in the group.

```
tb = dt << Get Script Group( "Tabulate" );
dt << Run Script( tb[2] ); // runs the second script in the Tabulate group
```

`Get Script Groups Names` returns the group names in a list:

```
gn = dt << Get Script Groups Names; // returns {"Maps", "Tabulate"}

// move the Maps group to the end of the table scripts
dt << Move Script Group(gn[1], to last);
```

Select a Script Group

`Select Script Group` selects the named table script group, a list of scripts in a group, or all scripts.

```
dt << Select Script Group; // select all scripts groups
dt << Select Script Group( "Tabulate" ); // select Tabulate group
dt << Select Script Group( {"Tabulate", "Maps"} );// select both groups
```

Ungroup Scripts

`Ungroup Scripts` removes specified table scripts or group from the group. You can send the message to the name of the script group or a list of scripts.

```
dt << Ungroup Scripts( "Tabulate" );
dt << Ungroup Scripts(
   {"Tabulate: Category and Description", "Tabulate: Category, Summary,
   Percentage",
   "Tabulate: Category and Resolution", "Tabulate: Resolution and
   Description"}
);
```

Rename a Script Group

You can rename a table script group using `Rename Script Group`, specifying the existing group name, and specifying the new group name.

```
dt << Rename Script Group( "Maps", "Street Maps" );
```

Delete a Script Groups

To delete a table script group, you get it first and then delete it.

```
a = dt << Get Script Group( "Street Maps" );
dt << Delete Scripts( a );
```

Formulas

The message `Suppress Formula Eval` takes a Boolean argument to specify whether formula evaluation should be suppressed or not. You might want to suppress evaluation if you plan to make numerous changes to the data table and do not want to wait for formula updates between steps.

```
dt << Suppress Formula Eval( 1 );
dt << Suppress Formula Eval( 0 ) ;
```

To accomplish the same effect for all data tables, use the `Suppress Formula Eval` command to turn off formulas globally. This is the same as the message above, except that you do not send it to a data table object.

```
Suppress Formula Eval( 1 ); // make formulas static globally
Suppress Formula Eval( 0 ); // make formulas dynamic globally
```

Note that formulas are not evaluated when they are installed in the column. Even when you force evaluation, they end up being evaluated again in the background. This can be a problem for scripts if they depend on the column having the values while the script is running. If you need a mechanism to control evaluation, use the `EvalFormula` command or the `Run Formulas` command.

To force a single column to evaluate, send an `Eval Formula` command to the column. You can even do this inside the command to create the column, after the formula clause:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Ratio",
   Numeric,
   Formula( :height / :weight ),
   Eval Formula
);
```

`dt << Run Formulas` performs all pending formula evaluations, including evaluations that are pending as a result of evaluating other formulas. This function is useful when you have a whole series of columns to run.

Tip: This method is preferred over `EvalFormula`. Although `EvalFormula` evaluates the formulas, it does not suppress the background task from evaluating them again. The background task takes great care to evaluate the formulas in the right order.

If you send the `Run Formulas` command to a data column, the evaluation is done at the time of the command, but it does not suppress the scheduled evaluations that are pending. Therefore, formulas might end up being evaluated twice if you also send the command to the data table and the data column. Being evaluated twice might be desirable for formulas that have random function in them, or it might be undesirable if they depend on randomization seeds being set. If you use random numbers and use the `Random Reset(seed)` feature to make a replicable sequence, then use the `Run Formulas` command, because it avoids a second evaluation.

Note: All platforms send a `Run Formulas` command to the data table to assure that all formulas have finished evaluating before analyses start.

Set Values without a Formula

`col << Set Each Value(expression)` evaluates the expression for each row of the data table and assigns the result to the column. It does not store the expression as a formula.

Delete Metadata

You can delete table variables, table properties (such as a script or a variable), and formulas from the data table, using the following commands:

```
dt << Delete Table Variable( name );
dt << Delete Table Property( name|{property1, property2, ...});
col << Delete Formula;
col << Delete Property( name|{property1, property2, ...} );
col << Delete Column Property( name );
```

Calculations

This section discusses functions for pre-evaluated columnwise and rowwise statistics and shows how JSL expressions work behind the scenes in the JMP formula calculator.

Pre-Evaluated Statistics

The following functions are special, pre-evaluated functions: `Col Maximum`, `Col Mean`, `Col Minimum`, `Col N Missing`, `Col Number`, `Col Quantile`, `CV` (Coefficient of Variation), `Col Standardize`, `Col Std Dev`, `Col Sum`, `Maximum`, `Mean`, `Minimum`, `NMissing`, `Number`, `Std Dev`, and `Sum`.

Note: Statistics are also computed with `Summarize` ("Store Summary Statistics in Global Variables"). Although the named arguments to `Summarize` have the same names as these pre-evaluated statistic functions, they are *not* calling the pre-evaluated statistic functions. The resemblance is purely coincidental.

All the statistics are *pre-evaluated*. That is, JMP calculates them once over the rows or columns specified and thereafter uses the results as constants. Because they are computed once and then used over and over again, they are more efficient to use in calculations than the equivalent formula-calculated results.

When JMP encounters a pre-evaluated function in a script, it immediately evaluates the function and then uses the result as a constant thereafter. Therefore, pre-evaluated functions enable you to use columnwise results for rowwise calculations. For example, if you use `Col Mean` inside a column formula, it first evaluates the mean for the column specified and then uses that result as a constant in evaluating the rest of the formula on each row. A formula might standardize a column using its pre-evaluated mean and standard deviation:

```
( Height-Col Mean( Height ) ) / Col Std Dev( Height )
```

For the Big Class.jmp data, `Col Mean(Height)` is 62.55 and `Col Std Dev(Height)` is 4.24. So for each row, the formula above would subtract 62.55 from that row's height value and then divide by 4.24.

Note: Pre-evaluated functions disregard the excluded row state, meaning that any excluded rows are included in calculations. For summary statistics that obey row exclusion, use the Distribution platform.

Columnwise Functions

The functions whose names begin with "`Col`" all work *columnwise*, or down the values in the specified column, and return a single number. For example, `Col Mean(height)` finds the mean of the values in all the rows of the column height and returns it as a scalar result. Some examples include the following:

```
Average Student Height = Col Mean( height );
Height Sigma = Col Std Dev( height );
```

With the `Col` functions, column properties such as Missing Value Codes assign data values that produce incorrect calculations. Suppose that the Missing Value Codes column property is assigned to the `x1` column to treat "999" as a missing value. Another column includes a formula that calculates the mean. To use the value "999" instead of a missing value to calculate the mean, refer to `Col Stored Value()` in the formula:

```
Mean( Col Stored Value( :x1 ), :x2, :x3 )
```

Rowwise Functions

The functions *without* "Col" listed below work *rowwise* across the values in the variables specified and return a column result. For example, `Mean(height, weight)` finds the mean of the height and weight for the current row of the data table. The rowwise statistics are valid only when used in an appropriate data table row context. The following are some possibilities:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );

// scalar result for row 7 assigned to JSL global variable
Row() = 7;
::scalar = Mean( height, weight );

// formula column created in data table
dt << New Column( "Scaled Ht-Wt Ratio",
   Formula( Mean( height, weight ) / age )
);

vector = J( 1, 40 ); // create a 1x40 matrix to hold results
For Each Row( vector[Row()] = Mean( height, weight ) ); // fill the vector
```

Rowwise functions can also take vector (column matrix) or list arguments:

```
myMu = Mean( [1 2 3 4] );
mySigma = Std Dev( {1, 2, 3} );
```

Calculator Formulas

You can store formulas in columns that are automatically evaluated to create the values in the cells of the column. If you open the formula, you get a calculator interface to edit the formula structurally. However, the formula is implemented with JSL, and you can obtain the text JSL form of any expression in the calculator by double-clicking it. The text can be edited, and when it is de-focused, it is compiled back into the structural form.

Note: There is no difference between a formula column created through the calculator window and one created directly through JSL with commands such as `New Column(..., Formula(...))` or `Col << Formula(...)`.

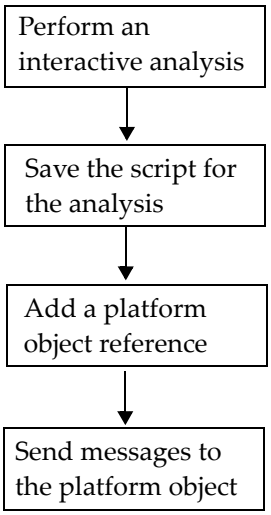
Chapter 10

Scripting Platforms

Create, Repeat, and Modify Analyses

If you run the same analysis frequently, you can script it to automate the process. Anyone can then run the script, ensuring the same results every time. To get started, perform your analysis interactively as you normally would with JMP, and then save a script that re-creates that analysis. You can modify the script to further customize it.

Figure 10.1 Typical Workflow for Scripting Platforms



This chapter is about scripting platforms, not reports. *Platform* object references and *report* object references receive different types of JSL messages. Platforms can run tests, draw plots, and so on. Reports can copy pictures, select display boxes, or close outline nodes. To learn about scripting reports, see the section "Navigate JMP Reports".

Tip: For additional scripting help, see the JMP Scripting Index (**Help > Scripting Index**) and the *JSL Syntax Reference*.

Contents

Example of Scripting a Platform . . . 477
Send Messages to a Platform . . . 480
Conventions for Messages and Arguments . . . 480
Send Multiple Messages . . . 481
Find Messages for Objects . . . 482
Interpret the Show Properties List . . . 483
Send Messages to an Object . . . 484
Specify Which Columns to Analyze . . . 485
Create Column References . . . 485
Specify Multiple Column Names at Once . . . 485
Enable Users to Specify Columns . . . 486
Specify a By Variable . . . 486
Filter by Value using a Where Statement . . . 489
Enable User Input . . . 490
Execute Embedded Red Triangle Options . . . 490
Make Platforms Invisible . . . 491
Specify Report Titles . . . 493
Common Messages for Platform Windows . . . 493
Scripting Considerations by Platform . . . 493
Categorical . . . 493
Control Charts . . . 494
Distribution . . . 498
DOE . . . 498
Fit Model . . . 500
Formula Depot . . . 505
Graph Builder . . . 506
Neural and Neural Net . . . 508
Partial Least Squares and PLS . . . 508
Process Capability . . . 508
Scatterplot 3D . . . 509
Text Explorer . . . 509
Explore Outliers . . . 510
Scripting-Only Messages and Arguments by Platform . . . 510
Bivariate Script-Only Messages . . . 512
Boosted Tree Script-Only Messages . . . 513
Bootstrap Forest Script-Only Messages . . . 514
Bubble Plot Script-Only Messages . . . 515
Choice Script-Only Messages . . . 515
Clustering Script-Only Messages . . . 516

Contingency Script-Only Messages 516
Contour Plot Script-Only Messages 516
Contour Profiler Script-Only Messages 516
Control Chart Script-Only Messages 517
Control Chart Builder Script-Only Messages 517
Cumulative Damage Script-Only Messages 517
Custom Profiler Script-Only Messages 518
Degradation Script-Only Messages 518
Destructive Degradation Script-Only Messages 518
Discriminant Script-Only Messages 518
Distribution Script-Only Messages 519
EMP Measurement Systems Analysis Script-Only Messages 519
Explore Missing Values Script-Only Messages 520
Explore Outliers Script-Only Messages 520
Factor Analysis Script-Only Messages 520
Fit Life by X Script-Only Messages 520
Fit Model Script-Only Messages 521
Fit Parametric Survival Script-Only Messages 523
Functional Data Explorer Script-Only Messages 524
Gaussian Process Script-Only Messages 524
K Means Cluster Script-Only Messages 524
K Nearest Neighbors Script-Only Messages 524
Latent Class Analysis Script-Only Messages 524
Life Distribution Script-Only Messages 525
Make Validation Column Script-Only Messages 526
MaxDiff Script-Only Messages 526
Mixture Profile Script-Only Messages 526
Model Screening Script-Only Messages 527
Multivariate Script-Only Messages 527
Naive Bayes Script-Only Messages 527
Neural Script-Only Messages 527
Nonlinear Script-Only Messages 528
Normal Mixtures Script-Only Messages 529
Oneway Script-Only Messages 529
Parallel Plot Script-Only Messages 529
Pareto Plot Script-Only Messages 530
Partial Least Squares Script-Only Messages 530
Partition Script-Only Messages 530
Principal Components Script-Only Messages 531
Process Capability Script-Only Messages 531

Process Screening Script-Only Messages 531
Profiler Script-Only Messages 532
Reliability Forecast Script-Only Messages 532
Reliability Growth Script-Only Messages 532
Response Screening Script-Only Messages 533
Scatterplot 3D Script-Only Messages 533
Structural Equation Models Script-Only Messages 534
Support Vector Machines Script-Only Messages 535
Surface Plot Script-Only Messages 536
Tabulate Script-Only Messages 539
Ternary Plot Script-Only Messages 539
Text Explorer Script-Only Messages 539
Time Series Script-Only Messages 540
Uplift Script-Only Messages 540
Variability Chart Script-Only Messages 541

Example of Scripting a Platform

If you are not familiar with JSL, it can be helpful to start by interactively performing your analysis in JMP, and then saving the analysis to a script. JMP creates the script for you, and you can then add to it or modify it as desired, by creating a platform object and sending it messages.

Overview

This example takes you through these steps:

1. "Launch the Analysis and Save the Script to a Window"
2. "Add a Platform Reference Object and Send Messages"
3. "Add a Data Table Reference and Save the Script"

Launch the Analysis and Save the Script to a Window

1. Select **Help > Sample Data Library** and open Big Class.jmp.

   This fictional data set contains the names, ages, sex, heights, and weights for 40 students.
2. Select **Analyze** > **Fit Y by X**.
3. Select height and click **Y, Response**.
4. Select age and click **X, Factor**.
5. Click **OK**.

   A Oneway plot of students' height by age appears.
6. Click the Oneway Analysis red triangle and select the following options:
   - **Means/Anova**
   - **Compare Means > Each Pair, Student's t**
7. Click the Oneway Analysis red triangle and select **Save Script > To Script Window**.

Here is the resulting script:

```
Oneway(
   Y( :height ), X( :age ),
   Each Pair( 1 ),
   Means( 1 ),
   Mean Diamonds( 1 )
);
```

Notice the following:

- The script begins with a call to the `Oneway()` function, which returns a Oneway object.

- The parentheses after the `Oneway()` function contain arguments that tell the Oneway function how to make the object.
  - The first two arguments, `Y` and `X`, are required at launch.
  - The next three arguments are optional: `Each Pair`, `Means`, and `Mean Diamonds`.
  - When you selected the Each Pair, Student's t option, the Each Pair feature was turned on.
  - When you selected the Means/Anova option, the Means and Mean Diamonds features were turned on.
- The scripting equivalent of turning an option on or off is the Boolean argument 0 (off) or 1 (on).

Add a Platform Reference Object and Send Messages

1. (Optional) Right-click in the script and select **Show Embedded Log**.

   Output that appears in the log is now easily visible below the script.
2. Choose a JSL variable to remember the Oneway platform object.

   In this example, use the name `oneObj`:

   ```
   oneObj = Oneway(
     Y( :height ),
     X( :age ),
     Each Pair( 1 ),
     Means( 1 ),
     Mean Diamonds( 1 )
   );
   ```

   This gives you a way to address the platform object and send messages to it.
3. Click **Run Script** .

   This creates the Oneway object and sets the variable `oneObj`. Now, send a message to tell the platform to turn on the Unequal Variances report.
4. At the end of the script, add the following line: `oneObj << Unequal Variances(1)`, as shown here:

   ```
   oneObj = Oneway(
     Y( :height ),
     X( :age ),
     Each Pair( 1 ),
     Means( 1 ),
     Mean Diamonds( 1 )
   );
   oneObj << Unequal Variances( 1 );
   ```
5. Highlight the Unequal Variances line of JSL and click **Run Script** .

Tip: You can also press Ctrl+R to run a highlighted line of JSL.

At the bottom of the report window, the Tests That the Variances Are Equal report now appears. Now, suppose that you want to see only that report (and the graph), and you want to close the other reports.

6. Add the following lines to the script:

```
rep = Report( oneObj );
rep["Oneway Anova"] << Close( 1 );
rep["Means Comparisons"] << Close( 1 );
```

The `Report()` function returns the report object for the Oneway platform, and stores a reference to the report in the JSL variable called `rep`. You can send messages or perform actions on the report object, such as closing specific report outlines.

7. Highlight these last 3 lines of the script and click **Run Script** .

In the report window, the Oneway Anova and the Means Comparisons reports are now closed. Only the initial graph and the Tests that the Variances are Equal report (which includes the Welch's Test report) appear.

If you close the Big Class.jmp sample data table at this point and then try to run the script, you are prompted to open a data table. A best practice is to precede the call to the Oneway platform with an `Open()` function to open the associated data table each time the script is run.

Add a Data Table Reference and Save the Script

1. Add the following line as the first line in the script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
```

If the data table is not open when the script is run, this line opens the associated data table.

Your finished script should look like this:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
oneObj = Oneway(
   Y( :height ),
   X( :age ),
   Each Pair( 1 ),
   Means( 1 ),
   Mean Diamonds( 1 )
);
oneObj << Unequal Variances( 1 );
rep = Report( oneObj );
rep["Oneway Anova"] << Close( 1 );
rep["Means Comparisons"] << Close( 1 );
```

Once you complete your script, save it for future access.

2. From the script window, select **File** > **Save As**. Enter a name for your script and save it to any directory.

Send Messages to a Platform

Before you can send messages to a platform, you need to create a platform object that you can send the messages to. Use the Send operator (<<).

The following line keeps a reference to a Oneway platform in the JSL variable called `oneObj`:

```
oneObj = Oneway( Y( height ), X( age ) );
```

Then, add the Send operator to send the platform object a message. In this example, the message is to turn on the Unequal Variances report:

```
oneObj << Unequal Variances( 1 );
```

To see which messages can be sent to a specific object, do one of the following:

- Use the autocomplete shortcut. On a new line, enter the object name, then the Send operator (<<), and press Ctrl+space (Windows) or Ctrl+Option+space (macOS). The following example shows the messages that you can send to the platform object `oneObj` on Windows:

  ```
  oneObj << (Ctrl+space)
  ```

  A list of messages appears. Click one to insert it.
- Use the `Show Properties ()` function. Specify the object in the parenthesis. The following example shows the messages that you can send to the platform object `oneObj`:

  ```
  Show Properties (oneObj)
  ```

  A list of messages appears in the log.

Conventions for Messages and Arguments

- For most messages, omitting the argument for a Boolean command enables the option. For example, all of these messages create a test for unequal variances:

  ```
  oneObj << Unequal Variances;
  oneObj << Unequal Variances( 1 );
  oneObj << Unequal Variances( "true" );
  oneObj << Unequal Variances( "yes" );
  oneObj << Unequal Variances( "present" );
  oneObj << Unequal Variances( "on" );
  ```

The first three approaches are recommended.

Notes:

- The quoted arguments "present", "absent", "on", "off", "switch", and "flip" are supported for backwards compatibility.
- When the name of the JMP option contains several options separated by a comma or slash (such as the **Means/Anova** option or the **T Square, T^2** option) you can use any *one* of the menu names. Or, you can use the full menu name, including the comma or slash, if the string is followed by n. For example, any of these messages create a Means/Anova test:

  ```
  oneObj << Means( 1 )
  oneObj << Anova( 1 )
  oneObj << "Means/Anova"n( 1 )
  ```

- When an option appears under a menu in JMP, the corresponding script message is the option itself without the parent menu. For example, in the Oneway red triangle menu, the Nonparametric menu has multiple options under it, including Wilcoxon Test. Simply use Wilcoxon Test as a function or message in a script:

  ```
  oneObj = Oneway( Y( height ), X( age ), Wilcoxon Test( 1 ));
  oneObj << Wilcoxon Test( 1 );
  ```

- When a menu in JMP contains values rather than options, in the script, specify the parent menu and the value as an argument. For example, in the Oneway red triangle menu, the Set Alpha Level menu has values, such as 0.10, 0.05, 0.01, and Other. To specify 0.01 in a script, add a line like the following:

  ```
  oneObj << SetAlphaLevel(0.01);
  ```

Send Multiple Messages

To send several messages, just add more << operators or more Send arguments:

```
dist << Quantiles( 1 ) << Moments( 1 ) << More Moments( 1 ) << Horizontal
  Layout( 1 );
Send( dist, Quantiles( 1 ), Moments( 1 ), More Moments( 1 ), Horizontal
  Layout(1));
```

Because << is an *eliding operator*, it combines arguments and works differently than if its arguments were grouped. You can stack multiple messages with extra << symbols to perform them all in order, from left to right.

Another way to stack messages is to send a list of messages:

```
dist << {Quantiles( 1 ), Moments( 1 ), More Moments( 1 ), Horizontal
  Layout( 1 )};
```

These approaches work well assuming that no value is returned as a result of one of your messages. However, if your goal is to send a message to the *result* of another message, add grouping parenthesis. The second line shows the wrong way to do it; the third line shows how to do it correctly:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
/* the following line is incorrect, and sends the Fit Mean message to the
   original data table object */
bv = dt << Run Script( "Bivariate" ) << Fit Mean( 1 );
/* the following line is correct, and sends the Fit Mean message to the
   resulting Bivariate object/report */
bv2 = ( dt << Run Script( "Bivariate" ) ) << Fit Mean( 1 );
```

Find Messages for Objects

Some messages apply to all platform objects, and others are specific to certain platforms. Once you have created a platform object, find out what messages you can send to the object, using any of the following approaches:

Use the Scripting Index in JMP

1. Select **Help** > **Scripting Index**.
2. Enter the object type that you are interested in.
3. Click the correct object in the resulting list. Messages for the object appear in the Items list.

Once you have created a platform object, use the Show Properties(obj) function

Replace `obj` with your named platform object. A list and brief description of all the messages the object can receive appears in the Log. To understand more about the Show Properties output, see "Interpret the Show Properties List".

Tip: The `Show Properties()` function also works with data tables and display boxes. See "How Can I See All of the Messages that Can be Sent to a Data Table Object?" and "The << Operator".

- Examine the platform in JMP launch windows and report windows. Most options that you find there have JSL equivalents with the same names and arguments.

Interpret the Show Properties List

Show Properties lists the messages that can be sent to a scriptable object, such as a data table, an analysis platform, or a display, producing a text report in the log. Each type of message is categorized and appears in brackets, and can be any of the following:

Message Type	Description	Example
`[Action]`	Correspond to options in the JMP interface.	`Redo Analysis [Action] (Rerun this same analysis in a new window. The analysis will be different if the data has changed.)`
`[Action Choice]` `[Enum]`	List specific choices for their arguments.	`Fit Polynomial [Action Choice] {2, quadratic, 3, cubic, 4, quartic, 5, 6}` `Draw [Enum] {Filled, Outlined, Filled and Outlined}`
`[Boolean]`	Turn options on or off. Arguments are usually 1 or 0. If a message is specified without an argument, sending the message changes it to the opposite state. `[Default On]` indicates that the option is on by default.	`Show Points [Boolean] [Default On]`

`[Subtable]`	List options within submenus. You use the option itself, not the parent item.	`Script [Subtable]`
		`Redo Analysis [Action] (Rerun this same analysis in a new window. The analysis will be different if the data has changed.)`
		`Save Script to Datatable [Action] (Return to the launcher for this analysis.)`

`[New Entity]`	Open a new window in the interface.	`Prediction Interval [New Entity] (Prediction Interval to contain a single future observation or the mean of m future observations.)`

For example, run the following JSL:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = Bivariate( Y( :height ), X( :age ) );
Show Properties( biv );
```

The object `biv` supports `Subtable`, `Boolean`, `Action`, and `Action Choice` type messages. `Enum` and `New Entity` messages are not supported.

Send Messages to an Object

Suppose that you want to send a message to an object in the output. For example, in Fit Model, you want to send the Response Limits message to the Profiler object, Use the subscript operator (`[ ]`) to refer to the Profiler object.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );

fm = dt << Fit Model(
   Y( :height ),
   Effects( :age, :sex, :weight, :age * :weight, :sex * :weight ),
   Personality( Standard Least Squares ),
   Emphasis( Effect Screening ),
   Run
);

obj = fm << Profiler(
   1,
   Confidence Intervals( 1 ),
   Desirability Functions( 1 ),
   Term Value( age( 12 ), sex( "F" ), weight( 105 ) )
);

Show( Profiler[1] << Get Desirability ); // show the limits before

Profiler[1] << (height << Response Limits(
   {Lower( 50, 0.95 ), Middle( 58, 0.7 ), Upper( 70, 0.066 ),
   Goal( "Minimize" ), Importance( 1 )}
));
Wait( 0.1 ); // for demonstration purposes
Show( Profiler[1] << Get Desirability ); // show the limits after
```

Specify Which Columns to Analyze

If you created your script interactively, you already specified columns in the launch window, so there is no need to specify them again. However, when creating a script from scratch, you normally want to specify the columns to analyze. For example, the following line launches the Distribution platform and specifies the height and weight columns as the Y variables:

```
Distribution( Y( height,weight ));
```

Create Column References

Tip: To see all the messages you can send to a column, go to **Help** > **Scripting Index** > **Data Table** > **Column Scripting**.

Creating a column reference enables you to access and send messages to the column. If you store each column reference in a JSL variable, you can use the variables instead of column names in the Y and X roles and elsewhere in the script.

For example, using Big Class.jmp, the following lines create Ycol as a column reference for weight, and Xcol as a column reference for height. The last line sends a message to Xcol to get the data type.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Ycol = Column( "weight" );
Xcol = Column( "height" );
Xcol << Get Data Type;
```

In the log, “Numeric” is output for the data type of the Xcol (height column).

Referring to a column reference with the : scoping operator (rather than plain text) is convenient in scripts saved to a data table. If the columns are renamed in the data table, the column references in the script are updated.

Specify Multiple Column Names at Once

If you have a lot of column names, instead of listing each one individually, you can use lists as column arguments. The following example gets the Y column names from a list and creates a Oneway analysis on each column in the list.

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
dt << Fit Group( Oneway( Y( {:SO2, :POP} ), X( :State ) ) );
```

If you do not know the column names when you are writing the script, use column indices, as done in the following example.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Distribution( Y( 1::N Col( dt ) ) );
```

You can also use indices when specifying effects as shown here:

```
dt << Fit Model( Y( :y ), Effects( 1 :: 4 ) );
```

Here's another example of specifying multiple column names. `5::31` selects columns 5 through 31.

```
Names Default To Here( 1 );
dt = Open( "$SAMPLE_DATA/Baltic.jmp" );
dt << Partial Least Squares( Y( :ls, :ha, :dt ), X( 5 :: 31 ), Go );
```

Enable Users to Specify Columns

Instead of specifying analysis columns yourself, you can enable a user to specify them, but you can still control what happens after they do so. In this case, use an empty specification. The following example runs the Distribution launch window:

```
Distribution();
```

In this scenario, the Distribution launch window appears and the user can choose the columns to analyze. After the user selects the columns and clicks **OK**, the Distribution report is shown as specified.

Specify a By Variable

In many JMP platforms, you can specify columns as By variables. To do this in a script, include a `By` argument in the platform command, listing each column as an argument.

The following example uses the Big Class.jmp data table, which contains the names, ages, sex, heights, and weights for 40 students. Create a bivariate report of weight by height, using sex as a By variable, and adding a variety of fits.

Create the Bivariate Report Using a By Variable

1. Select **File** > **New** > **Script**.
2. Add the following lines:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( weight ), X( height ), By( sex ));
```

The first line opens the Big Class.jmp sample data table. The second line creates a platform object called biv that runs a bivariate report of weight and height by sex.

3. Click **Run Script** .

   The report window shows two graphs, one where sex is F and one where sex is M.

Send Messages to the Entire Platform or to a Single By Level

1. If you do not have the log showing, turn it on. Right-click in the script and select **Show Embedded Log**.
2. Add this line to the script:

   ```
   Show( biv );
   ```

3. Highlight the `Show( biv )` line and click **Run Script** .

   In the log, rather than returning a single reference to a platform, `Bivariate[]`, the platform object returns a list of references: `biv = {Bivariate[], Bivariate[]}`. The two references correspond to the two levels (F and M) of the By variable, sex.

   You can direct messages to each By level individually or to all By levels.

4. Send a message to all By levels to add a linear regression fit. Add this line to the script:

   ```
   biv << Fit Line;
   ```

5. Highlight the line you just created and click **Run Script** .

   In the report window, a linear regression line is added to both graphs, with the corresponding Linear Fit reports.

6. Send a message to only the F level, adding a cubic polynomial fit. Add this line to the script:

   ```
   biv[1] << Fit Polynomial( 3 );
   ```

   Tip: The number for each By level corresponds to their order in the report window, which is usually alphanumeric. If the By column contains a Value Order column property, that ordering is followed.

7. Highlight the line you just created and click **Run Script** .

   In the report window, a polynomial fit line is added to the F graph, with the corresponding Polynomial Fit reports.

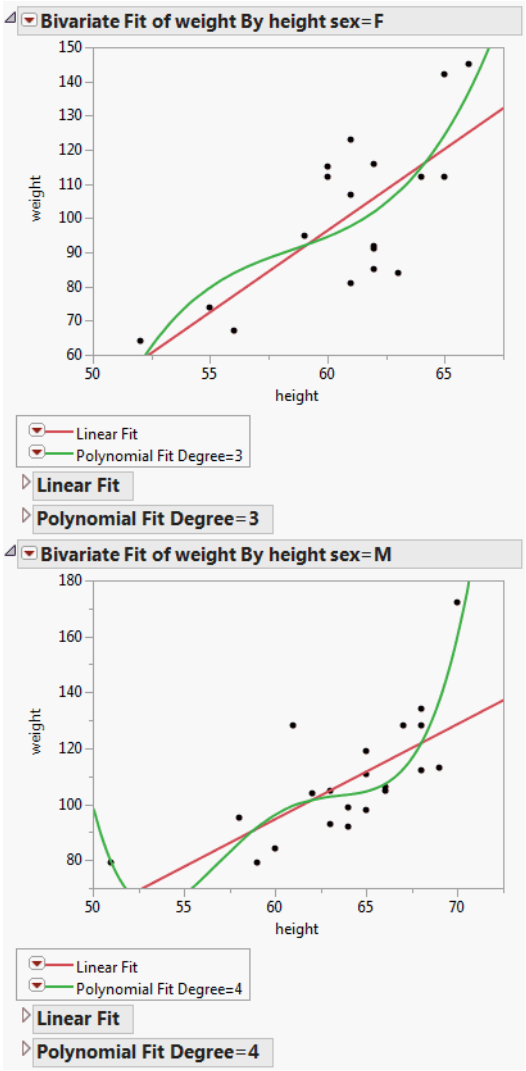
8. Send a message only to the M level, adding a quartic polynomial fit. Add this line to the script:

   ```
   biv[2] << Fit Polynomial( 4 );
   ```

9. Highlight the line you just created and click **Run Script** .

   In the report window, a quartic fit line is added to the M graph, with the corresponding Polynomial Fit reports.

Figure 10.2 By Group Reports



If you specify more than one column in `By` argument, graphs appear for each subgroup of each By variable. In this example, `By( sex, age )` would produce graphs for females age 12, females age 13, and so on, up to age 17. It would also produce graphs for males age 12 up to age 17.

Extract Results from a Report

The following example shows how to launch a platform with `By` groups and extract results (in this case, variances) from each group:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
```

```
// onew is the JSL variable holding a list of platforms
onew = dt << Oneway( x( :age ), y( :height ), by( :sex ), anova );

// r is a list of reports
r = onew << Report;

// nBy is the number of reports generated
nBy = N Items( r );

// vc is an array of as many rows as reports, one column, all set to zero
vc = J( nBy, 1, 0 );

// for each report, do the following:
For( i = 1, i <= nBy, i++,
  vc[i] = r[i] // vc[i] is the variance of the ith report
[Outline Box( "Analysis of Variance" ), // look for this outline

// then look for this column and get the second value
  Column Box( "Sum of Squares" )][2]
);
Show( vc ); // debugging, look in log to see this value

// byValues becomes a list of the values in the sex column
Summarize( byValues = By( :sex ) );

// make a new table with two rows (M.F) and two columns
New Table( "Variances" )
<< New Column( "Sex", // create a new column called Sex
  character,
  width( 8 ),
  values( byValues )

// create a new column called Variance
) << New Column( "Variance", Numeric, "Continuous", Values( vc ) );
```

Filter by Value using a Where Statement

You can use a `Where` statement to filter by a specific value. If you are adding a Where clause to a script that contains column references, you must first resolve the references. The following example limits the analysis to females and fits a line on the Bivariate report.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Ycol = Column( "weight" );
```

```
Xcol = Column( "height" );
biv2 = dt << Bivariate(
Y( Ycol ),
X( Xcol ),
Fit Line( ),
Where( :sex == "F" ) );
```

Enable User Input

When you send an `Action()` function to a platform, it evaluates an expression. For example, use `Action` messages when you want the launch window to appear. A user chooses columns in the launch window and the script continues running after the user runs the analysis.

The following example launches four platforms: Distribution, Bivariate, Oneway, and Contingency. The launch window for each platform appears, and the user chooses the analysis columns and clicks OK. The report window for each analysis appears, and then the next platform launch window comes up, until all platforms have been launched.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Distribution( Action( doit ) );
doit = Expr(
  New Window( "Bivariate", Bivariate( Action( doit2 ) ) )
);
doit2 = Expr(
  New Window( "Oneway", Oneway( Action( doit3 ) ) )
);
doit3 = Expr(
  New Window( "Contingency", Contingency( Action( doit4 ) ) )
);
doit4 = Expr();
```

Execute Embedded Red Triangle Options

In addition to the red triangle menu at the top of a report, some report windows contain embedded red triangle menus. These appear when you select an option that has sub-options, such as profilers. To execute options in embedded red triangle menus, use the `Get Scriptable Object` message. The following example creates a Life Distribution report then turns off the Confidence Intervals option in the Hazard Profiler red triangle menu:

```
dt = Open( "$SAMPLE_DATA/Reliability/Fan.jmp" );

// create a Life Distribution report with a lognormal distribution
obj = dt << Life Distribution(
```

```
        Y( :Time ),
        Censor( :Censor ),
        << Fit Lognormal
);

// get a reference to the Hazard Profiler red triangle menu
haz_prof = ( report( obj )[ "Hazard Profiler" ] << Get Scriptable Object);

// turn off confidence intervals in the Hazard Profiler
haz_prof << Confidence Intervals( 0 );
```

Make Platforms Invisible

In certain circumstances, you might want to make the platform analysis invisible. This means that no platform windows appear, and the platform runs in the background.

If you are doing a simulation or bootstrap analysis in your script, the script might call a platform hundreds or thousands of times. Perhaps you just want to pull one or two results out of each platform report. The following example performs a simulation and then creates a table of results and shows them in a Distribution report:

```
dt = As Table( (0 :: 10)`, << Column Names( {"X"} ) ); // create X variable

// set initial random seed to get reproducible results
Random Reset( 12345 );
dt << New Column( "Y",
   "Continuous",
   "Numeric",
   Set Formula( :X * 2 + 1 + Random Normal() )
);

// Y values have a random component
res = []; // define results matrix (empty to start)
For( i = 1, i <= 100, i++, // loop through simulation runs

   // for each iteration, fit a regression line and save parameter estimates
   bv = dt << Bivariate( Y( :Y ), X( :X ), Fit Line( 1 ), Invisible );
   res |/= (Report( bv )["Parameter Estimates"][

   // concatenate the results to the bottom of the results matrix
   Number Col Box( "Estimate" )] << Get as Matrix)`;

   // close the window (even though it's invisible)
   bv << Close Window;
      dt:Y << Eval Formula; // generate new Y values
```

```
);
// create table of results
dtres = As Table( res, << Column Names( {"Intercept", "Slope"} ) );

// show results in Distribution
dtres << Distribution( Y( :Intercept, :Slope ) );
```

If you want to run an analysis as an intermediate step in your script, and do not need the user to see the analysis, hide it with the `invisible` option. In this case, you might want to show only specific results from the report, which you can output to a journal or to the log.

- The following example runs the Distribution platform as invisible, and outputs only the CDF plot to a new journal:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dist = dt << Distribution( Y( :height ), CDF Plot( 1 ), invisible );
Report( dist )["CDF Plot"] << Journal;
dist << Close Window;
```

- The following example extracts the *F*-Ratio from an invisible Bivariate report, and outputs it to the log:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( x( :height ), y( :weight ), invisible );
biv << Fit Line;
r = biv << Report;
fratio = r[ColumnBox( "F Ratio" )][1];
r << Close Window;
Show( fratio );
  fratio = 1.15609545178219;
```

Tip: Invisible windows use resources that must be manually freed. Be sure to close the invisible window when the script is done with it.

You can also use the `invisible` option on the options in the Tables menu, such as Subset, Sort, Stack, and so on. The following example makes the Subset operation invisible:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Where( :age == 14 );
dt = dt << Subset( invisible );
subDt << Bivariate( x( :height ), y( :weight ), Fit Line );
```

Note that the preceding line could also be handled using a `WHERE` clause:

```
subDt << Bivariate( x( :height ),y( :weight ), Where( :age == 14 ), Fit Line
  );
```

Specify Report Titles

You can specify the title (shown in the title bar of a platform's report) by adding the `title` command to the launch request. The following example replaces the standard bivariate report's title with a user-specified one (My Title).

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Bivariate( x( :height ), y( :weight ), Title( "My Title" ) );
```

Common Messages for Platform Windows

For a list of common messages that you can send to most platforms, see the *JSL Syntax Reference*.

Scripting Considerations by Platform

This section explains specific considerations when scripting the following platforms:

- "Categorical"
- "Control Charts"
- "Distribution"
- "DOE"
- "Fit Model"
- "Formula Depot"
- "Graph Builder"
- "Neural and Neural Net"
- "Partial Least Squares and PLS"
- "Process Capability"
- "Scatterplot 3D"
- "Text Explorer"
- "Explore Outliers"

Categorical

The Free Text option in Categorical displays a Text Explorer report. Scripts that call the Free Text option in Categorical should be updated to use the Text Explorer platform.

Scripting for Supercategories

When ratings are involved in a data set (for example, a five point scale), you might want to know the percent of the responses in the top two or another subset of ratings. You can define a group of ratings in the data using the supercategories scripting option (instead of the column property).

The following example creates a new column called Floss Delimited 2, and adds a supercategory called Sleep Related, which contains two values: Before Sleep and Wake.

```
dt = Open( "$SAMPLE_DATA/Consumer Preferences.jmp" );
dt << New Column( "Floss Delimited 2",
   Character,
   Set Property(
      "Supercategories",
      {Group( "Sleep Related", {"Before Sleep,", "Wake,"} )}
   ), ...);
```

In the data table, under the Columns panel at left, scroll down to the new Floss Delimited 2 column. Click the column property indicator ✻ and select Supercategories. You can see the new Sleep Related supercategory with its two values.

Supercategories can also be specified in a Categorical launch command, where the properties are listed inside parentheses after the column name:

```
dt = Open( "$SAMPLE_DATA/Consumer Preferences.jmp" );
dt << Categorical(
   Supercategories(
      :Floss Delimited(
         {Group( "Sleep Related", {"Before Sleep,", "Wake,"} )}
      )
   ),
   Multiple Response( :Floss Delimited )
);
```

In the Categorical report, notice that the Sleep Related category has 203 cases. If you had run the Categorical platform without creating this supercategory, Before Sleep and Wake would appear as separate categories with 143 and 60 cases, respectively.

For more information about supercategories, see *Consumer Research*.

Control Charts

Using JSL, you can create control charts, customize tests, run alarm scripts, and more.

Reference Lines in Control Chart Builder

If you add a custom reference line to a Control Chart Builder chart through JSL, the reference line label must be something other than “LSL”, “USL”, or “AVG”. These names are keywords used by the Control Chart Builder platform.

The following example creates custom reference lines with unique labels.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
obj = dt << Control Chart Builder(
  Variables( Y( :height ) ),
  SendToReport(
    Dispatch(
      {},
      "height",
      ScaleBox,
      {Add Ref Line( 70, "Solid", "Black", "Upper Limit", 1 ),
      Add Ref Line( 55, "Solid", "Black", "Lower Limit", 1 )}
    )
  )
);
```

Customize Tests in Control Chart Builder

You can design custom tests and select or deselect multiple tests at once using the `Customize Tests` function. You specify the description, desired number, and label. This option is available only for Variables and Attribute chart types.

The following example creates a custom test called Test 1 and turns it on.

```
dt = Open( "$SAMPLE_DATA/Quality Control/Diameter.jmp" );
dt << Control Chart Builder(
  Variables( Subgroup( :DAY ), Y( :DIAMETER ) ),
  Customize Tests( Test 1( 2, "1" ) ), // test number, n, label
  Chart( Position( 1 ), Warnings( Test 1( 1 ) ) ), // turn Test 1 on
  Chart( Position( 2 ) )
);
```

Run Alarm Scripts

An alarm script alerts you when the data fail one or more tests. As an Alarm Script is invoked, the following variables are available, both in the issued script and in subsequent JSL scripts:

`qc_col` is the name of the column

`qc_test` is the test that failed

`qc_sample` is the sample number

`qc_phase` is the label of the phase during which the failure occurred

`qc_firstRow` is the first row in the sample

`qc_lastRow` is the last row in the sample

Example 1: Automatically Writing to a Log

One way to generate automatic alarms is to make a script and store it with the data table as a data table script or column property named QC Alarm Script.

The following example automatically writes a message to the log whenever a test fails:

```
dt = Open( "$SAMPLE_DATA/Quality Control/Coating.jmp" );
obj = dt << Control Chart(
   Sample Size( :Sample ),
   KSigma( 3 ),
   Chart Col( :Weight, XBar, R ),
   Alarm Script(
      Write(
         "Out of Control for test ",
         qc_test,
         " in column ",
         qc_col,
         " in sample ",
         qc_sample
      )
   )
);
obj << Test 1( 1 );
```

Example 2: Running a Chart with Spoken Tests

You can create a control chart with tests that are spoken aloud using the Speak function. Try the following example:

```
dt = Open( "$SAMPLE_DATA/Quality Control/Coating.jmp" );
dt << Control Chart(
   Alarm Script(
      Speak(
         Match( QC_Test,
            1, "One point beyond Zone A",
            QC_Test, 2,
            "Nine points in a row in zone C or beyond", QC_Test,
            5,
               "Two out of three points in a row in Zone A
               or beyond"
         )
      )
   ),
```

```
   Sample Size( :Sample ),
   Ksigma( 3 ),
   Chart Col( :Weight, Xbar( Test 1( 1 ), Test 2( 1 ), Test 5( 1 ) ), R )
);
```

You can have either of these scripts use any of the JSL alert commands, such as `Speak`, `Write`, or `Mail`.

Set Phase Limits

A *phase* is a group of consecutive observations in the data table. For example, phases might correspond to time periods during which a new process is brought into production and then put through successive changes. Phases generate, for each level of the specified Phase variable, a new sigma, set of limits, zones, and resulting tests.

The following example illustrates setting the limits for the different phases of Diameter.jmp:

```
dt = Open( "$SAMPLE_DATA/Quality Control/Diameter.jmp" );
dt << Control Chart(
   Phase( :Phase ),
   Sample Size( :DAY ),
   KSigma( 3 ),
   Chart Col(
      :DIAMETER,
      XBar(
         Phase Level(
            "1",
            Sigma( .29 ),
            Avg( 4.3 ),
            LCL( 3.99 ),
            UCL( 4.72 )
         ),
         Phase Level(
            "2",
            Sigma( .21 ),
            Avg( 4.29 ),
            LCL( 4 ),
            UCL( 4.5 )
         )
      ),
      R( Phase Level( "1" ), Phase Level( "2" ) )
   )
);
```

Distribution

You can set specification limits using the Spec Limits column property or using the `Spec Limits` argument in JSL:

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
dt << Distribution(
   Column( :OZONE ),
   Fit Distribution(
      Weibull(
         Spec Limits( LSL( 0.075 ), Target( 0.15 ), USL( 0.25 ) )
      )
   )
);
```

To set specification limits using K Sigma normal probabilities, use the `Set Spec Limits for KSigma` message. The first argument to the message is the value of K Sigma. By default, the limits are two-sided. To make them one-sided, use the optional `sided` argument. Specify `sided=1` for LSL only and `sided=2` for USL only.

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
obj = Distribution( Column( :CO ) );
obj << Fit Distribution(
   LogNormal( Set Spec Limits for KSigma( 3 ) )
);
```

DOE

Tip: For a complete list of DOE scripting options, select **Help > Scripting Index**, select **Objects** from the menu, and search for DOE.

To reproduce your DOE work, most DOE platforms provide these approaches:

- In the red triangle menu for the DOE window, the Save Script to Script Window option provides a script that reproduces the work that you have entered. This option is not available for the Nonlinear Design and Taguchi Array platforms.
- In the output design tables created by most DOE platforms, a script called DOE Dialog reproduces the work in the DOE window that created the design table. This script also contains the random seed (and number of starts or column starts, where appropriate) that enables you to reproduce your design exactly.

For many DOE platforms, the design table that is created also contains one or more scripts that you can run to conduct an appropriate analysis. These scripts are prefilled with the appropriate settings for your design.

The Random Seed

In constructing designs using the DOE platform, JMP uses a random seed to control certain actions that have a random component. A random seed can control the following:

- initializing search algorithms for design generation
- randomizing Run Order for design construction
- selecting a starting design for designs based on random starts

To reproduce a design using a script, you need to specify the random seed that generated them. For designs using random starts, set a random seed in the script before making the design.

The following example creates a Custom Design, sets the random seed, and makes the design:

```
DOE(
  Custom Design,
  Add Factor( Continuous, -1, 1, "X1", 0 ),
  Add Factor( Continuous, -1, 1, "X2", 0 ),
  Set Random Seed( 34067086 ),
  Make Design
);
```

Reproduce a Custom Design or a Main Effects Screening Design

Custom designs and main effects screening designs (constructed in the Screening Design platform) are generated by allocating a maximum number of seconds (Design Search Time) spent searching for a design. The default setting for Design Search Time is based on the complexity of the design. Even when two custom or main effects screening designs are constructed using the same random seed, the designs might not be identical because of different machine processing capabilities.

When you save the script for either of these designs using the Save Script to Script Window option, or when the DOE Dialog script is saved to the data table, the script specifies the Random Seed and the Number of Starts. The Number of Starts is the number of random starts used during the Design Search Time. The combination of the Set Random Seed and the Number of Starts options enables you to reproduce the design.

Note: If you write your own script, be aware that the order in which you specify options in your script can affect your results. For example, you should specify the design options before you make the design. In particular, if your design is constructed using a random seed, set it before you make the design.

Global Variables

Here are some global variables used to initiate or tune search algorithms.

Starting Design Enables you to specify a starting design. The following example replaces the random starting design with a specified matrix:

```
DOE Starting Design = matrix;
```

If a starting design is supplied, the Custom Design platform has only one start using this design.

K Exchange Value The coordinate exchange algorithm considers every row of factor settings for possible replacement in every iteration. You can limit the number of rows considered for exchange in an iteration so that only a small number of most likely rows is considered. The following example tells the algorithm to consider only the three most likely rows for exchange in each iteration:

```
DOE K Exchange Value = 3;
```

Bayes Diagonal Defines a vector that is added to the diagonal elements of the $X'X$ matrix. This new matrix is used in finding the D-optimal design. The following example adds the elements of `vector` to the diagonal elements of the $X'X$ matrix:

```
Bayes Diagonal = vector:
```

Fit Model

When scripting the Fit Model platform, here are a few general tips for controlling the behavior of the launch window:

- To keep the launch window open and fit the model at the same time, do one of the following (they are equivalent):
  - Include the `Run Model` message in your script.
  - Include both the `Run` message and the `Keep Dialog Open(1)` message.
- To run the model without showing the launch window, include the `Run` message in your script.

Model Scripts in the Data Table

When a data table contains a *Model* script and you launch Fit Model interactively, the Fit Model launch window is pre-populated according to the roles assigned by the table script.

Fit Group

If you want to fit different models and see a shared Profiler so that you can compare the models, use the `Fit Group` function. You can fit models such as least squares, nonlinear, neural, Gaussian processing, and mixed, in the same window with a shared Profiler. The following example creates both a Standard Least Squares model and a Gaussian Process model.

```
dt = Open( "$SAMPLE_DATA/Tiretread.jmp" );
obj = dt << Fit Group(
  Fit Model(
     Y( :ABRASION ),
     Effects(
        :SILICA & RS,
        :SILANE & RS,
        :SULFUR & RS,
        :SILICA * :SILICA,
        :SILICA * :SILANE,
        :SILANE * :SILANE,
        :SILICA * :SULFUR,
        :SILANE * :SULFUR,
        :SULFUR * :SULFUR
     ),
     Personality( "Standard Least Squares" ),
     Emphasis( Minimal Report ),
     Run()
  ),
  Gaussian Process(
     Y( :HARDNESS ),
     X( :SILICA, :SILANE, :SULFUR ),
     Set Correlation Function( "Cubic" )
  )
);
```

In the report, click the Fit Group red triangle and select **Profiler**. Scroll down to the Prediction Profiler. You can see the ABRASION row for the SLS model, and the HARDNESS row for the Gaussian Process model appearing in the same profiler.

Effects

Effects in the model can be more than just a list of columns, and can have a specialized syntax:

```
Effects( list of effects, list of effect macros, or both lists );
```

An effect can be a column name, a crossing of several column names with asterisk (*) notation, or nested columns specified with subscript bracket ([ ]) notation. Additional effect options can appear after an ampersand (&) character. Here are some examples:

```
A,                  // a column name alone is a main effect
A*B,                // a crossed effect, interaction, or polynomial
A[B],               // nested
A*B[C D],           // crossed and nested
effect&Random,      // a random effect
effect&LogVariance, // a variance model term
effect&RS,          // a response surface term
effect&Mixture,     // for an effect participating in a mixture
effect&Excluded,    // for an effect that with no model arguments
effect&Knotted,     // for a knotted spline effect
```

Effect macros are:

```
Factorial( columns ),   // for a full factorial design
Factorial2( columns ),  // for up to 2nd-degree interactions only
Polynomial( columns ),  // for a 2nd-degree polynomial only
```

Responses and Effects for MANOVA

To address an individual response function analysis, use a subscripted `Response`:

```
manovaObj << ( Response[ 1 ] << {response options} );
manovaObj << ( Response[ "Contrast" ] << {response options} );
```

Each response function supports the `Custom Test` message:

```
Custom Test( matrix, <Power Analysis( ... )>, <Label( "..." )> )
```

where each row of the *matrix* specifies coefficients for all the arguments in the model.

To address an individual Effect test, use subscripted `Effect` with a name or number:

```
manovaObj << ( Response[1] << ( Effect["Whole Model"] << {effect options} ) );
manovaObj << ( Response[1] << ( Effect[i] << {effect options} ) );
```

The effects are numbered:

- 0 for the intercept
- 1, 2, and so on, for regular effects
- *n*+1 for the "Whole Model" test, where *n* is the number of effects *not* including the intercept

Each effect in each response function supports the following messages, where each row of the *matrix* has coefficients for all the levels in the effect:

```
Test Details( 1 ),
Centroid Plot( 1 ),
```

```
Save Canonical Scores,
Contrast( matrix, <Power Analysis(...)> );
```

For example, the following JSL script adds test details for an effect to the report window:

```
dt = Open( "$SAMPLE_DATA/Dogs.jmp" );
manObj = dt << Fit Model(
   Y( logHist0, logHist1, logHist3, logHist5 ),
   Effects( dep1, drug, drug * dep1 ),
   Personality( "MANOVA" ),
   Run Model
);
manObj << Response Function( "Contrast" );
manObj << (Response["Contrast"] << (Effect["Whole Model"] <<

/* send the Test Details message to the Whole Model outline
under Contrast in the report window */
Test Details( 1 )));

/* send the Test Details message to response 1 (Contrast)
and effect 3 (drug*dep1) */
manObj << (Response[1] << (Effect[3] << Test Details( 1 )));
```

Send Function

Tip: Recall that the << operator is equivalent to the `Send()` function.

When you have multiple responses, you can send messages to a specific response column's fit. Use the following JSL (where *responseName* is the specific response):

```
fitObj << ( responseName << {options, ...});
```

The second `Send()` function finds the named response and sends the list of messages to it.

Note: If you send the messages directly to the *fitObj* with a single `Send()` function, the messages are sent to all responses.

In the following script, the last line sends a message to the AICc report for the ABRASION response:

```
dt = Open( "$SAMPLE_DATA/Tiretread.jmp" );
fitObj = dt << Fit Model(
   Y( :ABRASION, :MODULUS, :ELONG, :HARDNESS ),
   Effects(
      :SILICA & RS,
      :SILANE & RS,
      :SULFUR & RS,
```

```
        :SILICA * :SILICA,
        :SILANE * :SILICA,
        :SILANE * :SILANE,
        :SULFUR * :SILICA,
        :SULFUR * :SILANE,
        :SULFUR * :SULFUR
    ),
    Personality( "Standard Least Squares" ),
    Run
);
/* shows up under the Studentized Residuals plot*/
fitObj << (:ABRASION << {AICc( 1 )});
```

To send messages to an individual effect, nest even further:

```
fitObj << ( responseName << ((effectName) << effectOption ));
```

Standard Least Squares

For Standard Least Squares models that contain only fixed effects and more than one Y response, you can choose to fit the models for the Y responses together or separately. If some of the rows have missing values, the following apply:

- In a script, the Y responses are fit together by default. The model fits each Y using only those rows that are nonmissing for all of the Y variables. For example, a row that includes one or more missing Y values is *excluded* from the model. You can explicitly include `Run( "Fit Together" )` in a script to get the same result.
- If you include the `Run( "Fit Separately" )` option, the model fits each Y using all rows that are nonmissing for that particular Y. For example, a row that includes one or more missing Y values is *included* in the model.

The following script fits the Y responses together:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Fit Model(
    Y( :height, :weight ),
    Effects( :sex ),
    Personality( "Standard Least Squares" ),
    Emphasis( "Minimal Report" ),
    Run // or Run( "Fit Together" )
);
```

The following script fits the Y responses separately:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Fit Model(
    Y( :height, :weight ),
    Effects( :sex ),
```

```
    Personality( "Standard Least Squares" ),
    Emphasis( "Minimal Report" ),
    Run( "Fit Separately" )
);
```

When running the model in JMP, the user is prompted to select fitting the responses separately or together. The Fit Model launch window contains a Fit Separately option, which is deselected by default. For more information about how missing Y values are handled, see *Fitting Linear Models*.

Note: In models that contain a random effect, Y values are fit separately by default.

Formula Depot

Sending commands via JSL to a Formula Depot platform report requires that you first create an empty Formula Depot. Once you have a reference to the Formula Depot report, you can send commands to it. The following script opens a formula depot report and adds a model to the depot.

```
dt = Open( "$SAMPLE_DATA/Liver Cancer.jmp" );
fd = Formula Depot(); // open a new Formula Depot
obj1 = dt << Run Script( "Lasso Poisson, Validation Column" );
model = obj1 << (Fit[1] << Publish Prediction Formula );

/* Publish a model to the Formula Depot. The
following script opens a formula depot report, adds a model to
the depot, generates scoring code for the formula, saves the code to a file,
and closes all windows.*/
case_name = "fitLS_prediction";
dest_dir = "c:\"; //CHANGE ME to a better location
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Print( "Running " || case_name );
fm_ls = Fit Model(
   Y( :weight ),
   Effects( :age, :sex, :height ),
   Personality( "Standard Least Squares" ),
   Emphasis( "Minimal Report" ),
   Run
);
fd = fm_ls << Publish Prediction Formula;
Try(
   Print( "Generate Python Code" );
   code = fd << Generate Python Code( No Editor );
   Save Text File( dest_dir || case_name || ".py", code );
   Print( "Done." );
```

```
,
  Print( "Code generation failed for " || case_name || ", reason: " );
  Show( exception_msg );
);
fdw = Window( "Report: Formula Depot" );
fdw << Close Window( No Save );
Close( dt, No Save );
```

Graph Builder

The following is an example of the basic structure of a Graph Builder script:

```
Graph Builder(
Size( x, y ),
// Use any commands from the red triangle menu - Size is an example
Variables( role( column ), ...), //Required
Elements( element name( element options, ...), //Required
Data Filter( ... ),
// Use any display customizations here - Data Filter is an example
);
```

The following is an example of a simple Graph Builder script that creates a bar chart:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Graph Builder(Variables(
  X( :age ),
  Y( :weight ) ),
Elements( Bar( X, Y, Legend( 8 ) ) )
// Legend is not required, but provides an internal legend ID that you can use
  for later customization
);
```

The following is a more detailed Graph Builder script that creates a bar chart:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Graph Builder(
Size( 373, 332 ),
Show Control Panel( 0 ),
Variables(
  X( :age ),
  Y( :weight ),
  Y( Transform Column( "weight25",
       Formula( Col Quantile( :weight, 0.25, :age ) )
       ),
     Position( 1 )
/* By default each Y variable gets its own axis, but the Position(1)
option keeps each additional Y in the first slot and merges the axes. */
  ),
```

```
    Y( Transform Column( "weight75", Formula( Col Quantile( :weight, 0.75, :age
    ) ) ),
       Position( 1 )
    )
  ),
  Elements(
    Bar( X, Y( 2 ), Y( 3 ), Legend( 10 ), Bar Style( "Range" ) ),
  // This bar element uses only the second and third Y variables and
  // draws a Range bar between the values.*/
    Bar( X, Y( 1 ), Legend( 8 ),
       Bar Style( "Float" ),Summary Statistic( "Median" )
  // This bar element uses the first Y variable and draws its median with the
  // float style, which is a short horizontal line
    ),
    Points( X, Y( 1 ), Legend( 9 ) )
  // The Points element shows the raw data as markers.
  ),
  SendToReport(
  // Contains display customizations
    Dispatch( {}, "400", ScaleBox,
  // 400 is the fixed name
       {Legend Model( 10,
       Level Name( 0, "1st to 3rd
       quartiles",
       Item ID( "Mean(weight25)..Mean(weight75)", 1 )
  // Renames the legend, moves the legend name onto several lines, and changes
  // the color for legend 10 (the Range bar)
    ),
       Properties( 0, {Fill Color( 32 )},
       Item ID( "Mean(weight25)..Mean(weight75)", 1 )
  // Changes the color for legend 10 (the range bar)
    )
       ), Legend Model( 9,
       Properties( 0, {Marker Size( 4 )}, Item ID( "weight",1 ) )
  // Changes the marker size for legend 9 (the Points element)
    )}
  ),
  Dispatch( {}, "graph title", TextEditBox,
    {Set Text( "Weights with quartile ranges" )}
  // Changes the graph title
  ),
  Dispatch( {}, "Y title", TextEditBox, {Set Text( "weight" )} ),
  // Changes the Y axis title
  Dispatch( {}, "400", LegendBox,
    {Legend Position( {10, [0], 8, [1], 9, [-1]} )}
  // Hides the legend item for legend 9 by giving it a position of -1
```

```
    )
)
);
```

Neural and Neural Net

The Neural platform replaces the Neural Net platform. Scripts that use Neural Net are deprecated and might not work in the future. It is recommended to use the Neural platform for new scripts and update any old scripts using Neural Net to Neural.

Partial Least Squares and PLS

The Partial Least Squares (PLS) platform has replaced the PLS platform. Scripts that use PLS might not work in the future. It is recommended to use the Partial Least Squares platform for new scripts and update any old scripts using PLS to Partial Least Squares.

Process Capability

The Process Capability platform has replaced the Capability platform. Scripts that use Capability might not work in the future. It is recommended to use the Process Capability platform for new scripts and update any old scripts using Capability to Process Capability.

Specification Limits in JSL Scripts

Specification limits can be read from JSL scripts, column properties, or from a specification limits data table. As an example of reading in specification limits from JSL, consider the following example:

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
dt << Process Capability(
   Process Variables( :OZONE, :CO, :SO2, :NO ),
   Spec Limits(
      OZONE( LSL( 0.075 ), Target( 0.15 ), USL( 0.25 ) ),
      CO( LSL( 5 ), Target( 7 ), USL( 12 ) ),
      SO2( LSL( 0.01 ), Target( 0.04 ), USL( 0.09 ) ),
      NO( LSL( 0.01 ), Target( 0.025 ), USL( 0.04 ) )
   )
);
```

Using a Limits Data Table and JSL

There is no extra syntax needed to differentiate between the two table types (wide and tall) when they are read using JSL. This example is based on the CitySpecLimits.jmp sample data table. It places the specification limits data table inside an `Import Spec Limits()` expression.

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
dt << Process Capability(
   Process Variables( :OZONE, :CO, :SO2, :NO ),
   Spec Limits(
      Import Spec Limits(
         "$SAMPLE_DATA/CitySpecLimits.jmp"
)));
```

Scatterplot 3D

If your machine supports acceleration, and you want to display the 3-D scatterplot faster, you can turn on the Use Hardware Acceleration option, or in JSL, specify the `Scene3DHardwareAcceleration` global variable. A value of 1 turns it on, and a value of zero turns it off.

```
dt = Open( "$SAMPLE_DATA/Solubility.jmp" );
Scene3DHardwareAcceleration = 1;
dt << Scatterplot 3D(
   Y( :"1-Octanol"n, :Ether, :Chloroform, :Benzene, :Carbon Tetrachloride,
   :Hexane )
);
```

Text Explorer

By default, text is converted to all lowercase in the Text Explorer platform. You can use the `"From Source"` keyword in the Save Regex Column message to save the results of a custom regex that preserve the original case of the text. The first argument is the name of the column that is created in the data table. The second argument is the string `"From Source"` or `"From Result"`. The default behavior is equivalent to specifying the `"From Result"` keyword.

```
dt = Open( "$SAMPLE_DATA/Aircraft Incidents.jmp" );
obj = dt << Text Explorer(
   Text Columns( :Airport Name),
   Tokenizing( "Regex" ),
   Save Regex Column( "Output Column", "From Source" )
);
obj << Close Window();
```

Explore Outliers

The Explore Outliers platform provides options to identify, explore, and manage outliers in your univariate or multivariate data.

In the Probe.jmp sample data table, several columns contain outlier values of 9999. Many industries use nines as a missing value code. The following example selects VDP_PCOLL, VDP_PINNBASE, and VDP_PINPBASE, the columns with the highest number of 9s, and adds the Missing Value Codes column property to the columns. The script then rescans the columns and removes these columns from the report.

```
dt = Open( "$SAMPLE_DATA/Probe.jmp" );
obj = Explore Outliers(
   Y(
      :VDP_M1,
      :VDP_M2,
      :VDP_NBASE,
      :VDP_NEMIT,
      :VDP_NENBNI,
      :VDP_NSINK,
      :VDP_PBASE,
      :VDP_PBL,
      :VDP_PCOLL,
      :VDP_PEMIT,
      :VDP_PINNBASE,
      :VDP_PINPBASE,
      :VDP_PSINK,
      :VDP_SICR
   ),
   Quantile Range Outliers( 1 ),
   Show only columns with outliers( 1 ),
);
obj << Add Highest Nines to Missing Value Codes( :VDP_PCOLL, :VDP_PINNBASE,
   :VDP_PINPBASE );
obj << Rescan;
```

Scripting-Only Messages and Arguments by Platform

Some messages and arguments that can be sent to a platform object are scripting-only. This means that these options do not have corresponding options in JMP launch windows or reports. However, note that scripting-only options can appear as Preferences.

This section describes scripting-only messages and arguments for specific platforms.

Tip: For more information about any of these messages or arguments, and for specific examples, go to the Scripting Index (**Help** > **Scripting Index**).

- "Bivariate Script-Only Messages"
- "Boosted Tree Script-Only Messages"
- "Bootstrap Forest Script-Only Messages"
- "Bubble Plot Script-Only Messages"
- "Choice Script-Only Messages"
- "Clustering Script-Only Messages"
- "Contingency Script-Only Messages"
- "Contour Plot Script-Only Messages"
- "Contour Profiler Script-Only Messages"
- "Control Chart Script-Only Messages"
- "Control Chart Builder Script-Only Messages"
- "Cumulative Damage Script-Only Messages"
- "Custom Profiler Script-Only Messages"
- "Degradation Script-Only Messages"
- "Destructive Degradation Script-Only Messages"
- "Discriminant Script-Only Messages"
- "Distribution Script-Only Messages"
- "EMP Measurement Systems Analysis Script-Only Messages"
- "Explore Missing Values Script-Only Messages"
- "Explore Outliers Script-Only Messages"
- "Factor Analysis Script-Only Messages"
- "Fit Life by X Script-Only Messages"
- "Fit Model Script-Only Messages"
- "Fit Parametric Survival Script-Only Messages"
- "Functional Data Explorer Script-Only Messages"
- "Gaussian Process Script-Only Messages"
- "K Means Cluster Script-Only Messages"
- "K Nearest Neighbors Script-Only Messages"
- "Latent Class Analysis Script-Only Messages"
- "Life Distribution Script-Only Messages"
- "Make Validation Column Script-Only Messages"
- "MaxDiff Script-Only Messages"

- "Mixture Profile Script-Only Messages"
- "Model Screening Script-Only Messages"
- "Multivariate Script-Only Messages"
- "Naive Bayes Script-Only Messages"
- "Neural Script-Only Messages"
- "Nonlinear Script-Only Messages"
- "Normal Mixtures Script-Only Messages"
- "Oneway Script-Only Messages"
- "Parallel Plot Script-Only Messages"
- "Pareto Plot Script-Only Messages"
- "Partial Least Squares Script-Only Messages"
- "Partition Script-Only Messages"
- "Principal Components Script-Only Messages"
- "Process Capability Script-Only Messages"
- "Process Screening Script-Only Messages"
- "Profiler Script-Only Messages"
- "Reliability Forecast Script-Only Messages"
- "Reliability Growth Script-Only Messages"
- "Response Screening Script-Only Messages"
- "Scatterplot 3D Script-Only Messages"
- "Structural Equation Models Script-Only Messages"
- "Support Vector Machines Script-Only Messages"
- "Surface Plot Script-Only Messages"
- "Tabulate Script-Only Messages"
- "Ternary Plot Script-Only Messages"
- "Text Explorer Script-Only Messages"
- "Time Series Script-Only Messages"
- "Uplift Script-Only Messages"
- "Variability Chart Script-Only Messages"

Bivariate Script-Only Messages

The Fit Where message for a Bivariate object fits a curve or line to a single level of a categorical variable.

```
biv_object << Fit Where( WHERE_clause, fit_command );
```

The Curve message for a Bivariate object enables you to send a message to a curve or line that has already been fit in the Bivariate report. The curves are numbered sequentially in the order they appear in the report.

```
biv_object << ( Curve[n] << curve_message );
```

The Set Kernel message for a Nonpar Density curve in a Bivariate object enables you to set the kernel standard deviations for the X and Y values:

```
biv_object << ( Curve[n] << Set Kernel( x, y ) );
```

Boosted Tree Script-Only Messages

When you launch the Boosted Tree platform via JSL, you can use the following arguments that are available only in JSL:

```
Maximum Depth( number )
Use Excluded Rows for Validation( 0|1 )
```

The following messages for a Boosted Tree object are available only in JSL:

```
boosted_tree_object << Get Average Absolute Error Test;
boosted_tree_object << Get Average Absolute Error Training;
boosted_tree_object << Get Average Absolute Error Validation;
boosted_tree_object << Get Average Log Error Test;
boosted_tree_object << Get Average Log Error Training;
boosted_tree_object << Get Average Log Error Validation;
boosted_tree_object << Get Confusion Matrix Test;
boosted_tree_object << Get Confusion Matrix Training;
boosted_tree_object << Get Confusion Matrix Validation;
boosted_tree_object << Get Confusion Rates Test;
boosted_tree_object << Get Confusion Rates Training;
boosted_tree_object << Get Confusion Rates Validation;
boosted_tree_object << Get Gen RSquare Test;
boosted_tree_object << Get Gen RSquare Training;
boosted_tree_object << Get Gen RSquare Validation;
boosted_tree_object << Get Measures;
boosted_tree_object << Get Microseconds;
boosted_tree_object << Get Misclassification Rate Test;
boosted_tree_object << Get Misclassification Rate Training;
boosted_tree_object << Get Misclassification Rate Validation;
boosted_tree_object << Get MM SAS DATA Step;
boosted_tree_object << Get MM Tolerant SAS DATA Step;
boosted_tree_object << Get Prediction Formula;
boosted_tree_object << Get RMS Error Test;
boosted_tree_object << Get RMS Error Training;
boosted_tree_object << Get RMS Error Validation;
boosted_tree_object << Get ROC Area Test;
```

```
boosted_tree_object << Get ROC Area Training;
boosted_tree_object << Get ROC Area Validation;
boosted_tree_object << Get RSquare Test;
boosted_tree_object << Get RSquare Training;
boosted_tree_object << Get RSquare Validation;
boosted_tree_object << Get SAS DATA Step;
boosted_tree_object << Get Seconds;
boosted_tree_object << Get Tolerant Prediction Formula;
boosted_tree_object << Get Tolerant SAS DATA Step;
```

Bootstrap Forest Script-Only Messages

When you launch the Boosted Tree platform via JSL, you can use the following arguments that are available only in JSL:

```
Portion Bootstrap( fraction )
Time Limit( number )
Use Excluded Rows for Validation( 0|1 )
```

The following messages for a Bootstrap Forest object are available only in JSL:

```
bootstrap_forest_object << Get Average Absolute Error Test;
bootstrap_forest_object << Get Average Absolute Error Training;
bootstrap_forest_object << Get Average Absolute Error Validation;
bootstrap_forest_object << Get Average Log Error Test;
bootstrap_forest_object << Get Average Log Error Training;
bootstrap_forest_object << Get Average Log Error Validation;
bootstrap_forest_object << Get Confusion Matrix Test;
bootstrap_forest_object << Get Confusion Matrix Training;
bootstrap_forest_object << Get Confusion Matrix Validation;
bootstrap_forest_object << Get Confusion Rates Test;
bootstrap_forest_object << Get Confusion Rates Training;
bootstrap_forest_object << Get Confusion Rates Validation;
bootstrap_forest_object << Get Gen RSquare Test;
bootstrap_forest_object << Get Gen RSquare Training;
bootstrap_forest_object << Get Gen RSquare Validation;
bootstrap_forest_object << Get Measures;
bootstrap_forest_object << Get Microseconds;
bootstrap_forest_object << Get Misclassification Rate Test;
bootstrap_forest_object << Get Misclassification Rate Training;
bootstrap_forest_object << Get Misclassification Rate Validation;
bootstrap_forest_object << Get MM SAS DATA Step;
bootstrap_forest_object << Get MM Tolerant SAS DATA Step;
bootstrap_forest_object << Get Prediction Formula;
bootstrap_forest_object << Get RMS Error Test;
bootstrap_forest_object << Get RMS Error Training;
bootstrap_forest_object << Get RMS Error Validation;
```

```
bootstrap_forest_object << Get ROC Area Test;
bootstrap_forest_object << Get ROC Area Training;
bootstrap_forest_object << Get ROC Area Validation;
bootstrap_forest_object << Get RSquare Test;
bootstrap_forest_object << Get RSquare Training;
bootstrap_forest_object << Get RSquare Validation;
bootstrap_forest_object << Get SAS DATA Step;
bootstrap_forest_object << Get Seconds;
bootstrap_forest_object << Get Tolerant Prediction Formula;
bootstrap_forest_object << Get Tolerant SAS DATA Step;
```

Bubble Plot Script-Only Messages

The following messages for a Bubble Plot object are available only in JSL:

```
bubble_plot_object << Auto Stretching( string );
bubble_plot_object << Bubble Size( number );
bubble_plot_object << Color Levels( vector );
bubble_plot_object << Get Custom Path;
bubble_plot_object << Get Draw;
bubble_plot_object << Get Label;
bubble_plot_object << Get Shape;
bubble_plot_object << Go;
bubble_plot_object << Label Offset( list );
bubble_plot_object << Prev;
bubble_plot_object << Set Custom Path( ... );
bubble_plot_object << Speed( number );
bubble_plot_object << Step;
bubble_plot_object << Stop;
bubble_plot_object << Time Index( number );
bubble_plot_object << Title Position( x, y );
bubble_plot_object << Toggle Animation;
```

Choice Script-Only Messages

When you launch the Choice platform via JSL, you can set the acceptable criterion for parameter convergence. In addition, when using Hierarchical Bayes, you can set the number of burn-in iterations.

```
Convergence Criterion( fraction )
Number of Burn In Iterations ( number)
```

Clustering Script-Only Messages

The following messages for a Hierarchical Cluster object are available only in JSL:

```
hierarchical_cluster_object << Get Column Names;
hierarchical_cluster_object << Get Clusters;
hierarchical_cluster_object << GetDisplayOrder;
hierarchical_cluster_object << GetColumnDisplayOrder;
hierarchical_cluster_object << Get Distance Matrix;
hierarchical_cluster_object << Standardize( 0|1 );
```

The following messages for a KDTable object are available only in JSL:

```
kdtable_object << Distance between rows( row1, row2 );
kdtable_object << Insert rows( number|[ vector ] );
kdtable_object << K nearest rows( stop, <position> );
kdtable_object << Remove rows( number|[ vector ] );
```

Contingency Script-Only Messages

The following message is available only in JSL. It creates a data table that contains values from the Correspondence Analysis plot in a Contingency report.

```
contingency_obj << Correspondence Analysis( Make Table );
```

Contour Plot Script-Only Messages

The following message for a Contour Plot object is available only in JSL:

```
contour_plot_object << Set Alpha( number );
```

Contour Profiler Script-Only Messages

The following messages for a Contour Profiler object are available only in JSL:

```
contour_profile_object << Contour Value( y1( ... ), y2( ... ), ... );
contour_profile_object << Get Constraints;
contour_profile_object << Get Factor Settings;
contour_profile_object << Get Factor Settings Script;
contour_profile_object << Get Simulator;
contour_profile_object << Horizontal Factor( column );
contour_profile_object << Number of Plots Across( number );
contour_profile_object << Reset;
contour_profile_object << Term Value( x1( number ), x2( number ), ... );
contour_profile_object << Vertical Factor( column );
contour_profile_object << Y Colors( color1, color2, ... );
```

Control Chart Script-Only Messages

When you launch the Control Chart platform via JSL, you can use the following arguments that are available only in JSL:

```
Chart Col( column, chart_type( Avg( number ), UCL( number ), LCL( number ) )
Chart Type( string )
Get Limits( string )
Get Targets( string ) //Multivariate Control Charts
Use Excluded Points on MR
```

The following messages for a Control Chart object are available only in JSL:

```
control_chart_object << Point Marker( string ); //CUSUM charts
control_chart_object << Save Limits( string );
```

Control Chart Builder Script-Only Messages

When you launch the Control Chart Builder platform via JSL, you can use the following arguments that are available only in JSL:

```
Chart( position( i ), Add Limits( ... ) )
Chart( position( i ), Add Spec Limits( ... ) )
Chart( position( i ), Limits( ... ) )
Chart( position( i ), Points( ... ) )
Chart( position( i ), Set Control Limits( ... ) )
Class( string )
Customize Tests( ... )
KSigma( number )
Show Lower Control Limit( 0|1 )
Show Two Shewhart Charts( 0|1 )
Variables( ... )
Variables( ..., nTrials( column|number ) )
```

The following messages for a Control Chart Builder object are available only in JSL:

```
control_chart_builder_object << Chart( position( i ), ... );
control_chart_builder_object << Rerun All Tests;
control_chart_builder_object << Save Limits( string );
control_chart_builder_object << Size( width, height );
```

Cumulative Damage Script-Only Messages

The following message for a Cumulative Damage object is available only in JSL. It returns a list of model fit results.

```
cumulative_damage_object << Get Results;
```

Custom Profiler Script-Only Messages

The following messages for a Custom Profiler object are available only in JSL:

```
custom_profiler_object << Get Constraints;
custom_profiler_object << Get Factor Settings;
custom_profiler_object << Get Factor Settings Script;
custom_profiler_object << Get Objective;
custom_profiler_object << Get Objective Formula;
custom_profiler_object << Get Simulator;
custom_profiler_object << Objective Formula;
custom_profiler_object << Reset;
custom_profiler_object << Term Value( x1( number ), x2( number ), ... );
```

Degradation Script-Only Messages

The following messages for a Degradation object are available only in JSL:

```
degradation_object << Get Inverse Prediction Results;
degradation_object << Get Prediction Results;
degradation_object << Get Residuals;
degradation_object << Get Results;
degradation_object << Residual Plot;
degradation_object << Set Baseline;
```

Destructive Degradation Script-Only Messages

The following messages for a Destructive Degradation object are available only in JSL:

```
destructive_degradation_object << Control( yTrans, xTrans, distribution, path
  );
destructive_degradation_object << Model( yTrans, xTrans, distribution, path
  );
```

Discriminant Script-Only Messages

The following messages for a Discriminant object are available only in JSL:

```
discriminant_object << Get Descrim Matrices;
discriminant_object << Get Measures;
discriminant_object << Score Data( 0|1 );
```

Distribution Script-Only Messages

The following message for a Distribution object is available only in JSL. The message creates an array of handles to the fitted distributions, so you can send commands to specific distributions:

```
dist_object << ( Fit Handle[n] << ...);
```

You can also get a reference to a fitted distribution directly if you send the Fit Distribution message to a Distribution object. This enables you to send commands to specific distributions:

```
fit_object = dist_object << Fit Distribution( distribution_name );
fit_object << ...;
```

The following message for a Distribution object is available only in JSL. The message specifies that PpK labeling be used in the Process Capability reports within the Distribution report object.

```
dist_object << PpK Capability Labeling( 1 );
```

The following message for a Distribution object is available only in JSL:

```
dist_object << Outlier Box Plot Row Cutoff( number );
```

The following messages for a Distribution object use the name of a fitted distribution, such as `Fit Lognormal`, and they are available only in JSL:

```
dist_object << Fit Distribution Name( Process Capability( Set Proabilities for
  Quantile Spec Limits( ... ) );
dist_object << Fit Distribution Name( Process Capability( Set Sigma Multiplier
  for Quantile Spec Limits( ... ) );
```

EMP Measurement Systems Analysis Script-Only Messages

The following messages for an EMP Measurement Systems Analysis object are available only in JSL:

```
emp_msa_object << Conv Limit( number );
emp_msa_object << Dispersion Chart( 0|1 );
emp_msa_object << Include Interactions in Reproducibility( 0|1 );
emp_msa_object << Max Iter( number );
emp_msa_object << Set Alpha Level( number );
emp_msa_object << Set Random Seed( number );
emp_msa_object << Show Monitor Classification Legend( 0|1 );
emp_msa_object << Show Part Legend( 0|1 );
emp_msa_object << Show Shift Detection Profiler Legend( 0|1 );
```

Explore Missing Values Script-Only Messages

When you launch Automated Data Imputation through the Explore Missing Values utility via JSL, you can use the following arguments that are available only in JSL:

```
Maximum Dimension( number )
Validation Portion( number )
```

The following messages for an Explore Missing Values report are available only in JSL:

```
explore_missing_values_object << Get U V Sigma ADI Matrices;
```

Explore Outliers Script-Only Messages

When you launch K Nearest Neighbors Outliers through the Explore Outliers utility via JSL, you can use the following arguments that are available only in JSL:

```
Impute Missing( 0|1 )
```

The following messages for a Quantile Range Outliers object in Explore Outliers object are available only in JSL:

```
explore_outliers_obj << Get Quantile Outliers;
explore_outliers_obj << Save Quantile Outlier Limits;
```

When you launch Robust PCA Outliers through the Explore Outliers utility via JSL, you can use the following arguments that are available only in JSL:

```
Tolerance( number )
```

Factor Analysis Script-Only Messages

The following messages for a Factor Analysis model fit are available only in JSL. The model fits are numbered sequentially in the order they appear in the report.

```
factor_analysis_object << ( Fit[n] << Save Rotated Components );
factor_analysis_object << ( Fit[n] << Save Rotated Components with Imputation
  );
```

Fit Life by X Script-Only Messages

The following messages for a Fit Life by X object are available only in JSL:

```
fit_life_object << Add Density Curve to Scatterplot( number );
fit_life_object << Add Quantile Line to Scatterplot( number );
fit_life_object << Density( ... );
fit_life_object << Get Results;
fit_life_object << Hazard( ... );
```

```
fit_life_object << Maximum Iterations( number );
fit_life_object << Probability( ... );
fit_life_object << Quantile( ... );
fit_life_object << Rejection Sample Maximum Trials( number );
fit_life_object << Set Level of Quantile Line CI Bands( number );
fit_life_object << Set Scale( distribution );
fit_life_object << Set Scriptables( ... );
fit_life_object << TAF( ... );
fit_life_object << Time Acceleration Baseline( number );
```

Fit Model Script-Only Messages

The following messages for a Model Dialog object are available only in JSL:

```
model_dialog_object << Cauchy Fit( 0|1 ); //Response Screening
model_dialog_object << Choose High Target( 0|1 ); //Nominal Logistic
model_dialog_object << Generalized Distribution( string ); //Generalized
  Regression
model_dialog_object << GLM Distribution( string ); //Generalized Linear Model
model_dialog_object << Max Iterations( number ); //Partial Least Squares
model_dialog_object << Power Link Parameter( number ); //Generalized Linear
  Model
model_dialog_object << Robust Fit( 0|1 ); //Response Screening
model_dialog_object << Run Model;
```

The following sections describe scripting-only messages for MANOVA, Generalized Linear Models, Generalized Regression, Nominal and Ordinal Logistic, Standard Least Squares, and Stepwise.

MANOVA

The following message for a Fit MANOVA object is available only in JSL:

```
fit_model_object << ( Response[1] << (Effect[1] << ... ) );
```

The third message can be one of the following:

```
Test Details // show or hide the test details for an individual effect
Centroid Plot // show or hide the centroid plot for an individual effect
Save Canonical Scores // save the canonical scores for an individual effect
Contrast /* run a customized F test contrasting different levels for
        an effect in the model*/
```

The following example saves canonical scores for the Response Function, which in this data, is Sum:

```
dt = Open( "$SAMPLE_DATA/Dogs.jmp" );
obj = dt << Fit Model(
  Y( :LogHist0, :LogHist1, :LogHist3, :LogHist5 ),
```

```
    Effects( :drug, :dep1, :drug * :dep1 ),
    Personality( "Manova" ),
    Run( Response Function( Sum ) )
);
obj << (Response[1] << (Effect[1] << Save Canonical Scores) );
```

Generalized Linear Models

The following messages for a Fit Generalized Linear Model object are available only in JSL:

```
fit_model_object << GLM Distribution( distribution );
fit_model_object << Parametric Formula;
fit_model_object << Power Link Parameter( number );
```

Generalized Regression

The following messages for a Fit Generalized Regression object are available only in JSL:

```
fit_model_object << Fit( ... );
fit_model_object << Get X Matrix;
```

The following messages for a Fit Generalized Regression object are available only in JSL and must be sent to an individual model fit. The model fits are numbered sequentially in the order they appear in the report.

```
fit_model_object << (Fit[n] << Get Prediction Formula);
fit_model_object << (Fit[n] << Set Solution ID( number ));
fit_model_object << (Fit[n] << Step Backward);
fit_model_object << (Fit[n] << Step Forward);
```

Nominal and Ordinal Logistic

The following messages for Fit Nominal Logistic and Fit Ordinal Logistic objects are available only in JSL:

```
fit_model_object << Get Confusion Matrix Test;
fit_model_object << Get Confusion Matrix Training;
fit_model_object << Get Confusion Matrix Validation;
fit_model_object << Get Confusion Rates Test;
fit_model_object << Get Confusion Rates Training;
fit_model_object << Get Confusion Rates Validation;
fit_model_object << Get MM SAS Data Step;
fit_model_object << Get SAS Data Step;
fit_model_object << Get Measures;
fit_model_object << Get Probability Formulas;
```

The following message for a Fit Nominal Logistic object is available only in JSL:

```
fit_model_object << Positive Level( level );
```

Standard Least Squares

The following messages for a Fit Least Squares object are available only in JSL:

```
fit_model_object << Get Conditional Formula;
fit_model_object << Get Effect Names;
fit_model_object << Get Effect PValues;
fit_model_object << Get Estimates;
fit_model_object << Get Indiv Confid Limit Formula;
fit_model_object << Get Mean Confid Limit Formula;
fit_model_object << Get Measures;
fit_model_object << Get MM SAS DATA Step;
fit_model_object << Get Parameter Names;
fit_model_object << Get Parameterized Formula;
fit_model_object << Get Prediction Formula;
fit_model_object << Get Random Effect Names;
fit_model_object << Get Std Errors;
fit_model_object << Get Standard Error Formula;
fit_model_object << Get Variance Components;
fit_model_object << Get X Matrix;
fit_model_object << Get XPX Inverse;
fit_model_object << Get Y Matrix;
fit_model_object << Profiler( Extrapolation Control Criterion( string ) );
fit_model_object << Show VIF;
```

Stepwise

The following messages for a Fit Stepwise object are available only in JSL:

```
fit_stepwise_object << Backward Step;
fit_stepwise_object << Enter( column );
fit_stepwise_object << Finish; // preferred in scripts over << Go
fit_stepwise_object << Forward Step;
fit_stepwise_object << Get Measures;
fit_stepwise_object << Lock( column );
fit_stepwise_object << Prob to Enter( number );
fit_stepwise_object << Prob to Leave( number );
fit_stepwise_object << Remove( column );
fit_stepwise_object << Unlock( column );
```

Fit Parametric Survival Script-Only Messages

The following messages for a Fit Parametric Survival object are available only in JSL:

```
fit_parametric_survival_object << Estimate Time Quantile( ... );
fit_parametric_survival_object << Get Parameter Names;
fit_parametric_survival_object << Get Estimates;
```

```
fit_parametric_survival_object << Get Std Errors;
fit_parametric_survival_object << Get Effect Names;
fit_parametric_survival_object << Get Effect PValues;
```

Functional Data Explorer Script-Only Messages

The following messages for a Functional Data Explorer FDOE object are available only in JSL:

```
fdoe_object << Generalized Regression FPC Model( FPC Number( n ), Commands );
```

Gaussian Process Script-Only Messages

When you launch the Gaussian Process platform via JSL, you can use the following arguments that are available only in JSL:

```
Set Nugget( number )
```

K Means Cluster Script-Only Messages

The followings messages for a K Means Cluster object are available only in JSL:

```
kmeans_cluster_object << Get Statistics;
```

K Nearest Neighbors Script-Only Messages

When you launch the K Nearest Neighbors platform via JSL, you can use the following arguments that are available only in JSL:

```
Use Excluded Rows for Validation( 0|1 )
```

The following messages for a K Nearest Neighbors object are available only in JSL:

```
knn_object << Get Measures;
knn_object << ( Response[number] << Get Prediction Formula );
knn_object << ( Response[number] << Get Best K );
knn_object << ( Response[number] << Set K( number ) );
```

Latent Class Analysis Script-Only Messages

The following message for a Latent Class Analysis model fit constructs a script to create probability formula columns and returns them to the log. This message is available only in JSL.

```
latent_class_analysis_object << ( Fit[number] << Get Probability Formulas );
```

The following example sets a random seed in order for the model to be reproducible.

```
dt = Open( "$SAMPLE_DATA/Car Poll.jmp" );
obj = dt << Latent Class Analysis(
   Y( :sex, :marital status, :country, :size, :type ),
   Set Random Seed( 12345 ),
   Number of Clusters( 3 ),
);
```

Life Distribution Script-Only Messages

The following messages for a Life Distribution object are available only in JSL:

```
life_dist_obj << Compute Remaining Life Distribution( time0, time1 );
life_dist_object << Confidence Invterval Method( string );
life_dist_object << Density( number );
life_dist_object << Fit DS Frechet;
life_dist_object << Fit DS Loglogistic;
life_dist_object << Fit DS Lognormal;
life_dist_object << Fit DS Weibull;
life_dist_object << Fit Exponential;
life_dist_object << Fit Frechet;
life_dist_object << Fit GenGamma;
life_dist_object << Fit LEV;
life_dist_object << Fit LogGenGamma;
life_dist_object << Fit Logistic;
life_dist_object << Fit Loglogistic;
life_dist_object << Fit Lognormal;
life_dist_object << Fit Model( ... );
life_dist_object << Fit Normal;
life_dist_object << Fit SEV;
life_dist_object << Fit TH Frechet;
life_dist_object << Fit TH Loglogistic;
life_dist_object << Fit TH Lognormal;
life_dist_object << Fit TH Weibull;
life_dist_object << Fit Weibull;
life_dist_object << Fit ZI Frechet;
life_dist_object << Fit ZI Loglogistic;
life_dist_object << Fit ZI Lognormal;
life_dist_object << Fit ZI Weibull;
life_dist_object << Get Causes;
life_dist_object << Get Estimates;
life_dist_object << Get Formula;
life_dist_object << Get Life Distribution( i );
life_dist_object << Get Model Specification;
life_dist_object << Get Results;
```

```
life_dist_object << Hazard( number );
life_dist_object << Mean Remaining Life( Compute( vector ) );
life_dist_object << Omit( i, 0|1 );
life_dist_object << Probability( number );
life_dist_object << Quantile( number );
life_dist_object << Rejection Sample Maximum Trials( number );
life_dist_object << Select Distribution( ... );
life_dist_object << Select Scale( distribution );
life_dist_object << Set Scale( distribution );
life_dist_object << Subdistribution( ... );
life_dist_object << Suppress Plot( plot_name );
```

Make Validation Column Script-Only Messages

When you launch the Make Validation Column platform via JSL, you can use the following arguments that are available only in JSL:

```
Test Set( number )
Training Set( number )
Validation Set( number )
```

MaxDiff Script-Only Messages

When you launch the MaxDiff platform via JSL, you can set the acceptable criterion for parameter convergence. In addition, when using Hierarchical Bayes, you can set the number of burn-in iterations:

```
Convergence Criterion( fraction );
Number of Burn In Iterations ( number);
```

Mixture Profile Script-Only Messages

The following messages for a Mixture Profiler object are available only in JSL:

```
mixture_profiler_object << Contour Value( y1( ... ), y2( ... ), ... );
mixture_profiler_object << Get Constraints;
mixture_profiler_object << Get Factor Settings;
mixture_profiler_object << Get Factor Settings Script;
mixture_profiler_object << Get Simulator;
mixture_profiler_object << Left Factor( column );
mixture_profiler_object << Reset;
mixture_profiler_object << Right Factor( column );
mixture_profiler_object << Term Value( x1( ... ), x2( ... ), ... );
mixture_profiler_object << Top Factor( column );
```

Model Screening Script-Only Messages

When you launch the Model Screening platform via JSL, you can use the following arguments that are available only in JSL:

```
Model NParm Limit( number )
Set Probability Threshold( number )
SVM NRow Limit( number )
```

The following message for a Model Screening object is available only in JSL:

```
model_screening_object << Select Fit( ... );
```

Multivariate Script-Only Messages

The following message for a Multivariate object is available only in JSL:

```
multivariate_object << Get Correlation Matrix;
multivariate_object << Get Inv Correlation Matrix;
```

Naive Bayes Script-Only Messages

When you launch the Naive Bayes platform via JSL, you can use the following arguments that are available only in JSL:

```
Prior Bias( fraction )
Use Excluded Rows for Validation( 0|1 )
```

The following messages for a Naive Bayes object are available only in JSL:

```
naive_bayes_object << Get Measures;
naive_bayes_object << Get Probability Formulas;
```

Neural Script-Only Messages

Note: The Neural platform replaces the Neural Net platform. Scripts that use Neural Net are deprecated and might not work in the future. It is recommended to use the Neural platform for new scripts and update any old scripts using Neural Net to Neural.

When you launch the Neural platform via JSL, you can use the following arguments that are available only in JSL:

```
Multithreading( 0|1 )
N Boost( number )
Validation Method( ... )
```

The following messages for a Neural Fit object are available only in JSL:

```
neural_object << Fit( ... );
neural_object << ( Fit[n] << Get Average Absolute Error Test );
neural_object << ( Fit[n] << Get Average Absolute Error Training );
neural_object << ( Fit[n] << Get Average Absolute Error Validation );
neural_object << ( Fit[n] << Get Average Log Error Test );
neural_object << ( Fit[n] << Get Average Log Error Training );
neural_object << ( Fit[n] << Get Average Log Error Validation );
neural_object << ( Fit[n] << Get Confusion Matrix Test);
neural_object << ( Fit[n] << Get Confusion Matrix Training);
neural_object << ( Fit[n] << Get Confusion Matrix Validation);
neural_object << ( Fit[n] << Get Confusion Rates Test);
neural_object << ( Fit[n] << Get Confusion Rates Training);
neural_object << ( Fit[n] << Get Confusion Rates Validation);
neural_object << ( Fit[n] << Get Gen RSquare Test );
neural_object << ( Fit[n] << Get Gen RSquare Training );
neural_object << ( Fit[n] << Get Gen RSquare Validation );
neural_object << ( Fit[n] << Get Measures );
neural_object << ( Fit[n] << Get Misclassification Rate Test );
neural_object << ( Fit[n] << Get Misclassification Rate Training );
neural_object << ( Fit[n] << Get Misclassification Rate Validation );
neural_object << ( Fit[n] << Get MM SAS DATA Step );
neural_object << ( Fit[n] << Get NBoost );
neural_object << ( Fit[n] << Get Prediction Formula );
neural_object << ( Fit[n] << Get RMS Error Test );
neural_object << ( Fit[n] << Get RMS Error Training );
neural_object << ( Fit[n] << Get RMS Error Validation );
neural_object << ( Fit[n] << Get ROC Area Test);
neural_object << ( Fit[n] << Get ROC Area Training );
neural_object << ( Fit[n] << Get ROC Area Validation);
neural_object << ( Fit[n] << Get RSquare Test );
neural_object << ( Fit[n] << Get RSquare Training );
neural_object << ( Fit[n] << Get RSquare Validation );
neural_object << ( Fit[n] << Get SAS DATA Step );
neural_object << ( Fit[n] << Get Seconds);
```

Nonlinear Script-Only Messages

When you launch the Nonlinear platform via JSL, you can use the following arguments that are available only in JSL:

```
Second Deriv Method( 0|1 )
```

The following messages for a Nonlinear object are available only in JSL:

```
nonlinear_object << CL Alpha( number );
```

```
nonlinear_object << CL Limit( number );
nonlinear_object << Delta( number );
nonlinear_object << Finish;
nonlinear_object << Get CI;
nonlinear_object << Get Corr;
nonlinear_object << Get Cov;
nonlinear_object << Get Estimates;
nonlinear_object << Get Parameter Names;
nonlinear_object << Get SSE;
nonlinear_object << Get Std Errors;
nonlinear_object << Gradient Limit( number );
nonlinear_object << Iteraction Limit( number );
nonlinear_object << Loss is Neg LogLikelihood( 0|1 );
nonlinear_object << Numeric Chain Deriv Delta;
nonlinear_object << Obj Change Limit( number );
nonlinear_object << Relative Gradient( number );
nonlinear_object << Set Parameter( Name = Expr, ... );
nonlinear_object << Unlock Parameter( Name, ... );
```

Normal Mixtures Script-Only Messages

When you launch the Normal Mixtures platform via JSL, you can use the following arguments that are available only in JSL:

```
Mixtures N Starts( number )
Mixtures Tolerance( number )
```

The following messages for a Normal Mixtures object are available only in JSL:

```
normal_mixtures_object << Get Statistics;
```

Oneway Script-Only Messages

The following message for a Oneway object is available only in JSL:

```
oneway_object << Select Group( Xvalue );
```

Parallel Plot Script-Only Messages

The following message for a Parallel Plot object is available only in JSL:

```
parallel_plot_object << Reversed( columns );
```

Pareto Plot Script-Only Messages

The following message for a Pareto Plot object is available only in JSL:

```
No Plot( 0|1 );
```

Partial Least Squares Script-Only Messages

The following messages for a Partial Least Squares object are available only in JSL:

```
pls_object << ( Fit[ number ] << Get Measures ) );
```

Partition Script-Only Messages

When you launch the Partition platform via JSL, you can use the following arguments that are available only in JSL:

```
Initial Splits( ... )
Missing Value Order( Low( list of numeric columns ), High( list of numeric
  columns ) )
Multithreading( 0|1 )
Set Random Seed( number )
```

The following messages for a Partition object that return the requested item from the fitted model are available only in JSL:

```
partition_object << Get Average Absolute Error Test;
partition_object << Get Average Absolute Error Training;
partition_object << Get Average Absolute Error Validation;
partition_object << Get Average Log Error Test;
partition_object << Get Average Log Error Training;
partition_object << Get Average Log Error Validation;
partition_object << Get Confusion Matrix Test;
partition_object << Get Confusion Matrix Training;
partition_object << Get Confusion Matrix Validation;
partition_object << Get Confusion Rates Test;
partition_object << Get Confusion Rates Training;
partition_object << Get Confusion Rates Validation;
partition_object << Get Difference Formula;
partition_object << Get Gen RSquare Test;
partition_object << Get Gen RSquare Training;
partition_object << Get Gen RSquare Validation;
partition_object << Get Measures;
partition_object << Get Microseconds;
partition_object << Get Misclassification Rate Test;
partition_object << Get Misclassification Rate Training;
partition_object << Get Misclassification Rate Validation;
```

```
partition_object << Get MM SAS Data Step;
partition_object << Get MM Tolerant SAS Data Step;
partition_object << Get Prediction Formula;
partition_object << Get RMS Error Test;
partition_object << Get RMS Error Training;
partition_object << Get RMS Error Validation;
partition_object << Get ROC Area Test;
partition_object << Get ROC Area Training;
partition_object << Get ROC Area Validation;
partition_object << Get RSquare Test;
partition_object << Get RSquare Training;
partition_object << Get RSquare Validation;
partition_object << Get SAS Data Step;
partition_object << Get Seconds;
partition_object << Get Tolerant Prediction Formula;
partition_object << Get Tolerant SAS Data Step;
```

Principal Components Script-Only Messages

The following message for a Principal Components object is available only in JSL:

```
prin_comp_object << Save Principal Component Script( number );
```

Process Capability Script-Only Messages

The following messages for a Process Capability object are available only in JSL:

```
process_capability_object << Goal Plot( 1, Capability Lines( number ) );
process_capability_object << Capability Index Plot( 1, Capability Lines(
  number ) );
process_capability_object << Process Performance Plot( 1, Capability
  Boundary( number ) );
process_capability_object << Process Performance Plot( 1, Stability Boundary(
  number ) );
```

Process Screening Script-Only Messages

When you launch the Process Screening platform via JSL, you can use the following arguments that are available only in JSL:

```
Drift Alpha( number )
Make Detailed Shift Data( 0|1 )
Ppk Capability Boundary( number )
Remove( column )
Stability Index Boundary( number )
```

```
Stability Ratio Boundary( number )
```

The following messages for a Process Screening object are available only in JSL:

```
process_screening_object << RowStates;
process_screening_object << Select All;
```

Profiler Script-Only Messages

The following messages for a Profiler object are available only in JSL:

```
profiler_object << Colorize;
profiler_object << Converge Limit( number );
profiler_object << General Extrapolation Control Multiplier( number );
profiler_object << Get Constraints;
profiler_object << Get Desirability;
profiler_object << Get Factor Settings;
profiler_object << Get Factor Settings Script;
profiler_object << Get Main Indices;
profiler_object << Get Simulator;
profiler_object << Get Total Indices;
profiler_object << Max Cycles( number );
profiler_object << MaxIter( number );
profiler_object << Reset;
profiler_object << Term Value( factor( ... ) );
profiler_object << Trips( number )
```

Reliability Forecast Script-Only Messages

The following messages for a Reliability Forecast object are available only in JSL:

```
reliability_forecast_object << Forecast Options( ... );
reliability_forecast_object << Get Results;
```

Reliability Growth Script-Only Messages

The following messages for a Reliability Growth object are available only in JSL:

```
reliability_growth_object << Get Results;
(reliability_growth_object << Mean Time Between Failures Plot) << Options(
  Interval Size( number ) );
(reliability_growth_object << Mean Time Between Failures Plot) << Options(
  Sample MTBF Type( string ) );
```

Response Screening Script-Only Messages

The following message for a Response Screening object is available only in JSL:

```
response_screening_object << Get PValues;
```

Scatterplot 3D Script-Only Messages

The following messages for a Scatterplot 3D object are available only in JSL:

```
scatterplot_3D_object << Frame3D( Add Ellipsoid( ... ) );
scatterplot_3D_object << Frame3D( Add Markers( ... ) );
scatterplot_3D_object << Frame3D( Add Vector( ... ) );
scatterplot_3D_object << Frame3D( Get Axes );
scatterplot_3D_object << Frame3D( Get Box );
scatterplot_3D_object << Frame3D( Get Grab Handles );
scatterplot_3D_object << Frame3D( Get Graph Size );
scatterplot_3D_object << Frame3D( Get Grids );
scatterplot_3D_object << Frame3D( Get Hide Lights Border );
scatterplot_3D_object << Frame3D( get light active( number ) );
scatterplot_3D_object << Frame3D( get light color( number ) );
scatterplot_3D_object << Frame3D( get light position( number ) );
scatterplot_3D_object << Frame3D( Get Line Scale );
scatterplot_3D_object << Frame3D( Get Marker Quality );
scatterplot_3D_object << Frame3D( Get Marker Scale );
scatterplot_3D_object << Frame3D( Get Marker Transparency );
scatterplot_3D_object << Frame3D( Get Rotation );
scatterplot_3D_object << Frame3D( Get Text Scale );
scatterplot_3D_object << Frame3D( Get View Ortho );
scatterplot_3D_object << Frame3D( Get View Perspective );
scatterplot_3D_object << Frame3D( Get View Zoom );
scatterplot_3D_object << Frame3D( Get Wall Color );
scatterplot_3D_object << Frame3D( Get Walls );
scatterplot_3D_object << Frame3D( Get X Axis Color );
scatterplot_3D_object << Frame3D( Get X Axis Label );
scatterplot_3D_object << Frame3D( Get Y Axis Color );
scatterplot_3D_object << Frame3D( Get Y Axis Label );
scatterplot_3D_object << Frame3D( Get Z Axis Color );
scatterplot_3D_object << Frame3D( Get Z Axis Label );
scatterplot_3D_object << Frame3D( Set Axes( 0|1 ) );
scatterplot_3D_object << Frame3D( Set Box( 0|1 ) );
scatterplot_3D_object << Frame3D( Set Graph Size( x, y ) );
scatterplot_3D_object << Frame3D( Set Grids( 0|1 ) );
scatterplot_3D_object << Frame3D( Set Hide Lights Border( 0|1 ) );
scatterplot_3D_object << Frame3D( set light active( ... ) );
scatterplot_3D_object << Frame3D( set light color( ... ) );
```

```
scatterplot_3D_object << Frame3D( set light position( ... ) );
scatterplot_3D_object << Frame3D( Set Line Scale( number ) );
scatterplot_3D_object << Frame3D( Set Marker Quality( number ) );
scatterplot_3D_object << Frame3D( Set Marker Scale( number ) );
scatterplot_3D_object << Frame3D( Set Marker Transparency( number ) );
scatterplot_3D_object << Frame3D( Set Oscillation( ... ) );
scatterplot_3D_object << Frame3D( Set Rotation( ... ) );
scatterplot_3D_object << Frame3D( Set Spin( ... ) );
scatterplot_3D_object << Frame3D( Set Text Scale( number ) );
scatterplot_3D_object << Frame3D( Set View Ortho( 0|1 ) );
scatterplot_3D_object << Frame3D( Set View Perspective( number ) );
scatterplot_3D_object << Frame3D( Set View Zoom( number ) );
scatterplot_3D_object << Frame3D( Set Wall Color( number ) );
scatterplot_3D_object << Frame3D( Set Walls( 0|1 ) );
scatterplot_3D_object << Frame3D( Set X Axis Color( color ) );
scatterplot_3D_object << Frame3D( Set X Axis Label( string ) );
scatterplot_3D_object << Frame3D( Set Y Axis Color( color ) );
scatterplot_3D_object << Frame3D( Set Y Axis Label( string ) );
scatterplot_3D_object << Frame3D( Set Z Axis Color( color ) );
scatterplot_3D_object << Frame3D( Set Z Axis Label( string ) );
scatterplot_3D_object << Frame3D( XAxis( ... ) );
scatterplot_3D_object << Frame3D( YAxis( ... ) );
scatterplot_3D_object << Frame3D( Z Axis( ... ) );
scatterplot_3D_object << Nonpar Density Contour Settings( ... );
```

Structural Equation Models Script-Only Messages

The following messages for a Structural Equation Models object enable you to specify aspects of the model specification, the fitted model, and the path diagram, respectively. These messages are available only in JSL.

```
structural_equation_models_object << Model Specification( ... );
structural_equation_models_object << Fit( ... );
structural_equation_models_object << Path Diagram Properties( ... );
```

The following messages are available within a Model Specification() or Fit() message sent to a Structural Equation Models object.

```
structural_equation_models_object << Fit( Covariances( ... ) );
structural_equation_models_object << Fit( Loadings( ... ) );
structural_equation_models_object << Fit( Means( ... ) );
structural_equation_models_object << Fit( Model Name( string ) );
structural_equation_models_object << Fit( New Latent( string ) );
structural_equation_models_object << Fit( Regressions( ... ) );
structural_equation_models_object << Fit( Variances( ... ) );
```

The following messages are available within a Path Diagram Properties message sent to a Structural Equation Models object.

```
SEM_object << Path Diagram Properties( Constant Border Color( string ) );
SEM_object << Path Diagram Properties( Constant Fill Color( string ) );
SEM_object << Path Diagram Properties( Constant Font( font ) );
SEM_object << Path Diagram Properties( Constant Height( number ) );
SEM_object << Path Diagram Properties( Constant Shape( ... ) );
SEM_object << Path Diagram Properties( Constant Size Option( string ) );
SEM_object << Path Diagram Properties( Constant Text Color( string ) );
SEM_object << Path Diagram Properties( Constant Width( number ) );
SEM_object << Path Diagram Properties( Diagram Size( {x, y} ) );
SEM_object << Path Diagram Properties( Latent Border Color( string ) );
SEM_object << Path Diagram Properties( Latent Fill Color( string ) );
SEM_object << Path Diagram Properties( Latent Font( font ) );
SEM_object << Path Diagram Properties( Latent Height( number ) );
SEM_object << Path Diagram Properties( Latent Shape( ... ) );
SEM_object << Path Diagram Properties( Latent Size Option( string ) );
SEM_object << Path Diagram Properties( Latent Text Color( string ) );
SEM_object << Path Diagram Properties( Latent Width( number ) );
SEM_object << Path Diagram Properties( Lock Diagram( 0|1 ) );
SEM_object << Path Diagram Properties( Manifest Border Color( string ) );
SEM_object << Path Diagram Properties( Manifest Fill Color( string ) );
SEM_object << Path Diagram Properties( Manifest Font( font ) );
SEM_object << Path Diagram Properties( Manifest Height( number ) );
SEM_object << Path Diagram Properties( Manifest Shape( ... ) );
SEM_object << Path Diagram Properties( Manifest Size Option( string ) );
SEM_object << Path Diagram Properties( Manifest Text Color( string ) );
SEM_object << Path Diagram Properties( Manifest Width( number ) );
SEM_object << Path Diagram Properties( Path Styles( ... ) );
SEM_object << Path Diagram Properties( Paths Alpha Level( number ) );
SEM_object << Path Diagram Properties( Paths Color( string ) );
SEM_object << Path Diagram Properties( Paths Font( font ) );
SEM_object << Path Diagram Properties( Paths Opacity( number ) );
SEM_object << Path Diagram Properties( Paths Thickness( number ) );
SEM_object << Path Diagram Properties( Place Nodes( ... ) );
SEM_object << Path Diagram Properties( Rotate Loops( ... ) );
```

Support Vector Machines Script-Only Messages

The following messages for a Support Vector Machines object are available only in JSL:

```
svm_object << Get Measures;
svm_object << Get Prediction Formula;
```

Surface Plot Script-Only Messages

The following messages for a Surface Plot object are available only in JSL:

```
surface_plot_object << Frame3D( Get Axes );
surface_plot_object << Frame3D( Get Box );
surface_plot_object << Frame3D( Get Grab Handles );
surface_plot_object << Frame3D( Get Graph Size );
surface_plot_object << Frame3D( Get Grids );
surface_plot_object << Frame3D( Get Hide Lights Border );
surface_plot_object << Frame3D( get light active );
surface_plot_object << Frame3D( get light color );
surface_plot_object << Frame3D( get light position );
surface_plot_object << Frame3D( Get Line Scale );
surface_plot_object << Frame3D( Get Marker Quality );
surface_plot_object << Frame3D( Get Marker Scale );
surface_plot_object << Frame3D( Get Marker Transparency );
surface_plot_object << Frame3D( Get Rotation );
surface_plot_object << Frame3D( Get Text Scale );
surface_plot_object << Frame3D( Get View Ortho );
surface_plot_object << Frame3D( Get View Persepctive );
surface_plot_object << Frame3D( Get View Zoom );
surface_plot_object << Frame3D( Get Wall Color );
surface_plot_object << Frame3D( Get Walls );
surface_plot_object << Frame3D( Get X Axis Color );
surface_plot_object << Frame3D( Get X Axis Label );
surface_plot_object << Frame3D( Get Y Axis Color );
surface_plot_object << Frame3D( Get Y Axis Label );
surface_plot_object << Frame3D( Get Z Axis Color );
surface_plot_object << Frame3D( Get Z Axis Label );
surface_plot_object << Frame3D( Set Axes( 0|1 ) );
surface_plot_object << Frame3D( Set Box( 0|1 ) );
surface_plot_object << Frame3D( Set Graph Size( x, y ) );
surface_plot_object << Frame3D( Set Grids( 0|1 ) );
surface_plot_object << Frame3D( Set Hide Lights Border( 0|1 ) );
surface_plot_object << Frame3D( set light active( light number, 0|1 ) );
surface_plot_object << Frame3D( set light color( light number, red value,
   green value, blue value ) );
surface_plot_object << Frame3D( set light position( light number, X, Y, Z ) );
surface_plot_object << Frame3D( Set Line Scale( number ) );
surface_plot_object << Frame3D( Set Marker Quality( number ) );
surface_plot_object << Frame3D( Set Marker Scale( number ) );
surface_plot_object << Frame3D( Set Marker Transparency( fraction ) );
surface_plot_object << Frame3D( Set Oscillation( X, Y, Z, duration ) );
surface_plot_object << Frame3D( Set Rotation( X, Y, Z ) );
surface_plot_object << Frame3D( Set Spin( dx, dy, sx, sy ) );
```

```
surface_plot_object << Frame3D( Set Text Scale( number ) );
surface_plot_object << Frame3D( Set View Ortho( 0|1 ) );
surface_plot_object << Frame3D( Set View Perspective( fraction ) );
surface_plot_object << Frame3D( Set View Zoom( number ) );
surface_plot_object << Frame3D( Set Wall Color( number ) );
surface_plot_object << Frame3D( Set Walls( 0|1 ) );
surface_plot_object << Frame3D( Set X Axis Color( color ) );
surface_plot_object << Frame3D( Set X Axis Label( string ) );
surface_plot_object << Frame3D( Set Y Axis Color( color ) );
surface_plot_object << Frame3D( Set Y Axis Label( string ) );
surface_plot_object << Frame3D( Set Z Axis Color( color ) );
surface_plot_object << Frame3D( Set Z Axis Label( string ) );
surface_plot_object << Frame3D( XAxis( Min( number ), Max( number ), Inc(
  number ), Format( ) );
surface_plot_object << Frame3D( YAxis( Min( number ), Max( number ), Inc(
  number ), Format( ) );
surface_plot_object << Frame3D( ZAxis( Min( number ), Max( number ), Inc(
  number ), Format( ) );
surface_plot_object << Clip Sheet( 0|1 );
surface_plot_object << Data points Color( color );
surface_plot_object << Datapoints Choice( Off | Points | Needles | Mesh |
  Surface );
surface_plot_object << Dependent Variables Points( 0|1 );
surface_plot_object << Dependent Variables Response Grid( 0|1 );
surface_plot_object << Equation( column, <column>, <column>, <column> );
surface_plot_object << Frame3D( Scatterplot 3D options );
surface_plot_object << Iso Value( id, value );
surface_plot_object << Mode( "Sheet, points" | "Isosurface" | "Density grid" )
surface_plot_object << Resolution( number );
surface_plot_object << Response( column, ... );
surface_plot_object << Response Column Color Theme( Green to Black to Red |
  Green to White to Red | White to Black | Blue to Gray to Red | Blue to Green
  to Red | Spectral | Jet | White to Blue | White to Green | White to Red );
surface_plot_object << Response Column Fill( Solid | Continuous | Gradients |
  Discrete Gradients );
surface_plot_object << Response Column Gradient Lines( 0|1 );
surface_plot_object << Response Column Gradients( number );
surface_plot_object << Set Z Variable( column );
surface_plot_object << SetVariableAxis( column, <Current Value( number )>,
  <Axis Data( axis options )> );
surface_plot_object << SetXVariable( column );
surface_plot_object << SetYVariable( column );
surface_plot_object << SetZAxis( column, <Current Value( number )>, <Axis
  Data( axis options )> );
surface_plot_object << Surface Alpha( number );
```

```
surface_plot_object << Surface Color Method( "Solid" | formula, <"Solid" |
  formula>, <"Solid" | formula>, <"Solid" | formula> );
surface_plot_object << Surface Gradient Type( Solid | Continuous Gradients |
  Discrete Gradients );
surface_plot_object << Surface Gradients( number );
surface_plot_object << Surface Selector( 0|1 );
surface_plot_object << X Grid( 0|1 );
surface_plot_object << X Resolution( number );
surface_plot_object << X Rotate( degrees );
surface_plot_object << Y Grid( 0|1 );
surface_plot_object << Y Resolution( number );
surface_plot_object << Y Rotate( degrees );
surface_plot_object << Z Grid( 0|1 );
surface_plot_object << Z Resolution( number );
surface_plot_object << Z Rotate( degrees );
```

The following messages for a Surface Plot object act on one of the four surfaces or dependent variables and therefore need a number, <n>, where n=1, 2, 3, or 4.

```
surface_plot_object << Clip Sheet<n>( 0|1 );
surface_plot_object << Contour Color<n>( color );
surface_plot_object << Data points Color<n>( color );
surface_plot_object << Datapoints Choice<n>( Off | Points | Needles | Mesh |
  Surface );
surface_plot_object << Mesh Color<n>( color );
surface_plot_object << Response Column Color Theme<n>( Green to Black to Red |
  Green to White to Red | White to Black | Blue to Gray to Red | Blue to Green
  to Red | Spectral | Jet | White to Blue | White to Green | White to Red );
surface_plot_object << Response Column Fill<n>( Solid | Continuous | Gradients
  | Discrete Gradients );
surface_plot_object << Response Column Gradient Lines<n>( 0|1 );
surface_plot_object << Response Column Gradients<n>( number );
surface_plot_object << Show Contour<n>( Off | Below | Above | On Surface );
surface_plot_object << Show Mesh<n>( Off | X | Y | X and Y );
surface_plot_object << Show Surface<n>( Off | Both Sides | Above | Below Only
  );
surface_plot_object << Surface Alpha<n>( number );
surface_plot_object << Surface Color Range<n>( Data | Axis );
surface_plot_object << Surface Color Theme<n>( Green to Black to Red | Green
  to White to Red | White to Black | Blue to Gray to Red | Blue to Green to
  Red | Spectral | Jet | White to Blue | White to Green | White to Red );
surface_plot_object << Surface Color<n>( color );
surface_plot_object << Surface Gradient Type<n>( Solid | Continuous Gradients
  | Discrete Gradients );
surface_plot_object << Surface Gradients<n>( number );
surface_plot_object << Surface Lighting<n>( None | Low Reflection | Normal );
```

Tabulate Script-Only Messages

The following messages for a Tabulate object are available only in JSL:

```
tabulate_object << Add Table( ... );
MC: I decided to ignore the following that must be specified within a Column
  Table message OR a Row Table message OR within Add Table or Modify Table:
  Aggregate Statistics, Analysis Columns, Columns by Categories, Groupign
  Columns, Statistics,
tabulate_object << Change Item Label( ... );
tabulate_object << Display Column Width( ... );
tabulate_object << Modify Table( ... );
tabulate_object << Order by Count( ... );
tabulate_object << Plot Scale( min, max );
tabulate_object << Remove Column Label( Grouping Columns( column ) );
tabulate_object << Retore Column Label( Grouping Columns( column ) );
tabulate_object << Set Format( ... );
tabulate_object << Test Build( Sample Size( number ) );
tabulate_object << Test Data View;
tabulate_object << Uniform plot scale( 0|1 );
```

Ternary Plot Script-Only Messages

The following message for a Ternary Plot object is available only in JSL:

```
ternary_plot_object << Contour Value( [c1, c2, c3, c4, c5] );
```

Text Explorer Script-Only Messages

The following messages for a Text Explorer object are available only in JSL:

```
text_explorer_object << Add Delimiters( string )
text_explorer_object << Add Phrases( list )
text_explorer_object << Add Phrase Exceptions( list )
text_explorer_object << Add Stem Exceptions( list )
text_explorer_object << Add Stem Overrides( list )
text_explorer_object << Add Stop Words( list )
text_explorer_object << Add Stop Word Exceptions( list )
text_explorer_object << Add Recodes( list )
text_explorer_object << Add Recode Exceptions( list )
text_explorer_object << Cloud Width( pixels )
text_explorer_object << Custom Stemmer( function( {string, dot}, ... ) )
text_explorer_object << Latent Class Analysis( Set Random Seed( number ) );
text_explorer_object << Minimum Frequency for Phrase( n )
text_explorer_object << Phrases Alphabetical( Boolean )
text_explorer_object << Save Regex Column( string, <"From Source"> )
```

```
text_explorer_object << Set Delimiters( string )
text_explorer_object << Set Regex( ... )
text_explorer_object << Terms Alphabetical( Boolean )
```

The following messages for a Sentiment Analysis object in the Text Explorer platform are available only in JSL:

```
sentiment_analysis_object << Add Feature Words
sentiment_analysis_object << Add Intensifier Exception Words
sentiment_analysis_object << Add Intensifier Words
sentiment_analysis_object << Add Negation Exception Words
sentiment_analysis_object << Add Negation Words
sentiment_analysis_object << Add Sentiment Exception Words
sentiment_analysis_object << Add Sentiment Words
```

Time Series Script-Only Messages

The following messages for a Time Series object are available only in JSL:

```
time_series_object << Fit Recommended ETS( ... );
time_series_object << Get Model Specs;
time_series_object << Get Models;
time_series_object << Input Series( ... );
time_series_object << Keep Best Models( "AIC" | "SBC" );
time_series_object << Simple Moving Average Centering Method( "No Centering" |
   "Centered" | "Centered and Double Smoothed for Even Number of Terms" );
```

The following messages for a Simple Moving Average object in the Time Series platform are available only in JSL:

```
sma_object << Get Results;
sma_object << Remove Model( ... );
```

Uplift Script-Only Messages

When you launch the Uplift platform via JSL, you can use the following arguments that are available only in JSL:

```
Multithreading( 0|1 )
Set Random Seed( number )
```

The following messages for an Uplift object are available only in JSL:

```
uplift_object << Get Average Absolute Error Test;
uplift_object << Get Average Absolute Error Training;
uplift_object << Get Average Absolute Error Validation;
uplift_object << Get Average Log Error Test;
uplift_object << Get Average Log Error Training;
```

```
uplift_object << Get Average Log Error Validation;
uplift_object << Get Confusion Matrix Test;
uplift_object << Get Confusion Matrix Training;
uplift_object << Get Confusion Matrix Validation;
uplift_object << Get Confusion Rates Test;
uplift_object << Get Confusion Rates Training;
uplift_object << Get Confusion Rates Validation;
uplift_object << Get Difference Formula;
uplift_object << Get Gen RSquare Test;
uplift_object << Get Gen RSquare Training;
uplift_object << Get Gen RSquare Validation;
uplift_object << Get Measures;
uplift_object << Get Microseconds;
uplift_object << Get Misclassification Rate Test;
uplift_object << Get Misclassification Rate Training;
uplift_object << Get Misclassification Rate Validation;
uplift_object << Get Prediction Formula;
uplift_object << Get RMS Error Test;
uplift_object << Get RMS Error Training;
uplift_object << Get RMS Error Validation;
uplift_object << Get ROC Area Test;
uplift_object << Get ROC Area Training;
uplift_object << Get ROC Area Validation;
uplift_object << Get RSquare Test;
uplift_object << Get RSquare Training;
uplift_object << Get RSquare Validation;
uplift_object << Get Seconds;
uplift_object << Get Tolerant Prediction Formula;
```

Variability Chart Script-Only Messages

The following messages for a Variability Chart object are available only in JSL:

```
variability_chart_object << Analysis Type( string );
variability_chart_object << Conv Limit( number );
variability_chart_object << Max Iter( number );
variability_chart_object << Model( string );
variability_chart_object << Number Function Evals( number );
variability_chart_object << "Reduced Gauge R&R Report"n( 0|1 );
variability_chart_object << Set Random Seed( number );
```

Chapter 11

Display Trees

Create and Interact with Windows

After you learn the basics of scripting, you're ready to manipulate JMP reports and create new JMP windows. This chapter focuses on the following topics:

- navigating JMP display trees to manipulate items in report windows
- building new windows with custom results
- building modal windows
- interacting with the script editor
- converting deprecated `Dialog()` scripts to `New Window()` scripts

Display Tree Compatibility Issues

JMP platforms are generally stable from release to release. However, sometimes the structure of reports changes in major releases. When this happens, old scripts can fail because of the new structure. To avoid the continual updating of old scripts, write scripts so that they avoid dependence on the report structure. Using a string to index a box, followed by a numbered subscript to pick a specific child box, is most reliable across releases. See "Create a Reference by Subscripting". See "Compatibility Notes" for compatibility issues in JMP 16.

Note: This chapter shows how to create and use the most common display boxes. All display boxes are documented in the JMP Scripting Index, which is available in the Help menu.

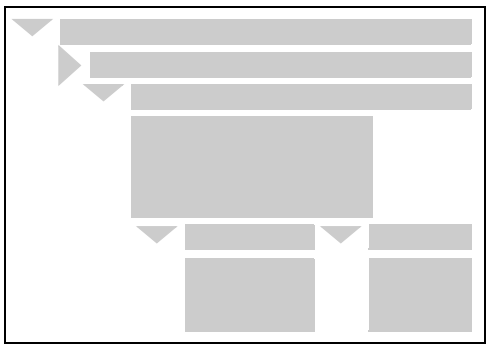
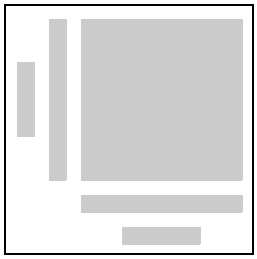
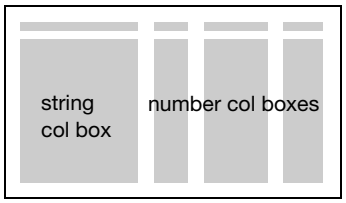
Contents

Navigate JMP Reports 545
Examples of Common Display Boxes 546
View the Display Tree 547
Show Display Box Properties 549
Display Box Object References 550
Send Messages to Display Boxes 555
Example of Creating a Report 562
Construct Custom Windows 565
Example of Creating a Graph Box 565
Extract Values from a Window 566
Constructors for New Windows 568
Interact with a Closing Window 608
Update an Existing Display 610
Set Function and Set Script 614
Get and Set Selected Values of Display Elements That Return Lists 616
Send Messages to Constructed Displays 618
Construct Display Boxes That Contain Platforms 619
Counting the Number of Child Boxes 621
Examples of Creating a Dashboard from Two Reports 622
Example of Creating a Cluster Platform Launch Window 628
Example of Constructing a Custom Platform 631
Example of Specifying Text and Background Colors 634
Modal Windows 635
Construct a Modal Window 635
Construct a Column Dialog 637
Comparison of Column Dialog and New Window 639
Constructors for Column Dialogs 640
Convert Deprecated Dialog to New Window 643
New Window Example 643
Example of Deprecated Dialog 644
Differences Between New Window and the Deprecated Dialog 646
Optional Scripts in New Window 653
Scripting the Script Editor 654
Technical Details 655
Write Tab Box and Tab Page Box Scripts 655

Navigate JMP Reports

As shown in Table 11.1, reports in JMP are built in a hierarchical manner by nesting and concatenating different types of rectangular boxes called *display boxes*. This section shows you how to navigate reports, extract data from them, and customize them.

Table 11.1 Common Display Boxes in JMP Reports

`Outline Box()` creates an outline hierarchy.	
`Picture Box()` combines axes, frames, and labels to make graphs.	
`Table Box()` is a special `H List Box()` whose elements are string and number columns.	

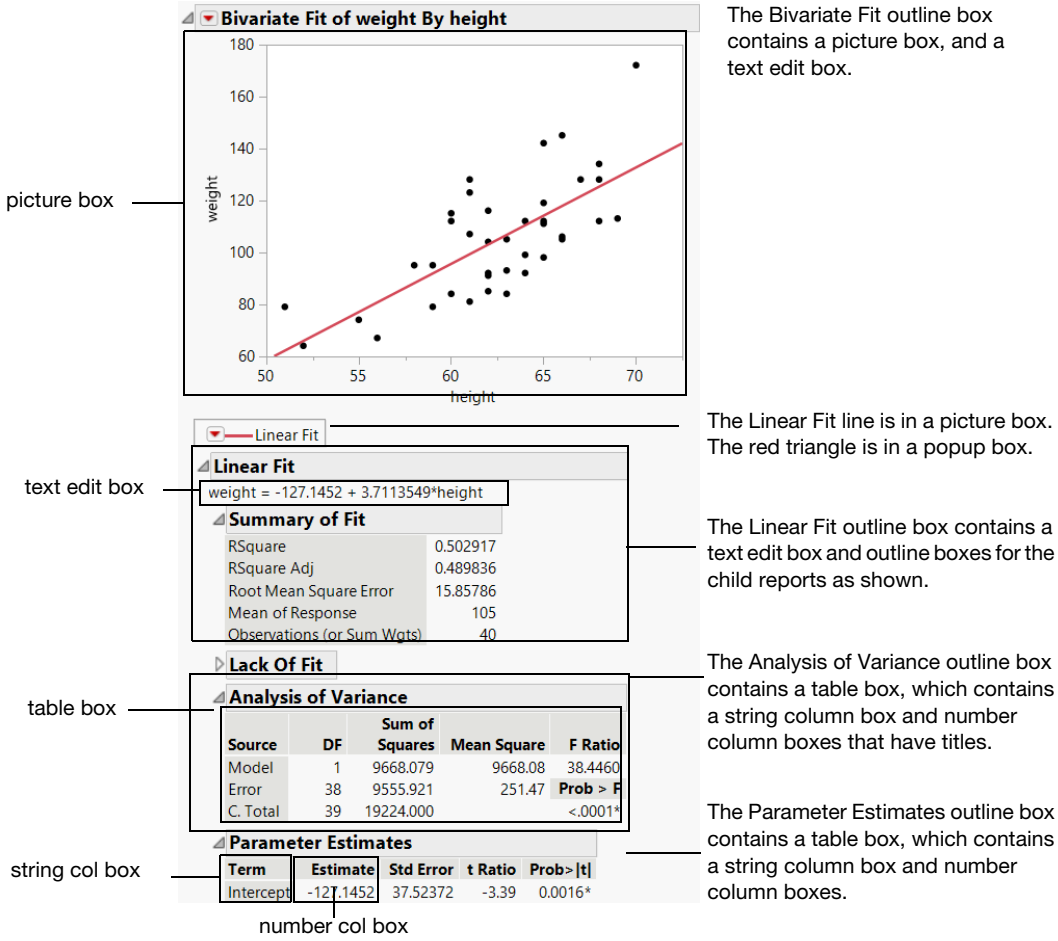
The hierarchical (or parent-child) relationships between display boxes help determine the sizing. The outermost display box is the parent; the inner display boxes are children. When a report is displayed, the parent display box detects the size of the last child box (the leaf) and then detects the size of the child box that precedes the last child box. The size is detected for each box on the way back up the tree. Then the positioning of the boxes takes places from the top down so that the boxes can be shown in an orderly arrangement.

The hierarchical (or parent-child) relationships between display boxes form a tree of display boxes, called the *display tree*. For (parent) boxes that contain other (child) boxes, the parent box's size is determined by the sizes of its child boxes rather than some fixed size, which keeps the whole display resizable. Similarly, positions of boxes are determined by positions of the boxes before them rather than using fixed positions. In tree terminology, the sizing is bottom-up and the positioning is top-down.

Examples of Common Display Boxes

Figure 11.1 shows a Bivariate report that consists of several common display boxes.

Figure 11.1 Display Boxes in a Report



Run the following script to create the report:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( :weight ), X( :height ), Fit Line() );
```

View the Display Tree

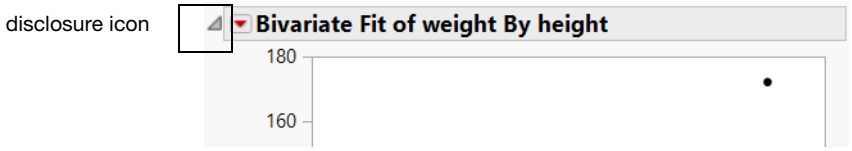
View the report structure to see how display boxes are arranged in the display tree.

1. Run the following script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate(
   Y( :weight ),
   X( :height ),
   Fit Mean( {Line Color( {57, 177, 67} )} )
);
```

2. Right-click the top disclosure icon and select **Edit > Show Tree Structure**.

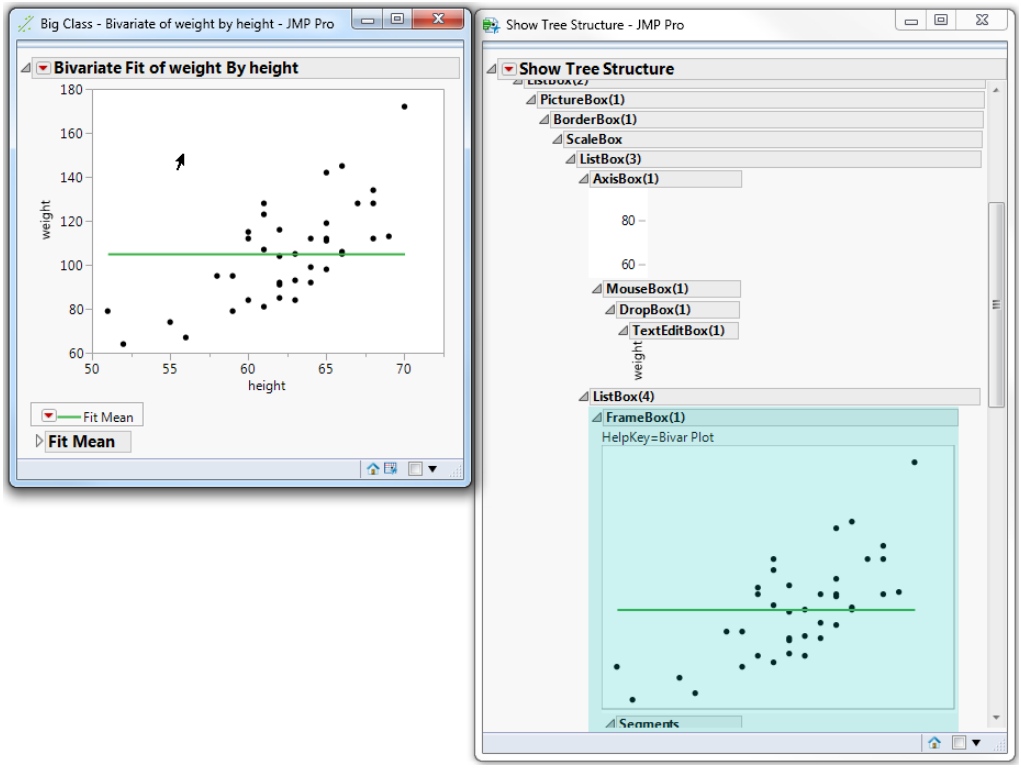
Figure 11.2 Top Disclosure Icon



In the Show Tree Structure window, you see the display box structure of the report as nested outline boxes. Each outline box corresponds to a display box in the original report.

3. Click the Bivariate graph. In Figure 11.3, the frame box that contains the graph is highlighted.

Figure 11.3 Show Tree Structure

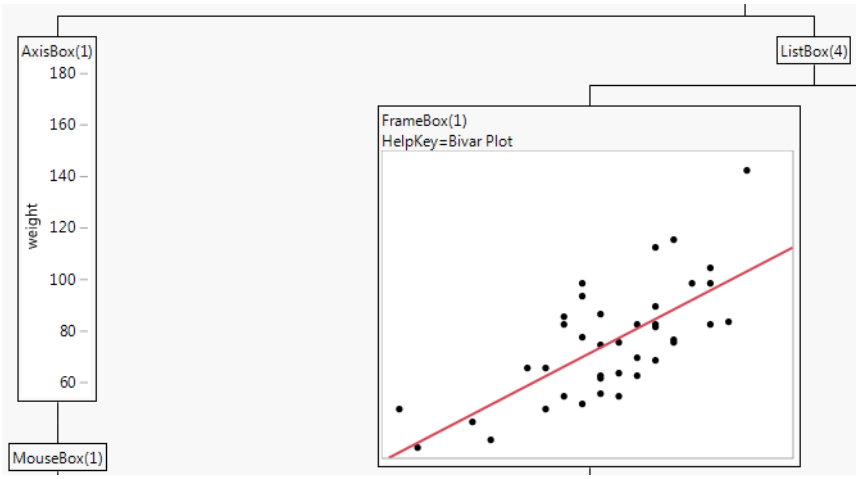


You can also obtain the tree structure through a script. Send the `Show Tree Structure()` message to any report. Or, send the message to a piece of the report (any display box object) to see the tree structure for just that part of the report.

The newer JMP tree structure provides a flat view of the display box tree rather than an outline view. To view the classic structure, press Shift before you right-click the disclosure icon and then select **Edit > Show Tree Structure**. Note the following differences between the newer and classic tree structure views:

- The classic tree structure is not linked to the original report; you cannot highlight specific portions of the tree.
- Siblings appear to the left or right of each other in the classic tree structure. In the newer tree structure, siblings are above or below each other.

Figure 11.4 Partial View of the Classic Tree Structure

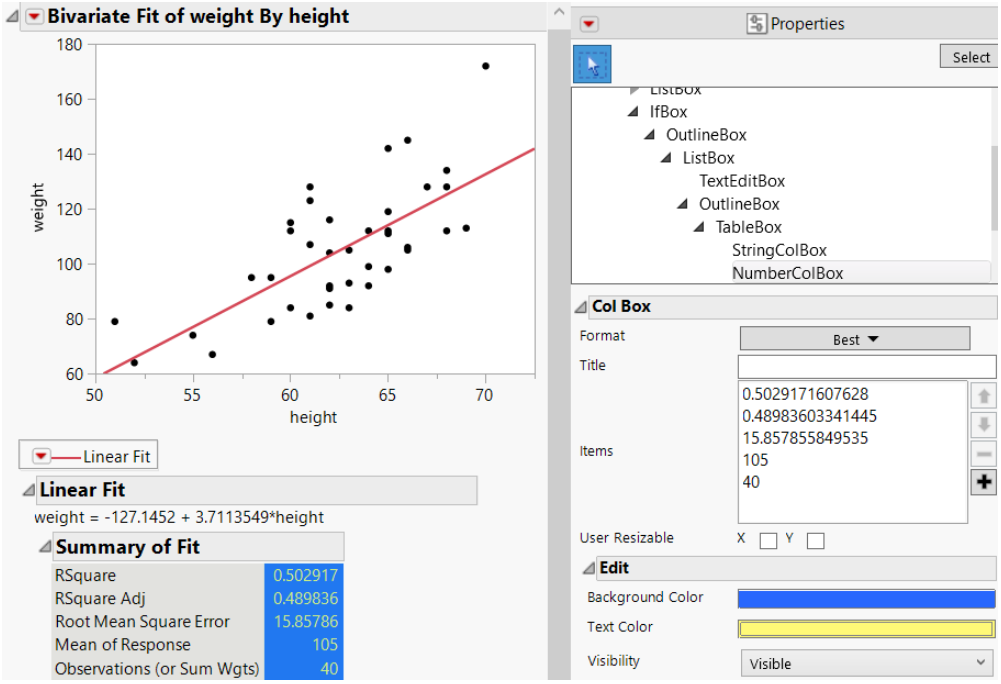


Show Display Box Properties

There is an easy way to format reports before you save them to a script. You can edit the display box properties such as color and position in the Show Tree Structure window. In the report window, select **Edit > Properties** (Windows) or **View > Show Properties** (macOS). Click the node that you want to edit.

In Figure 11.5, the Summary of Fit statistics were selected below the graph. The Background Color and Text Color were then changed in the Properties. These changes are preserved when you save the report as a script or journal the report.

Figure 11.5 Example of Showing Properties for a Plot



Display Box Object References

When you launch a platform, the results are in two places: a visible report, made of display boxes, and an invisible object, called a platform, that accepts commands (or messages).

One of the messages that you can send a platform is the `Report` message. The `Report` message retrieves the display box that holds the report.

The red triangle menu at the top of the report is connected to the platform object and is the way you interactively send messages to the platform.

To fit the mean in the Bivariate platform, assign a reference to the Bivariate platform.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( // biv references the analysis layer
   X( height ),
   Y( weight ),
   Fit Mean,
   Fit Polynomial( 4 )
);
```

To get a reference to the display box that represents the view of the platform, use the `Report` message. For example, add this expression to the preceding script:

```
rbiv = biv << Report; // rbiv references the report layer
```

Now you can manipulate the display boxes in the report layer as described in the following sections.

See What You Can Do with a Display Box

The following example creates a Bivariate report and stores its reference in the `rbiv` report object. It then puts the `rbiv` report object in a frame box and assigns the `fbx` variable.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( // biv references the analysis layer
   Y( :weight ),
   X( :height ),
   Fit Mean( {Line Color( {57, 177, 67} )} )
);
rbiv = biv << Report; // assign rbiv to the report layer
fbx = rbiv[Frame Box( 1 )];
```

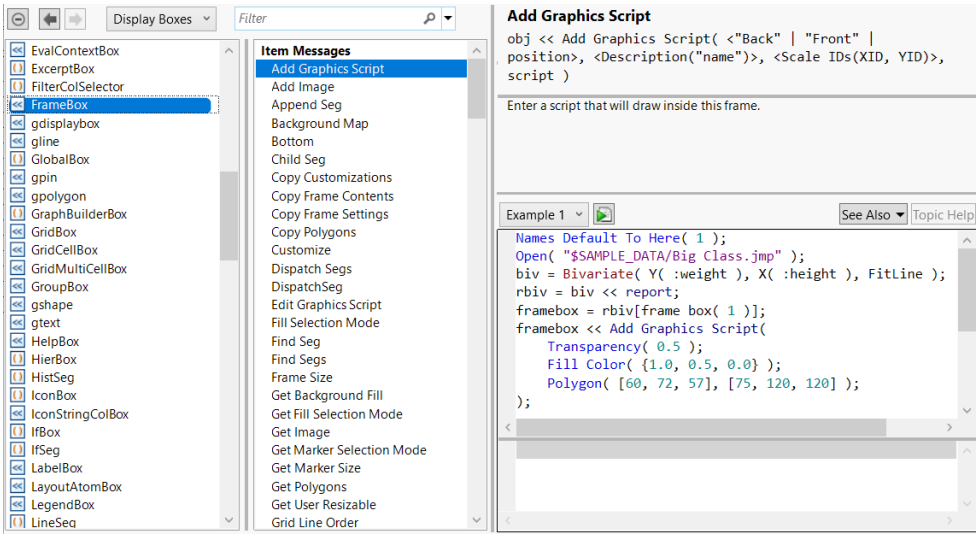
After fbx has been defined, you can send it a message.

```
fbx << Set Background Color( blue ); // colors the frame box in blue
```

The `Set Background Color` message works the same as right-clicking in the graph and selecting **Background Color > Blue**.

To find out which messages you can send to a frame box, select **Help > Scripting Index** and search for “frame box”.

Figure 11.6 Scripting Index Display Boxes List



Running the `Show Properties()` function is an alternative to using the Scripting Index. The messages for the specified display box are printed to the log.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( :weight ), X( :height ) );
Show Properties( biv );
   Show Points [Boolean] [Default On]
   Histogram Borders [Boolean]
   Fit Mean [Action](Fits a flat line at the mean.)
   ...
```

Many messages are the same as items in the pop-up menu for a given object. The next section, "Construct Custom Windows", discusses messages for common display boxes in more detail.

Create a Reference by Subscripting

Use the subscript operator to navigate to a specific display box in the report. The best practice is to refer to the display box with an index number. The following expression identifies the second outline box in the display tree:

```
var = Outline Box[2];
```

Another way to navigate a report is to search for a text string. The following expression identifies the report that has the specified text string:

```
var = report[text];
```

This expression finds the display box in *rpt* that has the title *text*. Note that *text* must be the complete title, not just a substring of the title. The "*text*" argument can also be any expression that evaluates to a text string. This approach is less reliable than referring to the display box with a subscript because the string in a report can easily change.

Typically, you want to assign the identified part of the report to a variable so that you can send messages to it easily. For example, to find the Analysis of Variance outline box in the Bivariate report, use this expression:

```
var = rbiv["Analysis of Variance"];
```

The sections below describe other methods for using a subscript operator to locate display boxes.

Wildcard String

You can use a wildcard character (such as "?") with a substring to match the rest of the outline title. The wildcard character "?" represents places in the search string where you want to match any sequence of any characters. The following script searches for a string that ends with "Mean", finds "Fit Mean", and then opens the Fit Mean outline:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate(
   Y( :weight ),
   X( :height ),
   Fit Mean( {Line Color( {57, 177, 67} )} ),
   Fit Polynomial( 4 )
);
rbiv = biv << Report; // return a reference to the Bivariate report
out1 = rbiv["? Mean"]; // find "Fit Mean" in the Bivariate report
out1 << Close( 0 ); // open the Fit Mean outline
```

Box Type and Wildcard String

Another method for locating display boxes is to include the display box and wildcard search string in the substring operator.

```
var = rbiv[Display Box( "?text" )];
```

This expression finds the outline box that contains the specified text string in the `rbiv` report. The `out1` variable assigns a reference to the outline box.

```
out1 = rbiv[Outline Box( "? Mean" )];
```

Box Type and Index Number

Another method for locating display boxes uses the *boxType* and an index number, *n*.

```
rpt[Display Box ( n )];
```

The expression above finds the *n*th of the specified display box of the type *boxType*. For example, this expression finds the first outline box in the `rbiv` report and assigns it to the `out1` variable:

```
out1 = rbiv[Outline Box( 1 )];
```

This method is less reliable across releases because the number of a display box might change.

Tip: To determine the index number of a display box, view the tree structure as described in "View the Display Tree".

Nested Display Boxes

To reference a display box that appears several layers down in a report, use this format:

```
var = rpt[arg1][arg2][arg3][...];
```

The expression above matches the last argument to a display box that is *contained by* the next to last argument's outline node, which is in turn contained by the argument before the next to last, and so on. In other words, it is a way of digging down into the hierarchy of an outline tree to identify a nested display box.

If you want to identify an item nested several layers down in an outline box, use a subscript with more than one argument of any of the above types. After locating the first item, JMP looks *inside* that item for the next, and so on.

```
rpt[arg1][arg2][arg3] // finds the first item then the next starting after
  that location, and so on
rpt[arg1, arg2, arg3] // different syntax, same result
```

Note that you can string together subscripts that have text or index numbers. The expression below finds the "Parameter Estimate" inside the outline node "Polynomial Fit Degree=4" and selects the third display box in the first table of the outline node.

```
out = rbiv["Polynomial Fit Degree=4"]["Parameter Estimates"][1][3] << Select;
```

"Example of Creating a Report" shows how to extract parameter estimates from the report.

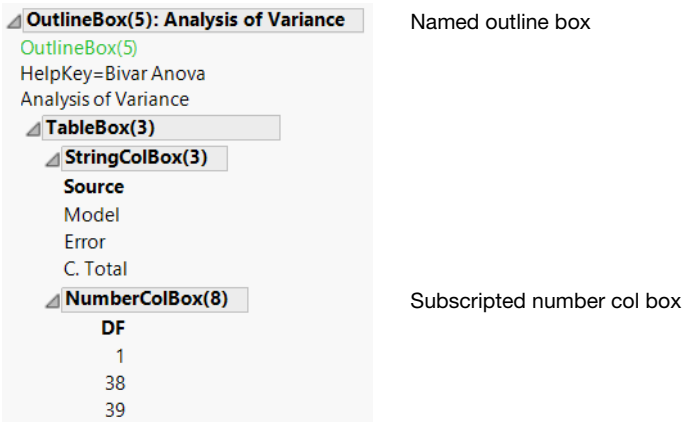
Subscripting Display Boxes for Compatibility

JMP might add or remove display boxes in a report from release to release, causing numbered subscripts to choose a different box. Using a string to index a box, followed by a numbered subscript to pick a specific child box, is more reliable across releases.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
rp = dt << Bivariate(
  Y( :weight ),
  X( :height ),
  Fit Line( {Line Color( {213, 72, 87} )} )
);
```

```
Print(
  "C.Total DF=",

  // string references the parent box
  Report( rp )[Outline Box( "Analysis of Variance" )][
  Number Col Box( 1 )][3] // subscript references the child box
);
```

Figure 11.7 Subscripted Display Box



The numbered subscripts (in the preceding example) are relative to the named box, which makes them less likely to change. However, this also means that the absolute numbers in Show Tree Structure can't be used directly.

Send Messages to Display Boxes

Use the display reference to send messages to the display elements using the `Send` or `<<` operator. For example, if `out2` is a reference to an outline node, you can ask the node to close itself if it is open:

```
out2 << Close(); // close the outline node
```

`Close()` toggles an outline node back and forth between closed and open states, just like selecting items in the red triangle menu. You can include a Boolean argument (1 or 0) for "close or leave closed" or "open or leave open."

```
rbiv["Fit Mean"] << Close;  // toggle open and closed
rbiv["Fit Mean"] << Close( 1 ); // close or leave closed
rbiv["Fit Mean"] << Close( 0 ); // open or leave open
```

Tip: The Report Invalid Display Box Messages option in the General preferences helps you identify invalid display box messages.

- The preference is off by default. When a display box receives an invalid message, JMP ignores those errors and continues executing the script.
- When this preference is on, an error appears in the log.

This option can be useful during script development but can cause unwanted log messages for existing scripts.

The << Operator

You can send messages to display boxes using the send (<<) operator. The operator makes it clear when the script is evaluating child arguments and optional arguments. It also helps make it clearer which argument is an option and which is a script to run inside the graph.

Here is an example of sending a Graph Box a message:

```
win = New Window( "Messages",
   gb = Graph Box(
      Frame Size( 400, 400 ),
      X Scale( 0, 25 ),
      Y Scale( 0, 25 )
   )
);
gb << Background Color( "red" );
```

Nesting and Precedence

When you stack (or *nest*) messages, each message is evaluated left to right.

- The following expression sends `message1` to `box`, then `message2` to `box`, then `message3` to `box`. Note that any of the messages can change `box` before the next message is sent.

  ```
  box << message1 << message2 << message3;
  ```

- The following expression sends `message1` to `box` and gets the result. `Message2` is sent to that result, and `message3` is sent to the result of `message2`.

  ```
  ( (box << message1) << message2) << message3
  ```

- The result of `message3` is assigned to the variable `x`.

  ```
  x = box << message1 << message2 << message3;
  ```

Here is an example of using nested messages:

```
win = New Window( "Messages", gb = Graph Box() );
```

```
// message 1 sets the background color
sz = gb << Background Color( "red" )

// message 2 saves the graph box as a PNG file
<<Save Picture( "$DOCUMENTS/red.png", "png" )

// message 3 sets the graph box background color to white
<<Background Color( "white" )

// message 4 returns the graph box size and prints it to the log
<<Get Size();
```

If you nest several messages, you might want to use parentheses to group the messages to be sure your script does what you want it to.

Customize Reports with the Send To Report and Dispatch Functions

The `Send To Report()` and `Dispatch()` functions provide a way for JMP to record display box modifications inside an analysis script in a self-contained way. For example, arguments within the two functions open and close outline nodes, resize graphics frames, or customize the colors in a graphics frame. You can also specify the arguments in separate JSL statements.

`Send To Report()` contains a list of options that change the display tree.

`Dispatch()` contains four arguments that change the display tree:

- The first argument is a list of outline nodes that need to be traversed to find the desired part of the display tree.
- The second and third arguments work together. The second argument is the name of a display element, and the third argument is the display element's type. These two arguments specify which particular part of the display tree is to be customized.
- The fourth argument is a list of options.

For example, open Big Class.jmp and run the attached Bivariate script. This generates a report with a fitted line. Open the Lack of Fit outline node, and close the Analysis of Variance outline node. Select **Save Script > To Script Window**. The following script appears in the script editor window:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate(
   Y( :weight ),
   X( :height ),
   Fit Line(),
   SendToReport(
      Dispatch(
         {"Linear Fit"},
         "Lack Of Fit",
```

```
            OutlineBox,
            {Close( 0 )} ),
         Dispatch(
            {"Linear Fit"},
            "Analysis of Variance",
            OutlineBox,
            {Close( 1 )}
         )
      )
   );
```

The `Send To Report()` function contains two `Dispatch()` functions that customize the default report.

- The first argument finds an outline node named “Linear Fit”.
- The second and third arguments finds an outline box named “Lack of Fit” below the “Linear Fit” outline.
- The fourth argument is the argument to send to this outline box. In this case, the message is `Close(0)`, in other words, open the node.

Note: If there are several outline nodes with identical names, subscripts are assigned to them. For example, if you have a Bivariate analysis with two quadratic fits (resulting in identical titles), when you dispatch a command to the second fit, the subscript [2] is added to the duplicated title.

The best way to deal with `Send to Report()` and `Dispatch()` is to first create a customized report interactively. Save the report as a script and then examine the script that JMP generates. Remember: the best JSL writer is JMP itself.

Create a Journal

Creating a journal enables you to capture the contents of a JMP window at a fixed moment in time. Suppose that a JMP window contains a graph and selected check boxes. The journal shows the graph and the check boxes that were selected when you created the journal.

The following examples show what you can do with a journal.

Suppose that you start with a Bivariate report.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( weight ), X( height ) );
rbiv = biv << Report;
```

Journal the results:

```
rbiv << Journal Window;
```

Save the journal to a file:

```
rbiv << Save Journal(
   "$DOCUMENTS/test.jrn"
);
```

Note that the above example uses macOS pathname conventions. On Windows, you can use forward slashes (which are appropriate for all platforms) or backslashes (which are appropriate only for Windows).

When a script uses a By variable, the result is a list of references to the analysis results for each By group. In order to journal all By member parts of the report, you need to use parent messages to get to the top of the report.

For example, the following script creates a Bivariate report with a By group and then journals the entire report:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate(
   Y( weight ),
   X( height ),
   By( sex )
);
( ( Report( biv[1] ) << Parent ) << Parent ) <<
Save Journal( "test.jrn" );
```

Journals with Grid Lines

When axes are collapsed, grid lines disappear when the report is journaled, pasted, or saved as a PDF file. Saving the report from JMP as a picture file or a Microsoft PowerPoint presentation retains the grid lines in the graph.

Using Common Messages

Update a Numeric Column

To update a numeric column in a display box table, use the `Set Values` message.

```
win = New Window( "Numeric Column",
   num = Number Col Box( "Values", [9, 10, 11] )
);
num << Set Values( [1, 2, 3] );
```

The matrix argument specifies the new numbers for the table.

Add the Thousands Separator in a Numeric Column

To add the thousands separator in a numeric column, use the `Set Format` message.

```
New Window( "Example",
   ncb = Number Col Box( "Random Numbers",
      {Random Uniform(), Random Uniform() * 10,
      Random Uniform() * 100, Random Uniform() * 1000,
      Random Uniform() * 10000}
   )
);
ncb << Set Format(10,3, "Fixed Dec", "Use thousands separator");
```

Update a String Column

To update a string column in a display box table, use the `Set Values` message.

```
win = New Window( "String Column",
   str = String Col Box( "Values", {"a", "b", "c"} )
);
str << Set Values( {"A", "B", "C"} );
```

The list argument specifies the new strings for the table.

Wrap Text

Generally, JMP automatically wraps text within `Text Box()`. However, the default wrap point can be overridden with a `Set Wrap` (*n*) message. The following script wraps the text at 200 pixels.

```
win = New Window( "Set Wrap",
   tb = Text Box(
      "Hover over a data point for more information."
   )
);
tb << Set Wrap( 200 );
```

Include Bullet Points in Text Boxes

You can add bullet points using the `Bullet Point( 1 )` message.

```
win = New Window( "Bullet List",
   text1 = Text Box( "Hover over a data point for more information." ),
   text2 = Text Box( "Draw a circle around a statistic for more information." )
);
text1 << Bullet Point( 1 );
text2 << Bullet Point( 1 );
```

Sending the `Bullet Point( 1 )` message to a text box places a bullet in front of the text and indents subsequent lines within that text box.

You can also send the `Bullet Point( 1 )` message inline as shown in the following example:

```
text1 = Text Box( "Hover over a data point for more information." ), << Bullet
  Point( 1 ) ),
```

Blink a Selected Display Box

You can use `Select`, `Reshow`, and `Deselect` to blink the selection highlight on a display box. As previously described, you can string together several << clauses to send an object several messages in a row. The results read left to right; for example, here `Select` is done first, then `Reshow`, then `Deselect`, then the other `Reshow`.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate(
  Y( :weight ),
  X( :height ),
  Fit Mean( )
);

// assign the bivariate analysis layer to rbiv
rbiv = biv << Report;

/* find outline box with the specified string in the
rbiv report and assigns it to the out2 variable */
out2 = rbiv[Outline Box( "? Mean" )];
out2 << Close( 0 ); // open the outline box

// find the first string col box in rbiv and
assign the scbx variable */
scbx = rbiv[String Col Box( 1 )];

Wait( .25 ); // wait .25 seconds
For( i = 0, i < 20, i++, // set i to 0, iterate through the loop 20 times

  /* select the string col box and reshow it,
  deselect it and reshow it */
  scbx << Select << Reshow << Deselect << Reshow
);
```

Add Shading or Borders to Table Headings

Headings are shaded and have column borders by default. To turn them off, use the `Set Shade Headings ( 0 )` message and the `Set Heading Column Borders ( 0 )` message. The following example shows the effect of turning shading and borders off:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Summarize(
  meanHt = Mean( Height ),
  minHt = Min( Height ),
```

```
   maxHt = Max( Height )
);
win = New Window( "Summary Results",
   tb = Table Box(
      Number Col Box( "Mean", meanHt ),
      Number Col Box( "Min", minHt ),
      Number Col Box( "Max", maxHt )
   )
);
Wait( 2 );
tb << Set Shade Headings( 0 );
Wait( 1 );
tb << Set Heading Column Borders( 0 );
Wait( 2 );
tb << Set Shade Headings( 1 );
Wait( 1 );
```

Example of Creating a Report

Here is a script to build a report from start to finish.

1. Open a data table.

   ```
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   ```

2. Launch a Bivariate platform and assign it to the platform reference `biv`.

   ```
   /* a reference to the platform */
   biv = dt << Bivariate( Y( weight ), X( height ) );
   ```

3. To find out what you can do with the Bivariate platform, search for Bivariate in the Objects list of the Help > Scripting Index. Here are a few options:

   ```
   biv << Fit Spline( 1000000 ) << Fit Mean;

   // degree 4, orange curve
   biv << Fit Polynomial( 4, {Line Color( RGBColor( 1, 0.5, 0 ) )});
   biv << ( Curve[1] << Line Color( red ) ); // first curve is red
   ```

4. Assign `rbiv` to the report layer:

   ```
   rbiv = biv << Report;
   ```

5. Get the frame box to a comfortable size and scroll up to the top to see the graph.

   ```
   rbiv[Frame Box( 1 )] << Size Window( 500, 700 );
   biv << Scroll Window( {0, 0} );
   ```

6. Get to work on the report. First, you need to create a reference, and then see what you can do with it.

   ```
   Show Properties( rbiv ); // show the properties of the report
   ```

Note: The log lists the messages to use with reports. Using `Show Properties()` is an alternative to the Scripting Index.

```
Close [Boolean]
GetOpen [Action] [Scripting Only]
SetOpen [Boolean] [Scripting Only]
...
```

7. Open the Fit Mean node of the outline:

```
rbiv["Fit Mean"] << Close( 0 );
```

`Close()` is a Boolean function (true or false), and passing 0 indicates that the outline is not closed.

8. Practice selecting some results (submit each line alone to see its result):

```
rbiv["Summary of Fit"] << Select;
rbiv["Parameter Estimates"] << Select;
rbiv["Analysis of Variance"] << Select;
rbiv["Polynomial Fit Degree=4", "Parameter ?", Column Box( "Estimate" )] <<
  Select; // dig down in the outline tree
rbiv << Deselect;
```

9. Change the format of the Estimate column in the 4-degree polynomial Parameter Estimates report.

```
pe = rbiv["Polynomial Fit Degree=4", "Parameter ?"];
ests = pe[Number Col Box( "Estimate" )];
ests << Set Format( 12, 6 );
```

The first argument to `Set Format` sets the column width by the number of characters to display. The second argument sets how many decimal places are shown in the table.

Figure 11.8 Applying Changes to a Report

before

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	-17.60008	84.82175	-0.21	0.8368
height	1.9521428	1.346006	1.45	0.1559
(height-62.55)^2	-0.220179	0.29996	-0.73	0.4678
(height-62.55)^3	0.0703948	0.035633	1.98	0.0561
(height-62.55)^4	0.0073899	0.004231	1.75	0.0895

after
Set Format(12,6)

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	-17.600085	84.82175	-0.21	0.8368
height	1.952143	1.346006	1.45	0.1559
(height-62.55)^2	-0.220179	0.29996	-0.73	0.4678
(height-62.55)^3	0.070395	0.035633	1.98	0.0561
(height-62.55)^4	0.007390	0.004231	1.75	0.0895

Note: You can even get a single number out of the table. For example, get the estimate for the cubic term by adding the following script to the preceding script:

```
terms = rbiv[Outline Box( "Parameter Estimates" )][String Col Box( "Term"
  )]; // find the Term column
ests = rbiv[Outline Box( "Parameter Estimates" )][Number Col Box(
  "Estimate" )]; // find the Estimate column
estimate = .; // in case there is no cubic term
Try(
    For( i = 1, i < 10, i++,

        // get(i) goes off the end of the table handled by Try
        If( Contains( terms << Get( i ), "^3" ),
         estimate = ests << Get( i );
            Break();
        )
    )
);
Show( estimate );
  0.070394822744608
```

10. Use a `For()` loop to count down to the row for the term that you want. The second argument to `For()` is a condition; as long as the condition tests true, looping continues. Here the test is "when the string in the `Terms` column is not `"^3"` and we have not reached the tenth row." As soon as the string *does* match, looping stops and `i`'s value is the number for the matching row. You then use `i` as a subscript to the `Get` message on the Estimates column.
11. You can also get values from boxes as a matrix, which you can then use for further computations or write to a data table. The following example also shows how to make a data table:

```
// get the values in the Sum of Squares column as a matrix
myVector = rbiv[Table Box( 5 )][Number Col Box( "Sum of Squares" )] << Get as
  Matrix;

/* create a new column in Big Class.jmp named "Sum of Squares"
insert the values of myVector in the column */
dt << New Column( "Sum of Squares", Values( myVector ) );

// put the values of the fifth table box into a new data table
rbiv[Table Box( 5 )] << Make Data Table( "ANOVA table" );
```

12. Now adjust the scales on the axes.

```
rbiv[Axis Box( 1 )] << Min( 70 ) << Max( 170 ); // set the Y axis
rbiv[Axis Box( 2 )] << Min( 50 ) << Max( 70 );  // set X axis
```

13. Copy the graph at the top of the report. Note that you need to select the picture box that contains the graph; selecting just the graph would leave its axes behind.

```
rbiv[Picture Box( 1 )] << Copy Picture;
```

Construct Custom Windows

You can use constructor functions to create objects and construct your own displays. For example, you might prompt the user for information, create custom reports, create custom launch windows, or add check box and radio button items to a window.

Start with a `New Window()`, including a title in quotes, and then construct display boxes inside the window. Make a reference to the new window by assigning it a variable so that you can send messages to it. When display objects are created or referred to by JSL, they are freely shared references until they are copied into another display box or until you close the window and they disappear. When you plug a display object into another display tree, JMP makes a copy of it that the new box owns.

The names of all display box constructors end in “Box”. There are similar objects that live inside graphics frames called display segs (segments such as markers and lines). The names of all display seg objects end in “Seg”.

Example of Creating a Graph Box

The following example creates a new window that contains a graph box:

```
win = New Window( "Simple Earth Map",
   gb = Graph Box(
      X Scale( -180, -60 ),
      Y Scale( 20, 80 ), // end of graph box parameters
      <<Background Map( Images( "Simple Earth" ) ) // beginning of script
   )
);
```

The parameters set the x-axis and y-axis scale and the frame size. The `Background Map` message then shows the map. Note that the parameters for `Graph Box()`, `X Scale` and `Y Scale`, come first. After the first comma, the script begins. If you put the script in the middle of the parameters in a `New Window()` function, the script might run but not produce the results you expect. Scripts are also easier for the users to read if the parameters are at the top of the new window.

Extract Values from a Window

After the user selects options in a window, you can extract those selections from the window using the `Return Result` message, `Get` message, or `Get Selected` message.

Extract Values: Method One

To have a `New Window()` script automatically return the results after the user clicks OK, include the `Return Result` message after the `Modal` message.

```
win = New Window( "Set a Value",
   <<Modal,
   <<Return Result,
   Text Box( "Set this value" ),
   variablebox = Number Edit Box( 42 ),
   Button Box( "OK" ),
   Button Box( "Cancel" )
);
Write( win["variablebox"] ); // create a subscript to the variablebox variable
   33 // the user typed "33" in the number edit box
```

Extract Values: Method Two

Include the `Get` message to return 1 for a selected check box and 0 for a deselected check box. To view the selection, add a `Show()` expression at the end of the script.

```
win = New Window( "V List Box",
   <<Modal,
   V List Box(
      kb1 = Check Box( "a" ),
      kb2 = Check Box( "b" ),
      kb3 = Check Box( "c" )
   ),
   Button Box( "OK",
      val1 = kb1 << Get; // get the value of the first check box
      val2 = kb2 << Get;
      val3 = kb3 << Get;
   )
);
Show( val1, val2, val3 ); // return variables after window closes
   val1 = 1; // first and second checkboxes are selected
   val2 = 1;
   val3 = 0; // third checkbox is not selected
```

Extract Values: Method Three

Include the `Get Selected` message to return the selected column names and then insert the columns in a Bivariate plot.

Note: Method two and method three of extracting values are alternate examples of the same method. The method, in both examples, uses a JSL callback from one of the controls in the window. The JSL callback captures a value from a display box in the window into a global variable that will be available after the window closes. In method two, the JSL callback is on a button box. In method three, the JSL callback is on a list box.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
xvar = .;
yvar = .;
win = New Window( "Return Values",
   <<Modal,
   // require the user to select two variables before clicking OK
   <<On Validate(
      Show( xvar, yvar );
         If( Is Missing( xvar ) | Is Missing( yvar ),
            0, // if xvar or yvar are missing, do nothing when OK is clicked
         1
      );
   ),
   Text Box( " Select two numeric columns. " ),
   H List Box(
      Text Box( " X, Factor " ),
      x = Col List Box(
         dt, // data table reference
         all, // display all columns from the data table
         xvar = (x << Get Selected)[1];
         // get the name of the selected column before the window closes
         Show( xvar );
      ),
      Text Box( "Y, Response" ),
      y = Col List Box(
         dt,
         all,
         yvar = (y << Get Selected)[1];
         Show( yvar );
      )
   )
);
xcol = Column( dt, xvar ); // get the columns
ycol = Column( dt, yvar );
```

```
dt << Bivariate( Y( ycol ), X( xcol ) ); // create a Bivariate plot
```

Constructors for New Windows

JSL supports both modal and non-modal windows.

- Modal windows must be closed before other windows can be used: clicking outside the modal window produces an error sound, and almost all script execution stops until the user closes the modal window.
- JMP reports and platform launch windows are examples of non-modal windows. Many non-modal windows can be open at once and you can switch between them freely. The Transparency dialog in a graph is an example of a modal window. After the Transparency dialog is opened, you must close it before continuing.

This section describes common display boxes that you can construct in a window. There are many other display boxes (such as axis boxes) in JMP that cannot be constructed. Display boxes that cannot be constructed are used by platforms in JMP. Even though they can't be constructed in JSL, they still have functions and messages that can be used from JSL.

Tip: In the Scripting Index (**Help** > **Scripting Index**), display boxes that cannot be constructed are marked by . Display boxes that you can construct are marked by .

Notes:

- The `Dialog()` function, which creates modal windows, is deprecated and will not work at all in future versions of JMP. Use either `New Window()` with the `Modal` message or `Column Dialog()` for column selection. See "Construct a Column Dialog" and "Convert Deprecated Dialog to New Window".
- In JMP, the following display boxes support only one direct child box: `Border Box()`, `Center Box()`, `If Box()`, `Mouse Box()`, `Scroll Box()`, and `Sheet Panel Box()`. To make a border box that appears to have multiple children, use a `V List Box()` or `H List Box()` as the only child and put multiple children in the `V List Box()` or `H List Box()`.

Container Display Boxes

Container display boxes enclose child display boxes.

Border Box

`Border Box()` adds space around the *displaybox* argument.

```
Border Box (Left( pix ), Right( pix ), Top( pix ), Bottom( pix ),
  Sides( int ), display box args);
```

`Left`, `Right`, `Top`, and `Bottom` add space around the *`displaybox`* argument. `Sides` draws a border around the box, as described in Table 11.2. Additional effects can also be applied to the borders using `Sides`, as described in Table 11.2. To add both an effect and a border, add the two numbers.

For example, the following example produces a text box with a border at the top and bottom (Draw Border value of 5) colored blue, the default color (Effect value of 16):

```
win = New Window( "Borders",
   bb = Border Box( Sides( 5 ), // top and bottom border
      Text Box( "Hello World!" )
   )
);
bb << Set Color( "red" ); // set the border color to red
```

The `bb` variable is assigned to the border box so that the color can be set on the last line.

Note: Border boxes support only one display box argument. To make a border box that appears to have multiple children, use a `V List Box()` or `H List Box()` as the only child and put multiple children in the `V List Box()` or `H List Box()`.

In the `Sides()` function, add the following values to get the borders that you want. For example, "5" draws a top and a bottom border.

1: top

2: left

4: bottom

8: right

`Border Box()` provides the additional effects described in Table 11.2.

Table 11.2 Border Box Sides() Additional Effects

Add to Number Above	Additional Effect
16	Make the border lines the same color as the operating system's highlight color. Not recommended because this color is different on different computers.
32	Fill the border box with white before drawing the border box's child.

Table 11.2 Border Box Sides() Additional Effects *(Continued)*

Add to Number Above	Additional Effect
64	Fill the border box with the background color before drawing the border box's child. This option makes a difference only if the border box's parent draws something behind the border box first.

Col List Box

`Col List Box()` returns a display box that shows a list of current data table columns.

```
Col List Box( <Data Table( <name> ), <all>|<character>|<numeric>,
   <Width( n )>, <Max Selected( n )>, <NLines( n )>, <Max Items( n )), <Min
   Items( n )>, <On Change( expression )>, <script> );
```

Note: To add a transform column to a custom display box, use `Filter Col Selector()` instead of `Col List Box()`. See "Filter Col Selector".

`All` specifies that all columns in the current data table should be included. `Width` is measured in pixels. `Max Selected` is the maximum number of items that might be selected in the list box. `NLines` is the number of lines displayed in the box. `Max Items` is the maximum number of items displayed in col list box. `Min Items` is the minimum number of items displayed in the col list box. `On Change` is the script that runs when the selection changes.

You can send a `Get Items` message to a col list box to retrieve a list of all columns selected. Here is an example script showing `Get Items` in use. See the Scripting Index in the Help menu for other messages and examples.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Get Items Demonstration",
   H List Box(
      chooseme =

      // Col List Box shows all columns, 100 pixels wide with 6 lines
      Col List Box( "All", width( 100 ), NLines( 6 ) ),

      /* line up box arranges button boxes in 1 column
      with 2 pixels of space around the boxes*/
      Line Up Box( N Col( 1 ), Spacing( 2 ),

         Button Box( "Add Column >>",
```

```
            /* append the selected column to previously
            selected columns */
            listcols << Append( chooseme << Get Selected );

            // get a list of selected columns from the col list box
               Chosen Columns = listcols << Get Items;
                  ),
         Button Box( "<< Remove Column",
            listcols << Remove Selected; // remove the selected column
            Chosen Columns = listcols << Get Items; // get the selected column
         ),
      ),
         // create another col list box with no columns inside
         listcols = Col List Box( width( 100 ), NLines( 6 ) ),
   ),
   Text Box( " " ),

   /* at the bottom of the h list box,
   show which columns Get Items returned */
   stuff = Global Box( Chosen Columns )

);
```

The `All` argument for `Col List Box()` includes all data table columns in the display box. Because all columns are required, you cannot remove selected rows using `Remove Selected` or `Remove All`. Instead, create an empty column list box and append all columns. Then the columns that you select can be removed.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
colNames = dt << Get Column Names; // get a list of column names

New Window( "Example",
// create Col List Box with all the column names from the data table
myBox = Col List Box( width( 200 ), nlines( 10 ) ) );
myBox << Append( colNames );

myBox << Set Selected( 3 ); // select one or all of the entries in myBox
myBox << Remove Selected(); // remove selected entries in myBox
```

Note: To show only the character or numeric columns from the data table in the col list box, send `Set Data Type(numeric|character)` to the col list box.

Col Span Box

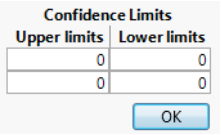
`Col Span Box()` creates spanned columns headers inside a table box. The top column header spans two child column headers.

```
Col Span Box( "title", display box args);
```

In the following script, the Confidence Limits header spans the Upper limits and Lower limits columns below it.

```
win = New Window( "Col Span Box",
   <<Modal,
   Table Box(
      Col Span Box(
         "Confidence Limits",
         neb = Number Col Edit Box( "Upper limits", [0, 0] ),
         Number Col Edit Box( "Lower limits", [0, 0] )
      )
   )
);
```

Figure 11.9 Confidence Limits



Filter Col Selector

`Filter Col Selector()` creates a display box that shows a list of items, for example, columns in a data table. The display box is similar to `Col List Box()` except that `Filter Col Selector()` enables the user to filter columns by name, modeling type, data type, and so on. And `Filter Col Selector()` supports transform columns; `Col List Box()` does not.

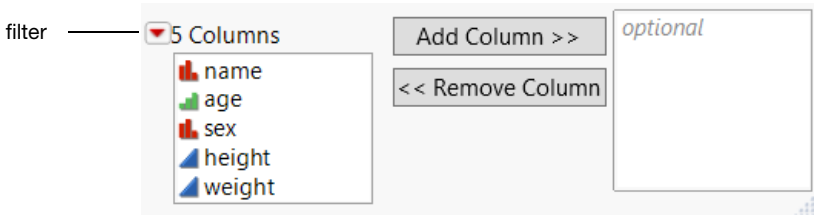
```
Filter Col Selector( <Data Table( name )>, <Width( pixels )>, <Nlines( n )>,
   <script>, <On Change( expr )>, display box args, ... );
```

Here's an example that shows how to use `Get Items` to display a list of selected columns from the filter col selector.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Filter Col Selector Example",
   H List Box(
      chooseme =

      // filter col selector shows all columns, 100 pixels wide with 6 lines
      Filter Col Selector( width( 100 ), NLines( 6 ) ),
```

```
        /* line up box arranges button boxes in 1 column
        with 2 pixels of space around the boxes */
        Line Up Box( N Col( 1 ), Spacing( 2 ),
                Button Box( "Add Column >>",

             /* append the selected column to previously
             selected columns */
             listcols << Append( chooseme << Get Selected );

                // get a list of selected columns from the filter col selector
                Chosen Columns = listcols << Get Items;
                     ),
           Button Box( "<< Remove Column",
              listcols << Remove Selected; // remove the selected column
              Chosen Columns = listcols << Get Items; // get the selected column
           ),
        ),
           // create another col list box with no columns inside
           listcols = Col List Box( width( 100 ), NLines( 6 ) ),
     )
  );
```

Figure 11.10 Filter Col Selector



H List Box and V List Box

`H List Box()` aligns boxes horizontally.

```
H List Box( <Align( center|bottom )>, display box args, ...);
```

`Align` specifies center or bottom alignment of the contents in the display box. The contents are top aligned by default.

The following script aligns the check boxes horizontally:

```
win = New Window( "H List Box",
   <<Modal,
   H List Box(
      kb1 = Check Box( "a" ),
      kb2 = Check Box( "b" ),
      kb3 = Check Box( "c" )
```

```
   ),
);
```

`V List Box()` aligns boxes vertically.

```
V List Box( <Align( center|right )>, display box args, ...);
```

`Align` specifies left, right, or center alignment of the contents in the display box. The contents are left aligned by default.

```
win = New Window( "V List Box",
   Text Box( "Select your favorite fruit." ),
   V List Box(
      Check Box( "apple" ),
      Check Box( "banana" ),
      Check Box( "orange" ),
   ),
);
```

Note: The default maximum width is 180 pixels. To set the width, set the width of the display box that encloses the V List Box.

Suppose that you want to enable the user to add a transform column to custom dialog box. Here is an example of how to create a horizontal list box with an Add and Remove button. The user right-clicks a column to create a transform column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
New Window( "Example",
     H List Box(
           ll1 = Filter Col Selector( ),
           Button Box( "Add", ll2 << Append( ll1 << Get Selected() ) ),
           ll2 = Col List Box( "Numeric", MaxItems( 1 ), nlines( 1 ) ),
           Button Box( "Remove", ll2 << Remove Selected )
     )
);
```

Note: Transform columns use formulas or calculations to define the column values. Closing the window deletes any transform columns. See *Using JMP*.

If Box

`If Box()` creates a display box whose contents are conditionally displayed.

```
If Box( 0|1, display box args );
```

The Boolean argument 1 displays the display boxes inside the if box. 0 does not display them.

The following script creates a splitter box with three scroll boxes. The second scroll box is enclosed in `If Box()`. Because `If Box()` is set to 1, the second scroll box appears in the window.

```
win = New Window( "If Box",
   H Splitter Box(
      Size( 500, 250 ),
      Scroll Box(),
      If Box( 1, Scroll Box() ), // if box sets the scroll box to show
      Scroll Box()
   )
);
```

When you set an if box to be hidden, then anything within the if box disappears from the report window. In the tree structure view, the hidden display box is no longer numbered. Hide the if box in this example, and the second scroll box is no longer numbered in the tree structure view. `Scroll Box(3)` becomes `Scroll Box(2)`. Change the argument for `If Box()` in this script from 1 to 0 and then send the `win << Show Tree Structure` message to see the structure view.

Note: If boxes support only one display box argument.

H Splitter Box and V Splitter Box

`H Splitter Box()` and `V Splitter Box()` create a display box that arranges the display boxes provided by the arguments in a horizontal or vertical layout (or panel). The splitter enables the user to interactively resize the panel.

```
H Splitter Box( <Size( h,v )>, display box args, <arguments> );
V Splitter Box( <Size( h,v )>, display box args, <arguments> );
```

`Size( h, v )` specifies the size of the splitter box in pixels. Inner display boxes are proportionately sized according to the width and height of the splitter box. For a list of arguments, see the Scripting Index in the JMP Help menu.

The following example puts a Bivariate and Oneway graph inside an H splitter box. Splitter boxes are especially useful when the content is set to auto-stretching as shown in the example.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "H Splitter Box",
   H Splitter Box(
      Size( 900, 600 ),
      biv = dt << Bivariate( Y( :weight ), X( :height ), Fit Line(), Line
   Style( Dotted ) ),
      one = Oneway( Y( :height ), X( :sex ), "Means/Anova"n( 1 ), Mean
   Diamonds( 1 ) )
   )
);
```

```
/* make a reference to the report layers
set the x and y axes on the frame boxes to auto stretch
when the user drags the splitter */
(biv << Report)[FrameBox( 1 )] << Set Stretch( "Window", "Window" );
(one << Report)[FrameBox( 1 )] << Set Stretch( "Window", "Window" );
```

Line Up Box

`Line Up Box()` shows the *display box* arguments in *n* columns. You can specify optional spacing, in pixels, for the space between columns.

```
Line Up Box( NCol( n ), <Spacing( pixels )>, display box args, ... );
```

`Spacing` specifies the number of pixels around the box.

In the following example, the Distribution reports are lined up in three columns:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Line Up Box",
   Line Up Box( NCol( 3 ), // line up the Distributions in 3 columns
      dist = Distribution(
         Continuous Distribution( Column( :height ) ),
         By( :age )
      )
   )
);
```

Outline Box

`Outline Box()` creates a display box to hold other display boxes and show them in an outline view.

```
Outline Box("title", display box args, ... );
```

In the following example, each outline contains a Distribution:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Outline Box",
   obj1 = Outline Box( "Age and Weight",
      Distribution( Column( :age, :weight ) ),

   ),
   obj2 = Outline Box( "Weight and Height",
      Distribution( Column( :weight, :height ) )
   )
);
```

To close an outline by default, send the `Close( 1 )` message to the outline object.

```
obj2 << Close( 1 );
```

The first argument for an outline box is the title. You can specify the title in the script, or the title can be based on another component, such as the value of a slider box. In the following example, as the user adjusts the slider, the title of the outline box is updated:

```
sliderValue = .6;
win = New Window( "Outline Box",
   Panel Box( "Slider Box",
      tb = Outline Box( "Value: " || Char( sliderValue ) ),
      sb = Slider Box(
         0,
         1,
         sliderValue,
         tb << Set Title( "Value: " || Char( sliderValue ) )
      )
   )
);
```

To make your own red triangle menu within an outline box, send the `Set Menu Script` message to the outline box reference. The following example also uses the `Set Submenu` message to create nested menu items.

```
win = New Window( "Red Triangle Menu Items",
ob = Outline Box( "Outline Box" ) );
ob << Set Menu Script(
   {"A", "", "A1", Print( "A1" ), "A2", Print( "A2" ), "B", "",
   "B1", Print( "B1" ), "B2", Print( "B2" ), "B3",
   Print( "B3" ), "C", Print( "C" ), "Selected", Print( "Selected" )}
);
ob << Set Submenu( 1, 2 ); // menu A with 2 items in submenu A1 and A2
ob << Set Submenu( 4, 3 ); // menu B with 3 items in submenu B1, B2, and B3
```

To indicate whether an item is selected, unavailable, or available, include `Set Menu Item State`. The following expression shows a check mark beside the ninth menu item (“Selected”) in the preceding script.

```
ob << Set Menu Item State( 9, 1 );
```

Specify the value -1 to disable the menu item or 0 to make it available.

Panel Box

`Panel Box()` encloses the *`displaybox`* argument in a labeled border.

```
Panel Box( "title", display box args );
```

In the following example, the panel box appears around the edge of the Distribution launch window:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Panel Box",
   H List Box( Panel Box( "My Distribution", Distribution() ) )
);
```

Display Element Boxes

Graph Box

`Graph Box()` constructs a display box that contains a frame box and axes.

```
Graph Box( "title", <X Scale( low, high ), <Y Scale( low, high )>, <Frame
   Size( h, v )>,< <XName( "name" )>, <YName( "name" )>, <SuppressAxes>,
   script );
```

`X Scale()` and `Y Scale()` specify the lower and upper limits of the x and y axes. `Frame Size()` specifies the size of the frame around the graph. `XName` and `YName` specify the name of the x and y axes.

The following example creates a simple graph:

```
win = New Window( "Graph Box",
   Graph Box(
      Frame Size( 300, 300 ),
      Marker( Marker State( 3 ), [11 44 77], [75 25 50] );
      Pen Color( "Blue" );
      Line( [10 30 70], [88 22 44] );
   )
);
```

To remove tick marks, add two arguments to the axes.

```
win = New Window( "Graph Box",
   Graph Box(
      Frame Size( 300, 300 ),
      xaxis( // remove tick marks from the x axis
         Show Major Ticks( false ),
         Show Minor Ticks( false ),
         Show Labels( false )
      ),
      yaxis( // remove tick marks from the y axis
         Show Major Ticks( false ),
         Show Minor Ticks( false ),
         Show Labels( false )
      ),
      Marker( Marker State( 3 ), [11 44 77], [75 25 50] );
```

```
        Pen Color( "Blue" );
        Line( [10 30 70], [88 22 44] );
    )
);
```

JMP provides a robust set of features to make your graphs interactive (such as dragging objects or capturing mouse clicks). "Scripting Graphs" describes graph box options in detail.

H Sheet Box and V Sheet Box

`Sheet Box()` lets you create a grid of plots. `H Sheet Box()` and `V Sheet Box()` contain display boxes and arrange them in columns and rows.

```
H Sheet Box( <<Hold( report ), display box args );
V Sheet Box( <<Hold( report ), display box args );
```

The general approach is to first consider which display boxes you want and in which arrangement. Create either an `H` or `V Sheet Box` and send it a `Hold` message for each graph to specify which element the sheet box should hold. Finally, create interior `H` or `V Sheet Box`es and tell each one which graph the sheet box should hold.

Here is an example of creating a sheet with four plots: a Bivariate plot, a Distribution, a Treemap, and a Bubble Plot.

First, open the data table and create a new window.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
nw = New Window( "Sheet Box Example",
```

Use a `V Sheet Box` to organize the window into two columns.

```
    V Sheet Box(
```

Send the `V Sheet Box()` four `Hold` messages, one for each graph. The order matters.

```
        <<Hold(
            dt << Bivariate( // graph 1
                Y( :weight ),
                X( :height ),
                Fit Line()
            )
        ),
        <<Hold(
            Distribution( // graph 2
                Continuous Distribution(
                    Column( :height ),
                    Horizontal Layout( 1 ),
                    Outlier Box Plot( 0 )
                )
            )
```

```
        ),
        <<Hold( Treemap( Categories( :age ) ) ), // graph 3
        <<Hold(
           Bubble Plot( // graph 4
              X( :height ),
              Y( :weight ),
              Sizes( :age ),
              Coloring( :sex ),
              Circle Size( 6.226 ),
              All Labels( 0 )
           )
        ),
```

Finally, add two `H Sheet Box`es to the `V Sheet Box` and tell each one which graph it should hold. Each H `Sheet Box` holds two side-by-side graphs. They are held by a `V Sheet Box`, so the `H Sheet Box`es are displayed vertically.

```
        H Sheet Box(
           Sheet Part(
              "",
```

`Sheet Part` displays a previously defined graph, held by `Excerpt Box`. The first argument is the number of the graph, determined by the order in which you defined the graphs. The first `H Sheet Box` contains the Bivariate graph on the left and the Distribution on the right. `{Picture Box( 1 )}` designates which picture box from the report to display. Typically, use `1`, the first picture box in the report.

```
              Excerpt Box( 1, {Picture Box( 1 )} ) // excerpt box 1
           ),
           Sheet Part(
              "Distribution of height",
              Excerpt Box( 2, {Picture Box( 1 )} ) // excerpt box 2
           )
        ),
        H Sheet Box(
           Sheet Part(
              "",
              Excerpt Box( 3, {Picture Box( 1 )} ) // excerpt box 3
           ),
           Sheet Part(
              "My Title Here",
              Excerpt Box( 4, {Picture Box( 1 )} ) // excerpt box 4
           )
        )
     )
  );
```

The `Sheet Part()` title is required. If you include an empty string as the title, the sheet part's title is the default report title, for example, "Bivariate Fit of weight By height".

Journal Box

The `Journal Box()` constructs a display box from instructions that are stored in a journal.

```
box = Journal Box("journal text");
```

where *"journal text"* is text that has been extracted from a journal file.

Since journal text has lots of rules about which boxes can be with other boxes, we recommend that you get journal text by highlighting an area of the report, selecting Edit > Journal, and save it as a journal. Open the journal in a text editor, and then paste it into your script as the `Journal Box()` argument.

We highly recommend that you use the `"\[ ... ]\"` quoting mechanism so that you do not have to escape double quotes within the journal text. This excerpt shows how to escape the journal text:

```
win = New Window( "Mosaic Plot",
   Journal Box(
      "\[ // beginning of quoting mechanism
   ...display box arguments...
   ]\"
   ) // end of quoting mechanism
);
```

Another way to get journal text is to send the `Get Journal` message to a display box. For example, run the following script and copy the journal text from the log into the `Journal Box()` function:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( y( :weight ), x( :height ) );
rbiv = biv << Report;
Print( rbiv << Get Journal ); // print the journal to the log
```

Picture Object

JSL has a Picture Object, which stores pictures of JMP output or formulas. You can take a picture of anything in a display box or create a picture of a text formula as it would look in the Formula Editor.

To create picture data, send a `Get Picture` message to a *display box*.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( y( :weight ), x( :height ) );
rbiv = biv << Report;
rbiv << Get Picture;
```

The `Expr As Picture()` function evaluates its argument and creates a picture of the expression, using the same formatting mechanism as the Formula Editor. If you have a literal expression as the argument, remember to enclose it in `Expr()` so that JMP just takes a picture of the result. The expression then is not evaluated.

```
win = New Window( "Formula",
   Expr As Picture( Expr( a + b * c + r / d + Exp( x ) ) )
);
```

After you have the picture, there are two ways of using it:

- Add the picture to a new display tree using a display box constructor.
- Write the picture to a file using `Save Picture()`.

  ```
  picture << Save Picture( "path", type );
  ```

  where *type* can be EMF (Windows), JPEG or JPG, GIF, PDF, or PNG.

On Windows, the Windows Specific preferences determine the resolution (or DPI), or you can run this expression:

```
Pref( Save Image DPI( <number> ) );
```

On macOS, the operating system determines the DPI.

Input Selector Display Boxes

Input selector display boxes enable users to enter information and interact with the display box. The display box script includes a script that runs when the user interacts with the display box. See "Set Function and Set Script" for more information about writing the scripts.

Button Box

`Button Box()` draws a button that contains text.

```
Button Box("text", <script>);
```

The following example creates a new window with an OK button and a Cancel button:

```
win = New Window( "Set the Value",
   <<Modal,
   Text Box( "Set this value" ),
   variablebox = Number Edit Box( 42 ),
   Button Box( "OK" ),
   Button Box( "Cancel" ) );
```

Note that a modal window must have at least one button. If you omit the `Button Box()` from your script, an OK button is automatically included. To omit the OK button, set the visibility to collapse.

```
win = New Window( "No OK Button",
```

```
    <<Modal,
    b = Button Box( "Close Window", b << Close Window ),
    bb = Button Box( "OK", <<Visibility( "collapse" ) )
);
```

You can add a tooltip for a button by sending a `Set Tip` message to it. The following script creates a button named "Submit". A tooltip that contains "Send my information." appears when you hover over the button.

```
Button Box( "Submit", << Set Tip( "Send my information." ) );
```

You can also send the message `Open Next Outline` as a script command, which causes the next outline box to open. If you are sending more than one message to the button box, this must be the first command listed. When the user clicks the next button in this example, the second outline box opens.

```
win = New Window( "Open Next Outline",
   Outline Box( "First",
      ex = Button Box( "Next", ex << Open Next Outline )
   ),
   Outline Box( "Second", Text Box( "Last" ), << Close )
);
```

Using the `Set Menu Items` message, you can create a button that contains a menu:

```
win = New Window( "Set and Get Menu Choice",
   bb = Button Box( "Select a dessert",

   // get menu choices when the button is clicked
      choice = bb << Get Menu Choice;
      Show( choice );
   )
);
bb << Set Menu Items( {"fruit", "no thank you", "-", "chocolate bar", "ice
   cream"} );
```

The `"-"` in the preceding script creates a menu separator. The separator counts as an item in the list. If you select `chocolate bar` from the menu, `4` is returned, not `3`.

To indicate whether an item is selected, unavailable, or available, include `Set Menu Item State`. The following example makes the fifth menu item ("ice cream") in the preceding script unavailable:

```
bb << Set Menu Item State( 5, -1 );
```

Specify 0 to make the menu item available or 1 to show a check mark before the menu item.

Note that, after you click the button, the button box doesn't know where the script is stored because the button is *disconnected* from the script. For example, relative paths don't work when executed from a button box because of this disconnection. An `Open()` script must evaluate the current directory and concatenate it to the relative file path to construct an absolute path.

```
nw = New Window( "Open File Example",
   Button Box( "Open",
      Open(
         Eval( Get Path Variable( "$SAMPLE_DATA" ) ) ||
         "Quality Control/Coating.jmp"
      )
   )
);
```

In the following example, the path is interpreted as an absolute path beginning with the root.

```
nw = New Window( "Example",
Button Box( "Open", Open( "/MyData/Big Class.jmp" ) ) );
```

Note: Line-break characters are ignored in button boxes.

Check Box

`Check Box()` constructs a display box to show one or more check boxes. Any number of the check box items can be selected simultaneously.

```
Check Box ({"item 1", "item 2", ...}, <script>);
```

The following example shows how to get the user's selection by sending the `Get Selected` message to the check box. Each selection is printed to the log.

```
win = New Window( "Check Box",
   cb = Check Box(
      {"apple", "banana", "orange"},
      scb = cb << Get Selected();
      Show( scb );
   )
);
   scb = {"banana", "orange"};
```

The following example shows one way to enable the user to click OK to close the window.

```
win = New Window( "Check Box",
   <<Modal,
   <<On Close(
      flagbox1 = ckbox <<Get( 1 );
      flagbox2 = ckbox <<Get( 2 );
      1; // On Close returns non-zero, so the window closes.
   ),
```

```
   ckbox = Check Box( {"Check Box 1", "Check Box 2"} ),
   ckbox << Set( 1, 1 ),
   ckbox << Set( 2, 0 )
);
```

Combo Box

`Combo Box()` creates a drop-down list.

```
Combo Box( {"item 1", "item 2", ...}, <script> );
```

"item1" and "item2" are strings in the drop-down list.

The `Editable` argument allows the user to enter text in the combo box. The following example creates a new window and an editable combo box with the list items "one", "two", and "three". The script prints the selected item name and index number to the log.

```
win = New Window( "Combo Box",
   cb = Combo Box(
      {"One", "Two", "Three"},
      Editable,
      selection = cb << GetSelected();
      Print( "Selected: " || selection );
      Print( "Index: " || Char( cb << Get() ) );
   )
);
```

Note that closing the window for an editable combo box can produce unexpected results. On Windows, the `Print()` script runs after you select an item from the list or enter the item name. On macOS, the `Print()` script runs after you select an item from the list or enter the item name and then press Return.

For cross-platform compatibility, include `!Is Empty` to test for an existing combo box before running the script.

```
win = New Window( "Combo Box",
   cb = Combo Box(
      {"One", "Two", "Three"},
      Editable,
      If( !Is Empty( cb ),
         selection = cb << GetSelected();
         Print( "Selected: " || selection );
         Print( "Index: " || Char( cb << Get() ) );
      )
   )
);
```

To make an item selectable or unselectable, send the `<<Enable Item` message to `Combo Box()`.

```
cb << Enable Item( 2, 0 ) // makes the second item unselectable
```

Note: If the editable combo box has no empty strings, nothing is selected by default. If the editable combo box has one or more empty strings, the first empty string is selected by default

Global Box

`Global Box` shows the name and current value of a JSL global variable.

```
Global Box( name );
```

The user can assign a new value to the global variable by editing the value directly in the window and pressing Enter or Return to commit the change. Global boxes automatically update the displayed value of a variable when the variable changes.

```
ex = Sqrt( 4 );
win = New Window( "Global", Global Box( ex ) );
```

The preceding script creates a new window named “Global” that displays the following result:

ex=2

The following example shows the value of the myGlobal variable on the graph:

```
myGlobal = 6;
   win = New Window( "Global Box",
      V List Box(
          gr = Graph Box(
              Frame Size( 300, 300 ),
              X Scale( 0, 10 ),
              Y Scale( 0, 10 ),
              Y Function( x, x ),
              Text( {5, 5}, "My global is ", myGlobal )
          ),
          Global Box( myGlobal ) // value of myGlobal
      )
   );
```

Note: When you are using `Global Box()`, remember that every time the variable is changed, JMP remeasures, reshows, and updates the window. Depending on the number of `Global Box()` objects, changes to the variable can affect the refresh rate of the window. JMP recommends avoiding numerous global boxes in deliverable scripts. Global boxes can be replaced with text boxes that the script can manually update when necessary.

List Box

`List Box()` creates a display box that shows a list box of items that can be selected.

```
List Box({"item 1", "item 2", ...}, <Width( n )>, <MaxSelected( n )>,
  <NLines( n )>, <script>);
```

Item names are case sensitive by default. `Width` is measured in pixels. `MaxSelected` is the maximum number of items that might be selected in the list box. `NLines` is the number of lines to display in the box, with a default value of 3.

Add items to a list box by either using `Append` or `Insert`:

```
win = New Window( "test", lb = List Box( {"a", "e"} ) );
lb << Append( {"f", "g"} ); // result is a, e, f, g
lb << Insert( {"b", "c", "a", "d"}, 1 ); // result is a, b, c, d, e, f, g
```

`Append` always adds the list to the end of the list box. `Insert` adds the list after the position specified.

You can have two mutually exclusive list boxes, so that the item you select in one box is deselected when you select an item in the other box. Send the `Clear Selection` message to deselect the item.

```
win = New Window( "Each Box Clears the Other Box",
   window:la = List Box(
      {"broccoli", "pepper", "spinach"},
      <<Set Script( window:lb << Clear Selection )
   ),
   window:lb = List Box(
      {"avocado", "pumpkin", "tomato"},
      <<Set Script( window:la << Clear Selection )
   )
);
```

A list box can also contain an image. In the following example, an image from the user's computer appears in the "first" list item. The JMP nominal icon appears in the "second" list item.

```
win = New Window( "List Box with Image",
    List Box(
        {{"first", "$SAMPLE_IMAGES/pi.gif"}, {"second", "nominal"}},
        width( 200 ),
    )
);
```

To set the width of the list box, use the `width` argument.

```
listA = {"a", "b", "c",
"This is a very, very long name like Mary Ann Johnston."};
New Window( "Example",
```

```
// set the width to the length of the string
  V List Box( listBoxA = List Box( listA, width( 0 ) ) ) );
```

To set the font of the list box, use the `Set Base Font` message.

```
New Window( "Example",
   fontobj = lb = List Box(
      {"First Item", "Second Item", "Third Item"},
      width( 200 ),
      max selected( 2 ),
      nlines( 6 )
   )
);
Wait( 2 );
fontobj << Set Base Font( "Title" );
```

`Set Base Font` also works with `Bar Seg`, `Col List Box()`, `Contour Seg`, `Filter Col Selector`, `Hier Box()`, `Number Edit Box()`, `Outline Box()`, `Poly Seg`, `Text Box()`, `Text Edit Box()`, and `Text Seg`.

Mouse Box

`Mouse Box()` creates a box that can make JSL callbacks for dragging and dropping, marking, or clicking and tracking mouse actions.

```
Mouse Box( display box args, messages );
```

The following example is from the DragDrop.jsl sample script in the Samples/Scripts folder. One list box contains “a”, “b”, and “c”. The other list box contains “d”, “e”, “f”. You can drag a letter from one list box to the other list box.

Note: Mouse boxes support only one display box argument.

```
win = New Window( "Drag and drop between list boxes",
   H List Box(
      MouseBox(
         one = List Box( {"a", "b", "c"} ),

         // if the mouse is over the preceding list box, drop on the other box
         <<setTrack(
            (two << parent) << setDropEnable( 1 );
            (one << parent) << setDropEnable( 0 );
         ),

         // watch the mouse movement (not clicks)
         <<setTrackEnable( 1 ),
```

```
            // when a drop happens...
            <<setDropCommit(
               Function( {this, clickPt, text},
               // find out what is selected in the other box
                  x = two << getSelected;
                  If( Words( text, "\!t" ) == x,
                     two << removeSelected; // and take it...
                     one << append( x ); // and add it to me
                  ,
                     Print( "Drag from the other box please." )
                  );
               )
            )
         ),
         Spacer Box( size( 20, 20 ) ),

         /* same comments as above with reversed logic
         for current box versus other box */
         MouseBox(
            two = List Box( {"d", "e", "f"} ),
            <<setTrack(
               (one << parent) << setDropEnable( 1 );
               (two << parent) << setDropEnable( 0 );
            ),
            <<setTrackEnable( 1 ),
            <<setDropCommit(
               Function( {this, clickPt, text},
                  x = one << getSelected;
                  If( Words( text, "\!t" ) == x,
                     one << removeSelected;
                     two << append( x );
                  ,
                     Print( "Drag from the other box please." )
                  );
               )
            )
         )
      )
   );
```

Number Col Edit Box

`Number Col Edit Box()` creates a display box that shows editable numbers in a column. The numeric entries can be in list or matrix form.

```
Number Col Edit Box("title", numbers);
```

The following script creates number edit boxes in which the user enters values. The `Return Result` message extracts the values.

```
x = y = z = 0;
win = New Window( "Number Col Edit Box",
   <<Modal,
   <<Return Result,
   Table Box(
      neb = Number Col Edit Box( "values", {x, y, z} )
   )
);
   {neb = {2, 4, 6}, Button( 1 )}
```

Number Edit Box

`Number Edit Box()` creates an editable number box that initially contains the value argument. You can also set the width of the box in characters.

```
Number Edit Box( initValue, <width>);
```

The following script shows an editable box that initially contains the number "42". The number that the user types is printed to the log.

```
win = New Window( "Set a Value in Number Edit Box",
   <<Modal,
   <<Return Result,
   Text Box( "Set this value" ),
   variablebox = Number Edit Box( 42 ),
   Button Box( "OK" ),
   Button Box( "Cancel" )
);
Write( win["variablebox"] ); // create a subscript to the variablebox variable
   33 // the user typed "33" in the number edit box
```

The following script shows how to make the initial value a date, and enables the user to choose a different date.

```
New Window( "Date/Time Selector",
      <<Modal,
      neb = Number Edit Box(
            Today(),
            23,
            <<Set Format( Format( "m/d/y h:m:s", 23, 0 ) )
      )
);
```

Popup Box

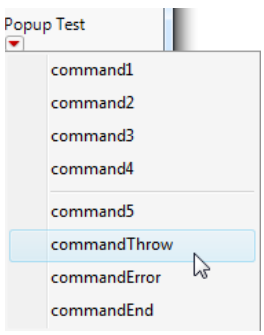
`Popup Box()` creates a red triangle menu that contains menu item commands.

```
Popup Box( {"command1", script1, "command2", script2, ...} );
```

The following example stores a command list in a variable and then uses `Popup Box()` to display the items:

```
List = {
"command1", Print( "command1" ),
"command2", Print( "command2" ),
"command3", Print( "command3" ),
"command4", Print( "command4" ),
"", Empty(), // make a separator line
"command5", Print( "command5" ),
"commandThrow", Throw( "commandThrow1" ),
"commandError", Sqrt( 1, 2, 3 ),
"commandEnd", Print( "commandEnd" )};

win = New Window( "Example",
  Text Box( "Popup Test" ),
  Popup Box( List );
);
```

Figure 11.11 Red Triangle Menu



Note that you can disable and re-enable the menu using the `Enable( Boolean )` message. An argument of 1 turns the menu on, and an argument of 0 turns the menu off. Using the previous example, assign the popup box to a variable and then send messages to it:

```
win = New Window( "Example",
  Text Box( "Popup Test" ),
  mymenu = Popup Box( List );
);

mymenu << Enable( 0 ); // disabled menu
```

You can also add a popup menu to an outline box. See "Outline Box" for an example.

Radio Box

`Radio Box()` creates a list of radio buttons.

```
Radio Box({"item 1", "item 2", ...}, <script>);
```

Only one item in the radio box can be selected at any time.

The following script creates "A", "B", and "C" radio buttons and selects the second box.

```
win = New Window( "Radio Box",
   V List Box(
      rb = Radio Box( {"A", "B", "C"}, <<Set( 2 ) ),
   )
);
```

"New Window Example" also provides an example of creating radio boxes.

Slider Box

`Slider Box()` creates a slider control for picking any value for the variable, within the minimum and maximum range that you specify.

```
Slider Box( min, max, global variable, script, <Set Width( n )>, <Rescale
   Slider( min, max )> );
```

`Set Width` specifies the width of the slider box. `Rescale Slider` specifies the minimum and maximum values.

When the slider is moved, the value given by the current position of the slider is assigned to the global variable. Thus, `Slider Box()` is another way to parameterize a graph.

```
ex = .5;
win = New Window( "Slider Box",
   tb = Text Box( "Value: " || Char( ex ) ),
   sb = Slider Box(
      0,
      1,
      ex,
      tb << Set Text( "Value: " || Char( ex ) )
   )
);
sb << Set Width( 100 ) << Rescale Slider( 0, .8 );
```

The following script creates multiple slider boxes without using a unique global variable to hold the current value for each one:

```
sb1 = Slider Box( 1, 10 );
```

```
// create a slider box with only the min and max value
sb2 = Slider Box( -10, 10 );
sb1 << Set Function( // set the script or the function
   Function( {this},
      {},
      Show( this << Get, sb2 << Get )
   )
);
sb2 << Set Script( Show( sb2 << Get, sb1 << Get ) );
New Window( "Values", sb1, sb2 ); // place the slider boxes in a window
```

As you move the slider, the value is printed to the log.

See "Example of Slider Boxes and Range Slider Boxes" for another example of slider boxes.

Spacer Box

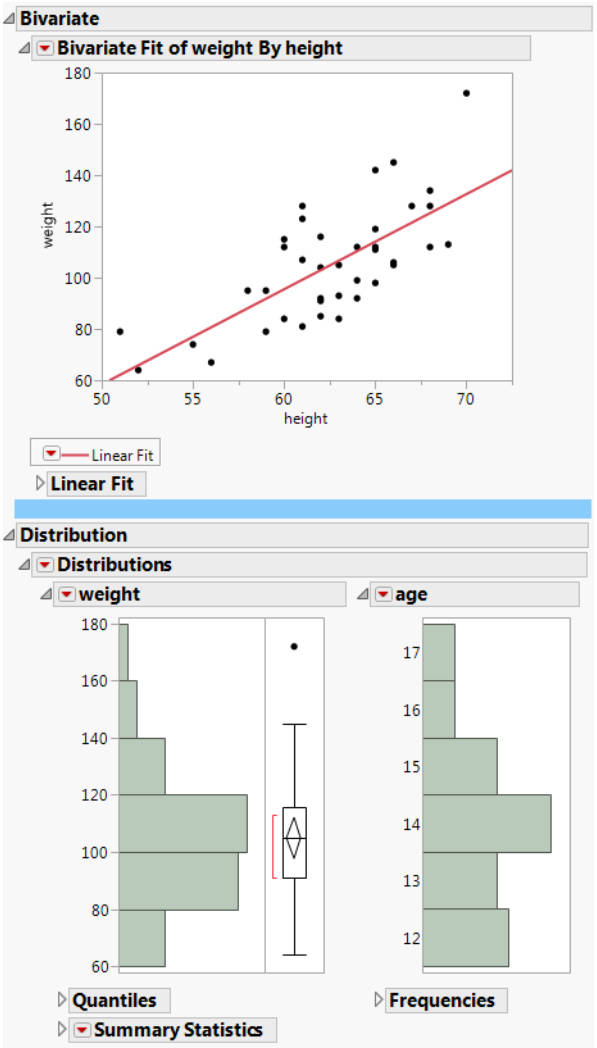
`Spacer Box()` creates space between display boxes.

```
Spacer Box(<size(horizontal_pixels,vertical_pixels)>, <color(color)>)
```

The following script places a 450 pixel wide and 15 pixel tall blue spacer box between the two outline boxes.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Spacer Box",
   V List Box(
      Outline Box( "Bivariate",
         biv = dt << Run Script( "Bivariate" ),
         // Color string consists of percentages of RGB values
         Spacer Box( Size( 450, 15 ), Color( {.53, .80, .98} ), )
      ),
      Outline Box( "Distribution",
         V List Box( dist = dt << Run Script( "Distribution" ), )
      )
   )
);
```

Figure 11.12 Spacer Box Example



String Col Edit Box

`String Col Edit Box()` creates a column named title in a table containing the string items listed. The string boxes are editable.

```
String Col Edit Box( "title", {strings} );
```

The following script creates a column of editable text boxes, preceded by the title “Names”:

```
a = b = c = "";
win = New Window( "String Col Edit Box",
   <<Modal,
   <<Return Result,
```

```
   Table Box(
      seb =
      String Col Edit Box(
         "Names",
         {a, b, c}
         )
      )
);
   {seb = {"apple", "orange", "banana"}, Button( 1 )}
```

Tab Box and Tab Page Box

`Tab Page Box()` organizes the title and contents of the page in one display box. When `Tab Page Box()` is inside `Tab Box()`, a tabbed window with multiple tabbed pages appears.

```
Tab Box( Tab Page Box( Title( "page title 1" ), <options>, contents of
   page 1), Tab Page Box( "page title 2", <options>, contents of page 2),
   ...);
```

The following script creates three tab boxes:

```
win = New Window( "Tab Box",
   Tab Box(
      Tab Page Box(
         Title( "First page" ),
         Text Box( "first line of first page" ),
         Text Box( "second line of first page" )
      ),
      Tab Page Box(
         Title( "Second page" ),
         Text Box( "first line of second page" ),
         Text Box( "second line of second page" )
      ),
      Tab Page Box(
         Title( "Third page" ),
         Text Box( "first line of third page" ),
         Text Box( "second line of third page" )
      )
   )
);
```

Figure 11.13 Tab Boxes

You can specify which tab should be selected by sending the `Set Selected(n)` message, where *n* is the tab number, to the tab box object.

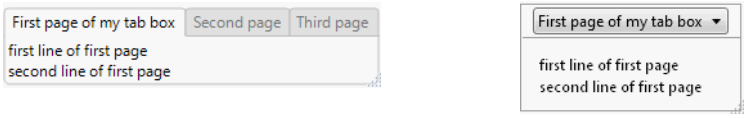
The `Set Style` message lets you select the visual appearance of the tab box. The default value is `tab`. Other options include the following:

- `combo` creates a combo box.
- `outline` creates an outline node.
- `vertical spread` displays the tab title vertically.
- `horizontal spread` displays the tab title horizontally.
- `minimize size` bases the tab style on the width of the tab title. In the following example, the first tab title is fairly long:

```
win = New Window( "Tab Box",
   tb = Tab Box(
      Tab Page Box(
         Title( "First page of my tab box" ),
         Text Box( "first line of first page" ),
         Text Box( "second line of first page" )
      ),
      Tab Page Box(
         Title( "Second page" ),
         Text Box( "first line of second page" ),
         Text Box( "second line of second page" )
      ),
      Tab Page Box(
         Title( "Third page" ),
         Text Box( "first line of third page" ),
         Text Box( "second line of third page" )
      )
   )
);
tb << Set Style( "minimize size" );
```

Minimizing the size of the tab box converts the tab box into a combo box. Figure 11.14 compares the default and minimized tab boxes.

Figure 11.14 Default Tab Box (Left) and Minimized Tab Box (Right)



For `Tab Page Box()` examples, see "Examples of Creating a Dashboard from Two Reports".

Tips:

- Unlike with other display boxes, you must define the title in the `Title()` function.
- The `Set Style` message works for both bare words and quoted words. However, you cannot assign the style to a variable and then pass the variable as the argument.
- You can combine `Tab Box()` with `Splitter Box()` to create docked tabs, which can be dragged to drop zones in a window to create a different layout. See "Examples of Creating a Dashboard from Two Reports".
- To delete a tab, you can either send `Delete(index)` to the tab box, or you can send `Delete Box()` to a tab page box. Note that the entire display box (all tabs) is deleted, not just a single tab.
- Use the `Visibility("Collapsed")` message to hide a tab.
- When tabs in a tab list are not wide enough for the titles, send the `Set Overflow Enabled(1)` message to `Tab Box()`. A triangle symbol is shown to the right of a tab list, and the tab titles are truncated using ellipses.
- When `Tab Page Box()` is created or dragged outside a `Tab Box()`, it is a stand-alone container. The title appears in a shaded box at the top of the page, similar to the title in `Sheet Part()`; the title does not appear in an interactive tab.
- See "Write Tab Box and Tab Page Box Scripts" for more information about the differences between `Tab Box()` and `Tab Page Box()`.
- In Bubble Plot and Graph Builder scripts, set the `Fit to Window` message to `"On"` to resize the window when you resize the graph in `Tab Box()`.
- To close a tab, first make a tab closeable by sending `Closeable(1)` to the tab page box and then send the `Close` message.
- Closeable tabs can be closed with the middle mouse button.

Text Box

`Text Box()` creates a non-editable text box. Text boxes are frequently used as labels of other controls.

```
Text Box( "text" );
```

You can format the text with HTML tags. For example, the following script formats text in bold:

```
win = New Window( "Formatted Text",
  Text Box( "This is <b>bold</b> text.",
  <<Markup) );
```

Make sure you close nested tags correctly and escape the quotes as shown here:

```
"This is <font color=\!"Blue\!" face=\!"Lucida Sans Unicode\!"
  size=\!"24\!">blue</font> text. This is a <background
```

```
color=\!"yellow\!">yellow</background> background. This is <font
color=\!"Blue\!"><background color=\!"yellow\!">blue text on a yellow
background</background></font>."
```

See "Example of Specifying Text and Background Colors" for another example.

Note: Rotated text boxes support `Markup` text except for larger text boxes with multiple lines, multiple formats or justifications applied.

Text Edit Box

`Text Edit Box()` creates an editable text box.

```
Text Edit Box ( "text" );
```

You can add a script to a text edit box by sending it a script message. This is usually most convenient at the time the box is created. Simply add the script message as the last argument.

The following example prints "Changed" in the log each time the text edit box is changed:

```
win = New Window( "Text Edit Box",
   Text Edit Box( "Change Me", <<Script( Print( "Changed" ) ) )
);
```

Assign a reference to the text edit box object to access its contents. The following example prints the value of the text edit box to the log each time the box changes:

```
win = New Window( "Text Edit Box",
   teb = Text Edit Box( "Change Me",
      <<Script( Print( teb << Get Text ) )
   )
);
```

Text edit boxes are empty. However, you can specify placeholder text as a visual cue for the user to enter a value in the box. The placeholder text is only a hint and does not affect the value of the text field.

The following example produces the text edit box shown in Figure 11.15. The `Hint` option returns `"mm/dd/yyyy"` and formats it as light gray text.

```
win = New Window( "Text Edit Box",
   Text Edit Box( "Current Date" ),
   Text Edit Box( "", Hint( "mm/dd/yyyy" ) )
);
```

Figure 11.15 Text Edit Box with Placeholder Text

Tip: To omit the border around the editable box and the white background color, send the `White Box Style( 0 )` message to the text edit box.

If you need to use a text edit box for your user to enter passwords, you can apply a style to the text edit box to replace all characters entered with an asterisk. For example, the following example creates a text edit box whose value is the string "a". When a user types a new string into the text edit box, each character is displayed as an asterisk, and the message "Changed!" is printed to the log.

```
win = New Window( "Text Edit Box",
   teb = Text Edit Box( "a",
      Password Style( 1 ),
      Set Script( Print( "Changed!" ) )
   )
);
```

You can also send a text edit box a message to either use password style or to stop using password style.

```
q << Password Style( 1 ); // set the text edit box to password style
q << Password Style( 0 ); // set the text edit box to standard style
```

Filtering Display Boxes

`Data Filter Source Box()` and `Data Filter Context Box()` work together to filter data in multiple graphs. `Data Filter Source Box()` defines which graph is the "source" of the selection filter. When data are selected in the source graph, graphs in the common `Data Filter Context Box()` are updated.

Data Filter Context Box

`Data Filter Context Box()` creates a display box that enables you to filter data in multiple graphs. The function determines the number of graphs that the filter applies to, for example, all graphs in a report.

```
Data Filter Context Box( display box args );
```

In the following example, the `H List Box()` that contains the graphs is wrapped in `Data Filter Context Box()`. After you run the script, select values in the filter to see both graphs update.

```
dt = Open( "$SAMPLE_DATA/Hollywood Movies.jmp" );
win = New Window( "Shared Local Filter",
   Data Filter Context Box(

   // enclose the h list box that contains distribution and graph builder
      H List Box(
```

```
            dt << Data Filter( Local ), // add the local data filter
            dt << Distribution(
               Weight( :Profitability ),
               Nominal Distribution( Column( :Lead Studio Name ) ),
               Nominal Distribution( Column( :Genre ) ),
               Histograms Only
            ),
            dt << Graph Builder(
               Variables(
                  X( :Genre ),
                  Y( :Domestic Gross ),
                  Y( :Foreign Gross, Position( 1 ) )
               ),
               Show Control Panel( 0 ),
               Elements(
                  Bar(
                     X,
                     Y( 1 ),
                     Y( 2 ),
                     Legend( 2 ),
                     Bar Style( "Side by side" ),
                     Summary Statistic( "Mean" )
                  ),
                  Frequencies( 0 ),
               )
            )
         )
      )
   );
```

Data Filter Source Box

`Data Filter Source Box()` and `Data Filter Context Box()` work together to filter data in multiple graphs. `Data Filter Source Box()` defines which graph is the "source" of the selection filter. When data are selected in the source graph, graphs in the common `Data Filter Context Box()` are updated.

`Data Filter Source Box(` *`display box args`* `);`

The following script lets you select data in Distribution graphs, and only that data is shown in the Graph Builder graph:

```
dt = Open( "$SAMPLE_DATA/Hollywood Movies.jmp" );
win = New Window( "Shared Local Filter",
   Data Filter Context Box(

   // enclose data filter source box and the graph builder platform
      H List Box(
```

```
            Data Filter Source Box(
               dt << Distribution( // use distribution as the source graph
                  Weight( :Profitability ),
                  Nominal Distribution( Column( :Lead Studio Name ) ),
                  Nominal Distribution( Column( :Genre ) ),
                  Histograms Only
               )
            ),
            dt << Graph Builder(
               Variables(
                  X( :Genre ),
                  Y( :Domestic Gross ),
                  Y( :Foreign Gross, Position( 1 ) )
               ),
               Show Control Panel( 0 ),
               Elements(
                  Bar(
                     X,
                     Y( 1 ),
                     Y( 2 ),
                     Legend( 2 ),
                     Bar Style( "Side by side" ),
                     Summary Statistic( "Mean" )
                  ),
                  Frequencies( 0 ),
               )
            )
         )
      )
   );
```

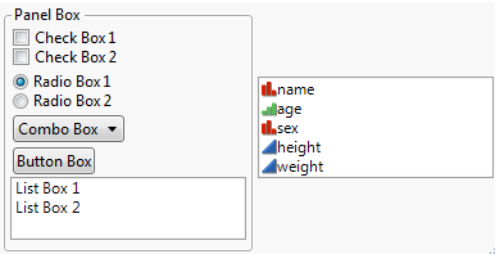
Examples of Combined Display Boxes

The following example generates a sample of many controls illustrated in the previous sections.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Window Controls",
   Line Up Box( N Col( 2 ), Spacing( 3 ),
      Panel Box( "Panel Box",
         Check Box( {"Check Box 1", "Check Box 2"} ),
         Radio Box( {"Radio Box 1", "Radio Box 2"} ),
         Combo Box( {"Combo Box"} ),
         Button Box( "Button Box" ),
         List Box( {"List Box 1", "List Box 2"} )
      ),
      Col List Box( "all" )
```

```
    )
);
```

Figure 11.16 Example of Many Interactive Display Elements



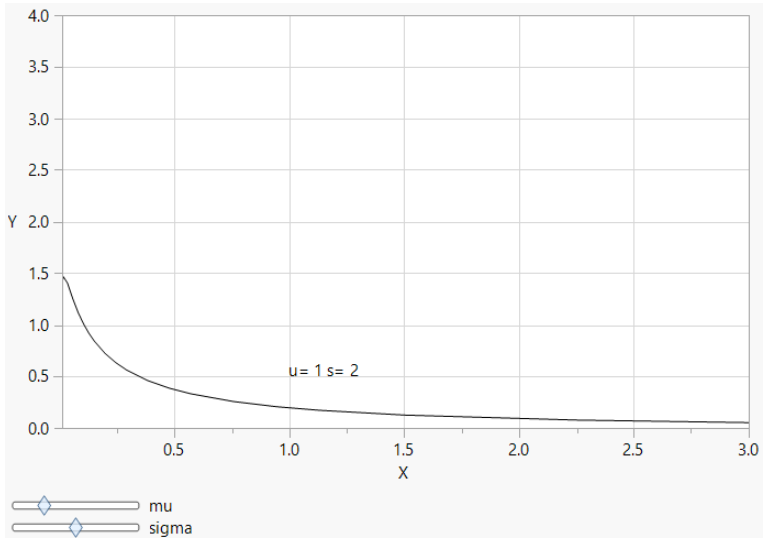
Example of Slider Boxes and Range Slider Boxes

The following example combines a graph with two sliders and a button by gluing the graph box, two horizontal boxes, and a button together in a vertical list box:

```
// Slider LogNormal
lU = 1;
lS = 2;
win = New Window( "LogNormal Density",
   V List Box(
      gr = Graph Box( // define the arguments and script for the graph box
         Frame Size( 500, 300 ),
         X Scale( 0.01, 3 ),
         Y Scale( 0, 4 ),
         XAxis( Show Major Grid ),
         YAxis( Show Major Grid ),
         Y Function(
            Exp( -(Log( x ) - Log( lU )) ^ 2 / (2 * lS ^ 2) ) / (lS * x *
            Sqrt( 2 * Pi() )),
            x
         );

         /* display the specified text at 1 on the x axis
         and .5 on the y axis */
         Text( {1, .5}, "u= ", lU, " s= ", lS );
               ),

      /* define the minimum and maximum range
      insert the value of lU and updates the display as the slider moves */
      H List Box( Slider Box( 0, 4, lU, gr << Reshow ), Text Box( "mu" ) ),
         H List Box(
         Slider Box( 0, 4, lS, gr << Reshow ),
         Text Box( "sigma" )
```

```
        ),
        Button Box( "Reshow", gr << Reshow )
    )
);
Show( gr );
gr << Reshow; // update the display of the Graph Box
```

Figure 11.17 Example of Sliders and Buttons in a Report Window

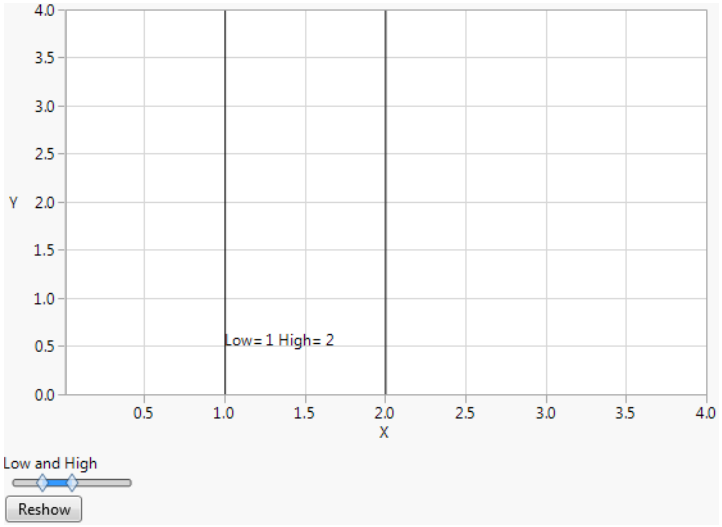


The following script uses a `Range Slider Box()` instead of a `Slider Box()` to perform the same function:

```
lLow = 1;
lHigh = 2;
win = New Window( "Range Slider",
   V List Box(
      gr = Graph Box(
         Frame Size( 500, 300 ),
         X Scale( 0.01, 4 ),
         Y Scale( 0, 4 ),
         XAxis( "Show Major Grid" ),
         YAxis( "Show Major Grid" ),
         X Function( lLow, x );
         X Function( lHigh, x );
         Text( {1, .5}, "Low= ", lLow, " High= ", lHigh );
      ),
      V List Box(
         Text Box( "Low and High" ),
         sb = Range Slider Box( 0, 4, lLow, lHigh, gr << Reshow )
```

```
      )
   ),
   Button Box( "Reshow",
      lLow = 1;
      lHigh = 2;
      gr << Reshow;
      sb << Reshow;
   )
);
Show( gr );
gr << Reshow;
```

Figure 11.18 Example of Using Range Slider Box

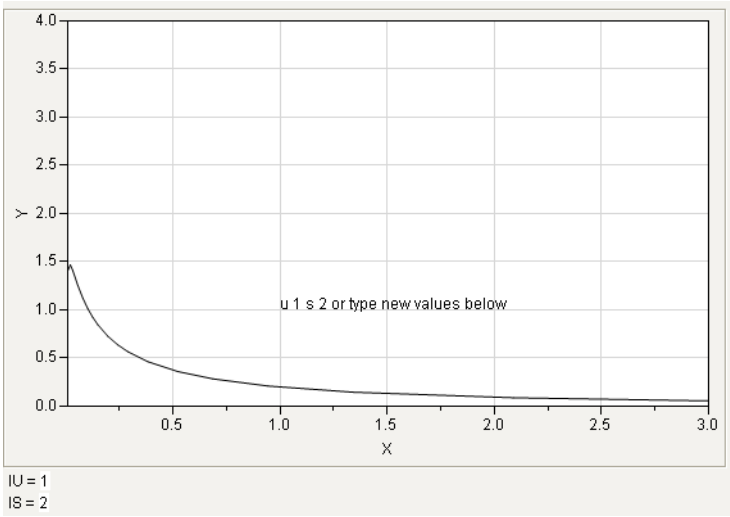


The script below is similar but uses `Global Box()` as an editable text box instead of two `Slider Box()` controls:

```
// Global LogNormal
lU = 1;
lS = 2;
win = New Window( "LogNormal Density",
   V List Box(
      gr = Graph Box( // define the arguments and script for the Graph Box
         Frame Size( 500, 300 ),
         X Scale( 0.01, 3 ),
         Y Scale( 0, 4 ),
         XAxis( "Show Major Grid" ),
         YAxis( "Show Major Grid" ),
         Y Function(
            Exp( -(Log( x ) - Log( lU )) ^ 2 / (2 * lS ^ 2) ) / (lS * x *
```

```
            Sqrt( 2 * Pi() )),
            x
         );

         // display the specified text at 1 on the x and y axes
         Text( {1, 1}, "u= ", lU, " s= ", lS, " or type new values below" );
      ),

      // display the lU variable value in the first H List Box
      H List Box( Global Box( lU ) ),

      // display the lS variable value in the second H List Box
      H List Box( Global Box( lS ) )
   )
);
```

Figure 11.19 Example of Using a Global Box Instead of Sliders



Example of a Calendar and Date Selector

You can create a calendar or date selector using `Calendar Box()` and `Number Edit Box()`.

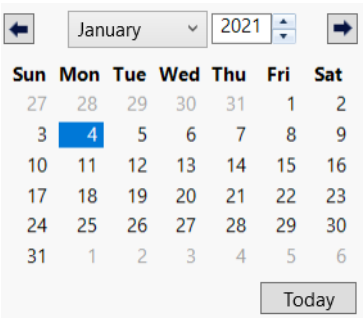
Calendar Box

With `Calendar Box()`, you can select dates from a calendar and input the year. The following example creates a calendar with October 5, 1989 initially selected. You can select any date 60 days before or 60 days after October 5, 1989.

```
win = New Window( "Calendar Box", cal = Calendar Box() );
date = Date MDY( 01, 04, 2021 );
```

```
cal << Date( date );
cal << Show Time( 0 ); // omit the time

/* earliest date that can be selected is 60 days before 01/04/2021
"start" truncates the value so the time is not included */
cal << Min Date( Date Increment(date, "Day", -60, "start" ) );

// latest date that can be selected is 60 days after 01/04/2021
cal << Max Date( Date Increment(date, "Day",  60, "start" ) );
// print the abbreviated date to the log
cal << Set Function( Function( {this}, Print( Abbrev Date(this << Get Date())
  ) ) );
```

Figure 11.20 Creating a Calendar



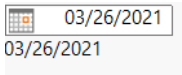
Number Edit Box

With `Number Edit Box()`, you can enter a date in the box or click the calendar button to select the date from a calendar.

The following example shows how to create an interactive calendar using `Number Edit Box()`. The user enters a date in the box or clicks the calendar to select a date from the calendar. The number appears in a text box below the number edit box. “01/04/2021” is initially selected.

```
f = Function( {this}, // callback
   textbox << Set Text( Format( this << Get, "m/d/y" ) ) // get the date
);
win = New Window( "Date Selector",
   numbox = Number Edit Box( 0, 20, << Set Function( f ) ),
   textbox = Text Box( "" ) // specified date
);
numbox << Set Format(
   Format( "m/d/y", 12 ) // date format and number of characters displayed
   );
numbox << Set( Date MDY( 01, 04, 2021 ) ); // default date
```

Figure 11.21 Creating a Date Selector



Note that a callback is assigned to the `f` variable. In the callback, `this` refers to the number edit box, the box that calls the function later. Referring to `this` prevents the need for using the `numbox` variable within the callback. The `Get` message is sent to the `this` argument to get the specified date.

Example of Constructing a Summary Results Report

The following example uses the `Summarize()` function to collect summary statistics on the Height column of Big Class.jmp and then uses display box constructors to show the results in a nicely formatted window.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Summarize(
   a = by( age ),
   c = count,
   sumHt = Sum( Height ),
   meanHt = Mean( Height ),
   minHt = Min( Height ),
   maxHt = Max( Height )
);
win = New Window( "Summary Results",
   Table Box(
      String Col Box( "Age", a ),
      Number Col Box( "Count", c ),
      Number Col Box( "Sum", sumHt ),
      Number Col Box( "Mean", meanHt ),
      Number Col Box( "Min", minHt ),
      Number Col Box( "Max", maxHt )
   )
);
```

This produces the summary results shown in Figure 11.22.

Figure 11.22 Producing a Customized Summary Report

Age	Count	Sum	Mean	Min	Max
12	8	465	58.125	51	66
13	7	422	60.2857	56	65
14	12	770	64.1667	61	69
15	7	452	64.5714	62	67
16	3	193	64.3333	60	68
17	3	200	66.6667	62	70

The following script saves the report as a Microsoft PowerPoint presentation.

```
Show Properties( win );
win << Save Presentation( "$DOCUMENTS/Summary.pptx" );
```

Interact with a Closing Window

A user can close a window by clicking the OK or Cancel button or by clicking the red Close button in the top corner of the window. A message can also be sent to close the window. `On Validate` and `On Close` expressions determine what happens you interact with the window.

- `On Validate` evaluates its expression *when the OK button is clicked*. The expression is not evaluated for any other button click or action. If the expression evaluates as true, the window closes. If the expression evaluates as false, the window is not closed. The script for an OK button runs when you click the button, no matter what the `On Validate` expression does.
- `On Close` evaluates its expression *right before the window closes*. The `On Close` return value is like the `On Validate` return value and can prevent the window from closing. However, the best practice for `On Close` is to always return 1 so that you do not accidentally create a window that cannot be closed.

  If the window closes, `On Close` is evaluated. `On Close` is not evaluated if the `On Validate` expression returns false, because the window is not closed.

Tips:

- `On Close` works for modal or non-modal windows. `On Validate` works only for modal windows.
- Do not use `On Close` for validation because it does not distinguish between OK and Cancel. Use `On Close` to clean up resources (such as invisible data tables) because it always runs when the window closes.
- Do not use `On Validate` to clean up resources. Use `On Close` instead or in addition to `On Validate`, because `On Validate` does not run for the Cancel button or red Close button.

On Close() Example

Suppose that a non-modal window is a report window, and the resource that needs cleaning up is an invisible data table. The `On Close` script closes the data table. This is good because the user might not know that the data table is open and has no easy way to close it.

```
dt = New Table( "Untitled", << Invisible,
    New Column( "x", Set Values( [1, 2, 3, 4] ) ),
    New Column( "y", Set Values( [4, 2, 3, 1] ) ) )
);

dt << Bivariate(
    y( dt:y ), x( dt:x ), <<On Close( Close( dt, "nosave" ) )
```

```
);
```

Before the script is run, no data tables are open. The Untitled invisible table is created and displayed in the Home Window and in the Data Tables list above the script. Then the graph is created. After you close the report window, the invisible table is closed by `On Close`.

On Validate() Example

Suppose that a modal window contains a question that the user must answer before closing the window. The following example shows how to validate the input when the user closes the window.

```
win = New Window( "Validate Example",
   <<Modal,
   <<Return Result,
   <<On Validate(
      If( (!Is Missing( variablebox << Get ) & 40 <= variablebox << Get <= 50),
         ( 1 ), // else the user did not enter the correct answer
         tb << Set Text(
            Match( Random Integer( 1, 3 ),
               1, "Are you sure?",
               2, "Guess again.",
               3, "Please read the question."
            )
         );
         ( 0 );
      )
   ),
   Text Box( "I'm thinking of a number between 40 and 50. What's your guess?"
   ),
   variablebox = Number Edit Box( . ),
   H List Box( Button Box( "OK" ), Button Box( "Cancel" ) ),
   tb = Text Box( "" )
);

New Window( "Modal Window",
   <<Modal,
   Text Box(
      If(

         // user clicked Cancel or the red Close button
         win["Button"] == -1, "Try again when you feel better.",
         win["variablebox"] == 42, "Good job.", // user entered 42
         "Take the test again." // user did not enter 42 and clicked OK
      )
   )
);
```

Update an Existing Display

Sometimes, you do not know how many display boxes will appear in a future report. For example, you might write a generic script that analyzes and reports on one or more variables. You do not know how many display boxes are needed, because the number of variables can change from one run of the script to the next.

The following sections describe how to add and delete display boxes from a report using `Append()`, `Prepend()`, `Delete()`, and `Sib Append()`.

Append

Use the `Append` message to add a display box to the bottom of an existing display. In the script, construct a single, empty box, then append boxes to it for each variable in the analysis. You can also use the `Append` message to modify an existing display with custom information or organization.

The following code example assumes that there is a list of effect names in the variable `effectsList`, and that each one corresponds to a column in a matrix `varprop`. In other words, `effectsList[1]` is the label for `varprop[0,1]`; `effectsList[2]` is the label for `varprop[0,2]`; and so on.

```
varprop = [0 1 2, 3 4 5, 6 7 8];
effectsList = {"one", "two", "three"};
```

First, an empty `Outline Box` containing an `H List Box` is created. The interior empty container is given the name `hb`:

```
win = New Window( "H List Box Example",
   Outline Box( "Variance Proportions", hb = H List Box() )
);
```

Then, a `for` loop steps through the `effectsList` and adds a `Number Col Box` for each element of `effectsList`:

```
For( i = 1, i <= N Items( effectsList ), i++,
   Eval(
      Substitute(
            Expr(
               hb << Append(
                  Number Col Box( effectslist[i], varprop[0, i] )
               )
            ),
         Expr( i ), i
      )
   )
);
```

`Sib Append()` is similar to `Append()` but makes the display box a sibling of the last child display box. For an example that compares `Append()` to `Sib Append()`, see Figure 11.24.

Prepend

The `Prepend` message works just like `Append` but adds the item at the beginning of the display box rather than at the end. If the display box is one of several that do not allow appending, then `Prepend` delegates the command to a child display box that can accept the command. It is fine to apply it to the top of the tree.

For example, the following example creates a Bivariate report with a button box at the top.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( height ), X( weight ), Fit Line );

/* select the first outline box in the biv report layer
send the Prepend message to the first outline box */
(biv << Report)(Outline Box( 1 )) << Prepend(

     // prepend a button box that fits a quadratic curve
     Button Box( "Click here for curve", biv << Fit Polynomial( 2 ) )
);
```

Click the **Click here for curve** button to add a quadratic curve to the graph.

You can accomplish the same thing by appending the button to the top of the tree:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( height ), X( weight ), Fit Line );
biv << Report << Prepend( // send the prepend message to the report

  // prepend the button box to the top of the report
  Button Box( "Click here for curve", biv << Fit Polynomial( 2 ) )
);
```

Delete

The `Delete` message removes the specified display box and all its children from the report. This is useful with the `Append` and `Prepend` messages for building completely dynamic displays. In the example below, a text box is replaced with another text box. In this case, the script could have used `Set Text`, but many display boxes cannot change their content.

```
win = New Window( "X",
   list = V List Box(
      t1 = Text Box( "t1" ),
      t2 = Text Box( "t2" )
   )
);
```

```
t1 << Delete;
list << Append( t1 = Text Box( "t1new" ) );
```

Note: Avoid deleting display boxes from a JMP platform. Send the `Hide( 1 )` message or the `Visibility( "Hidden" )` message to the display box that you want to hide.

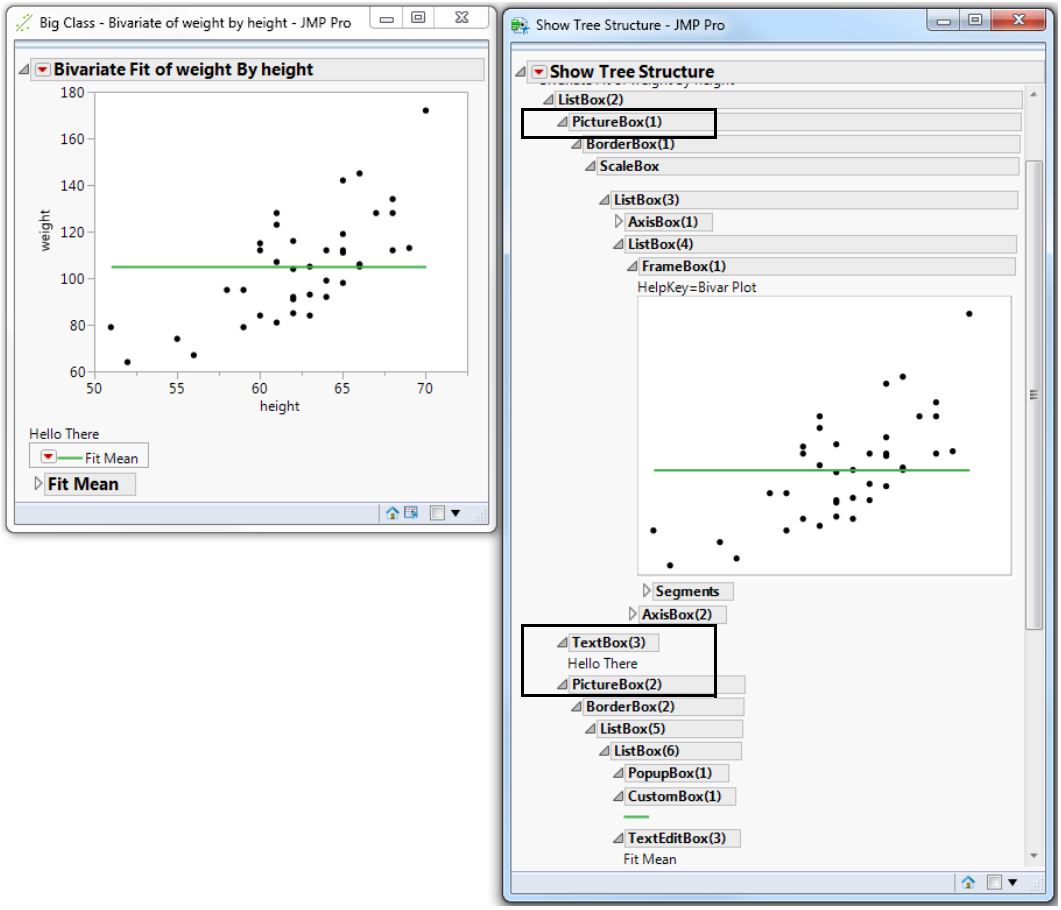
Sib Append

You can use the `Sib Append` message to add a display box immediately after an existing display box. Figure 11.23 shows two picture box trees. `Picture Box( 1 )` holds the Bivariate scatterplot. `Picture Box( 2 )` holds the Fit Mean menu, determined by seeing the green line and the Fit Mean text box.

Suppose you wanted to insert a text box in between these two boxes. You want to append a sibling to `Picture Box( 1 )`, so you send it the `Sib Append` message:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate(
   Y( :weight ),
   X( :height ),
   Fit Mean( {Line Color( {57, 177, 67} )} )
);

/* the report function takes the biv object and returns a report object.
 append the text box immediately after the first picture box */
Report( biv )[Picture Box( 1 )] << Sib Append( Text Box( "Hello There" ) );
```

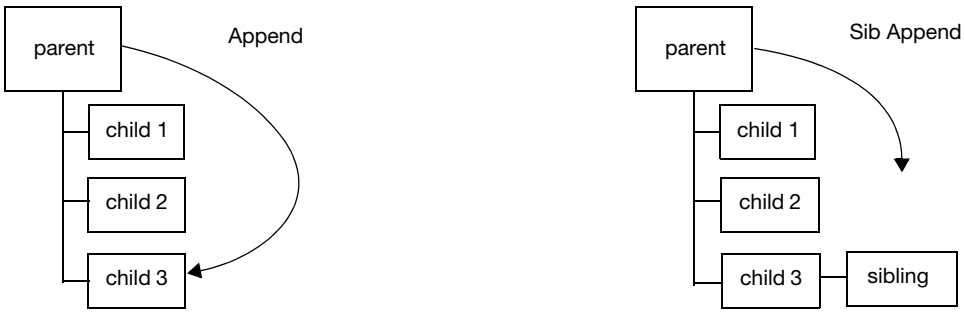
Figure 11.23 Appending a Sibling Text Box



Note: In Figure 11.23, the appended sibling text box appears below the first picture box in the Show Tree Structure View window. In the classic Show Tree Structure View window, siblings appeared next to each other in the display tree. See "View the Display Tree".

Figure 11.24 shows the difference between and `Append()` and `Sib Append()`. `Append()` makes the display box a *child* of the parent display box. `Sib Append()` makes the display box a *sibling* of the last display box.

Figure 11.24 Append (Left) and Sib Append (Right)



Note: If the parent display box is not a `V List Box()`, `Sib Append()` wraps that display box in `V List Box()`.

Set Function and Set Script

You can use the `Set Script` message to have a display box control (for example, a button box or combo box) run a script when it is clicked with the mouse.

```
win = New Window( "Set Script Example",
   ex = Button Box( "Press me" )
);
ex << Set Script( Print( "Pressed" ) );
```

The preceding script prints the following text to the log when the button box is clicked:

```
"Pressed"
```

Alternatively, you can use the `Set Function` message to have a display box control run a specific function where the first argument is the specific display box. `Set Function` enables you to build more object-oriented scripts and to create a larger system.

```
win = New Window( "Set Function Example",
   Button Box( "Press me",

      // functions specified with Set Function get 'this' display box
      << Set Function(
         Function(
            {this},
            this << Set Button Name( "Thanks" )
         )
      )
   )
);
```

The preceding script creates a button box with the name “Press me”. When the button is clicked, the function called by `Set Function` changes the name to “Thanks”.

If you have multiple buttons that can be serviced by the same script, `Set Function` is much simpler than `Set Script` because it tells you which button was pressed. In this example, both check boxes use the script that begins on the first line.

```
f = Function( {this, idx}, // idx just changed
   Write(

   /* <<Get Items returns a list of all items; indexing that list with idx
   is the name of the item that is changing */
      "\!n changing item=" || (this << Get Items())[idx] || " to " || Char(

   // <<Get returns the new value of that item
   this << Get( idx ) ) ||
      "\!n new selection=" || Char( this << Get Selected ) || "\!n"
   )// <<Get Selected returns a list of the currently checked (selected) items
);
New Window( "Example",
   H List Box( // 'f' is the 'named' function, used twice
      V List Box( Text Box( "Column 1" ), Check Box( {"a", "b"}, <<Set
   Function( f ) ) ),
      Spacer Box( size( 50, 50 ) ),
      V List Box( Text Box( "Column 2" ), Check Box( {"c", "d"}, <<Set
   Function( f ) ) )
   )
);
```

In the following example of `Set Function`, the same function is used to create the buttons.

```
New Window( "Cash Box",
   V List Box(
      H Center Box( TB = Text Box() ), // TB will show total of coins
      LB = Lineup Box( N Col( 3 ) ), // LB will hold buttons, added below
      H Center Box( // CLEAR button will reset the total
         Button Box( "CLEAR",
            total = 0;
            TB << Set Text( Char( total ) );
         )
      )
   )
);
coins = {1, 5, 10, 25, 50, 100}; // coins is a list of the button labels
total = 0;
```

```
/* loop creates the buttons and shows that the same function
is used for each */
For( iButton = 1, iButton <= N Items( coins ), iButton++,
   LB << Append( Button Box( Char( coins[iButton] ), Set Function(
   buttonFunction ) ) )
);
buttonFunction = Function( {this},
   total = total + Num( this << Get Button Name );
   TB << Set Text( Char( total ) );
);
```

The function called for any button, except CLEAR, uses the button's name to determine what to do. Rather than using the button's name, you might want to use a sibling display box by `this<<sib` or perhaps `(this<<parent)<<child` to find the first sibling of this button.

Notes:

- The `Set Script` and `Set Function` messages work for button boxes, calendar boxes, check boxes, combo boxes, list boxes, popup boxes, radio boxes, range slider boxes, slider boxes, and spin boxes.
- You cannot use both `Set Script` and `Set Function` at the same time. Use `Set Function` if you need to reference a specific display box object.

Get and Set Selected Values of Display Elements That Return Lists

You can use the `Set Selected(Item Number, <State>, <Run Script(0|1)>)` message to pre-select an item. You can use this message separately to a saved display box reference, or you can specify it inline as a list box << message.

To retrieve the selected value, use the `Get Selected` message, which returns the value of the selected item. The `Get Selected Indices` message returns the index number of the selected item.

```
antennaList = {"Dish","Helical","Polarizing","Radiant Array"};

win = New Window( "Test List", // method 1: display box reference
   listObj = List Box(
      antennaList,
      Print(
         "iList",
         listObj << Get Selected,
         listObj << Get Selected Indices
      )
   )
);
listObj << Set Selected( 2, 1 );
```

```
win = New Window( "Test List", // method 2: inline
   listObj = List Box(
      antennaList,
      <<Set Selected( 2, 1 ),
      Print(
         "iList",
         listObj << Get Selected,
         listObj << Get Selected Indices
      )
   )
);
```

Both of these scripts print the following text to the log:

```
"iList"
{"Helical"}
{2}
```

In the preceding examples, the `Print` expression is executed when the `Set Selected` message is completed. To prevent the script from running, include `Run Script(0)` as the last argument. `Run Script( 0|1 )` controls whether a display box on-change script runs after a `Set` or `Set Selected` message.

```
antennaList = {"Dish","Helical","Polarizing","Radiant Array"};
win = New Window( "Test List",
   listObj = List Box(
      antennaList,
      Print(
         "iList",
         listObj << Get Selected,
         listObj << Get Selected Indices
      )
   )
);
listObj << Set Selected( 2, 1, Run Script( 0 ) );
```

With `Run Script( 1 )`, the script is executed when the `Set` message is completed, even if the value is unchanged. (The script does not run again if the user selects the same value.) With `Run Script( 0 )`, the script does not run.

On most interactive display boxes, the script does not run on most interactive display boxes if you leave out `Run Script()`. However, on `List Box()`, the script runs by default, which is consistent with previous behavior.

Send Messages to Constructed Displays

When you assign a constructed display to a name, that name becomes a reference to the window, which in turn owns the display boxes inside it. Using subscripts, you can then send messages to the display boxes inside the window.

For example, the following script creates an interactive sine wave graph. The script automates the interaction by sending messages to the frame box inside the window. Note the frame box assignment to *tf* in the middle of the script).

```
amplitude = 1;
freq = 1;
phase = 0;
win = New Window( "Wiggle Wave",
   Graph Box(
      Frame Size( 500, 300 ),
      X Scale( -5, 5 ),
      Y Scale( -5, 5 ),
      Y Function( amplitude * Sine( x / freq + phase ), x );
      Handle(

         /* current values of phase and amplitude position
         the square handle.
         script begins after the last comma after amplitude */
         phase,
         amplitude,

         /* x and y have been set to the handle's position
         but the handle doesn't move unless the values specified
         in the first two arguments are updated */
         phase = x;
         amplitude = y;
      );

      // Handle works similarly except that the y value is .5
      Handle( freq, .5, freq = x );

      Text( // display a text string in the graph
         {3, 4},
         "amplitude: ",
         Round( amplitude, 4 ), // display the current value
         {3, 3.5},
         "frequency: ",
         Round( freq, 4 ), // display the current value
         {3, 3},
         "phase: ",
         Round( phase, 4 ) // display the current value
```

```
      );
   )
);
tf = win[Frame Box( 1 )]; // get the frame box (the graph)
For( amplitude = -4, amplitude < 4, amplitude += .1,// animate the amplitude
   tf << Reshow // force the graph to update
);
amplitude = 1; // use for loops for more complex movement
freq = 1;
phase = 0;
For( i = 0, i < 1000, i++,
   amplitude += (Random Uniform() - .5);
   amplitude = If(
      amplitude > 4, 4,
      amplitude < -4, -4,
      amplitude
   );
   freq += (Random Uniform() - .5) / 20;
   phase += (Random Uniform() - .5) / 10;
   tf << Reshow;
   Wait( .05 );
);
```

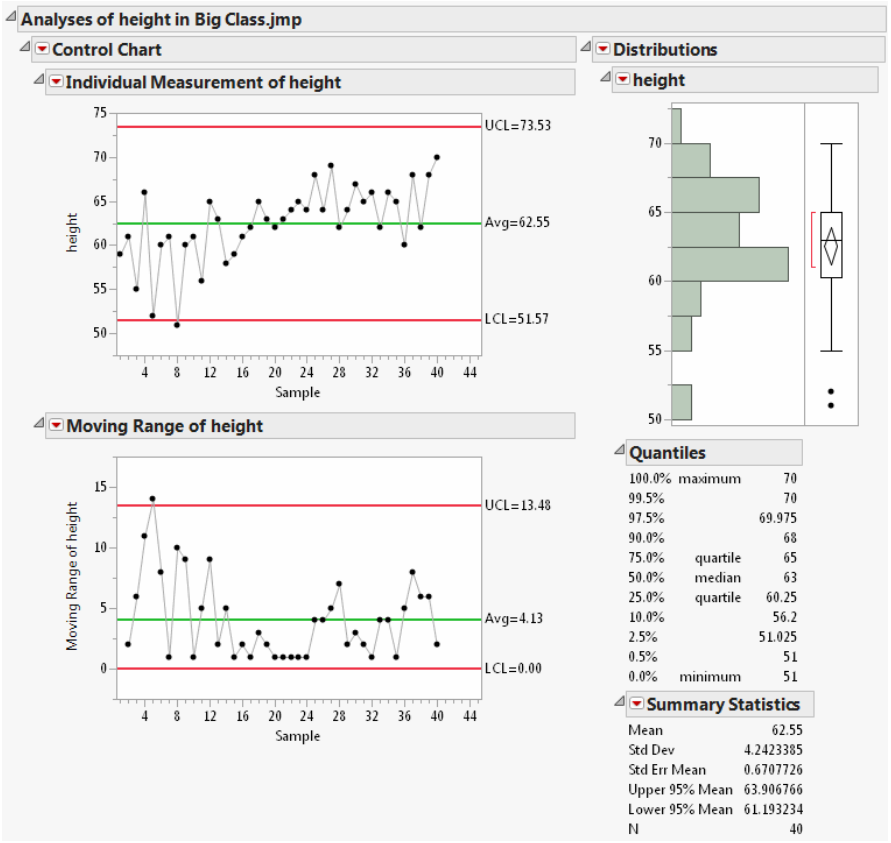
Construct Display Boxes That Contain Platforms

Another type of display that you might want to construct is your own combination of results from the analysis platforms in JMP. Write the platform script inside a display box and assemble the display boxes into a window. For ease in routing messages to it later, assign the whole script to a reference.

The following example creates several graphs and reports in one window:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
csp = New Window( "Platform Example",
   Outline Box( "Analyses of height in Big Class.jmp",
      H List Box(
         cc = Control Chart(
            Chart Col( Height, "Individual
            Measurement", "Moving Range" ),
            K Sigma( 3 )
         ),
         dist = Distribution( Columns( Height ) )
      )
   )
);
```

Figure 11.25 Example: Multiple Graphs in One Report Window



Now you can work with the window by sending messages to the reference *csp*. This is a display box reference, whose capabilities are similar to those of a `Report` for a platform. You can use multiple-argument subscripting to locate specific items within the outline tree:

```
csp["Control ?", "moving range ?"] << Close;
csp["Dist?", "quantiles"] << Close;
```

Notice that the preceding script not only assigned the whole window to a reference (`csp`) but also assigned the platform launch scripts to names (`cc` and `dist`) within their display boxes. This makes it easy to route messages to the platforms. You could, in turn, get the reports for these and have yet another way to manipulate display boxes. The following are equivalent messages that reopen the nodes:

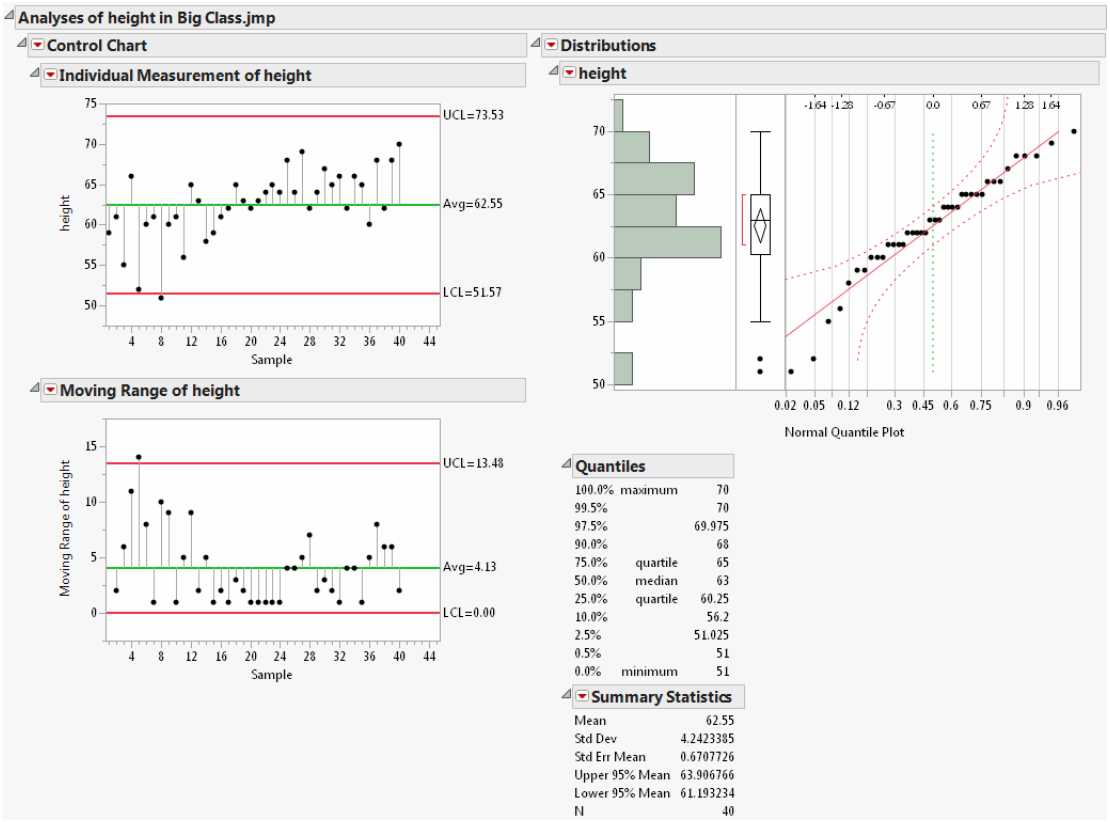
```
rcc = cc << Report;
rdist = dist << Report;
rcc["moving range ?"] << Close;
rdist["quantiles"] << Close;
```

You can send messages directly to the platform references themselves. Find out your options by searching for "Distribution" and "Control Chart" in the JMP Scripting Index. For example, Control Chart provides a message called `Needle` that creates a needle chart. Distribution provides a message called `Normal Quantile Plot`.

To execute these options from JSL, send them as messages to the platform references:

```
cc << Needle;
dist << Normal Quantile Plot;
```

Figure 11.26 Changing a Custom Report



Counting the Number of Child Boxes

Suppose that you want to count the number of child text boxes in a list box. The most reliable way is to use XPath to find a specific string in the text boxes and then use `N Items()` to count the number of text boxes.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = Bivariate(
```

```
    Y( :weight ),
    X( :height ),
    Group By( :age ),
    Fit Line( {Line Width( 3 )} )
);

/* start with the root list box, find a text edit box
that starts with “Linear Fit age==”, and count the number of children */
N Items( biv << XPath( "//ListBox[starts-with(TextEditBox,'Linear Fit
  age==')]" ) );
```

Examples of Creating a Dashboard from Two Reports

In JMP, you create dashboards in Dashboard Builder. See *Using JMP*. This section shows how to write scripts to create dashboards.

Dashboard with One Row of Reports

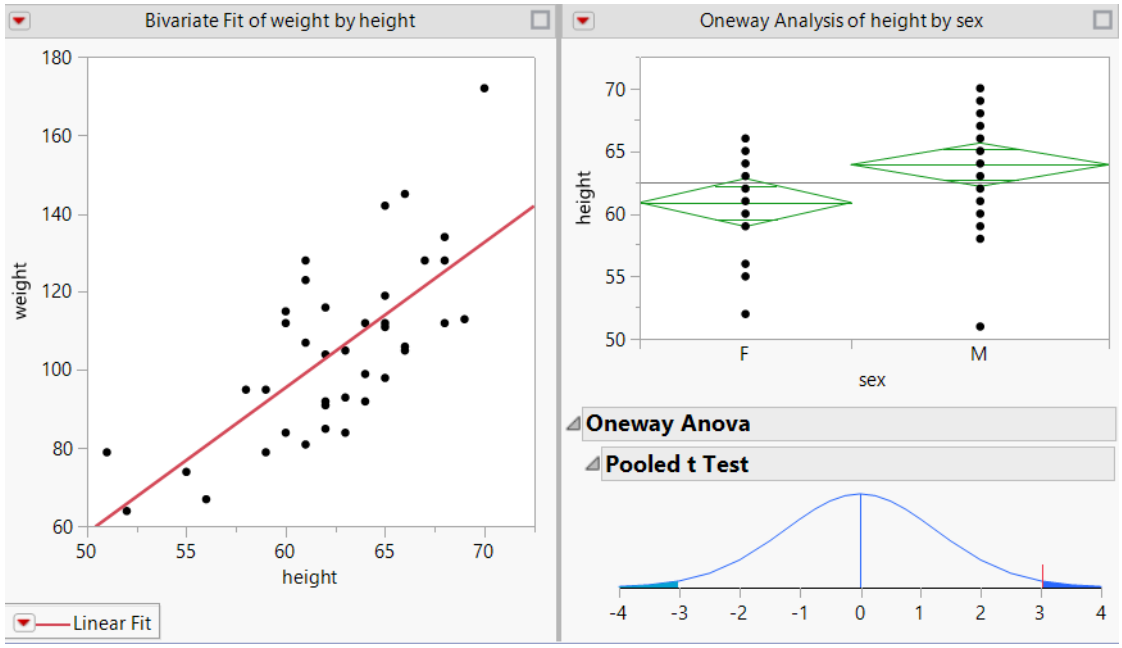
In the report window, reports appear in tab page boxes. You reposition reports in a dashboard by clicking the tab title and dragging the tab page to a highlighted drop zone. The highlighted drop zones indicate how to arrange reports horizontally, vertically, or in tabs. If a valid drop zone is not selected, the box returns to its original location.

This example shows how to write a script that creates a dashboard with two reports in a row:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
New Window( "My Dashboard",
  H Splitter Box( // place reports in a splitter box
  Size( 700, 400 ),
     Tab Page Box( // place report in a tab page box
        Title( "Bivariate Fit of weight by height" ),
        dt << Bivariate( // Bivariate report
           Y( :weight ),
           X( :height ),
           Fit Line,
           Report View( "Summary" ) // show only the graph
        ),
        <<Moveable( 1 ) // make the report moveable
     ),
     Tab Page Box(
        Title( "Oneway Analysis of height by sex" ),
        dt << Oneway( // Oneway report
           Y( :height ),
           X( :sex ),
           Means( 1 ),
           Mean Diamonds( 1 ),
```

```
            Report View( "Summary" )
         ),
         <<Moveable( 1 )
      ),
      /* each tab box can be dragged to a different location
      required for moveable display boxes */
      <<Dockable( 1 )
   )
);
```

Figure 11.27 Dashboard with Two Reports



Note: For dockable and moveable tabs that contain a single platform, the top-level outline box is subsumed into the tab title. This means that the red triangle options now appear in the title bar of the tabs.

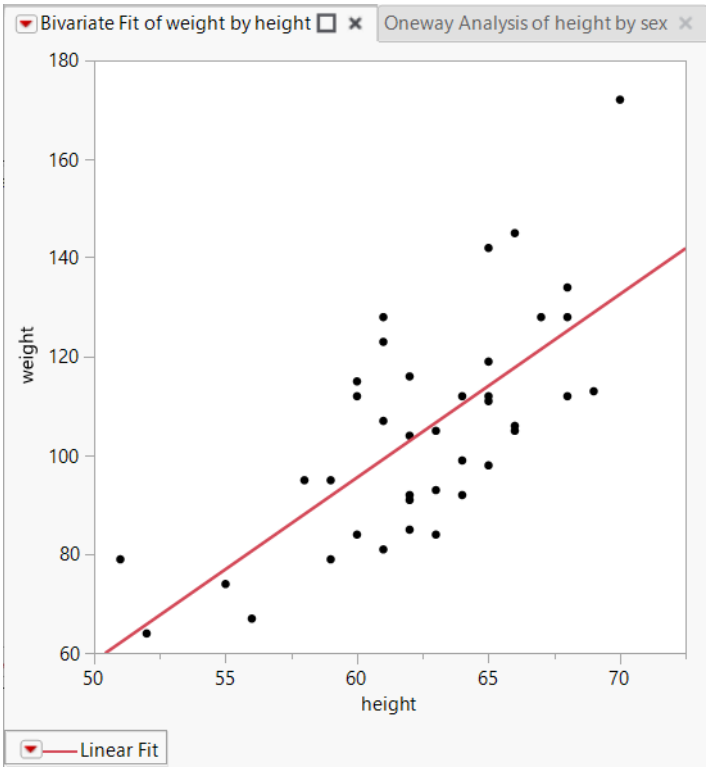
Dashboard with Tabs

The following script creates a dashboard that places each report on a tab:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
New Window( "My Dashboard",
   Tab Box( // place reports in a tab box
      Tab Page Box( // place report in a tab page box
         Title( "Bivariate Fit of weight by height" ),
```

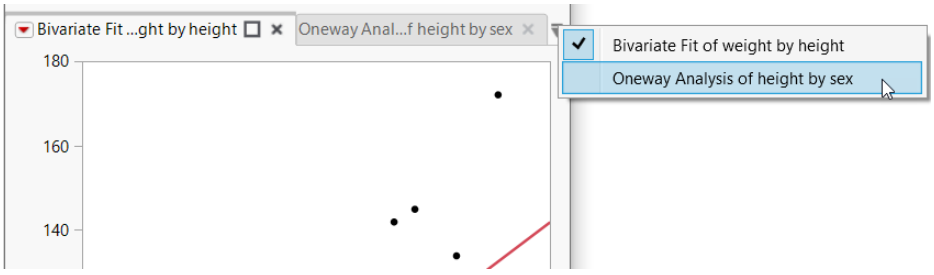
```
            dt << Bivariate( // Bivariate report
               Y( :weight ),
               X( :height ),
               Fit Line,
               Report View( "Summary" ) // show only the graph
            ),
            <<Moveable( 1 ), // make the report moveable
            <<Closeable( 1 ) // make the report closeable
         ),
         Tab Page Box(
            Title( "Oneway Analysis of height by sex" ),
            dt << Oneway( // Oneway report
               Y( :height ),
               X( :sex ),
               Means( 1 ),
               Mean Diamonds( 1 ),
               Report View( "Summary" )
            ),
            <<Moveable( 1 ),
            <<Closeable( 1 )
         ),
         /* each tab box can be dragged to a different location
         required for moveable display boxes */
         <<Dockable( 1 ),
         <<Set Overflow Enabled( 0 ) // both tab pages appear
      )
   );
```

Figure 11.28 Dashboard with Tabs



In a dashboard with several tabs, the window widens to show all tabbed reports. You might want to show only the first report and then let the user select the other tabbed reports from a list. Change `Set Overflow Enabled( 0 )` near the end of the script to `Set Overflow Enabled( 1 )`. Figure 11.29 shows the result.

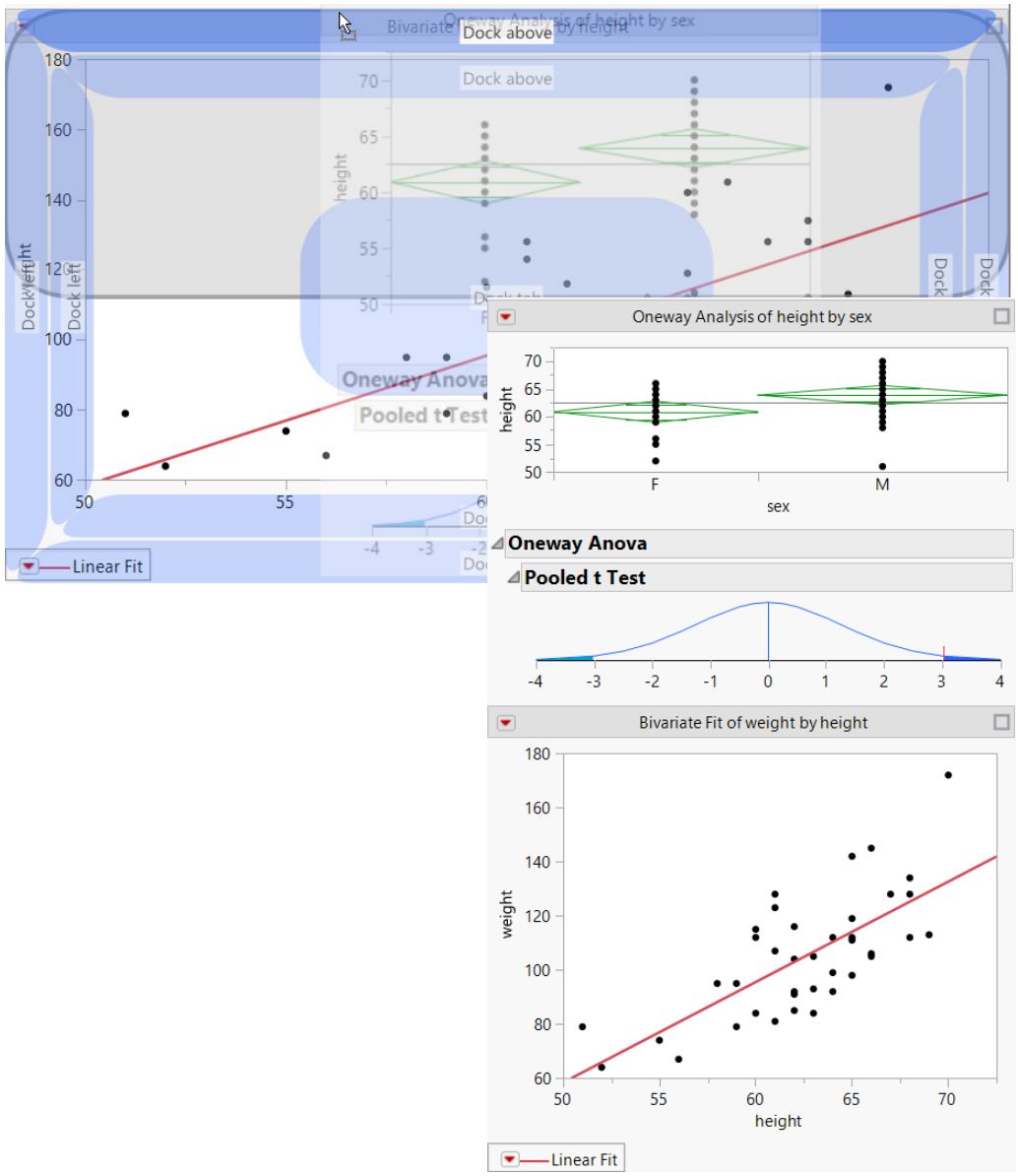
Figure 11.29 Dashboard with Overflow Tabs



How to Dock Tabs in Tab Boxes

A docking layout is a collection of `Tab Page Box()` objects and docking containers that enable reports to be rearranged. In the report window, you drag a docked tab to a highlighted drop zone to create a new layout. Figure 11.27 shows two docked reports. In Figure 11.30, the Oneway report was dragged to the highlighted drop zone to display the reports in one column.

Figure 11.30 Docking a Tab in a Dashboard



You combine `Tab Box()` with `Splitter Box()` to create docked tabs. Tab boxes and splitter boxes must directly contain other docking containers or the tab pages that define the content. Note the stacked boxes from a portion of the preceding example:

```
H Splitter Box( // place reports in a splitter box
  Size( 700, 400 ),
  Tab Page Box( // place report in a tab page box
```

The following expressions specify that the tab box and splitter box are docking containers, within which tab pages can be docked:

```
Tab Box << Set Dockable();
Splitter Box << Set Dockable();
```

In the report window, you reposition tab pages by clicking the tab title and dragging the tab page to a highlighted drop zone. The highlighted drop zones indicate how to arrange reports horizontally, vertically, or in tabs. If a valid drop zone is not selected, the box returns to its original location.

Closing Tabs

The following expression creates a close button to the right of the report title on a closeable tab page:

```
Tab Page Box << Set Closeable();
```

Clicking the close button deletes the tab and its contents. Parent containers might also be removed if they are no longer necessary (for example, if there are no more tabs in a tab box or only a single child in a splitter box).

Use the `Tab Page Box <<Set Close` message to run a JSL script when the close button is clicked. A return value of "1" indicates that the tab is closed. A return value of "0" indicates that closing the tab is not allowed at that time. Closing the tab might not be allowed as the result of a user prompt or based on the state of the program.

Controlling Wide Tabs

When a docking layout includes tab page boxes, you may place many pages in a single `Tab Box()` container. In this case, the titles may become wider than the actual content or wider than desired in the overall display. The `Tab Box <<Set Overflow Enabled( 0|1 )` message specifies that the tab titles should not cause the tab box to grow wider. If the titles do not all fit, a pop-up button appears in the upper right corner of the tab box to select tabs that are in the "overflow list". See Figure 11.29 for an example.

Example of Creating a Cluster Platform Launch Window

The following example creates a simplified replica of the Cluster platform launch window, which then launches the platform with the given arguments.

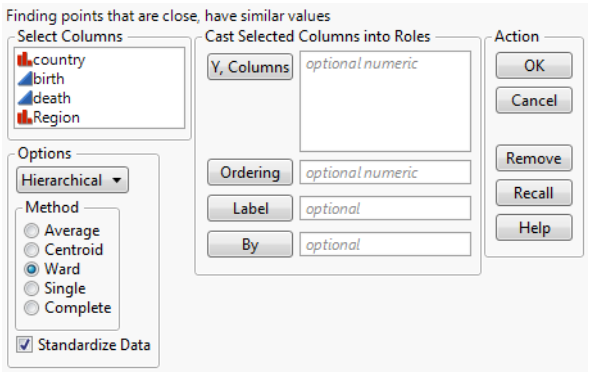
Note: Some functions (`Recall` and `Help`) are not implemented in the script, so an alert window is shown when the Recall and Help button are clicked. In addition, switching from Hierarchical to K-Means clustering does not change anything, unlike the real Cluster launch window.

```
dt = Open( "$SAMPLE_DATA/Birth Death.jmp" );
nc = N Col( dt ); // number of columns in the window
lbWidth = 130; // width of the window

// define the method list
methodList = {"Average", "Centroid", "Ward", "Single", "Complete"};
notImplemented = Expr(
   win = New Window( "Feature Not Implemented Yet", <<Modal, Button Box( "OK"
   ) )
);
clusterDlg = New Window( "Clustering", // create the window
   <<Modal,
   Border Box( Left( 3 ), Top( 2 ),
      V List Box(
         Text Box( "Finding points that are close, have similar values" ),
         H List Box(
            V List Box(
               Panel Box( "Select Columns",
                  colListData = Col List Box(
                     All,
                     width( lbWidth ),
                     NLines( Min( nc, 10 ) )
                  )
               ),
               Panel Box( "Options",
                  V List Box(
                     comboObj = Combo Box(
                        {"Hierarchical", "K-Means"},
                        <<Set( 1 )
                     ),
                     Panel Box( "Method",
                        methodObj = Radio Box( methodList, <<Set( 3 ) )
                     ),
                     checkObj = Check Box( {"Standardize Data"}, <<Set( 1, 1 ) )
                  )
               )
            ),
            Panel Box( "Cast Selected Columns into Roles",
               Line Up Box( N Col( 2 ), Spacing( 3 ),
                  Button Box( "Y, Columns",
                     colListY << Append( colListData << GetSelected )
                  ),
                  colListY = Col List Box(
                     width( lbWidth ),
                     NLines( 5 ),
                     "numeric"
```

```
            ),
            Button Box( "Ordering",
               colListO << Append( colListData << GetSelected )
            ),
            colListO = Col List Box(
               width( lbWidth ),
               NLines( 1 ),
               "numeric"
            ),
            Button Box( "Label",
               colListL << Append( colListData << GetSelected )
            ),
            colListL = Col List Box( width( lbWidth ), NLines( 1 ) ),
            Button Box( "By",
               colListB << Append( colListData << GetSelected )
            ),
            colListB = Col List Box( width( lbWidth ), NLines( 1 ) )
         )
      ),
      Panel Box( "Action",
         Line Up Box( N Col( 1 ),
            Button Box( "OK",
               If( (comboObj << Get) == 1,
                  Hierarchical Cluster(
                     Y( Eval( colListY << GetItems ) ),
                     Order( Eval( colListO << GetItems ) ),
                     Label( Eval( colListL << GetItems ) ),
                     By( Eval( colListB << GetItems ) ),
                     Method( methodList[methodObj << Get]
                     ),
                     Standardize( checkObj << Get( 1 ) )
                  ),
                  KMeansCluster( Y( colListY << GetItems ) )
               );
               clusterDlg << Close Window;
            ),
            Button Box( "Cancel", clusterDlg << Close Window ),
            Text Box( " " ),
            Button Box( "Remove",
               colListY << RemoveSelected;
               colListO << RemoveSelected;
               colListL << RemoveSelected;
               colListB << RemoveSelected;
            ),
            Button Box( "Recall", notImplemented ),
            Button Box( "Help", notImplemented )
```

```
            )
         )
      )
   )
)
);
```

Figure 11.31 The Cluster Launch Window



Example of Constructing a Custom Platform

An example in "Manipulating Expressions", showed how to use the `Substitute Into()` function to input coefficients for a quadratic polynomial into the quadratic formula and then use the formula to calculate the roots of the polynomial. That example required specifying the coefficients as arguments to `Substitute Into()`.

The section "Construct a Column Dialog" shows an example to collect coefficients from the user using a modal dialog box.

This section further develops the example into a complete customized platform that first displays a dialog box to ask for coefficients, finds the roots, and then displays the results along with a graph in a custom window.

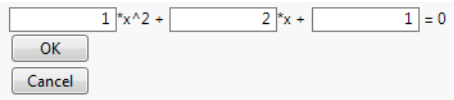
```
// create a window to collect coefficients from the user
myCoeffs = New Window( "Find the roots for the equation",
   <<Modal,
   H List Box(
      a = Number Edit Box( 1 ),
      Text Box( "*x^2 + " ),
      b = Number Edit Box( 2 ),
      Text Box( "*x + " ),
      c = Number Edit Box( 1 ),
      Text Box( " = 0" )
   ),
```

```
      Button Box( "OK",
         a = a << Get;
         b = b << Get;
         c = c << Get;
         Show( a, b, c );
      ),
      Button Box( "Cancel" )
   );

   /* calculate the results: The quadratic formula is
   x=(-b + - sqrt(b^2 - 4ac))/2a. Plug the coefficients into
   the quadratic formula */
   x = {Expr(
      (-b + Sqrt( b ^ 2 - 4 * a * c )) / (2 * a)
   ), Expr(
      (-b - Sqrt( b ^ 2 - 4 * a * c )) / (2 * a)
   )};
   xx = Eval Expr( x );


   results = Expr( // store the solution list
   /* test whether real roots were found and make an appropriate display
   if yes (for example, with the window's defaults), show roots and a graph */
      xmin = xx[1] - 5;
      xmax = xx[2] + 5;
      ymin = -20;
      ymax = 20;
      win = New Window( "The roots of a quadratic function",
         V List Box(
            Text Box( "The real roots for the equation " ),
            Text Box( "    " || Expr( po ) || " = 0" ),
            H List Box( Text Box( "are x=" ), Text Box( xxx ) ),
            Text Box( " " ), // to get a blank line
            Graph Box(
               Frame Size( 200, 200 ),
               X Scale( xmin, xmax ),
               Y Scale( ymin, ymax ),
               Line Style( 2 ),
               H Line( 0 ),
               Line Style( 0 ),
               Y Function( polynomial, x ),
               Line Style( 3 ),
               Pen Color( 3 ),
               V Line( xx[1] ),
               V Line( xx[2] ),
               Marker Size( 2 ),
```

```
            Marker( 0, {xx[1], 0}, {xx[2], 0} )
         )
      )
   );
);
error = Expr(

/* if no (for example, with a=3, b=4, c=5), put up an error
window with a helpful graph */
   win = New Window( "Error",
      V List Box(
         Text Box( " " ),
         Text Box( "  Polynomial " || po || " has no real roots.  " ),
         Text Box( " " ),
         Text Box( "Examine a graph of the function to get an idea why." ),
         Graph Box(
            Frame Size( 200, 200 ),
            X Scale( -20, 20 ),
            Y Scale( -20, 20 ),
            Line Style( 2 ),
            H Line( 0 ),
            Line Style( 0 ),
            Y Function( polynomial, x )
         )
      )
   )
);

/* either way, the script needs to have some strings ready
rewrite the polynomial with the coefficients specified */
polynomial = Expr( a * x ^ 2 + b * x + c );

// store this instance of the polynomial as a string
po = Char( Eval Expr( polynomial ) );

// store the solution list as a string
xxx = Char( Eval Expr( x ) );

If( Is Missing( xx[1] ) | Is Missing( xx[2] ), // now it's ready for a test
   error,
   results
);
```

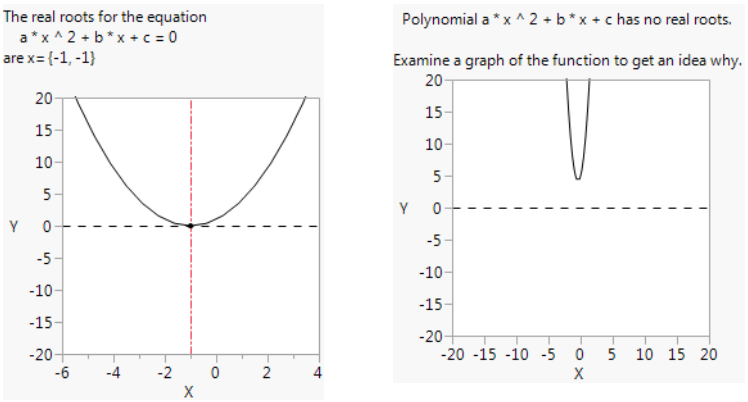
When you run the preceding script, you first see a window like this:

Figure 11.32 A Custom Platform Launch Window



Clicking **OK**, displays a results window (Figure 11.33, left), with either the roots or an error message. Rerun the script, input 5, 4, and 5 respectively and click **OK**. Note that JMP displays an error message (Figure 11.33, right).

Figure 11.33 The Custom Platform's Report

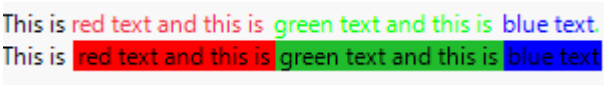


Example of Specifying Text and Background Colors

The `markup` argument in a display box enables you to write HTML code to specify text and background colors. The color can be a hex value (starting with "0x"or "#") or a known JMP color. Here is an example:

```
New Window( "Colors",
   Text Box(
      "This is <font color=\!"Red\!">red text and this is <font
   color=\!"0x00ff00\!">green text and this is <font color=\!"#0000FF\!">blue
   text</font>.",
      <<markup
   ),
   Text Box(
      "This is <background color=\!"#ff0000\!"> red text and this
   is<background color=\!"Green\!"> green text and this is<background
   color=\!"0x0000FF\!"> blue text</background>",
      <<markup
   )
);
```

Figure 11.34 HTML Text and Background Colors



Modal Windows

As described in "Constructors for New Windows", *modal* means that the user must interact with the window immediately. Clicking outside the window produces an error sound. Script execution stops until the user responds to the window.

JMP provides two types of modal window functions:

- `New Window()` with the `Modal` message. This creates a new window in which you arrange other display boxes.
- `Column Dialog()` constructs a window for column role assignments as found in many launch windows. The specialized constructors for `Column Dialog()` mean that you have less flexibility than with `New Window()`. However, `Column Dialog()` lets you easily create a launch window.

Construct a Modal Window

When you submit a script with a modal window, JMP draws the window, waits for the user to make choices and click **OK**, and then stores a list of the variables with their values. Any attempt to click outside the window produces an error sound, and script execution is suspended until the user clicks **OK** or **Cancel**.

The `Column Dialog()` function is specifically intended to prompt users to choose columns from the current (topmost) data table. Windows created by `Column Dialog()` are also modal windows.

Tips:

- Put all the modal windows near the beginning of the script, if possible. This way, all the user interaction can be accomplished at once, and then users can leave JMP to finish its work unattended.
- Make sure your modal windows give the user enough information. Do not just present a number field. Tell users how the number is used. If there are limits for valid responses, say so.

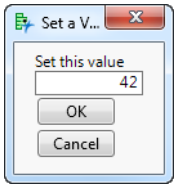
A very simple modal window might request a value for one variable:

```
win = New Window( "Set a Value",
   <<Modal,
```

```
    Text Box( "Set this value" ),
    variablebox = Number Edit Box( 42 ),
    Button Box( "OK" ),
    Button Box( "Cancel" )
);
```

Notice that the argument to `Number Edit Box()` is the default value, 42.

Figure 11.35 Sample Modal Display Box



If you click **OK**, the window closes and {`Button(1)`} is returned. To get the value entered in the box, use the `Return Result` message and a subscript into the window. Note that the deprecated `Dialog()` returns the list of variable assignments the same way.

```
win = New Window( "Set a Value",
    <<Modal,
    <<Return Result,
    Text Box( "Set this value" ),
    variablebox = Number Edit Box( 42 ),
    Button Box( "OK" ),
    Button Box( "Cancel" )
);

// create a subscript to the variablebox variable
Write( win["variablebox"] );
```

In this example, the user typed 33 in the number edit box. That value is stored in `win`.

If you click **Cancel**, the window closes, it returns {`Button(-1)`}, and the script continues.

To detect whether the user clicked Cancel, add the following expression:

```
If( win["Button"] == 1,
    Print( win["variablebox"] );
    ,
    Print("Canceled")
);
```

Note that a modal window must have at least one button. If you omit the `Button Box()` from your script, an OK button is automatically included. To omit the OK button, set the visibility to collapse.

```
win = New Window( "No OK Button",
```

```
    <<Modal,
    b = Button Box( "Close Window", b << Close Window ),
    bb = Button Box( "OK", <<Visibility( "collapse" ) )
);
```

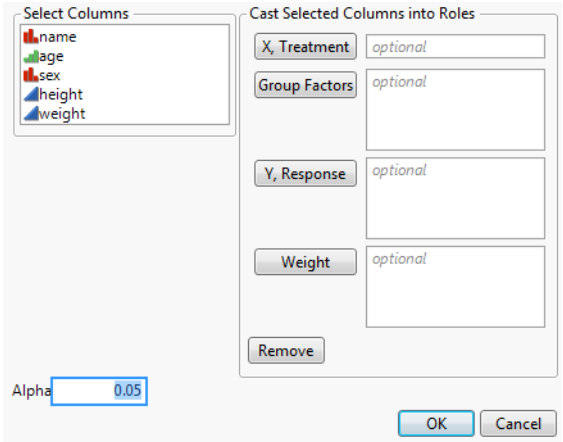
Note: Modal windows require at least one button for dismissing the window. You can use **OK**, **Yes**, **No**, or **Cancel** to label modal window buttons. If you do not include at least one of these buttons in a modal window, JMP adds an **OK** button.

Construct a Column Dialog

`Column Dialog()`, a variant of the deprecated `Dialog()` function, lets you prompt for column selections from the current data table or contextual data table. The `OK`, `Cancel`, and `Remove` buttons and the list of columns to choose from the data table are added automatically.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dlg = Column Dialog(
   Col ID = Col List( "X, Treatment", Max Col( 1 ) ),
   Group = Col List( "Group Factors" ),
   Split = Col List( "Y, Response" ),
   w = Col List( "Weight" ),
   H List( "Alpha", alpha = Edit Number( .05 ) )
);
```

Figure 11.36 Column Dialog



The following example returns a list similar to this one, depending on the user's choices:

```
{Col ID = {}, Group = {}, Split = {}, w = {}, alpha = 0.05, Button( -1 )}
```

For each destination list, a `Col List` clause must be a direct argument of `Column Dialog()` (not nested inside some other argument). An optional `MaxCol(`*`n`*`)` argument restricts the number of data columns that can be chosen to *n*. Lists are *always* returned, although they can sometimes be empty lists. You can include as many as twelve `Col List` clauses.

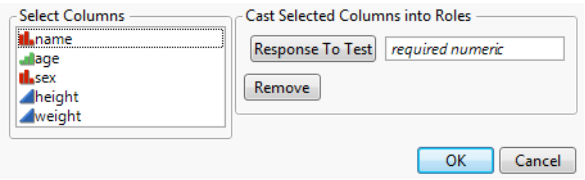
Other items permitted in the deprecated `Dialog()` are permitted in `Column Dialog()` and have the same functionality. You can specify the minimum and maximum number of columns that are allowed in a column dialog box with the `MinCol` and `MaxCol` arguments.

You can also specify the modeling type of the columns that are allowed to be selected (`Ordinal`, `Nominal` or `Continuous`). You can set the width of the list using `Select List Width(`*`pixels`*`)` argument. To set the width of the column list, use `Width(`*`pixels`*`)` inside the `Col List()` function.

The following example generates a column dialog box that only allows the selection of exactly one numeric column:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
rt_dlg = Column Dialog(
   cv = Col List( "Response To Test", MaxCol( 1 ), MinCol( 1 ), DataType(
   "Numeric" ) )
);
```

Figure 11.37 Restricting Selection of Columns



The `Data Type` choices are `Numeric`, `Character`, and `Any`.

In addition, use the `Columns` specification to pre-fill some column selections. For example, the following script assigns height to the X role and weight and age to the Y role:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dlg = Column Dialog(
   xCols = Col List( "X, Factors", Columns( :height ) ),
   yCols = Col List( "Y, Response", Columns( :weight, :age ) )
);
```

Comparison of Column Dialog and New Window

`New Window()` and `Column Dialog()` enable you to construct a launch window that has a column selector. Use `Column Dialog()` to construct a simple launch window. Several items are added automatically to column dialogs: the Remove and Cancel buttons, the list of columns, and an OK button (if no buttons are explicitly defined).

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dlg = Column Dialog(
   x = ColList( "X", Max Col( 1 ) ),
   y = ColList( "Y", Max Col( 1 ) )
);

// returns dlg = {x = {:weight}, y = {:height}, Button(1)};
Show( dlg );

If( dlg["Button"] == 1,
   xCol = dlg["x"];
   yCol = dlg["y"];
);

Show( xCol, yCol );
   xCol = {:height};
   yCol = {:weight};
```

`New Window()` is more flexible because of numerous optional arguments. In the following example, the user must select two columns before clicking the OK button. A Bivariate graph is then created. Validating the user's selections is not possible in `Column Dialog()`.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
xvar = .;
yvar = .;
win = New Window( "Example of Returning Values",
   <<Modal,

   // require the user to select two variables before clicking OK
   <<On Validate(
      Show( xvar, yvar );
         If( Is Missing( xvar ) | Is Missing( yvar ),

         // if xvar or yvar are missing do nothing when OK is clicked
         0,
         1
      );
   ),
   Text Box( " Select two numeric columns. " ),
   H List Box(
```

```
      Text Box( " X, Factor " ),
      x = Col List Box(
         dt, // data table reference
         all, // display all columns from the data table

         // get the name of the selected column before the window closes
         xvar = (x << Get Selected)[1];
         Show( xvar );
      ),
      Text Box( "Y, Response" ),
      y = Col List Box(
         dt,
         all,
         yvar = (y << Get Selected)[1];
         Show( yvar );
      )
   )
);
If (win["Button"] == 1, // if the user clicks OK...
xcol = Column( dt, xvar ); //  get the columns
ycol = Column( dt, yvar );
dt << Bivariate( Y( ycol ), X( xcol ) ); // create a Bivariate plot
);
```

Constructors for Column Dialogs

Constructors such as `Edit Number()` and `Check Box()` can be used in column dialogs, which construct launch windows with a column selector. The following example shows a typical column dialog with an `Edit Number()` box:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column Dialog(
   ex x = ColList( "X", Max Col( 1 ) ),
   HList( "Alpha",
   ex = Edit Number( .05 ) )
);
```

Table 11.3 describes the column dialog constructors.

Notes:

- A column dialog automatically includes the list of columns in the data table.
- A column dialog also automatically includes OK, Cancel, and Remove buttons.

Table 11.3 Column Dialog Constructors

Constructor	Syntax	Explanation
`Button`	`Button( "OK" ), Button( "Cancel" )`	Draws an OK or a Cancel button. If OK is clicked, `Button(1)` is returned. If Cancel is clicked, `Button(-1)` is returned.
`Check Box`	*var*`( CheckBox( "`*Text after box*`", <1|0> ) )`	When OK is clicked, a selected check box assigns 1 to the variable. A check box that is not selected assigns 0 to the variable. Add an optional 1 for the check box to be selected (on), or 0 for it to be not selected (off) when the window first appears. The default value is 0 (off).
`Col List`	*var*`=Col List( "`*role*`", <MaxCol( `*n*` )>, <Datatype(` *type* `)> )`	Creates a selection destination with a *role* button; the user's choices are returned in a list item of the form *var*={*choice 1*, *choice 2*, …, *choice n*}. You can specify the minimum and maximum number of columns using `MinCol(` *n* `)` or `MaxCol(` *n* `)`. You can specify the required data type of the column using `Datatype(`*type*`)`. The choices for *type* are `Numeric`, `Character`, or `Rowstate`.
`Combo Box`	*var*`( ComboBox( "`*choice1*`", "`*choice2*`", ...) )`	Produces a menu with the choices listed. The first choice is the default. Choices evaluate to quoted text strings. Choices can also be inside a list.
`Edit Number`	*var*`( Edit Number(` *number* `) )`	Produces an edit field for a number with *number* as the default value. When OK is clicked, the number entered in the field is assigned to the variable.

Table 11.3 Column Dialog Constructors *(Continued)*

Constructor	Syntax	Explanation
Edit Text	*var*(Edit Text("*string*", <width(*x*)>))	Produces an edit field for a string with *string* as the default value. You can also specify the minimum width of the box in pixels. The default width is 72 pixels. When OK is clicked, the text entered in the field is assigned to the variable.
HList	HList(*item*, *item*, ...)	Top aligns and spaces the *items* in a horizontal row. Placing a pair of VLists within an HList produces a top aligned, spaced pair of columns.
Line Up Box	Line Up Box(*n*, *item_11*, *item_12*, ..., *item_1n*, ..., *item_nn*)	Lines up the *items* listed in *n* columns, where *item_ij* is the *j*th item of the *i*th row.
List Box	var=List Box({"item", "item", ...}, width(50), max selected(2), NLines(6))	Creates a display box that contains a list of items.
Radio Buttons	*var*(RadioButtons("*choice1*", "*choice2*", ...))	Produces a vertical, left-justified list of radio buttons with the choices specified. The first choice is the default. When OK is clicked, the button that is selected is assigned to the variable. Choices must evaluate to quoted text strings.
string	"*string*"	Draws text in the window. For example, can include a labeling string before an Edit Number field. All strings must be quoted.
VList	VList(*item*, *item*, ...)	Left aligns and spaces the *items* in a vertical column. Placing a pair of HLists within a VList produces a left-aligned, spaced pair of rows.

Convert Deprecated Dialog to New Window

Because `New Window()` provides more display options and better control over the content and functions of the window, rewrite your `Dialog()` scripts.

Note: `Dialog()` scripts will not run in the future.

The following sections illustrate how to rewrite `Dialog()` scripts as `New Window()` scripts.

- "New Window Example"
- "Example of Deprecated Dialog"
- "Differences Between New Window and the Deprecated Dialog"
- "Optional Scripts in New Window"

New Window Example

The following example creates a window with multiple vertical and horizontal list boxes:

```
win = New Window( "New Window Example",
   <<Modal,
   <<ReturnResult, // ensure that you get the same result as with Dialog
   V List Box(
      V List Box(
         Text Box( "Radio Frequency Embolism Projection" ),
         Line Up Box(
            NCol( 2 ),
            Text Box( "Lower Spec Limit" ),
            lsl_box = Number Edit Box( 230 ),
            Text Box( "Upper Spec Limit" ),
            usl_box = Number Edit Box( 340 ),
            Text Box( "Threshold" ),
            threshold_box = Number Edit Box( 275 )
         ),
         H List Box(
            Panel Box( "Type of Radio",
                  rb_box1 = Radio Box( {"RCA", "Matsushita",
                  "Zenith", "Sony"} )),
            Panel Box( "Type of Antenna" ,
                  rb_box2 = Radio Box( {"Dish", "Helical", "Polarizing",
                  "Radiant Array"} ) )
         ),
         cb_box1 = Check Box( "Emission Synchronization" ),
         Text Box( "Title for plot" ),
         title_box = Text Edit Box( "My projection" ),
```

```
            H List Box( Text Box( "Quality" ),
            cb_box2 = Combo Box( {"Fealty", "Loyalty", "Piety", "Obsequiousness"}
      ) )
         ),
         H List Box(
            Align( Right ),
            Spacer Box(),
            Button Box( "OK",
               lsl = lsl_box << Get;
               usl = usl_box << Get;
               threshold = threshold_box << Get;
               radio_type = rb_box1 << Get;
               antenna = rb_box2 << Get;
               synch = cb_box1 << Get;
               title = title_box << Get Text;
               quality = cb_box2 << Get;
            ),
            Button Box( "Cancel" )
         )
      )
   );
   If(win["Button"] == 1,

   // if the user clicks OK, show the selections in the log
      Show( "OK", lsl, usl, threshold, radio_type, antenna, synch, title,
      quality);
      ,
      Show( "Canceled" ); // if the user clicked Cancel, print "Canceled"
   );
```

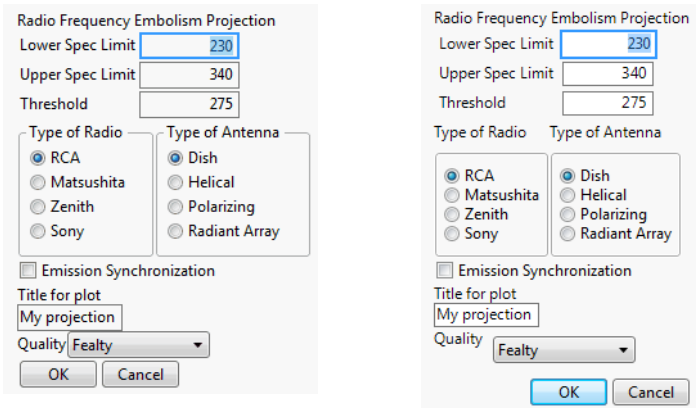
Example of Deprecated Dialog

Compare the previous `New Window()` example to this example of the deprecated `Dialog()`.

```
dlg = Dialog(
   Title( "Dialog Example" ),
   H List(
      V List(
         "Radio Frequency Embolism Projection",
         Lineup( 2,
            "Lower Spec Limit", lsl = Edit Number( 230 ),
            "Upper Spec Limit", usl = Edit Number( 340 ),
            "Threshold", threshold = Edit Number( 275 )
         ),
         H List(
            V List(
```

```
                "Type of Radio",
                type = Radio Buttons( "RCA", "Matsushita", "Zenith", "Sony" )
             ),
             V List(
                "Type of Antenna",
                antenna = Radio Buttons( "Dish", "Helical", "Polarizing",
                   "Radiant Array" )
             )
          ),
          synch = Check Box( "Emission Synchronization", 0 ),
          "Title for plot",
          title = Edit Text( "My projection" ),
          H List(
             "Quality",
             quality = Combo Box( "Fealty", "Loyalty", "Piety", "Obsequiousness"
             )
          )
       ),
       V List( Button( "OK" ), Button( "Cancel" ) )
    )
);
If( dlg["Button"] == 1,
   Show(
      "OK",
      dlg["lsl"],
      dlg["usl"],
      dlg["threshold"],
      dlg["type"],
      dlg["antenna"],
      dlg["synch"],
      dlg["title"],
      dlg["quality"]
   ),
   Show( "Canceled" )
);
```

Figure 11.38 Results from New Window (left) and the Deprecated Dialog (right)



Note: Use `Return Result` to get the same result in `New Window()` as with the deprecated `Dialog()`. See "Construct a Modal Window" for an example.

Differences Between New Window and the Deprecated Dialog

This section shows the differences between display boxes in `New Window()` and the deprecated `Dialog()` scripts. Table 11.4 summarizes those differences. More information can be found in the sections that follow the table.

Table 11.4 Display Boxes in New Window and Deprecated Dialog Scripts

Box Type	New Window	Deprecated Dialog

"Strings and Lists"	`win = New Window( "Combo Box",`
	`  <<Modal,`
	`  <<ReturnResult,`
	`  cb = Combo Box( {"True", "False"} ),`
	`);`
		`Dialog( Title( "Combo Box" ),`
		`  cb = Combo Box( "True", "False" ),`
		`);`

"List Boxes and Check Boxes"	`win = New Window( "H List Box",`
	`  <<Modal,`
	`  <<ReturnResult,`
	`  H List Box(`
	`  kb1 = Check Box( "a" ),`
	`  kb2 = Check Box( "b" ),`
	`  kb3 = Check Box( "c" )`
	`  ),`
	`);`
		`dlg = Dialog( Title( "H List" ),`
		`  H List(`
		`  kb1 = Check Box( "a", 0 ),`
		`  kb2 = Check Box( "b", 0 ),`
		`  kb3 = Check Box( "c", 0 )`
		`  ),`
		`);`

Table 11.4 Display Boxes in New Window and Deprecated Dialog Scripts *(Continued)*

Box Type	New Window	Deprecated Dialog
"Line Up Items"	win = New Window("Line Up Box", <<Modal, <<ReturnResult, V List Box(Line Up Box(NCol(2), Text Box("Set this value"), var1=Number Edit Box(42), Text Box("Set another value"), var2=Number Edit Box(86),),), H List Box(Button Box("OK"), Button Box("Cancel")); // returns: {Button(1)}	dlg = Dialog(V List(Line Up(2, "Set this value", variable=Edit Number(42), "Set another value", var2=Edit Number(86)), H List(Button("OK"), Button("Cancel")))); // returns: {variable = 42, var2 = 86, Button(1)}
"Radio Boxes"	win = New Window("Radio Box", <<Modal, <<ReturnResult, Panel Box("Select", rbox = Radio Box({"a", "b", "c"})));	dlg = Dialog(Title("Radio Buttons"), rb = Radio Buttons({"a", "b", "c"}),);
"Text Edit Boxes"	win = New Window("Text Edit Box", <<Modal, <<ReturnResult, V List Box(Text Box("Strings"), str1 = Text Edit Box("The"), str2 = Text Edit Box("quick"),));	dlg = Dialog(Title("Edit Text"), V List("Strings", str1 = Edit Text("The"), str2 = Edit Text("quick"),));

Table 11.4 Display Boxes in New Window and Deprecated Dialog Scripts *(Continued)*

Box Type	New Window	Deprecated Dialog

"Number Edit Boxes"	`win = New Window( "Number Edit Box",`
	`<<Modal,`
	`<<ReturnResult,`
	`V List Box(`
	`Text Box("Random Numbers"),`
	`num1 = Number Edit Box(`
	`Random Uniform()),`
	`num2 = Number Edit Box(`
	`Random Uniform() * 10),`
	`...);`
		`dlg = Dialog(`
		`Title( "Edit Number" ),`
		`V List(`
		`"Random Numbers",`
		`num1 = Edit Number(`
		`Random Uniform() ),`
		`num2 = Edit Number(`
		`Random Uniform() * 10 ),`
		`...);`

Strings and Lists

A major difference between the `New Window()` and the deprecated `Dialog()` functions is how you specify lists and strings as arguments. In `New Window()`, the items must be placed in a list. In the deprecated `Dialog()`, items are generally separated by commas. For example, compare the following two instances of a combo box.

```
win = New Window( "Combo Box", // modal New Window
   <<Modal,
   <<Return Result,
   cb = Combo Box( {"True", "False"} ), // items are in a bracketed list
);

Dialog( Title( "Combo Box" ),// modal Dialog
   cb = Combo Box( "True", "False" ), // items are separated by commas
   );
```

Note: The script does not explicitly define the OK button. For modal windows, JMP automatically adds the OK button. Explicitly define each additional button.

Additionally, the deprecated `Dialog()` could include `" "` to indicate empty text. `New Window()` requires that empty text appear in a `Text Box (" ")`.

List Boxes and Check Boxes

In `New Window()`, boxes are horizontally aligned with `H List Box()` and vertically aligned with `V List Box()`. In the deprecated `Dialog()`, you use `H List` and `V List` instead.

Note: Boxes are vertically aligned by default if `H List Box()` or `V List Box()` are omitted.

The following example creates a window with three horizontal check boxes and an OK button:

```
win = New Window( "H List Box",
   <<Modal,
   <<Return Result,
   H List Box(
      kb1 = Check Box( "a" ),
      kb2 = Check Box( "b" ),
      kb3 = Check Box( "c" )
   ),
);
```

Here is the same window created with the deprecated `Dialog()` and an `H List()`:

```
dlg = Dialog( Title( "H List" ),
   H List(
         kb1 = Check Box( "a", 0 ),
         kb2 = Check Box( "b", 0 ),
         kb3 = Check Box( "c", 0 )
     ),
);
```

Line Up Items

In `New Window()`, use `Line Up Box()` to arrange items in the number of columns that you specify. In the deprecated `Dialog()`, you used `Line Up`.

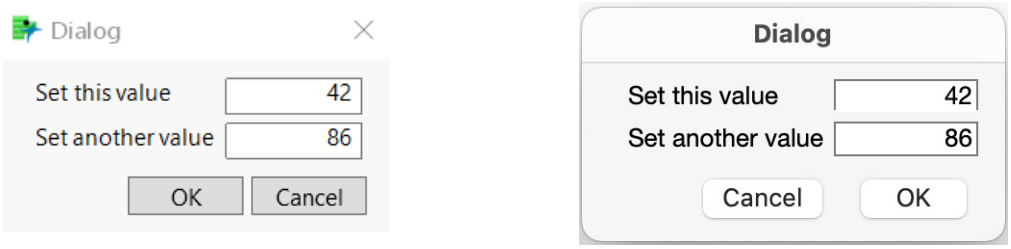
The following example arranges the text boxes in one column and the number edit boxes in another column:

```
win = New Window( "Line Up Box",
   <<Modal,
   <<Return Result,
   V List Box(
      Lineup Box( N Col( 2 ),
         Text Box( "Set this value" ),
         var1 = Number Edit Box( 42 ),
         Text Box( "Set another value" ),
         var2 = Number Edit Box( 86 ),

      ),
   ),
   H List Box( Button Box( "OK" ), Button Box( "Cancel" ) ) )
);
```

Clicking **OK** returns the following result in the log window:

```
List( 3 elements ) assigned.
```

Figure 11.39 Default Dialog Arrangements for Windows (left) and macOS (right)



Here's the same window created with the deprecated `Dialog()` and `Line Up`:

```
dlg = Dialog(
   V List(
      Line Up( 2,
         "Set this value", variable = Edit Number( 42 ),
         "Set another value", var2 = Edit Number( 86 )
      ),
      H List( Button( "OK" ), Button( "Cancel" ) )
   )
);
```

Clicking **OK** returns the following result in the log window:

```
{variable = 42, var2 = 86, Button(1)}
```

Note: JMP does exert some control over OK and Cancel button positions to ensure that dialog boxes are consistent with what the operating system expects. In certain cases, JMP needs to override your `H List Box()`, `V List Box()`, and `Line Up Box()` settings for `Button( "OK" )` and `Button( "Cancel" )`. Do not be alarmed if the result is slightly different from what you expect.

Radio Boxes

Another difference between `New Window()` and the deprecated `Dialog()` is the usage of `Radio Box`.

In `New Window()`, you must define the `Panel Box()` container for `Radio Box()` if you want the radio buttons in a box.

```
win = New Window( "Radio Box",
   <<Modal,
   <<Return Result,
   Panel Box( "Select",
      rbox = Radio Box( {"a", "b", "c"} )
   )
);
```

In the deprecated `Dialog()`, the `Radio Buttons()` automatically appear in a panel box.

```
dlg = Dialog( Title( "Radio Buttons" ),
   rb = Radio Buttons( {"a", "b", "c"} ),
);
```

Text Edit Boxes

In `New Window()`, use `Text Edit Box()` to create editable boxes that contain specified strings. In the deprecated `Dialog()`, you used `Edit Text()`.

```
win = New Window( "Text Edit Box",
   <<Modal,
   <<Return Result,
   V List Box(
      Text Box( "Strings" ),
      str1 = Text Edit Box( "The" ),
      str2 = Text Edit Box( "quick" ),
      str3 = Text Edit Box( "brown" ),
      str4 = Text Edit Box( "fox" ),
      str5 = Text Edit Box( "jumps" ),
      str6 = Text Edit Box( "over" ),
      str7 = Text Edit Box( "the" ),
      str8 = Text Edit Box( "lazy" ),
      str9 = Text Edit Box( "dog" ) ) );
```

Here is the same window created with the deprecated `Dialog()` and `Edit Text()`:

```
dlg = Dialog(
   Title( "Edit Text" ),
   V List(
      "Strings",
      str1 = Edit Text( "The" ),
      str2 = Edit Text( "quick" ),
      str3 = Edit Text( "brown" ),
      str4 = Edit Text( "fox" ),
      str5 = Edit Text( "jumps" ),
      str6 = Edit Text( "over" ),
      str7 = Edit Text( "the" ),
      str8 = Edit Text( "lazy" ),
      str9 = Edit Text( "dog" )
   )
);
```

Alternatively, use `String Col Edit Box()` to create editable boxes within in a table structure that contains specified strings. For example, the following script creates a window named "String Col Edit Box". The window contains a column (labeled "Strings") of editable boxes, each of which contains the specified string:

```
win = New Window( "String Col Edit Box",
   <<Modal,
   <<Return Result,
   steb = String Col Edit Box(
      "Strings",
      {"The", "quick", "brown", "fox", "jumps", "over", "the", "lazy", "dog"}
   )
);
```

Note: In the preceding example, `New Window()` assigns only two elements to the list (the specified list items and the button).

Number Edit Boxes

In `New Window()`, use `Number Edit Box()` to create editable boxes that contains the specified numbers. In the deprecated `Dialog()`, you used `Edit Number()`.

```
win = New Window( "Number Edit Box",
   <<Modal,
   <<ReturnResult,
   V List Box(
      Text Box( "Random Numbers" ),
      num1 = Number Edit Box( Random Uniform() ),
      num2 = Number Edit Box( Random Uniform() * 10 ),
      num3 = Number Edit Box( Random Uniform() * 100 ),
      num4 = Number Edit Box( Random Uniform() * 1000 )
   ),
);
```

Here is the same window created with the deprecated `Dialog()` and `Edit Number()`:

```
dlg = Dialog(
   Title( "Edit Number" ),
   V List(
      "Random Numbers",
      num1 = Edit Number( Random Uniform() ),
      num2 = Edit Number( Random Uniform() * 10 ),
      num3 = Edit Number( Random Uniform() * 100 ),
      num4 = Edit Number( Random Uniform() * 1000 )
   ),
);
```

Another method for creating the same window uses `Number Col Edit Box()` to create editable boxes within in a table structure that contains specified numbers. For example, the following script creates a window named "Number Col Edit Box". The window contains a column (labeled "Random Numbers") of editable boxes, each of which shows the specified type of random number:

```
win = New Window( "Number Col Edit Box",
   <<Modal,
   <<ReturnResult,
   nceb = Number Col Edit Box(
      "Random Numbers",
      {num1 = Random Uniform(), num2 = Random Uniform() * 10, num3 =
      Random Uniform() * 100, num4 = Random Uniform() * 1000}
   ),
);
```

Note: In the preceding example, the `Number Col Edit Box()` assigns only two elements to the list. The `New Window()` and deprecated `Dialog()` examples for `Number Col Edit Box()` assign five elements to the list.

Optional Scripts in New Window

One of the benefits of using `New Window()` is the ability to add optional scripts to display boxes. In the deprecated `Dialog()`, the following combo box could not include an optional script. If you wanted an action associated with any of the display box controls, you had to place a script as the control's last argument. For example:

```
win = New Window( "Combo Box",
<<Modal,
<<ReturnResult,
   comboObj = Combo Box(
      {"True", "False"},
      << Set( 1 ),
      Print( comboObj << Get )
   )
);
```

When the user selects a different value, the selected item number (1 or 2 in this case, because there are two items in the combo box) is printed to the log.

Scripting the Script Editor

Even the script editor window is a display tree in JMP, which means you can write a JSL script to write and save another JSL script.

There is no `New Script` command. Instead, to open a new script window, you use the `New Window()` function and then send it a message to tell it that it's a script window:

```
ww = New Window( "Window Title", <<Script, "Initial Contents" );
```

The last argument is optional. If you include a string, the new script window contains that string.

In the `New Window` example above, `ww` is a reference to the display box that is the entire window. To write to a script window, you first need to get a reference to the part of the display box that you can write to, which is called a script box:

```
ed = ww[Script Box(1)];
```

Using the reference `ed`, you can add text, remove text, and get the text that is already there.

```
ed << Get Text();
   "Initial Contents"
```

Use `Set Text` to set all the text in the script window. The following command clears all text in the Script Window and then adds `aaa=3;` followed by a return:

```
ed << Set Text( "aaa=3;\!N" );
```

Use `Append` to add additional text to the end of the script window.

```
ed << Append Text( "bbb=1/10;" );
ed << Append Text( "\!Ncc=4/100;" );
```

Use the `Get Line Text` command to get the text at the line of a specified line number. Use the `Set Line Text` command to replace a specified line of text with new text.

```
ed << Get Line Text( 2 );
ed << Set Line Text( 2, "bbb = 0.1;" );
```

Use the `Get Line Count` message to get the total number of lines in the script. The `Get Lines` message returns a list of each line in the script as a string.

```
ed << Get Line Count();
ed << Get Lines();
```

Use the `Reformat` message to automatically format a script for easier reading.

```
ed << Reformat();
```

To run an entire script in a script window, use it the `Run` message.

```
ed << Run();
```

To close the script window without saving the script, send the window the `Close Window` message, just like you can do with any JMP window.

```
ww << Close Window( nosave );
```

To save and view the script, use the `Save Text` and `Load Text File` messages:

```
ww = New Window( "Test", <<Script, "Open(\!"$SAMPLE_DATA/Big
  Class.jmp\!");");
ww << Save Text( "$TEMP/Test.jsl" );
Write( Load Text File( "$TEMP/Test.jsl" ) );
```

Note: When the script is loaded into the script window, the window is considered "dirty" or modified. Use `<<Set Dirty(0)` to remove this property. Otherwise, you are prompted to save the script when you close the window.

To save the script and close the window, follow this example:

```
ww = New Window( "Test", <<Script, "Open(\!"$SAMPLE_DATA/Big
  Class.jmp\!");");
ww << Save Text( "$TEMP/Test.jsl" );
ww << Close Window();
```

Technical Details

Write Tab Box and Tab Page Box Scripts

`Tab Page Box()` organizes the title and contents of the page in one display box. When `Tab Page Box()` is inside `Tab Box()`, a tabbed window with multiple tabbed pages appears. See "Tab Box and Tab Page Box".

In previous versions of JMP, display boxes such as `V List Box()` contained tab contents. Consider the following script:

```
win = New Window( "Tab Box",
   tb = Tab Box(
      "First page", // name of the tab
      V List Box( // tab contents
         Text Box( "first line of first page" ),
         Text Box( "second line of first page" )
      ),
      "Second page", // name of the tab
      V List Box( // tab contents
         Text Box( "first line of second page" ),
         Text Box( "second line of second page" )
      ),
```

```
    )
);
```

We recommend that `Tab Box()` contain a `Tab Page Box()` for each tab. The preceding example is rewritten as follows:

```
win = New Window( "Tab Box",
   tb = Tab Box(
      Tab Page Box( // tab contents
         Title( "First page of my tab box" ), // name of the tab
         Text Box( "first line of first page" ),
         Text Box( "second line of first page" )
      ),
      Tab Page Box( // tab contents
         Title( "Second page" ), // name of the tab
         Text Box( "first line of second page" ),
         Text Box( "second line of second page" )
      )
   )
);
```

Note: `Tab Page Box()` requires that the title be defined in a `Title()` function.

When a tab page box is created or dragged outside a tab box, the tab page box is a stand-alone container. The title appears in a shaded box at the top of the page, similar to the title in `Sheet Part()`; the title does not appear in an interactive tab. "Dashboard with One Row of Reports" provides an example.

Deprecated Tab Box and Tab Page Box Messages

Some messages previously used for `Tab Box()` are deprecated. Their counterparts are compatible with `Tab Page Box()`.

Table 11.5 Deprecated Tab Box() Messages

Deprecated Tab Box Message	Current Tab Page Box Message
`Set Title`	`Title`
`Set Tip`	`Tip`
`Set Icon`	`Icon`
`Set Closeable`	`Closeable`

See "Tab Box and Tab Page Box" for more information about the two display boxes.

Chapter 12

Scripting Graphs

Editing and Creating Two-Dimensional Graphs

The JSL script for a graph contains expressions that create shapes and lines, apply fill patterns, assign background colors, and so on. You can also right-click a graph and select Customize to add the same expressions.

This chapter also describes how to customize hover labels with rich text, images, and drill-down graphs.

For information about scripting three-dimensional graphs, see the chapter "Three-Dimensional Scenes".

Contents

Add Scripts to Graphs 659
Specify the Order of Graphical Elements 660
Create New Graphs from Scratch and Customize Them 666
Customize Graphs 667
Create Shapes for a Bubble Plot 669
How to Find Shapes 672
Add Lines, Arrows, Points, Shapes, and Text 673
Draw Lines 673
Draw Arrows 675
Draw Markers 677
Draw Pies and Arcs 678
Draw Regular Shapes: Circles, Rectangles, and Ovals 680
Draw Irregular Shapes: Polygons and Contours 682
Add Text 685
Specify Colors 686
Specify Transparency 690
Add Fill Patterns 691
Specify Line Types 694
Draw with Pixels 695
Interactive Graphs 696
Handle() 696
Mousetrap() 701
Drag Functions 703
Troubleshoot Interactive Graphs 706
Create a Color Picker 706
Clip Display Segs in Graphics 707
Create Background Maps 709
Graph Elements 710
Plot Functions 711
Get the Properties of a Graphics Frame 717
Add a Legend 717
Hover Labels 718
Add Rich Text to Hover Labels Using Textlets 719
Add Graphs or Images to Hover Labels Using Graphlets 722
Update Names and Values in Hover Labels Using Gridlets 732
Create Drill-Down Graphs 737
Work with the Hover Label Execution Context 740

Add Scripts to Graphs

When you right-click a graphics frame, you can enter or paste JSL commands as shown in the following example. The script usually contains drawing commands that run in the context of the graphics frame. The context of the graphics frame includes the data range of the axes and the order that the data and scripts are drawn in.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Select **Analyze** > **Fit Y by X**.
3. Choose weight for Y and height for X, and then click **OK**.
4. Right-click inside the graph and select **Customize**.

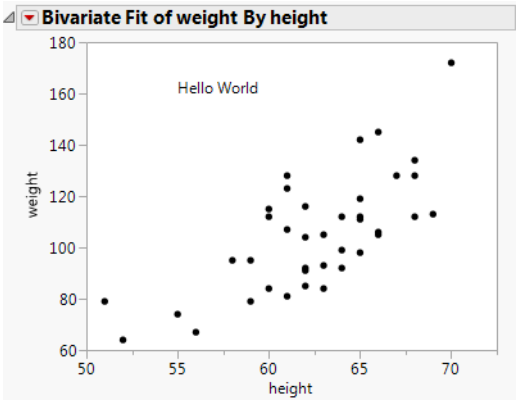
Figure 12.1 Customizing the Graph Options

5. Click the **Add** button ➕ to add a new graphics script.
6. Type the following text and click **OK**.

```
Text( {55, 160}, "Hello World" );
```

Now the graph has a text element at the graph's *x*-coordinate 55 and *y*-coordinate 160.

Figure 12.2 Adding a Script to a Graph Interactively



By default, the data points appear on top of graphics.

For example, add a new script in the Customize Graph window:

```
Fill Color( "Green" ); Rect( 57, 175, 65, 110, 1 );
```

A solid green rectangle appears. The list of scripts shows the order in which each script is drawn, so the first item on the list is drawn first.

- If you want the rectangle behind everything, move the rectangle script above the text script: select the rectangle script and click the up button .
- If you want the rectangle on top of everything, select the rectangle script and click the down button .

You can arrange all the scripts in a graph into the drawing order that you prefer. Any new script is initially added directly after the item in the list that is selected.

Tip: To use a script that references a column name, use `Column( colname )` or a colon (`:colname`) to scope it properly.

Hint: To see the JSL for the above actions, select **Save Script > To Script Window** from the red triangle menu.

Specify the Order of Graphical Elements

You can also add graphics elements using JSL instead of doing so interactively:

```
Frame Box <<Add Graphics Script(<order>, <"description">, script)
```

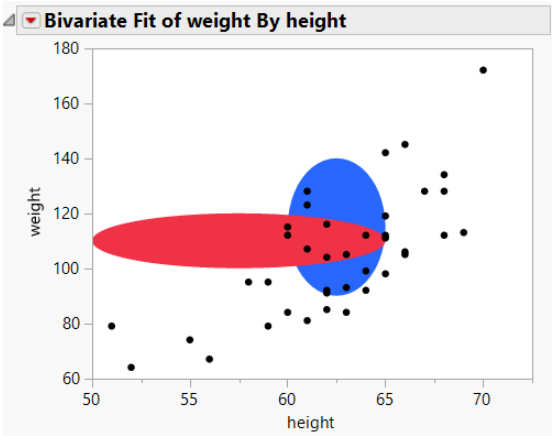
When you add graphical elements using JSL, they are drawn on top of whatever is already in the graph.

- The optional `order` argument specifies in what order to draw the graphics element. The order can be the keyword `Back` or `Front` *or* an integer that specifies the drawing order for a number of graphics elements. For example, if you add an oval to a scatterplot, the oval is drawn on top of the markers. The keyword `Back` or `1` cause the oval to be drawn last. `Front` or `2` means the object is drawn first.
- The optional `Description` argument is a string that appears in the Customize Graph window next to the graphics script.

To specify the drawing order for a number of graphics elements, use an integer for the `order` argument to determine where each is drawn in relation to the others. The following script first adds a blue oval and then adds a red oval over the blue oval. Both ovals are behind the points.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
op = dt << Bivariate(
  X( :height ),
  Y( :weight ),
);

Report( op )[Frame Box( 1 )] << Add Graphics Script(
  1, // draw the blue oval
  Description( "Blue Oval" ),
  Fill Color( "Blue" );
  Oval( 60, 140, 65, 90, 1 );
);

Report( op )[Frame Box( 1 )] << Add Graphics Script(
  2, // draw the red oval over the blue oval
  Description( "Red Oval" ),
  Fill Color( "Red" );
  Oval( 50, 120, 65, 100, 1 );
);
```

Figure 12.3 Specifying the Drawing Order



Copy and Paste Frame Contents or Settings

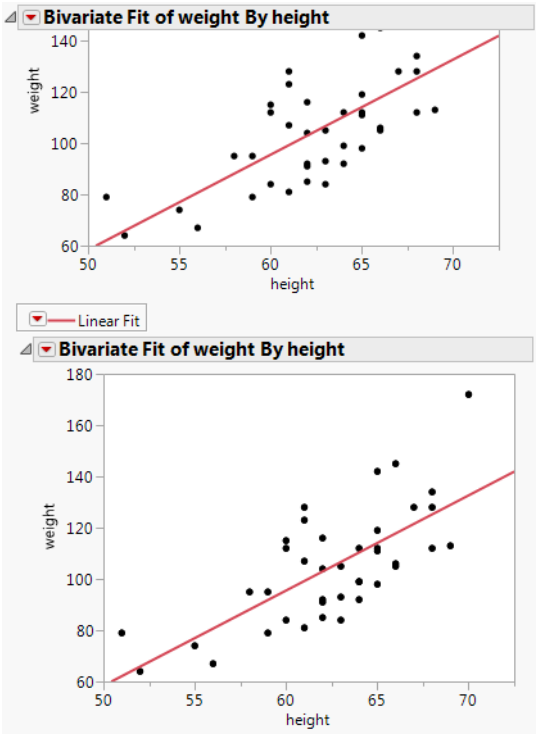
You can copy and paste the contents or the settings of a frame with these JSL commands:

```
obj << Copy Frame Contents // contents such as a graph or line of fit
obj << Paste Frame Contents
obj << Copy Frame Settings // settings such as background color
obj << Paste Frame Settings
```

The following example creates two Bivariate graphs, adds a line of fit to the first graph, and then copies and pastes the line to the second graph.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( y( :weight ), x( :height ) );
rbiv1 = biv << Report; // create a report object
rbiv2 = rbiv1 << Clone Box; // create a copy of the Bivariate graph

/* place the cloned box below the Bivariate graph
to create a second Bivariate graph */
rbiv1 << Append( rbiv2 );

framebox1 = rbiv1[Frame Box( 1 )]; /* assign references to
the reports' frame boxes*/
framebox2 = rbiv2[Frame Box( 1 )];
biv << Fit Line; // fit a line to the first Bivariate graph

// copy the frame contents from framebox1
framebox1 << Copy Frame Contents;

// paste the frame contents to framebox2
framebox2 << Paste Frame Contents;
```

Figure 12.4 Copying and Pasting Frame Contents

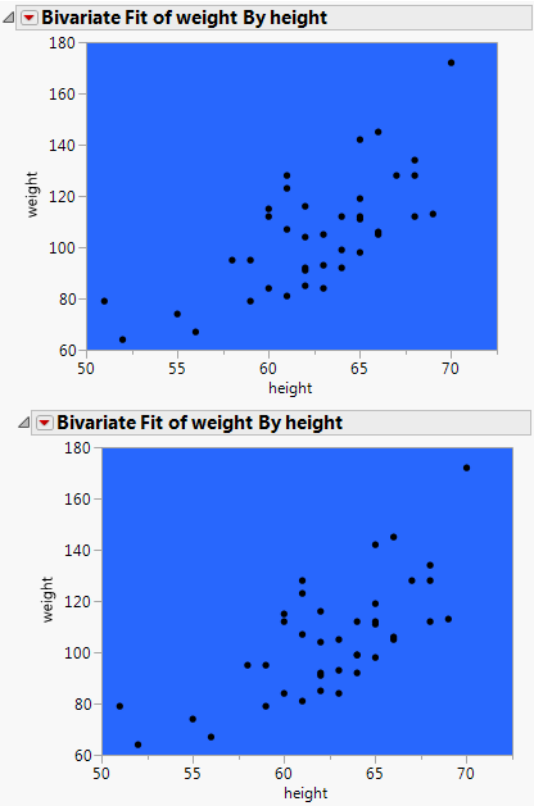


The following example creates two Bivariate graphs, sets the background of the first graph to blue, and then copies and pastes the blue background setting to the second graph:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );

biv1 = dt << Bivariate( y( :weight ), x( :height ) );

// create the first Bivariate graph
rbiv1 = biv1 << Report; // create a report object
rbiv2 = rbiv1 << Clone Box; // create a copy of the Bivariate graph

/* place the cloned box below the Bivariate graph
to create a second Bivariate graph */
rbiv1 << Append( rbiv2 );

framebox1 = rbiv1[Frame Box( 1 )]; /* assign a reference to the
                                      reports' frame boxes */
framebox2 = rbiv2[Frame Box( 1 )];

// assign the background color to framebox1
framebox1 << Background Color( "Blue" );
```

```
// copy the frame settings from framebox1
framebox1 << Copy Frame Settings;

// paste the frame settings to framebox2
framebox2 << Paste Frame Settings;
```

Figure 12.5 Copying and Pasting the Background



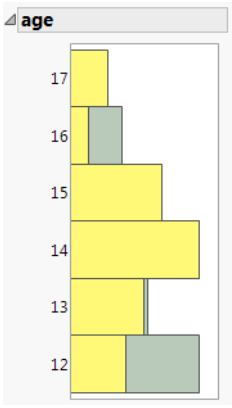
Copy and Paste Histograms

The following example shows how to create two histograms and copy the second histogram into the first histogram. A platform will not save a pasted-in segment, so you'll need to run a script like this to reproduce the graph.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dist = dt << Distribution(
   SendToByGroup( {:sex == "F"} ),
   Nominal Distribution( Column( :age ) ),
   Histograms Only,
   By( :sex ),
```

```
   SendToByGroup(
      {:sex == "M"},
      SendToReport(
         Dispatch(
            {"Distributions sex=M", "age"},
            "Distrib Nom Hist",
            FrameBox,
            {DispatchSeg(
               Hist Seg( 1 ),

               // set the histogram color for males
               Fill Color( "Light Yellow" ),
            )}
         )
      )
   )
);

For( i = 2, i <= N Items( dist ), i++,
   Report( dist[i] )[FrameBox( 1 )] << Copy Frame Contents;

   // copy the the second histogram into the first histogram
   Report( dist[1] )[FrameBox( 1 )] << Paste Frame Contents;
);

New Window( "Distribution", Outline Box( "age", Report( dist[1] )[Picture Box(
   1 )] ) );
```

Figure 12.6 Pasted Histogram



Interactively Copy and Paste Frame Contents and Settings

To interactively copy and paste the frame settings, follow these steps:

1. Right-click the graph and select **Edit > Copy Frame Settings**.
2. Right-click the frame to which you want to paste the settings and select **Edit > Paste Frame Settings**.

To interactively copy and paste the frame contents, follow these steps:

1. Right-click the graph and select **Edit > Copy Frame Contents**.
2. Right-click the frame to which you want to paste the contents and select **Edit > Paste Frame Contents**.

Create New Graphs from Scratch and Customize Them

Graphics scripts are set up inside the `Graph Box()` command within a `New Window()` command.

```
New Window("title", <arguments>, Graph Box( named arguments,..., script));
```

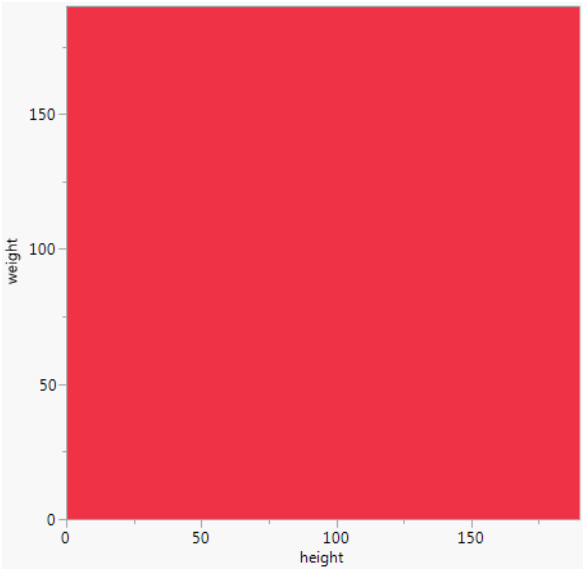
The following arguments are named arguments for `Graph Box()`:

```
Frame Size( horizontal, vertical ), // size in pixels
X Scale( xmin, xmax ), Y Scale( ymin, ymax ), // range of x and y axes
X Name( "x" ), Y Name( "y" ), // names for x and y axes
Suppress Axes // omit the axes
```

For example, `Graph Box()` accepts named arguments as shown in the following example, which creates a graph with a red background.

```
win = New Window( "Named arguments",
   Graph Box(
      Frame Size( 300, 300 ), // construct the graph box
      X Scale( 0, 190 ), // set the x-axis range
      Y Scale( 0, 190), // set the y-axis range
      X Name( "height" ),
      Y Name( "weight" ),
      <<Background Color( "Red" ) // set the background color
   )
);
```

Figure 12.7 Creating a Graph



Alternatively, you can use the send << operator to send commands to the `Graph Box` instead of using the named arguments. This example also creates a graph with a red background.

```
win = New Window( "Messages",
   Graph Box(
      <<Frame Size( 300, 300 ),
      <<XAxis( 0, 190 ),
      <<YAxis( 0, 190 ),
      <<Background Color( "Red" )
   )
);
```

Customize Graphs

You can also make changes to graphs in a script. For example, create a window with a graph, get a reference to the report, and then set the frame box size.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( :weight ), X( :height ), Fit Line );
rbiv = biv << Report;
rbiv[Frame Box( 1 )] << Frame Size( 400, 400 );
```

To see a list of possible messages for any given display box object, select **Help > Scripting Index** and select **Display Boxes** from the list. Running the `Show Properties()` command is an alternative to using the Scripting Index. For example, here is a partial list of messages that you can send to an axis:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( :weight ), X( :height ), Fit Line );
rbiv = biv << Report;
Show Properties( rbiv[Axis Box( 1 )] );
   Axis Settings [Action](Provides a way to bring up the axis dialog or set
      configurable axis options for a given axis.)
   Revert Axis [Action](Restore the settings that this axis had originally.)
   Add Axis Label [Action]
   Remove Axis Label [Action]
   ...
```

The following expressions change the font of both axis labels (weight and height in the example above) to 12-point, italic Arial Black:

```
rbiv[Text Edit Box( 1 )] << Set Font( "Arial Black", 12, Italic );
rbiv[Text Edit Box( 2 )] << Set Font( "Arial Black", 12, Italic );
```

or

```
rbiv[Text Edit Box( 1 )] << Set Font( "Arial Black" );
rbiv[Text Edit Box( 1 )] << Set Font Style( "Italic" );
rbiv[Text Edit Box( 1 )] << Set Font Size( 12 );
rbiv[Text Edit Box( 1 )] << Set Font( "Arial Black" );
rbiv[Text Edit Box( 2 )] << Set Font Style( "Italic" );
rbiv[Text Edit Box( 2 )] << Set Font Size( 12 );
rbiv[Text Edit Box( 2 )] << Set Font Name( "Arial Black" ) << Set Font Size(
  12 ) << Set Font Style( Italic )
```

The default font specified in the Preferences is applied if the font is not installed on the computer.

Specifying a Missing Color Value

When you specify a color theme in a script, you can specify the color of a missing value. Use the `Missing` argument. In this example, the missing color is yellow. Refer to an online RGB value table for the color values.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Graph Builder(
   Size( 528, 450 ),
   Show Control Panel( 0 ),
   Variables(
      X( :height ),
      Y( :weight ),
      Color( Transform Column( "wc", Formula( Root( :weight - 80 ) ) ) ) )
```

```
    ),
    Elements( Points( X, Y, Legend( 8 ) ) ),
    SendToReport(
       Dispatch( {}, "400", ScaleBox,
          {Legend Model( 8,
             Properties( 0, {gradient(
                {Color Theme(
                   {"Blue to Gray to Red Copy", 4099, {{42, 63, 255}, {192,
                   192, 192}, {252, 11, 11}, Missing( {255, 211, 0} )}}
                )}
             )},
             Item ID( "wc", 1 )
             )
          )}
       )
    )
);
```

Create Shapes for a Bubble Plot

Bubble Plot supports predefined shapes and custom shapes. Shapes can give visual cues about the data or provide a more interesting shape than a circle. For example, you can show arrows (Figure 12.8) or a hurricane symbol on a map (Figure 12.9).

Specify a Predefined Shape

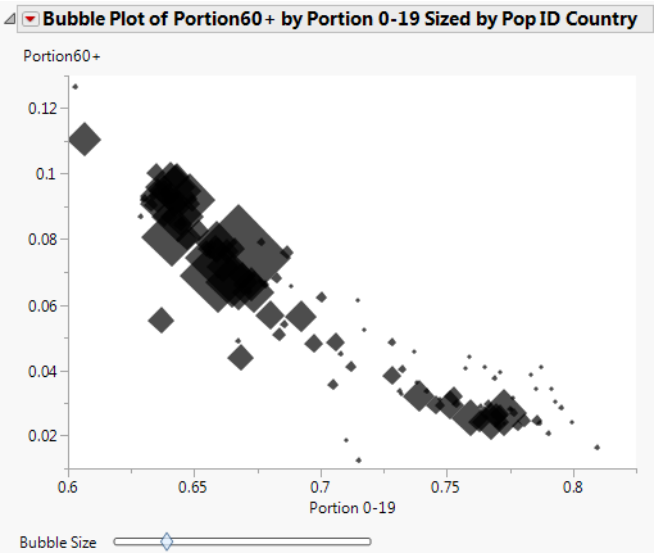
The `Set Shape` message specifies the predefined shape:

```
bp << Set Shape( "Circle"|"Triangle"|"Square"|"Diamond"|"Arrow" );
```

The following example creates the bubble plot and diamond shape shown in Figure 12.8:

```
dt = Open( "$SAMPLE_DATA/PopAgeGroup.jmp" );
bp = dt << Bubble Plot(
   X( :"Portion 0-19"n ),
   Y( :"Portion60+"n),
   Sizes( :Pop ),
   ID( :Country )
);
bp << Set Shape( "Diamond" );
```

Figure 12.8 Diamond Shapes



Create a Custom Shape

You can create custom shapes only in JSL through a combination of two messages:

```
bp << Set Custom Path( pathMatrix | pathText );
bp << Set Shape( "Custom" );
```

- In `Set Custom Path()`, the path can consist of an *N*x3 matrix or a string that contains SVG code. The default value is 0.

  ```
  bp << Set Custom Path(
     "M-1,-1 L-1,1 L0,0.5 L1,1 L1,-1 L0,-0.5 L-1,-1 Z, ..." );
  ```

- In `Set Shape()`, specify `"Custom"`.

  ```
  bp << Set Shape( "Custom" );
  ```

The following script creates the map and custom hurricane shape shown in Figure 12.9.

```
dt = Open( "$SAMPLE_DATA/Hurricanes.jmp" );
dt << Bubble Plot(
   X( :Longitude ),
   Y( :Latitude ),
   Sizes( "Wind (Knots)"n ),
   Time( :Date ),
   Coloring( :Landfall in USA ),
   ID( :Name and ID ),
   Speed( 220 ),
   Bubble Size( 51.43 ),
   Time Index( 729.9 ),
```

```
   Trail Bubbles( 1 ),
   All Labels( 0 ),
   No Labels( 0 ),
/* rotate shapes in the direction of the movement
   good for animated plots */
   Orient Shapes( 1 ),
   Set Custom Path( // specify the SVG string
      "M-0.99999898,0.00921400000000006 C-0.99999898,-0.51266322
   -0.55204076,-0.93622284 -0.000119060000000104,-0.93622284
   C0.1835362,-0.93622284 0.3556076,-0.88949112 0.5035232,-0.80766502
   C0.3236248,-1.36413554 0.0260398000000001,-1.815694896 -0.3555946,-2
   C0.4067376,-1.7567781 0.9372286,-0.91116206 1.0000004,0.00829720000000009
   C1.0000004,0.5301742 0.5518026,0.9548002 -0.00011905999999982,0.9547998
   C-0.18139054,0.9547998 -0.35150496,0.9090838 -0.49816916,0.8292164
   C-0.31759686,1.3771358 -0.0174124399999999,1.8176124 0.3602494,1.999999
   C-0.38991788,1.7606584 -0.92375834,0.908779 -0.99999898,0.00921400000000034
   z"
   ),
   Set Shape( "Custom" ), // set the custom shape
   Title Position( -88.679, 59.73 ),
   SendToReport(
      Dispatch(
         {},
         "Bubble Plot",
         FrameBox,
         {Frame Size( 743, 603 ), Background Map( Images( "Simple Earth" ) ),
         Grid Line Order( 3 ), Reference Line Order( 2 )}
      )
   )
);
```

Figure 12.9 Custom Hurricane Shape

How to Find Shapes

Search Wikimedia Commons or Google Images and include the keyword *SVG*. Look for SVG examples that are composed of stroke paths. SVG can be used to represent very complex drawings, with gradients and other elements. Only a path representation, which provides both outline and fill modes, is supported.

You might need to translate and scale the SVG that you find in order to get correct coordinates. Inkscape or similar tools can be useful for doing this manipulation. To copy the path from Inkscape, select **Edit > XML Editor**, find the `svg:path` element, and copy the path string from the `d Attribute` into `Set Custom Path()`.

Add Lines, Arrows, Points, Shapes, and Text

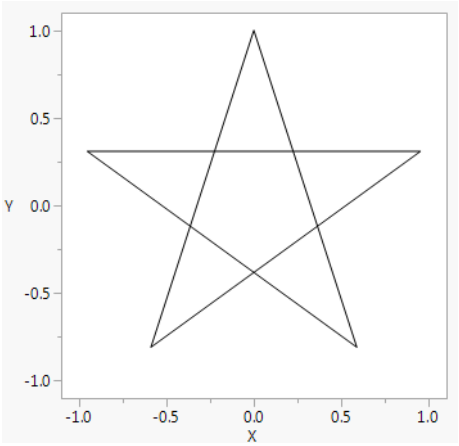
- "Draw Lines"
- "Draw Arrows"
- "Draw Markers"
- "Draw Pies and Arcs"
- "Draw Regular Shapes: Circles, Rectangles, and Ovals"
- "Draw Irregular Shapes: Polygons and Contours"
- "Add Text"
- "Specify Colors"
- "Specify Transparency"
- "Add Fill Patterns"
- "Specify Line Types"
- "Draw with Pixels"

Draw Lines

`Line()` draws lines between points.

```
win = New Window( "Five-Point Star",
   Graph Box(
      Frame Size( 300, 300 ),
      X Scale( -1.1, 1.1 ),
      Y Scale( -1.1, 1.1 ),
      Line(
         {Cos( 1 * Pi() / 10 ), Sin( 1 * Pi() / 10 )},
         {Cos( 9 * Pi() / 10 ), Sin( 9 * Pi() / 10 )},
         {Cos( 17 * Pi() / 10 ), Sin( 17 * Pi() / 10 )},
         {Cos( 5 * Pi() / 10 ), Sin( 5 * Pi() / 10 )},
         {Cos( 13 * Pi() / 10 ), Sin( 13 * Pi() / 10 )},
         {Cos( 1 * Pi() / 10 ), Sin( 1 * Pi() / 10 )}
      )
   )
);
```

Figure 12.10 Using Lines to Draw a Star



You can either specify the points in two-item lists as demonstrated above or as matrices of *x* and then *y* coordinates. Matrices are flattened by rows. You can use either row or column vectors, as long as you have the same number of elements in each matrix. The following expressions have the same effect:

```
Line( {1,2}, {3,0}, {2,4} ); // several {x,y} lists
Line( [1 3 2],[2 0 4] ); // row vectors
Line( [1,3,2], [2,0,4] ); // column vectors
Line( [1 3 2], [2,0,4] ); // one of each
```

The star example could also be drawn this way. Note that the script must include the full `Matrix( {...} )` notation rather than `[ ]` shorthand because the entries are expressions. The following example uses the `Matrix()` function.

```
win = New Window( "Five-Point Star",
   Graph Box(
      Frame Size( 300, 300 ),
      X Scale( -1.1, 1.1 ),
      Y Scale( -1.1, 1.1 ),
      Line(
         Matrix(
            { // the x coordinates
            Cos( 1 * Pi() / 10 ), Cos( 9 * Pi() / 10 ), Cos( 17 * Pi() / 10 ),
            Cos( 5 * Pi() / 10 ), Cos( 13 * Pi() / 10 ), Cos( 1 * Pi() / 10 )}
         ),
         Matrix(
            { // the y coordinates
            Sin( 1 * Pi() / 10 ), Sin( 9 * Pi() / 10 ), Sin( 17 * Pi() / 10 ),
            Sin( 5 * Pi() / 10 ), Sin( 13 * Pi() / 10 ), Sin( 1 * Pi() / 10 )}
         )
      ) ) );
```

`HLine()` draws a horizontal line across the graph at the *y*-value that you specify. Similarly, `VLine()` draws a vertical line down the graph at the *x*-value that you specify. Both functions support drawing multiple lines by using a matrix of values in the *y* argument. See "Mousetrap()".

Draw Arrows

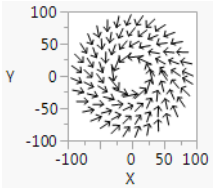
`Arrow()` draws an arrow from the first point to the second point. The default arrowhead is scaled to 1 plus the square root of the length of the arrow. To set the length of the arrowhead, add an optional first argument, specifying the length of the arrowhead in pixels. The following example draws simple arrows.

```
win = New Window( "Arrowheads",
   Graph Box(
      Pen Size( 4 );

      /* matrix 1 defines the x coordinates of an arrow
      matrix 2 defines the y coordinates of an arrow */
      Arrow( 20, [10 30 90], [88 22 44] );
   )
);
```

The following example draws arrows in a circular pattern.

```
win = New Window( "Hurricane",
   Graph Box(
      Frame Size( 100, 100 ),
      X Scale( -100, 100 ),
      Y Scale( -100, 100 ),
      For( r = 35, r < 100, r += 20,
         ainc = 2 * Pi() * 3 / r;
         For( a = 0, a < 2 * Pi(), a += ainc,
            x = r * Cosine( a );
            y = r * Sine( a );
            aa = a + ainc * 45 / r;
            rr = r - r / 6;
            x2 = rr * Cosine( aa );
            y2 = rr * Sine( aa );

            /* list 1 defines the start point for the arrow
            list 2 defines the end point for the arrow */
            Arrow( {x, y}, {x2, y2} );
         );
      )
   )
);
```

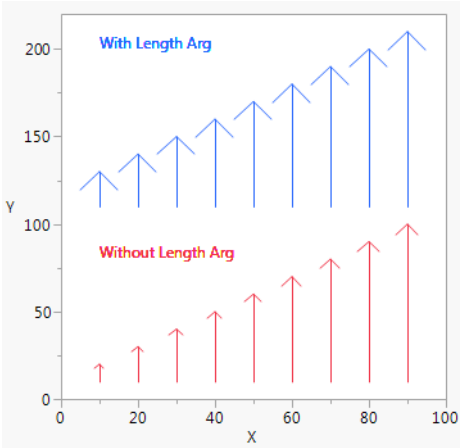
Figure 12.11 Drawing Arrows



The following example compares drawing with a specified length (19 pixels) and drawing with the default arrow head size.

```
win = New Window( "Arrowheads",
   Graph Box(
      Frame Size( 300, 300 ),
      X Scale( 0, 100 ),
      Y Scale( 0, 220 ),
      x = 10;
      y1 = 10;
      y2 = y1 + 10;
      For( i = 1, i < 10, i++,
         Pen Color( "Red" );
         Arrow( {x, y1}, {x, y2} );

         y2 += 10;
         y1 += 100;
         y2 += 100;
         Pen Color( "Blue" );
         Arrow( 20, {x, y1}, {x, y2} );

         x += 10;
         y1 -= 100;
         y2 -= 100;

         Text Color( "Red" );
         Text( {10, 80}, "Without Length Arg" );

         Text Color( "Blue" );
         Text( {10, 200}, "With Length Arg" );
      );
   )
);
```

Figure 12.12 Arrowhead Sizes



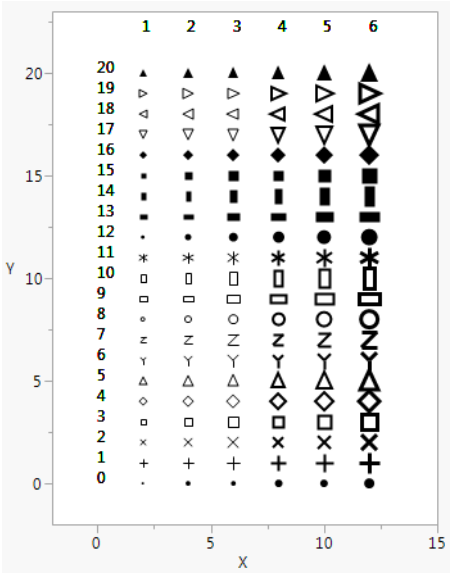
As with `Line()`, you can either specify the points in two-item lists as demonstrated above or as matrices of *x* and then *y* coordinates.

Draw Markers

`Marker()` draws a marker of the type that you specify (1–20) in the first argument at the point that you specify in the second argument. `Marker Size()` scales markers from 0 to 6 (dot to XXXL). To set markers to the size that is specified in the Preferences, use a value of `-1`.

```
ymax = 20;
win = New Window( "Markers",
   Graph Box(
      Frame Size( 300, 400 ),
      X Scale( -2, ymax - 5 ),
      Y Scale( -2, ymax + 3 ),
      For( j = 1, j < 7, j++,
         Marker Size( j );
         For( i = 0, i < (ymax + 1), i++,
            Marker( i, {j * 2, i} );
            Text( {0, i}, i );
            Text( {j * 2, ymax + 2}, j );
         );
      )
   )
);
```

Figure 12.13 Drawing Markers



You can also include a row state argument before, after, or instead of the marker ID argument. By using `Combine States()`, you can set multiple row states inside `Marker()`. Try substituting each of the following lines in the preceding script:

```
Marker( i, Color State( i ), {j * 2, i} );
Marker( Color State( i ), i, {j * 2, i} );
Marker(
   Combine States( Color State( i ), Marker State( i ), Hidden State( i ) ),
   {j * 2, i}
);
```

As with `Line()` and `Arrow()`, you can also specify points as matrices of *x* and then *y* coordinates.

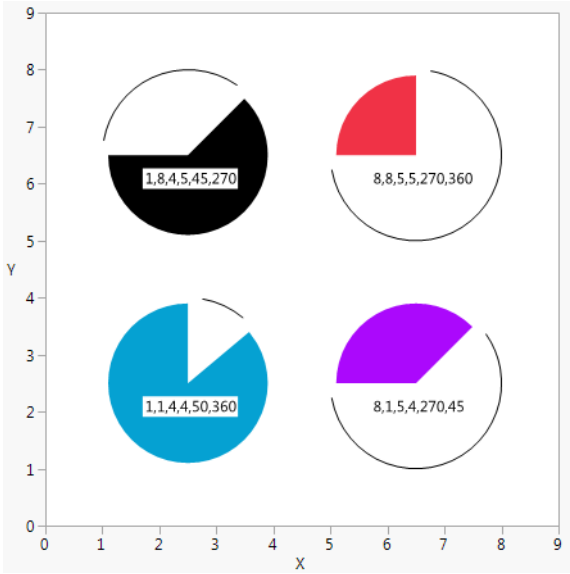
Draw Pies and Arcs

`Pie()` and `Arc()` draw wedges and arc segments. The first four arguments are *x1, y1, x2,* and *y2,* the coordinates of the rectangle to inscribe. The last two arguments are the starting and ending angle in degrees, where 0 degrees is 12 o'clock and the arc or slice is drawn clockwise from start to finish.

```
win = New Window( "Pies and Arcs",
   Graph Box(
      Frame Size( 400, 400 ),
      X Scale( 0, 9 ),
      Y Scale( 0, 9 ),
```

```
        Fill Color( "Black" ), // top left
        Pie( 1.1, 7.9, 3.9, 5.1, 45, 270 ),
        Text( Erased, {1.75, 6}, "1,8,4,5,45,270" ),
        Arc( 1, 8, 4, 5, 280, 35 ),
        Fill Color( "Red" ), // top right
        Pie( 7.9, 7.9, 5.1, 5.1, 270, 360 ),
        Text( Erased, {5.75, 6}, "8,8,5,5,270,360" ),
        Arc( 8, 8, 5, 5, 370, 260 ),
        Fill Color( "BlueCyan" ), // bottom left
        Pie( 1.1, 1.1, 3.9, 3.9, 50, 360 ),
        Text( Erased, {1.75, 2}, "1,1,4,4,50,360" ),
        Arc( 1, 1, 4, 4, 370, 40 ),
        Fill Color( "Purple" ), // bottom right
        Pie( 7.9, 1.1, 5.1, 3.9, 270, 45 ),
        Text( Erased, {5.75, 2}, "8,1,5,4,270,45" ),
        Arc( 8, 1, 5, 4, 55, 260 )
    )
);
```

Figure 12.14 Drawing Pies and Arcs



Draw Regular Shapes: Circles, Rectangles, and Ovals

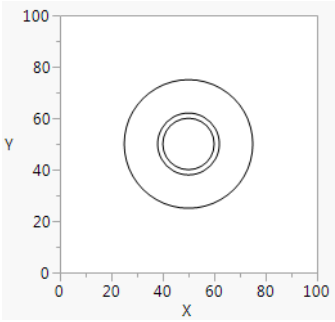
Circles

`Circle()` draws a circle with the center point and radius given. Subsequent arguments specify additional radii.

```
win = New Window( "Circles",
   Graph Box( Frame Size( 200, 200 ),
      Circle( {50, 50}, 10, 12, 25 )
   )
);
```

`Circle()` also has a final optional argument, `"fill"`. This string indicates that all circles defined in the function are filled with the current fill color. If `"fill"` is omitted, the circle is empty.

Figure 12.15 Drawing Circles



Note that a circle is always a circle, even if you resize the graph to a different aspect ratio. If you want your circle to change aspect ratios (in other words, cease being a circle) when the graph is resized, use an oval instead.

If you do not want your circle to resize if the graph is resized, specify the radius in pixels instead:

```
win = New Window( "Circles",
   Graph Box(
      Frame Size( 200, 200 ),
      Circle( {50, 50}, Pixel Radius( 10 ), Pixel Radius( 12 ), Pixel Radius(
   25 ) )
   )
);
```

Rectangles

`Rect()` draws a rectangle from the diagonal coordinates you specify. The coordinates can be specified either as four arguments in order (*left*, *top*, *right*, *bottom*), or as a pair of lists ({*left*, *top*}, {*right*, *bottom*}).

```
win = New Window( "Rectangles",
   Graph Box(
      Frame Size( 200, 200 ),
      Pen Color( 1 );
      Rect( 0, 40, 60, 0 );
      Pen Color( 3 );
      Rect( 10, 60, 70, 10 );
      Pen Color( 4 );
      Rect( 50, 90, 90, 50 );
      Pen Color( 5 );
      Rect( 0, 80, 70, 70 );
   )
);
```

`Rect()` has an optional fifth argument, *fill*. Specify 0 to get an unfilled rectangle and 1 to get a filled rectangle. The default value for *fill* is 0. The rectangle is filled with the current `fill color`.

Any negative fill argument produces an unfilled frame inset by one pixel:

```
win = New Window( "Framed Rectangle",
   Graph Box( Frame Size( 200, 200 ), Rect( 0, 40, 60, 0, -1 ) )
);
```

Ovals

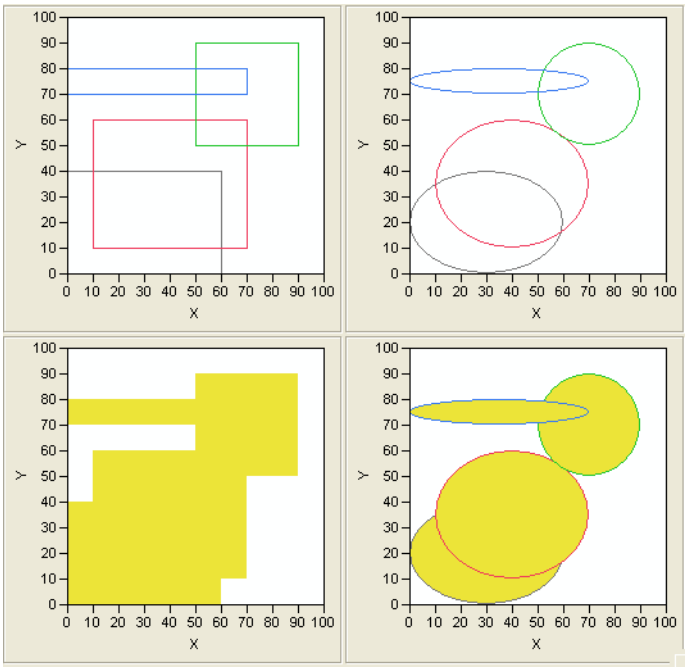
`Oval()` draws an oval inside the rectangle given by its *x1*, *y1*, *x2*, and *y2* arguments:

```
win = New Window( "Ovals",
   Graph Box(
      Frame Size( 200, 200 ),
      Pen Color( 1 );
      Oval( 0, 40, 60, 0 );
      Pen Color( 3 );
      Oval( 10, 60, 70, 10 );
      Pen Color( 4 );
      Oval( 50, 90, 90, 50 );
      Pen Color( 5 );
      Oval( 0, 80, 70, 70 );
   )
);
```

`Oval()` also has an optional fifth argument, *`fill`*. Specify 0 to get an unfilled rectangle and 1 to get a filled oval. The default value for *`fill`* is 0. The oval is filled with the current `fill color`.

Figure 12.16 shows rectangles and ovals, drawn both filled and unfilled. Notice that filled rectangles do not have outlines. Ovals do have outlines. If you want a filled rectangle with an outline, you must draw the filled rectangle and then draw an unfilled rectangle with the same coordinates.

Figure 12.16 Rectangles and Ovals, Unfilled and Filled



Draw Irregular Shapes: Polygons and Contours

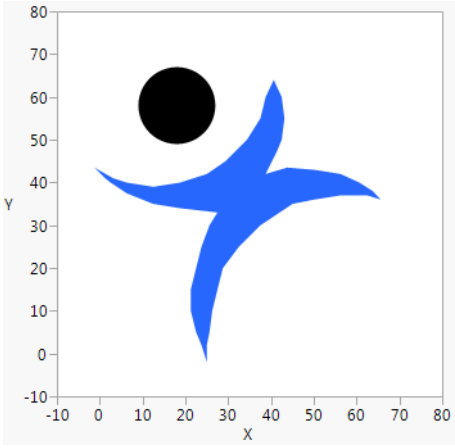
Polygons

`Polygon()` works similarly to `Line()`. They both connect points, but `Polygon()` returns to the first point to close the polygon and filling the resulting area. You can specify the points as individual points in two-item lists (as shown in "Draw Markers") or as matrices of *x* and then *y* coordinates. Matrices are flattened by rows, so you can use either row or column vectors, as long as you have the same number of elements in each matrix. First set up the matrices of points and then call them inside `Polygon()`.

```
gCoordX = [25, 23.75, 22.5, 21.25, 21.25, 22.5, 23.75, 25.625, 27.5, 18.75,
  12.5, 6.25, 2.5,
```

```
1.25, -1.25, 3.125, 6.25, 12.5, 18.75, 25, 29.375, 34.375, 37.5, 38.75,
  40.625, 42.5, 43.125,
42.5, 41.25, 38.75, 43.75, 50, 56.25, 60.625, 63.75, 65.625, 62.5, 56.25, 50,
  45, 37.5, 32.5,
28.75, 27.5, 26.25, 25.625, 25];
gCoordY = [-2, 2, 5, 10, 15, 20, 25, 30, 33, 34, 35, 37.5, 40, 41, 43.5, 41,
  40, 39, 40, 42, 45,
50, 55, 60, 64, 60, 55, 50, 47, 42, 43.5, 43, 42, 40, 38, 36, 37, 37, 36, 35,
  30, 25, 20, 15, 10, 5, 2];
win = New Window( "The JMP Man",
  Graph Box(
     Frame Size( 300, 300 ),
     X Scale( -10, 80 ),
     Y Scale( -10, 80 ),
     Pen Color( "Black" );
     Fill Color( "Blue" );
     Polygon( gCoordX, gCoordY );
     Fill Color( "Black" );
     Circle( {18, 58}, 9, "FILL" );
  )
);
```

Figure 12.17 Drawing a Polygon



A related command, `In Polygon()`, tells whether a given point falls inside the specified polygon. This code checks some points from the JMP man pictured in Figure 12.17:

```
In Polygon( 0,60, GcoordX,GCoordY );  // return 0
In Polygon( 30,38, GcoordX,GCoordY ); // return 1
```

You can add `In Polygon()` to the JMP man script. Run the following script, and then click various locations in the picture and watch the log window. "In" and "out" are printed to the log when you click the body.

```
gCoordX = [25, 23.75, 22.5, 21.25, 21.25, 22.5, 23.75, 25.625, 27.5, 18.75,
   12.5, 6.25, 2.5, 1.25, -1.25, 3.125, 6.25, 12.5, 18.75, 25, 29.375, 34.375,
   37.5, 38.75, 40.625, 42.5, 43.125, 42.5, 41.25, 38.75, 43.75, 50, 56.25,
   60.625, 63.75, 65.625, 62.5, 56.25, 50, 45, 37.5, 32.5, 28.75, 27.5, 26.25,
   25.625, 25];
gCoordY = [-2, 2, 5, 10, 15, 20, 25, 30, 33, 34, 35, 37.5, 40, 41, 43.5, 41,
   40, 39, 40, 42, 45, 50, 55, 60, 64, 60, 55, 50, 47, 42, 43.5, 43, 42, 40,
   38, 36, 37, 37, 36, 35, 30, 25, 20, 15, 10, 5, 2];
win = New Window( "The JMP Man",
   Graph Box(
      Frame Size( 300, 300 ),
      X Scale( -10, 80 ),
      Y Scale( -10, 80 ),
      Pen Color( "Black" );
      Fill Color( "Black" );
      Polygon( gCoordX, gCoordY );
      Mousetrap(
         {},
         Print( If( In Polygon( x, y, gCoordX, gCoordY ), "in", "out" ) )
      );
   )
);
```

Contours

`Contour()` draws contour lines using a grid of coordinates.

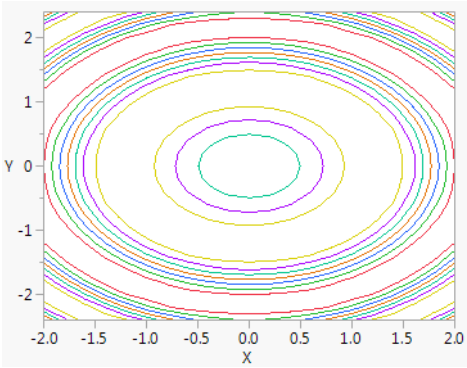
```
Contour( xVector, yVector, zGridMatrix, zContour, <zColors> );
```

Suppose that you have an **n** by **m** matrix *zGridMatrix* of values on some surface (defined across the *n* values of *xVector* by the **m** values of *yVector*). `Contour()` draws the contour lines that are defined by the values in *zContour* in the colors defined by *zColors*. Here is an example:

```
x = (-10 :: 10) / 5;
y = (-12 :: 12) / 5;
grid = J( 21, 25, 0 );
z = [-.75, -.5, -.25, 0, .25, .5, .75];
zcolor = [3, 4, 5, 6, 7, 8, 9];
For( i = 1, i <= 21, i++,
   For( j = 1, j <= 25, j++,
      grid[i, j] = Sin( (x[i]) ^ 2 + (y[j]) ^ 2 )
   )
);
```

```
Show( grid );
win = New Window( "Hat",
   Graph Box(
      X Scale( -2, 2 ),
      Y Scale( -2.4, 2.4 ),
      Contour( x, y, grid, z, zcolor )
   )
);
```

Figure 12.18 Drawing Contour Lines



Add Text

You can use `Text()` to draw text at a given location.

```
Text( <properties>, ( {x, y}|{left, bottom, right, top} ), "text" );
```

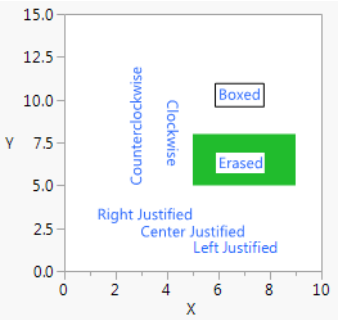
The point and text can be in any order and repeated. You can precede the point and text with an optional first argument: `Center Justified`, `Right Justified`, `Erased`, `Boxed`, `Counterclockwise`, or `Clockwise`. `Erased` is for removing whatever would otherwise obscure the text in a graph. It paints a background-colored rectangle behind the text.

In the following example, notice how the erased text appears inside a white box over the green `Rect()`.

```
mytext = New Window( "Adding Text",
   Graph Box(
      Frame Size( 200, 200 ),
      Y Scale( 0, 15 ),
      X Scale( 0, 10 ),
      Text Size( 9 );
      Text Color( "blue" );
      Text( {5, 1}, "Left Justified" );
      Text( Center Justified, {5, 2}, "Center Justified" );
      Text( Right Justified, {5, 3}, "Right Justified" );
```

```
        Fill Color( 4 );
        Rect( 5, 8, 9, 5, 1 );
        Text( Erased, {6, 6}, "Erased" );
        Text( Boxed, {6, 10}, "Boxed" );
        Text( Clockwise, {4, 10}, "Clockwise" );
        Text( Counterclockwise, {3, 5}, "Counterclockwise" );
    )
);
```

Figure 12.19 Drawing Text in a Graph Box



There is a variant of the `Text()` function that draws a string inside the rectangle that is specified by four coordinates. Here is the syntax:

```
Text( {left, top, right, bottom}, string);
```

Specify Colors

The following commands control colors:

- `Fill Color()` for solid areas
- `Level Color()` for categorical data
- `Pen Color()` for lines and points
- `Back Color()` for the background of text (similar to the box around the erased text in Figure 12.19)
- `Background Color()` for the graph's background color
- `Font Color()` for added text

Fill colors overwrite pen colors for drawn shapes. You do not get both, as in some drawing packages. To get both a fill and a pen line, draw two shapes, one with fill and one without. You can select a color with a single numeric argument, a color name in quotation marks, or an RGB value. The standard colors are specified with numbers 0–15 (both 0 and 15 are black) or by their names (Table 12.1).

Tip: For more shades, add the word Dark, Medium Dark, Medium Light, or Light before a color name in quotation marks.

Table 12.1 Standard JMP Colors

Number	Name
0	Black
1	Gray
2	White
3	Red
4	Green
5	Blue
6	Orange
7	BlueGreen
8	Purple
9	Yellow
10	Cyan
11	Magenta
12	YellowGreen
13	BlueCyan
14	Fuchsia

The following script creates a graph that shows all JMP colors:

```
Text Color( 0 );
New Window( "Colors",
  Graph Box(
     FrameSize( 640, 400 ),
     Y Scale( -1, 17 ),
     X Scale( -3, 12 ),
     k = 0;
     For( jj = 1, jj <= 12, jj += 2,
        l = 15;
```

```
            For( i = 0, i <= 15 & k < 85, i++,
               thiscolor = Color To RGB( k );
               Fill Color( k );
               thisfill = 1;
               If( thiscolor == {1, 1, 1},
                  Pen Color( 0 );
                  thisfill = 0;
               ,
                  Pen Color( k )
               );
               Rect( jj, l + .5, jj + .5, l, thisfill );
               Text( {jj - 1, l}, "color ", k );
               k++;
               l--;
            );
         );
         jj = -2;
         color = {"Black", "Gray", "White", "Red", "Green", "Blue", "Orange",
      "BlueGreen",
         "Purple", "Yellow", "Cyan", "Magenta", "YellowGreen", "BlueCyan",
      "Fuschia", "Black"};
         For(
            i = 0;
            l = 15;, i <= 15 & l >= 0,
            i++;
            l--;,
            Text( {jj, l}, color[i + 1] )
         );
      ) );
```

Larger numbers cycle through shading variations of the same color sequence. A script demonstrating this appears under "Colors and Markers". Values outside the range 0–84 are not accepted.

Table 12.2 How Numbers Map to Colors

Number	Result
16–31	Medium Dark
32–47	Medium Light
48–63	Dark
64–79	Light

Table 12.2 How Numbers Map to Colors *(Continued)*

Number	Result
80–84	Light, Medium Light, Gray, Medium Dark, Dark

If you prefer to use RGB values, create a list with the fraction for each color in red, green, and blue order.

```
Pen Color( {.38,.84,.67} ); // teal
```

`RGB Color()` and `Color to RGB()` convert color values between JMP color numbers and the red-green-blue system. The following example finds the RGB values for JMP color 3 (red):

```
Color to RGB( 3 );
  {0.941176470588235, 0.196078431372549, 0.274509803921569}
```

Likewise, `HLS Color()` and `Color to HLS()` convert color values between JMP color numbers and the hue-lightness-saturation system.

```
win = New Window( "Color Wheel",
   Graph(
      Frame Size( 200, 200 ),
      For( hue = 0, hue < 360, hue += 30,
         y = 50 - 40 * Cos( hue * 2 * Pi() / 360 );
         x = 50 + 40 * Sin( hue * 2 * Pi() / 360 );
         Fill Color( HLS Color( hue / 360, 0.5, 1 ) );
         Oval( x - 10, y - 10, x + 10, y + 10, 1 );
      )
   )
);
```

Finally, `Heat Color()` returns the JMP color that corresponds to a value in any color theme that is supported by Cell Plot, Treemap, and so on. The syntax is:

```
Heat Color( n, <<"theme" )
```

The theme message is optional, and the default value is `"Blue to Gray to Red"`. You can specify any color theme, including custom color themes. You can also create and use an anonymous color theme as shown in the following examples:

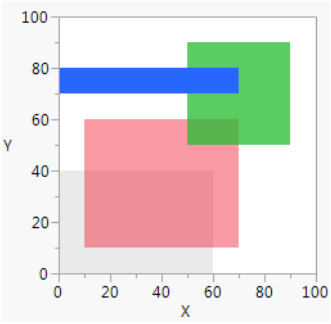
```
Heat Color( z, <<{"", {{1, 1, 0}, {0, 0, 1}} } );
Heat Color( z, <<{"", {blue, green, yellow} } );
```

Specify Transparency

In a graphics environment (like a Frame Box), use the Transparency function to set the level of transparency. The argument, alpha, can be any number between zero and one. The value 0 means clear and drawing has no effect. The value 1 means completely opaque and is the usual drawing mode. Intermediate values build semi-transparent color layers on top of what has already been drawn below it. The following example script illustrates transparency with rectangles.

```
win = New Window( "Transparency",
   Graph Box(
      Frame Size( 200, 200 ),
      Pen Color( "gray" );
      Fill Color( "gray" );
      Transparency( 0.25 );
      Rect( 0, 40, 60, 0, 1 );

      Pen Color( "red" );
      Fill Color( "red" );
      Transparency( 0.5 );
      Rect( 10, 60, 70, 10, 1 );

      Pen Color( "green" );
      Fill Color( "green" );
      Transparency( 0.75 );
      Rect( 50, 90, 90, 50, 1 );

      Pen Color( "blue" );
      Fill Color( "blue" );
      Transparency( 1 );
      Rect( 0, 80, 70, 70, 1 );
   )
);
```

Figure 12.20 Transparency and Rectangles



Add Fill Patterns

`Fill Pattern()` sets the pattern for filled areas.

Matrix Example

Specify a mask (a matrix of values between 0 and 1) or an image. Each value in the matrix creates a pixel.

```
win = New Window( "Example",
   Graph Box(
      Fill Pattern(
         [1 0.5 0 0, 0.5 0 0 1, 0 0 1 0.5, 0 1 0.5 0]
      );
      Polygon( [10 30 90], [88 22 44] );
   )
);
```

Named Pattern Example

Specify a name in quotation marks that corresponds to a pattern shown in Table 12.3.

```
win = New Window( "Named Pattern Example",
   Graph Box(
      Fill Pattern( "vertical light" );
      Polygon( [10 30 90], [88 22 44] );
   )
);
```

Graphic Example

Specify an image in quotation marks. If the image is not installed on the viewer's computer, question marks appear in the shape.

```
win = New Window( "Graphic Example",
   Graph Box(
      Fill Pattern( Open( "$SAMPLE_IMAGES/pi.gif", "gif" ) );
      Polygon( [10 30 90], [88 22 44] );
   )
);
```

Table 12.3 Fill Patterns

Pattern Name	Pattern
left slant light	

Table 12.3 Fill Patterns *(Continued)*

Pattern Name	Pattern
right slant light	
vertical light	
horizontal light	
grid light	
hatch light	
h wave light	
v wave light	
hollow circle	
left slant medium	
right slant medium	
vertical medium	
horizontal medium	
grid medium	
hatch medium	
h wave medium	
v wave medium	
filled circle	

Table 12.3 Fill Patterns *(Continued)*

Pattern Name	Pattern
left slant heavy	
right slant heavy	
vertical heavy	
horizontal heavy	
grid heavy	
hatch heavy	
h wave heavy	
v wave heavy	
diamond	
left slant heavy b	
right slant heavy b	
random	
square	
square offset	
wide	
tall	
checkerboard	

Table 12.3 Fill Patterns *(Continued)*

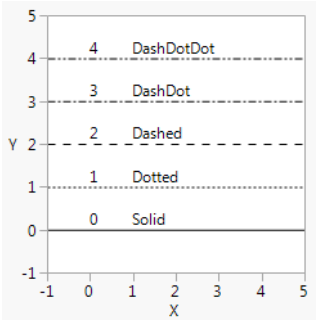
Pattern Name	Pattern
grid dots	
triangle up	
triangle down	
triangle left	
triangle right	
weave light	
weave heavy	
honeycomb	

Specify Line Types

You can also control `Line Style()` by number (0–4) or name (`Solid`, `Dotted`, `Dashed`, `DashDot`, `DashDotDot`). Figure 12.21 shows the numerical values for each line type.

```
linestyles = {"Solid", "Dotted", "Dashed", "DashDot", "DashDotDot"};
win = New Window( "Line Styles",
   Graph Box(
      Frame Size( 200, 200 ),
      X Scale( -1, 5 ),
      Y Scale( -1, 5 ),
      For( i = 0, i < 5, i++,
         Line Style( i );
         H Line( i );
         Text( {0, i + .1}, i );
         Text( {1, i + .1}, linestyles[i + 1] );
      )
   )
);
```

Figure 12.21 Line Styles



To control the thickness of lines, set a `Pen Size` and specify the line width in pixels. The default is 1 for single-pixel lines. For printing, think of `Pen Size` as a multiplier for the default line width, which varies according to your printing device.

```
win = New Window( "Pen Size",
   Graph Box(
      Pen Size( 2 ); // double-width lines
      Line( [10 30 90], [88 22 44] );
   )
);
```

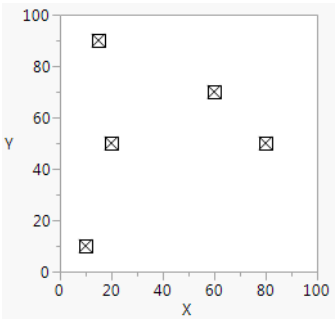
Draw with Pixels

You can also draw using pixel coordinates. First, set the `Pixel Origin()` in terms of graph coordinates, and then use `Pixel Move To()` or `Pixel Line To()` in pixel coordinates relative to that origin. The main use for `Pixel()` functions is for drawing custom markers that do not vary with the size or scale of the graph. You can store a marker in a script and then call it within any graph. This example uses `Function()` to store pixel commands in a script with its own arguments, *x* and *y*.

```
ballotBox = Function( {x, y},
   Pixel Origin( x, y );
   Pixel Move To( -5, -5 );
   Pixel Line To( -5, 5 );
   Pixel Line To( 5, -5 );
   Pixel Line To( -5, -5 );
   Pixel Line To( 5, 5 );
   Pixel Line To( -5, 5 );
   Pixel Move To( 5, 5 );
   Pixel Line To( 5, -5 );
);
win = New Window( "Custom Markers",
   Graph Box(
```

```
        Frame Size( 200, 200 ),
        ballotBox( 10, 10 );
        ballotBox( 15, 90 );
        ballotBox( 20, 50 );
        ballotBox( 80, 50 );
        ballotBox( 60, 70 );
    )
);
```

Figure 12.22 Drawing Custom Markers



Interactive Graphs

`Handle()` and `Mousetrap()` are functions for making interactive graphs that respond to clicking and dragging. `Handle()` lets you parametrize a graph by adding a handle-marker that can be dragged around with the mouse, executing the graph's script at each new location. `Mousetrap()` is similar, but it takes its arguments from the coordinates of a click without dragging a handle. The main difference is that `Handle()` only catches mousedowns at the handle-marker's location, but `Mousetrap()` catches mousedowns at any location.

Another approach is to place buttons or slider controls outside the graph with `Button Box()`, `Slider Box()`, or `Global Box()`.

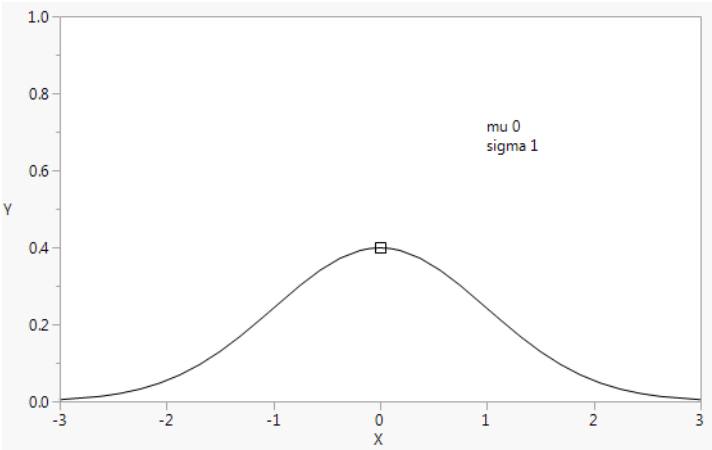
Handle()

`Handle()` places a marker at the coordinates given by the initial values of the first two arguments and draws the graph using the initial values of the arguments. You can then click and drag the marker to move the handle to a new location. The first script is executed at each mousedown to update the graph dynamically, according to the new coordinates of the handle. The second script (optional, and not used here) is executed at each mouseup, similarly; see the example for "Mousetrap()".

```
// Normal Density
mu = 0;
sigma = 1;
rsqrt2pi = 1 / Sqrt( 2 * Pi() );
win = New Window( "Normal Density",
  Graph Box(
     Frame Size( 500, 300 ),
     X Scale( -3, 3 ),
     Y Scale( 0, 1 ),
     Y Function( Normal Density( (x - mu) / sigma ) / sigma, x );
     Handle(
        mu,
        rsqrt2pi / sigma,
        mu = x;
        sigma = rsqrt2pi / y;
     );
     Text( {1, .7}, "mu ", mu, {1, .65}, "sigma ", sigma );
  )
);
```

Run demoPlotProb.jsl in the JMP Samples/Scripts folder to see Beta Density, Gamma Density, Weibull Density, and LogNormal Density graphs. The output for Normal Density is shown in Figure 12.23. Because we cannot show you the picture in motion, be sure to run this script yourself.

Figure 12.23 Normal Density Example for `Handle()`



To avoid errors, be sure to set the initial values of the handle's coordinates, as in the first line of this example.

If you want to use some function of a handle's coordinates, such as in the normal density example, you should adjust the arguments for `Handle()`. Otherwise, the handle marker would run away from the mouse.

```
Y Function( a * x ^ b );
Handle( a, b, a = 2 * x, b = y );
```

Suppose you drag the marker from its initial location to (3,4). The argument *a* is set to 6 and *b* to 4; the graph is redrawn as $Y = 6x^4$; and the handle is now drawn at (6,4), several units away from the mouse. To compensate, you would adjust the first argument to handle, for example.

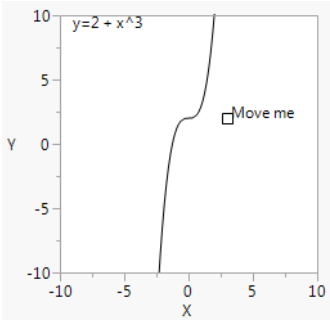
```
Handle( a / 2, b, a = 2 * x; b=y );
```

To generalize, suppose you define the `Handle()` arguments as functions of the handle's (x, y) coordinates. For example, `a=f(x)` and `b=g(y)`. If `f(x)=x` and `g(y)=y`, then you would specify simply `a,b` as the first two arguments. If not, you would solve $a = f(x)$ for x and solve $b = g(y)$ for y to get the appropriate arguments.

You can use other functions to constrain `Handle()`. For example, here is an interactive graph to demonstrate power functions that uses `Round()` to prevent bad exponents and to keep the intercepts simple.

```
a = 3;
b = 2;
win = New Window( "Intercepts and Powers",
   Graph Box(
      Frame Size( 200, 200 ),
      X Scale( -10, 10 ),
      Y Scale( -10, 10 ),
      Y Function( Round( b ) + x ^ (Round( a )), x );
      Handle(
         a,
         b,
         a = x;
         b = y;
      );
      Text( {a, b}, " Move me" );
      Text( {-9, 9}, "y=", Round( b ), " + x^", Round( a ) );
   )
);
```

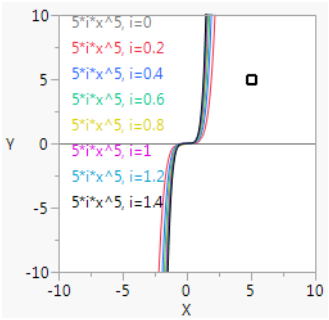
Figure 12.24 Intercepts and Powers for `Handle()`



`Handle()` and `For()` can be nested for complex graphs.

```
a = 5;
b = 5;
win = New Window( "Powers",
   Graph Box(
      Frame Size( 200, 200 ),
      X Scale( -10, 10 ),
      Y Scale( -10, 10 ),
      For( i = 0, i < 1.5, i += .2,
         Pen Color( 1 + 10 * i );
         Text Color( 1 + 10 * i );
         Y Function( i * x ^ Round( a ), x );
         Handle(
            a,
            b,
            a = x;
            b = y;
         );
         h = 9 - 10 * i;
         Text( {-9, h}, b, "*i*x^", Round( a ), ", i=", i );
      ) ) );
```

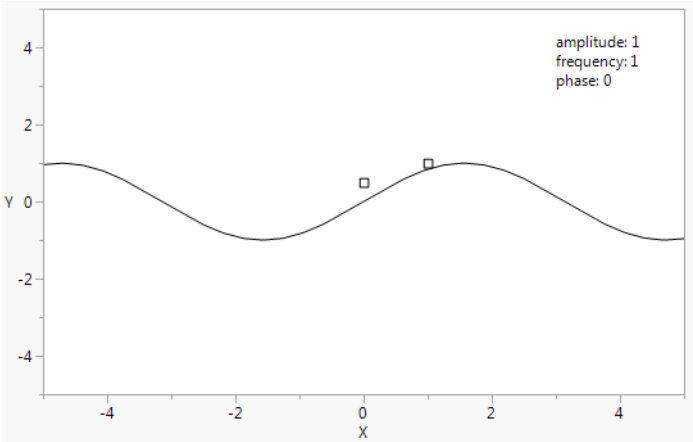
Figure 12.25 Nested Handle() and For()



You can use more than one handle in a graph:

```
amplitude = 1;
freq = 1;
phase = 0;
win = New Window( "Sine Wave",
   Graph Box(
      Frame Size( 500, 300 ),
      X Scale( -5, 5 ),
      Y Scale( -5, 5 ),
      Y Function( amplitude * Sine( x / freq + phase ), x );
      Handle( // first handle
         freq,
         amplitude,
         freq = x;
         amplitude = y;
      );
      Handle( phase, .5, phase = x ); // second handle
      Text(
         {3, 4},
         "amplitude: ",
         Round( amplitude, 4 ),
         {3, 3.5},
         "frequency: ",
         Round( freq, 4 ),
         {3, 3},
         "phase: ",
         Round( phase, 4 )
      );
   )
);
```

Figure 12.26 Two Handles



Mousetrap()

`Mousetrap()` takes arguments for a graph from the coordinates of a mouse click. The first script is executed after each mousedown and the second script after each mouseup to update the graph dynamically, according to the new coordinates of the handle. As with `Handle`, it is important to set the initial values for the `Mousetrap's` coordinates. If you include both `Mousetrap()` and `Handle()` in a graph, put the `Handle()` before the `Mousetrap()`. Then `Handle()` has a chance to catch clicks before a `Mousetrap()` does.

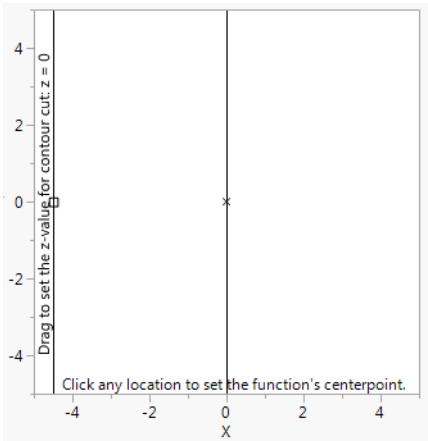
This example uses both `Mousetrap()` and `Handle()` to draw a three-dimensional function centered on the `Mousetrap()` coordinates, where the single contour line takes its value from a `Handle()`.

```
x0 = 0;
y0 = 0;
z0 = 0;
win = New Window( "Viewing a 3-D Function in Flatland",
   Graph Box(
      Frame Size( 300, 300 ),
      X Scale( -5, 5 ),
      Y Scale( -5, 5 ),
      Contour Function(
         Exp( -(x - x0) ^ 2 ) * Exp( -(y - y0) ^ 2 ) * (x - x0),
         x,
         y,
         z0 / 10
      );
```

```
        // get the z-cut values from a handle
        Handle( -4.5, z0, z0 = Round( y * 10 ) / 10 );
              V Line( -4.5 );
        Text Size( 9 );
        Text(
           Counterclockwise,
           {-4.6, -4},
           "Drag to set the z-value for contour cut: z = " || Char( z0 / 10 )
        );
        Marker Size( 2 );
        Marker( 2, {x0, y0} );
        Mousetrap( // set the origin to the click-point
           x0 = x;
           y0 = y;
        );
        Text(
           {-4.25, -4.9},
           "Click any location to set the function's centerpoint."
        );
     )
  );
```

Note: Instead of setting `x0` to `x` and `y0` to `y` in the `Mousetrap()` expression, you might want to use `Function( {xx, yy}, Show( xx, yy ); x0 = xx; y0 = yy );`. `xx` and `yy` are formal parameters to the function. They come from arguments that the mousetrap passed when the user clicked on the graph. The `Function( {xx, yy}...)` alternative makes the `xx` and `yy` names explicit, and makes it clear the you can name them anything, not just `x`, `y`.

Figure 12.27 Mousetrap() and Handle()



You might use Mousetrap() to collect points in a data table, such as for visually interpolating points in a graph. The following example illustrates a script that could be adapted and added to a data plot (such as a scatterplot from Fit Y by X) for that purpose. After you run the script and click in the graph, the point at which you click is recorded in the data table.

```
dt = New Table( "dat1" );
New Column( "xx", Numeric );
New Column( "yy", Numeric );
x = 0;
y = 0;
Add Point = Expr(
   dt << Add Rows( 1 );
   Row() = N Row();
   :xx = x;
   :yy = y;
);
win = New Window( "Add Points",
   Graph Box(
      Frame Size( 500, 300 ),
      X Scale( -5, 5 ),
      Y Scale( -5, 5 ),
      For Each Row( Marker( {xx, yy} ) );
      Mousetrap( {}, Add Point );
   )
);
```

Notice that the first script argument is empty. At mousedown, nothing happens. The second script, add point, is executed at mouseup to add a data point. If you click, drag, and release, then the point that is added to your data set is the point where you let go of the mouse button; it is not the point where you pressed down the mouse button.

Drag Functions

There are five Drag() functions to perform similar functions to Handle() and Mousetrap() but with more than one point at a time. For *n* coordinates in matrices listed as the first two arguments:

- Drag Marker() draws n markers.
- Drag Line() draws a connected line with n vertices and $n-1$ segments.
- Drag Rect() draw a filled rectangle using the first two coordinates, ignoring any further coordinates.
- Drag Polygon() draws a filled polygon with n vertices.

- `Drag Text()` draws a text item at the coordinates. Or if there is a list of text items, it draws the *i*th list item at the *i*th (*x*,*y*) coordinate. If there are fewer list items than coordinate pairs, the last item is repeated for remaining points.

The syntax for these commands:

```
Drag Marker( xMatrix, yMatrix, dragScript, mouseupScript );
Drag Line( xMatrix, yMatrix, dragScript, mouseupScript );
Drag Rect( xMatrix, yMatrix, dragScript, mouseupScript );
Drag Polygon( xMatrix, yMatrix, dragScript, mouseupScript );
Drag Text( xMatrix, yMatrix, "text", dragScript, mouseupScript );
```

They all must have L-value arguments for the coordinates. The coordinates are literal matrices or names of matrix values that are modified if you click a vertex and drag it to a new position. The script arguments are optional and behave the same as with `Handle()`. However, there is no *x* nor *y* that is modified as in `Handle()`.

`Drag()` functions are ways to display data that the user can adjust and then capture the adjusted values. Consider the earlier script to draw the JMP man. `Drag Polygon()` makes it possible to draw an editable JMP man; using a matching `Drag Marker()` statement makes the vertices more visible. And, similar to the `Mousetrap()` example, you can save the new coordinates to a data table. Notice how `:` and `::` operators avoid ambiguity among matrices and data table columns with the same names.

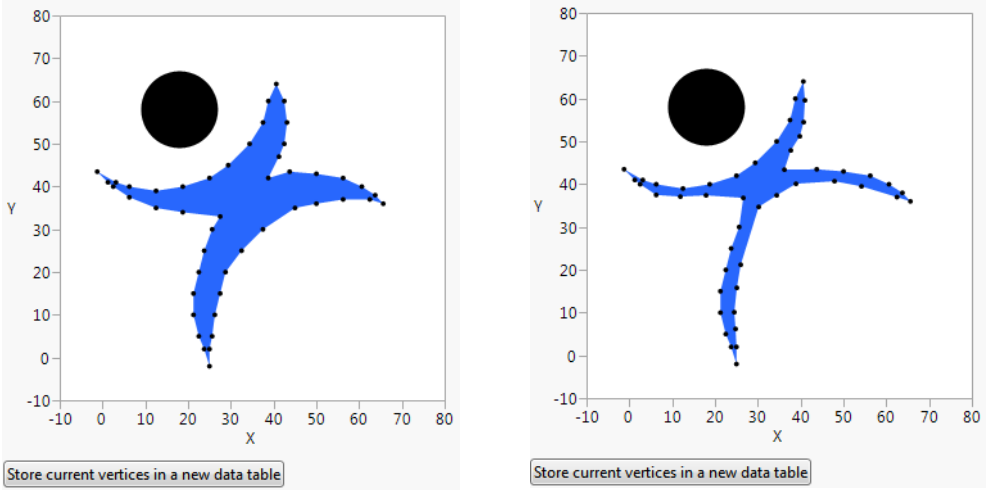
You could just as easily put *`storepoints`* in the fourth argument of `Drag Polygon()` or `Drag Marker()`, but that would create a data table after each drag. You probably just want a single data table when you are finished. Regardless, the values in *`gCoordX`* and *`gCoordY`* update with each drag.

```
::i = 1;
storepoints = Expr(
  mydt = New Table( "My coordinates" || Char( i ) );
  i++;
  New Column( "GCoordX", Numeric );
  New Column( "GCoordY", Numeric );
  mydt << Add Rows( N Row( GcoordX ) );
  :GCoordX << Values( ::GcoordX );
  :GCoordY << Values( ::GcoordY );
);

::GcoordX = [25, 23.75, 22.5, 21.25, 21.25, 22.5, 23.75, 25.625, 27.5, 18.75,
  12.5, 6.25, 2.5, 1.25, -1.25, 3.125, 6.25, 12.5, 18.75, 25, 29.375, 34.375,
  37.5, 38.75, 40.625, 42.5, 43.125, 42.5, 41.25, 38.75, 43.75, 50, 56.25,
  60.625, 63.75, 65.625, 62.5, 56.25, 50, 45, 37.5, 32.5, 28.75, 27.5, 26.25,
  25.625, 25];
::GcoordY = [-2, 2, 5, 10, 15, 20, 25, 30, 33, 34, 35, 37.5, 40, 41, 43.5, 41,
  40, 39, 40, 42, 45, 50, 55, 60, 64, 60, 55, 50, 47, 42, 43.5, 43, 42, 40,
  38, 36, 37, 37, 36, 35, 30, 25, 20, 15, 10, 5, 2];
```

```
win = New Window( "Redraw the JMP Man!",
   V List Box(
      Graph Box(
         Frame Size( 300, 300 ),
         X Scale( -10, 80 ),
         Y Scale( -10, 80 ),
         Fill Color( "blue" );
         Drag Polygon( GcoordX, GCoordY );
         Pen Color( "gray" );
         Drag Marker( GcoordX, GCoordY );
         Fill Color( {0, 0, 0} );
         Circle( {18, 58}, 9, "FILL" );
      ),
      Button Box( "Store current vertices in a new data table", storepoints )
   )
);
```

Suppose you want the body of the JMP man to be smaller. Here is how he looks before and after some judicious vertex-dragging. Clicking the button after re-shaping the JMP Man executes the *storepoints* script to save his new, slender figure in a data table of coordinates.

Figure 12.28 Redraw the JMP Man



This example uses two functions discussed under "Constructors for New Windows":

- `Button Box()`, which creates controls outside the graph.
- `V List Box()`, which combined the graph box and the button box in the same graph window.

Troubleshoot Interactive Graphs

If your interactive graphs do not work as expected, make sure that you supply initial values for the `Handle()` or `Mousetrap()` coordinates (and other globals as needed). Also make sure that the values make sense for the graph.

Create a Color Picker

Let users select a color to apply to graphs by creating a color picker. Users can select a predefined color or create their own color. You can also specify a default color in your script.

- Use the named or numbered colors shown in "Colors and Markers".

  or

- Specify values between 0 and 1 for each of the red, blue, and green components in `RGB Color()`.

Examples of Creating Color Pickers

The following example does not name the window, so the default title is used. Because the script does not specify a color, Black is selected by default. If the user selects a color in the color picker window, then the lines are drawn in the selected color.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Run Script( "Bivariate" );
Wait( 0 );
pickColor = Pick Color();
Report( biv )[AxisBox( 1 )] <<
Add Ref Line( 105, "Dashed", pickColor, "Mean" );
```

The following example creates a color picker with the window title "Select a Line Color". BlueGreen is preselected.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Run Script( "Bivariate" );
Wait( 0 );
pickColor = Pick Color( "Select a Line Color", "BlueGreen" );
Report( biv )[AxisBox( 1 )] <<
Add Ref Line( 105, "Dashed", pickColor, "Mean" );
```

The following example creates a color picker with the window title "Select a Background Color". BlueGreen is preselected.

```
picked color = Pick Color( "Select a Background Color", "BlueGreen" );
New Window( "Example",
        gb = Graph Box(
```

```
                Frame Size( 300, 300 ),
                Marker( Marker State( 3 ), [11 44 77], [75 25 50] );
                Line( [10 30 70], [88 22 44] );
        )
);
gb[FrameBox(1)] << Background Color ( picked color );
```

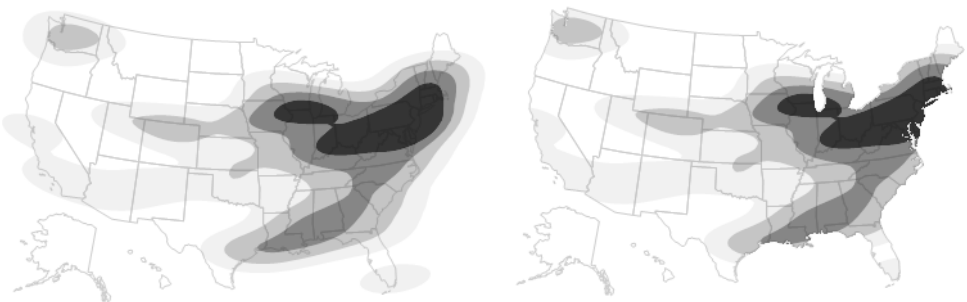
The following example specifies the RGB values for red, green, and blue to create pink. If you do not know the JMP color name or number, you might consider this option.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Run Script( "Bivariate" );
Wait( 0 );
pickColor = Pick Color( "RGB", RGBCOLOR( 1/*red*/, 0/*green*/, 1/*blue*/ ) );
Report( biv )[AxisBox( 1 )] <<
Add Ref Line( 105, "Dashed", pickColor, "Mean" );
```

Clip Display Segs in Graphics

Display segs (such as reference lines, grid lines, and contours) can be clipped to conform to the natural boundaries of the underlying data. This gives you a better view of the boundaries. For example, Figure 12.29 shows contours that are clipped to the boundaries of a map of the United States.

Figure 12.29 Unclipped (Left) and Clipped (Right) Boundaries



Send the `Clip Shape` message to the Contour Seg object to clip the display segs.

```
Open( "$SAMPLE_DATA/Cities.jmp" );
gb = Graph Builder(
   Size( 653, 396 ),
   Show Control Panel( 0 ),
   Variables( X( :Longitude ), Y( :Latitude ) ),
   Elements( Contour( X, Y, Legend( 2 ) ) ),
   SendToReport(
```

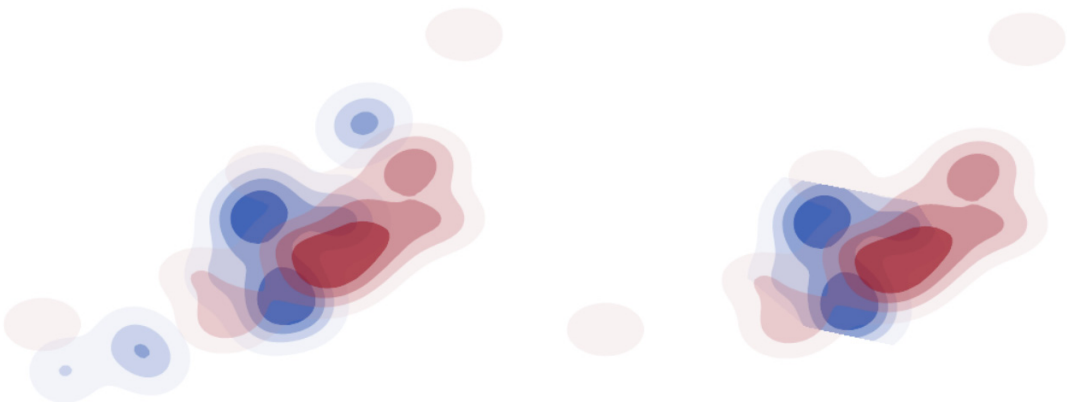
```
      Dispatch(
         {},
         "Graph Builder",
         FrameBox,
         {Background Map( Boundaries( "US States" ) ), Grid Line Order( 2 ),
   Reference Line Order( 3 )}
         )
      )
);
cs = (gb << Report)[FrameBox( 1 )] << Find Seg( Contour Seg( 1 ) );

Wait( 2 );

cs << Clip Shape( Boundaries( "US States" ) );
```

Tip: `Clip Shape(Empty())` removes the clipping path.

Here's an example of defining the clip shape path with a matrix:

```
Open( "$SAMPLE_DATA/Big Class.jmp" );
gb = Graph Builder(
   Show Control Panel( 0 ),
   Variables( X( :height ), Y( :weight ), Overlay( :sex ) ),
   Elements( Contour( X, Y, Legend( 3 ) ) )
);
r = ( gb << Report );
cs = r[FrameBox( 1 )] << Find Seg( Contour Seg( 1 ) );
cs << Clip Shape( Path( [60 80 1, 50 140 2, 65 120 2, 70 65 -2] ) );
```

Figure 12.30 Unclipped (Left) and Clipped (Right) Contours



The matrix [60 80 1, 50 140 2, 65 120 2, 70 65 -2] indicates the following:

1. Start at 60, 80.

2. Draw a line to 50, 140.
3. Draw a line to 65, 120.
4. Draw a line to 70, 65.
5. Connect back to the first point.

Create Background Maps

Background maps can be scripted in JSL. You can write a script that creates a graph and then turns on the background map in the script.

There are two types of background maps: `Images()` and `Boundaries()`. Each of these takes a parameter, which is the name of the map to use. The name is one of the maps listed in the window.

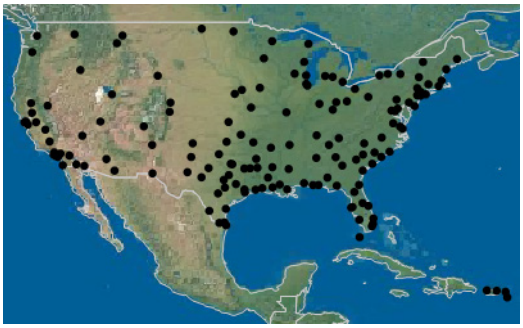
- For `Images()`, the choices are Simple Earth, Detailed Earth, NASA, Street Map Service, and Web Map Service. If you use Web Map Service, then there are two additional parameters: the WMS URL and the layer supported by the WMS server.
- For `Boundaries()`, the choices vary since boundaries can be user-defined. A typical choice is World.

The following example uses the Simple Earth as an image and World as a boundary:

```
dt = Open( "$SAMPLE_DATA/Air Traffic.jmp" );
dt << Graph Builder(
   Size( 1101, 603 ),
   Show Control Panel( 0 ),
   Variables( X( :Longitude ), Y( :Latitude ) ),
   Elements( Points( X, Y, Legend( 8 ) ) ),
   SendToReport(
      Dispatch(
         {},
         "Graph Builder",
         Frame Box,
         {Background Map( Images( "Simple Earth" ), Boundaries( "World" ) ),
         Grid Line Order( 3 ), Reference Line Order( 4 )}
      )
   )
);
```

Figure 12.31 shows an excerpt from the map:

Figure 12.31 JSL Scripting Example



To change the script to use a WMS server, the command would look like this:

```
Background Map ( Images ( "Web Map Service",
  "http://sedac.ciesin.columbia.edu/geoserver/wms",
  "gpw-v3:gpw-v3-population-density_2000" ), Boundaries ( "US States" ) )
```

To see which layers are available on a WMS server, install the WMS Explorer Add-In. Download the add-in from the JMP File Exchange at https://community.jmp.com/docs/DOC-6095. Note that some WMS servers are not reliable. If the server is down, or if the user does not have an Internet connection, the WMS map does not appear.

Tips:

- To see the syntax for the JSL script, add a background map through the user interface. Then from the red triangle menu, select **Save Script > To Script Window** to see the script that is generated.
- To specify image and boundary names in the JSL script, use the names shown in the Set Background Map window. (Right-click a map and select **Graph > Background Map**).
- Look at sample data to find examples of background map scripts. Select **Help > Sample Data Library** and open Napoleons March.jmp, Pollutants Map.jmp, or San Francisco Crime.jmp. Right-click the map table script and select **Edit** to see the `Background Map()` function.

Graph Elements

You can use the following commands inside `Graph Box()` statements. This chapter focuses on the JSL that is specific to graphing, but you can also use general script commands such as `For` and `While`.

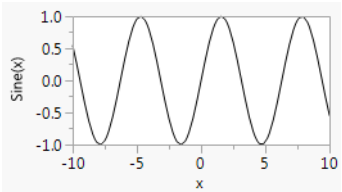
Note: `New Window()` expects window and display box commands, and platforms expect platform commands. Therefore, the general script commands such as `For` and `While` might not work inside platforms or `New Window()`, or you might get unexpected results.

Plot Functions

A `Y Function()` function is used to draw smooth functions. The first argument is the expression to be plotted. The second argument is the name of the X variable in the expression.

```
win = New Window( "Sine Function",
   Graph Box(
      Frame Size( 200, 100 ),
      X Scale( -10, 10 ),
      Y Scale( -1, 1 ),
      X Name( "x" ),
      Y Name( "Sine(x)" ),
      Y Function( Sine( x ), x ) ) );
```

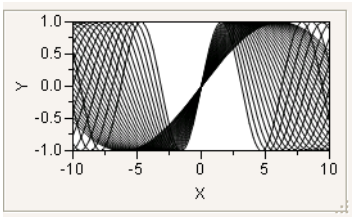
Figure 12.32 Sine Wave



You can use `For` to overlap several sine waves:

```
win = New Window( "Overlapping Sine Waves",
   Graph Box(
      Frame Size( 200, 100 ),
      X Scale( -10, 10 ),
      Y Scale( -1, 1 ),
      For( i = 1, i <= 4, i += .1,
         Y Function( Sine( x / i ), x )
      )
   )
);
```

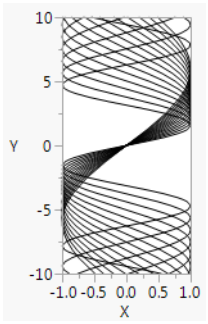
Figure 12.33 Overlapping Sine Waves



Similarly, `X Function()` is for drawing a graph where the symbol is varied on the Y variable.

```
win = New Window( "Overlapping Sine Waves",
   Graph Box(
      Frame Size( 100, 200 ),
      X Scale( -1, 1 ),
      Y Scale( -10, 10 ),
      For( i = 1, i <= 4, i += .2,
         X Function( Sine( y / i ), y )
      )
   )
);
```

Figure 12.34 Overlapping Sine Waves along the X-Axis



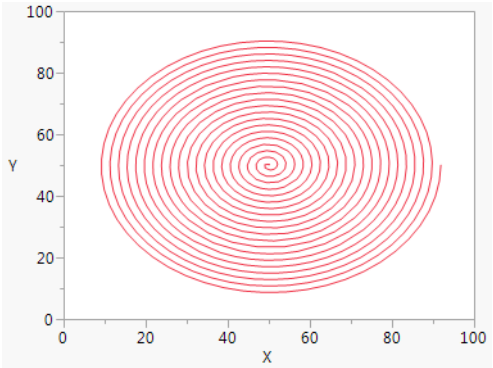
An `XY Function()` draws a smooth curve using a pair of formulas (parametric equations) that depend on a third variable. The third variable's value is incremented from a minimum value to a maximum value to generate the X-Y pairs.

```
win = New Window( "Spiral",
   Graph Box(
      Pen Color( "red" ); // red line color
      xCenter = 50; // location of the center of the curve on the x axis
      yCenter = 50; // location of the center of the curve on the y axis
      minAngle = 0;
      maxAngle = Pi() * 2 * 20;
      XY Function(
```

```
            xCenter + ((ta / 3) * Cos( ta )),
            yCenter + ((ta / 3) * Sin( ta )),
            ta,
            Min( minAngle ),
            Max( maxAngle ),
            Inc( Pi() / 100 )
         );
      )
   );
```

In this example, `Sin()` and `Cos()` use `ta` as an argument (rotates) *and* as a factor (expands). (`Sin()` and `Cos()` use radians, not degrees.)

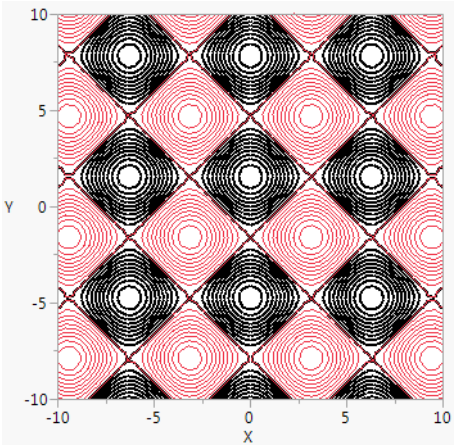
Figure 12.35 Spiral Parametric Plot



`Contour Function()` is an analogous way to represent a three-dimensional function in a two-dimensional space. The final argument specifies the value(s) for the contour line(s). The argument can be a value, an indexed range of values using `::`, or a matrix of values.

```
win = New Window( "Bird's eye view of the egg carton function",
   Graph Box(
      Frame Size( 300, 300 ),
      X Scale( -10, 10 ),
      Y Scale( -10, 10 ),
      Pen Color( "black" );
      Pen Size( 2 );
      Contour Function( Sine( y ) + Cosine( x ), x, y, (0 :: 20) / 5 );
      Pen Color( "red" );
      Pen Size( 1 );
      Contour Function( Sine( y ) + Cosine( x ), x, y, (-20 :: 0) / 5 );
   )
);
```

Figure 12.36 Egg Carton Function



`Normal Contour()` draws normal probability contours for *k* populations and two variables. The first argument is a scalar probability or a matrix of probability values for the contours. Subsequent arguments are matrices to specify means, standard deviations, and correlations. The mean and standard deviation matrices have dimension *k* × 2. The correlation matrix should be *k* × 1, where the first row pertains to the first contour, the second row to the second contour, and so on. The first column is for *x* and the second column for *y*. Consider the following example:

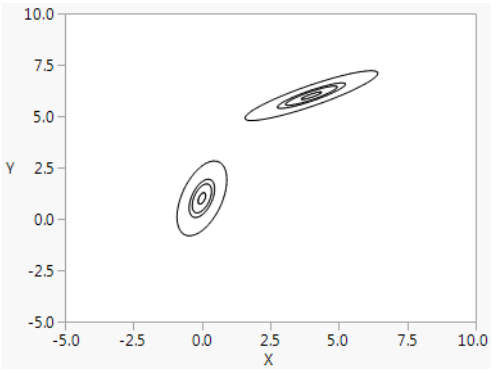
```
Normal Contour(
   [ prob1,
     prob2,
     prob3, ...],
   [ xmean1 ymean1,
     xmean2 ymean2,
     xmean3 ymean3, ...],
   [ xsd1 ysd1,
     xsd2 ysd2,
     xsd3 ysd3, ...],
   [ xycorr1,
     xycorr2,
    xycorr3, ...]);
```

The following script draws contours at probabilities 0.1, 0.5, 0.7, and 0.99 for two populations and two variables. The first population has *x* mean 0 and *y* mean 1, with standard deviation 0.3 along the *x* axis and 0.6 along the *y*-axis, and with correlation 0.5. The second has *x* mean 4 and *y* mean 6, with standard deviation 0.8 along the *x* axis and 0.4 along the *y*-axis, and with correlation 0.9.

```
win = New Window( "Normal Contours",
   Graph Box(
```

```
        X Scale( -5, 10 ),
        Y Scale( -5, 10 ),
        Normal Contour( [.1, .5, .7, .99], [0 1, 4 6], [.3 .6, .8 .4], [.5, .9] )
    )
);
```

Figure 12.37 Normal Contour Function



`Normal Contour()` is a general way to accomplish effects such as Bivariate's density ellipses. The Bivariate script in the Football.jmp sample data creates an example.

Gradient Function

`Gradient Function()` fills a set of rectangles on a grid according to a color determined by the expression value as it crosses a range corresponding to a range of colors.

`Gradient Function( `*`expression`*`, `*`xname`*`, `*`yname`*`, [`*`zlow`*`, `*`zhigh`*`], ZColor( [`*`colorLow`*`, `*`colorHigh`*`] ), <XGrid( `*`min`*`, `*`max`*`, `*`incr`*` )>, <YGrid( `*`min`*`, `*`max`*`, `*`incr`*` )>, <Transparency( t )>;`

Here are explanations of the elements in the `Gradient Function()` syntax:

```
GradientFunction(
    /* the expression to be contoured, which is a function in terms of
    the two variables that follow expression*/
    Expression,

    // the two variable names used in the expression
    xname, yname,

    // the low and high expression values the gradient is scaled between
    [zlow, zhigh],

    // the colors that correspond to the low value and the high value
    ZColor([colorLow, colorHigh])
```

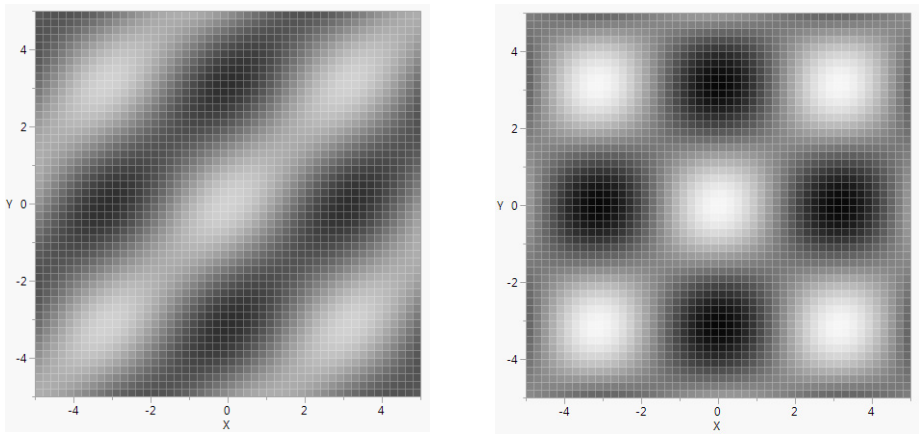
```
// optional specification for the grid of values
<XGrid(min, max, incr),>
<YGrid(min, max, incr)> );

/* the value 0 means clear. The value 1 means completely opaque and
is the usual drawing mode. */
<Transparency>
```

The `ZColor()` values must be numeric codes rather than names. You can use the color menu indices (such as 0=black, 1=grey, 2=white, 3=red, 4=green, and 5=blue) found in "Specify Colors".

The following example uses `Gradient Function()` to create an animated graph.

```
phase = 0.7;
win = New Window( "Gradient Function",
   a = Graph(
      Frame Size( 400, 400 ),
      X Scale( -5, 5 ),
      Y Scale( -5, 5 ),
      Gradient Function(
         phase * Sine( x ) * Sine( y ) + (1 - phase) * Cosine( x ) * Cosine( y
   ),
         x,
         y,
         [-1 1],
         zcolor( [0, 2] )
      )
   )
);
b = a[FrameBox( 1 )];
For( i = 1, i <= 5, i++,
   For( phase = 0, phase < 1, phase += 0.05,
      b << Reshow;
      Wait( 0.01 );
   );
   For( phase = 1, phase > 0, phase -= 0.05,
      b << Reshow;
      Wait( 0.01 );
   );
);
```

Figure 12.38 Gradient Function



Get the Properties of a Graphics Frame

Several functions are useful for getting properties of an existing graphics frame:

H Size Returns the horizontal size of the graphics frame in pixels.

V Size Returns the vertical size of the graphics frame in pixels.

X Origin Returns the *x*-value for the left edge of the graphics frame.

X Range Returns the distance from the left to right edges of a display box.

Y Origin Returns the *y*-value for the bottom edge of the graphics frame.

Y Range Returns the distance from the bottom to top edges of a display box.

In the following expressions, the first line calculates the right edge. The second line calculates the top edge.

```
Oval(
   ...,
   rightEdge = X Origin() + X Range();
   topEdge = Y Origin() + Y Range();
);
```

Add a Legend

To add a legend to a graph, send the `Row Legend` message to the frame box. In the `Row Legend` message, specify which column you want to base the legend on and whether the legend affects colors and markers.

For example, using Big Class.jmp, submit the following JSL to turn on a legend based on the age column. Set both colors and markers by values in the age column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( y( :height ), x( :weight ) );
rbiv = biv << Report;
framebox = rbiv[Frame Box( 1 )];

/* make a legend with the "age" title
display colors and markers in the legend */
framebox << Row Legend( "age", Color( 1 ), Marker( 1 ) );
```

The `Color()` and `Marker()` arguments are optional. Colors are on and markers are off by default.

To use a continuous scale if your variable is nominal or ordinal, you can use a continuous color theme with the `Continuous Scale( 1 )`.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( y( :weight ), x( :height ) );
rbiv = biv << Report;
framebox = rbiv[Frame Box( 1 )];
framebox << Row Legend( "age", Color( 1 ), Continuous Scale( 1 ) );
```

To interactively add a legend, right-click the graph, select **Row Legend**, and change the column settings.

Hover Labels

When you hover over a visual element in a graph, usually you see a basic description of the data point on a hover label.

You can customize what appears on the hover label using the following methods:

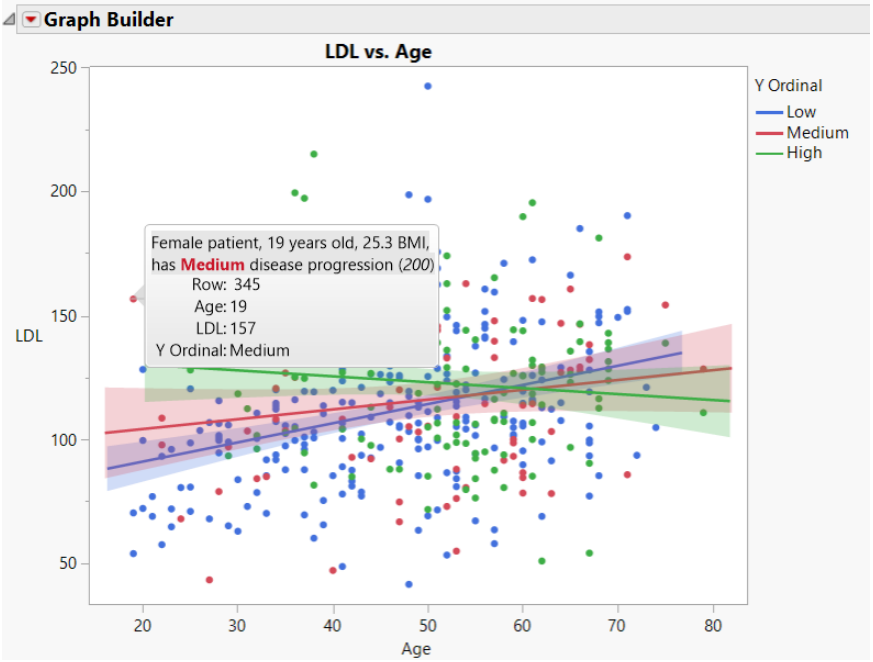
- "Add Rich Text to Hover Labels Using Textlets"
- "Add Graphs or Images to Hover Labels Using Graphlets"
- "Update Names and Values in Hover Labels Using Gridlets"
- "Create Drill-Down Graphs"
- "Work with the Hover Label Execution Context"

Add Rich Text to Hover Labels Using Textlets

A *textlet* adds richly formatted text to the hover label. You might use a textlet to describe data in an explanatory paragraph or to emphasize important information by making it bold or applying a color.

The textlet in Figure 12.39 shows text that is formatted in a gray box with red text.

Figure 12.39 Textlet Example



The following script creates the textlet shown in Figure 12.39.

```
Names Default To Here( 1 );
dt = Open( "$SAMPLE_DATA/Diabetes.jmp" );

// create the baseline visualization
gb = dt << Graph Builder(
   Size( 534, 456 ),
   Show Control Panel( 0 ),
   Variables( X( :Age ), Y( :LDL ), Overlay( :Y Ordinal ) ),
   Elements( Points( X, Y, Legend( 8 ) ), Line Of Fit( X, Y, Legend( 10 ) ) )
);

// get a handle to the graph frame box
rpt = gb << Report;
```

```
frame = rpt[Framebox( 1 )];

/* configure the textlet -- a hover label extension that adds
   a rich text paragraph to the hover label. */
frame << Set Textlet(
    /* "Setup" is a JSL fragment that uses Hover Label Execution Contect
     variables to define content that will be embedded in the
     rich text paragraph.*/
   Setup(
      // map numerical values to text
      local:gender = If( Column( local:_dataTable, "Gender" )[local:_firstRow]
         == "1",
         "Male",
         "Female"
      );
      // map a categorical value to text color
      local:color = Match( local:_Y Ordinal,
         "Low", "Medium Dark Blue",
         "Medium", "Medium Dark Red",
         "High", "Medium Dark Green"
      );
      // capture values from other columns for the current row
      local:progression = Column( local:_dataTable, "Y" )[local:_firstRow];
      local:BMI = Column( local:_dataTable, "BMI" )[local:_firstRow];
   ),
   /* markup defines the rich text template. It supports HTML-like markup such
   as bold, italic and other font tags. Variable references between braces
   ({}) are substituted with their actual value. */
   Markup(
      "{local:gender} patient, {local:_age} years old, {local:BMI} BMI,
      has <font color='{local:color}'><b>{local:_Y Ordinal}</b></font>
      disease progression (<i>{local:progression}</i>)"
   )
);
// pin a hover label to display a textlet-created rich text paragraph.
frame << Add Pin Annotation(
   Seg( Marker Seg( 2 ) ),
   Index( 64 ),
   Index Row( 344 ),
   UniqueID( 568154464 ),
   FoundPt( {96, 222} ),
   Origin( {18.68125, 156.93706122449} ),
   Tag Line( 1 )
);
```

Texlet Commands

See the example in "Add Rich Text to Hover Labels Using Textlets" for more information about writing texlet expressions.

Setup

`Setup()` defines variables used in the rich text that is displayed on the hover label.

```
...
frame << Set Textlet(
    /* "Setup" is a JSL fragment that uses Hover Label Execution Contect
     variables to define content that will be embedded in the
     rich text paragraph.*/
  Setup(
     // map numerical values to text
     local:gender = If( Column( local:_dataTable, "Gender" )[local:_firstRow]
        == "1",
        "Male",
        "Female"
     );
     // map a categorical value to text color
     local:color = Match( local:_Y Ordinal,
        "Low", "Medium Dark Blue",
        "Medium", "Medium Dark Red",
        "High", "Medium Dark Green"
     );
     // capture values from other columns for the current row
     local:progression = Column( local:_dataTable, "Y" )[local:_firstRow];
     local:BMI = Column( local:_dataTable, "BMI" )[local:_firstRow];
  )
...
```

In the interface, Setup is called "JSL Variables" on the Textlets Markup tab.

Markup

`Markup()` specifies a rich text paragraph template based on a subset of HTML tags. Can contain JSL variable references enclosed in delimiters. The default delimiters are braces ({}).

```
...
Markup(
     "{local:gender} patient, {local:_age} years old, {local:BMI} BMI,
     has <font color='{local:color}'><b>{local:_Y Ordinal}</b></font>
     disease progression (<i>{local:progression}</i>)"
  )
```

In the interface, Markup is called "HTML Markup" on the Textlets Markup tab.

See *Using JMP* for more information about the supported font formatting tags.

Delimiters

`Delimiters()` specifies the delimiters that indicate variables for replacement in the Markup area. For example, you might use square bracket delimiters for JMP variables if the script contains C++ code.

```
Delimiters( "[]" )
```

In the interface, Delimiters is on the Textlets Other tab.

Width

`Width()` enables you to specify a preferred width in pixels for the hover label.

```
Width( 300 )
```

In the interface, Width is on the Textlets Other tab.

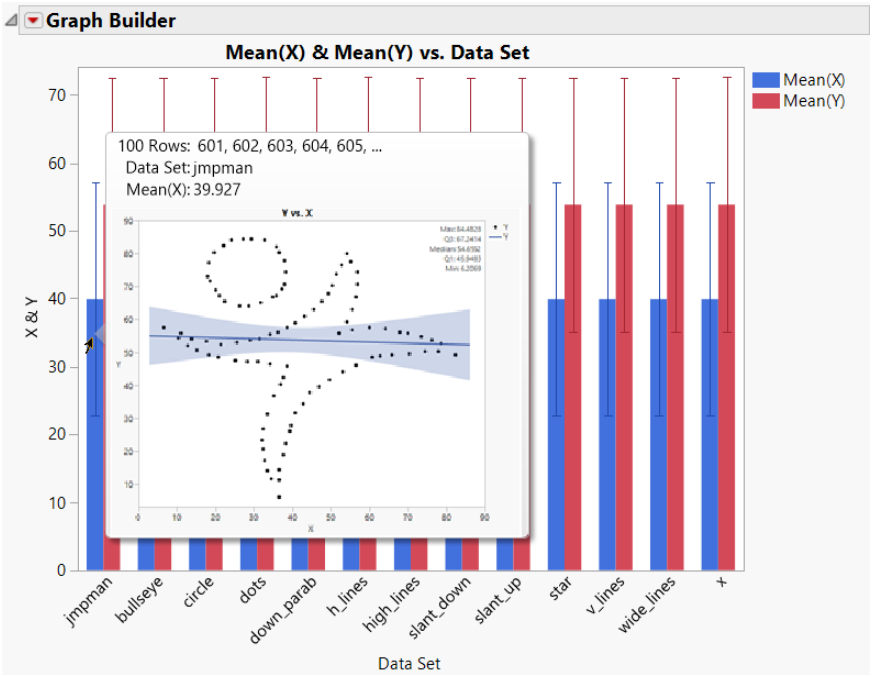
Add Graphs or Images to Hover Labels Using Graphlets

You can add a thumbnail of a related graph or an image to a hover label using a *graphlet*. These graphs and images can have actions associated with them that are triggered by a mouse click. For example, you can click a graph to launch it, or click an image to open a related link.

Example of a Graphlet in Graph Builder

Click the JMP man graph shown in Figure 12.40 to open it in Graph Builder and customize the graph. See *Using JMP* for details about opening a graph in a platform, customizing the graph, and then saving the updated graph in the original graph.

Figure 12.40 Graphlet Example



The following script creates the graph and hover label that are shown in Figure 12.40.

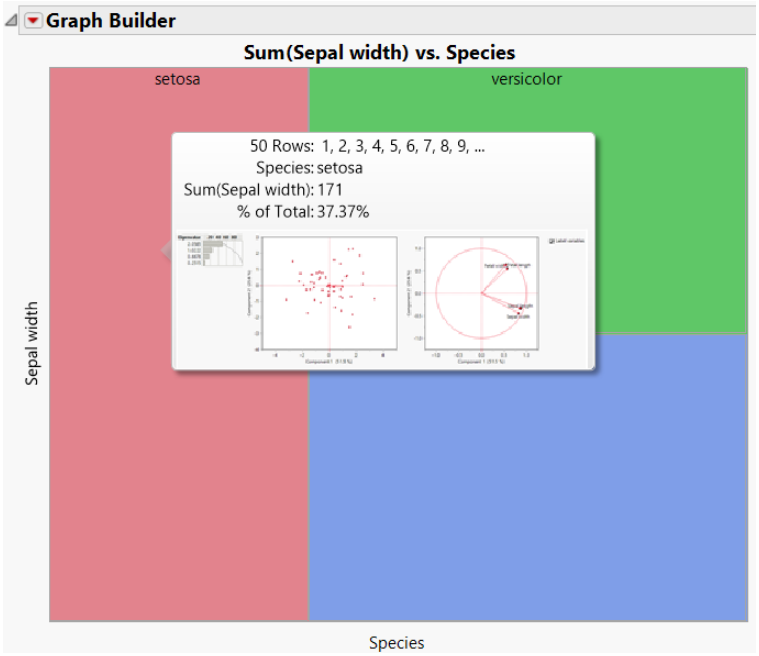
```
Names Default To Here( 1 );
// open the data table
dt = Open( "$SAMPLE_DATA/JMP Man Dozen.jmp" );

// launch the baseline visualization
gb = Graph Builder(
    Size( 529, 466 ),
    Show Control Panel( 0 ),
    Variables( X( :Data Set ), Y( :X ), Y( :Y, Position( 1 ) ) ),
    Elements(
        Bar(
            X,
            Y( 1 ),
            Y( 2 ),
            Legend( 12 ),
            Error Interval( "Standard Deviation" )
        )
    )
);

// get a reference to the visualization frame
```

```
frame = (gb << Report)[FrameBox(1)];

/* use the "Set Graphlet" command to define the visualization
that will be rendered inside the hover label using the
"Picture" attribute */
frame << Set Graphlet(
    /* The value of the "Picture" attribute is a JSL fragment
     that is evaluated to generate a thumbnail. The thumbnail is
     displayed inside the hover label. In this case, the hover
     label shows a scatterplot with a caption box. */
  Picture(
       Graph Builder(
           Show Control Panel( 0 ),
           Variables( X( :X ), Y( :Y ) ),
           Elements(
               Points( X, Y, Legend( 1 ) ),
               Line Of Fit( X, Y, Legend( 3 ) ),
               Caption Box( X, Y, Legend( 4 ),
                Summary Statistic( "5 Number Summary" ) )
           )
       )
    /* Because the optional "Click" attribute is not defined, the
        "Picture" attribute script is re-used to launch the associated
        platform when the user clicks the graphlet thumbnail. */
  )
);
```

Example of a Graphlet in Principal Components

Create a graphlet in the Principal Components platform. When the user clicks the hover label, the Principal Components report window appears. The user can then explore and filter the data.

Figure 12.41 Prinicipal Components Graphlet



The following script creates the graphlet shown in Figure 12.41:

```
dt = Open( "$SAMPLE_DATA/Iris.jmp" );
gb = dt << Graph Builder(
   Size( 531, 451 ),
   Show Control Panel( 0 ),
   Variables( X( :Species ), Y( :Sepal width ) ),
   Elements( Treemap( X, Y, Legend( 4 ) ) ),
   SendToReport(
      Dispatch(
         {},
         "Graph Builder",
         FrameBox,
         {Set Graphlet(
            Picture(
               Principal Components(
                  Y( :Sepal length, :Sepal width, :Petal length, :Petal width
   ),
                  Z( :Species ),
                  Estimation Method( "Default" ),
                  "on Correlations"
               )
            )
         )}
```

```
        )
    )
);
```

Graphlet Commands

Picture

A required script that adds a thumbnail of a graph or an image to the hover label.

`Picture()` is always executed in a Hover Label Execution Context derived from the current visual element. The first `Picture Box` (outside of a local data filter) found will be used to generate the displayed image.

If `Picture()` is a call to a platform, a local data filter is included.

```
...
Picture(
        Graph Builder(
            Show Control Panel( 0 ),
            Variables( X( :X ), Y( :Y ) ),
            Elements(
                Points( X, Y, Legend( 1 ) ),
                Line Of Fit( X, Y, Legend( 3 ) ),
                Caption Box( X, Y, Legend( 4 ),
                 Summary Statistic( "5 Number Summary" ) )
            )
        )...
```

See "Add Graphs or Images to Hover Labels Using Graphlets" for a complete example.

Click

An optional script for user-defined actions to be executed when the user clicks the associated image.

- Clicking runs the main graph's platform (for example, Graph Builder), which launches in its own window.
- If Click is not provided, the script provided in `Picture` is evaluated in a Hover Label Execution Context scope.

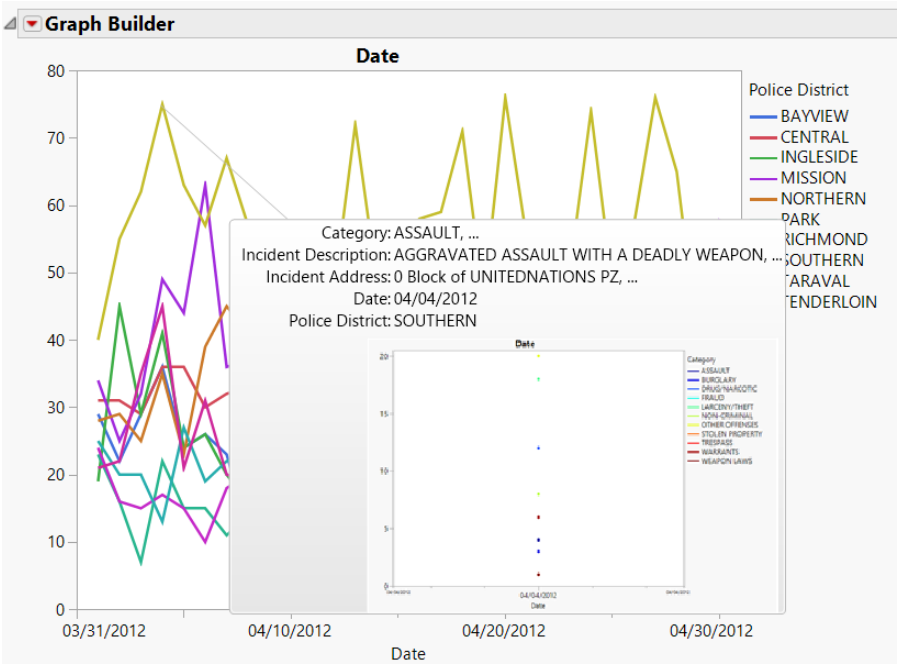
Skip Filters

An optional script. By default, any code executed in the context of a hover label customization (such as a graphlet) executes in a context defined in the hovered visual element (for example, a bar in a bar chart). This context includes a local data filter based on the current visual role assignment, and it is usually what you want. However, in some rare cases, you don't want the filter, or maybe part of the filter, so that the next graph generated by the graphlet can include more of the original data.

- If the Skip Filters field evaluates to a non-0 value (usually a positive integer such as 1) the selected Skip Filters Columns will be removed from the filter that will be applied to the new graphlet.
- Otherwise (for example, if it evaluates to 0), the selected columns are ignored and the filter is created as usual.
- If the Skip Filters Flag field is empty and no columns are selected in Skip Filters Columns (the default), then no columns are skipped when creating filters for the new window. Note that Skip Filters Flag is a JSL expression, which can be as complex as needed.
- If Skip Filters Flag field is not empty, but Skip Filter Columns is empty (no columns are selected), then all columns are filtered; the graphlet or new visualization includes all the data used by the original visualization. See "Example of Defining the Context of a Scatterplot Matrix and Ellipse".

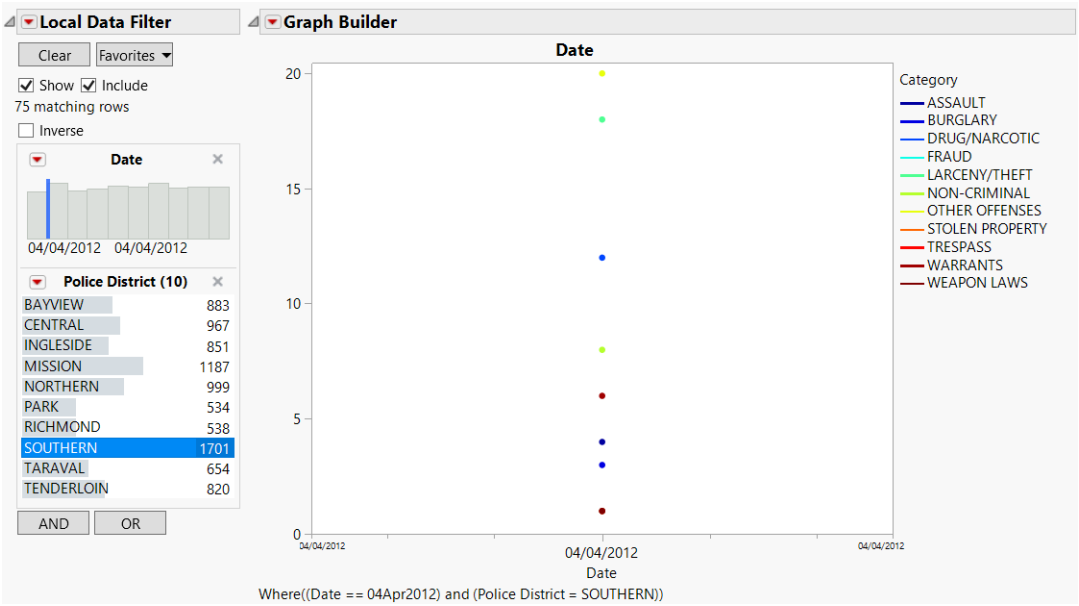
Suppose that you are looking at crime over time in the San Francisco Crime.jmp sample data table. You use a line chart where X = Date, an overlay of Police District, and no Y assignment. Lines represent counts over time per district. Now you want to use a graphlet to look at crime over time inside each district, broken down by category. Create this new graph and select **Save > to Clipboard** and then **Paste Graphlet** on the original graph to create the graphlet. However, the hovered graph shown in Figure 12.42 is not what you expect.

Figure 12.42 Initial Filtered Data



JMP created a filter based on both roles, X and the overlay. This is easier to see if you click on the graphlet to launch the new graph, and then expand the local data filter (on the left, collapsed by default). The filter on a single day causes the line chart to collapse into what looks like a marker plot as shown in Figure 12.43.

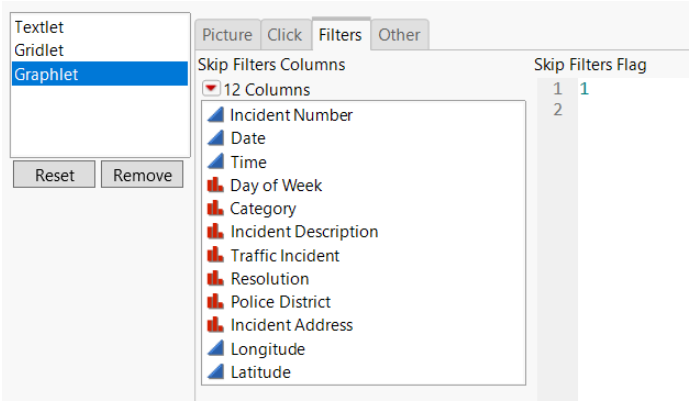
Figure 12.43 Local Data Filter with a Line Chart that Looks like a Marker Plot



You want the overlay value to restrict what you see in the new graph. However, you don't want the date value to do so. You still want to filter by the "SOUTHERN" Police District, but you do not want to filter on "04/14/2012" Date.

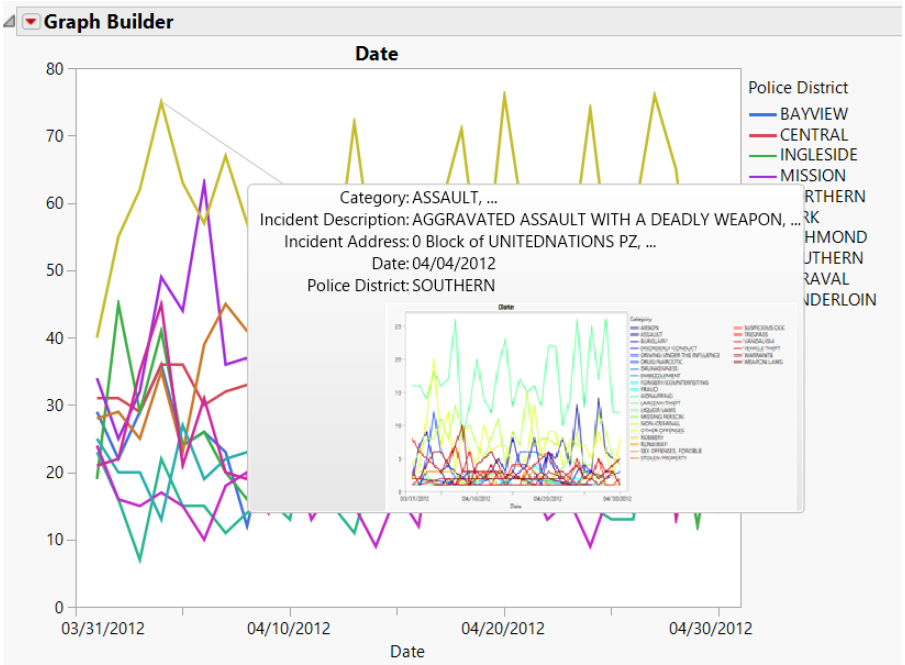
The solution is to use the Skip Filters Flag to remove the filter on Date from your original graph. On the Hover Label Editor, the script looks like this:

Figure 12.44 Example of Skip Filters Flag script in the Hover Label Editor



After the filter is removed, the end result is what you expect as shown in Figure 12.45.

Figure 12.45 Line Chart where Skip Filters Flag = 1



The following script creates the graph shown in Figure 12.45.

```
Names Default To Here( 1 );

dt = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
gb = dt << Graph Builder(
   Size( 528, 456 ),
   Show Control Panel( 0 ),
   Variables( X( :Date ), Overlay( :Police District ) ),
   Elements( Line( X, Legend( 4 ) ) ),
   SendToReport(
      Dispatch(
         {},
         "Graph Builder",
         FrameBox,
         {Set Graphlet(
            Picture(
               Graph Builder(
                  Size( 528, 456 ),
                  Show Control Panel( 0 ),
                  Variables( X( :Date ), Overlay( :Category ) ),
                  Elements( Line( X, Legend( 1 ) ), Smoother( X, Legend( 2 ) ) )
               )
```

```
            ),
            Skip Filters( 1, {:Date} )
          )}
        )
      )
    );
```

The Skip Filters Flag field holds an expression that is evaluated in the hovered visual element context. If expression evaluates to a truthful value (usually a positive integer such as 1), the selected “Skip Filters Columns” will be removed from the filter that will be applied to the new graphlet. Otherwise (for example, if it evaluates to zero), the selected columns are ignored and the filter is created as usual.

Title

Title is an expression that evaluates to a string. The result is the title that appears on the title bar of the window that contains the graph. Variables defined in the Hover Label Evaluation Context can be used in the Title expression.

For example, the following expression indicates the baseline visualization type and how many drilling levels have been used so far:

```
Title("Graphlet for " || local:_displaySegName ||
       ", depth is " || Char( local:_drillDepth ))
```

If the Title expression is not specified, the new window is named “Graphlet”.

Note: In the JMP interface, Title is on the Hover Label Editor Other tab of a graphlet.

Reapply

Reapply is an expression that indicates whether the graphlet definition should be reapplied to new graphs or platforms created by clicking on a graphlet image. This is the core mechanism behind the ability to drill down using graphlets.

```
Reapply( local:_underlyingRows > 1 )
```

Note: In the JMP interface, Reapply is on the Hover Label Editor “Other” tab of a graphlet.

If Reapply is empty (the default setting), the graphlet definition is not reapplied. New windows launched by clicking on this graphlet doesn’t show graphlets on its hover labels.

Update Names and Values in Hover Labels Using Gridlets

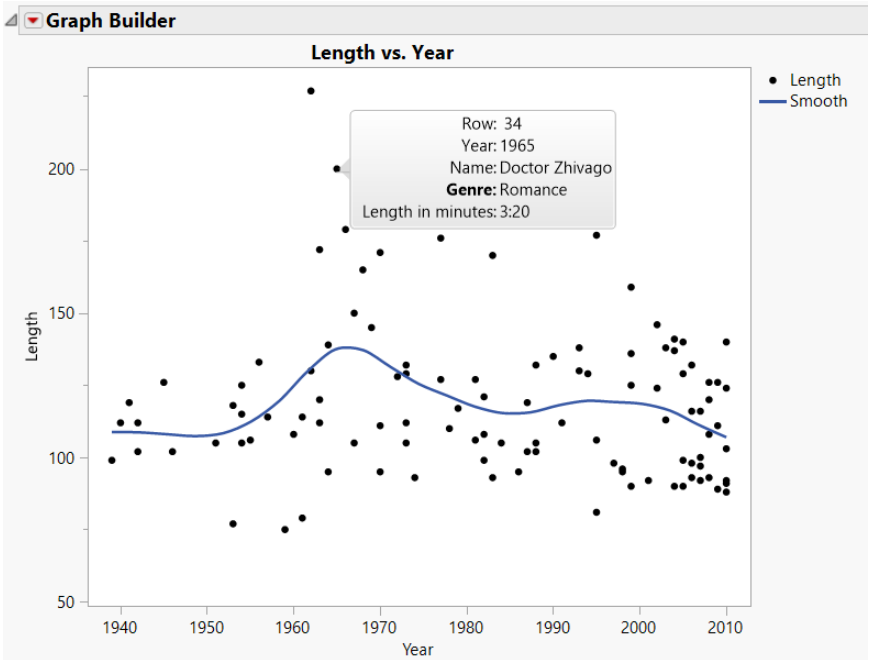
A *gridlet* supports the full customization of the hover label name and value pairs that reflect the underlying visual data. Data can be added, removed, or renamed. You can also change the style and numeric formats.

When you hover over a visualization in a graph, you see content in the hover label for that marker. A grid entry is text that appears on the hover label by default, either from the label columns or from the columns associated with the visualization. By default, the column names that are assigned to roles (and their corresponding values) are shown, along with columns that are labeled. You can customize these column names or values or add other content. Here are a few things you can do:

- add or remove a column name and its value
- rename a grid label
- add a new grid entry with a dynamic value and an optional click action. For example, you can insert an entry named “See also” with a clickable link to a Wikipedia page as the value.
- applying a style to text (for example, formatting text in bold)
- formatting numeric values (for example, setting the number of decimal places to 2)

Figure 12.46 shows adding Genre, Name, and Length in minutes, reformatting the minutes, and formatting “Genre” in bold.

Figure 12.46 Customized Gridlet



Example of Customizing a Gridlet

The following script creates the graph and hover label shown in Figure 12.46.

```
Names Default To Here( 1 );
dt = Open( "$SAMPLE_DATA/Movie Inventory.jmp" );

// launch the baseline visualization
dt << Graph Builder(
   Size( 534, 456 ),
   Show Control Panel( 0 ),
   Variables( X( :Year ), Y( :Length ) ),
   Elements( Points( X, Y, Legend( 3 ) ), Smoother( X, Y, Legend( 4 ) ) ),
   SendToReport(
      Dispatch(
         {},
         "Graph Builder",
         FrameBox,
         {Set Gridlet(
            // delete the Length column
            Expunge( {{Matcher( "Length" )}} ),
            // add the Name, Genre, and Length in minutes columns
            Annex(
               {{Matcher( "Name" ), value( :Name[local:_firstRow] )},
               {Matcher( "Genre" ), value( :Genre[local:_firstRow] )},
               {Matcher( "Length in minutes" ), value( local:_Length * 60 )}}
            ),
            Reformat(
            // reformat the Length in minutes format
               {{Matcher( "Length in minutes" ), Format( "hr:m", 80 ), 80}}
            ),
            Style(
            // format "Genre" in bold nine-point text
               {{Matcher( "Genre", Scope( Name ) ), Font("Segoe UI", 9,
                  "Bold")
               }}
            )
         ), Add Pin Annotation(
            Seg( Marker Seg( 1 ) ),
            Index( 33 ),
            Index Row( 33 ),
            UniqueID( 610643713 ),
            FoundPt( {508, 199} ),
            Origin( {1964.8951552795, 199.00456244898} ),
            Tag Line( 1 )
         )}
      )
```

```
    )
);
```

Gridlet Commands

The first argument in all `Set Gridlet` commands is a matcher, which identifies its scope of action (the grid entries it affects). A matcher can refer to the following:

- A single grid entry: by name, matching against a string. A grid entry is text that appears on the hover label by default, either from the label columns or from the columns associated with the visualization.
- Multiple grid entries: by column reference, regular expression or using the argument `All`.

For example, a grid entry for Mean(height) can be matched against the following:

- the name `"Mean(height)"n`
- the column reference `:height`
- the regular expression `PatReg("^.*\(height\)$")`
- the special argument `All`, which always matches all rows in the grid.

If there is a match, the command-associated action is carried against the matching entries using the arguments.

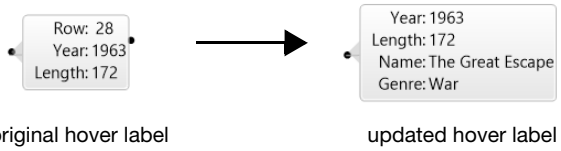
The exception is the `Annex` command, which adds a new entry to the grid if there are no matches with existing entries (to avoid unplanned naming collisions).

Example of Adding and Deleting Columns

This example describes how to use the `Expunge` command to remove the first entry added to the hover label when the underlying table has no Label columns, followed by two instances of the `Annex` command to add new rows. Note that you could have used a single `Annex` instance and passed to it a list of lists instead.

Figure 12.47 shows the original hover label and the updated hover label. Notice that Row was removed from the label. Name and Genre were added.

Figure 12.47 Example of the Expunge and Annex Commands



The following script creates the updated hover label shown in Figure 12.47.

```
Names Default To Here( 1 );
```

```
dt = Open( "$SAMPLE_DATA/Movie Inventory.jmp" );

// create the baseline visualization
gb = dt << Graph Builder(
        Size( 526, 454 ),
        Show Control Panel( 0 ),
        Variables( X( :Year ), Y( :Length ) ),
        Elements( Points( X, Y, Legend( 3 ) ), Smoother( X, Y, Legend( 4 ) ) )
    );

rpt = gb << Report();
box = rpt[Framebox( 1 )];
box << Set Gridlet(
     // delete Row
      Expunge( {{Matcher( 1 )}} ),
      Annex(
        // add the Name and Genre columns to the hover label
        {{Matcher( "Name" ), value( :Name[local:_firstRow] )},
         {Matcher( "Genre" ), value( :Genre[local:_firstRow] )}}
      )
);
```

Annex (Append)

`Annex({`*`matcher`*`, `*`value`*`(<script>), <Click(<script>)>})` adds a new row to the hover label grid. The `Annex` command corresponds to the Append tab in the Hover Label Editor Gridlet panel.

This command takes three arguments:

- `Matcher`: The new name. `Matcher` should not match existing names.
- `Value`: Required. The script is evaluated in the Hover Label Execution Context. The result is the value that appears for the new row. See "Work with the Hover Label Execution Context" for details.

  ```
  box << Set Gridlet(
         Annex( { "LDL, HDL", value(
         Char(:LDL[local:_firstRow]) || ", " || Char(:HDL[local:_firstRow])
         ) })
     );
  ```

- `Click`: Optional. `Click` is evaluated in the Hover Label Execution Context when the user clicks on annexed entry value. This option is equal to the `Click` attribute in graphlets. If present, the value portion of the new entry is rendered using a link style (blue text color with an underline) to indicate that it can be clicked.

Expunge (Delete)

`Expunge({Matcher})` deletes the matcher. The `Expunge` command corresponds to the Delete tab in the Hover Label Editor Gridlet panel.

All matching entries are removed from the grid.

```
box << Set Gridlet( Expunge( {"age"} ) );
```

Rename

`Rename({`*`Matcher`*`, Value(<string>)})` replaces the name portion of the matched entry or entries with the given value. The `Rename` command corresponds to the Rename tab in the Hover Label Editor Gridlet panel.

```
box << Set Gridlet( Rename( {Matcher( "name" ), Value( "students" )} ), );
```

Reformat

`Reformat({`*`Matcher`*`, <`*`format`*`>})` takes a matcher and an argument that specifies a format to be applied to the matched row (or rows) grid numerical value or values. Matching entries are changed to reflect the specified format.

```
box << Set Gridlet( Reformat( {:height, Format( "Scientific", 10, 4 )} ) );
```

Style

`Style({`*`Matcher`*`, <`*`style1`*`>, <`*`style2`*`>, ...})` takes a matcher and one or more style arguments. Here are the supported styles:

- Background Color
- Text Color
- Font (which further supports font family, size and style)
- Justification (left, center, or right)
- Matching entries are changed to reflect the specified styles.

```
box << Set Gridlet(
   Style(
      {{Matcher( "sex" ), Text Color( "Red" ), Font("Papyrus", 14,
      "Bold Underline")}}
   )
);
```

Create Drill-Down Graphs

A drill-down graph is created when you click on the thumbnail. Clicking on the hover graph opens it in a new window, enabling you to see the graph in more detail and also make modifications such as changing the data filter. For example, click a cell on a treemap to see details about the data in a new window (Figure 12.48). Press Ctrl and click to replace the contents of the last launched window instead of opening a new window.

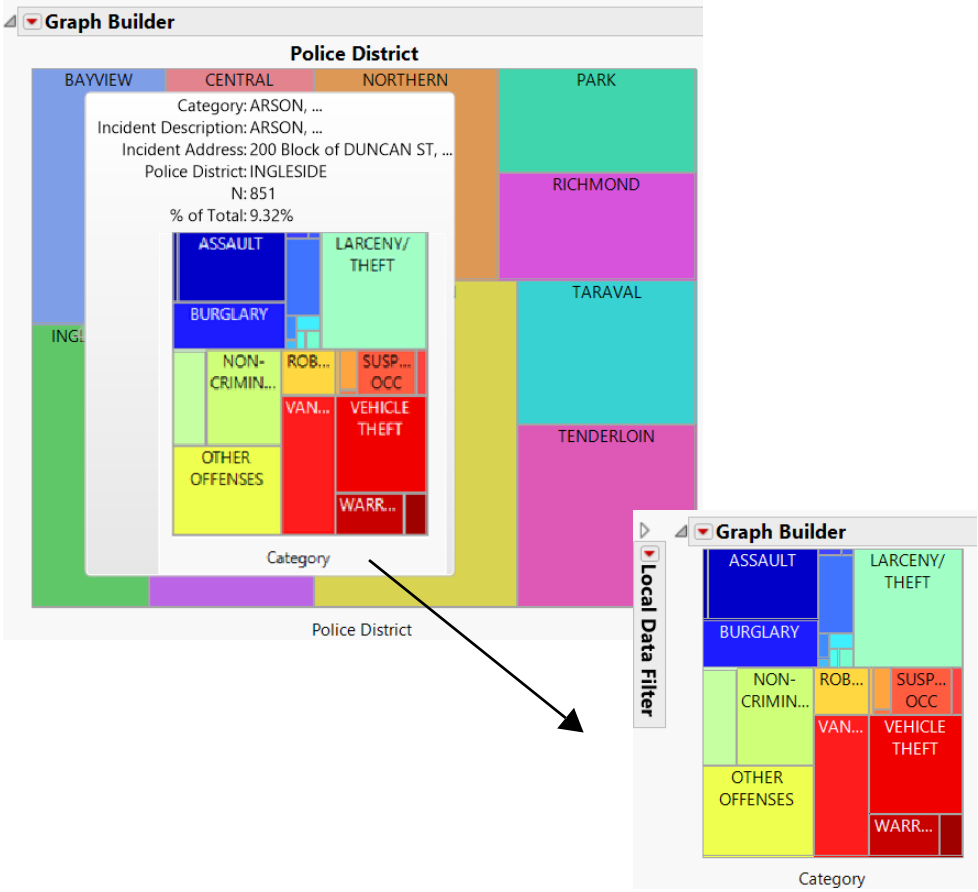
- The Next in Hierarchy column property enables you to define the behavior of categorical drill downs in preset graphs. This column property defines only one level of the drill down, from a category in a grouping role in the current visualization to the category in the grouping role in the next one, launched by the graphlet.
- If the Next in Hierarchy column property is not defined, the list of categorical or nominal columns from the data table that have not been added to a filter is used.

Notes:

- The Next in Hierarchy column property is not supported in graphlets defined using Paste Graphlet. See *Using JMP* for details about the Paste Graphlet command.
- If there is no measurement, JMP tries to get one from the data table and uses Count (or no measurement) if there are no numeric columns in the data table.

Figure 12.48 shows drilling-down on a cell in a treemap.

Figure 12.48 Example of a Drill-Down Graph on a Hover Label



The following script creates the simplified versions of the treemap presets shown in Figure 12.48.

```
Names Default To Here( 1 );

/* traverse the given list of column names, defining the Next in Hierarchy
   column property so that it creates a chain to enable the drill down */
setHierarchy = Function( {dt, cols},
   {Default Local},
   nCols = N Items( cols );
   If( nCols < 2,
      Write( "\!NsetHierarchy: Insufficient number of columns!" ),
      col = Column( dt, cols[1] );
      For( i = 2, i <= nCols, i++,
```

```
            nextCol = cols[i];
            Try(
               Column( dt, nextCol ),
               Write( "Invalid column name: " || nextCol );
               Throw();
            );
            // Eval() is needed for now
            col << Set Property( "Next in Hierarchy", Eval( nextCol ) );
            nih_name = col << Get Property( "Next in Hierarchy" );
            col = Column( dt, nextCol );
         );
      );
   );

   // open San Francisco Data.jmp
   dt = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );

   // add a Next in Hierarchy chain to the data table
   setHierarchy( dt, {"Police District", "Category", "Incident Description"} );

   /* Graph Builder treemap based on the first level of our hierarchy, "Police
      District" */
   gb = dt << Graph Builder(
      Size( 495, 440 ),
      Show Control Panel( 0 ),
      Variables( X( :Police District ) ),
      Elements( Treemap( X ) )
   );

   // get a handle to the graph framebox
   rpt = gb << Report;
   frame = rpt[Framebox( 1 )];

   /* configure the graphlet -- a hover label extension that adds a visualization
      thumbnail to the hover label. Clicking on the thumbnail launches the
      associated visualization in its own window (or replaces the graph in the
      current window on a Ctrl-click). */
   frame << Set Graphlet(
      // define the thumbnail and launch treemap
      Picture(
         // next-level grouping is retrieved from the Next in Hierarchy property
         local:propColName = local:_groupings[1] << GetProperty( "Next In
      Hierarchy" );
         local:nextGrouping = Column( local:_dataTable, local:propColName );

         // define the Graph Builder graph for the thumbnail
```

```
      Graph Builder(
         Size( 200, 250 ),
         Show Legend( 0 ),
         Show Title( 0 ),
         Show Footer( 0 ),
         Show Control Panel( 0 ),
         Variables( X( local:nextGrouping ) ),
         Elements( Treemap( X ) )
      );
   )
);

// add a pinned annotation to display the graphlet
frame << Add Pin Annotation(
   Seg( TreeMapSeg( 1 ) ),
   Index( 2 ),
   Index Row( 7033 ),
   UniqueID( 581248018 ),
   FoundPt( {52, 248} ),
   Origin( {27.6544421487603, 200.14987244898} ),
   Tag Line( 1 )
);
```

Work with the Hover Label Execution Context

Hover Label Execution Context variables define the environment in which your graphlet and textlet scripts are executed.

Local Data Filter

When defining a graphlet, you specify how to visualize the data that appears as a thumbnail. But how do you specify which rows should be displayed?

This information is specified indirectly you hover over different visual elements of the baseline graph (the graph to which a graphlet definition was added). Suppose that the baseline graph is a bar chart. Hovering over one of the bars indicates that all the rows behind that bar (and only those rows) should be included in the generation of the visualization. This visualization shows up as a thumbnail on the hover label. In other words, the hovered visual element provides the data context in which the visualization is generated.

This data context is expressed in a where clause that is computed by JMP based on:

- the type of graph used in the baseline visualization
- the columns assigned to its different roles
- the current values for the particular visual element being inspected

The resulting where clause is then used to create a local data filter that is assigned to the platform specified in the graphlet definition.

The following graph roles are supported as sources for the creation of local data filters:

- X, including nested columns
- Y, if not representing a summary statistic or other calculation
- Group X
- Group Y
- Wrap
- Page
- overlays

Numeric columns are used in comparison to a single value or checked for inclusion in a range, depending on the underlying visualization type. Visualizations that group numeric data in bins such as histograms and heat maps (and the box in box plots) result in the use of corresponding ranges in the where clause.

If present, the where clause of a Local Data Filter associated with the baseline visualization is captured as well.

Local Scope Context

The JSL expressions that configure different aspects of a graphlet and other hover label extensions (such as the textlet `Setup` and the gridlet `Append` and `Click` scripts) are executed inside a local namespace. This namespace is initialized with local variables that reveal information from the execution context: the hover label grid entries and references to the underlying visual element and associated data. These variables are used extensively by the graphlet presets.

Figure 12.4 describes variables that are provided in a graphlet local namespace. See "Complete Example of the Hover Label Execution Context Variables".

Note: To see the variables at work, right-click on a graph, select **Hover Label Editor**, select the Graphlet panel, and then select one of the presets. The code that appears in the Picture field illustrates how many of the Hover Label Excecution Context variables are used to control the graphlet's behavior.

Table 12.4 Hover Label Execution Context Variables

Variable	Description

`_mode`	A string indicating the current evaluation context. Values include `Picture`, `Click`, `Textlet`, `Graphlet`, and `Other`. You can use `_mode` to customize a Graph Builder expression to show more or less information.
	From the JMP log:
	`{"_mode", "Textlet"},`
	Example from the Bar preset:
	`If( local:_mode == "Picture",`
	`  local:sizeDef = "Size( 200, 250 ),";`
	`  local:l = 0;`
	`,`
	`  local:sizeDef = "";`
	`  local:l = 1;`
	`);`
	Note: `Other` means the other attributes that take expressions (for example, `SkipFiltersFlag`).

`_groupings`	A list of the current grouping (X or overlay roles) data column references. See "Example of Defining the Context of a Scatterplot Matrix and Ellipse". `{"_groupings", {:sex}}`
`_measurements`	A list of the current measurements (Y role) data column references. `{"_measurements", {:height}}`
`_filters`	A list of the current local data filter data column references. `{"_filters", {:sex}}`

Table 12.4 Hover Label Execution Context Variables *(Continued)*

Variable	Description

_where	A textual representation of the where clause associated with the hovered visual element. The expression is parsed to be used with JSL statements that take where clauses such as `Get Rows Where`. `_where` is useful to cache complex calculations that are based on context.
	`expr = Parse( local:_where );`
	`r = local:_dataTable << Get Rows Where( expr );`
	`ss = dt << Subset( Private, Invisible, Rows( r ) );`
	`Summarize( ss, exg = By( :sex ), exm = Mean( :height ) );`
	Another example:
	`{"_where", ":sex == \!"F\!""}`

_whereExpr	The where clause that is associated with the hovered visual element as a JSL expression.
	`local:_h = Round(Mean( :height[local:_dataTable << Get Rows Where( local:_whereExpr )] ), 2);`
	Another example:
	`{"_whereExpr", :sex == "F"}`

_displaySegName	The class name of the underlying seg. Useful when creating descriptive text or in the definition of graph-dependent drill-down logic.
	`Title("Graphlet for " || local:_displaySegName || ", depth is " || Char( local:_drillDepth ))`
	Another example:
	`{"_displaySegName", "BarSeg"}`

_dataTable	A reference to the underlying data table. Useful for creating subsets. See "Add Rich Text to Hover Labels Using Textlets" for an example. `{"_dataTable", Data Table( "Big Class" )}`

Table 12.4 Hover Label Execution Context Variables *(Continued)*

Variable	Description
`_summaryStatistic`	A string with the current summary statistic. Useful for passing to new graphs in a drill down. `{"_summaryStatistic", "Mean"}`
`_drillDepth`	An integer that indicates how many drill-down levels have been opened before. Useful to limit drill-down chains or to control `SkipFilters` logic. See "Example of Defining the Context of a Scatterplot Matrix and Ellipse" `{"_drillDepth", 1}`
`_firstRow`	The 1-based index of the first row associated with the hovered visual element. See "Example of Adding and Deleting Columns". `{"_firstRow", 1}`
`_underlyingRows`	The count of rows associated with the visual element that appears in the hover label. You can use this variable to decide the type of the visualization based on the number of rows to display, going from the low granularity of a heat map to a dense marker plot if the number of rows is small enough (and vice versa). See "Reapply" for an example. `{"_underlyingRows", 18}`

In addition to these variables, a variable is defined for each of the hover label grid entries. The name of this variable is taken for the grid entry name, with a '_' prefix.

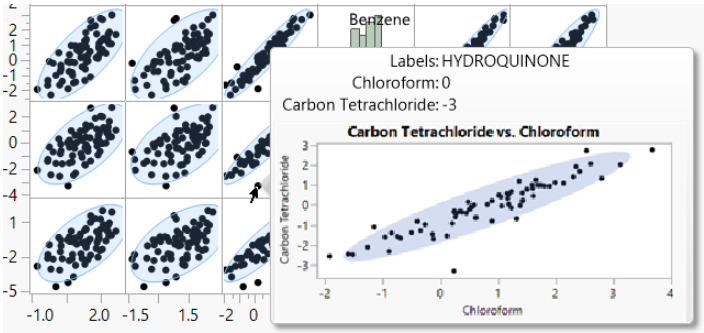
Note: See "Example of Defining the Context of a Scatterplot Matrix and Ellipse" for examples of these variables.

As an example, using Hover Label Execution Context variables, you can create a scatterplot matrix, hover over a point to show a close-up ellipse, and then click the ellipse to modify it in Graph Builder. The graphlet shows the same visualization as the one in the underlying scatterplot matrix cell (a scatterplot of the two columns associated with that cell).

Clicking on the thumbnail then has the effect of "extracting" the cell from the matrix and opening it up as an independent visualization.

The script shown in "Example of Defining the Context of a Scatterplot Matrix and Ellipse" enables the user to extract the matrix cells in the main scatterplot matrix.

Figure 12.49 Viewing an Ellipse in a Scatterplot Matrix



Example of Defining the Context of a Scatterplot Matrix and Ellipse

The following script shows an example of using the context in which the script runs. This script creates the scatterplot matrix shown in Figure 12.49.

```
Names Default To Here( 1 );

dt = Open( "$SAMPLE_DATA/Solubility.jmp" );
win = New Window( "Scatterplot Matrix Graphlet Example",
   plt = dt << Multivariate(
      Y(
         "1-Octanol"n,
         :Ether,
         :Chloroform,
         :Benzene,
         :Carbon Tetrachloride,
         :Hexane
      ),
      Estimation Method( "Row-wise" ),
      Scatterplot Matrix(
         Density Ellipses( 1 ),
         Shaded Ellipses( 1 ),
         Ellipse Color( 69 ),
         Horizontal( 1 )
      )
   )
);
frame = win << xpath( "//FrameBox" );
frame << Set Graphlet(
   Picture(
      Show( local:_displaySegName );
```

```
        If( local:_displaySegName == "MarkerSeg",
           // scatterplot cells
           Graph Builder(
              Size( 400, 200 ),
              Show Control Panel( 0 ),
              Show Legend( 0 ),
              Variables(
                 X( local:_groupings[1] ),
                 Y( local:_groupings[2] )
              ),
              Elements( Points( X, Y ), Ellipse( X, Y ) )
           ),
           // else histograms in matrix diagonal
           If( local:_N > 5, /* Note: only present if baseline graph
              is a histogram.*/
              Graph Builder(
                 Size( 400, 400 ),
                 Show Control Panel( 0 ),
                 Show Legend( 0 ),
                 Variables( X( local:_groupings[1] ) ),
                 Elements( Histogram( X ) )
              ),
              Graph Builder(
                 Size( 400, 400 ),
                 Show Control Panel( 0 ),
                 Show Legend( 0 ),
                 Variables( X( local:_groupings[1] ) ),
                 Elements( Points( X ) )
              )
           )
        );
     ),
     // skip filters if you are at the first drill level -- the original graph
     SkipFilters( local:_drillDepth == 1 ), // no list -- all filters
     Title( "Graphlet" ),
     Reapply( local:_displaySegName == "HistSeg" & local:_N > 5 )
  );
```

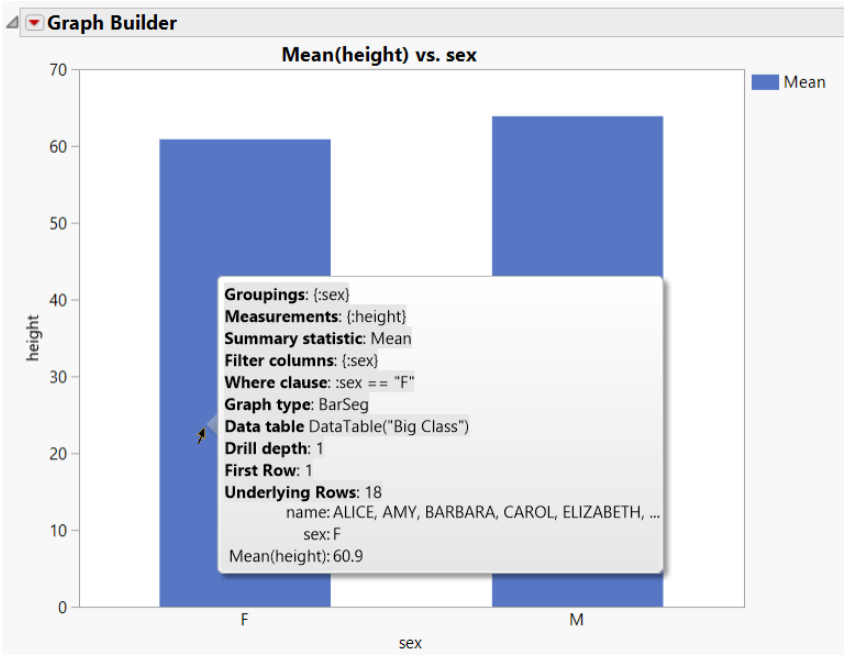
Complete Example of the Hover Label Execution Context Variables

The following script produces the graph shown in Figure 12.50. When you hover over a bar, the hover label appears. Details about the data and the Hover Label Context Execution variables appear in the log.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Graph Builder(
   Size( 528, 450 ),
```

```
   Show Control Panel( 0 ),
   Variables( X( :sex ), Y( :height ) ),
   Elements( Bar( X, Y, Legend( 3 ) ) ),
   SendToReport(
      Dispatch(
         {},
         "Graph Builder",
         FrameBox,
         {Set Textlet(
            Setup(
               local:encodedWhere = XML Encode( local:_where );
               /* Show() provides additional information about the
               HLEC variables in the log */
               Show( Namespace( "local" ) << Get Contents() );
            ),
            Markup(
               "
<b>Groupings</b>: {local:_groupings}
<b>Measurements</b>: {local:_measurements}
<b>Summary statistic</b>: {local:_summaryStatistic}
<b>Filter columns</b>: {local:_filters}
<b>Where clause</b>: {local:encodedWhere}
<b>Graph type</b>: {local:_displaySegName}
<b>Data table</b> {local:_dataTable}
<b>Drill depth</b>: {local:_drillDepth}
<b>First Row</b>: {local:_firstRow}
<b>Underlying Rows</b>: {local:_underlyingRows}
"
            )
         )}
      )
   )
);
```

Figure 12.50 Bar Chart with Textlet



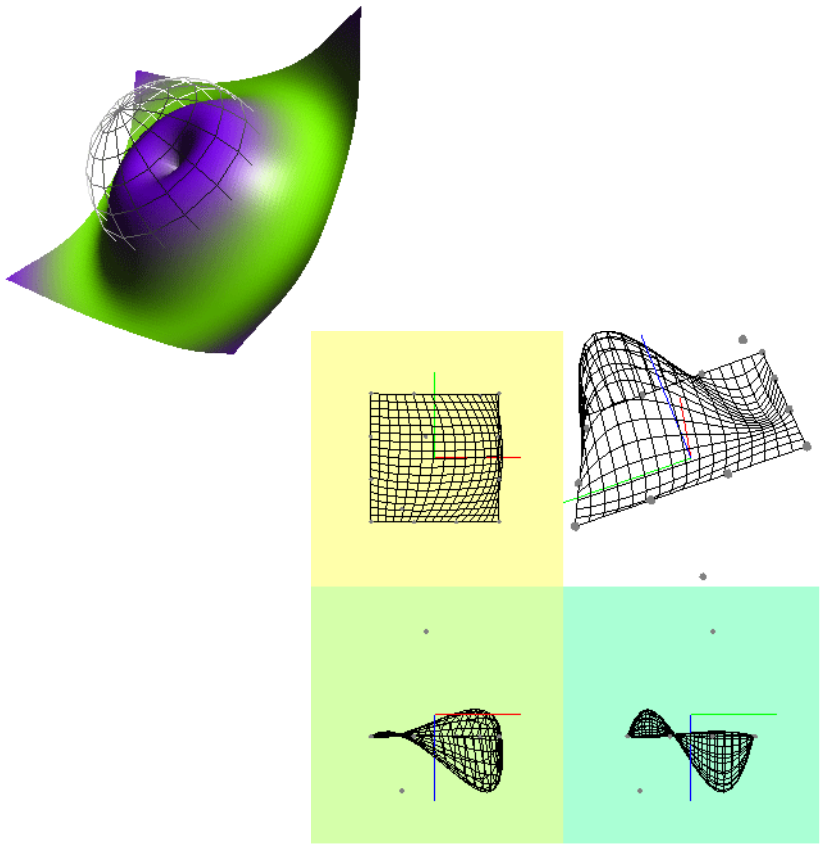
Chapter 13

Three-Dimensional Scenes

Scripting in Three Dimensions

JSL includes commands for scripting three-dimensional scenes derived from OpenGL. Although not a complete OpenGL® implementation, JSL's 3-D scene commands enable complex, interactive plots to be constructed and viewed. The Surface Plot platform in JMP is built using JSL scene commands.

Figure 13.1 Sample Three-Dimensional Shapes



Contents

About JSL 3-D Scenes . . . 751
JSL 3-D Scene Boxes . . . 751
Setting the Viewing Space . . . 754
Setting Up a Perspective Scene . . . 755
Setting up an Orthographic Scene . . . 756
Changing the View . . . 757
The Translate Command . . . 757
The Rotate Command . . . 757
The Look At Command . . . 759
The ArcBall . . . 760
Graphics Primitives . . . 761
Primitives Example . . . 764
Controlling the Appearance of Primitives . . . 765
Other Uses of Begin and End . . . 771
Drawing Spheres, Cylinders, and Disks . . . 771
Construction . . . 771
Lighting . . . 772
Drawing Text . . . 773
Using Text with Rotate and Translate . . . 773
Using the Matrix Stack . . . 774
Lighting and Normals . . . 778
Creating Light Sources . . . 778
Lighting Models . . . 780
Normal Vectors . . . 781
Shading Model . . . 781
Material Properties . . . 782
Alpha Blending . . . 783
Fog . . . 783
Example . . . 783
Bézier Curves . . . 784
One-Dimensional Evaluators . . . 785
Two-Dimensional Evaluators . . . 787
Using the Mouse . . . 788
Pick Commands . . . 789
Three-Dimensional Arguments . . . 790
Improve the Display of Graphics . . . 790

About JSL 3-D Scenes

JMP's 3-D scene language is built on top of the OpenGL API, extending, replacing, and leaving out various parts of the OpenGL API. As such, the 3-D scene language is not certified or licensed by Silicon Graphics, Inc. under the OpenGL API.

This chapter documents JMP's JSL commands for creating 3-D scenes but is not a tutorial on OpenGL programming. If you are not familiar with OpenGL programming, you might want supplemental material. If you are familiar with OpenGL programming, you still need to read this chapter because some items are nonstandard.

JMP is shipped with sample files in the Scene3D subfolder of Sample Scripts to get you started and give you some ideas. Some of the example scripts are similar to some of the examples in this chapter; some are almost complete applications.

The website https://opengl.org is a good source for information, as is your favorite search engine.

JMP's Scene 3-D language does some work for you that the OpenGL API requires you to do for yourself. JMP makes text easy, gives you a built-in arcball controller, and makes sure the matrix operations that belong on the model view stack and projection stack go on their respective stacks. JMP uses its own display list manager so that your scenes can be journaled and played back later, and provides a select mechanism that calls back to your JSL code to tell you what object in your scene is under the mouse, with almost no extra effort on your part. At this time, JMP does not provide access to some features, like texturing.

JSL 3-D Scene Boxes

These commands are necessary to set up and configure a 3-D scene.

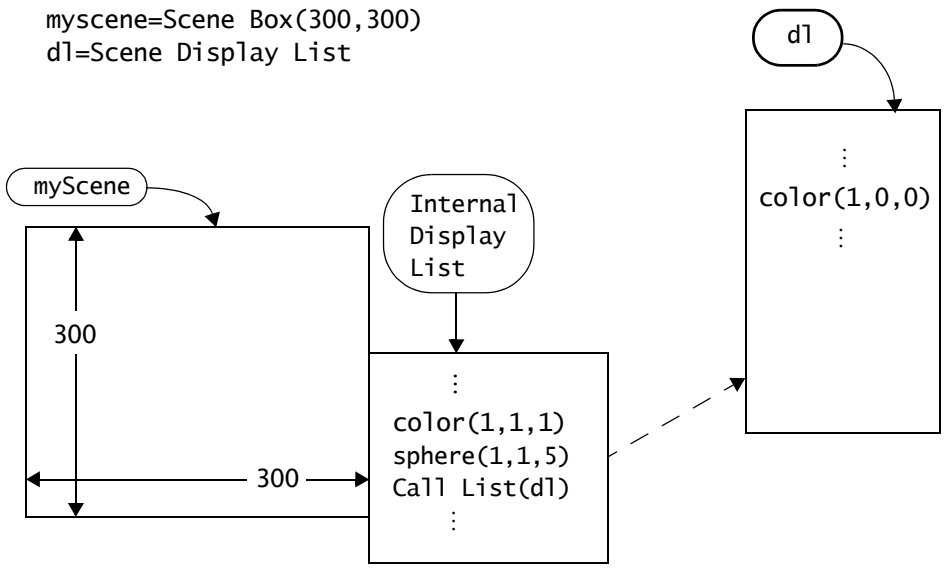
Like all displays in JMP (detailed in "Display Trees"), 3-D scenes must be placed in a display box (in this case, a Scene Box). This box is then placed in a window. Therefore, a simple 3-D scene script has the following form.

```
myScene = Scene Box(300, 300); // create a 300 by 300 pixel scene box
...(commands to set up the scene)...
New Window("3-D Scene", myScene); // draw the scene in a window
...(commands that manipulate the scene)
```

The scene can be sent messages that construct elements in the scene. Typical messages alter the viewer's vantage point, construct physical elements in the scene itself, or manipulate lights and textures. These messages are maintained in a display list and are manipulated in one of two ways:

- They are sent as messages to the scene, which immediately adds them to the scene's internal display list.
- They are sent as messages to a display list stored in a global variable, which is called by the scene's display list later.

Figure 13.2 Constructing Scenes



For example, of commands sent to the scene's display list directly, consider the following small script. Each of the commands is explained in detail later in the chapter.

```
scene = Scene Box( 400, 400 ); // make a scene box
New Window( "Example 1", scene ); // put the scene in a window
scene << Perspective( 45, 3, 7 ); // define the camera
scene << Translate( 0.0, 0.0, -4.5 ); // move to (0,0,-4.5) to draw
scene << Color( 1, 0, 0 ); // set the RGB color of the text
scene << Text( center, baseline, 0.2, "Hello, World." ); // add text
scene << Update; // update the scene
```

The first two lines create a scene and place it in a window. The `Perspective` command defines the viewing angle and field depth. By sending it as a message to the scene, it is immediately added to the scene's display list. Because the "Hello World" text is drawn at the origin (0, 0, 0), the `Translate` command is added to the display list. This command moves the camera back a bit so that the origin is in the field of vision. The color is set to red with the `Color` command, the text is drawn, and the `Update` command causes the scene to be rendered (in other words, causes the display list that contains the commands to be drawn.)

Figure 13.3 Hello World

Equivalently, the commands to construct the display can be accumulated in a display list stored in a global variable, which is then sent to the scene all at once. To define a global variable as a display list, assign it using the `Scene Display List` function. For example, to use the global `greeting` as a display list, issue the command

```
greeting=Scene Display List();
```

Display commands can then be sent as messages to `greeting`. An equivalent "Hello World" example using a display list follows.

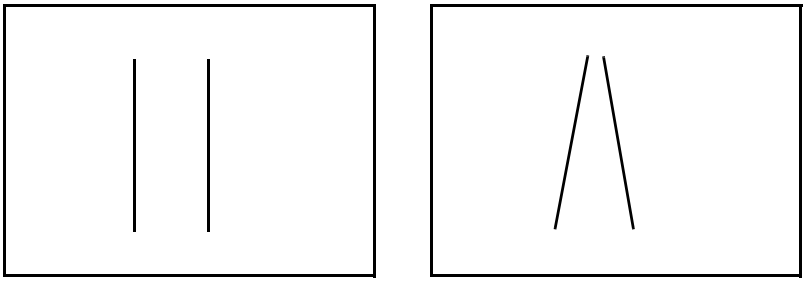
```
greeting = Scene Display List();
greeting << Color( 1, 0, 0 ); // set the RGB color of the text
greeting << Text( center, baseline, 0.2, "Hello, World." ); // add text

// draw the window and send it the stored display list
scene = Scene Box( 400, 400 ); // make a scene box
New Window( "Example 1", scene ); // put the scene in a window
scene << Perspective( 45, 3, 7 ); // define the camera
scene << Translate( 0.0, 0.0, -4.5 ); // move to (0,0,-4.5) to draw
scene << Call List( greeting ); // send the display list to the scene
scene << Update; // update the scene
```

Note which commands were separated into the display list, and which were applied to the scene directly. Those that manipulate the camera (`Translate` and `Rotate`) are applied to the scene. Those that define the object (`Color` and `Text`) were relegated to the display list. This is done so that the display list can be called many times to replicate the object at different positions.

Setting the Viewing Space

3-D scenes can be rendered in two ways. *Orthographic* projections place the elements in a box, where coordinates are not changed to accommodate the perspective of the viewer. *Perspective* projections modify the display to simulate the position of the elements in relation to the position of the viewer. For example, two parallel lines (like railroad tracks) stay parallel in orthographic projections, but seem to connect at a distance in perspective projections.

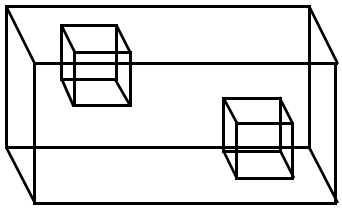
Figure 13.4 Parallel Lines in an Orthographic Projection (left) and a Perspective Projection (right)



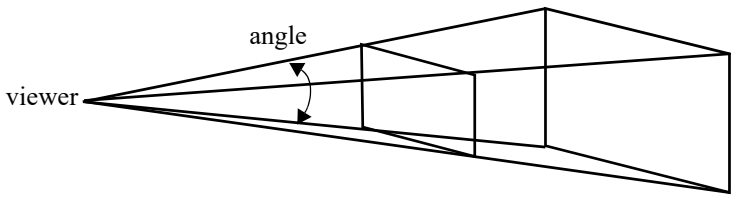
As another example, imagine looking at a tube edge-on (like a telescope). In an orthographic projection, the tube would appear as a thin circle. In a perspective projection, the circle would have a thickness; the hole at the far end of the tube would appear smaller than the close hole, and the interior of the tube is visible.

Therefore, the viewable space of an orthographic projection is a rectangular shape. That of a perspective projection is the frustum of a pyramid (that is, a pyramid whose top has been sliced off).

Figure 13.5 Comparing Projections



Orthographic Scene



Perspective Scene

In general, perspective projections give a more realistic view of the world, since it mimics the way an eye or a camera sees. Orthographic projections are important when it is essential to preserve dimensions, such as an architectural CAD program.

Setting Up a Perspective Scene

To set up a perspective scene in JSL, send the `Perspective` command to a display list.

```
Perspective (angle, near, far)
```

where *`angle`* is the viewing angle, *`near`* is the distance to the near plane, and *`far`* is the distance to the far plane, as illustrated in the drawing above. A couple of things need to be remembered when defining the viewing space.

- Items outside the viewing space (for example, closer than the *`near`* plane or farther than the *`far`* plane) are not drawn. They are clipped off.
- The ratio of *`far`* to *`near`* needs to be small. This enables the rendering engine to effectively determine which items should be drawn "on top of" other items, simulating closeness of items. The *near* argument must be greater than zero.

The "Hello World" example contains the line

```
scene << Perspective( 45, 3, 7 ); // define the camera
```

This defines a 45-degree viewing angle, with a near plane 3 units from the viewer and a far plane 7 units from the viewer.

The viewing angle functions in the same way as a wide angle or telephoto lens on a camera. Small viewing angles zoom into a drawing; wide angles zoom out. In other words, a small viewing angle maps the screen space onto a small portion of the scene, resulting in apparently larger scene elements. A large viewing angle maps the screen space onto a large portion of the scene, resulting in apparently small screen elements. The size of scene elements can therefore be manipulated using the *angle* argument of the `Perspective` function. The picture here shows the hello world script with perspective angles of 45 and 90 degrees.

Figure 13.6 Changing the Perspective

As an alternative to the `Perspective` command, you can define the actual viewing frustum with the `Frustum` command.

```
Frustum(left, right, bottom, top, near, far);
```

The frustum's viewing volume is defined by (*left, bottom, near*) and (*right, top, near*), which specify the (*x, y, z*) coordinates of the lower left and upper right corners of the near clipping plane; *near* and *far* give the distances from the viewpoint to the near and far clipping planes.

Setting up an Orthographic Scene

Orthographic scenes are specified in ways similar to perspective scenes. Issue the command

```
Ortho(left, right, bottom, top, near, far)
```

which specifies the four corners of the near plane, the distance to the near plane, and the distance to the far plane.

If you are dealing with a simple 2-D environment, you can set up a two-dimensional orthographic scene with the command

```
Ortho2D (left, right, bottom, top)
```

which specifies the corners of the two-dimensional view.

Changing the View

One of the advantages of creating a 3-D scene is the ease that they can be viewed from different angles and positions. The `Translate` and `Rotate` commands let you set the position from which you view the scene.

In addition, you can use the `ArcBall` command to enable the user to change the viewing angle interactively.

The Translate Command

You have actually seen the `Translate` command in earlier sample scripts. It sets the position from which the scene is viewed. The arguments give the amount to move from the current position in the *x*, *y*, and *z* direction.

```
Translate (x, y, z)
```

For example,

```
Translate( 0.0, 0.0, -2 );
```

moves the origin two units in the negative *z* direction.

Initially, the origin and camera were at the same place. Now, the camera can see the origin because the camera faces down the negative *z* axis.

The Rotate Command

The `Rotate` command is used to modify the viewing angle of a scene. It has the following format.

```
Rotate (degrees, xAxis, yAxis, zAxis)
```

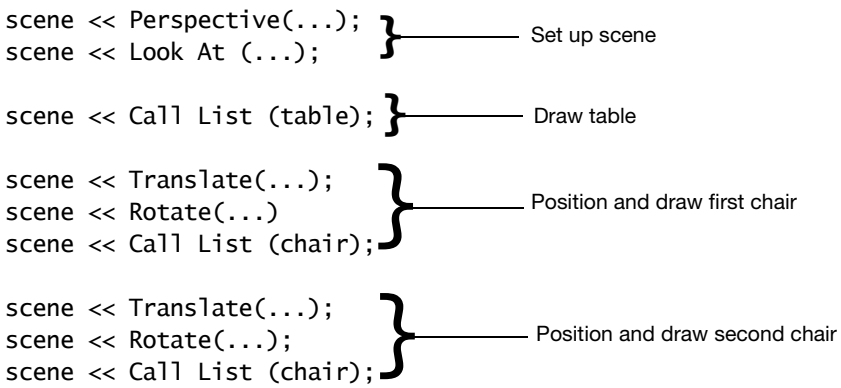
This rotates by degrees around the axis described by the vector (*xAxis*, *yAxis*, *zAxis*). For example, to rotate a model 90 degrees about the *x*-axis, use `Rotate( 90, 1, 0, 0 )`.

You can also specify the three axis values in a matrix. For example, `Rotate( 90, [1, 0, 0] )`.

Note: The `Rotate` command uses degrees, in contrast to JMP's trigonometric functions, which use radians.

`Translate` and `Rotate` are also used to position objects with respect to each other. The first `Translate` or `Rotate` can be thought of as positioning everything that follows with respect to the camera. Subsequent `Translate` and `Rotate` commands are used to position objects, such as spheres, cylinders, disks, and display lists in `Call List` and `ArcBall` commands. For example, suppose you have a display list named `table` and another named `chair`. Your scene might look like this:

Figure 13.7 Using Translate and Rotate

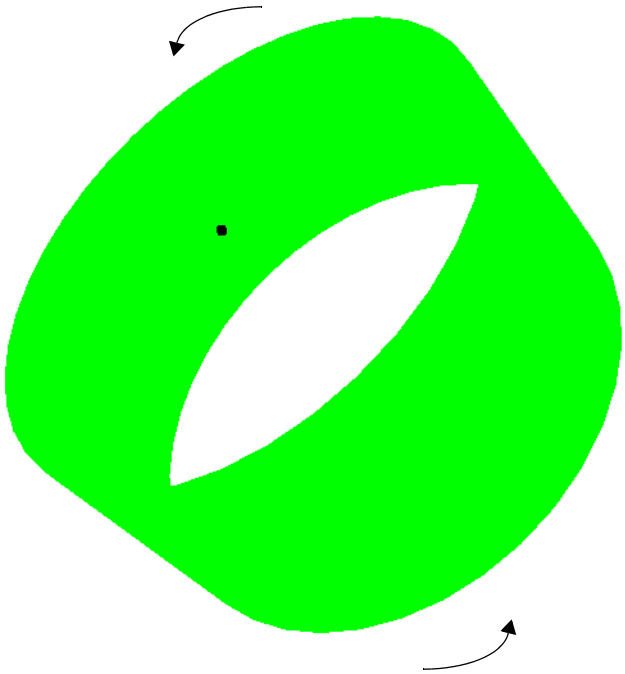


The following example uses the `Rotate` command inside a `For` loop to continuously change the viewing angle of a scene. It draws a cylinder that swings around a central point. This central point is shown by a small sphere.

```
scene = Scene Box( 600, 600 ); // make a scene box...holds an OpenGL scene.

New Window( "Example 1", scene ); // put the scene in a window

For( i = 1, i < 360, i++,
   scene << Clear;

   // the lens is 50 degrees, near is 1 units from the camera, far is 10.
   scene << Perspective( 50, 1, 10 );
   scene << Translate( 0.0, 0.0, -2 );

   scene << Rotate( i, 1, 0, 0 );
   scene << Rotate( i * 3, 0, 1, 0 );
   scene << Rotate( i * 3 / 2, 0, 0, 1 );
```

```
scene << Color( 0, 1, 0 ); // green for cylinder
scene << Cylinder( 0.5, 0.5, 0.5, 40, 10 );
scene << Color( 0, 0, 0 ); // black for sphere
scene << Sphere( 0.01, 10, 5 );
scene << Update;
Wait( 0.01 ); );
```

Figure 13.8 Rotating a Cylinder



Note the use of the `Update` command at the end of the scene messages. This command tells JMP to make the displayed screen agree with the current state of the display list. It is important to clear the list at the beginning (so that the list does not contain the old angles as well as the current) and update the scene after each change.

The Look At Command

The `Look At` command is an alternative way to set the camera view.

`Look At(` *eyeX*, *eyeY*, *eyeZ*, *centerX*, *centerY*, *centerZ*, *upX*, *upY*, *upZ* `)`

The Look At command puts the camera at the *eye* coordinates and points it toward the *center* coordinates. The *up* vector describes how the camera is rotated on its line of sight. The model is typically constructed at the origin. Therefore, a JMP scene should have either a `Look At` or a `Translate` command near its beginning to move the camera away from the origin.

First, clear the scene box of any commands from the previous frame.

```
scene << Clear;
```

Then use one of these projections:

```
scene << Perspective( 45, 2, 10 );
scene << Frustum( -.5, .5, -.5, .5, 1, 10 );
scene << Ortho( -2, 2, -2, 2, 1, 10 );
scene << Ortho2d( -2, 2, -2, 2 );
```

Note: If you use the `ortho2d` projection, you should not also set the camera position using either `Translate` or `Look At`.

Finally, use either `Translate` or `Look At` to set the camera position:

```
scene << Translate(0.0, 0.0, -4.5);
/* the camera faces down the negative Z axis.
   move it back so that 0,0,0 is in view. */
scene << Look At( /*eye*/ 3,3,3, /*center*/ 0,0,0, /*up*/ 1,0,0 );
/*this is much easier. */
```

Once the scene and camera position are set, add your model.

The ArcBall

Sometimes you want a scene to rotate based on the movements of the mouse. The Surface Plot platform in JMP is an example of a 3-D scene that rotates based on mouse movements.

An *ArcBall* creates a sphere around the 3-D scene and enables the user to click the sphere's surface and drag it around, thus causing the scene to rotate.

Use an `ArcBall` instead of a `Call List` command to place the scene in an ArcBall. Scenes that are attached to an ArcBall automatically respond to clicks and drags of the mouse. Custom programming is not needed. However, rotations made in the arcball are not saved. (Technically, the ArcBall is surrounded by an implicit `Push Matrix` and `Pop Matrix` block, so the movements are gone after it returns. See "Using the Matrix Stack" for more information about pushing and popping.)

For example, examine the script from "Primitives Example". Change the single line

```
scene << CallList(shape); // send the display list to the scene
```

so that it reads

```
scene << ArcBall(shape,2); // send the display list to an arcball
```

This displays the script with an associated arcball with diameter 2. When you run the script and the window appears, right-click and select **Show ArcBall > Always** from the menu that appears.

Note: ArcBall comes from an article by Shoemake (1994) found in *Graphics Gems IV*, published by Academic Press.

This sets the display so that the ArcBall is always showing. Click and drag on the ArcBall to rotate the scene. The right-click menu with Background Color, Use Hardware Acceleration, and Show ArcBall is always available, whether the scene is displayed through a platform, in a journal, or through JSL.

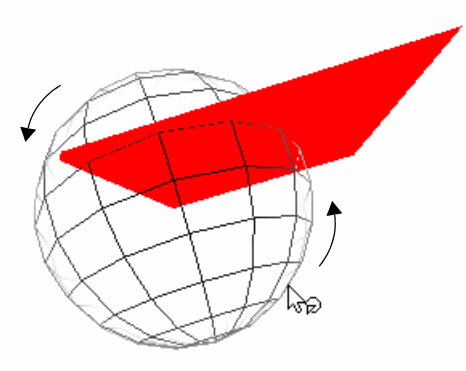
Note: The ArcBall does not have to be showing to react to mouse commands. It is shown here for display purposes only.

You can also set the display state of the ArcBall in JSL using the `Show ArcBall` command.

```
scene << Show Arcball(state)
```

where *state* is `During Drag`, `Always`, or `Never`.

Figure 13.9 Showing the Arc Ball



Graphics Primitives

All scenes in JSL are built with a small number of graphics primitives. These fundamental elements function as the building blocks for complicated scenes.

Every graphics primitive involves specifying vertices. In some cases, the vertices are simply drawn as points. In others, the vertices are connected to form polygons. To draw a primitive, you must specify the type of primitive and the coordinates and properties of the vertices involved. In JSL, this specification is accomplished through the `Begin` and `End` statements.

```
scene<<Begin(primitive type);
...(commands specifying vertices and their properties)...
scene<<End();
```

To specify the coordinates of the vertices, use the `vertex` command.

```
scene<<Begin(primitive type);
scene<<Vertex(x, y, z);
...
scene<<End();
```

The options for *`primitive type`* are the following. In these examples, assume that `v0`, `v1`, and so on, have been specified between a `Begin` and `End` pair, similar to the following.

```
scene<<Begin(primitive type);
scene<<Vertex(x0,y0,z0) // specify vertex v0
scene<<Vertex(x1,y1,z1) // specify vertex v1
...
scene<<Vertex(xn,yn,zn) // specify vertex vn
scene<<End();
```

Table 13.1 Primitive Types

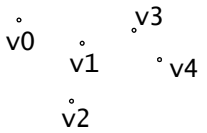
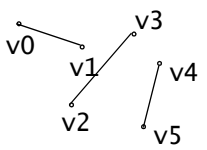
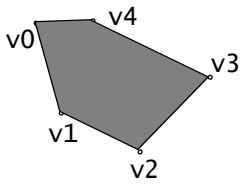
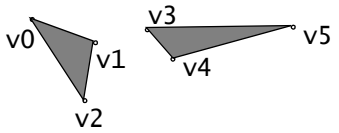
primitive type=POINTS	Draws a point at each of the vertices.	
primitive type=LINES	Draws a series of (unconnected) line segments. Segments are drawn between `v0` and `v1`, between `v2` and `v3`, and so on. If `n` is odd, the last vertex is ignored.	
primitive type=POLYGON	Draws a polygon using the points `v0`,..., `vn` as vertices. Three vertices must exist, or nothing is drawn. In addition, the polygon specified must not intersect itself and must be convex. If the vertices do not satisfy these conditions, the results are unpredictable.	
primitive type=TRIANGLES	Draws a series of (disconnected) triangles using vertices `v0`, `v1`, `v2`, then `v3`, `v4`, `v5`, and so on. If the number of vertices is not an exact multiple of 3, the final one or two vertices are ignored.	

Table 13.1 Primitive Types *(Continued)*

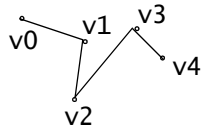
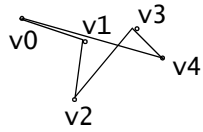
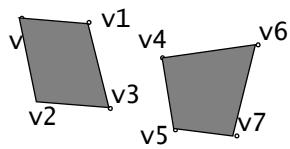
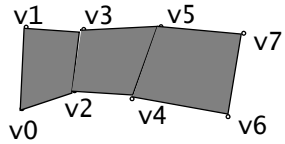
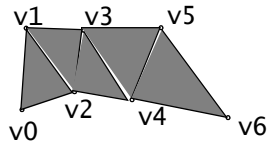
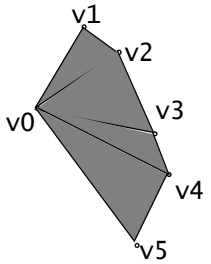
primitive type=LINE_STRIP	Draws a line segment from v0 to v1, then from v1 to v2, and so on. Therefore, n vertices specify n–1 line segments. Nothing is drawn unless there is more than one vertex. There are no restrictions on the vertices describing a line strip; the lines can intersect arbitrarily.	
primitive type=LINE_LOOP	Same as LINE_STRIP, except that a final line segment is drawn from the last vertex to the first, completing a loop.	
primitive type=QUADS	Draws a series of quadrilaterals (four-sided polygons) using vertices v0, v1, v2, v3, then v4, v5, v6, v7, and so on. If the number of vertices is not a multiple of 4, the final one, two, or three vertices are ignored.	
primitive type=QUAD_STRIP	Draws a series of quadrilaterals (four-sided polygons) beginning with v0, v1, v3, v2, then v2, v3, v5, v4, then v4, v5, v7, v6, and so on. The number of vertices must be at least 4 before anything is drawn, and if odd, the final vertex is ignored.	
primitive type=TRIANGLE_STRIP	Draws a series of triangles (three-sided polygons) using vertices v0, v1, v2, then v2, v1, v3 (note the order), then v2, v3, v4, and so on. The ordering is to ensure that the triangles are all drawn with the same orientation so that the strip can correctly form part of a surface. There must be at least three vertices for anything to be drawn.	

Table 13.1 Primitive Types *(Continued)*

primitive type=TRIANGLE_FAN	Same as TRIANGLE_STRIP, except that the vertices are v0, v1, v2, then v0, v2, v3, then v0, v3, v4, and so on.	

Primitives Example

The following short example illustrates the use of a graphics primitive.

```
shape = Scene Display List(); // create a display list and send it commands
shape << Color( 1, 0, 0 ); // set the RGB color of the text
shape << Begin( POLYGON );
shape << Vertex( 0, 0, 0 );
shape << Vertex( 0, 3, 0 );
shape << Vertex( 3, 3, 0 );
shape << Vertex( 5, 2, 0 );
shape << Vertex( 4, 0, 0 );
shape << Vertex( 2, -1, 0 );
shape << End();

// draw the window and send it the stored display list
scene = Scene Box( 400, 400 ); // make a scene box
New Window( "Primitive", scene ); // put the scene in a window
scene << Perspective( 90, 3, 7 ); // define the camera
scene << Translate( 0.0, 0.0, -5 ); // move to (0,0,-5) to draw
scene << Call List( shape ); // send the display list to the scene
scene << Update; // update the scene
```

The first section of the script creates a display list named `shape`. Inside this display list, a polygon is defined using six vertices.

The second section of the script creates a scene box and a new window. It then uses the `Call List` message to put the list in the display.

Note that all the *z*-coordinates are zero, which makes sure the polygon lies in a plane. Polygons that do not lie in a plane can cause unpredictable results.

Experiment with the line

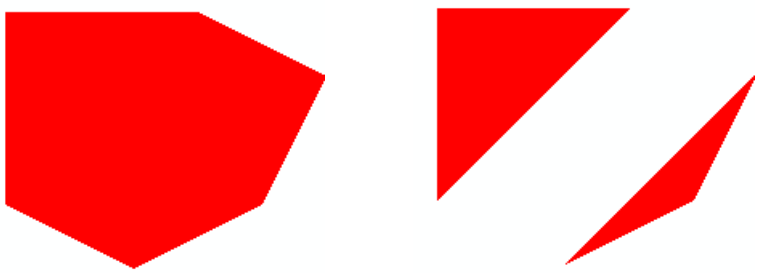
```
shape <<Begin(POLYGON);
```

by changing it to some of the other primitive types. For example, changing it to

```
shape <<Begin(TRIANGLES);
```

results in a different picture.

Figure 13.10 Polygon (left) and Triangles (right)



Controlling the Appearance of Primitives

JSL has several commands that let you tailor-make the appearance of primitive drawing objects. You can also specify the widths of lines and their stippling pattern (that is, whether they are dashed, dotted, and so on.)

Size and Width

To set the point size of rendered objects, use the `Point Size` command.

```
Point Size( n )
```

where *n* is the number of pixels. Note that this might not be the actual number of pixels rendered, depending on other settings such as anti-aliasing and your hardware configuration.

Set the line width using the `Line Width` command

```
Line Width( n)
```

where *n* is the number of pixels. The argument *n* must be larger than zero and is, by default, one.

Stippling Pattern

To make stippled lines, use the `Line Stipple` command.

```
Line Stipple( factor, pattern )
```

Factor is a stretching factor. *Pattern* is a 16-bit integer that turns pixels on or off. Use `Enable(LINE_STIPPLE)` to turn the effect on.

To construct a line stippling pattern, write a 16-bit binary number that represents the stippling pattern that you desire. Note that the pattern should read from right to left, so your representation might seem backward to the way it is rendered. Convert the binary number to an integer and use this as the *`pattern`* argument.

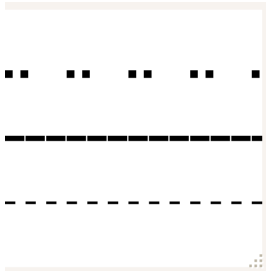
For example, imagine you want the dotted line pattern 0000000011111111. This is equal to 255 in decimal notation, so use the command `Line Stipple(1, 255)`.

The *`factor`* argument expands each binary digit to two digits. In the example above, `Line Stipple(2, 255)` would result in 0000000000000000111111111111111.

For example, the following script draws three lines, each of different widths (the `Line Width` commands) and stippling patterns.

```
scene = Scene Box( 200, 200 ); // make a scene box...holds an OpenGL scene

New Window( "Stipples", scene ); // put the scene in a window
scene << Ortho( -2, 2, -2, 2, -1, 1 );
scene << Color( 0, 0, 0 ); // set the RGB color of the text

scene << Enable( LINE_STIPPLE );
scene << Line Width( 2 );
scene << Line Stipple( 1, 255 );

scene << Begin( LINES );
scene << Vertex( -2, -1, 0 );
scene << Vertex( 2, -1, 0 );
scene << End();

scene << Line Width( 4 );
scene << Line Stipple( 1, 32767 );

scene << Begin( LINES );
scene << Vertex( -2, 0, 0 );
scene << Vertex( 2, 0, 0 );
scene << End();

scene << Line Width( 6 );
scene << Line Stipple( 3, 51 );

scene << Begin( LINES );
scene << Vertex( -2, 1, 0 );
scene << Vertex( 2, 1, 0 );
scene << End();
scene << Update;
```

Figure 13.11 Stipples



Note: Stipple patterns "crawl" on rotating models because they are in screen pixels, not model units. Lines in the model change length on the screen even though nothing changes in model units.

Fill Pattern

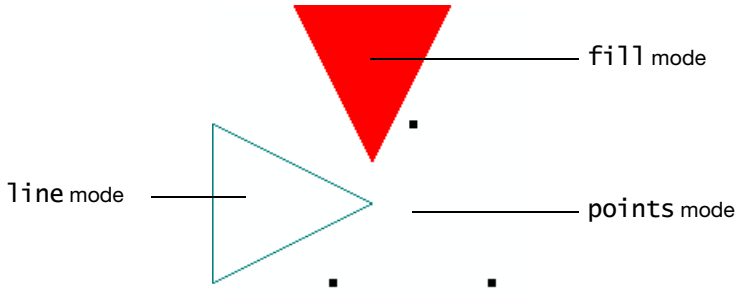
Polygons are rendered with both a front and a back, and the drawing mode of each side is customizable. This enables the user to see the difference between the back and front of the polygon.

To set the drawing mode of a polygon, use the `Polygon Mode` command.

```
Polygon Mode( face, mode )
```

where *`face`* can be `FRONT`, `BACK`, or `FRONT_AND_BACK`, and *`mode`* can be `POINT`, `LINE`, or `FILL`.

Figure 13.12 Points, Line, and Fill Modes



For example, the following script creates a display list that defines a triangle. This display list is used three times in conjunction with `Translate`, `Rotate`, and `Color` commands to draw triangles in three positions. In addition, the `Polygon Mode` command changes the drawing mode of each triangle. Note there is no explicit call to the `FILL` mode, since it is the default.

The following table dissects the script, showing how the `Translate` and `Rotate` commands accumulate to manipulate a single display list.

Table 13.2 Translate and Rotate Commands

Code from above script	comments
`shape = Scene Display List();`	Creates a display list

`shape << Begin( TRIANGLES );`
`shape << Vertex( 0, 0, 0 );`
`shape << Vertex( -1, 2, 0 );`
`shape << Vertex( 1, 2, 0 );`
`shape << End();`
	Creates a display list named shape that holds vertices for the triangles. All the z vertices are zero since this is a two dimensional scene

`scene = Scene Box( 200, 200 );`
`New Window( "Fill Modes", scene );`
`scene << Ortho2d( -2, 2, -2, 2 );`
	Put the scene in a display box, and create a new window.

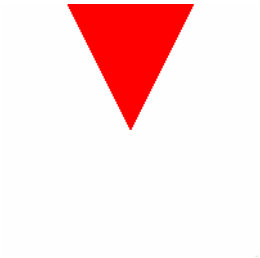
`scene << Color(1,0,0);`
`scene << Call List( shape );`
 	
`// update the scene to see the triangle`
`scene << Update;`
	Draw the first triangle in red.
	 

Table 13.2 Translate and Rotate Commands *(Continued)*

Code from above script	comments

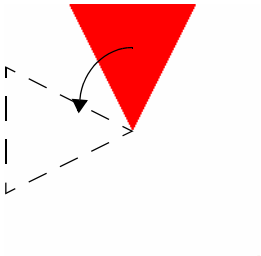
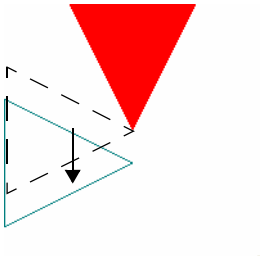
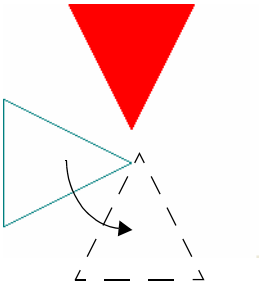
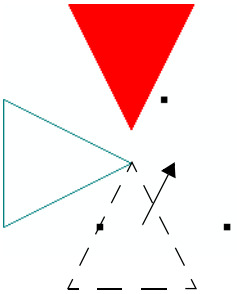
`scene << Rotate ( 90, 0, 0, 1 );`
`scene << Translate ( -0.5, 0, 0 );`
`scene << Color( 0, 0.5, 0.5 );`
`scene << Polygon Mode( FRONT_AND_BACK, LINE );`
`scene << Call List( shape );`
 	
`// update the scene to see the triangle`
`scene << Update;`
	Draw the second triangle in teal. Note that we first rotate the triangle.
	 
	And then translate it.
	 

Table 13.2 Translate and Rotate Commands *(Continued)*

Code from above script	comments

`scene << Rotate( 90, 0, 0, 1 );`
`scene << Translate( -0.5, -1, 0 );`
`scene << Color( 0, 0, 0 );`
 	
`// large points so that they are visible`
`scene << Point Size( 5 );`
`scene << Polygon Mode( FRONT_AND_BACK, POINT );`
`scene << Call List( shape );`
 	
`// update the scene to see the triangle`
`scene << Update;`
	Draw the third triangle as black points. First rotate.
	 
	And then translate to get the final picture.
	 

Some developers use the fill mode in concert with the line mode to draw a filled polygon with a differently colored border. However, due to the way the figures are rendered, they sometimes do not line up correctly. The `Polygon Offset` command is used to correct for this so-called "stitching" problem.

`Polygon Offset (`*`factor`*`, `*`units`*`)`

To enable offsetting, use `Enable(POLYGON_OFFSET_FILL)`, `Enable(POLYGON_OFFSET_LINE)`, or `Enable(POLYGON_OFFSET_POINT)`, depending on the desired mode. The actual offset values are calculated as m*(*`factor`*)+r*(*`units`*). m is the maximum depth slope of the polygon and r is the smallest value guaranteed to produce a resolvable difference in window coordinate depth values. Start with `Polygon Offset(1,1)` if you need this.

An example of `Polygon Offset` is in the Surface Plot platform, when a surface and a mesh are displayed on top of each other, or a surface and contours displayed on top of each other. In either case, the surface would interfere with the lines if the lines were not moved closer or the surface moved farther from the viewer.

Other Uses of Begin and End

Although vertices are typically specified between begin and end statements, there are other commands that are valid. These commands are discussed in other sections of this chapter.

- `Vertex` adds a vertex to the list
- `Color` changes the current color
- `Normal` sets the normal vector coordinates
- `Edge Flag` controls drawing of edges
- `Material` sets material properties
- `Eval Coord` and `Eval Point` generate coordinates
- `Call List` executes a display list.

Drawing Spheres, Cylinders, and Disks

There are several pre-defined commands that allow for quick rendering of spheres, cylinders, and disks. The advantage of these commands is not only their ease-of-use, but that they have special lighting properties (their "normals") built in.

Construction

The following commands are used to construct cylinders, disks, partial disks, and spheres.

Cylinders

```
Cylinder( baseRadius, topRadius, height, slices, stacks )
```

`baseRadius` is the radius of the cylinder's base. Similarly, *`topRadius`* is the radius of the top. *`height`* is the height of the cylinder.

`Slices` can be 10 for a reasonably accurate cylindrical shape. Using `QuadricNormals(Smooth)` helps the appearance.

`Stacks` sets the number of vertices available for lighting reflections. Use a larger value for *`Stacks`* for accurate "hot-spots".

Disks

The following command draws a paper-thin disk with an *`innerRadius`* hole in the middle.

```
Disk( innerRadius, outerRadius, slices, loops )
```

Like `Cylinder`, *`slices`* controls the accuracy of the curve and *`loops`* makes more vertices (for lighting accuracy).

```
Partial Disk( innerRadius, outerRadius, slices, loops, startAngle,
  sweepAngle )
```

The `Partial Disk` command works like `Disk`, but with a slice of the disk removed. Specify the part of the disk that is showing using *`startAngle`* and *`sweepAngle`*.

Spheres

The following command draws a sphere with the specified *`radius`*.

```
Sphere( radius, slices, stacks )
```

The *`slices`* can be thought of as longitudes and *`stacks`* as latitudes. About 10 of each make a nicely drawn sphere.

Lighting

It is not necessary to make specific calculations of normal vectors (as is the case for customized surfaces) for spheres, disks, and cylinders. However, you can use the following commands to customize the automatic lighting.

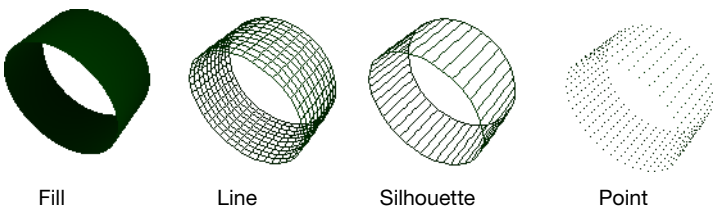
`Quadric Normals(`*`mode`*`)` tells what type of normal should be automatically generated. The argument *`mode`* can be `None`, `Flat`, or `Smooth`. `Flat` makes faceted surfaces. `Smooth` makes the normals at each vertex be the average of the adjacent polygons.

`Quadric Orientation(`*`mode`*`)` determines which way the normals point. The argument *`mode`* can be `Inside` or `Outside`.

`Quadric Draw Style(`*`mode`*`)` specifies the drawing mode. The argument *`mode`* can be `Fill`, `Line`, `Silhouette`, or `Point`.

JMP uses the values that you set for `Quadric Normals`, `Quadric Orientation`, and `Quadric Draw Style` for subsequently generated cylinders, disks, and spheres.

Figure 13.13 Draw Styles



Note: Other OpenGL documentation refers to quadric objects. JMP has only one and always uses it.

Drawing Text

As shown in the "Hello World" example above, text is added to a scene using the `Text` command.

`Text( horz, vert, size, string, <billboard>)`

- *horz* can be `Left`, `Center`, or `Right` justification.
- *vert* can be `Top`, `Middle`, `Baseline`, or `Bottom` justification.
- *size* represents the height of a capital letter M in model coordinates.
- *string* is the text to draw.
- *billboard* is an optional argument that causes the text to rotate with the model. Text with this option always faces the viewer.

The font is always the JMP Text font. You can change the text font from the preferences menu, but because of the way JMP caches fonts for scenes, changes might not take effect until JMP is restarted.

Note: Text is not part of the standard OpenGL definition.

Using Text with Rotate and Translate

The following example uses the text command in conjunction with the `Translate` and `Rotate` commands.

```
scene = Scene Box( 600, 600 ); // make a scene box...holds an OpenGL scene
New Window( "Example 2", scene ); // put the scene in a window

// the "lens" is 45 degrees, near is 3 units from the camera, far is 7
scene << Perspective( 45, 3, 7 );

// move the world so that 0,0,0 is visible in the camera
scene << Translate( 0.0, 0.0, -4.5 );

// rotate the first text about the Y (vertical on screen) axis
scene << Rotate( 30, 0, 1, 0 );
scene << Color( 1, 0, 2 ); // magenta
scene << Text( "center", "baseline", .2, "Top magenta string" );
```

```
// the next string is even farther away from the camera
scene << Translate( 0.0, 0.0, -2.0 );

// rotate the second text about the Y (vertical on screen) axis
scene << Rotate( 30, 0, 1, 0 );
scene << Color( 0, 1, 1); // aqua blue
scene << Text( "center", "baseline", .2, "Back aqua blue very long string" );

// update the display box in the window using the current display list
scene << Update;
```

Figure 13.14 Rotating and Translating Text Strings

Note that the aqua blue string extends backward beyond the far clipping plane. Change the 7 to 10 in the `Perspective` command to see the complete string.

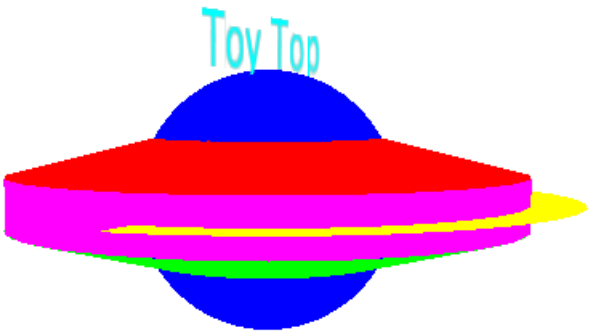
Using the Matrix Stack

JMP 3-D scenes use a matrix stack to keep track of the current transform. The stack is initialized to the identity matrix, and each time a translate, rotate, or scale command is given, the top matrix on the stack is changed.

Note: Unlike many OpenGL implementations, JMP does not use a transposed matrix.

The JSL example below uses `Push Matrix` and `Pop Matrix` to position pieces of the toy top and then return to the origin. This is faster than using the `Translate` command a second time in reverse.

Figure 13.15 Drawing With a Matrix Stack




```
toyTop = Scene Display List();
toyTop << Push Matrix;
toyTop << Translate( 0, 0, .1 );
toyTop << Color( 1, 0, 0 ); // red

// baseRadius, topRadius, height, slices, stacks
toyTop << Cylinder( 1, .2, .2, 25, 5 );
toyTop << Pop Matrix;
toyTop << Push Matrix;
toyTop << Translate( 0, 0, -.1 );
toyTop << Rotate( 180, 1, 0, 0 );
toyTop << Color( 0, 1, 0 ); // green
toyTop << Cylinder( 1, .2, .2, 25, 5 );
toyTop << Pop Matrix;

toyTop << Color( 0, 0, 1 ); // blue
toyTop << Sphere( .5, 30, 30 ); // radius, slices, stacks

toyTop << Color( 1, 1, 0 ); // yellow

// innerRadius, outerRadius, slices, rings, startAngle, sweepAngle
toyTop << Partial Disk( 1, 1.2, 25, 2, 0, 270 );

toyTop << Push Matrix;
toyTop << Translate( 0, 0, -.1 );
toyTop << Color( 1, 0, 1 ); // magenta
toyTop << Cylinder( 1, 1, .2, 25, 3 );

// baseRadius, topRadius, height, slices, stacks
toyTop << Pop Matrix;
```

```
toyTop << Push Matrix;
toyTop << Rotate( 90, 1, 0, 0 );
toyTop << Translate( 0, .5, 0 );
toyTop << Color( 0, 1, 1 ); // cyan
toyTop << Text( "center", "baseline", .2, "Toy Top" );
toyTop << Pop Matrix;

scene = Scene Box( 600, 600 ); // make a scene box...holds an OpenGL scene

New Window( "Example 3", scene ); // put the scene in a window
scene << Perspective( 45, 3, 7 );
scene << Translate( 0.0, 0.0, -4.5 );
scene << Rotate( -85, 1, 0, 0 );
scene << Rotate( 65, 0, 0, 1 );
scene << Call List( toyTop );

scene << Update; // update the display box
```

There are some cases where you want to replace the current matrix on the stack. For these cases, use the `Load Matrix` command.

```
Load Matrix(m)
```

where *m* is a 4x4 JMP matrix that is loaded onto the current matrix stack.

Similar is the `Mult Matrix` command

```
Mult Matrix(m)
```

When the `Mult Matrix` command is issued, the matrix on the top of the current matrix stack is multiplied by *m*.

The following matrices perform some simple commands.

Translation:

```
[1  0  0  x
0  1  0  y
0  0  1  z
0  0  0  1]
```

In the following rotation matrices, `c` = cos(angle) and `s`=sin(angle).

Rotation about X axis:

```
[1  0  0  0
0  c -s  0
0 -s  c  0
0  0  0  1]
```

Rotation about Y axis:

```
[c  0  s  0
 0  1  0  0
-s  0  c  0
 0  0  0  1]
```

Rotation about Z axis:

```
[c -s  0  0
 s  c  0  0
 0  0  1  0
 0  0  0  1]
```

For example, here are two equivalent (except for the translation being opposite) ways to translate and rotate a display list.

```
// first way uses matrix
gl << Push Matrix;
xt = Identity( 4 ); // translate this one left by .75
xt[1, 4] = -.75;
xr = Identity( 4 ); // rotate this one, cos needs radians, not degrees
xr[2, 2] = Cos( 3.14159 * a / 180 );
xr[2, 3] = -Sin( 3.14159 * a / 180 );
xr[3, 2] = Sin( 3.14159 * a / 180 );
xr[3, 3] = Cos( 3.14159 * a / 180 );
yr = Identity( 4 );
yr[1, 1] = Cos( 3.14159 * a / 180 );
yr[1, 3] = Sin( 3.14159 * a / 180 );
yr[3, 1] = -Sin( 3.14159 * a / 180 );
yr[3, 3] = Cos( 3.14159 * a / 180 );
zr = Identity( 4 );
zr[1, 1] = Cos( 3.14159 * a / 180 );
zr[1, 2] = -Sin( 3.14159 * a / 180 );
zr[2, 1] = Sin( 3.14159 * a / 180 );
zr[2, 2] = Cos( 3.14159 * a / 180 );

// order of multiplication matters with matrices
gl << Mult Matrix( xt * xr * yr * zr );
gl << Arcball( dl, 1 );
gl << Pop Matrix;

// second way uses functions
gl << Push Matrix;
gl << Translate( .75, 0, 0 ); // translate this one right by .75
gl << Rotate( a, 1, 0, 0 ); // rotate this one in degrees
gl << Rotate( a, 0, 1, 0 ); // order of operations also matters here
gl << Rotate( a, 0, 0, 1 );
```

```
gl << Arcball( dl, 1 );
gl << Pop Matrix;
```

It is not possible to read back the current transform matrix, because the matrix exists only while the display list is drawing, not while your JSL script is creating it. If you must know its content, create it in JSL and use `Load Matrix` to put it on the stack.

Lighting and Normals

The following methods enable you to add lighting, materials, and normal vectors to your shapes. Using these methods, models can appear shiny or light-absorbing.

- "Creating Light Sources"
- "Lighting Models"
- "Normal Vectors"
- "Shading Model"
- "Material Properties"
- "Alpha Blending"
- "Fog"
- "Example"

Creating Light Sources

Light sources are specifications of a color, position, and direction. JSL allows for up to eight lights (numbered 0 to 7) defined by the `Light` command, where *n* is the number of the light.

```
Light( n, argument, value, ... value )
```

Note: To turn each light on, issue an `Enable(`*`Lighting`*`)` and an `Enable(light`*`n`*`)` command, where *n* is the light number. Then, move the light to a position in the scene with a `Light(`*`n`*`, `*`POSITION`*`, `*`x`*`, `*`y`*`, `*`z`*`)` command.

The value of *argument* can be any one of those shown in Table 13.3. The table shows default values for each argument.

Table 13.3 Light Arguments and Default Values

Argument	Default Value	Meaning
AMBIENT	(0, 0, 0, 1)	Ambient RGBA intensity
DIFFUSE	(1, 1, 1, 1)	diffuse RGBA intensity
SPECULAR	(1, 1, 1, 1)	specular RGBA intensity
POSITION	(0, 0, 1, 0)	(x, y, z, w) position
SPOT_DIRECTION	(0, 0, -1)	(x, y, z) direction of spotlight
SPOT_EXPONENT	0	spotlight exponent
SPOT_CUTOFF	180	spotlight cutoff angle
CONSTANT_ATTENUATION	1	constant attenuation factor
LINEAR_ATTENUATION	0	linear attenuation factor
QUADRATIC_ATTENUATION	0	quadratic attenuation factor

Note: The default values for DIFFUSE and SPECULAR in this table apply only to Light 0. For other lights, the default value is (0, 0, 0, 1) for both arguments.

The first three arguments (AMBIENT, DIFFUSE, and SPECULAR) are used to color the light. DIFFUSE is the argument that is most closely associated with the physical color of the light. AMBIENT refers to the property of the light when it functions as a background light. SPECULAR alters the way a light is reflected off a surface.

Specify the position of the light using the POSITION argument. Nonzero values of the fourth (w) coordinate position the light in homogenous object coordinates.

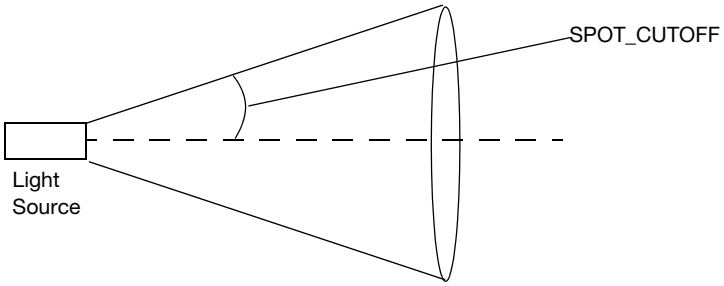
Light in the real-world decreases in intensity as distance from the light increases. Since a directional light is infinitely far away, it does not make sense to attenuate its intensity as a function of distance. However, JSL attenuates a light source by multiplying the contribution of the source by an attenuation factor

$$\text{attenuation factor} = \frac{1}{c + ld + qd^2}$$

where c = CONSTANT_ATTENUATION, l = LINEAR_ATTENUATION, and q = QUADRATIC_ATTENUATION.

To create a spotlight, limit the shape of the light to a cone. Use the `SPOT_CUTOFF` argument to define the side of the cone, as shown in the following illustration.

Figure 13.16 Spotlight



In addition to the cutoff angle, you can control the intensity and direction of the light distribution in the cone. `SPOT_DIRECTION` specifies the direction for the spotlight to point; `SPOT_EXPONENT` influences how concentrated the light is.

Lighting Models

Lighting models are specified with the `Light Model` command.

```
Light Model( argument, value,...,value )
```

Light models specify three attributes of lights.

- The global ambient light intensity
- Whether the viewpoint is local or is an infinite distance away
- Whether lighting calculations should be performed differently for the front and back faces of objects.

Table 13.3 shows the three valid arguments for the `Light Model` command.

Table 13.4 Light Model Arguments and Default Values

Argument	Default Value	Meaning
`LIGHT_MODEL_AMBIENT`	(0.2, 0.2, 0.2, 1)	Ambient RGBA intensity of the entire scene
`LIGHT_MODEL_LOCAL_VIEWER`	0 (false)	how specular reflection angles are computed
`LIGHT_MODEL_TWO_SIDE`	0 (false)	nonzero values imply two-sided lighting

Normal Vectors

Normal vectors point in a direction perpendicular to a surface. For a plane, all normals are the same. For a more complicated surface, normals are more complicated. JSL enables you to specify the normal vector for each vertex. These normals specify the orientation of the surface in space, necessary for lighting calculations. Accurate normals assure accurate lighting.

The normal vector is of length 1 and is perpendicular to the vertex. Typically, a vertex is shared between several polygons. A smooth shaded effect is desired, so the perpendicular at the vertex is calculated as a (possibly weighted) average of the polygon's normals. It is important to calculate the "outward" normal for polygons unless two-sided shading is enabled because only the outer face of the polygon is illuminated. With a scaled polygon, the normal's length is not 1 after scaling, and the lighting is wrong.

Normal vectors are set at the same time the surface is constructed, and are specified with the `Normal` command. Use the `Enable(NORMALIZE)` command to have the normals re-normalized to 1 each time the scene is drawn.

Shading Model

The shading model of a polygon is set using the `Shade Model` command.

```
Shade Model (mode)
```

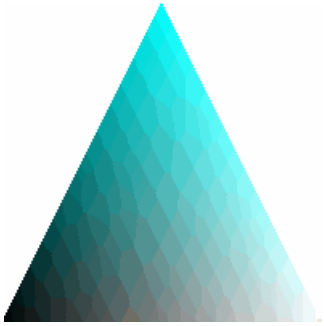
where *mode* can be `SMOOTH` (the default) or `FLAT`. `SMOOTH` shading interpolates the colors of the primitive from one vertex to the next. `FLAT` mode duplicates the color of one vertex across the entire primitive.

The following script changes the color at each of a triangle's vertices. The `FILL` shade model interpolates the color of the interior automatically.

```
scene = Scene Box( 200, 200 ); // make a scene box...holds an OpenGL scene.

New Window( "Shade Model", scene ); // put the scene in a window
scene << Clear;
scene << Ortho2D( -1, 1, -1, 1 );

scene << Shade Model( SMOOTH );
scene << Polygon Mode( FRONT_AND_BACK, FILL );
scene << Begin( TRIANGLES );
scene << Color( 0, 0, 0 ); // black
scene << Vertex( -1, -1, 0 );
scene << Color( 0, 1, 1 ); // cyan
scene << Vertex( 0, 1, 0 );
scene << Color( 1, 1, 1 ); // white
scene << Vertex( 1, -1, 0 );
scene << End();
```

```
scene << Update;
```

Figure 13.17 Shading



Material Properties

To set the material properties of a surface, use the `Material` command.

```
Material( face, argument, value,...value )
```

face can be `Front`, `Back`, or `Front_and_back`. (Note that the material properties can be set separately for the front and back faces of a polygon.)

Table 13.5 shows the arguments and default values for `Material` arguments.

Table 13.5 Material Arguments and Default Values

Argument	Default Value	Meaning
`AMBIENT`	(0.2, 0.2, 0.2, 1.0)	Ambient color of material
`DIFFUSE`	(0.8, 0.8, 0.8, 1.0)	Diffuse color of material
`AMBIENT_AND_DIFFUSE`		Both `AMBIENT` and `DIFFUSE`
`SPECULAR`	(0.0, 0.0, 0.0, 1.0)	Specular color of material
`SHININESS`	0	Specular exponent that can range from 0 to 128.
`EMISSION`	(0, 0, 0, 1)	Emissive color of material

Alpha Blending

The BlendFunc command allows for alpha blending. To use it, send a BlendFunc message to a scene, for example:

```
scene << BlendFunc(SRC_ALPHA, ONE_MINUS_SRC_ALPHA)
```

SRC_ALPHA and ONE_MINUS_SRC_ALPHA are OpenGL constants that tell BlendFunc to use alpha to blend against the existing display buffer. Disabling *z*-buffer testing or rendering primitives from back to front might be needed for some applications. By default, the *z*-buffer tests prevent anything from drawing behind a transparent polygon after it is drawn.

More information about all the constants available to BlendFunc (many of which are not useful to the JSL programmer) are available in the OpenGL documentation at opengl.org.

Fog

Fog enables figures to fade into the distance, making for more realistic models. All types of geometric figures can be fogged. To turn fog on, enable the FOG argument.

```
Enable (FOG)
```

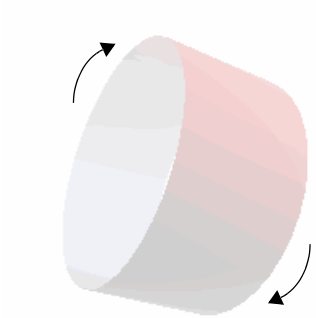
Example

The following example uses several of the concepts presented in this section, including lighting, fog, and normalization. It draws a spinning cylinder that is affected by two lights.

```
scene = Scene Box( 300, 300 ); // make a scene box
New Window( "Cylinder", scene ); // put the scene in a window

For( i = 1, i < 360, i++,
  scene << Clear;

  // the lens is 45 degrees, near is 3 units from the camera, far is 7.
  scene << Perspective( 50, 1, 10 );

  // move the world so that 0,0,0 is visible in the camera
  scene << Translate( 0.0, 0.0, -2 );
  scene << Enable( Lighting );
  scene << Enable( Light0 );
  scene << Enable( Light1 );
  scene << Light( Light0, POSITION, 1, 1, 1, 1 ); // near viewer
  scene << Light( Light0, DIFFUSE, 1, 0, 0, 1 ); // red light
  scene << Light( Light1, POSITION, -1, -1, -1, 1 ); // behind object
  scene << Light( Light1, DIFFUSE, .5, .5, 1, 1 ); // blue-gray light
  scene << Enable( Fog );
```

```
    scene << Enable( NORMALIZE );

    scene << Rotate( i, 1, 0, 0 );
    scene << Rotate( i * 3, 0, 1, 0 );
    scene << Rotate( i * 3 / 2, 0, 0, 1 );
    scene << Cylinder( 0.5, 0.5, 0.5, 40, 10 );
    scene << Update;
    Wait( 0.01 );
);
```

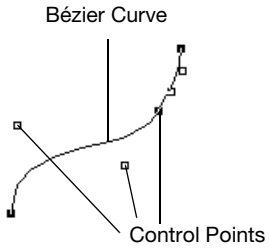
Figure 13.18 Fog



Bézier Curves

A complete discussion of Bézier curves is beyond the scope of the *Scripting Guide*. JSL has several commands for defining and drawing curves and their associated meshes.

Figure 13.19 Bézier Curve



One-Dimensional Evaluators

To define a one-dimensional map, use the `Map1` command.

```
Map1( target, u1, u2, stride, order, matrix )
```

The *`target`* argument defines what the control points represent. Values of the *`target`* argument are shown in Table 13.6. Note that you must use the `Enable` command to enable the argument.

Table 13.6 Map1 *`Target`* Arguments and Default Values

`target` Argument	Meaning
`MAP1_VERTEX_3`	(x, y, z) vertex coordinates
`MAP1_VERTEX_4`	(x, y, z, w) vertex coordinates
`MAP1_INDEX`	color index
`MAP1_COLOR_4`	R, G, B, A
`MAP1_NORMAL`	normal coordinates
`MAP1_TEXTURE_COORD_1`	s texture coordinates
`MAP1_TEXTURE_COORD_2`	s, t texture coordinates
`MAP1_TEXTURE_COORD_3`	s, t, r texture coordinates
`MAP1_TEXTURE_COORD_4`	s, t, r, q texture coordinates

The second two arguments (*`u1`* and *`u2`*) define the range for the map. The *`stride`* value is the number of values in each block of storage. In other words, the value is the offset between the beginning of one control point and the beginning of the next control point. The *`order`* should equal the degree of the curve plus one. The *`matrix`* holds the control points.

For example, `Map1( MAP1_VERTEX_3, 0, 1, 3, 4, <4x3 matrix> )` is typical for setting the two end points and two control points to define a Bézier line.

You use the `MapGrid1` and `EvalMesh1` commands to define and apply an evenly spaced mesh.

```
MapGrid1( un, u1, u2 )
```

sets up the mesh with *`un`* divisions spanning the range *`u1`* to *`u2`*. Code is simplified by using the range 0 to 1.

```
EvalMesh1( mode, i1, i2 )
```

actually generates the mesh from *i1* to *i2*. The *mode* can be either POINT or LINE. The EvalMesh1 command makes its own Begin and End clause.

The following example script demonstrates a one-dimensional outlier. A random set of control points draws a smooth curve. Only the first and last points are on the curve. Using NPOINTS=4 results in a cubic Bézier spline.

```
boxwide = 500;
boxhigh = 400;

gridsize = 100; // bigger for finer divisions

/* We suggest you use only values between 2 and 8 (inclusively). Numbers
   beyond these might be interpreted differently, depending on implementation.
   This value is the degree+1 of the fitted curve */
NPOINTS = 4;

point = J( NPOINTS, 3, 0 );

// create an array of x,y,z triples
For( x = 1, x <= NPOINTS, x++,
   point[x, 1] = (x - 1) / (NPOINTS - 1) - .5;
      // x from -.5 to +.5
   point[x, 2] = Random Uniform() - .5; // y is random in same range

   // z is always zero, which causes the curve to stay in a plane
   point[x, 3] = 0;
);

spline = Scene Box( boxwide, boxhigh );

// data from -.5 to .5 in x and y; this is a little larger
spline << Ortho( -.6, .6, -.6, .6, -2, 2 );

spline << Enable( MAP1_VERTEX_3 );
spline << MapGrid1( gridsize, 0, 1 );
spline << Color( .2, .2, 1 ); // blue curve

spline << Map1( MAP1_VERTEX_3, 0, 1, 3, NPOINTS, point );
spline << Line Width( 2 ); // not-so-skinny curve
spline << EvalMesh1( LINE, 0, gridsize ); // also try LINE, POINT

spline << Color( .2, 1, .2 );
spline << Point Size( 4 ); // big fat green points

// show the points and label them
For( i = 1, i <= NPOINTS, i++,
```

```
    spline << Begin( "POINTS" );
    spline << Vertex( point[i, 1], point[i, 2], point[i, 3] );
    spline << End;
    spline << Push Matrix;
    spline << Translate( point[i, 1], point[i, 2], point[i, 3] );
    spline << Text( center, bottom, .05, Char( i ) );
    spline << Pop Matrix;
);

New Window( "Spline", spline );
```

https://www.tinaja.com/glib/bezconn.pdf offers an explanation of connecting cubic segments so that both the slope and the rate of change match at the connection point. This example does not illustrate doing so; there is only one segment here.

Two-Dimensional Evaluators

Two-dimensional evaluators follow their one dimensional counterparts, and are used in a similar way.

```
Map2(target, u1, u2, ustride, uorder, v1, v2, vstride, vorder, matrix)
Eval Coord2(u, v)
```

Values for the *target* argument are the same as those shown in Table 13.6 with *Map1* replaced with *Map2* appropriately. The *u1, u2, v1,* and *v2* values specify the range of the two-dimensional mesh.

For example, `Map2(MAP2_VERTEX_3, 0, 1, 3, 4, 0, 1, 12, 4, <16x3 matrix>)` is typical for setting the 16 points that define a Bézier surface.

Use the `MapGrid2` and `EvalMesh2` commands to define and apply an evenly spaced mesh.

```
MapGrid2(un, u1, u2, vn, v1, v2)
```

sets up the mesh with *un* and *vn* divisions spanning the range *u1* to *u2* and *v1* to *v2*. Code is simplified by using ranges that span 0 to 1.

```
EvalMesh2(mode, i1, i2, j1, j2)
```

actually generates the mesh from *i1* to *i2* and *j1* to *j2*. The *mode* can be `POINT`, `LINE`, or `FILL`. The `EvalMesh2` command makes its own `Begin` and `End` clause.

Using the Mouse

Mouse activity is supported through two feedback functions. The Patch Editor.jsl sample script uses these functions to support the dragging and dropping of points. Part of that script, the call-back function for mouse activity, is explained below. To run the script, open PatchEditor.jsl in the Scene3D folder inside the JMP Sample Scripts folder.

```
topClick2d = Function( {x, y, m, k},
   dragfunc( x, boxhigh - y, m, 1, 2 );
   1;
);
frontClick2d = Function( {x, y, m, k},
   dragfunc( x, boxhigh - y, m, 1, 3 );
   1;
);
rightClick2d = Function( {x, y, m, k},
   dragfunc( x, boxhigh - y, m, 2, 3 );
   1;
);

Click3d = Function( {x, y, m, k, hitlist},
   If( m == 1,
      If( N Items( hitlist ) > 0,
         CurrentPoint = hitlist[1][3], /* first matrix in the list is the
   closest; 3rd element of matrix is ID*/
         CurrentPoint = 0
      );
      makePatch();
   );
   0; /* cares only about initial mouse down. return 1 if drag, release is
   needed, but then arcball does not happen. */
);

/* after one of the 3 Click2d functions figures out which axis of the model is
   represented by the screen X, Y, pass in to this common code */
dragfunc = Function( {x, y, m, ix, iy}, /* ix, iy are the index of the X, Y,
   or Z part of the coord in the points matrix */
   If( CurrentPoint > 0,
      points[CurrentPoint, ix] = (x / boxwide) * (orthoright - ortholeft) +
   ortholeft;
      points[CurrentPoint, iy] = (y / boxhigh) * (orthotop - orthobottom) +
   orthobottom;
      makepatch();
   )
);
```

When a 2-D function is called, the arguments are X, Y, M, K.

- X and Y are the coordinates of the mouse.
- M shows the state of the mouse and button. M=0 says that the mouse button is up. M=1 says that the button was just pressed. M=2 says that the button is down and the mouse is moving. M=3 says that the button was just released.
- K is related to the keys Shift, Alt, and Ctrl. K=1 for the Shift key, K=2 for the Ctrl (Command) key, and K=3 for the Alt (Option) key.

The 3-D function is called similarly. The arguments are X, Y, M, K, `hitlist`, where `hitlist` is a list of matrices

```
[znear, zfar, id1, id2, id3, ...]
```

`znear`, `zfar` is the Z distance from the camera of the near and far edge of the object. The matrices are sorted from near to far by the midpoint of `znear`, `zfar`. The `id`s in the list are the `pushname`, `loadname`, and `popname` values you just put in the display list.

The drag functions use a return value to tell if mouse processing should continue. That is the trailing "1" you see in the functions. Anything else stops the mouse tracking. This is needed because the 2-D and 3-D functions do not run in parallel. You might want the 2-D to return 0 and the 3-D to return 1 so the tracking would happen in 3-D rather than 2-D.

Pick Commands

This `SceneBox` callback gets the 2-D mouse coordinates, and then uses `Pick` to determine the "named" object under the mouse. For example, `hitlist` is a 5x5 pixel pick box around `x,y`; up to 1000 items returned, but just the leaf names. The format of the return is determined by the last argument (1 returns a simple array, 0 returns a sorted (by depth) list of arrays).

```
Track2d=function({x, y, m, k},
hitlist = theSceneBox<<pick( x, y, 5, 5, 1000, 1 );
  if ( nrow(hitlist) > 0, // something IS in the pick box
     ... hitlist[1..n] // are names that you put in the display list
  ) );
```

Contrast this with a call back `Track3d` function, where the pick rectangle is always 1x1 and picking only happens when the mouse moves. This is almost always what you want, but points are difficult to pick because the 1x1 pick area is the same small size as the point. This function lets you pick without a mouse move.

The `Track3d` function always provides a depth-sorted list of arrays; each array can describe multiple names in a hierarchy (pushname and popname construct a hierarchy of objects). The sorting can be very slow when thousands of objects are selected. The final argument (1, above) controls whether the pick function replaces the sorted list of arrays with a simple array. The simple array contains only the "leaf" names, not higher level names.

Three-Dimensional Arguments

Arguments enable you to specify special modes and settings. To enable an argument, use the `Enable(argument)` command. To disable an argument, use the `Disable(argument)` command. Available arguments are shown in Table 13.7.

Table 13.7 Arguments

ALPHA_TEST	LIGHT5	MAP2_TEXTURE_COORD_3
AUTO_NORMAL	LIGHT6	MAP2_TEXTURE_COORD_4
BLEND	LIGHT7	MAP2_VERTEX_3
CLIP_PLANE0	LIGHTING	MAP2_VERTEX_4
CLIP_PLANE1	LINE_SMOOTH	NORMALIZE
CLIP_PLANE2	LINE_STIPPLE	POINT_SMOOTH
CLIP_PLANE3	MAP1_COLOR_4	POLYGON_OFFSET_FILL
CLIP_PLANE4	MAP1_INDEX	POLYGON_OFFSET_LINE
CLIP_PLANE5	MAP1_NORMAL	POLYGON_OFFSET_POINT
COLOR_LOGIC_OP	MAP1_TEXTURE_COORD_1	POLYGON_SMOOTH
COLOR_MATERIAL	MAP1_TEXTURE_COORD_2	POLYGON_STIPPLE
CULL_FACE	MAP1_TEXTURE_COORD_3	SCISSOR_TEST
DEPTH_TEST	MAP1_TEXTURE_COORD_4	STENCIL_TEST
DITHER	MAP1_VERTEX_3	
FOG	MAP1_VERTEX_4	
LIGHT0	MAP2_COLOR_4	
LIGHT1	MAP2_INDEX	
LIGHT2	MAP2_NORMAL	
LIGHT3	MAP2_TEXTURE_COORD_1	
LIGHT4	MAP2_TEXTURE_COORD_2	

Improve the Display of Graphics

Hardware acceleration determines how fast and smoothly graphics are drawn. On Windows, hardware acceleration is turned off by default.

Note: JMP uses OpenGL 1.1 to perform 3-D rendering.

If you need OpenGL and have the hardware support, turn on hardware acceleration in one of the following ways:

- Right-click the graph and select **Use Hardware Acceleration**.
- Send the `Use Hardware Acceleration(1)` message to the scene box.

- In your script, set the global variable `Scene3DHardwareAcceleration = 1;` If you want hardware acceleration to be the default across runs of JMP, add it to your jmpStartAdmin.jsl file. Because it is a global variable, it will remain in effect until changed.

Chapter 14

Extending JMP

External Data Sources, Analytical Tools, and Automation

This chapter discusses scripting features that are particularly useful for production settings, such as the following:

- a data feed for capturing real-time data from a laboratory instrument
- using SAS, MATLAB, R, or Python through JMP Scripting Language (JSL)
- using JMP with Microsoft Excel
- connecting to databases
- controlling JMP externally by OLE automation
- parsing XML
- publishing to JMP Public or JMP Live
- communicating with REST web services
- communicating with web APIs

Some general JSL commands that might be of particular interest for use in a production setting include `Caption`, `Speak`, `Print`, `Write`, and `Mail`. These commands are described in "Functions that Communicate with Users".

Contents

Real-Time Data Capture on Windows . . . 797
Create a Data Feed Object . . . 797
Read in Real-Time Data . . . 799
Manage a Data Feed with Messages . . . 800
Examples of Data Feed . . . 802
Dynamic Link Libraries (DLLs) . . . 804
Use Sockets in JSL . . . 806
Socket-Related Commands . . . 807
Messages for Sockets . . . 808
Database Access . . . 809
Run Query Builder Queries . . . 809
Open Database Function . . . 810
Creating a Database Connection and Executing SQL . . . 812
Write a SQL Query . . . 813
Work with SAS . . . 814
Make a SAS DATA Step . . . 814
Create SAS DATA Step Code for Formula Columns . . . 814
SAS Variable Names . . . 815
Connect to a SAS Metadata Server . . . 815
Get the Values of SAS Macro Variables . . . 818
Preferences . . . 819
Sample Scripts . . . 819
Work with MATLAB . . . 821
Install MATLAB . . . 821
Work with R . . . 822
Installing R . . . 822
JMP to R Interfaces . . . 824
R JSL Scriptable Object Interfaces . . . 825
Conversion Between JMP Data Types and R Data Types . . . 825
Troubleshooting . . . 828
Examples . . . 829
Work with Python . . . 831
Install Python . . . 831
Troubleshooting the JMP Python Integration . . . 832
JMP-to-Python Interfaces . . . 836
Equivalent Data Types for Python Send() . . . 836
Equivalent Data Types for Python Get() . . . 837
JMP Scoping Operators and Python . . . 838
Python Name() and Python Send() Examples . . . 839

Additional Python Integration Examples . . . 840
Work with Microsoft Excel . . . 842
Parse XML . . . 842
OLE Automation . . . 844
Share Reports on JMP Live . . . 845
JMP Live Prerequisites . . . 845
Get an API Key . . . 845
Publish a Single Report to JMP Live . . . 846
Publish Several Reports to a JMP Live Folder . . . 847
Delete Reports in a JMP Live Folder . . . 847
Replace the Contents of a JMP Live Report . . . 848
Search for a JMP Live Report or Folder ID . . . 848
Update Data in JMP Live Reports . . . 849
Work with Other Content in JMP Live Reports . . . 850
HTTP Requests in JMP Live . . . 850
Communicate with REST Web Service . . . 850
Examples of HTTP Requests . . . 850
OAuth 2.0 for Web APIs . . . 853
Security . . . 853
Using Postman for Redirect URLs . . . 853
Authorization Code Grant . . . 854
Implicit Grant . . . 855
Resource Owner Grant . . . 855
Using a Token . . . 856
OAuth 2.0 Resources . . . 856
Communicate with SAS Cloud Analytic Services . . . 856
About CAS Actions . . . 857
Connect to the CAS Server . . . 858
Create and Submit a CAS Action . . . 859
Create and Submit a CAS DATA Step Action . . . 859
Create a New CAS Server . . . 859
Import Data from the CAS Server . . . 859
View a CAS Dataset . . . 860
Get a List of Available CAS Libraries . . . 860
Export Data from a Data Table to a CAS Server . . . 860
Convert a CAS Table to a Data Table . . . 860
Delete a CAS Table . . . 863
Remove a CAS Table . . . 863
Verify the CAS Server Connection . . . 863
Get Details about the Current Connection . . . 863

Disconnect from the CAS Server 863
Terminate All CAS Sessions 863

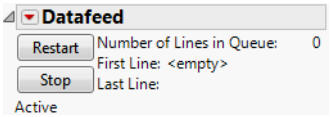
Real-Time Data Capture on Windows

A data feed is a real-time method to read data continuously, such as from a laboratory measurement device connected to a serial port. A data feed object sets up a concurrent thread with a queue for input data lines that arrive in real time and are processed with background events. You set up scripts to interpret the data lines and push the data to data tables, or do whatever else your process requires.

For example, submit this to get records from com1: and list them in the log.

```
feed = Open Datafeed(
  Connect( Port( "com1:" ), Baud( 9600 ), DataBits( 7 ) ),
  Set Script( Print( feed << GetLine ) )
);
```

Figure 14.1 A Datafeed Window Shows the Status and Offers Controls



Create a Data Feed Object

To create a data feed object, use the `Open Datafeed()` function with arguments specifying details about the connection:

```
feed = Open Datafeed( options );
```

No arguments are required. You can simply create the data feed object and then send messages to it. Messages might include connecting to a port or setting up a script to process the data coming in. However, you would typically set up the basic operation of the data feed in the `Open Datafeed()` function and subsequently send messages as needed to manage the data feed. Any of the options below work both as options inside `Open Datafeed()` or as messages sent to a data feed object.

It's a good idea to store a reference to the object in a global variable, such as *feed* above, so that you can easily send messages to the object. You can store a reference to an existing object by using a subscript; for example, to store a reference to the second data feed created:

```
feed2 = Datafeed[2];
```

Data Feed Options

To connect to a live data source, use `Connect( )` and specify details for the port. Each setting takes only one argument; in this syntax summary the symbol | between argument choices means "or." The `Port` specification is needed if you want to connect, otherwise the object still works, but is not connected to a data feed. The last three items, `DTR_DSR`, `RTS_CTS`, and `XON_XOFF`, are Boolean to specify which control characters are sent back and forth to indicate when the data feed is ready to get data. Typically, you would turn on at most one of them.

```
feed = Open Datafeed(
  Connect(
     Port( "com1:" | "com2:" | "lpt1:" | ... ),
     Baud( 9600 | 4800 | ... ),
     Data Bits( 8 | 7 ),
     Parity( None | Odd | Even ),
     Stop Bits( 1 | 0 | 2 ),
     DTR_DSR( 0 | 1 ), // Data Terminal Ready
     RTS_CTS( 0 | 1 ), // Request To Send | Clear To Send
     XON_XOFF( 1 | 0 ) // Transmitter On | Transmitter Off
  )
);
```

This command creates a scriptable data feed object and stores a reference to it in the global variable *feed*. The `Connect()` argument starts up a thread to watch a communications port and collect lines. The thread collects characters until it has a line, and then appends it to the line queue and schedules an event to call the script.

Note: A line of data from the feed might generate multiple rows in a data table or might be a fraction of one row. You provide JSL to parse the data lines and to add rows to a data table as needed.

`Set Script()` attaches a script to the data feed object. This script is triggered by the `Open Datafeed` message whenever a line of data arrives. The argument for `Set Script()` is simply the script to run, or a global containing a script.

```
feed = Open Datafeed( Set Script( myScript ) );
feed = Open Datafeed( Set Script( Print( feed << Get Line ) ) );
```

A data feed script typically uses `Get Line` to get a copy of one line and then does something with that line. Often it parses the line for data and adds it to some data table.

Read in Real-Time Data

The term *live data feed* describes the way an external data source sends information via a physical or a logical communication link to another device. You can connect JMP to a live data feed through the serial port of your Windows computer to read a stream of incoming data in real time. Remember the following:

- The data feed must come through a standard nine-pin serial port. Data cannot be read through a USB port unless there is a driver that can simulate a serial port.
- You need to know the exact baud rate, parity, stop bits, and data bits for the attached device.

Once you obtain the numbers for your device, enter them into the `Open Datafeed()` command in the script below. (The `4800`, `even`, `2`, and `7` in the script below are examples, so replace them with your information). Then connect the data feed to your computer and open and run the script:

```
streamScript = Expr(
   line = feed << Get Line;
   Show( line );
   len = Length( line );
   Show( len );
   If( Length( line ) >= 1,
      Show( "Hi" );
      Show( line );
      field = Substr( line, 5, 8 );
      Show( field );
      x = Num( field );
      Show( x );
      If( !Is Missing( x ),
         dt << Add Row( {:Column1 = x} );
         Show( x );
      );
   );
);
feed = Open Datafeed(
   Baud Rate( 4800 ),
   parity( even ),
   Stop bits( 2 ),
   Data bits( 7 )
);
feed << Set Script( streamScript );
feed << Connect;
```

To ensure harmony between the communications settings for JMP and the instrument reading data from an external source, select **File > Preferences > Communications**. Refer to the documentation for your instrument to find the appropriate settings.

Manage a Data Feed with Messages

A data feed object responds to several messages, including `Connect` and `Set Script`. These are detailed above as arguments for `Open Datafeed()`. They can also be sent as messages to a data feed object that already exists:

```
feed << Connect( port( "com1:" ), baud( 4800 ), databits( 7 ), parity( odd ),
  stopbits( 2 ) );
feed << Set Script( myScript );
```

The following messages could also be used as arguments to `On Datafeed()`. However, it would be more common to send them as messages to a data feed object that is already present.

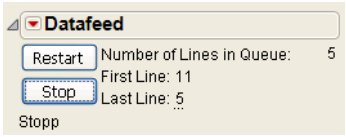
You can send lines to a data feed from a script. This is a quick way to test a data feed. If you do not have a device available, `Queue Line` does not send data to a device; it simulates getting data from a device. Include a text argument or a global that stores text:

```
feed << Queue Line( "14" );
feed << Queue Line( myValue );
```

Here is a test script to queue five lines of data:

```
feed << Queue Line( "11" );
feed << Queue Line( "22" );
feed << Queue Line( "33" );
feed << Queue Line( "44" );
feed << Queue Line( "55" );
```

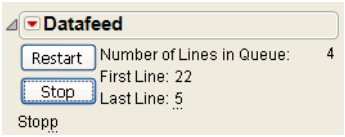
Figure 14.2 Datafeed: 5 Lines Queued



To get the first line currently waiting in the queue, use a `Get Line` (singular) message. When you get a line, it is removed from the queue. Five lines were queued with the test script above, and `Get Line` returns the first line and removes it from the queue:

```
feed << Get Line
   "11"
```

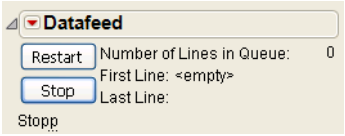
Figure 14.3 Datafeed: 4 Lines Queued



To empty all lines from the queue into a list, use `Get Lines` (plural). This returns the next four lines from the test script in list `{ }` format.

```
myList = feed << GetLines;
   { "22", "33", "44", "55" }
```

Figure 14.4 Datafeed: 0 Lines Queued



To stop and later restart the processing of queued lines, either click the **Stop** and **Restart** buttons in the Datafeed window, or send the equivalent messages:

```
feed << Stop;
feed << Restart;
```

To close the data feed and its window:

```
feed << Close;
```

To disconnect from the live data source:

```
feed << Disconnect
```

Table 14.1 Datafeed Messages

Message	Syntax	Explanation
`Open Datafeed` `Datafeed`	`feed = Open Datafeed( commands )`	Creates a data feed object. Any of the following can be used as commands inside `Open Datafeed` or sent as messages to an existing data feed object. `Datafeed` is a synonym.
`Set Script`	`feed << Set Script( script )`	Assigns the *`script`* that is run each time a line of data is received.
`Get Line`	`feed << Get Line`	Returns and removes one line from the data feed queue.

Table 14.1 Datafeed Messages *(Continued)*

Message	Syntax	Explanation
Get Lines	`feed << Get Lines`	Returns as a list and removes all lines from the data feed queue.
Queue Line	`feed << Queue Line( string )`	Sends one line to the end of the data feed queue.
Stop	`feed << Stop`	Stops processing queued lines.
Restart	`feed << Restart`	Restarts processing queued lines.
Close	`feed << Close`	Closes the data feed object and its window.
Connect	`feed << Connect( Port( "com1:" | "lpt1:" | ... ), Baud( 9600 | 4800 | ... ), Data Bits( 8 | 7 ), Parity( None | Odd | Even ), Stop Bits( 1 | 0 | 2 ), DTR_DSR( 0 | 1 ), RTS_CTS( 0 | 1 ), XON_XOFF( 1 | 0 ) )`	(Windows only) Sets up port settings for the connection to the device. The symbol | between arguments represents "or"; choose one argument for each setting.
Disconnect	`feed << Disconnect`	(Windows only) Disconnects the device from the data feed queue but leaves the data feed object active.

Examples of Data Feed

Reading Data

Here is a typical data feed script. It expects to find a string 3 characters long starting in column 11. If it does, it uses it as a number and then adds a row to the data table in the column "thickness."

```
feed = Open Datafeed( );
myScript = Expr(
   line = feed << Get Line;
   If( Length( line ) >= 14,
      x = Num( Substr( line, 11, 3 ) );
```

```
        If( !Is Missing( x ),
           dt << Add Row( {thickness = x} )
        );
    );
);
```

Assign the script to the data feed object by using `Set Script`:

```
feed << Set Script( myScript );
```

Set Up a Live Control Chart

Here is a sample script that sets up a new data table and starts a control chart based on the data feed.

```
dt = New Table( "Gap Width" ); // make a data table with one column
dc = dt << New Column( "gap", Numeric, Best );

// set up control chart properties
dc << Set Property(
  "Control Limits",
  {XBar( Avg( 20 ), LCL( 19.8 ), UCL( 20.2 ) )}
);
dc << Set Property( "Sigma", 0.1 );

// make the data feed
feed = Open Datafeed();
feedScript = Expr(
  line = feed << Get Line;
  z = Num( line );
  Show( line, z ); // if logging or debugging
  If( !Is Missing( z ),
     dt << Add Row( {:gap = z} )
  );
);
feed << Set Script( feedScript );

// start the control chart
Control Chart Builder(
       Show Capability( 0 ),
       Variables( Y( :gap ) ),
       Set Subgroup Size( 5 )
);
```

```
/* Either start the feed from the device or test-feed some data
to see it work (comment out one of the lines):
feed << Connect( Port( "com1:" ), Baud( 9600 ) );*/
For( i = 1, i <= 20, i++,
  feed << Queue Line( Char( 20 + Random Uniform() * .1 ) );
  Wait( .1 ); // simulate a delay in the feed
);
```

Store the Script in a Data Table

You can further automate the production setting by placing a data feed script such as the one above in an `On Open` data table property. A property with this name is run automatically each time the table is opened (unless you set a preference to suppress execution). If you save such a data table as a template, opening the template runs the data feed script and creates a new data table.

Dynamic Link Libraries (DLLs)

Note: 64-bit JMP cannot load 32-bit DLLs. You must recompile a 32-bit DLL for JMP to be able to load it.

You can extend JMP functionality by using JMP Scripting Language (JSL) to load a DLL and call functions exported by that DLL. There is one JSL command and six messages that implement this functionality.

```
dll_obj = Load DLL("path" <, AutoDeclare(Boolean | Quiet | Verbose) |Quiet |
  Verbose > )
```

`Load DLL()` loads the DLL in the specified *`path`*. Use the `AutoDeclare(Quiet)` argument to suppress log window messaging.

Use the `Declare Function` message to declare a function that is defined in the DLL.After you declare the function, you can call it.

```
dll_obj <<Declare Function("name", Convention(named_argument),
  Alias("string"), Arg(type, "string"), Returns(type), other_named_arguments)
```

The `Alias` defines an alternate name that you can use in JSL. For example, if you declared `Alias("MsgBox")` for a function that is named `"Message Box"` in the DLL, then you would call it as follows:

```
result = dll_obj <<MsgBox(...)
```

Here are the named arguments for `Convention`:

- STDCALL or PASCAL

- CDECL

The type argument for both `Arg` and `Returns` can be one of the following:

Table 14.2 Types for Arg and Returns

`Int8`	`UInt8`	`Int16`	`UInt16`
`Int32`	`UInt32`	`Int64`	`UInt64`
`Float`	`Double`	`AnsiString`	`UnicodeString`
`Struct`	`IntPtr`	`UIntPtr`	`ObjPtr`

See the *JSL Syntax Reference* for the `Declare Function` message arguments.

Finally, use the `UnLoadDLL` message to unload the DLL:

```
dll_obj << UnLoadDLL
```

Note: Refer to the documentation for that function provided by the DLL author when you declare a function. If the argument types and calling convention declared do not match the actual function in the DLL, calling the function could cause JMP to terminate.

Example

In the following example, the DLL contains a function named `netPresentValue`, which takes four parameters. `Declare Function()` defines these parameters and the data type that the function returns so that JMP can successfully call the function.

```
xftdll = Load Dll( "c:\ExtFunctionTests\ExtFunctionTests_x64.dll" ),

Show( xftdll );

xftdll << Declare Function(
  "netPresentValue",
 Convention( CDECL ),
  Alias( "npv" ), // use this name to call the function in JMP
  Arg( Double, "discount rate" ),
  Arg( Double, "cash flows per period" ),
  Arg( Int32, "number of cash flows" ),
  Arg( Double, Array, "array of cash flow values" ),
  Returns( Double )
);

result = xftdll << npv( 0.10, 1, 10, [ 100, 100, 100, 100, 100,
```

```
    100, 100, 100, 100, 100 ]
);
```

Other DLL Messages

The `Show Functions` message sends any functions that have been declared using `Declare Function()` to the log:

```
dll_obj << Show Functions;
```

If you are writing your own DLL, you can create functions in it using JSL. The `Get Declaration JSL` message sends any JSL functions in your DLL to the log:

```
dll_obj << Get Declaration JSL;
```

Use Sockets in JSL

Another tool that can be useful in establishing a live data feed is JSL Sockets. You can create two types of sockets using JSL:

Stream Stream sockets create a reliable connection between JMP and another computer. The other computer might be running JMP, or it might be a vending machine, data collector, printer, or other device that is capable of socket communication. Some devices implement their interface as an HTTP web server.

Datagram Datagram sockets create a less reliable connection between JMP and another computer. Datagrams are connectionless and information might arrive multiple times, not at all, and out of order. A datagram connection does not include all the overhead of a stream connection that does guarantee reliability. Because datagrams are connectionless, the destination address must be supplied each time (and for the same socket that can be different each time).

Once a socket is created, it can do two things: wait for a connection from another socket, or make a connection to another socket. Here is a simple example program that makes a connection to another computer's web server to get some data:

```
tCall = Socket( );
tCall << Connect( "www.jmp.com", "80" );
tCall << Send( Char To Blob( "GET / HTTP/1.0~0d~0a~0d~0a", "ASCII~HEX" ) );
tMessage = tCall << Recv( 1000 );
text = Blob To Char( tMessage[3] );
Show( text );
tCall << Close( );
```

The first line creates a socket and gives it a reference name (`tCall`). By default, a stream socket is created. You can designate the type of socket to create with an optional argument: `socket(STREAM)` or `socket(DGRAM)`.

The second line connects the `tCall` socket to port 80 (which is generally the HTTP port) of the JMP website.

The third line sends a `GET` request to the JMP web server; this message tells the JMP web server to send the JMP home page back. The / that follows the word GET should be the path to the page to be opened. A / opens the root page.

The fourth line receives up to 1000 bytes from the JMP web server and stores a list of information in `tMessage`. Each socket call returns a list. The first element of the list is the name of the call, and the second is a text message, which might be `ok` or a longer diagnostic message. Additional elements, if present, are specific to the call. In this case, the third element in the list is the data received.

The fifth line converts the binary information received into a character string. `tMessage[3]` is the third item in the list returned by `Recv`; it is the data from the JMP web server.

The sixth line displays the data in the log.

The last line closes the socket. The web server has already closed the far end, so this socket either needs reconnecting or proper disposal (`close`).

See the Scripting Index in the Help menu for more examples. The JMP Samples/Scripts folder also contains several examples of scripts using sockets.

Socket-Related Commands

Before creating and using a socket, you might need to retrieve information about the end that you want to connect to. `GetAddrInfo( )` and `GetNameInfo( )` each takes an address argument and an optional port argument and returns a list of information. For example:

```
Print( Get Addr Info( "www.sas.com" ) );
Print( Get Addr Info( "www.sas.com", "80" ) );
Print( Get Name Info( "149.173.5.120" ) );
   {"Get Addr Info", "ok", {"PF_INET", "SOCK_0", "IPPROTO_0", "149.173.5.120",
      "0"}}
   {"Get Addr Info", "ok", {"PF_INET", "SOCK_0", "IPPROTO_0", "149.173.5.120",
      "80"}}
   {"Get Name Info", "ok", {"PF_INET", "SOCK_0", "IPPROTO_0",
      "chess.exnet.sas.com", "0"}}
```

Sometimes there can be more than one answer. In that case, the sublist might be repeated one or more times. These functions can be quite slow; you probably should not try to build a data table of every website name with it. For IPV6 compatibility, you should generally use names like “www.sas.com” rather than the numerical form of an address.

Messages for Sockets

Once you have created a socket with `Socket( )`, there are many messages that you can send to it.

connect Connects to a listening socket. Returns a list: `{"connect", "ok"}` if the connection was successful; or an error if not (for example, `{"connect", "CONNREFUSED: The attempt to connect was forcefully rejected. "}`).

close Closes the connection when you are finished with it. Returns a list (for example, `{"Close", "ok"}`).

send Sends a STREAM message to the other end of the socket.

sendto Sends a DGRAM message to the other end of the socket.

recv Receives a STREAM message. The data comes back in a list, along with some other information. `Recv` takes a required numeric argument that specifies the number of bytes to accept.

recvfrom Receives a DGRAM message.

ioctl Controls the socket's blocking behavior. By default, JMP sockets block if no data is available; the socket does not return control to the JSL program until data *is* available. This makes scripts easy to write, but not particularly robust if the remote end of the connection fails to supply the data. A socket that is set for non-blocking behavior always returns immediately, either with an `ok` return code and some data, or with a "`WOULDBLOCK: ...`" return code, which means if it were a blocking socket, it would have to wait (block progress of the next JSL statement) until data became available.

Important: Background operations that use a JSL callback avoid this issue; a socket used in a background `recv`, `recvfrom`, or `accept` is set to non-blocking and is polled during `wait` statements and when JMP is otherwise idle.

`Ioctl` returns a list. For example, `{"ioctl", "ok"}`, or `{"ioctl", "NOTCONN: The socket is not connected. "}` if the socket has not been bound (see `bind`, below) or connected already.

bind Tells the server socket what address the client socket listens on. `Bind` associates a port on the local machine with the socket. This is required before a socket can `Listen`. (See below). `Bind` is not usually used on sockets that connect; the operating system selects an unused port for you. `Bind` is needed for a server because anyone that wants to connect to the server needs to know what port is being used. A common port is `80`, the HTTP port. `Bind` returns a list. For example, `{"bind", "ok"}`, or `{"bind", "ADDRNOTAVAIL: The specified address is not available from the local machine. "}` if you try binding to a name that is not on your machine. Another socket can connect to this socket if it knows your machine name and the number.

listen Tells the server socket to listen for connections. A listening socket is listening for connections from other sockets. You need only to put the socket into listen mode once. `Accept` (see below) is used to accept a connection from another socket. `Listen` returns a list. For example, `{"listen", "ok"}`, or `{"listen", "INVAL: The socket is (or is not, depending on context) already bound to an address. or, Listen was not invoked prior to accept. or, Invalid host address. or, The socket has not been bound with Bind. "}` if your bind call did not succeed.

accept Tells the server socket to accept a connection and return a new connected socket. `Accept` returns a list stating what happened, and if successful, a new socket that is connected to the socket at the other end. For example, the returned list might be `{"Accept", "ok", "localhost", socket( )}`. In this case `localhost` is the name of the machine that just connected, and the fourth argument is a socket that you can use to `send` or `recv` a message.

getpeername Asks about the other end of a connection. `GetPeerName` returns a list with information about the other end's socket: `{"getpeername", "ok", "127.0.0.1", "4087"}`. If this is a server socket, you can discover the address and port of the client that connected. If this is a client socket, you re-discover the name and port of the server that you used in the connect request.

getsockname Asks about this end of a connection. `GetSockName` returns a list with information about this socket: `{"getsockname", "ok", "localhost", "httpd"}`. If this is a client socket, you can discover the port the operating system assigned. If this is a server socket, you already know this information from a `bind` you already made.

Database Access

Run Query Builder Queries

Query Builder is the preferred method for selecting and importing data from a SQL database without writing SQL statements. You can preview the data before importing it into a data table. Share your queries so that other users can customize and run the queries.

After you create the query, you can run the query with a JSL script. Use `Include()` to include the query in your script without opening it.

```
query = Include( "$DOCUMENTS/Airline.jmpquery" );
query << Run Foreground( );
```

Instead of running the query in the foreground, you can use `Run Background()` to run the query in the background or `Run()` to use the Query Builder preference for running queries. By default, queries run in the background.

See the *JSL Syntax Reference* for more information about query messages.

Open Database Function

JMP supports ODBC access to SQL databases through JSL with the `Open Database` function.

```
dt = Open Database(
   "Connect Dialog" | "DSN=...", // data source
   "sqlStatement" | "dataTableName" | "SQLFILE=...", // SQL statement
   Invisible,  //Optional keyword to hide the table upon importing it
   "outputTableName" // new table name
);
```

Note: The characters `$# -+/%()&|;?` in a database table name must be quoted with square brackets [].

The first argument is a quoted connection string to specify the data source. It should be either of the following:

- `"Connect Dialog"` to display the Select Data Source window (Windows) or Choose DSN window (macOS).
- `"DSN="` and then the data source name and any other information needed to connect to the data source. On Windows, the data source name is shown in the name column of the User DSN or System DSN tab of the ODBC Data Source Administrator. On macOS, the DSN is shown in the ODBC Manager or iODBC Driver Manager. The rest of the string depends on the ODBC data source.

For example:

```
"DSN=dBASE Files;DBQ=C:/Program Files/SAS/JMP/16/Samples/Import Data;"
```

The second argument is a double-quoted string that can be one of the following:

1. An SQL statement to execute. For example, the second argument might be a `SELECT` statement in a quoted string like the following:

   ```
   "SELECT AGE, SEX, WEIGHT FROM BIGCLASS"
   ```

   The SQL must conform to the SQL that the data source supports. That is, a table named "Big Class" would need to be appropriately quoted because of the space between “Big” and “Class” (if whitespace is supported). The method for quoting depends on the data source and is usually done with ", ’, ‘, or [] characters.
2. The name of a data table. In this case, the effect is an SQL `"SELECT * FROM"` statement for the data table indicated. For example, `Open Database` would in effect execute the statement `"SELECT * FROM BIGCLASS"` if you specify this for the second argument:

   ```
   "BIGCLASS"
   ```

3. "SQLFILE=" and a path to a text file containing an SQL statement to be executed. For example, with the following argument, JMP would attempt to open the file mySQLFile.txt from the C:\ directory and then execute the SQL statement in the file.

   ```
   "SQLFILE=C:\mySQLFile.txt"
   ```

The optional `Invisible` argument creates a hidden data table. Hidden data tables remain in memory until they are explicitly closed, reducing the amount of memory that is available to JMP. To close the hidden data table, call `Close(dt)`, where `dt` is the data table reference.

The optional *outputTableName* argument is optional and specifies the name of the output table to be created, if any. Note that `Open Database` does not always return a data table. The return value might be null. Whether it returns a data table depends on the type of SQL statement executed. For example, a `SELECT` statement would return a data table, but a `DROP TABLE` statement would not.

To save a table back to a database through JSL, send the data table reference a `Save Database( )` message:

```
dt << Save Database( "connectInfo", "TableName" );
```

The first argument works the same way as it does in `Open Database`. Note that some databases do not allow you to save a table over one that already exists. If you want to replace a table in a database, use a DROP TABLE SQL statement in an Open Database command:

```
Open Database ( "connectinfo", "DROP TABLE TableName" );
```

Note: JMP 13 or later can save tables with spaces and mixed case in the names if the data source supports them. Most data sources do support spaces, except Apache Hive and Apache Hadoop. Mixed case is preserved, but for the most part, SQL is case insensitive.

The following script opens a database with an SQL query, saves it back to the database under a new name, and then deletes the new table.

```
dt = Open Database(
   "Connect Dialog",
   "SELECT age, sex, weight FROM \!"Bigclass$\!"",
   "My Big Class"
);
dt << Save Database( "Connect Dialog", "MY_BIG_CLASS" );
Open Database( "Connect Dialog", "DROP TABLE BIGCLASS.MY_BIG_CLASS" );
```

Note: When you import data from an ODBC database, a table variable is added that can contain user ID and password information. To prevent this from happening, set the following preference: `pref(ODBC Hide Connection String(1))`. You can also select the **File** menu (Windows) or **JMP** menu (macOS), **Preferences > Tables,** and then select **ODBC Hide Connection String**. See *Using JMP*.

Creating a Database Connection and Executing SQL

You can use the following functions to handle more complex database operations:

```
db = Create Database Connection( "Connection String With Password" );
Execute SQL( db, "SQL statement", <invisible>, <"New Table Title"> );
Close Database Connection( db );
```

Using these three functions, you can open a connection, call `Execute SQL` several times, and then close the connection. `Create Database Connection` returns a handle for use in `Execute SQL` and `Close Database Connection`.

Depending on the SQL submitted, a table might or might not be returned. A SELECT statement typically returns a JMP table. INSERT INTO would not return a table, because it is modifying one in the database.

Examples

Open a connection to your database:

```
dbc = Create Database Connection(
  "DSN=dBASE Files;DBQ=$SAMPLE_IMPORT_DATA/;"
);
```

Execute one or more SQL statements using this connection:

```
dt = Execute SQL( dbc,
  "SELECT HEIGHT, WEIGHT FROM Bigclass", "NewTable"
);
```

When you are finished, close your connection.

```
Close Database Connection( dbc );
```

You might want to create a database connection without selecting a table. The script would open the Database Open window with the connection already showing. We strongly recommend that you encrypt this script. Here's an example:

```
dbc = Create Database Connection(
  "DSN=dBASE Files;DBQ=$SAMPLE_IMPORT_DATA/;UID=MyDatabase;pwd=MyPassword");
Main Menu( "Open Table" );
```

Note: `Execute SQL()` creates a connection string that includes the cleartext password in the resulting data table. We encourage you to create the SQL query in Query Builder and then run the query in JSL. Then the password is not included in the data table.

Write a SQL Query

The `Query()` function enables you to manipulate JMP data tables using SQL statements. The SQL statement is the only required argument to `Query()`. However, most interesting SQL statements refer to one or more data tables. Any data table that the SQL statement refers to must be passed in using a *table reference* argument.

A table reference argument can be a data table reference (`dt` in the following example):

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
result = Query( dt, "SELECT * FROM 'Big Class' WHERE age > 14;" );
```

The SQL statement in the preceding example includes the name of the JMP data table. If the data table path is long, you might want to use an alias for the data table in the SQL statement. To use an alias, pass the table reference as a `Table()` argument:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
result = Query( Table( dt, "t1" ), "SELECT * FROM t1 WHERE age > 14;" );
```

Using `Table()` to alias your data tables also enables you to write SQL statements for multiple queries with tables that have different names without rewriting the statements.

Private and Hidden Data Tables

If you want the data table that is produced by `Query()` to be either private or hidden, you can pass either `Private` or `Invisible` as arguments to `Query()`.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
result = Query( dt, invisible, "SELECT * FROM 'Big Class' WHERE age > 14;" );
```

Returning a Single Value

The `Query()` function normally returns the result of the query as a JMP data table. However, sometimes you might write a SQL query that returns a single value; you would rather have `Query()` return that value instead of placing the value in a JMP data table. You can do that by passing the `Scalar` argument.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
result = Query(
  Table( dt, "t1" ),
  Scalar,
  "SELECT AVG(height) FROM t1 WHERE age > 14;"
);
```

The preceding query returns 65, the average height of students older than 14.

Work with SAS

Make a SAS DATA Step

Sending `Make SAS Data Step` to a data table returns the text for a SAS DATA Step that can re-create the data table in SAS. For example,

```
dt << Make SAS Data Step
```

prints a DATA Step to the log that can be used in the SAS Program Editor.

Sending `Make SAS Data Step Window` produces this code in a window with a .SAS suffix, so that it can be easily sent to SAS.

Create SAS DATA Step Code for Formula Columns

Sending `Get SAS Data Step for Formula Columns` to a data table includes column formulas in the SAS data step code. Here is an example that outputs the formula for the Ratio column:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Ratio", Formula( :height / :weight ));
dt << Get SAS Data Step for Formula Columns;
```

This script output the following code to the log:

```
/*%PRODUCER: JMP - DataTable Formulas */
/*%TARGET: Ratio */
/*%INPUT: height */
/*%INPUT: weight */
/*%OUTPUT: Ratio */
/* Code to score Ratio */
Ratio =height/weight
drop ;
```

To get formulas for all columns, omit the column names as shown here:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Get SAS Data Step for Formula Columns;
```

You can also include column formulas in scoring code for SAS Model Manager. Send `Get MM SAS Data Step for Formula Columns` to the data table.

```
dt = Open( "$SAMPLE_DATA/Tiretread.jmp" );
dt << Get MM SAS Data Step for Formula Columns;
```

The results of this script are also shown in the log.

As with `Get SAS Data Step for Formula Columns`, specifying column names is optional.

SAS Variable Names

`SAS Open For Var Names()` opens a SAS data set only to obtain the names of its variables, returning those names as a list of strings.

The rules for SAS variable names are more strict than those of JMP. The `SAS Name` function converts JMP variable names to SAS variable names, changing special characters and blanks to underscores, and various other transformations to produce a valid SAS name.

```
result = SAS Name(name);
result = SAS Name({list of names});
```

If the argument is a list of names, the result is a blank-separated character string of names. For example,

```
SAS Name({“x 1”, “x 2”})
```

produces

```
“ x_1 x_2”
```

Connect to a SAS Metadata Server

You can connect to a SAS server and work directly with SAS data sets. Making connections and interacting with SAS data sets is scriptable through JSL. For information about connecting to a metadata server in the JMP interface, see *Using JMP*.

Note: If you connect to a physical workspace server, there is no metadata server involved, so metadata security is never applied. You must connect to a SAS Metadata Server and then connect to a logical workspace server. Then metadata security is enforced on the metadata-defined libraries you access.

Make the Connection

First, use the `Meta Connect()` function to connect to a SAS metadata server:

```
connected = Meta Connect( "MyMetadataServer", port );
```

To specify the SAS version for the metadata server, use the `SASVersion` named argument:

```
connected = Meta Connect( "MyMetadataServer", port, SASVersion("9.4") );
```

If you supply only the machine name (for example, myserver.mycompany.com) and the port, you are prompted to provide the authentication domain, your user name, and your password. You can also specify all that in JSL:

```
connected = Meta Connect( "MyMetadataServer", port, "authdomain", "user
  name", "password" );
```

When you are finished using the SAS metadata server, use `Meta Disconnect()` to disconnect the connection. No arguments are necessary; the command closes the current metadata server connection.

You can see the repositories that are available on a metadata server and set the one that you want to use:

```
Meta Get Repositories();
   {"Foundation"}
Meta Set Repository( "Foundation" );
```

Note that if there is only one repository available, it is selected automatically and you do not need to explicitly set it.

Once your repository is set, you can view the servers that are available:

```
mylist = Meta Get Servers();
   {"SASMain", "Schroedl", "SASMain_ja", "SASMain_zh", "SASMain_ko",
      "SASMain_fr", "SASMain_de", "SASMain_Unicode"}
```

Next, set your SAS connection. You can also use this command to connect directly to a local or remote server instead of using a metadata server.

```
conn = SAS Connect( "SASMain" );
```

Now you can send `Disconnect` and `Connect` messages to the `conn` object to close and open the SAS connection.

```
conn << Disconnect();
conn << Connect();
```

This is an example of using an object and messages with SAS server connections. You might have also connected to SAS servers using global functions. If so, the `Disconnect` and `Connect` messages do not affect those global connections. However, if there is no active global connection, that global connection is set to the connection opened by the object.

Automatically Connect SAS Libraries

When you connect to a SAS server, use `Connect Libraries` to automatically connect metadata-defined libraries.

```
conn = SAS Connect( "SASMain", Connect Libraries( 1 ) );
```

All metadata-defined libraries are connected, which can slow down your connection.

To connect specific libraries later, use the `SAS Connect Libref()` function or `Connect Libref` message to a SAS server object.

View SAS Libraries

After you connect to a SAS server, use the `Get Lib Ref()` function to view the libraries on that server:

```
conn << Get Lib Refs();
   {"BOOKS", "EGSAMP", "GENOMICS", "GISMAPS", "JMPSAMP", "JMPTEST",
   "MAILLIB", "MAPS", "OR_GEN", "ORION_RE", "ORSTAR", "SASHELP",
   "SASUSER", "TEMPDATA", "TSERIES", "V6LIB", "WORK", "WRSTEMP"}
```

If the library containing the data that you want is not assigned, assign it:

```
conn << Assign Lib Refs( "MyLib", "$DOCUMENTS/data" );
```

Open SAS Data Sets

First, assign a SAS library reference:

```
conn << Assign Lib Refs( "MyLib", "$DOCUMENTS" );
```

The first argument is any name that you want to use to refer to the library reference. The second is the path on the server where the data sets are located.

Next, get the list of data sets in the selected library:

```
datasets << Get Data Sets( "MyLib" );
   {"ANDORRA", "ANDORRA2", "ANYVARNAME", "BOOKS", "BOOKSCOPYNOT", "BOOKS_VIEW",
      "CATEGORIES", "DATETIMETESTS", "MOREUGLY", "NOTTOOUGLY", "PAYPERVIEW",
      "PUBLISHERS", "PURCHASES", "PURCHASES_FULL",
```

Now you can open a data set:

```
conn << Import Data( "MyLib", "PURCHASES" );
```

or

```
conn << Import Data( librefs[1], datasets[12] );
```

or

```
conn << Import Data( "MyLib.PURCHASES" );
```

Now you can get information about any SAS data set in that library. For example, you can get a list of variables:

```
conn << Get Var Names( "MyLib.PURCHASES" );
   {"purchaseyear", "purchasemonth", "purchaseday", "bookid", "catid",
   "pubid", "price", "cost"}
```

With that information, you can choose to import only part of the data set by specifying the variables to import.

```
conn << Import Data( librefs[1], datasets[12], columns( bookvars[1],
   bookvars[2], bookvars[4] ) );
```

Note: The name specified in `Import Data` can be a friendly metadata library name or a SAS 8-character library name.

Save SAS Data Sets

To save a JMP data table or an imported SAS Data Set, use the `SAS Export Data()` command:

```
conn << Export Data( dt, librefs[1], datasets[4], ReplaceExisting );
```

Run a Stored Process

To get a reference to a stored process:

```
stp = Meta Get Stored Process( "Samples/Stored Processes/Sample: Hello World"
  );
```

There is no way to acquire a list of stored processes through JSL; you must know the path to the stored process that you want to run.

To run it, send the stored process a message:

```
stp << Run();
```

Submit SAS Code from JMP

You can also directly submit SAS code and get back SAS results. For example:

```
conn << Submit( "proc print data=sashelp.class; run;" );
```

Two optional arguments control whether you see the output and the SAS log in JMP:

```
conn << Submit( "SAS Code" <,No Output Window(True|False)> <,Get
  Log(True|False )> );
```

You can also see the SAS Log at any time using the command

```
conn << Get Log();
```

`Get Log()` returns the contents, which can be placed in a JSL variable and used like any JSL string.

Get the Values of SAS Macro Variables

JMP provides several methods for querying SAS macro variables.

To show the `systime` value:

```
systime = sas << Get Macro Var("SYSTIME");
show(systime);
```

To show the defined SAS macro variables:

```
macro_names = sas << Get Macro Var Names();
show(macro_names);
```

To iterate through the SAS macro variables and print out the values:

```
macro_names = sas << Get Macro Var Names();
For( i = 1, i <= N Items( macro_names ), i++,
  macro_value = sas << Get Macro Var( macro_names[i] );
  output = macro_names[i] || " " || Char( macro_value );
  Show( output );
);
```

To submit SAS code that defines “test” as a SAS macro variable and then gets the value from SAS:

```
sas << Submit( "%let test = 1;" );
test = sas << Get Macro Var( "test" );
Show( test );
```

All macro variable values will evaluate to numbers, if possible, otherwise they will be characters.

Preferences

To get the current SAS version preference, use:

```
Get SAS Version Preference();
```

To set the current SAS version preference, use:

```
Preference( SAS Integration Settings( SASVersion( "9.4" ) ) );
```

Sample Scripts

The JMP Samples/Scripts/SAS Integration folder contains sample scripts. To run the stored process scripts successfully, the stored processes need to be placed on your SAS Metadata Server. The stored processes can be found in the sampleStoredProcesses.spk file, also in this folder.

To import sampleStoredProcesses.spk into your SAS Metadata Server

Caution: We recommend that you import these stored processes into a SAS Metadata Server that is used for testing rather than into a production system.

1. Run SAS Management Console.

2. Connect to your SAS Metadata Server using an account with administrative privileges.
3. Expand the BI Manager node in the left pane of the SAS Management Console.
4. Navigate to the folder in the tree under which you would like to create the imported sample stored processes.
5. Right-click that folder in either the left pane or the right pane of the SAS Management Console and select **Import**.

   The Import Wizard appears.
6. Enter the full path to sampleStoredProcesses.spk or use the **Browse** button to navigate to it.
7. Select **All Objects** in the Import Options section of the wizard.
8. Click **Next**.

   The next panel reports that during the import process, you must specify values for **Application servers** and **Source code repositories**.
9. Click **Next**.

   Select which of the application servers defined in your SAS Metadata Server that you would like to use to execute the imported stored processes.
10. Select an application server from the drop-down list under Target.
11. Click **Next**.

    Select the source code repository (directory) defined on your SAS Metadata Server where you would like the SAS code for the imported stored processes to be placed.
12. Select a source code repository from the drop-down list under Target Path.
13. Click **Next**.

    The next panel gives a summary of what occurs if you click **Import**.
14. Review the information about the panel, and if it looks correct, click **Import**.
15. During the Import process, you might be asked to provide login credentials for connecting to the metadata server to perform the import. Provide credentials with administrative privileges and click **OK**.

After the import completes, you will find a folder named BIP Tree under the folder that you imported the stored processes into. Under BIP Tree is a folder named JMP Samples, and in the JMP Samples folder are two sample stored processes: Shoe Chart and Diameter.

Please note that the paths to the sample stored processes needs to be adjusted in the sample scripts storedProcessHTML.jsl and storedProcessJSL.jsl to match the folder into which you imported the sample stored processes. Otherwise, these scripts will not work correctly.

Work with MATLAB

MATLAB, a product of MathWorks Inc., provides an interactive working environment for analyzing and visualizing computational models. MATLAB is available for Windows, Mac OS X, and Linux (64-bit). JMP supports MATLAB on both Windows and macOS platforms.

You can interact with MATLAB using JMP Scripting Language (JSL):

- Submit statements to MATLAB from within a JSL script.
- Exchange data between JMP and MATLAB.
- Display graphics produced by MATLAB.

See the *JSL Syntax Reference* for more information about using MATLAB functions in JMP.

Textual output and error messages from MATLAB appear in the log window.

Install MATLAB

MATLAB must be installed on the same computer as JMP. For the supported version of MATLAB, see the JMP website: https://www.jmp.com/system/.

How JMP Finds MATLAB

JMP delays loading MATLAB until a JSL-based script requires access to it. When you run a JSL script that calls MATLAB, JMP locates the software based on the operating system's `PATH` environment variable (for example, `C:\Program Files\MATLAB\R2012a\`).

Test Your Install

To verify that your computer is able to run JSL-based scripts using MATLAB:

1. Run the following JSL script:

```
MATLAB Init();
MATLAB Submit( "m = magic(3)" );
magicMat = MATLAB Get( m );
Show( magicMat );
MATLAB Term();
```

   The MATLAB function `M = magic(3)` returns a 3-by-3 matrix using integers in the range of 1 to 3^2 with equal row and column sums. This matrix is called a *magic square*.

2. Select **View > Log**.

   You should see the following response in the log window:

```
m =
```

```
      8     1     6
      3     5     7
      4     9     2


magicMat =
[ 8 1 6,
  3 5 7,
  4 9 2];
0
```

If you see the following message in the log window:

```
An installation of MATLAB cannot be found on this system.
```

1. Add a new environment variable with the `name` of `MATLABROOT` and a value of C:\Program Files\MATLAB\R2012a\ or C:\Program Files (x86)\MATLAB\R2012a\.

   Note: The path entered depends on the MATLAB installation path.

2. Verify that the `MATLAB` path is included in the `PATH` variable.
3. Run the script again to ensure JMP can access MATLAB.

Work with R

You can interact with R using JSL:

- Submit statements to R from within a JSL script.
- Exchange data between JMP and R.
- Display graphics produced by R.

Text output and error messages from R appear in the log window.

Installing R

Install R on the same computer as JMP. You can download R from the Comprehensive R Archive Network website:

https://cran.r-project.org

Note: For macOS Catalina and later versions, download the notarized and signed package.

Because JMP is a 64-bit application, you must install the corresponding 64-bit version of R. For the supported version of R, see the system requirements on the JMP website: https://www.jmp.com/system.

Override Default R Install Location on Windows

Normally JMP determines the R_HOME environment variable internally if it is not defined in the Windows system registry:

computer\HKEY_LOCAL_MACHINE\SOFTWARE\R-code\R\InstallPath

You might have installed more than one version of R on your computer. To override the default R installation location, define the R_HOME environment variable using either of the two following methods:

1. Create the variable in your system environment variables using the Control Panel, select **Start > Control Panel > System > Advanced system settings**.
2. Click **Environment Variables**.
3. In the System variables pane, click **New**.
4. Type R_HOME for the **Variable name**.
5. Type the path to the R .exe file (for example, C:\Program Files\R\R-2.15.3).
6. Click **OK** and click **OK** again to close the System Properties window.

or

Create the variable using the JSL `Set Environment Variable()` function:

```
Set Environment Variable( "R_HOME", "C:\Program Files\R\R-2.15.3" );
```

Override the Default R Install Location on macOS

You might have installed more than one version of R on your computer. To use an older version, change the R_HOME environment variable to point to the R version that you want to use.

To specify the R_HOME location, follow these steps:

1. Open a terminal window.
2. Type the following code, where *value* is the location of R.

   `export R_HOME=value`
3. Press Enter.

How JMP Finds R on Windows

JMP delays loading R until a JSL-based script requires access to it. When JMP needs to load R, it follows the standard steps for finding R on a Windows computer:

1. Look up the environment variable `R_HOME`.

   If the variable exists, load R from the specified directory.
2. If the environment variable `R_HOME` does not exist, look up the `InstallPath` value in the Windows registry under the following key:

   `HKEY_LOCAL_MACHINE\SOFTWARE\R-core\R`

   If the `InstallPath` value exists, load R from the specified directory.
3. If the `InstallPath` value does not exist, an error message states that R could not be found.

How JMP Finds R on macOS

JMP uses the value of the `R_HOME` environment variable if the variable exists. Otherwise, it looks in `/Library/Frameworks/R.framework/Versions/Current/Resources` for R.

Testing Your Setup

To test that your computer is able to run JSL-based scripts that use R, run the following JSL script:

```
R Init( );
R Submit( "
   x <- 1:5
   x
" );
R Term( );
```

You should see the following output in the log:

```
[1] 1 2 3 4 5
```

JMP to R Interfaces

The following JMP interfaces are provided to access R. The basic execution model is to first initialize the R connection, perform the required R operations, and then terminate the R connection. In most cases, these functions return 0 if the R operation was successful, or an error code if it was not. If the R operation is not successful, a message is written to the log. The single exception to this is `R Get()`, which returns a value.

R JSL Scriptable Object Interfaces

The R interfaces are also scriptable using an R connection object. A scriptable R connection object can be obtained using the `R Connect()` JSL function.

Conversion Between JMP Data Types and R Data Types

Table 14.3 shows what JMP data types can be exchanged with R using the `R Send( )` function. Sending lists to R recursively examines each element of the list and sends each base JMP data type. Nested lists are supported.

Table 14.3 Equivalent JMP and R Data Types for R Send( )

R Data Type	JMP Data Type
Double	Numeric
String	String
Double Matrix	Matrix
List	List
Data Frame	Data Table
Integer	Row State
Date and Time	Datetime
Time/Duration	Duration
Paired List	Associative Array

Example

```
R Init();
X = 1;
R Send( X );
S = "Report Title";
R Send( S );
M = [1 2 3, 4 5 6, 7 8 9];
R Send( M );
R Submit( "
X
S
M
" );
```

```
R Term();
```

Table 14.4 shows what JMP data types can be exchanged with R using the `R Get()` function. Getting lists from R recursively examines each element of the list and sends each base R data type. Nested lists are supported.

Table 14.4 Equivalent JMP and R Data Types for R Get( )

R Data Type	JMP Data Type
Double	Numeric
Logical (Boolean)	Numeric (0 | 1)
String	String
Integer	Numeric
Date and Time	Datetime
Time/Duration	Duration
Factor	Expanded to a list of Strings or a Numeric matrix
Data Frame	Data Table
List	List of converted R data types
Matrix	Numeric Matrix
Numeric Vector	Numeric Matrix
String Vector	List of Strings
Graph	Picture
Time Series	Matrix
Paired List	Associative Array

JMP Scoping Operators and R

A JMP object sent to R using `R Send()` uses the same JMP reference as the name of the R object that gets created. For example, sending the JMP variable `dt` to R creates an R object named `dt`. The colon and double colon scoping operators (`:` and `::`) are not valid in R object names, so these are converted as follows:

- A single colon scoping operator is replaced with a period (`.`).

  For example, sending `nsref:dt` to R creates a corresponding R object named `nsref.dt`.

- A double colon scoping operator (designating a global variable) is ignored.

  For example, sending `::dt` to R creates a corresponding R object named `dt`.

Using R Name() with R Send()

The `R Name()` option to `R Send()` has an argument that is a quoted string that contains a valid R object name. The JMP object sent to R becomes an R object with the name specified. For example:

```
R Send( jmp_var_name, R Name( "r_var_name") );
R Submit( "print(r_var_name)" )
```

Example

This example creates a variable `x` in the `Here` namespace, a variable `y` in the global namespace, and a variable `z` that is not explicitly referenced to any namespace. The variable `z` defaults to Global unless `Names Default To Here(1)` is on. These variables are then passed to R.

```
Here:x = 1;
::y = 2;
z = 3;

R Init(); // initiate the R connection

/* send the Here variable to R
Here:x creates the R object Here.x */
R Send( Here:x );

/* note that the JMP log labels the output with the original JMP variable
   reference Here:x */
R Submit( "print(Here.x)" );

R Send( ::y ); // ::y create the R object y
R Submit( "print(y)" );

// to use a different name for the R object, use the R Name() option
R Send( Here:x, R Name( "localx" ) );

/* The R Name option to the R Send() command creates the R object named
   "localx"" corresponding to the JMP variable "Here:x". Again the log shows
   the original corresponding JMP variable name. */
R Submit( "print(localx)" );
R Send( z ); // z creates the R object z
R Submit( "print(z)" );
```

Troubleshooting

Recording Output

On Windows, if you want to record output to the graphics window, send the following R code using `R Submit( )`.

```
windows.options( record = TRUE );
```

Character Vectors

A JMP list of strings is not the same as an R character vector. If you send a list of strings to R, it becomes a list of strings in R, not a character vector. You can use the R function Unlist to convert it:

```
R Init();
X = {"Character", "JMP", "List"};
R Send( X );

/* R output is:
[1] "list"
*/
R Submit( "class(X)" );

/* Object Y is now a character vector. The R output is:
[1] "character"
*/
R Submit( "Y<-unlist(X)
     class(Y)" );
R Term();
```

Element Names

A feature of an R list (called attributes) lets you associate a name with each element of the list. You can use the name to access that element instead of having to know the position of it in the list

In the following example, the list that is created in R has two elements named x and y that are created using the `List()` function of R. When your bring the R list into JMP and then send it back to R, the names are lost. Therefore in R, you cannot access the first matrix using `pts$x`. Instead, you must use the index using `pts[[1]]`.

```
R Init();
R Submit("
   pts <- list(x=cars[,1], y=cars[,2])
   summary(pts)
");
```

```
JMP_pts = R Get( pts );

R Send( JMP_pts );
R Submit("
   Summary( JMP_pts )
");
R Term();
```

Examples

Sending a Data Table to R

This example initiates an R connection, sends a data table to R, prints it to the log, and closes the R connection.

```
R Init();
dt = Open( "$SAMPLE_DATA/Big Class.jmp", invisible );
R Send( dt ); // send the opened data table represented by dt to R;
R Submit( "print( dt )" );
R Term();
```

Creating Objects in R

This example initiates an R connection, creates an R object, retrieves the object into JMP, and closes the R connection.

```
R Init();
R Submit(
   "
   L3 <- LETTERS[1:3]
   d <- data.frame(cbind(x=1, y=1:15), Group=sample(L3, 15, repl=TRUE))
"
);
R Get( d ) << NewDataView;
R Term();
```

Using R Functions and Graphics

This example initiates an R connection and plots the normal density function in R using the R graphics device,. Then the graph is retrieved from R and displayed in JMP. Finally, the R connection is closed.

```
R Init();
R Submit( "\[plot(function(x) dnorm(x), -5, 5, main = "Normal(0,1) Density")
   ]\" );
```

```
picture = R Get Graphics( "png" );
New Window( "Picture", picture );
Wait( 10 );
R Term();
```

Simple Matrix Addition in R

This example initiates an R connection, sends a matrix to R, creates a matrix in R, adds them together, returns the new matrix to JMP, and closes the R connection.

```
R Init();
X = J( 2, 2, 1 );
R Send( X );
R Submit(
  "
  X                                    #prints X to the log
  Y <- matrix(1:4, nrow=2, byrow=TRUE)  #makes a 2x2 matrix object Y
  Y                                    #prints Y to the log
  Z <- X + Y #matrix object Z is addition of X and Y
"
);
Z = R Get( Z );
R Term();
Show( Z );
```

A Bootstrap Sample

See the file JMPtoR_bootstrap.jsl in the sample scripts folder for an example script.

This script performs a bootstrap simulation by using the JMP to R Project integration.

The script produces a JMP window that asked the user to specify the variable to perform bootstrapping over. Then the user selects a statistic to compute for each bootstrap sample. Finally, the data is sent to R using the R interface in JSL.

The boot package in R is used to call the `boot()` function and the `boot.ci()` function to calculate the sample statistic for each bootstrap sample and the bootstrap confidence interval.

The results are brought back to JMP and displayed using the JMP Distribution platform.

Work with Python

The JMP-Python integration allows JMP users to interact with a large amount of scientific, statistical, and data analysis tools written in Python. JMP implements this capability by loading the Python shared library into the running JMP process. JSL gives Python access to JMP.

The JMP installer does not contain Python. Several Python implementations are available. Visit https://www.python.org/download/alternatives to view the options.

JMP has several methods for finding an instance of Python installed on the computer. You should not need to set any environment variables or additional arguments to the JSL `Python Init()` statement. See "Troubleshooting the JMP Python Integration" for more information.

The Python integration enables you to perform the following tasks:

- submit statements to Python from within a JSL script
- exchange data between JMP and Python
- display graphics produced by Python

Text output and error messages from Python appear in the log window.

Install Python

JMP 16 is a 64-bit program; only 64-bit Python is supported

- Python.org 3.6 through 3.9.1 distributions are supported on Windows and macOS. On Apple silicon, you will not have full functionality until third-party packages have been updated.
- Anaconda 3.6 distribution is supported on Windows only.
- Anaconda 3.7 distribution is not supported on Windows.
- The way Python is built from versions 3.6 to 3.7 changed. In Python 3.7, Python must be installed for all users. Otherwise, the libraries that the Python DLL depend on cannot be located at DLL load time. Other Python distributions might work fine installed for a single user or might also require a system-wide install.
- JMP does not currently support Python virtual environments.

Notes:

- Python is not distributed with JMP. Download Python from the Python web site at https://python.org.
- Several Python implementations are available. Visit https://www.python.org/download/alternatives to view the alternatives.

- On macOS, installing Python 3.6.x from https://python.org installs in /Library/Frameworks/Python.framework.

Required Python Packages

For the JMP-to-Python interfaces to work properly, you must install certain Python packages and their prerequisites. In many Python installations, such as Anaconda, these packages are already installed. If you install Python from Python.org, also install the following packages:

numpy https://www.numpy.org

pandas https://pandas.pydata.org

matplotlib https://matplotlib.org

scipy https://www.scipy.org

sqlite3 https://www.sqlite.org

PyQt5 https://pypi.org/project/PyQt5/

Use the standard Python update tools to install these packages for your Python configuration. You must use the correct pip-tool for your installation. The pip-tool name is dependent on the what product has installed Python. The installation instructions for Python have instructions on using pip tools.

Troubleshooting the JMP Python Integration

Sometimes JMP has trouble locating the correct version of Python. The first thing to try is running Python from the command line. If Python doesn't run from a terminal or console window, something is wrong with the Python installation.

Note: Before you proceed with troubleshooting, verify that you're running a supported version of Python. See "Install Python".

How JMP Finds Python on Windows

JMP delays loading Python until a JSL-based script requires access to it. When JMP loads Python, it needs to be able to find the Python shared library and any of the shared libraries that the Python DLL depends on. Once the Python interpreter is loaded, Python needs to be able to find the distribution's standard Python modules. To guide this process, JMP uses registry entries, environment variables, and user-specified paths.

Python also looks for certain environment variables. Avoid using any user-specified paths or environment variables, and let JMP find Python in the registry. The following methods help JMP locate Python if the automatic location doesn't work:

- JMP_PYTHON_MODULE_PATH is a JMP-specific version of PYTHONPATH. This variable is the replacement for Python Sys Path. In JMP_PYTHON_MODULE_PATH, specify standard OS paths. Separate the paths with a colon on macOS. Separate the paths with a semicolon on Windows.
- JMP_LIB_PYTHON_PATH is a JMP-specific version of PYTHONHOME. The variable contains the full path to the Python shared library, or the parent directory to the shared library.

Windows finds Python in the following order of precedence:

1. JMP looks at the environment variables Python Sys Path (deprecated), PYTHONINSTALLPATH (deprecated), and JMP_PYTHON_MODULE_PATH, in that order. JMP uses the first variable that is found.
2. JMP checks JMP_LIB_PYTHON_PATH for a path to the Python shared library.
3. JMP checks for options that are specified in the Python Init() or Python Connect() function calls. Path() corresponds to the location of the Python shared library. Use Python Version() allows you to specify the version of Python to look for.

   Python Sys Path provides the mechanism to set Python's sys.path variable. If present, script parameters take precedence over the environment variables. There is one caveat. After JMP loads the Python shared library, the library is unloaded only when JMP shuts down. As a result, the Path() portion is only effective on the first load of Python, and does not allow for running different versions of Python in different scripts without shutting down JMP and restarting.
4. JMP reads the Windows registry to determine if Python has been installed. It looks at appropriate locations based on whether the installation was for an individual user or for all users on the Windows machine. If you have specified a Python version, other versions of Python will not be considered.

If JMP cannot locate Python, try the following option in the Python Init() expression:

```
Python Init( Init Trace( "TRUE" ) );
```

JMP prints details to the log about where it detects Python, including paths and registry entries for Python.

Typically, JMP determines the equivalent to the PYTHONHOME environment variable automatically if it is defined in the Windows registry.

Computer\<root>\SOFTWARE\Python\PythonCore\<version>\InstallPath

<root> is either the root key HKEY_LOCAL_MACHINE or HKEY_CURRENT_USER depending on where and how you installed Python. *<version>* is the Python version number.

If multiple versions of Python are installed on your computer, you must specify which installation for JMP to use.

- The easiest method is to specify the version using the Use Python Version() argument in Python Init() or Python Connect() functions.

- Use the `Path("path to an installed Python DLL or shared library")` argument with the `Python Init()` or `Python Connect()` function.
- Define the `JMP_LIB_PYTHON_PATH` environment variable as described below.

Change the Default Python Install Location on Windows

Create the variable in your system environment variables using the Control Panel.

1. Select **Start > Control Panel > System > Advanced system settings**.
2. Click **Environment Variables**.
3. In the System variables pane, click **New**.
4. Type `JMP_LIB_PYTHON_PATH` for the Variable name.
5. Type the path to the Python<version>.dll file, for example:
   `C:/Program Files/Python/<version>/Python<version>.dll`
6. Click **OK** and click **OK** again to close the System Properties window.

macOS: Override How JMP Finds Python

Locating Python on the Mac works the same as on Windows except for the step of looking in the registry. JMP looks in a couple of typical installation directories so see if it can locate Python. Python typically installs in `/Library/Frameworks/Python.framework`. A compatible version of Python located in the standard system location should work without any changes to environment variables or parameters passed to JSL.

If Python is installed in an alternate location, set configuration variables in your shell environment as needed. JMP on macOS uses the same environment variables as described in "How JMP Finds Python on Windows".

`JMP_LIB_PYTHON_PATH` is the path to the Python shared library `libpython3.x.dylib`, where *x* is the minor version. If environment variables are needed, you typically only need to set `JMP_LIB_PYTHON_PATH`.

If JMP is unable to find install Python packages, you might also need to set the variable `JMP_PYTHON_MODULE_PATH`. The `JMP_PYTHON_MODULE_PATH` contains the same paths that the Python call print(sys.path) outputs. When you set an environment variable, use host OS path conventions. When the variable is passed in JSL, use a JSL list.

One way is to have Python tell you what it thinks its `sys.path` should be. In the example in this section, the same Python commands can be run on Windows to determine `sys.path`.

In the terminal or console window, enter the following boldface commands to get a string that specifies the location of the Python packages:

```
MDz320:~ shlori$ python --version
Python 3.6.4 :: Anaconda, Inc.
MDz320~ shlori$ python3
Python 3.6.4 |Anaconda, Inc.| (default, Jan 16 2018, 12:04:33) [GCC 4.2.1
  Compatible Clang 4.0.1 (tags/RELEASE_401/final)] on darwin Type "help",
  "copyright", "credits" or "license" for more information.
>>> import sys
>>> print(sys.path)
  ['', '/anaconda3/lib/python36.zip', '/anaconda3/lib/python3.6',
     '/anaconda3/lib/python3.6/lib-dynload',
     '/anaconda3/lib/python3.6/site-packages',
     '/anaconda3/lib/python3.6/site-packages/aeosa']
```

Test Your Setup

To test that your computer is able to run JSL scripts that use Python, run the following script:

```
Python Init();
Python Submit( "\[basket = ['apple', 'orange', 'pear']
print(basket)]\" );
Python Term();
```

You should see the following output in the log:

```
['apple', 'orange', 'pear']
0
```

JMP-to-Python Interfaces

JMP provides the interfaces to access Python that are described in the following sections. The basic execution model is to first initialize the Python connection, perform the required Python operations, and then terminate the Python connection. In most cases, these functions return 0 if the Python operation was successful or an error code if it was not. If the Python operation is not successful, a message is written to the log. The single exception to this is `Python Get()`, which returns a value.

Python JSL-Scriptable Object Interfaces

The Python interfaces are also scriptable using a Python connection object. A scriptable Python connection object can be obtained using the `Python Connect()` function.

```
// return a Python connection scriptable object
PythonConnection = Python Connect();

// return 1 if a Python connection is active. Otherwise, returns 0.
x = PythonConnection << Is Connected;
Show( x );
PythonConnection << Disconnect;
   x = 1;
   0
```

Equivalent Data Types for Python Send()

Table 14.5 shows what JMP data types can be exchanged with Python using the `Python Send()` function. Sending lists or associative arrays to Python recursively examines each element of the list or associative array and sends each base JMP data type. Nested lists and associative arrays are supported.

Table 14.5 Equivalent JMP and Python Data Types for Python Send()

JMP Data Type	Python Data Type
Boolean	Boolean
Data Table	Pandas.DataFrame
Associative Array	Dictionary

Table 14.5 Equivalent JMP and Python Data Types for Python Send() *(Continued)*

JMP Data Type	Python Data Type
Numeric	Float
Matrix	Float Matrix
List	List
String	Unicode String

Python Send() Example

```
Python Init();
X = 1;
Python Send( X );
S = "Report Title";
Python Send( S );
M = [1 2 3, 4 5 6, 7 8 9];
Python Send( M );
Python Submit( "\[
print(X)
print(S)
print(M)
]\" );
Python Term();
```

```
1.0
Report Title
[[ 1.  2.  3.]
 [ 4.  5.  6.]
 [ 7.  8.  9.]]
0
```

Equivalent Data Types for Python Get()

Table 14.6 shows which JMP data types can be exchanged with Python using the `Python Get()` function. Getting lists from Python recursively examines each element of the list and gets each base Python data type. Nested lists are supported.

Table 14.6 Equivalent Python and JMP Data Types for Python Get()

Python Data Type	JMP Data Type
Array	Numeric Matrix
Date and time	Numeric

Table 14.6 Equivalent Python and JMP Data Types for Python Get() *(Continued)*

Python Data Type	JMP Data Type
Pandas.DataFrame	Data Table
Dictionary	Associative Array
Graph	Picture Object
Integer	Integer
List	List of converted Python data types
Matrix	Numeric Matrix
Set	List
Unicode String	String

Python Get() Example

```
Python Init();
x1 = [1, 2, 3];
Python Send( x1 );
x2 = Python Get( x1 );
Show( x1, x2 );
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
Python Send( dt1 );
dt2 = Python Get( dt1 );
dt2 << New Data View;
Close( dt1 );
Python Term();
```

JMP Scoping Operators and Python

A JMP object sent to Python with `Python Send()` uses the same JMP reference as the name of the Python object that gets created. For example, sending the JMP variable `dt` to Python creates a Python object named `dt`.

The colon and double colon JSL scoping operators (`:` and `::`) are not valid in Python object names, so the colons are converted as follows:

- A single colon scoping operator is replaced with an underscore (_).

  For example, sending `nsref:dt` to Python creates a corresponding Python object named `nsref_dt`.
- A double colon scoping operator (designating a global variable) is ignored.

For example, sending `::dt` to Python creates a corresponding Python object named `dt`.

Note: The `Python Name` option for `Python Send()` has a quoted string argument that contains a valid Python object name. The JMP object sent to Python becomes a Python object with the specified name.

Python Name() and Python Send() Examples

The following example creates the `jmp var name` variable, assigns it to the Python name `python_var_name`, submits the `print` statement, and closes the connection.

```
Python Init();
jmp var name = 25;
Python Send( jmp var name, Python Name( "python_var_name" ) );
Python Submit( "print(python_var_name)" );
Python Term();
```

```
25.0

0
```

The following example creates a variable *x* in the Here namespace, a variable *y* in the global namespace, and a variable *z* that is not explicitly referenced to any namespace. The variable *z* defaults to Global unless `Names Default To Here(1)` is on. These variables are then passed to Python.

```
Here:x = 1;
::y = 2;
z = 3;

Python Init(); // initiate the Python connection

/* send the Here variable to Python
Here:x creates the Python object Here_x */
Python Send( Here:x );

Python Submit( "print(Here_x)" );

Python Send( ::y ); // ::y creates the Python object y
Python Submit( "print(y)" );

// to use a different name for the Python object, use the Python Name() option
Python Send( Here:x, Python Name( "localx" ) );

/* The Python Name option to the Python Send() command creates the Python
   object named "localx", which corresponds to the JMP variable "Here:x". */
Python Submit( "print(localx)" );
```

```
Python Send( z ); // z creates the Python object z
Python Submit( "print(z)" );
Python Term();
  1.0

  2.0

  1.0

  3.0

  0.0
```

Additional Python Integration Examples

Send a Data Table to Python

This example initiates a Python connection, sends a data table to Python, prints it to the log, and closes the Python connection.

```
Python Init();
dt = Open( "$SAMPLE_DATA/Big Class.jmp", invisible );
Python Send( dt ); // send the opened data table represented by dt to Python
Python Submit( "print( dt )" );
Python Term();
          name     age   sex   height   weight
  0       KATIE    12    F       59       95
  1      LOUISE    12    F       61      123
  2        JANE    12    F       55       74
  ...
  0
```

Create Objects in Python

This example initiates a Python connection, creates a Python object, retrieves the object into JMP, and closes the Python connection.

```
Python Init();
Python Submit("\[
import numpy as np
import pandas as pd
# Generate basic series
ss = pd.Series(np.random.randn(5), index=['a', 'b', 'c', 'd', 'e'])
print(ss)
sn = pd.Series(np.random.randn(5))
print(sn)

# Generate Data Frames from Series
```

```
s1 = pd.Series(np.random.randn(5), index=['a', 'b', 'c', 'd', 'e'])
print(s1)
dfs1 = pd.DataFrame(s1)
print(dfs1)

d1 = {'one' : pd.Series([1., 2., 3.], index=['a', 'b', 'c']),
      'two' : pd.Series([1., 2., 3., 4.], index=['a', 'b', 'c', 'd'])}
print(d1)
dfd1 = pd.DataFrame(d1)
print(dfd1)

d2 = {'one' : [1., 2., 3., 4.],
      'two' : [4., 3., 2., 1.]}
print(d2)
dfd2 = pd.DataFrame(d2)
print(dfd2)
]\"
);
ss = Python Get( ss );
Show( ss );
sn = Python Get( sn );
Show( sn );
dfs1 = Python Get( dfs1 );
dfs1 << New Data View;
dfd1 = Python Get( dfd1 );
dfd1 << New Data View;
dfd2 = Python Get( dfd2 );
dfd2 << New Data View;
Python Term();
   a   -0.384649
   b    0.455864
   c   -2.214405
   d    0.260955
   e    1.342286
   dtype: float64
   ...
```

Matrix Operations in Python

In JMP, a simple matrix looks like this:

```
xx = [1 2 3, 4 5 6, 7 8 9 ];
```

In numpy, the matrix is an array:

```
xx = numpy.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

In order to submit a matrix as a numpy array, you must translate the JMP matrix.

```
q = "new = numpy.array ([";
```

```
For( i = 1, i <= N Items( xx ), i++,
   zz = Matrix( xx[i] );
   q = q || Char( zz ) || ",";
);
q = q || "])";
Show( q );

Python Submit( "import numpy
" || q );
Python Submit( "print(new)" );
```

Work with Microsoft Excel

You can script the Profiler interface to Microsoft Excel, although not the Transfer to JMP interface. Here is the basic syntax:

```
excel_obj = Excel Profiler(
   Workbook( "excel_workbook_path" ),
   Model( "name_of_model" )
);
```

The `Model` argument is optional. If you supply only the workbook, you are prompted to select the model. You do not need to specify the worksheet, because the model is found no matter where it is located.

Once you have an Microsoft Excel Profiler object, you can send the following messages to run the JMP prediction profiler using the Microsoft Excel model. For example:

```
excel_obj <<Prediction Profiler( 1 );
```

Parse XML

JSL has several commands available to parse XML.

`Parse XML( `*`string`*`, On Element( "`*`tagname`*`", Start Tag( `*`expr`*` ), End Tag( `*`expr`*` ) ) );`

parses an XML expression using the `On Element()` expression for specified XML tags.

`value = XML Attr( "`*`attribute name`*`" );`

extracts the string value of an XML argument when evaluating a `Parse XML()` expression.

```
value = XML Text();
```

extracts the string text of the body of an XML tag when evaluating a `Parse XML()` expression.

Example of Parsing XML

Suppose that a Microsoft Excel file contains one row of data from Big Class.jmp. The file is saved as the valid XML document BigclassExcel.xml, shown here and also saved in the JMP Samples/Import Data folder.

```
<?xml version="1.0" encoding="UTF-8"?>
<Workbook xmlns="urn:schemas-microsoft-com:office:spreadsheet"
 xmlns:o="urn:schemas-microsoft-com:office:office"
 xmlns:x="urn:schemas-microsoft-com:office:excel"
 xmlns:ss="urn:schemas-microsoft-com:office:spreadsheet"
 xmlns:html="http://www.w3.org/TR/REC-html40">
 <Worksheet ss:Name="Bigclass">
  <Table ss:ExpandedColumnCount="5" ss:ExpandedRowCount="41"
   x:FullColumns="1"
   x:FullRows="1">
   <Row>
    <Cell><Data ss:Type="String">name</Data></Cell>
    <Cell><Data ss:Type="String">age</Data></Cell>
    <Cell><Data ss:Type="String">sex</Data></Cell>
    <Cell><Data ss:Type="String">height</Data></Cell>
    <Cell><Data ss:Type="String">weight</Data></Cell>
   </Row>
   <Row>
    <Cell><Data ss:Type="String">KATIE</Data></Cell>
    <Cell><Data ss:Type="Number">12</Data></Cell>
    <Cell><Data ss:Type="String">F</Data></Cell>
    <Cell><Data ss:Type="Number">59</Data></Cell>
    <Cell><Data ss:Type="Number">95</Data></Cell>
   </Row>
  </Table>
 </Worksheet>
</Workbook>
```

The following script reads BigclassExcel.xml and creates a JMP data table with the information in it. This script, named ParseXML.jsl, is in the JMP Samples/Scripts folder.

```
file contents = Load Text File( "$SAMPLE_IMPORT_DATA/BigclassExcel.xml" );
Parse XML( file contents,
   OnElement(
      "urn:schemas-microsoft-com:office:spreadsheet^Worksheet",
      StartTag(
         sheetname = XML Attr(
            "urn:schemas-microsoft-com:office:spreadsheet^Name",
            "Untitled"
         );
         dt = New Table( sheetname );
         row = 1;
```

```
            col = 1;
         )
      ),
      OnElement(
         "urn:schemas-microsoft-com:office:spreadsheet^Row",
         StartTag(
            If( row > 1, // first row has column names
               dt << Add Rows( 1 )
            )
         ),
         EndTag(
            row++;
            col = 1;
         )
      ),
      OnElement( "urn:schemas-microsoft-com:office:spreadsheet^Cell", EndTag(
      col++ ) ),
      OnElement(
         "urn:schemas-microsoft-com:office:spreadsheet^Data",
         EndTag(
            data = XML Text( collapse );
            If( row == 1,
               New Column( data, Character( 10 ) ), // first row has column names
               Column( col )[row - 1] = data // and other rows have data
            );
         )
      )

   );
```

OLE Automation

Most of JMP can be driven through OLE automation. Please see the Automation Reference.pdf at https://www.jmp.com/en_us/support/jmp-documentation.html for more information about automating JMP. This document introduces how to automate JMP through Visual Basic and using Visual C++ with MFC. It also contains more information about the methods and properties that JMP exposes to automation clients like Visual Basic and Visual C++.

The JMP Samples/Automation folder contains several sample Visual Basic .Net, and Visual C# .Net files.

Share Reports on JMP Live

Write JSL scripts to automate the generation of JMP analyses and publish them to JMP Live.

- "JMP Live Prerequisites"
- "Get an API Key"
- "Publish a Single Report to JMP Live"
- "Publish Several Reports to a JMP Live Folder"
- "Delete Reports in a JMP Live Folder"
- "Replace the Contents of a JMP Live Report"
- "Search for a JMP Live Report or Folder ID"
- "Update Data in JMP Live Reports"
- "Work with Other Content in JMP Live Reports"
- "HTTP Requests in JMP Live"

JMP Live Prerequisites

Before you can use JSL to publish reports to JMP Live, do the following:

1. Get an API key (if required by a JMP Live administrator). See "Get an API Key".
2. Establish a connection to a JMP Live server. See *Using JMP*. When you add a JMP Live server, you need to enter your API key (if required).

   When you establish a connection, a JMP Live Connection is created in JSL:

   ```
   liveconnection = New JMP Live( Connection( "My JMP Live" ), Prompt( If Needed
     ));
   ```

   where `"My JMP Live"` is the name of the JMP Live server connection you created. If authentication is required, an interactive prompt appears. You can change the `Prompt` to `No`, but if authentication is required, the connection fails.

Get an API Key

Before you can script reports and publish them to JMP Live, you might need an API key. JMP Live administrators determine whether an API key is required for scripting.

An API key is unique for a user and a JMP Live server. That means that if you obtain a new API key, it overrides the previous one, and any scripts using the previous API key will need to be updated to use the new API key.

To obtain an API key:

1. Log into JMP Live.
2. At the top, click *your user icon* and select **Settings**.
3. Click **Generate API Key**. Make a note of your key.

Publish a Single Report to JMP Live

Publish a single report to JMP Live using a JMP Live Report.

The following example creates a Bivariate report and an empty JMP Live Report called `webreport`, adds the Bivariate report to the `webreport`, and publishes the `webreport` to JMP Live:

```
dt = Open( "$SAMPLE_DATA\Big Class.jmp" );
// Create a Bivariate report.
biv = Bivariate( X( :Height ), Y( :Weight ));
// Create an empty JMP Live Report.
webreport = New Web Report();
/* Add the Bivariate report to the webreport. A title is required and a
   description is optional.*/
webreport << Add Report( biv, Title( "Bivariate Report" ), Description( "A
   test of the new scripting" ));
/* Publish the webreport to JMP Live as a private report. Use the JMP Live
   Connection you established and return a JMP Live object called liveresult.
   */
liveresult = liveconnection << Publish( webreport );
```

You can use JSL to make the report visible to everyone or share the report with groups in JMP Live. See the `Set Public` and `Share With Groups` messages in the JMP Scripting Index at Help > Scripting Index.

Work with a JMP Live Result

The `Publish` function returns a JMP Live Result (`liveresult`). Use JSL scripting functions on a JMP Live Result to get information, such as whether the publishing succeeded or failed, failure codes, and more:

```
// Return the outcome of the last action (1 for success, 0 for failure).
liveresult << Succeeded ();
// Return any error messages from the last action.
liveresult << Get Error Message();
```

The JMP Live Result can be made scriptable and turned into a JMP Live Report (for a single report) or a JMP Live Folder object (for multiple reports) so that you can send commands to it. The following example changes a JMP Live Result into a scriptable JMP Live Report and then changes the report access from private to public:

```
livereport = liveresult << As Scriptable();
livereport << Set Public( 1 );
```

See the JMP Live Report object in the JMP Scripting Index at Help > Scripting Index.

Publish Several Reports to a JMP Live Folder

Publish several reports to JMP Live by creating a JMP Live Folder and adding the JMP Live Report to the folder. A JMP Live Report object can hold a single report or multiple reports.

The following example creates a scriptable JMP Live Folder and a JMP Live Report called `webreport`, adds a Bivariate and a Oneway report to the `webreport`, and adds the reports to a folder:

```
/* Create a folder in JMP Live. A title is required and a description is
   optional. Use the JMP Live Connection you established and return a JMP Live
   object called liveresult. */
liveresult = liveconnection << Create Folder( Title( "Test Folder"
  ),Description( "Folder created for script testing" ));
// Make the liveresult scriptable.
folder = liveresult << As Scriptable();
// Create an empty JMP Live Report.
webreport = New Web Report();
// Add a Bivariate report to the webreport.
webreport << Add Report( biv, Title( "Bivariate Report" ));
// Add a Oneway report to the webreport.
webreport << Add Report( oneway, Title( "Oneway Report" ));
// Add the reports to the folder.
liveresult = folder << Add Reports To Folder( webreport );
```

You can use JSL to share the folder with JMP Live groups. See the `Share With Groups` object in the JMP Scripting Index at **Help** > **Scripting Index**.

Delete Reports in a JMP Live Folder

You can delete reports and folders using JSL. Folders must be empty before they can be deleted.

```
returncode = liveconnection << Delete Report( report << Get ID() );
returncode = liveconnection << Delete Folder( folder << Get ID() );
```

In both of these cases, a value of 1 (True) or 0 (False) will indicate if the operation was successful.

Replace the Contents of a JMP Live Report

You can replace the contents of an existing report rather than creating a new report with updated content. You need to know the ID of the report you want to replace. If you have the report object available, you can use the `Get ID( )` function to return the ID.

The following example creates a new JMP Live Report called `webreport2`, adds an updated Oneway report, and replaces an existing Oneway report using the report ID:

```
// Create an empty JMP Live Report.
webreport2 = New Web Report();
/* Add a Oneway report to webreport3. A title is required and a description is
   optional. */
webreport2 << Add Report( oneway, Title( "New Oneway Report" ), Description(
   "Replace a report" ));
/* Use the JMP Live Connection you established to replace the report with the
   specified ID with webreport2. */
liveconnection << Replace( webreport2, ID(report << Get ID() ));
```

Search for a JMP Live Report or Folder ID

For many JSL functions, you need to know the report or folder ID. Using JSL, you can search by keyword (in the title or description) or by the publisher of the report or folder. If your search may return many results, you can specify how many results to return at a time.

The following example searches all reports that contain the word "Production" in the title or description, and are published by a user with the display name "dianesmith". It also specifies that the search will return up to ten items at a time.

```
/* Search for reports with the keyword Production that are published by the
   user dianesmith. Return no more than 10 results at a time. */
liveresult = liveconnection << Find Reports( Search( "Production" ),
   Publisher( "dianesmith" ), ( PageSize( 10 ));
// Change the liveresult into JMP Live Reports and make them scriptable.
livereports = liveresult << As Scriptable();
// Show how many results are returned.
show( livereports << Get Number Of Items());
// Return the first report in list of results.
report1 = livereports[ 1 ];
// Show the title of the first report.
show( report1 << Get Title());
```

In the example above, if you might have more than ten results, you can page through them using the Next and Previous functions:

```
// Put the next 10 results into the next page.
nextpage = livereports << Next();
```

```
// Show the current page number.
show( livereports << Get Current Page Number());
// Get the first item in the list of items in the next page.
next1 = nextpage[ 1 ];
// Show the title of the first report.
show( next1 << Get Title());
// Show the number of items in the current page.
show( nextpage << Get Number Of Items());
```

If `Next()` finds no further pages, an error appears, which you should catch to proceed with further processing.

Tip: If you know the number of pages to expect, you can jump directly to a page. See the `Get Page` function in the JMP Scripting Index at **Help** > **Scripting Index**.

Update Data in JMP Live Reports

If the data behind a JMP Live report changes, you can update the data and regenerate the report. You can update the data for a standalone report (using the report ID) or for all the reports in a folder (using the folder ID).

The following example updates data for an individual report. Suppose that in the Big Class.jmp data table, a subject's age changed and you want to update an associated age report on JMP Live.

```
dt:Age[1] = 77;
/* Use the JMP Live Connection you established to update the data for the
   report with the specified ID based on the updated Big Class.jmp data table.
   */
updateresult = liveconnection << Update Data( report << Get ID( ), Data( dt,
   "Big Class.jmp" ));
// Show the outcome of the update.
show( updateresult << Succeeded );
```

Similarly, you can also specify the folder that the report resides in:

```
updateresult = liveconnection << Update Data( folder << Get ID( ), Data( dt,
   "Big Class.jmp" ));
```

Only reports in the folder using the specified data table update.

Work with Other Content in JMP Live Reports

Primarily in JSL, you work with JMP Live folders and reports. However, you can publish images to JMP Live reports. In JSL, you can use a generic object called a JMP Live Post to manage images or other content that does not fall under reports or folders. See the `JMP Live Post` function in the JMP Scripting Index at **Help** > **Scripting Index**.

HTTP Requests in JMP Live

To invoke HTTP methods in JMP Live, use `Get HTTP Request()`:

```
lc = New JMP Live(Connection("JMP Live Connection"), Prompt(If Needed));
hrequest = lc << Get HTTP Request();
hrequest << URL(url);
hrequest << Method("GET");
data = hrequest << Send();
json_data = Parse JSON(data);
show(json_data);
```

For more details, see "Communicate with REST Web Service".

Communicate with REST Web Service

Representational state transfer (REST) or RESTful web services is a way of providing interoperability between computer systems on the Internet. Use JSL to communicate with a RESTful application program interface (API) that uses HTTP requests to GET, POST, PUT, and DELETE data. For example, you can get data from a web service and return it as a data table. Some web services are Amazon Web Services (AWS), DropBox, and Text-Processing.com.

Notes:

- Strings that contain "&" are truncated. "+" is converted to a space.
- `New HTTP Request()` supports redirects, for example, from HTTP to HTTPS.

Examples of HTTP Requests

The following examples communicate with the National Aeronautics and Space Administration (NASA) API. You must request a key from https://api.nasa.gov/index.html#apply-for-an-api-key and assign it to the `key` variable to run the scripts successfully.

Write Data to the Log

```
url = "http://technology.nasa.gov/api/query/patent/all";
request = New HTTP Request( URL( url ), Method( "Get" ) );
data = request << Send();
Write( data );
   {"results":[["59fa04859600022b4d2e076f","LAR-TOPS-48","All-Organic
      Electroactive Device","NASA's Langley Research Center offers you an
      all-organic electroactive device system fabricated with single-wall
      carbon nanotube (SWCNT)....}
   // JSON output written to the log
```

Notes:

- The query string is at the end of the URL, following "?". For example, in https://www.google.com/search?q=jmp, "q=jmp" is the query string. `New HTTP Request()` automatically URI-encodes key/value pairs in a query string (and FORM).
- When you put an associative array in either a query string or a form, the key/value pairs get URI encoded. Spaces, ampersands, and other characters get encoded to ASCII, and the ampersand is escaped. The encoding is a "%" encoding.
- The web service defines the valid key/value pairs that it accepts. Refer to the API documentation for valid key/value pairs.
- For Kerberos authentication, specify the password as follows:

  `Password(":")`

Return Data as an Associative Array and Write it to the Log

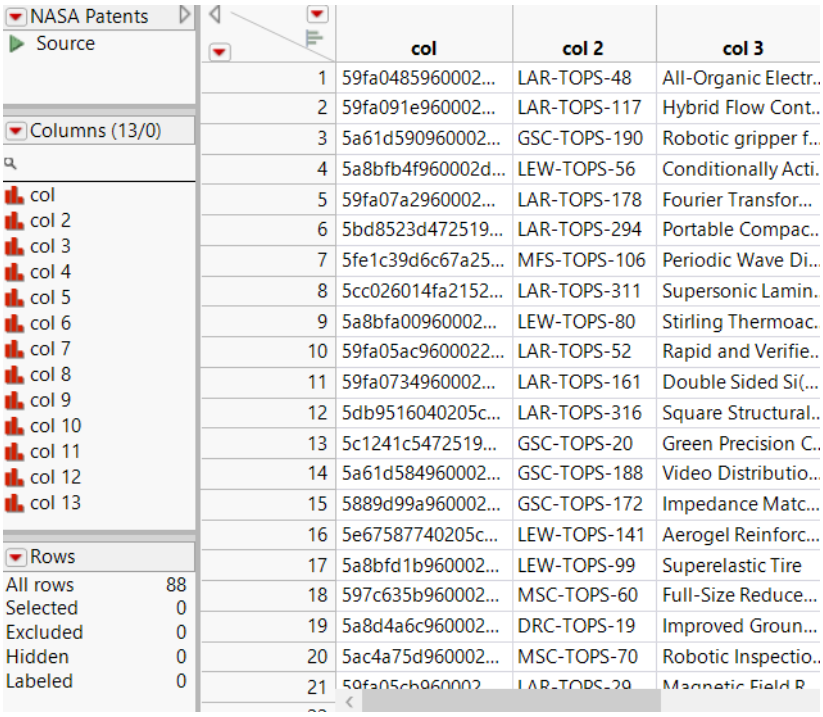
```
url = "http://technology.nasa.gov/api/query/patent/all";
request = New HTTP Request( URL( url ), Method( "Get" ) );
json = request << Send();
jsl_json = Parse JSON( json );
// turns JSON data into a JSL associative array
results = jsl_json["results"];
Write( results );
   {{"59fa04859600022b4d2e076f", "LAR-TOPS-48", "All-Organic Electroactive
      Device", "NASA's Langley Research Center offers you an all-organic
      electroactive device system fabricated with single-wall carbon nanotube
      (SWCNT)....}}
   // associative array
```

Return Data as a JMP Data Table

```
url = "http://technology.nasa.gov/api/query/patent/all";
request = New HTTP Request( URL( url ), Method( "Get" ) );
json = request << Send();
jsl_json = Parse JSON( json );
json = As JSON Expr( jsl_json["results"] );
dt = JSON To Data Table(
```

```
    json,
    JSON Settings(
       Stack( 0 ),
       Col(
          "/root",
          Column Name( "root" ),
          Fill( "Use Once" ),
          Type( "Pandas Values" ),
          Format( {"Best"} ),
          Modeling Type( "Continuous" )
       )
    )
  );
  dt << Set Name( "NASA Patents" );
```

Figure 14.5 Data Imported in a Data Table



OAuth 2.0 for Web APIs

OAuth 2.0 is an authorization standard for accessing personal user information through web APIs. An authorization standard is an agreed upon set of rules by many major web companies that are responsible for authorizing who can access information, and what scope of information they can access. This authorization is granted in the form of a limited time access *token* (a unique identifier specific to the individual and the scope of information they can access). Due to this two-pronged approach of limiting both users and scopes, OAuth is the current standard for securing personal information. Web sites such as Google, Salesforce, Reddit, and Dropbox use OAuth 2.0.

Since OAuth secures your personal information, it can appear to be fairly complex. OAuth is even more complicated due to the variety of ways it is implemented, which are called *grant types*. We'll walk through the different ways you can get an OAuth token in JSL. Please note that not all methods will work with all companies, so you'll have to do a little research to find out which grant type to use. Usually, the company will have documentation of this either with their APIs or possibly in a separate OAuth section.

We recommend that you familiarize yourself with the OAuth protocol at the official website (https://oauth.net/2/).

Note: Only OAuth 2.0 is supported in JSL. OAuth 1.0 and 1.0a are still used by a handful of companies (such as Twitter) but are largely regarded as inferior to 2.0.

Security

OAuth tokens can contain sensitive information. Depending on the grant type, you might need to provide your password or a client secret. JMP automatically handles this information for you.

Information that Google uses to authenticate your account is stored in C:/Users/<user name>/AppData/Roaming/SAS/JMP/OAuthTokens.jmp. (Your Google password is not stored.) After using an OAuth token on your personal account, you can delete all columns except for User and Client ID. Keep in mind that if you need to share your script, others won't have the User and Client ID columns saved on their machine. We recommend moving this information to an encrypted JSL script before sharing the OAuth script.

Using Postman for Redirect URLs

JMP does not provide a redirect URL for free. However, using Postman, it is easy to set one up yourself. Some examples that follow use Postman to redirect requests.

1. Go to https://www.getpostman.com.

2. Click the orange sign-in button in the top right corner.
3. Sign in with your Google account, or proceed to create your own account.
4. You can then use https://app.getpostman.com/oauth2/callback as a redirect URL.

Authorization Code Grant

The Authorization Code grant usually sends *refresh tokens*. Refresh tokens are unique in that they act as a sort of pseudo, permanent access token. Although you won't access sensitive information with them, a refresh token is all you need to get more access tokens. Treat it like you would your password.

The following code snippet creates an OAuth token using the Authorization Code grant. Notice this requires `Request Auth()`, `Client Secret()`, and `Redirect URL()`. Remember, after you make the token once, you can remove everything but the User and Client ID from OAuthTokens.jmp.

```
token = New OAuth2 Token(
  User( "Test User" ),
  Client ID( "12ab" ),
  Client Secret( "3456dEfG" ),
  Request Auth(
     Scope( "history" ),
     Auth URL( "https://example.com/services/oauth2/authorize" ),
  ),
  Redirect URL( "https://www.getpostman.com/oauth2/callback" ) ,
  Token URL( "https://example.com/services/oauth2/token" )
);
```

Note: See your API documentation for more information about how to get values such as the client secret and token URL.

After you run this code once, clean it up and make it secure.

```
token = New OAuth2 Token(
  User( "Test User" ),
  Client ID( “12ab” ),
);
```

Existing Refresh Tokens

You might already be familiar with OAuth and have saved a refresh token to your script. The following snippet creates an OAuth token with that existing token.

```
token = New OAuth 2 Token (
  User( "yourgoogleaccount@gmail.com" ),
```

```
  Refresh Token( "1a2b3c4e5F" ),
  Token URL( "https://www.example.com/oauth2/token" ),
  Client ID( "12ab" ),
  Client Secret( "3456dEfG" )
);
```

After you run this code once, clean it up and make it secure.

```
token = New OAuth2 Token(
  User( "yourgoogleaccount@gmail.com" ),
  Client ID( "12ab" )
);
```

Implicit Grant

The Implicit grant type doesn't require any sensitive information. If you're going to share a script with other people, it would be easiest to use this grant type. Note, however, that because sensitive information isn't required, services are less likely to implement this flow.

The following code snippet creates an OAuth token using the Implicit grant. This requires `Redirect URL()`.

```
token = New OAuth2 Token(
  User( "Implicit Flow"),
  Client ID( "123abc" ),
  Redirect URL( "https://www.getpostman.com/oauth2/callback") ,
  Token URL( "https://example.com/services/oauth2/token" )
);
```

Resource Owner Grant

The Resource Owner grant is the only type that doesn't require a redirect URL. If setting up Postman (the API development platform) is proving difficult, try this grant type.

The following snippet will create an OAuth token using the Resource Owner grant. This requires `Client Secret()` and `Password()`.

```
token = New OAuth2 Token(
  User( "yourgoogleaccount@gmail.com" ),
  Password( "wordP@ss0192" ),
  Client ID( "1234abcd" ),
  Client Secret( "5678EfGh" ),
  Token URL( "https://example.com/services/v1/token" )
);
```

After you run this code once, clean it up and make it secure.

```
token = New OAuth2 Token(
```

```
    User( "yourgoogleaccount@gmail.com" ),
    Client ID( "1234abcd" ),
);
```

Using a Token

In order to access a web API that requires OAuth, pass the token to `HTTP Request()` using the OAuth2 field.

```
request = HTTPRequest(
    URL( apiEndpoint ),
    Method( "POST" ),
    OAuth2( token ),
);
```

OAuth 2.0 Resources

The official website for OAuth 2.0 is https://oauth.net/2/.

https://auth0.com/docs/protocols/oauth2 provides more information about the OAuth 2 protocol.

The OAuth 2.0 standard, RFC6749, is at https://tools.ietf.org/html/rfc6749.

Google provides comprehensive documentation and examples of OAuth 2.0 at https://developers.google.com/identity/protocols/OAuth2.

Communicate with SAS Cloud Analytic Services

SAS Viya uses Cloud Analytic Services (CAS) code to perform calculations in the cloud. This feature provides a way for SAS Viya users and JMP users to collaborate. You can write CAS code in JMP to import data from and export data to the CAS server, convert CAS JSON text to a JMP data table, get a list of available CAS libraries, and more.

- See the SAS CAS documentation at https://developer.sas.com/apis/cas/rest/current/apidoc.html for more information.
- The SAS CAS documentation for CAS actions cover CASL, R, Python, Lua.

This section covers the following topics:

- "About CAS Actions"
- "Connect to the CAS Server"
- "Create and Submit a CAS Action"
- "Create and Submit a CAS DATA Step Action"
- "Create a New CAS Server"
- "Import Data from the CAS Server"
- "View a CAS Dataset"
- "Get a List of Available CAS Libraries"
- "Export Data from a Data Table to a CAS Server"
- "Convert a CAS Table to a Data Table"
- "Delete a CAS Table"
- "Remove a CAS Table"
- "Verify the CAS Server Connection"
- "Get Details about the Current Connection"
- "Disconnect from the CAS Server"
- "Terminate All CAS Sessions"

About CAS Actions

The smallest unit of work for the CAS server is a CAS action. CAS actions can load data, transform data, compute statistics, perform analytics, and create output.

You configure each action by specifying a set of input parameters. Running a CAS action on the CAS server processes the action's parameters and the data, and creates an action result. See *CAS Actions* at https://developer.sas.com/apis/cas/actions.html.

The JSON parameters are specified in the CAS documentation *CAS REST API*. See https://developer.sas.com/apis/cas/rest/current/apidoc.html. For JSON, you can look at the Lua format and make the appropriate adjustments.

For example, in Figure 14.6, notice the difference between the JSON syntax and the Lua syntax. JSON uses the `"name" : value` format. Lua uses the `name = value` syntax.

Figure 14.6 Differences Between JSON and LUA

JSON Syntax

```
{
    "attrTable": "string",
    "attrs": [ {
        column": "string",
        key": "string",
        value": "string" |
                number
}, {...} ],
...
}
```

LUA Syntax

```
{
    attrTable="string",
    attrs={ {
        column="string",
        key="string",
        value="string" |
              64-bit integer |
              integer |
              double |
             binary-large-object
      }, {...} },
,,,
}
```

Connect to the CAS Server

You can show the login window and prompt the user to provide the URL for the CAS server and their login information.

```
CAS Connect( Prompt( "Never"|"Always"|"IfNeeded" ) );
```

To add the login details to the script, follow this example:

```
CAS Connect( URL( URL ), Username( username ), Password( password ) );
```

Information about the CAS server appears in the log after the user logs in.

To specify a trusted certificate for https connections to the CAS server, use the optional Certificates argument.

```
CAS Connect( URL( URL ), Certificates( string ) );
```

To specify whether a certificate is verified, add either `Verify Certificates()` or `No Verify Certificates()` to the script:

```
Verify Certificates( 0|1 )
No Verify Certificates( 0|1 )
```

To specify a proxy server, add `Proxy Server()` to the script:

```
Proxy Server( "https://mycompany.com:80" )
Proxy User( username )
```

To bypass a proxy server, add `Bypass Proxy()` to the script:

```
Bypass Proxy( "https://localhost:80" )
```

Create and Submit a CAS Action

```
New CAS Action( Action( "builtins.echo" ), JSON( echo ) );
```

`Action` is the JSL variable name. builtins.echo is the name of the CAS action that SAS created. `echo` contains the JSL arguments.

Create and Submit a CAS DATA Step Action

```
New CAS DATA Step action( Code( code ) );
```

code defines the contents of the DATA step. Action is the JSL variable name.

```
cas = Current CAS Connection();
code =
"\[
   data temp;
   x = 9.1; y = 6; z = sqrt(x**2 + y**2);
   A = "SAS"; B = "Statistics";
   put _ALL_;              /* display all variables and values */
   run;
]\";
action = New CAS DATA Step action( Code( code ) );
cas << Submit( action );
```

Create a New CAS Server

`New CAS Server()` creates a new CAS server.

```
url = "URL";
cas = New CAS Server( Connect( URL( url ), Prompt(
   "Never"|"Always"|"IfNeeded" ) ) );
```

Import Data from the CAS Server

`CAS Import Data()` imports data from the CAS server:

```
CAS Import Data( libref.dataset, <Private|Invisible>, <"Use Labels for Var
   Names">);
```

- `invisible` hides the data table from view but shows it in the JMP Home Window. `private` hides the data table completely. Specify `private` if the user doesn't need to interact with the data table.
- `"Use Labels for Var Names"` indicates that the table name and the column names will be set to the SAS dataset label and the column labels. The appropriate column property, SAS Name SAS Label, is used.

View a CAS Dataset

CAS datasets are on the CAS file system. Datasets in memory are not listed.

To see the datasets on the CAS server that you want to import, use `CAS Get Data Sets()` after you connect to the server.

The following example shows how to list datasets that are only in casuser:

```
datasets = CAS Get Data Sets( casuser ); // quote the argument
Show( datasets );
```

The following example shows how to lists datasets that you have permissions to view:

```
datasets = CAS Get Data Sets();
Show( datasets );
```

Get a List of Available CAS Libraries

Libraries are namespaces, just like SAS libraries, librefs, or libnames. Use `CAS Get Libraries()` to get libraries from the CAS server.

```
libraries = CAS Get Libraries();
Show( libraries );
```

Export Data from a Data Table to a CAS Server

`CAS Export Data()` exports a data table to a CAS server to run an action on it.

```
CAS Export Data(
  Open( "$SAMPLE_DATA/Big Class.jmp" ),
  "casuser",
  "Big Class"
);
```

`CAS Export Data()` also has two arguments:

- `Save(0)` saves the data in memory.
- `Save(1)` saves the data to the file system.

Convert a CAS Table to a Data Table

`CAS Table to Data Table()` converts JSON data to a data table. A JSON element in the results of an action contains "`_ctb`". This is a marker indicating that it is formatted to represent a CAS table. You can send the JSON results to `CAS Table To Data Table()`. JMP will create a JMP data table from the JSON.

If there's more than one result, you might need to perform a little extra work as shown with `Get Results` at the end of the following example. This example shows how to export data, run an action, and save the data in several data tables.

```
cas = Current CAS Connection();
If( IsEmpty(cas),
   cas = CAS Connect( URL( url ), Username( username ), Prompt( "IfNeeded"
   ));
);
dt = Open( "$SAMPLE_DATA/Animals.jmp" );
cas << Export Data( dt, "CASUSER.Animals" );
json = "\[
{
  "table": {
   "name": "Animals",
   "caslib": "CASUSER"
  },
  "class": [
    "species",
    "subject",
    "season"
  ],
  "model": {
    "depVar": "miles",
    "effects": [
      {
        "vars": [
          "species",
          "season"
        ],
        "interaction": "BAR"
      }
    ],
    "printsol": true,
    "cl": false,
    "dfmethod": "RESIDUAL"
  },
  "random": [
    {
      "depVars": "miles",
      "effects": [
        {
          "vars": [
            "subject"
          ],
          "nest": [
            "species"
```

```
            ]
          }
        ]
      }
    ],
    "method": "REML"
}
]\";
action = New CAS Action(
   Action( "mixed.mixed" ),
   JSON( json )
);
rc = cas << Submit( action );
if(rc,
   results = action << Get Results;
   keys = results << Get Keys();
   For(i = 1, i <= N Items( keys ), i++,
      dt = CAS Table To Data Table( results[keys[i]] );
   );
);
cas << Disconnect();
```

`Get Results` enables you to examine the results of a CAS action; it's a JSL associative array. `Get JSON` gives you the raw JSON; it's a JSL string.

The JSON data is in the format discussed at https://developer.sas.com/apis/cas/rest/current/apidoc.html. Search for "ResultsTable".

In `CAS Table to Data Table()`, specify the JSON string, open the data table as invisible or private, and control the table name and column names.

```
json = "jsonstring";
CAS Table to Data Table( json, <Invisible(Boolean)|Private(Boolean)>, <"Use
   Labels For Var Names">);
```

- `Invisible` hides the data table from view but shows it in the JMP Home Window. `private` hides the data table completely. Specify `Private` if the user doesn't need to interact with the data table.
- `"Use Labels for Var Names"` indicates that the table name and the column names will be set to the SAS dataset label and the column labels. The appropriate column property, SAS Name SAS Label, is used.

Note: CAS ResultsTable attribute values (peers to "`_ctb`") found in the "`_ctb`" JSON are also set as table variables. The "CREATETIME" attribute creates a character table variable with a locale-formatted timestamp (for example, 04/03/2019 4:38:05 PM).

Delete a CAS Table

`CAS Delete Table()` deletes the underlying file from the CAS file system.

```
CAS Delete Table( table name, <remove> ); // quote the table name
```

The `CAS Delete Table()` function has a remove option, which removes an in-memory table from the CAS server. `CAS Remove Table()` serves the same purpose.

Remove a CAS Table

`Remove CAS Table()` removes an in-memory table from the CAS server.

```
CAS Remove Table( casuser, Big Class ) // quote both arguments
```

Verify the CAS Server Connection

`CAS Is Connected()` determines whether the CAS server is connected.

```
connected = CAS Is Connected();
Show( connected );
  connected = 1;
```

Get Details about the Current Connection

`Current CAS Connection()` obtains details about the current connection: the URL of the CAS server, your user name, and the session ID.

```
connection = Current CAS Connection();
Show( connection );
```

Disconnect from the CAS Server

`CAS Disconnect()` disconnects from the current CAS server and optionally terminates the session. By default, the session is terminated when disconnected.

```
CAS Disconnect(<"Terminate"|"NoTerminate">);
```

Terminate All CAS Sessions

`CAS Terminate Sessions()` ends all sessions owned by the current user except for the current session.

```
CAS Teminate Sessions();
```

You can also specify the session ID to terminate the specified session.

```
CAS Terminate Sessions(session ID);
```

Use `Get Session()` to get the session ID.

```
url = "http://myCASserver"; // quoted string
cas = CAS Connect(
   URL( url ),
   Username( "my_username" ), // quoted string
   Prompt( "IfNeeded" )
);
session_id = cas << Get Session;
```

To terminate the current session, use `CAS Disconnect()`.

Chapter 15

Creating Projects

Organizing Project Files in JMP

A JMP project provides a way to organize files that you use in an analysis. You can add JMP files (reports, data tables, scripts, journals, and so on) and non JMP files, such as Microsoft Word or Adobe PDF files. After you perform an analysis, the data table, report, and graph appear in the project on tabs. You can display multiple reports or graphs, display the project or script editor log, and run scripts from the project. Reports and graphs remain linked to the data table.

Projects help you avoid the clutter of several open JMP windows. You can maximize the project window to have a bigger view of the project. And when you save the project, the state of the project is saved (for example, the open reports and the layout of the window).

Contents

Create a Simple Project 867
Create a Project with a Bookmarks Pane and Log 868
Running Scripts in a Project 870
Create a Bookmark Group 872
Close Tabs in Projects 872
Get a Project 873
Reset the Window Layout 874
Run Startup Scripts to Control Workspaces 875

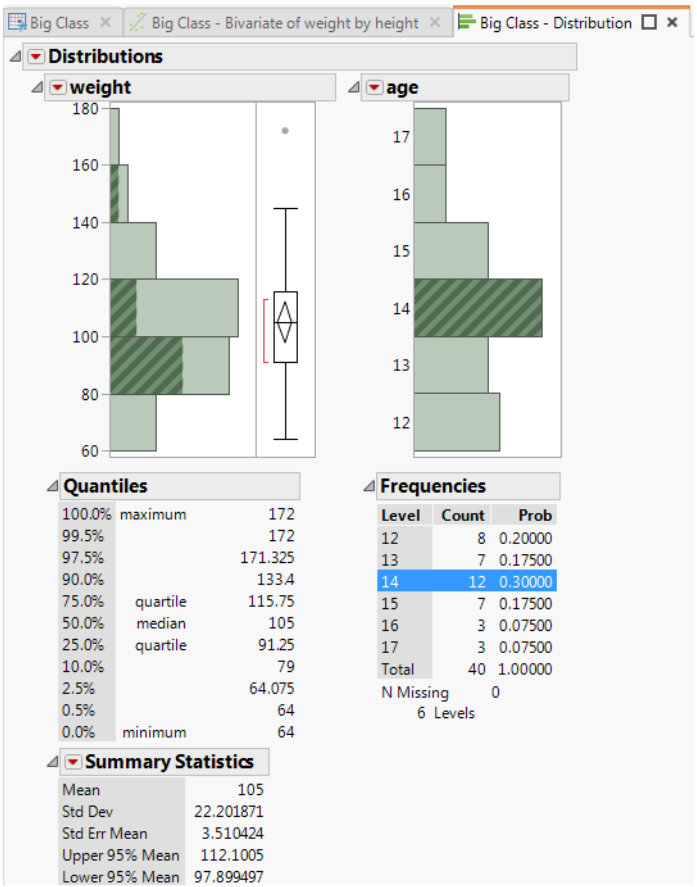
Create a Simple Project

A JMP project provides a way to organize files that you use in an analysis and display the output. A simple project might display the data table, graphs, and reports on separate tabs in the project window.

```
project = New Project();
project << Run Script(
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   dt << Run Script( "Bivariate" );
   dt << Run Script( "Distribution" );
);
```

Figure 15.1 shows the project with selected bars, showing that the graph is still linked to the data table.

Figure 15.1 Example of a Simple Project



Note: The best way to learn how to write a script for a project is to change the project extension to .zip, open the zip file, and then look for a file called project.jsl in the root directory.

Create a Project with a Bookmarks Pane and Log

A more complicated project might include a Bookmarks pane (to show file, folders, and groups that use frequently), and a project log. The data table and output are in separate tabs.

Begin by creating a new project and creating a bookmark.

```
project = New Project();
project << Add Bookmarks({
   File( "$SAMPLE_DATA/Big Class.jmp" )
});
```

`Run Script` opens the data table and creates output inside the project.

```
project << Run Script(
   Open(
      "$SAMPLE_DATA/Big Class.jmp",
      Set Window ID( "data" ) // unique window ID
   );
   Distribution(
      Set Window ID( "distrib" ),
      Continuous Distribution( Column( :weight ) ),
      Nominal Distribution( Column( :age ) )
   );
   Graph Builder(
      Set Window ID( "graphbuilder" ),
      Size( 443, 409 ),
      Show Control Panel( 0 ),
      Fit to Window( "Maintain Aspect Ratio" ),
      Variables( X( :age ), Y( :height ), Y( :weight ) ),
      Elements( Position( 1, 1 ), Line( X, Y, Legend( 1 ) ) ),
      Elements( Position( 1, 2 ), Bar( X, Y, Legend( 2 ) ) )
   );
);
```

`Set Layout` specifies the layout of the tabs that show the data table and output.

```
project << Set Layout(
   V Splitter Box(
      Size( 1215, 700 ),
      <<Set Sizes( {0.77, 0.23} ),
      <<Dockable(),
      V Splitter Box(
```

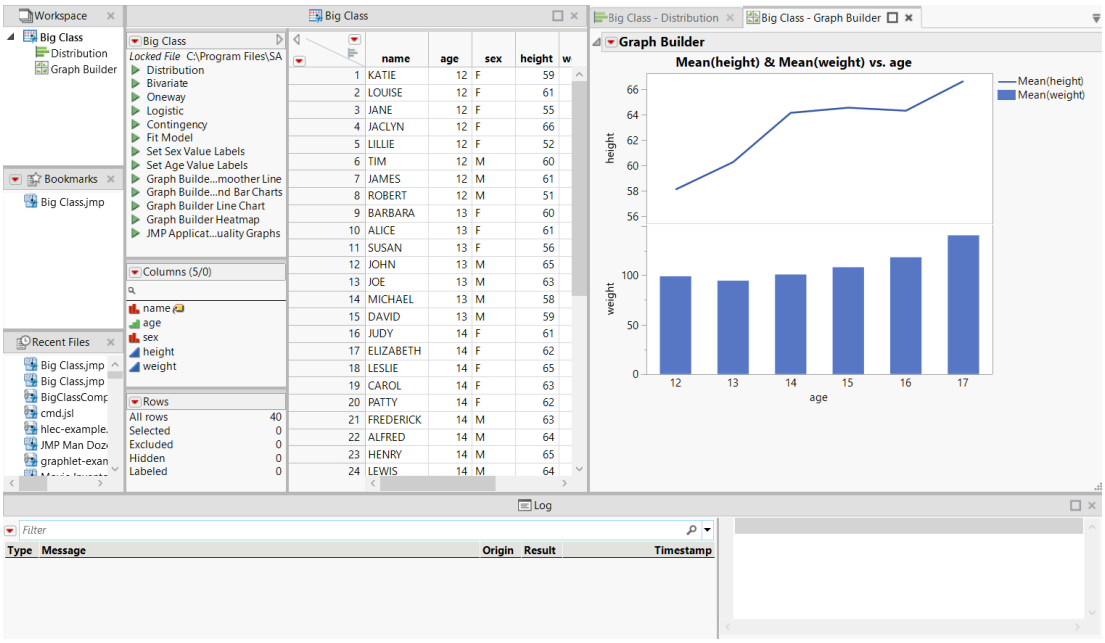
```
            Size( 1215, 540 ),
            <<Set Sizes( {1} ),
            <<Dockable(),
            H Splitter Box(
               Size( 1215, 540 ),
               <<Set Sizes( {0.11, 0.42, 0.47} ),
               <<Dockable(),
               V Splitter Box(
                  Size( 128, 540 ),
                  <<Set Sizes( {0.33, 0.33, 0.33} ),
                  <<Dockable(),
                  Tab Page Box(
                     Title( "Window List" ), // show the Window List
                     Window ID( "Window List" )
                  ),
                  Tab Page Box(
                     Title( "Bookmarks" ), // show the Bookmarks pane
                     Window ID( "Bookmarks" )
                  ),
                  Tab Page Box(
                     Title( "Recent Files" ), // show the Recent Files pane
                     Window ID( "Recent Files" )
                  )
               ),
               Tab Page Box( // show the data table
                  Title( "Big Class" ),
                  Window ID( "data" )
               ),
               Tab Box(
                  <<Dockable(),
                  Tab Page Box( // show the Distribution graph and reports
                     Title( "Big Class - Distribution" ),
                     Window ID( "distrib" )
                  ),
                  Tab Page Box( // show the Graph Builder graph
                     Title( "Big Class - Graph Builder" ),
                     Window ID( "graphbuilder" )
                  )
               )
            )
         ),
         Tab Page Box( Title( "Log" ), Window ID( "Log" ) )
      )
   );
```

Save saves the project.

```
// if the project has already been saved, the Save message
// with no argument saves the file to the existing location
project << Save( "$DOCUMENTS/Big Class.jmpprj" );
```

Note: If you're saving a project with unsaved documents in JSL, the project doesn't close and you get a log message that the project has unsaved documents. All files in a project must be saved or closed before the project can be saved. However, if you interactively save a project with unsaved documents, you are prompted to save the documents.

Figure 15.2 Example of a Complex Project



Open the project from another script:

```
project = Open( "Big Class.jmpprj" );
```

In this example, the script that you are running to open the project is in the same folder as the project.

Running Scripts in a Project

When you run a script outside of a project, the reports generated by the script appear in different windows. When you run a script inside a project, the reports appear in different tabs inside the project.

Running a JSL script outside of a project

Open a Script Editor window by selecting File > New > Script. Paste and run the following script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Run Script( "Bivariate" );
dt << Run Script( "Distribution" );
Wait( 2 );
Get Window( "Big Class - Distribution" ) << Close Window();
```

Note that this script opens three JMP windows: one for the data table and one for each report. Close the reports and then the data table.

Running a JSL script inside of a project

Create a new project by selecting File > New > Project. From the project window, create a new script editor window by selecting File > New > Script. Note that the script editor window opens as a tab in the project. Paste the same JSL script you previously ran into the script editor and run it:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Run Script( "Bivariate" );
dt << Run Script( "Distribution" );
Wait( 2 );
Get Window( "Big Class - Distribution" ) << Close Window();
```

Note that this time, the data table and the reports all opened as tabs in the project, and that `Get Window()` correctly found the distribution tab. In general, all JSL functions that open, find, manipulate, and close JMP windows, tables, and reports work from within a project the same as they would outside projects.

Running a JSL script using a project's Run Script message

You can use the `Run Script` message to run JSL in the project's context. Open a Script Editor window, then paste and run the following code:

```
project = New Project();
project << Run Script(
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   dt << Run Script( "Bivariate" );
   dt << Run Script( "Distribution" );
   Wait( 2 );
   Get Window( "Big Class - Distribution" ) << Close Window();
);
```

Note that the script inside the `Run Script` message runs the same as if it had been in a script editor window inside the project.

Scoping Variables

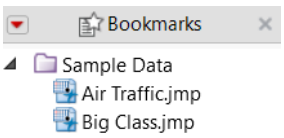
JMP can have several projects open at the same time, as well as items not open in any project. Each project has its own global scope, which prevents conflicts caused by identically named variables in different projects. In the rare case that you do want to share specific variables among all opened projects, use the `:::` root operator.

Create a Bookmark Group

You can create bookmark groups to organize the bookmarks in a project. A bookmark group can contain bookmarked files, folder, and additional bookmark groups. The bookmark group exists only in the project, not on your computer. Use `Group()` to create the bookmark group.

```
project = New Project();
project << Add Bookmarks(
        Group(
           "Sample Data",
           Expanded( 1 ), // open the group
           {File( "$SAMPLE_DATA/Air Traffic.jmp" ),
           File( "$SAMPLE_DATA/Big Class.jmp" )}
        )
);
project << Set Layout(
   H Splitter Box(
      <<Set Sizes( {0.15, 0.85} ),
      Tab Page Box( Window ID( "Bookmarks" ), )
   )
);
project << Save( "$DOCUMENTS/My Project.jmpprj" );
```

Figure 15.3 "Sample Data" Group in a Project



Close Tabs in Projects

Close a tab in a project by identifying the window with `Get Window()` and then closing it with `Close Window`.

```
project = New Project();
```

```
project << Run Script(
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   dt << Run Script( "Bivariate" );
   dt << Run Script( "Distribution" );
);

Wait( 2 ); // for demonstration purposes

// returns a reference to the open "Big Class - Distribution" window
dist = Get Window( Project( project ), "Big Class - Distribution" ) ;
dist << Close Window();
```

To close all tabs in a project, use `Get Window List << Close Window()`.

```
project = New Project();
project << Run Script(
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   dt << Run Script( "Bivariate" );
   dt << Run Script( "Distribution" );
);

Wait( 2 ); // for demonstration purposes

Get Window List( Project( project ) ) << Close Window();
```

Get a Project

When running a JSL script from within a project, you can use the `This Project()` function to obtain that project. The following example displays the name of the current project in a new window.

```
project = This Project();
If( Is Empty( project ),

   /* if the script is not run from a project, display this
   sentence in the new window */
   New Window( "My Project", Text Box( "Script isn't running in a project." )
   ),

   /* if the script is run from a project, display the window title
   in the new window */
   New Window( "My Project", Text Box( project << Get Window Title() ) )
);
```

`Get Project()` takes the same arguments as the `Window()` function. Omit the arguments to get a list of the currently open projects, include a string to find a specific project by title, or include a number to get the *i*th open project.

```
Open( "$SAMPLE_PROJECTS/Big Class.jmpprj" );
Open( "$SAMPLE_PROJECTS/Sports.jmpprj" );

// print the title of Sports.jmpprj
Print( Get Project( 2 ) << Get Window Title() );
```

`Get Project()` is also helpful when you want to save a project or set the layout from within the project.

```
Get Project() << Save(...);
Get Project() << Set Layout(...);
```

Reset the Window Layout

Suppose that you create a project and customize the layout of the windows. For example, you might hide the Window List. Use the `Reset Layout` message to return the windows to their original layout.

```
project = New Project();
project << Run Script( // open data table and create a graph
       Open(
              "$SAMPLE_DATA/Big Class.jmp",
             Set Window ID( "data" )
       );
       New Window( "Big Class - Bivariate of weight by height",
              Set Window ID( "bivariate" ),
              Bivariate(Y( :weight ), X( :height ),
              )
       ) << Set Window Icon( "Bivariate" );
);

Wait(3); // for demonstration purposes
project << Set Layout( // set a custom layout
       H Splitter Box(
              <<Set Sizes( {0.2, 0.8} ),
              Tab Page Box( Window ID( "Bookmarks" ) ),
                V Splitter Box(
                     <<Set Sizes( {0.53, 0.47} ),
                     Tab Page Box(
                         Window ID( "bivariate" ) ),
                     Tab Page Box(
                         Window ID( "data" ) )
```

```
                )
            )
);

Wait( 3 ); // for demonstration purposes
project << Reset Layout(); // return to the original layout
```

Run Startup Scripts to Control Workspaces

A workspace is either a project window or the windows that are not in a project (for example, the home window). A default workspace loads when JMP starts, and then each time you create or open a project, JMP creates another workspace.

Consider writing a workspace startup script to customize variables for a specific project. You might also construct a new customized window in a workspace startup script to put a copy of that window in each project. This is an alternative to having a single copy of the script that runs outside of projects.

The workspace startup script runs each time a new workspace loads. This means that the workspace startup script runs once when JMP starts (when the "default workspace" loads) and then runs again each time a new project is loaded. The workspace startups script runs in the context of the project, so any windows it opens or variables it sets are for that project only.

Name the script workspaceStart.jsl and place it in one of the following folders, as appropriate for your operating system. When JMP starts, JMP looks for the workspaceStart.jsl script in these folders in the order in which they are listed here. The first one that is found is run, and the search immediately stops.

Note: Some path names in this section refer to the "JMP" folder. On Windows, in JMP Pro, the "JMP" folder is named "JMPPro".

On Windows:

1. C:/Users/<username>/AppData/Roaming/SAS/JMP/16
2. C:/Users/<username>/AppData/Roaming/SAS/JMP

On macOS:

1. /Users/<username>/Library/Application Support/JMP/16
2. /Users/<username>/Library/Application Support/JMP

The workspaceStart.jsl script runs only for a particular user on a computer. You can add a script named workspaceStartAdmin.jsl in one of the following places, as appropriate for your operating system. This script is run for every user on a computer. JMP runs workspaceStartAdmin.jsl first if found. Then JMP runs workspaceStart.jsl if found.

On Windows:

1. C:/ProgramData/SAS/JMP/16
2. C:/ProgramData/SAS/JMP

On macOS:

1. /Library/Application Support/JMP/16
2. /Library/Application Support/JMP

Chapter 16

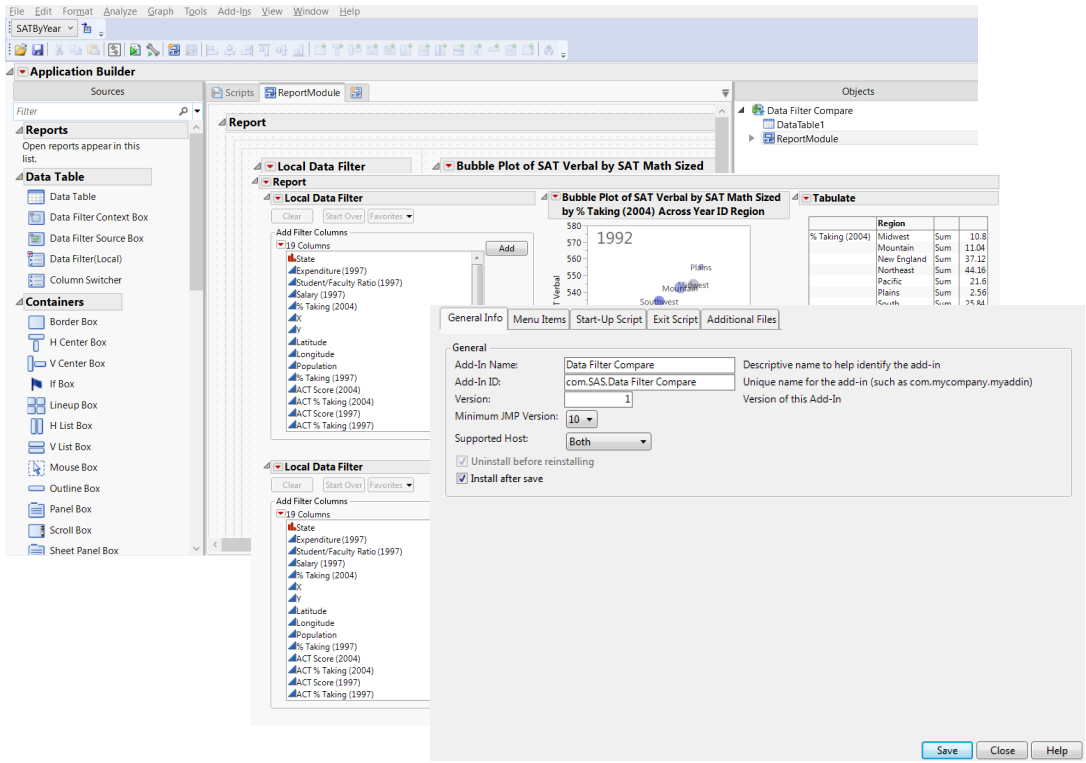
Creating Applications

Application Builder

In addition to providing platforms for quickly analyzing data, JMP lets you create applications to support your specific needs. You often perform the same tasks every day (such as running Distribution and Fit Model analyses on a data table and viewing the results). In JMP's Application Builder, you can create an application that shows the results for both analyses in one window. Application Builder's drag-and-drop interface reduces the amount of scripting required to create an application.

Add-In Builder visually guides you through creating an add-in that users can install in JMP. And after creating an application in Application Builder, use Add-In Builder to save the application as an add-in. This allows users to install the application and quickly run it inside JMP.

Figure 16.1 Creating Customized Reports in Application Builder



Contents

Create Applications with Application Builder 879
Example of Creating an Application 879
Application Builder Terminology 881
Design an Application 882
Application Builder Window 883
Application Builder Red Triangle Options 884
Create an Application 886
Edit or Run an Application 897
Options for Saving Applications 898
Additional Examples of Creating Applications 899
Compile Add-Ins with the JMP Add-In Builder 915
Compile Scripts into an Add-In Using Add-In Builder 915
Edit an Add-In 920
Share an Add-In 920
Installing Multiple Add-Ins 920
Register an Add-In Using JSL 921
Create an Add-In Manually 921
Manage JMP Add-Ins 923

Create Applications with Application Builder

Application Builder's drag-and-drop interface lets you visually design windows with buttons, lists, graphs, and other objects. This saves you the step of writing scripts to create these objects.

Application Builder also helps boost the productivity of experienced JMP Scripting Language (JSL) programmers. You can start designing a program in Application Builder and edit the automatically generated scripts. Integrate your own scripts to create even more powerful custom applications.

You can use Add-In Builder with Application Builder. First, use Application Builder to write complex scripts that create new platforms and interact with users and JMP. Then, use Add-In Builder to easily distribute your complex scripts so that any JMP user can access them without having to open and run the scripts. See "Compile Add-Ins with the JMP Add-In Builder"

Notes:

- Creating simple applications in Application Builder requires little to no JSL programming. However, to create most simple applications, Dashboard Builder might be sufficient. Dashboard Builder provides a subset of features that are found in Application Builder. See *Using JMP*.
- In each version of JMP, Application Builder takes advantages of new features that are not supported in earlier releases. For maximum compatibility, we recommend that you create applications in the earliest version of JMP that the application will be run in.

Example of Creating an Application

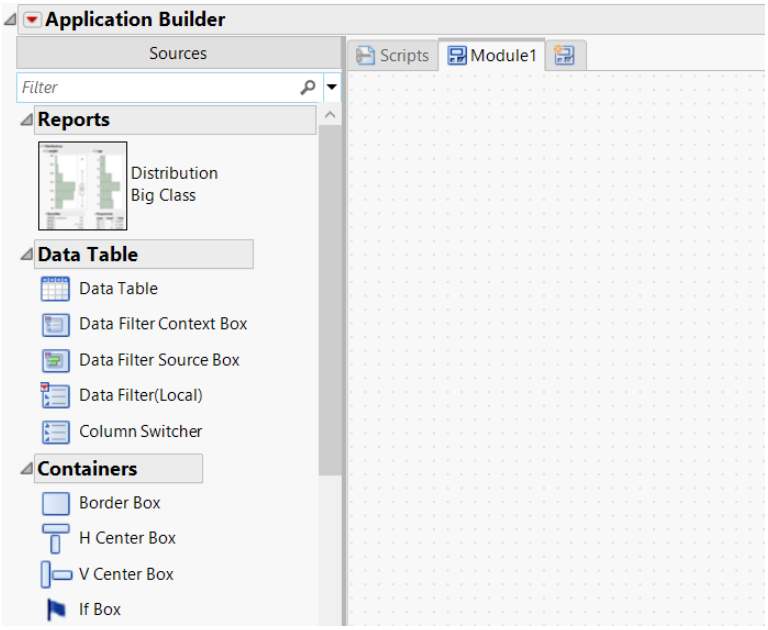
Application Builder helps you create instant applications without writing JSL. One example is a JMP window in which you arrange reports, graphs, and other objects to make them easy to interact with at once.

This example shows how to create a simple instant application for a Distribution report. This application lets you analyze preselected variables rather than having to select them in a launch window.

1. Select **Help > Sample Data Library** and open Big Class.jmp.
2. Run the **Distribution** table script to generate the report.
3. Select **File > New > Application** (or **File > New > New Application** on macOS).
4. Click the Blank Application box.

   A new blank application appears. Notice the Distribution report in the left column.

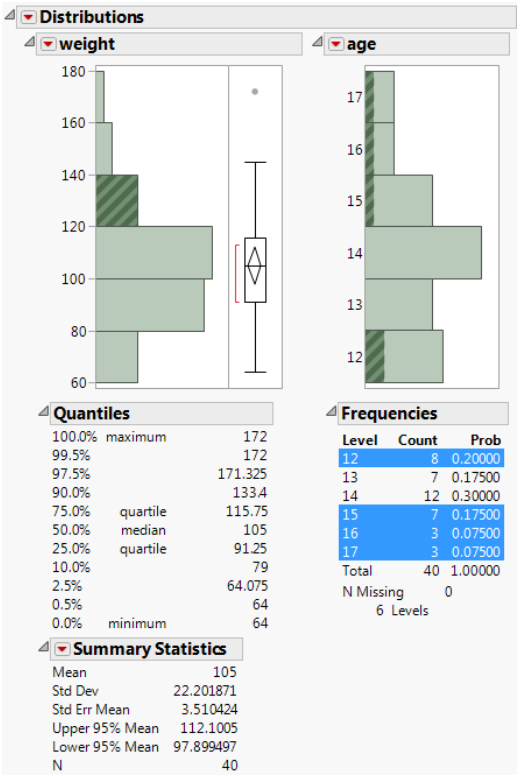
Figure 16.2 New Application with Distribution Report in the Sources Panel



5. In the Sources panel, drag the Distribution report onto the application workspace (the blank area on the Module1 tab).
6. Click the Application Builder red triangle and select **Run Application**.

   The Distribution report appears in a new window. Note that the histograms are interactive and still associated with the data table. When the data changes, you run the application again, and the Distribution report is updated.

Figure 16.3 Example of an Instant Application



	name	age	sex	height	weight
1	KATIE	12	F	59	95
2	LOUISE	12	F	61	123
3	JANE	12	F	55	74
4	JACLYN	12	F	66	145
5	LILLIE	12	F	52	64
6	TIM	12	M	60	84
7	JAMES	12	M	61	128
8	ROBERT	12	M	51	79
9	BARBARA	13	F	60	112
10	ALICE	13	F	61	107
11	SUSAN	13	F	56	67
12	JOHN	13	M	65	98
13	JOE	13	M	63	105
14	MICHAEL	13	M	58	95
15	DAVID	13	M	59	79
16	JUDY	14	F	61	81

Application Builder Terminology

An *application* consists of modules. A *module* contains the objects and scripts that are compiled into an application. When you run an application, JMP can show a separate window for each module. For example, one window might show a **Create Graphs** button. After the user clicks the button, the graphs appear in a new window.

There are terms you should become familiar with before using Application Builder. Knowing these terms helps you understand how a module works, even if you need to minimally edit an automatically generated script.

Application The top-level file that consists of one or more modules.

Module A collection of objects, messages, instances, and other JSL statements that are compiled into an application to create new JMP features.

Instant Application An application that typically consists of reports and does not require JSL scripting. You can also create an instant application, or “dashboard”, in Dashboard

Builder. Similar to Application Builder, Dashboard Builder has a drag-and-drop interface but with pre-configured display boxes.

Custom Application An application that demonstrates custom behavior through JSL scripting.

Objects and Messages An *object* is a dynamic entity in JMP, such as a data table, a data column, a platform results window, a graph, and so on. Most objects can receive messages that instruct the object to perform some action on itself.

A *message* is a JSL statement that is directed to an object. That object knows how to evaluate the message.

Module Instance The occurrence of a module in the application. In a complex application, you might create a start-up script that creates multiple instances of one or more modules.

Namespace A collection of unique variable names and corresponding values that prevents collisions among scripts.

JMP creates the `Application` namespace and the `ModuleInstance` namespace automatically. Symbols in the `Application` namespace are visible only to scripts in the application; the namespace is not available to scripts not created in the application. For more information about namespaces, see "Advanced Scoping and Namespaces".

Variable A unique name to which you assign a value. JMP creates the `thisApplication` and `thisModuleInstance` variables automatically.

- The `thisApplication` variable (used in the `Application` namespace) contains module names, which are available in any script in the application.
- The `thisModuleInstance` variable (used in the `ModuleInstance` namespace) contains box and script names, which are valid only in their own modules.

Container A display box object (such as a tab or outline) into which you drag other objects.

Design an Application

Before creating the application, you must identify its purpose, decide on which graphical elements to include, and figure out which JSL scripts you need to write.

Purpose What do you want the application to accomplish? Which features of JMP do you need to customize (for example, a launch window, report, or graph)?

Graphical Elements How many modules do you need? Look at the list of graphical objects in Application Builder and decide which objects to include in your application. The Sources panel shows dozens of boxes and icons that you can drag and drop into a module. Also look at the preinstalled samples to see whether a similar module exists.

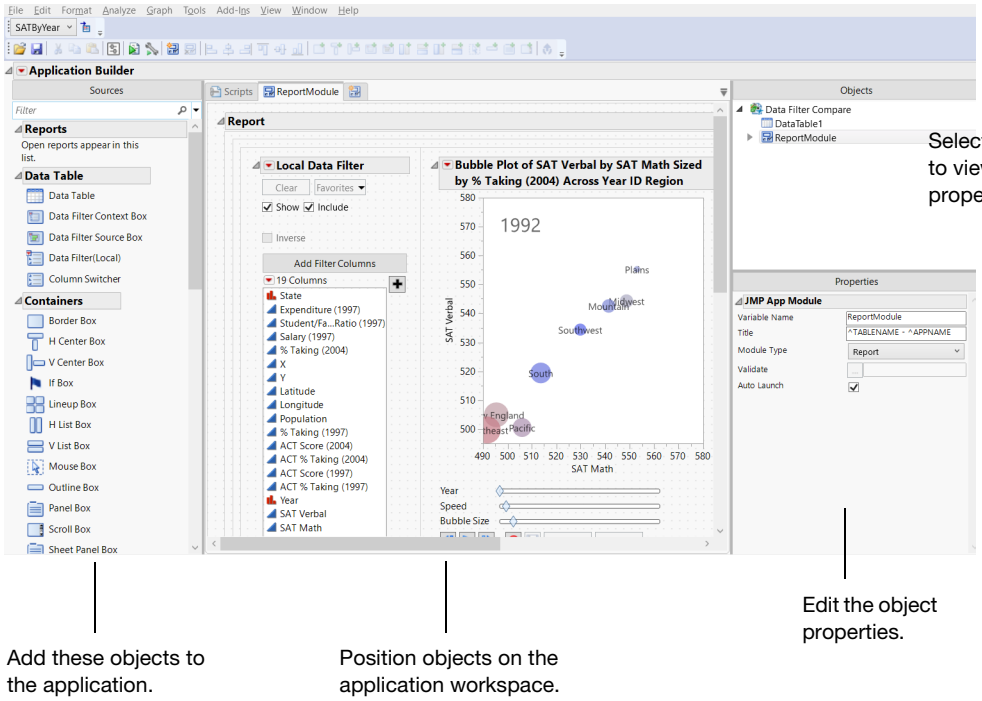
Scripting Depending on the complexity of the application, you write JSL scripts. See "Write Scripts" for more information about writing scripts for an application.

A non-interactive application that only shows objects does not require scripts. One example is a window showing reports that you have already generated in JMP.

Application Builder Window

Figure 16.4 shows an application under development in Application Builder.

Figure 16.4 The Application Builder Window



The Application Builder window provides the following features:

- The toolbar provides quick access to many features, such as aligning objects and inserting common display boxes. (Select **View > Toolbars > Application Builder** to show the toolbar.)
- The Sources panel shows objects that you can include in the application, including open reports and graphs. Right-click an object and select **Scripting Help** for information about the object and its properties.
- The workspace in the middle shows dotted grid lines, which help you line up objects. Drag objects onto one or more Module tabs in the workspace and write scripts on the Scripts tab to control the objects' functionality.

- The Objects panel shows a hierarchical view of the application. Along with the application, each module and its nested objects are listed. You can click an object to show its properties or to select it on the workspace.
- In the Properties panel, you set properties for each object, such as the location, width, or name of an object. The properties vary depending on the type of object.

Tip: To hide the grid and turn off snap-to-grid for all applications that you create, select **File > Preferences > Platforms > JMP App** and deselect these options. You can also deselect these features in the Application Builder red triangle menu.

Application Builder Red Triangle Options

The Application Builder red triangle menu provides options for running and debugging the application, opening sample applications, showing the grid, and more.

Run Application Starts the application. A window for each module opens, and you can interact with the application as the user will.

Debug Application Opens the script in the JSL Debugger to troubleshoot errors. See "Debug or Profile Scripts".

Snap to Grid Aligns (or snaps) objects to the nearest dotted grid lines as you drag them on the workspace. Selected by default.

Show Grid Displays dotted grid lines on the workspace. Selected by default.

Show Properties Shows or hides the Objects and Properties panels.

Auto Scroll Automatically scrolls horizontally or vertically as you drag an object near the edges of the workspace. Selected by default.

Dashboard Mode Displays the application in the simpler Dashboard Builder window. When you drag a report, data filter, or data table to the application, it is automatically placed in splitter and tab page boxes. You cannot add scripts in Dashboard Mode.

Be aware that switching back and forth does not change anything that you have already added to the application. Switching modes changes the way the interface works when you add new content.

Source Panel Shows or hides the Sources panel (the left column in which application components are displayed) and categorizes the icons. Both settings are selected by default.

Save Script Lets you save the application to a data table, journal, script window, template, or add-in. See "Options for Saving Applications".

Table 16.1 Sample Applications Installed with JMP

Six Quality Graphs.jmpappsource	Creates three Control Charts, a Distribution report, and a Capability Analysis report.
Data Filter Compare.jmpappsource	Creates two Bubble Plots. Each plot has its own data filter and Tabulate report. See this application for an example of the `Data Filter Context Box` function.
Data Table Application.jmpappsource	Lets the user select data table columns to stratify and the sampling rate. Display the data table to see the results.
Graph Launcher.jmpappsource	Lets the user enter an equation and axes settings and then create a graph. The initial window remains open, so the user can modify the equation and create new graphs.
Instant App.jmpappsource	Combines two Multivariate reports (Principal Components/Factor Analysis and T Square with All Principal Components).
Instant App Customized.jmpappsource	A modified version of Instant App.jmpappsource. Options for selecting the principal components report, changing the marker size, and showing means are included.
Launcher With Report.jmpappsource	Lets the user select data table columns in a launch window and then creates a graph.
Parameterized Instant App.jmpappsource	Lets the user select data table columns and then creates two Multivariate reports. An argument is assigned to the Y role. This means that the reports can be created from any open data table, not just the table specified in the application.
Parameterized MSA Combo Chart.jmpappsource	Creates a collection of reports for Measurement System Analysis (MSA).
Presentation.jmpappsource	Creates an onscreen presentation, similar to a slideshow, with navigation buttons and an embedded script.
R Application.jmpappsource	Lets the user select columns and then shows multivariate data in the shape of a face (the Chernoff faces). Requires the R TeachingDemos package.

Table 16.1 Sample Applications Installed with JMP *(Continued)*

SAS Application.jmpappsource	Runs a SAS script and then adds the output to a report. Prompts you to log on to a SAS server if you are not already connected.

Create an Application

After writing the application specifications, you create a blank application in JMP and begin adding objects and scripts.

This section provides the basic steps for creating an application:

- "Create a New Application"
- "Arrange and Remove Objects"
- "Customize Object Properties"
- "Write Scripts"

Note: In each version of JMP, Application Builder takes advantages of new features that are not supported in earlier releases. For maximum compatibility, we recommend that you create applications in the earliest version of JMP that the application will be run in.

Create a New Application

To create a new blank application, follow these steps:

1. Select **File > New > Application** (or **File > New > New Application** on macOS).
2. Select the Blank Application.

   The Application Builder window appears.
3. Select **File > Save** and save the file as a JMP Source File. The extension is .jmpappsource.

After you create the blank application, you add open reports, display boxes, and other objects to the workspace. See "Example of Creating an Application".

Manage Modules

When you create a new application, a Module tab appears by default. You can add objects to this module to create interface items. To create additional windows for your application, you can add and customize new modules.

Add Objects to a Module

Follow these steps to add objects to a module:

1. In the Sources panel, select the type of object that you want to add.
2. Drag the object onto the **Module1** tab (or double-click the object).
3. Select the object and update its properties in the Properties panel. See "Customize Object Properties".
4. Add scripts to scriptable objects. See "Write Scripts".
5. (Optional) To prevent a module from launching when you run the application, select the module in the Objects panel and deselect **Auto Launch**. (You might do this while testing one out of multiple modules.)
6. Click the Application Builder red triangle and select **Run Application** to test your application.

   Depending on how the modules are used, it may not be possible to test one module without creating the other modules.

Create a Module

1. Select **Format > Add Module.**
2. In the Properties panel, select the **Module Type**:
   - Dialog
   - Dialog with Menu
   - Modal Dialog
   - Launcher
   - Report
   - Display Box

Tip: When you run an application, the window title is the name of the data table, if applicable, followed by a hyphen and the application name. Change the application name by modifying text in the JMP App object's **Name** property.

Rename a Module

Change the **Variable Name** in the module's object properties.

Delete a Module

Select **Format > Delete Module.**

Delete a Data Table

Select the Data Table object, right-click and select **Delete.**

Note: You cannot delete the data table if it has any associated objects. Delete any objects and then delete the data table.

Modal Dialog Modules

Modal Dialog modules have some special behaviors to be aware of when using them.

- `ret = Module1 << Create Instance()` will not return until the dialog is complete. By this time, the dialog has been destroyed. Therefore, the return value will not be a handle to the Module Instance. Like `New Window()`, the return will instead be of the form `{Button(1 | -1), User Data}`. (1 | -1) indicates whether the OK (1) or Cancel (-1) button was pressed.
- `User Data()` is a new property on the Module Instance. During execution of the dialog, the Module script can set the user data:

  ```
  thisModuleInstance << Set User Data(...);
  ```

  to store something to be parsed by the caller. The user data will be evaluated at the time that it is set. There is also a corresponding `Get User Data()`.
- Like `New Window()`, if there is no OK or Cancel button included, an OK button will be added.
- There is an optional `Validate` script property on the Module Instance. This script is only used for Modal dialogs and behaves like `On Validate()` in `New Window()`. It is called when the OK button is pressed, and returns 1 to accept the input or 0 to disallow the close.

  Tip: If you use the Validate property, you can call `thisModuleInstance << Set User Data()`.

- Unlike other module types, a Modal Dialog will not show until the module script has completed. For other module types, the window is created during the call to `thisModuleInstance << Create Objects`. JMP cannot display a Modal Dialog here because control would stop and there would be no way to initialize the contents of the boxes in the display. Since the window is not created yet, you will not be able to do actions like set the window title or set an on-close script.

Arrange and Remove Objects

Drag objects from the Sources panel onto the workspace or double-click them. A blue border appears around the edge of a selected object. Then you can position the selected object in several ways. You can also change the container of an object or insert an object into a new container.

Tip: To quickly select an object, select it in the Objects panel. This method is particularly helpful when the internal object completely covers the container, as with a border that has an internal text box.

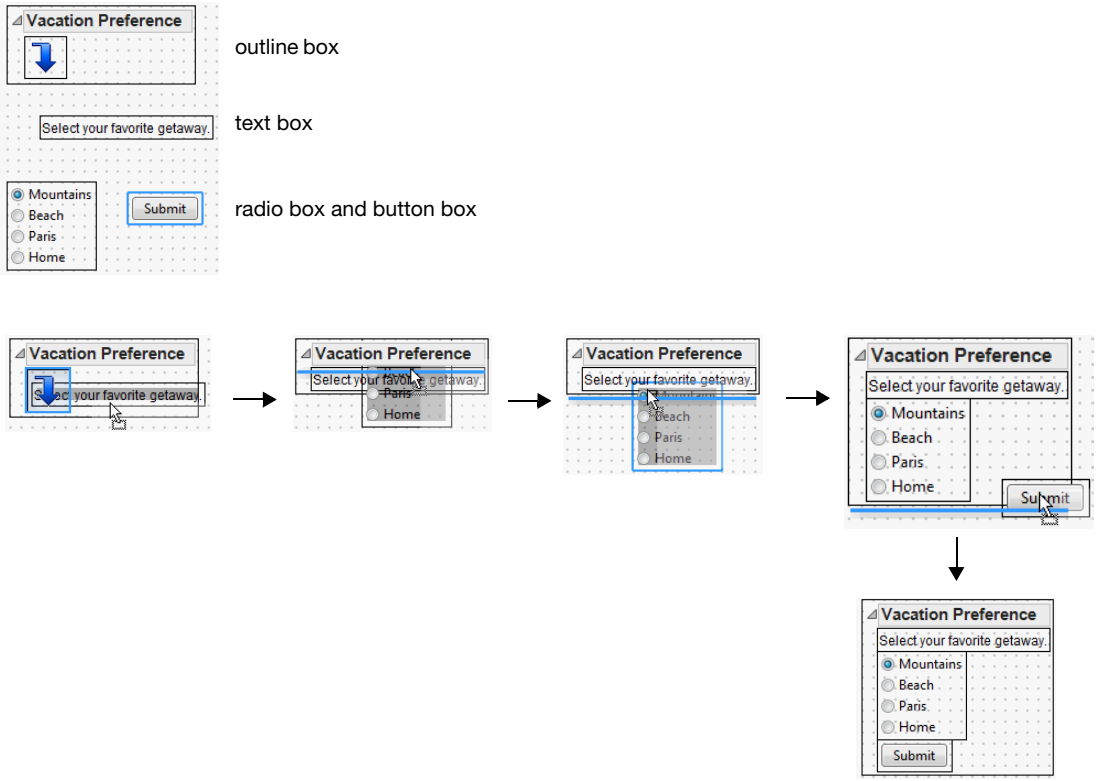
Drag an Object

One way to position objects is by dragging them around the workspace.

- The top and left edges of an object align (or snap) to the nearest dotted grid lines as you drag them on the workspace. For a more precise placement, click the Application Builder red triangle and deselect **Snap to Grid**. You can also turn off **Snap to Grid** for all applications in the JMP Preferences. Select **File (Windows) or JMP (macOS)> Preferences > Platforms > JMP App** to change the option.
- Position objects in containers instead of positioning them independently on the workspace. For example, place two button boxes in a V List Box to align them vertically.
- When you drag an object, press Shift to restrict movement to a single axis (for example, to move the object horizontally without moving the vertical position).
- When you drag an object into a container, the arrow indicates where you can drop the object.
- When you drag an object over another one, a blue line shows where you can drop the object.

Figure 16.5 shows the various methods for dragging objects inside a container.

Figure 16.5 Examples of Dragging Objects into a Container



Change the X and Y Positions on a Container Object

- To position a container object precisely, select the object and then change the **X Position** and **Y Position** properties. After you enter a new X position, press Tab to see the object move to the new position and then enter the new Y position.
- To place an object in the upper left corner, right-click the object and select **Move to Corner**. This is a shortcut to setting the X and Y positions to 0. On macOS, press Ctrl+Command and then select **Move to Corner**.

Line Up Multiple Objects

To line up objects horizontally or vertically, select the objects and then select an option from the **Format > Align Boxes** menu.

Tips:

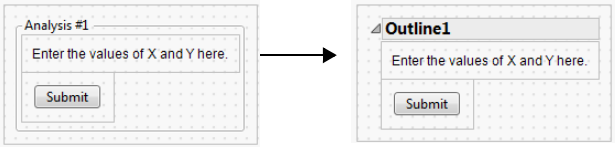
- The **Align Boxes** options are unavailable when you select a container rather than the internal objects.

- If you right-click a container, you might inadvertently select one of the internal objects instead. Right-click in the workspace instead.

Change the Type of Container

After inserting some containers, you can change them to a different type of container without having to re-create the object. Suppose that you create a panel that contains text and a button. To see what the panel looks like as an outline, select the panel, right-click in the module, and then select **Change Container > Outline Box**.

Figure 16.6 Change a Panel to an Outline



In this example, you would also change the outline title to match that of the panel.

Tip: To change the orientation of a list box quickly, select or deselect **Horizontal** in the object properties.

Insert Objects into a New Container

The toolbar at the top of the Application Builder window provides buttons for containers such as border boxes and mouse boxes (Figure 16.7).

To insert selected objects into a container, click the appropriate button on the toolbar or select **Format > Add Container**.

Figure 16.7 Container Toolbar

Duplicate Objects

When you copy and paste an object, the second instance of the object is renamed. The scripts attached to the object are also renamed. Pressing Ctrl (or Command on macOS) and dragging the object also creates duplicates of the originals.

Delete an Object

- Remove one or more objects by selecting them and then pressing Delete.
- Drag the objects outside of the module.
- On macOS, select the objects, press Ctrl+Command, and then select **Delete**.

- On Windows, select **Edit > Clear** to delete all objects in a module.
- Select the object and then press Backspace.

If you no longer need the object's script, you can delete the script also.

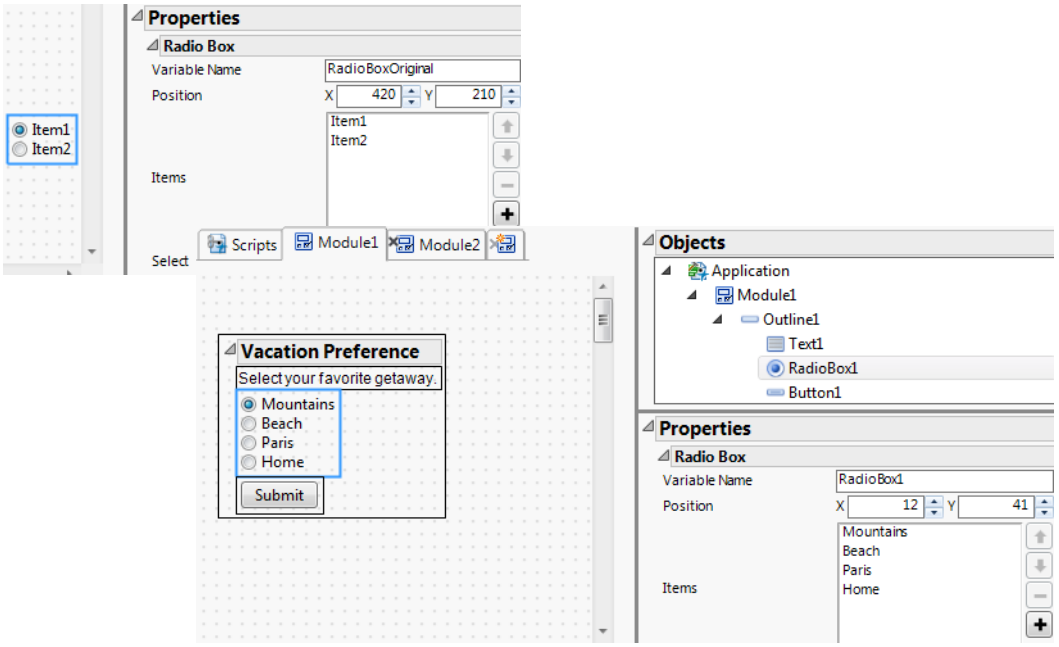
Customize Object Properties

After you drag an object from the Sources panel onto a module, customize the object properties in the Properties panel. Editing the object properties saves you the step of writing JSL for list items, button names, and so on.

Variable names are case- and space-insensitive. When a script contains an object named "Button1", Application Builder warns if you try to rename another object "Button 1".

To create the radio box list items in Figure 16.8, double-click the placeholder items "Item1" and "Item2" and enter the new list item.

Figure 16.8 Radio Box Object Properties



For more information about an object, right-click the object and select **Scripting Help**. The JMP Scripting Index appears, and the object that you selected is highlighted. The Scripting Index often includes a script that you can run to see an example of the object.

To read more about values in the Properties panel, hover over the value.

Application Properties

The Application properties identify the run password, data table name, and more. Select the application in the Objects list.

Name Appears after the data table name in the application's title bar.

Description Enter a description of the application.

Auto Launch Shows a launch window in which the user selects the arguments that have been defined in the application. This launch window is not one of the defined modules; the modules will be instanced based on their own Auto Launch property.

Encrypt Prevents users from editing the application in a text editor. Only applications that the user runs are encrypted (the .jmpapp file and the application in a JMP add-in). Scripts that you save to a data table, journal, and add-in are encrypted. For more information about encryption, see "Encrypt and Decrypt Scripts".

Run Password Enter the password required to run the application. To test the password, run the application outside Application Builder.

Table Module Properties

Inserting the Data Table object creates a data table object. To define properties such as the data table path, select the data table module in the Objects panel. Then modify the following options in the Properties panel.

Variable Name Specifies the name of the data table object. This name appears in the Properties and Objects panels and in the application's JSL script.

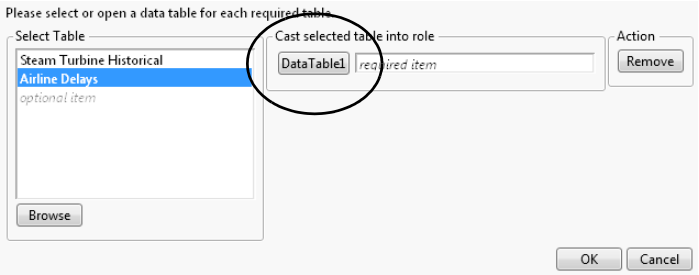
Path Specifies the absolute or relative filepath for the data table used in the application. You can precede the data table name with a path variable (such as $HOME or $USER_APPDATA).

Application Builder opens the specified data table when you edit the application. If the Path property is empty or the data table cannot be found, you are prompted to open the table.

When you close a data table and objects in the application depend on the data table, the objects are removed from the application and a warning message appears. To restore the objects, reopen the application.

Label Specifies the string used when prompting the user to open a data table. Figure 16.9 shows the default value.

Figure 16.9 Label in Data Table Prompt Window



Location Determines how the data table used in the application is selected when the user runs the application.

- Current Data Table: Uses the current data table. If no data tables are open, the user is prompted to open one. This option gives users the flexibility to use their own data.
- Full Path: Uses the data table specified in the Path property.
- Name: Uses the first open data table with the specified name. Otherwise, JMP uses the data table specified in the Path property. Note that if the application is moved to another computer, the link to the data table is broken. Consider storing the data table in a shared location.
- Prompt: Asks the user to select an open data table or browse to select a data table.
- Script: Uses the data table defined in the application or module script.
- Embedded Script: Reads the Source script in the data table that was used to generate the report. The Source script is run when you open the application.

Invisible Hides the data table from view but lists it in the JMP home window. This option is available for the Full Path and Name locations.

Write Scripts

After adding an object to the module, you write a script for the object to provide functionality. For example, you might want the user to click a button, select a directory, and then select a data table. Or you might have an application that displays a different graph based on which radio button the user selects.

Application and Module Namespaces

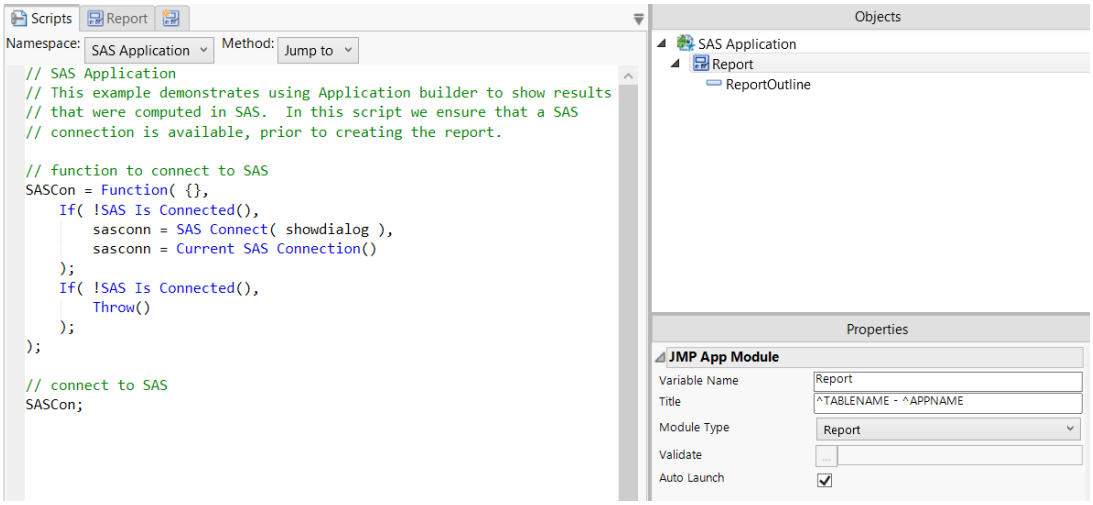
To prevent variable names and values from conflicting among scripts, Application Builder automatically creates the Application namespace and a namespace for each module:

- In the Application namespace, you write scripts that are executed when the user runs the application. Functions defined in the Application namespace can be used in any module.

- In the Module namespaces, you write scripts that are run when an instance of that module is created. If clicking a button in the application opens a new launch window, that launch window is an instance of the module. Two instances of the same module have their own copies of any variables or functions.

To see the scripts in these namespaces, click the **Scripts** tab and then select the namespace in the Namespace list (or in the Objects panel).

Figure 16.10 Application and Module Namespaces



There are two types of scripts: named scripts and anonymous scripts.

Write a Named Script

A *named script* is a function that several controls can use. The `this` argument tells which control is calling the function. In the following example, `Get Button Name` is sent to the `this` argument to print the button name to the log when the button is clicked:

```
Button1Press = Function({this}, Print(this <<Get Button Name))
```

On another button, use `Button1Press` script to produce the same results.

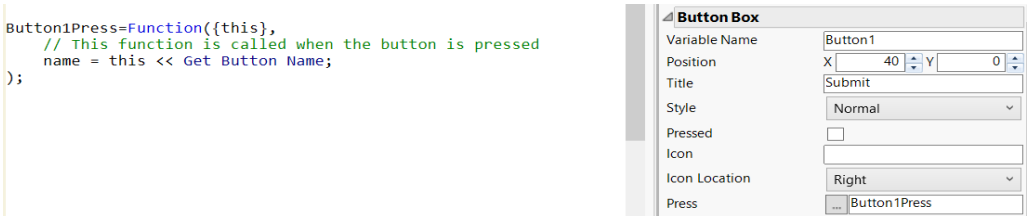
Two other advantages of named scripts: when you add a named script to an object, JMP inserts a placeholder script on the Scripts tab, which you then edit. On the Scripts tab, you can also select a script from the Method list to navigate to that code, which is particularly helpful in long scripts.

Add a named script:

1. Right-click the object, select **Scripts**, and then select the script that you want to add. In this example of creating a button, select the **Press** script. (On macOS, press Control, and then select **Scripts > Press**.)

The object's placeholder script appears on the **Scripts** tab (Figure 16.11). The name of the named script, *Button1Press*, shows up in both the script and the object properties.

Figure 16.11 New Script and Script Properties



2. Edit the object's placeholder script and properties to provide the necessary functionality. For example, the button box shown in Figure 16.11 was renamed. The Title property was changed to *Submit*, the text that appears on the button.

Tip: After you add a script to an object and then delete the object, delete the object's script from the **Scripts** tab if you no longer need the script. This feature prevents scripts that you might want in the future from being deleted.

If you rename the script in the object properties, rename it on the **Scripts** tab also. And if the script is used in another part of the application, rename it there as well.

Write an Anonymous Script

An *anonymous script* is available only to the object that defines it. For example, you might want a simple `Print` statement for one button that is not used elsewhere. By writing an anonymous script, you reduce the number of names to manage in the script. Anonymous scripts also reduce clutter among more important named scripts in the Scripts tab, because you add them to the object properties.

The following examples show two types of anonymous scripts:

```
Print(Button1 <<GetButtonName) // simple anonymous
Function({this}, Print(this <<GetButtonName)) // parameterized anonymous
```

Notice that the simple script sends a message to the "Button1" variable, but `this` is the control in the parameterized script.

Objects such as check boxes might provide additional arguments after `this` that are not otherwise available. One example is an argument that tells which check box in a column of check boxes just changed.

Write an anonymous script:

1. Select the object and click ... in the object properties.

   The anonymous script editor appears.

2. Enter the script and click **OK**.

   The text of the anonymous script appears in the object's properties (rather than the name of a named script).

Tip: Avoid copying and pasting anonymous functions to simplify code maintenance; use a named script instead if the script is needed in more than one place.

Show Specific Scripts

There are several ways to view a script for a specific object:

- Double-click the object on the **Module** tab.
- When an object has multiple scripts, right-click the object, select **Scripts**, and then select the script name. You can also select the **Scripts** tab, select the module name from the **Namespace** list and then select the script name from the **Method** list. Likewise, to see the application scripts, select **Application** from the **Namespace** list.

In each case, the **Scripts** tab appears with the cursor in the first line of the object's script.

Tip: To make a poorly formatted script easier to read, right-click and select **Reformat Script**.

Copy and Paste Objects with Scripts

When you copy and paste an object that has a script, the second instance of the object and script are renamed.

Edit a Platform Script

You can select the object for a platform in an application and edit the script for that platform. For example, you might edit the script for the Multivariate platform to show the scatterplot matrix or to use different columns.

Right-click the platform object on the workspace and select **Edit Platform Script**. After you edit the script and click OK, the object on the workspace is updated.

Edit or Run an Application

To open an application for editing, select **File > Open**, select the .jmpappsource file, and then select **Open**.

Note: If the Table property is empty or the specified data table cannot be found, the application runs anyway but objects requiring the data table will fail to create and JMP displays a warning.

To run an open application, click the Application Builder red triangle and select **Run Application**.

To run a closed application, select **File > Open**, select the .jmpapp file, and then select **Open**.

On Windows, you can also open or run an application from the JMP Home Window by doing one of the following:

- Drag the file from Windows Explorer onto the JMP Home Window or onto a blank application window.
- Double-click the file in Recent Files list.
- Right-click either the .jmpappsource or the .jmpapp file and select either **Edit Application** or **Run Application**.

Options for Saving Applications

JMP provides several options for saving application files.

- When you select **File > Save As** on Windows, you can save the application as an application source file (.jmpappsource), an application (.jmpapp), or a script.
- Selecting **File > Export** on macOS enables you to save the application as an application file or a script. To save the application as an application source file on macOS, select **File > Save As**.

The .jmpappsource file allows an application developer to save the application as is, in order to continue editing the application later. When reopened, the App Builder interface remains in same state as when it was saved. The data tables used to create the application must still be available.

The .jmpapp file is used to distribute applications for others to use. The application runs immediately after opening the file. The application resolves the data tables using the current data tables, files on disk, or by prompting the user, depending on the application settings.

The Save Script red triangle menu provides additional options for saving scripts. When an encrypted script is saved, JMP encloses the script in the `JSL Encrypted()` function to preserve white space and comments.

To Data Table Saves the script to the data table that was used to produce the report. This enables you to run the script again from the data table to recreate the results. When you edit this script from the data table, the application opens in Application Builder, not the script editor.

To Journal Saves a button that runs the script in a journal. The script is added to the current journal. The script contains the path to the data table. Note that if the data table cannot be found, the script does not run.

To Script Window Opens a script editor window and adds the script to it. If you have already saved a script to a script window, additional scripts are added to the bottom of the same script window. Note that if you modify the application significantly in a script window, you might not be able to edit the application in Application Builder. This option is useful when you want to embed the application in a larger JSL process.

Tip: This can also be accomplished by saving as a .jmpapp file and using `Open()` to run the application. However, saving the script can reduce the number of files.

To Add-In Lets JMP users install the application and launch it from a JMP menu. See "Compile Add-Ins with the JMP Add-In Builder".

To Template Saves the application as a template from which you can create new applications. The templates are saved in the following locations based on your operating system and JMP version:

- Windows (JMP): "/C:/Users/<username>/AppData/Roaming/SAS/JMP/<version number>/ApplicationTemplates"
- Windows (JMP Pro): "/C:/Users/<username>/AppData/Roaming/SAS/JMPPro/<version number>/ApplicationTemplates"
- macOS: "/Users/<username>/Library/Application Support/JMP/<version number>/ApplicationTemplates"

Additional Examples of Creating Applications

The following examples illustrate various uses for applications in JMP.

Parameterizing Variables

The following example shows how to create an application with parameterized variables. Users select the variables in a launch window, and then predefined reports are generated.

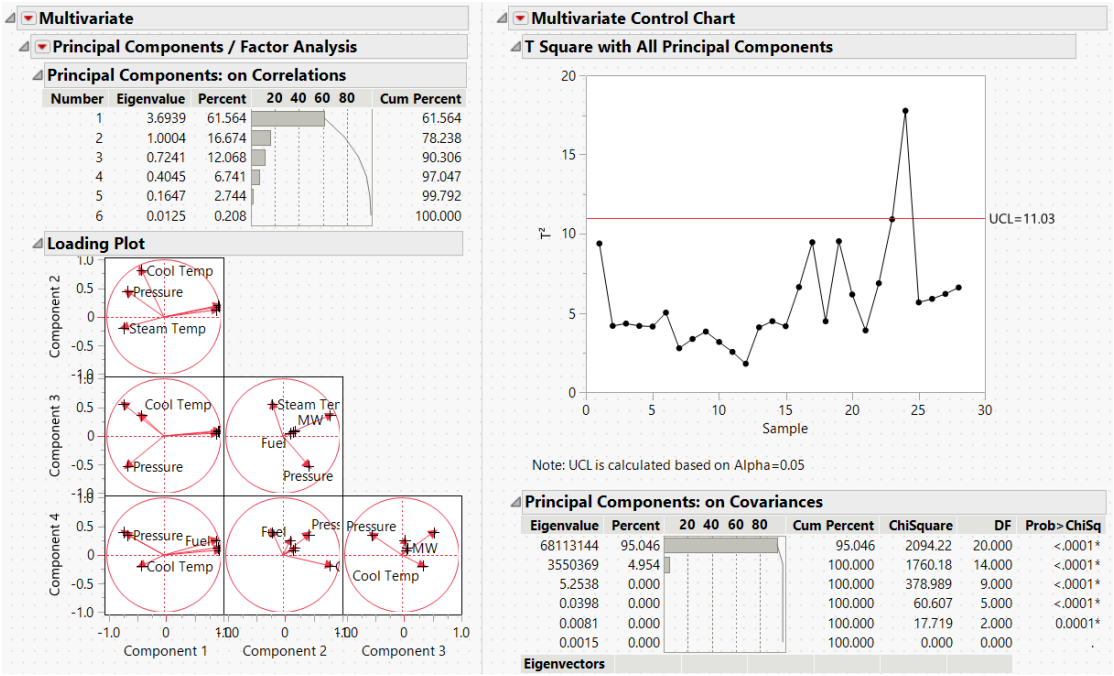
1. Select **Help > Sample Data Library** and open Quality Control/Steam Turbine Historical.jmp.
2. Run the Principal Component Analysis and Loading plot table scripts to generate the reports.
3. Select **File > New > Application**.
4. Select **Blank Application**.

   The Application Builder window appears.
5. Enlarge the window.
6. In the Sources panel, drag each Multivariate report onto the application workspace in a single row.

7. Select **Edit > Select All**.
8. Select **Format > Add Container > H List Box** to align the reports horizontally.
9. Select each report and type `yvar` next to **Y Variable** in the Objects panel.
10. Click the Application red triangle and select **Run Application**.
11. Select the Fuel, Steam Flow, Steam Temp, and MW variables and then click the **Y** button.
12. Click **OK**.

    New Multivariate reports appear in one window.

Figure 16.12 Multivariate Application



Tip: The absolute path to the data table is inserted automatically in the application's Table property. If the data is sample data, the `$SAMPLE_DATA` path variable is automatically used in the path. You can also enter an absolute or relative path. Remember that the user must have access to this path.

Filtering Data in Multiple Reports

In an application that contains several reports, you can select data in one report and then view only that data in all other reports contained in the same window.

To set up the filter, follow these steps:

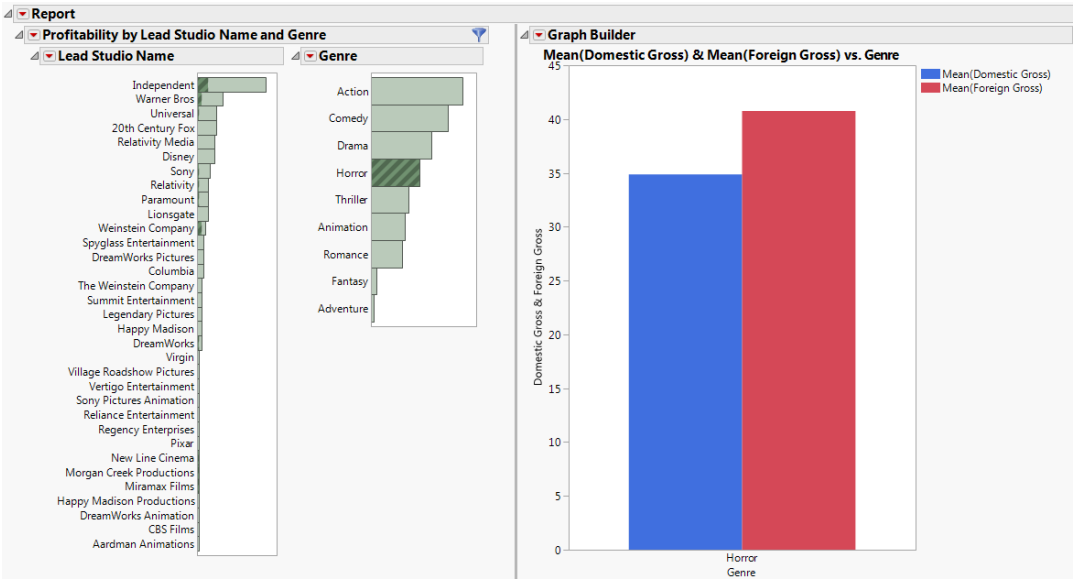
1. Create the application and add two reports to the workspace.
2. Select both reports. From the toolbar, select the HListBox icon.
3. Right-click the primary report and select **Use as Selection Filter**.

   This places the primary report display box in a Data Filter Source Box and the parent report display box in a Data Filter Content Box.
4. Click the Application Builder red triangle and select **Run Application**.

   The reports appear in one window (Figure 16.13).
5. Test the application by selecting a histogram bar in the primary report.

   Only data for the selected bar appears in the second report.

Figure 16.13 Example of Filtered Content



Filtering Data in Separate Reports Using a Local Data Filter

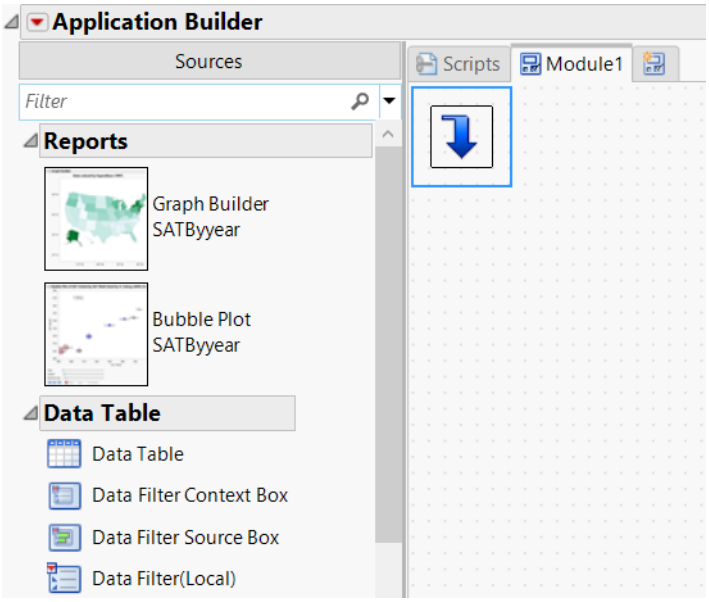
The following example illustrates how to add a Local Data Filter to multiple reports and then filter each report separately.

1. Select **Help** > **Sample Data Library** and open SATByYear.jmp.
2. Run these scripts: Bubble Plot by Region and Graph Builder Map.
3. Resize the report windows to make them smaller, so that a 2x2 arrangement will fit on the screen.
4. Select **File** > **New** > **Application** and choose **Blank Application**.

5. Double-click **Data Filter Context Box**.

   This adds a data filter context box to the workspace. Everything placed in this box will share the same Local Data Filter.

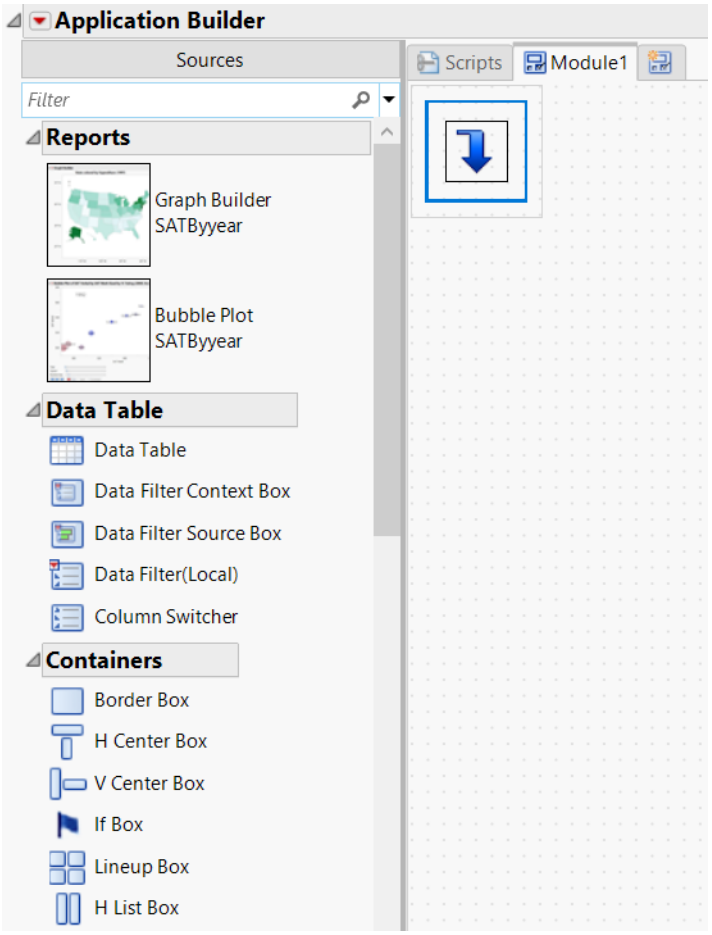
Figure 16.14 Adding a Data Filter Context Box



6. Under Containers, drag a **H List Box** into the Data Filter Context Box.

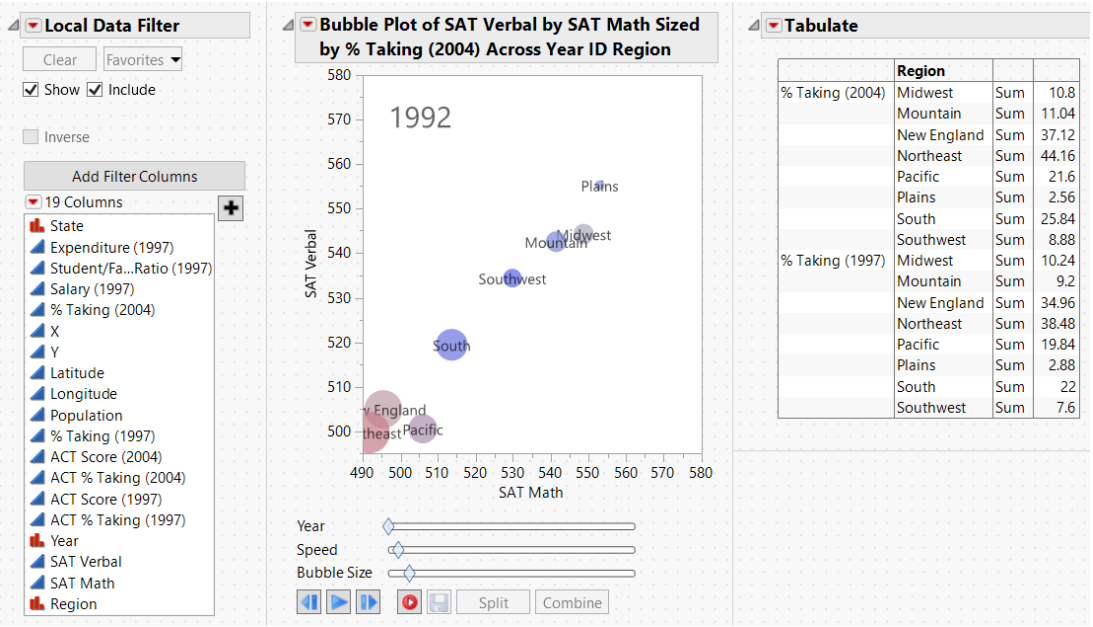
   This allows you to put multiple objects in this box and arrange them horizontally.

Figure 16.15 Adding a H List Box



7. Now drag a Data Filter (local), the Bubble Plot, and the Tabulate report into the H List Box.

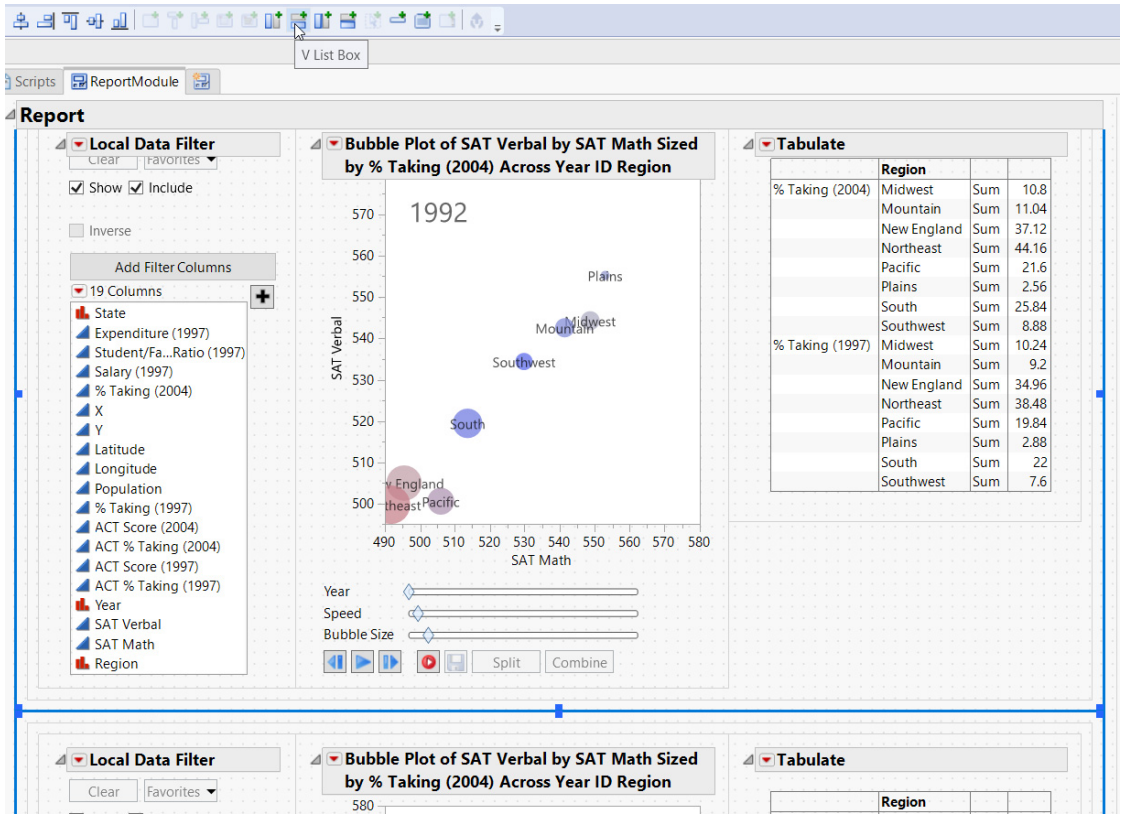
Figure 16.16 Adding Local Data Filter and Two Reports



8. Select the entire Data Filter Context Box. Press Ctrl and drag it below itself, to place a copy below the original set of reports.
9. Select both Data Filter Context boxes, then on the toolbar, click the V List Box icon.

   This arranges both reports into a single list box for display purposes.

Figure 16.17 Selecting Both Data Filter Context Boxes and the V List Box Icon



10. Click the Application Builder red triangle and select **Run Application**.
11. In the Local Data Filter on the top, click SAT Math, then click the Add button . Move the slider to see how it only impacts the reports at top. Do the same for the bottom set of reports.

Selecting Dates

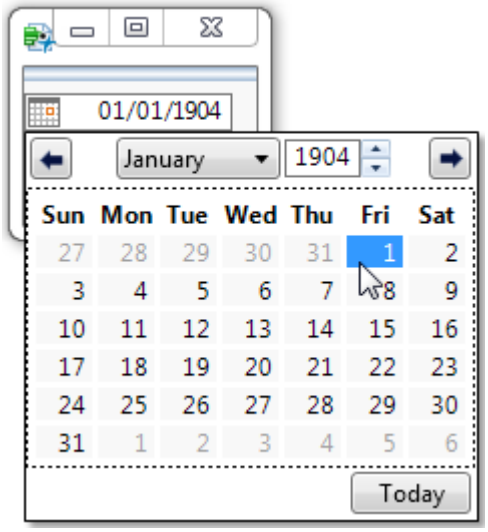
To insert a date selector window into an application, follow these steps:

1. Drag a Number Edit Box from the Sources panel to the workspace.
2. Select the Number Edit Box and then click the button next to Format in the Properties panel.
3. Select **Date** from the list and then select **m/d/y**.
4. Click outside the list.
5. Click the Application Builder red triangle and select **Run Application**.

   The Number Edit Box, which shows a date, appears in a new window.

6. Click the calendar icon to view the date selector window.

Figure 16.18 Example of a Date Selector



The date selector enables you to select the month and year as well as the date and time for the box.

Creating a Launch Window and Report

The Launcher with Report sample application combines a platform launch window with the resulting report. The sample application is displayed when you select **File > New > Application**.

The following example shows you how to arrange objects and include scripts that make up the application.You will recreate the Launcher with Report sample application so that you can create your own applications similarly.

Note: In the Launcher with Report sample application, the variable names of display boxes are customized to describe the display box. For example, “Text1” was renamed “Description” in the sample application. To reduce the number of steps in this example, you do not rename the variables unless one of the scripts refers to the variable name.

1. Select **File > New > Application** and click the **Launcher with Report** sample.

   You will copy and paste JSL from this sample into your copy. Notice that the Iris.jmp sample data table opens when you open the application.
2. Select **File > New > Application** and click **Blank Application**.

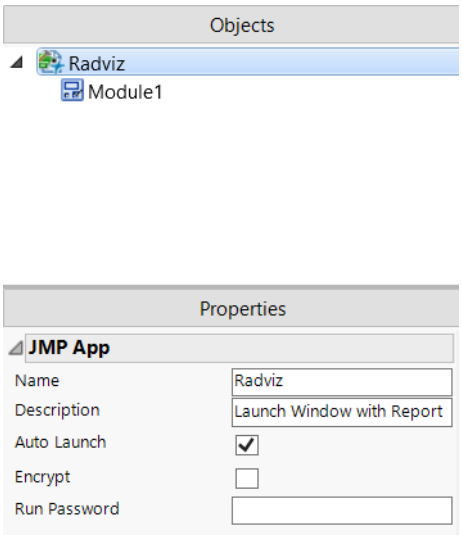
Create the Launch Module

The Launch Module contains objects that create a launch window.

1. In the Objects panel of the blank application, select Application.
2. In the Properties panel:
   - Change the Name from Application to “Radviz” and press Enter.

     This name appears in the title bar when you run the application.
   - Change the Description to “Launch Window with Report”.

Figure 16.19 Customizing the Application

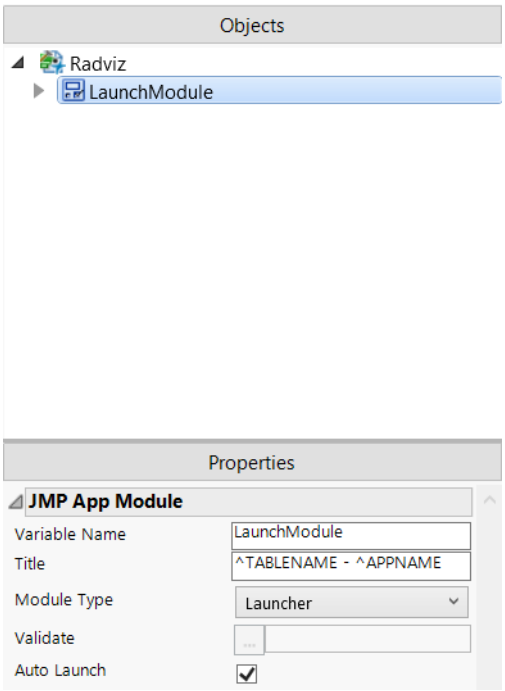


3. Click the **Scripts** tab and select Radviz from the Namespace list.
4. After the green comments, on a new line, type the following expression:

   ```
   dt = Current Data Table();
   ```

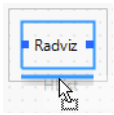
   One of the scripts that you add to this application refers to `dt` to identify the data table.
5. In the Objects panel, select **Module1**.
6. In the Properties panel:
   - Change the Variable Name to “LaunchModule” and press Enter.
   - Change the Module Type to Launcher.
   - Make sure that Auto Launch is selected so that the launch window opens when you run the application.

Figure 16.20 Customizing the LaunchModule



7. Click the **LaunchModule** tab and drag a V List Box (under Containers) onto the workspace.
8. Drag a Text Box (under Display) over the arrow of the V List Box.
9. Double-click Text1, change it to “Radviz”, and then press Enter.
10. Drag an H List Box (under Containers) below the border of the Text Box until a blue line appears.

Figure 16.21 Adding the H List Box



11. Because you have made many changes, select **File > Save** and rename the file. Make sure that .jmpappsource is selected as the file type.

Create the Select Columns Box

1. Drag a Panel Box (under Containers) over the H List Box arrow.
2. In the Properties panel, change the Title to “Select Columns” and press Enter.
3. Drag a Col List Box(All) (under Input) over the Panel Box arrow.

4. In the Properties panel, change the Variable Name to “ColumnList” and press Enter.

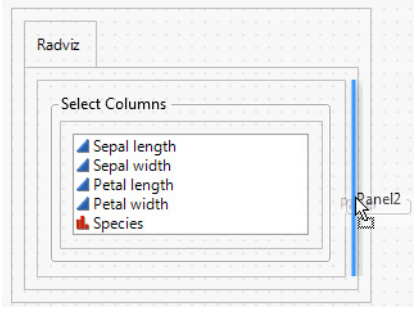
   The script you add later refers to this variable name.
5. Select **DataTable1** in the Objects panel.

   The current data table, Iris.jmp, appears in the Path box.

Create the Cast Selected Columns into Roles Box

1. Drag a Panel Box (under Containers) to the right of the existing Panel Box until a blue line appears.

Figure 16.22 Adding the Panel Box



Note: Make sure that you drag the Panel Box to the H List Box shown in Figure 16.22.

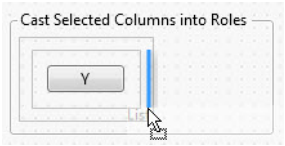
2. In the Properties panel, change the Title to Cast Selected Columns into Roles and press Enter.
3. Drag an H List Box (under Containers) over the Panel Box arrow.

Create the Contents of the Cast Selected Columns into Roles Box

1. Drag a Button Box (under Buttons) over the H List Box arrow.
2. In the Properties panel:
   - Change the Title to “Y” and press Enter.
   - In the Press box, type “SetY” and press Enter.

   You copy and paste the script for this button later.
3. Drag a Col List Box (under Input) inside the right border of the Y Button Box.

Figure 16.23 Adding the Col List Box



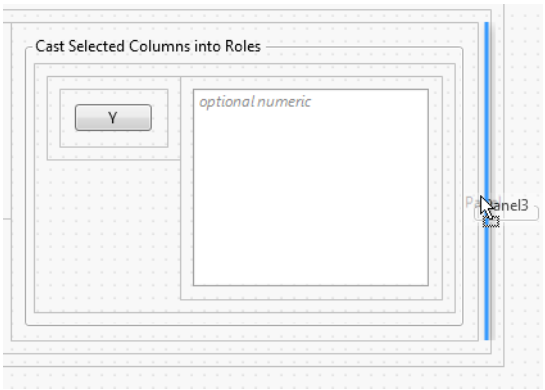
4. In the Properties panel:
   – Change the Variable Name to "YList" and press Enter.
     The script that you add later refers to this variable name.
   – Change the Data type to Numeric.
     The words "optional numeric" appear in the Col List Box.

Tip: Change Min Items to display the minimum number of required items in the Col List Box. If you type 3, then "required numeric" appears three times in the box as a hint to the user.

Create the Action Panel

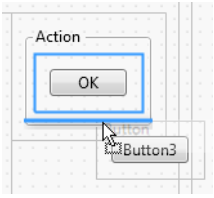
1. Drag a Panel Box (under Containers) to the right of the Panel Box that you just added.

Figure 16.24 Adding a Panel Box



2. In the Properties panel, change the Title to "Action" and press Enter.
3. Drag a Button Box (under Buttons) over the Panel Box arrow.
4. In the Properties panel:
   – Change the Title to OK and press Enter.
   – In the Press box, type "OnOK" and press Enter.
5. Drag another Button Box (under Buttons) below the OK button.

Figure 16.25 Adding a Cancel Button



6. In the Properties panel:
   - Change the Title to “Cancel” and press Enter.
   - In the Press box, type “OnCancel”.
7. Drag a Spacer Box (under Spacers) below the Cancel button.
8. Drag a Button Box (under Buttons) below the Spacer Box.
9. In the Properties panel:
   - Change the Title to “Remove” and press Enter.
   - In the Press box, type “OnRemove” and press Enter.
10. Select the outer border of the objects on the workspace.
11. Right-click and select **Move to Corner**.

    The extra space around the selected object is removed.
12. Select **File > Save** to save your changes in the.jmpappsource file.

Create the Launch Module Script

1. Go back to the Launcher with Report sample application window.
2. Click the Scripts tab.
3. Select LaunchModule as the Namespace and copy the script.
4. Go back to your application window and click the Scripts tab.
5. Delete the script in the LaunchModule namespace and paste the script from the sample application.
6. Select **File > Save** to save your changes in the.jmpappsource file.

Create the Report Module

The Report Module contains an object that builds a graph.

1. In your application window, click the button to the right of the LaunchModule tab to add a new module.
2. In the Properties panel, change the Variable Name to “ReportModule” and press Enter.
3. Make sure that the Module Type is Report.

4. Deselect Auto Launch to prevent the report from opening when the application is run.
5. Drag an Outline Box (under Containers) to the workspace.
6. In the Properties panel, change the Title to “Radviz” and press Enter.
7. Drag a Graph Box (under Display) over the Outline Box arrow.
8. In the Properties panel, change the Variable Name to “Graph” and press Enter.
9. Select the outer border of the objects on the workspace.
10. Right-click and select Move to Corner.

    The extra space around the selected object is removed.
11. Select **File > Save** to save your changes in the.jmpappsource file.

Create the Report Module Script

1. Go back to the Launcher with Report sample application window.
2. Make sure that the Scripts tab is displayed.
3. Select ReportModule as the Namespace and copy the script.
4. Go back to your application window and click the Scripts tab.
5. Delete the script in the ReportModule namespace and paste the script from the sample application.
6. Select **File > Save** to save your changes in the.jmpappsource file.

Your application should look like the Launcher with Report sample application.

Create the Radviz Script

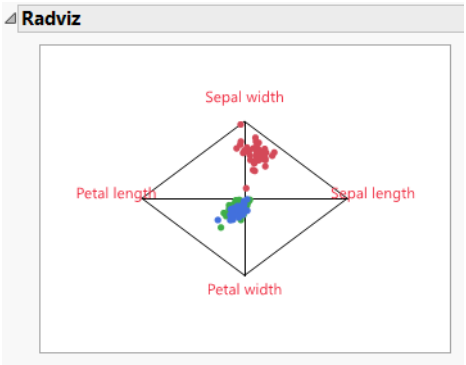
1. Go back to the Launcher with Report sample application window.
2. Make sure that the Scripts tab is displayed.
3. Select Radviz as the Namespace and copy the script.
4. Go back to your application window and click the Scripts tab.
5. Delete the script in the Radviz namespace and paste the script from the sample application.
6. Select **File > Save** to save your changes in the.jmpappsource file.

Test and Save the Final Application

1. In your application window, click the Application Builder red triangle and select **Run Application**.
2. Select the Sepal length and width and Petal length and width columns, click **Y**, and then click **OK**.

A graph of the data should appear.

Figure 16.26 Iris.jmp Graph



If the graph does not appear or errors are generated, make sure that you typed in the correct variable names when necessary. Also make sure that you added the scripts to the Scripts tab for each module.

3. On Windows, select **File > Save As**, select **JMP Application File** from the Save as type list, and then click **Save**.

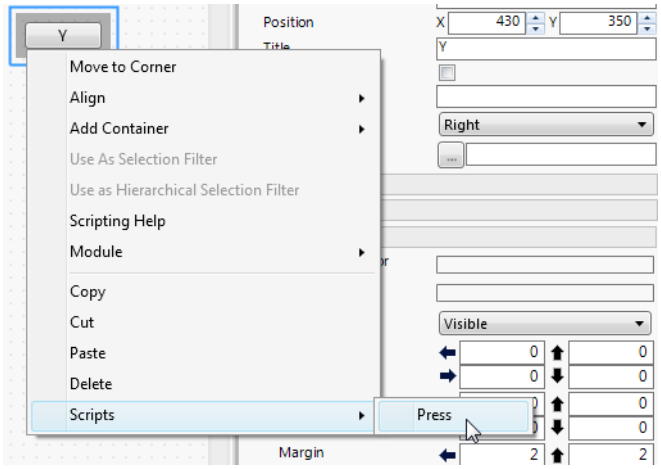
   On macOS, select **File > Export**, select JMP Application, and click **Next** to save the application.

   Your newly created application should match the Launcher with Report sample application.
4. Distribute the .jmpapp file to users. This file cannot be modified.
5. Save the .jmpappsource file so you can edit the application and resave it as a JMP application.

About the Application Scripts

When you create an application, you write scripts to provide functionality to display boxes. You can right-click a display box and select a script from the Scripts menu.

Figure 16.27 Adding a Script Interactively



Selecting a script from the Scripts menu adds a placeholder script to the Scripts tab for the selected module. For example, the Press script shown in Figure 16.27 adds the following script to the Scripts tab:

```
Button5Press=Function({this},
   // This function is called when the button is pressed
   name = this << Get Button Name;
);
```

Fill in the placeholder script with your own JSL. In the application that you previously created, you named two variables `SetY` and `ColumnList`. Rewrite the preceding placeholder script:

```
SetY = Function({},
   // This function will be called when the Y button is
   // pressed

   items = ColumnList << Get Selected;
   YList << Append(items);
);
```

`Get Selected` is passed to the `ColumnList` because you gave the Col List Box this variable name (`ColumnList`) in the object properties. `Get Selected` returns a list of the selected columns. `Append(items)` appends the selected columns to YList, the variable assigned to the Col List Box in the Cast Selected Columns into Roles window.

As shown when you previously created the application, you can write the script from scratch and indicate the script name in the object properties. To get help with syntax for a selected object, right-click the object and select Scripting Index.

Compile Add-Ins with the JMP Add-In Builder

A JMP add-in is a JSL script that you can run anytime from the JMP Add-Ins menu. You can create submenus to group your JMP add-ins and have many levels of menus, if desired.

Experienced JSL script writers can create scripts that extend JMP in many ways (for example, add a custom analytical tool or a user interface to communicate with a database). The JMP add-in architecture simplifies deploying and using these complicated scripts.

You can send co-workers a suite of scripts and tell them how to run them. Another option is sending a single add-in file that any JMP user can install and then use just like any other part of JMP.

As a JMP user, you might be given add-ins to use by co-workers. You can also find add-ins on the JMP website at https://www.jmp.com/addins.

Compile Scripts into an Add-In Using Add-In Builder

To create a JMP add-in, you write the scripts and gather files that are used in the add-in. Then you compile the scripts into an add-in.

- On Windows, select **File > New > Add-In**.
- On macOS, select **File > New > New Add-In**.
- In Dashboard Builder, click the red triangle and select **Save Script > To Add-In**.

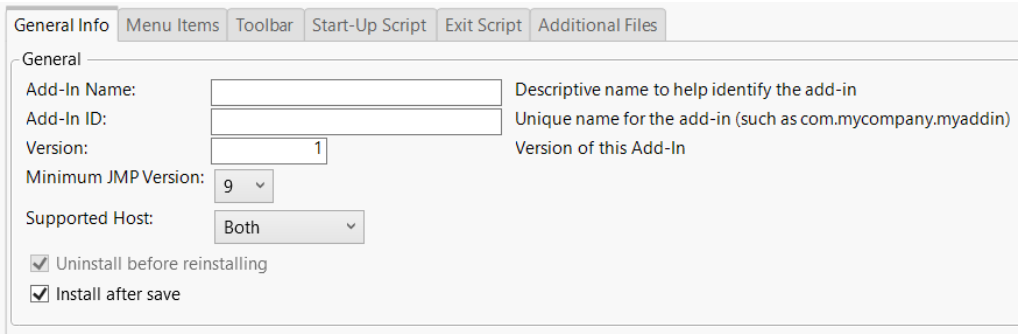
The process of compiling scripts into an add-in involves the following steps:

- "Add General Information"
- "Create Menu Items"
- "Create Toolbars (Windows Only)"
- "Specify Start-up and Exit Scripts (Optional)"
- "Add Additional Files"
- "Save the Add-In"
- "Test the Add-In"

Add General Information

First, in the **General Info** tab, add the general information that identifies and sets up your add-in.

Figure 16.28 Add-In Builder General Info Tab



1. Enter a name for the add-in.

   This is the name of the registered add-in, which appears in the **View > Add-Ins** window.

2. Enter a unique identifier string.

   Unique ID strings are case-insensitive. To ensure uniqueness, it is strongly recommended to use reverse-DNS names (for example, `com.mycompany.myaddin`). The ID string must meet the following requirements:

   – It can be up to 64 characters in length.
   – It must begin with a letter.
   – It should consist only of letters, numbers, periods, and underscores.
   – It should contain no spaces.

   In JSL, use this string to refer to the add-in (for example, in calls to `Get Addin()` or as an argument to $ADDIN_HOME).

3. Enter the version of the add-in.

   If you decide to make changes to the add-in at a later date, you can update the version number and then verify that users have the correct version.

4. Select the minimum version of JMP that the add-in works on.

   Note: Add-ins were introduced in JMP 9, so no prior versions are supported. And when saving an application as an add-in, select 10 or 11 as the JMP minimum version.

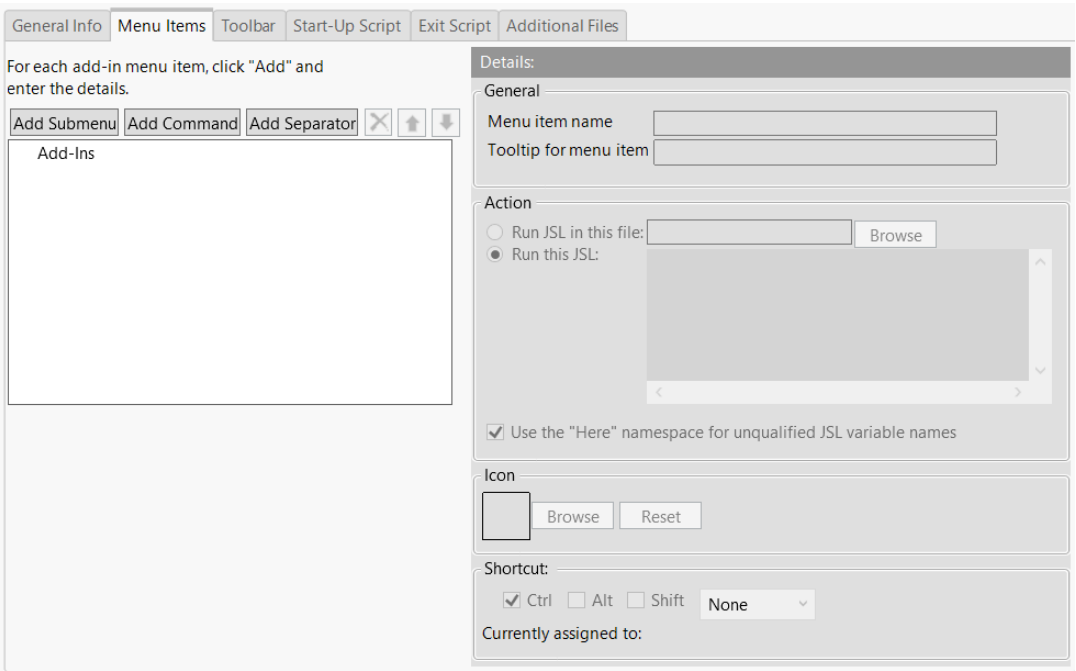
5. Select whether you want the add-in to be supported on Windows, macOS, or both.
6. (Optional) Select the check box next to **Install after save** if you want to install the add-in after saving it.

   If you do not select this option, the add-in is not installed once you save it, and it does not appear as a selectable menu item in the **Add-Ins** menu.

Create Menu Items

1. Click the **Menu Items** tab.

Figure 16.29 Add-In Builder Menu Items Tab



2. (Optional) Click **Add Submenu**.

   If you have multiple add-ins, you can group them under a submenu.
3. If you add a submenu, next to **Menu item name**, enter the name of the submenu.

   This name appears in the **Add-Ins** menu.
4. Click **Add Command**.
5. Next to **Menu item name**, enter the name of the add-in command.
6. (Optional) Next to **Tooltip for menu item**, enter the content that appears as a tooltip when the users place their cursor over the menu item.
7. Add the script. Select either **Run this JSL** and copy and paste in your script, or select **Run JSL in this file** and click **Browse** to find the file containing your script.
8. (Optional) Select **Use the "Here" namespace for unqualified JSL variable names** to ensure that all unqualified JSL variables are in the `Here` namespace, and local only to the script.

Notes:

- If your script creates a custom menu or toolbar, the variables are in the `Here` namespace by default.
- For more information about the `Here` namespace, see "Advanced Scoping and Namespaces".

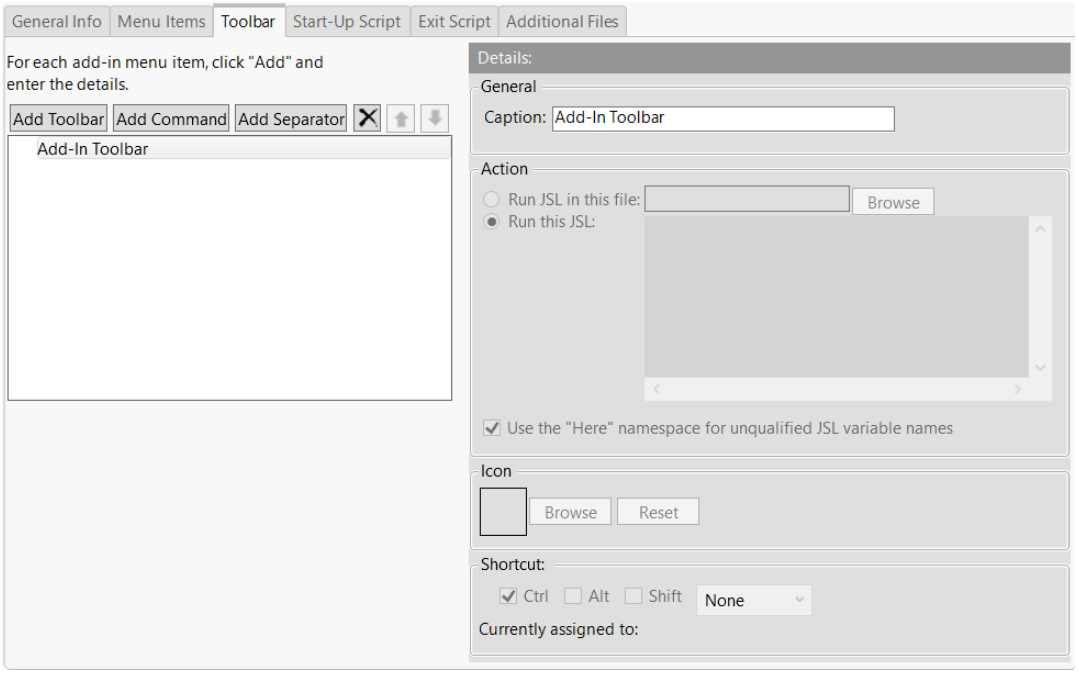
9. (Optional) Browse to add an icon that appears next to the menu item in the **Add-Ins** menu.
10. (Optional, Windows only) Create a keyboard shortcut for the add-in.
11. To add multiple menu items, repeat the steps.

    You can add multiple levels of submenus and add-in commands.
12. Click **Save** and save the add-in to the desired directory.
13. Click **Close**.

Create Toolbars (Windows Only)

1. Click the **Toolbar** tab.

Figure 16.30 Add-In Builder Toolbar Tab



2. Click **Add Command**.
3. Next to Caption, type the name of the toolbar item. This name is only for internal use. The user does not see it.

4. Add the script. Select either **Run this JSL** and copy and paste in your script, or select **Run JSL in this file** and click **Browse** to find the file containing your script.
5. (Optional) Select **Use the "Here" namespace for unqualified JSL variable names** to ensure that all unqualified JSL variables are in the `Here` namespace, and local only to the script.
6. Click **Save** and save the add-in to the desired directory.
7. Click **Close**.

Note: For more information about customizing menus in JMP, see *Using JMP*.

Specify Start-up and Exit Scripts (Optional)

Click the **Start-Up Script** tab to add a script that runs when JMP starts up (and the add-in starts). You can select an existing script (**Run JSL in this file**) or copy and paste in a script (**Run this JSL**). For example, you could provide a message telling the user that the add-in is installed upon start-up.

Click the **Exit Script** tab to add a script that runs when JMP exits or when you disable the add-in. You can select an existing script (**Run JSL in this file**) or copy and paste in a script (**Run this JSL**). For example, you could provide a prompt for the user to export a JMP data table upon exiting or disabling the add-in.

Add Additional Files

If your script calls other scripts, or contains graphics or data tables, add those files here.

Save the Add-In

Save the add-in by clicking the **Save** button on any tab. This effectively creates the add-in.

- If you selected the **Install after save** option in the General Info tab, then the add-in menu item appears in the **Add-Ins** menu immediately.
- If you did not select the **Install after save** option, when you open the saved add-in file, you are prompted to install the add-in.

Test the Add-In

Once your add-in is installed, test your add-in:

1. Select **View > Add-Ins**.
2. Select your add-in and click **Unregister**.
3. Reinstall your add-in by either selecting **File > Open** in JMP, or by double-clicking your .jmpaddin file.
4. Ensure that the menu and toolbar button run your script correctly and that the script itself runs correctly.

Edit an Add-In

To edit a saved add-in:

1. Select **File > Open**.
2. Navigate to and select the add-in file.
3. Select one of the following options:
   - On Windows click the arrow to the right of the **Open** button and select **Open using Add-In Builder**.
   - On macOS, select the **Edit after opening** option.
4. Click **Open**.

   The file opens in the Add-In Builder. Update the arguments and then save the changes.

Share an Add-In

Once you have a .jmpaddin file, you can share the file with other users. Email the file or you place it in a shared location, such as a network folder, or on the JMP File Exchange (located online in the JMP User Community).

When JMP users open the file, the add-in files are extracted into the appropriate folder, and the add-in is registered and installed. The add-in now appears in the user's JMP Add-Ins menu.

Installing Multiple Add-Ins

If you want to install multiple add-ins, copy the add-ins into the following location:

- Add-In files on Windows are located here:
  - %ALLUSERSPROFILE%/SAS/JMP/AddIns (any user on this machine can access the add-in)
  - %APPDATA%/SAS/JMP/AddIns (only the current user on this machine can access the add-in)
- Add-In files on macOS are located here:
  - /Library/Application Support/JMP/AddIns (any user on this machine can access the add-in)
  - ~/Library/Application Support/JMP/AddIns (only the current user on this machine can access the add-in)

When JMP starts, the addinRegistry.xml file is read, which contains information about previously registered JMP add-ins. Then JMP looks in the add-in folders for any other add-ins and installs them automatically.

Notes:

- The Home Folder for discovered add-ins does not have to be the AddIns subfolder in which the addin.def file was found. The addin.def file can be the only file in that subfolder and have a Home setting that points to some other location where the add-in files actually reside.
- If an automatically discovered add-in has the same unique ID as an add-in that was explicitly registered, the automatically discovered add-in is used.

Register an Add-In Using JSL

If your add-in files are not contained within a .jmpaddin file, you can use the `Register Addin()` JSL function to manually register the addin.def file. This installs and registers the add-in.

- For information about the JSL functions, see the *JSL Syntax Reference*.
- For information about creating the addin.def file, see "Create an Add-In Manually".

Notes:

- JMP might find a file named addin.def in the specified home folder. If so, values from that file are used for any optional arguments that are not included in the `Register Addin()` function.
- The addin.def file is used only for values that are not provided in the `Register Addin()` function. This function is useful while developing, but not necessary, since the addin.def file is enough to register an add-in.

Create an Add-In Manually

The addin.def file is a simple text file containing name-value pairs that provide registration information about a JMP add-in. Here are the name-value pairs to include in the addin.def file:

`name` Optional. The name that can be displayed in the JMP user interface wherever add-in names are displayed, instead of the unique ID. This name is displayed if no localized names are provided or when JMP is run under a language for which you did not provide a localized name. Named *Add-In Name* in Add-In Builder.

`name_xx` Optional. Allows the user-friendly name to be localized for different languages, where *xx* is the two-letter ISO 639-1 code for the language. If you include localized names,

you should still include a language-neutral name in case JMP is running under regional settings for which you do not have a localized name. Not available in Add-In Builder.

`id` Required. The unique ID for your add-in. The string can contain up to 64 characters. The string must begin with a letter and contain only letters, numbers, periods, and underscores. Reverse-DNS names are recommended to increase the likelihood of uniqueness. Named *Add-In ID* in Add-In Builder.

`home` Optional. The path to the add-in files. The Home Folder for the add-in is assumed to be the folder where `addin.def` is located. You need to include a setting for home only if the Home Folder is somewhere else (for example, a network shared folder). Not available in Add-In Builder.

`home_win` Optional. The path to the add-in files to be used when JMP is running on Windows. Overrides the value specified for home on Windows, if any. Not available in Add-In Builder.

`home_mac` Optional. The path to the add-in files to be used when JMP is running on the macOS. Overrides the value specified for home on macOS, if any. Not available in Add-In Builder.

`autoLoad` Optional, Boolean. The default value is True (1). Determines whether this add-in is initially configured to load automatically during JMP start-up. Named *Install after save* in Add-In Builder.

`host` Optional. Valid values are `Win` and `Mac`. Named *Supported Host* in Add-In Builder.

`minJMPVersion` Optional. Valid values are integers corresponding to the JMP major version that is the minimum version that the add-in supports. Named *Minimum JMP Version* in Add-In Builder.

`maxJMPVersion` Optional. Valid values are integers corresponding to the JMP major version that is the maximum version that the add-in supports. Use this setting *only* if there is a known incompatibility between your add-in and a specific version of JMP. You should provide a new version of the add-in for later versions of JMP. Not available in Add-In Builder.

Example of a addin.def File

```
id="com.mycompany.myaddin"
name="My Add-In's Friendly Name"
name_fr="My Add-In's French Name"
name_de="My Add-In's German Name"
home="\\server\share\myjmpaddin"
Autoload=1
MinJMPVersion=9
```

Example of a JMP Add-In

A sample add-in named Simple Calculator.jmpaddin is located in one of the following folders:

- On Windows: C:/Program Files/SAS/JMP/16/Samples/Scripts
- On macOS: /Library/Application Support/JMP/16/Sample/Scripts

Note: On Windows, in JMP Pro, the "JMP" folder is named "JMPPro".

To see what the add-in contains, change the extension to .zip and unzip it into a new folder. To see how it works, change the extension back to .jmpaddin and install.

The add-in contains the following files:

addin.def

Provides the specification for JMP to register the add-in. It contains only these two lines:

```
id="com.jmp.sample.calculator"
name="Simple Calculator"
```

addin.jmpcust

Provides the menu customization file that is created when you interactively create a custom menu. Your customizations are shown only to the current user; other users who log on to the computer and open JMP do not see your personal toolbars and menus. This example places the add-in menu item into the default **Add-Ins** menu.

calculator.jsl

A JSL script that creates a basic calculator.

calc_icon.gif

The image used as the calculator's icon.

To download more add-ins, visit
https://community.jmp.com/t5/File-Exchange/ct-p/FileExchange.

Manage JMP Add-Ins

Install Add-Ins

A JMP add-in is a file that has the .jmpaddin extension. You can install it in one of two ways:

1. Select **File > Open**.
2. Navigate to the .jmpaddin file and select it.

3. Click **Open**.

You can also double-click the .jmpaddin file or drag it onto the JMP Home Window.

View Your Add-Ins

Select **View > Add-Ins** to see the add-ins that you have already installed.

Reorder Your Add-Ins

Select **View** > **Add-Ins**, then use the Load Order arrows to move each add-in where you want it to appear in the Add-Ins menu.

Update Add-Ins

If you are given an update to an add-in you already have installed, you can just install the update as you did the original add-in. Doing so overwrites the old version with the new one.

Disable, Enable, and Remove Add-Ins

To temporarily disable an add-in without removing it

1. Select **View > Add-Ins**.
2. Select the add-in that you want to disable.
3. Deselect the **Enabled** check box.

To enable a disabled add-in

1. Select **View > Add-Ins**.
2. Select the add-in that you want to enable.
3. Select the **Enabled** check box.

To unregister an add-in

1. Select **View > Add-Ins**.
2. Select the add-in that you want to remove.
3. Click **Unregister**.

Chapter 17

Common Tasks

Getting Started with Sample Scripts

Examining working scripts line-by-line is one of the best ways to learn JSL. This chapter describes common tasks in JMP, such as converting date/time values and extracting specific values from reports. Sample scripts that address these issues are installed with JMP in the Samples/Scripts folder so you can run them yourself.

Contents

Run a Script at Start Up 927
Convert Character Dates to Numeric Dates 928
Format Date-Time Values and Subset Data 929
Create a Formula Column 930
Extract Values from an Analysis into a Report 932
Create an Interactive Program 936

Run a Script at Start Up

You can run the same script every time you start JMP. For example, you might include the definitions of some utility functions or a namespace that you want available throughout your JMP session. You might also set some preferences in a startup script to keep preferences consistent. Preferences persist across JMP sessions, but you could explicitly reset some preferences in case they were changed at some point in the previous session.

Name the script jmpStart.jsl and place it in one of the following folders, as appropriate for your operating system. When JMP starts, JMP looks for the jmpStart.jsl script in these folders in the order in which they are listed here. The first one that is found is run, and the search immediately stops.

Note: Some path names in this section refer to the "JMP" folder. On Windows, in JMP Pro, the "JMP" folder is named "JMPPro".

On Windows:

1. C:/Users/<username>/AppData/Roaming/SAS/JMP/16
2. C:/Users/<username>/AppData/Roaming/SAS/JMP

On macOS:

1. /Users/<username>/Library/Application Support/JMP/16
2. /Users/<username>/Library/Application Support/JMP

The jmpStart.jsl script runs only for a particular user on a computer. You can add a script named jmpStartAdmin.jsl in one of the following places, as appropriate for your operating system. This script is run for every user on a computer. JMP runs jmpStartAdmin.jsl first if found. Then JMP runs jmpStart.jsl if found.

On Windows:

1. C:/ProgramData/SAS/JMP/16
2. C:/ProgramData/SAS/JMP

On macOS:

1. /Library/Application Support/JMP/16
2. /Library/Application Support/JMP

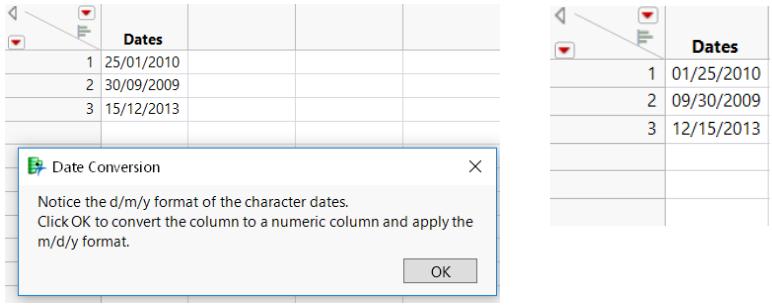
Convert Character Dates to Numeric Dates

Data might appear to be numeric in the data table. However, the column properties may specify a character data type. To manipulate the data as date/time values, convert the column to a numeric column and specify how you want the values to appear.

Convert Dates.jsl creates a data table, specifies the data input format, changes the column to a numeric continuous column, and applies the m/d/y format (Figure 17.1).

```
// Create a data table with character dates.
dt = New Table( "Example",
   Add Rows( 3 ),
   New Column( "Dates",
      Character,
      Nominal,
      Set Values( {"25/01/2010", "30/09/2009", "15/12/2013"} )
   )
);

// Display a modal dialog for the user to confirm the format conversion.
nw = New Window( "Date Conversion",
   <<Modal,
   tb = Text Box(
      "Notice the d/m/y format of the character dates.
Click OK to convert the column to a numeric column and apply the m/d/y
   format."
   )
);

/* Apply the Numeric data type.
Specify the Informat (input format) value "d/m/y".
Specify the Format (display format) value "m/d/y".
Apply the Continuous modeling type */
col = Column( dt, "Dates" );
col << Data Type( "Numeric", Informat( "d/m/y" ), Format( "m/d/y" ) );
col << Modeling Type( "Continuous" );
```

Figure 17.1 Converting Character Dates (Before and After)



When you change the column's data type from character to numeric, defining the format in which the data were entered is important. In this example, `Informat( "d/m/y" )` defines the input format. `Format( "m/d/y" )` defines the new display format. If `Informat()` is omitted, the `Format()` value is applied as both the input and display format. This results in missing values for some data.

Modify Convert Dates.jsl to see for yourself.

1. Open Convert Dates.jsl from the sample scripts folder.
2. Right-click the script window and select **Show Line Numbers**.
3. On line 9, change "25/01/2010" to "01/25/2010".
4. On line 27, delete `Informat( "d/m/y" ),` (including the comma).
5. Run the script.

   `Format( "m/d/y" )` is applied to the column. Only "01/25/2010" appears in the column. The other values are missing; "30/09/2009" and "15/12/2013" are not valid m/d/y values.

Format Date-Time Values and Subset Data

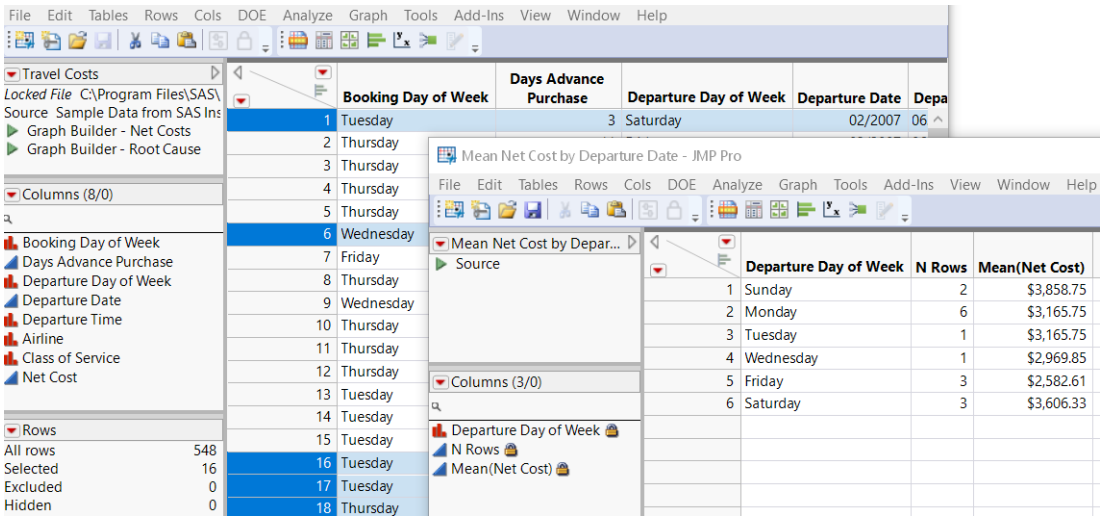
How can you work with dates in JMP? JMP provides a number of formats that you can use to make comparisons and then subset data based on the date.

Select Where Using Dates.jsl applies the Date MDY format to a column of departure dates and subsets the data. A summary table of mean net costs by departure date then appears (Figure 17.2).

```
/*  How can you work with dates in JSL? JMP provides a number of formats
that you can use to make comparisons and then subset data based on the date.
*/

hdt = Open( "$SAMPLE_DATA/Travel Costs.jmp" );
```

```
/* Apply the Date MDY format to Departure Date values and then select only
February dates. */
hdt << Select Where(
      (Date MDY( 02, 01, 2007 ) <= :Departure Date < Date MDY( 03, 1, 2007 ))
      );

/* Subset the selected rows into two tables: one table contains February
departure dates, the other contains all data for those departure dates. */
nt1 = hdt << Subset( Columns( :Departure Date ),
   Output Table Name( "February Departure Date" ) );
nt2 = hdt << Subset( Output Table Name( "February Data" ) );

/* Create a summary table, grouping mean cost by day of week that departure
took place. */
sumDt = nt2 << Summary(
   Group( :Departure Day of Week ),
   Mean( :Net Cost ),
   Output Table Name( "Mean Net Cost by Departure Date" )
);
```

Figure 17.2 The Original Table and the Final Summary Table

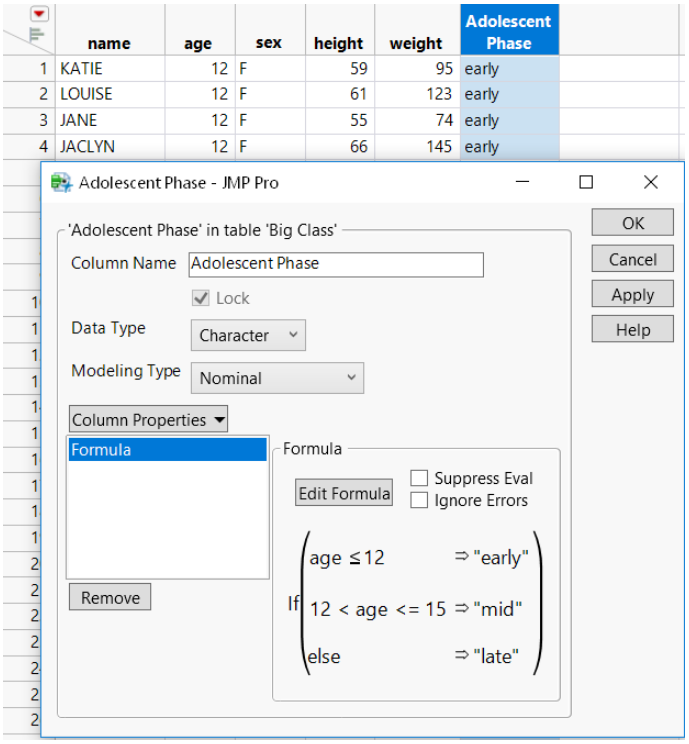


Create a Formula Column

How do you create a formula column that combines conditional expressions with value comparisons? Create a Formula Column.jsl shows how to create a new formula column that evaluates ages in Big Class.jmp and returns the result in the new column (Figure 17.3).

```
/* Scenario:

How do you create a formula column that combines conditional expressions
with value comparisons? This script shows how to create a new formula
column that evaluates ages in Big Class.jmp and returns the conditional
result in the new column. */

dt = Open( "$SAMPLE_DATA/Big Class.jmp" );

/* Create a new character column for the formula.
Insert "early" in the new column if the age is less than or
equal to 12. Insert "mid" if the age is less than or equal to
15 but greater than 12. For ages greater than 15, insert "late".
*/

dt << New Column( "Adolescent Phase",
   Character,
   Formula(
      If( :age <= 12, "early",
         12 < :age <= 15, "mid",
         "late"
         )
      )
);
```

Figure 17.3 Conditional Expression in Formula



Extract Values from an Analysis into a Report

How do you capture specific results of an analysis into a custom report using JSL?

The JMP platforms in the Analyze and Graph menus contain two objects known as the analysis and report layers. Messages are sent to the analysis layer that generate the desired results.

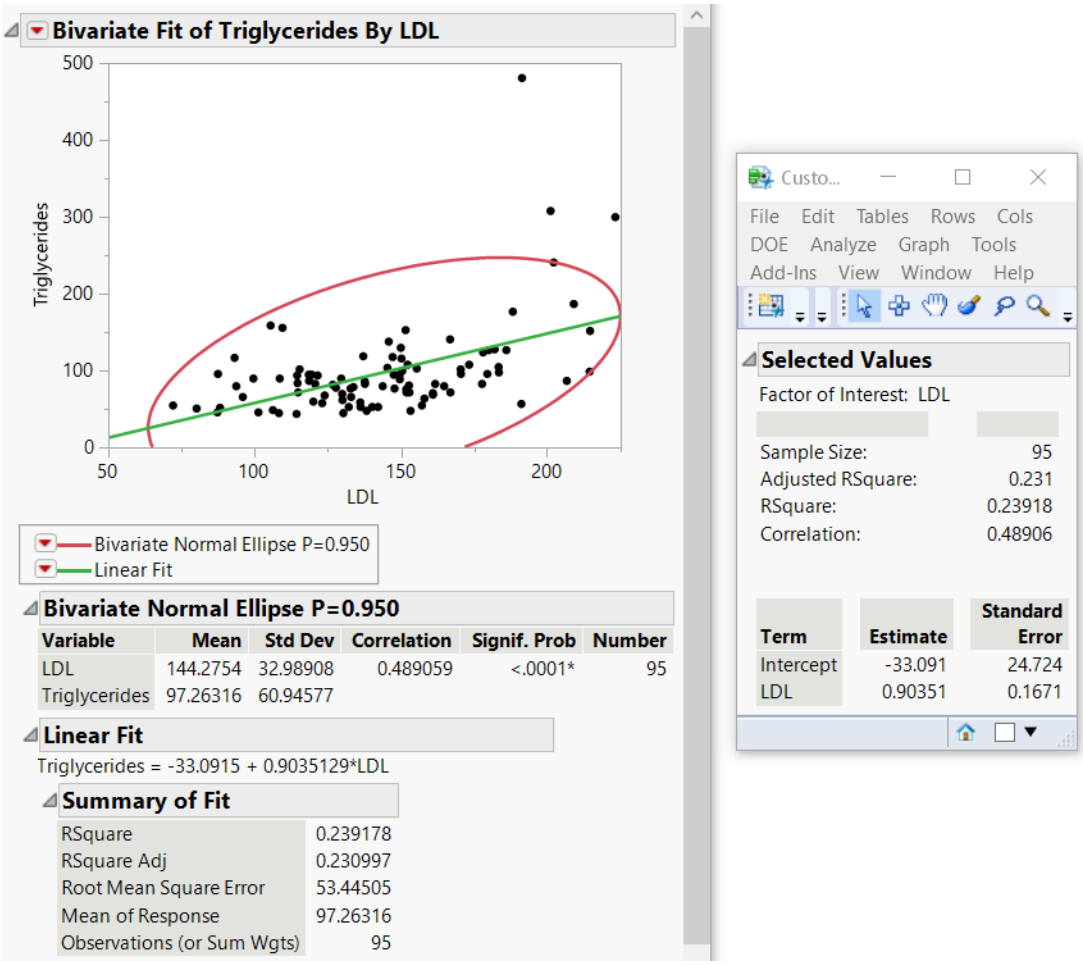
Extract Values from Reports.jsl performs a Bivariate analysis and shows results such as the sample size, RSquare, and Correlation in a new report window. Figure 17.4 shows the Bivariate report along with the customized report, though the Bivariate report window closes after the script is finished.

```
/* Scenario: How do you capture specific results of an analysis
in a report using JSL?

The JMP platforms in the Analyze and Graph menus contain two objects
known as the analysis and report layers.

Messages are sent to the analysis layer that generate the desired results.

This script performs a Bivariate analysis and shows results
such as the sample size, RSquare, and Correlation in a new report window.
*/

sd = Open( "$SAMPLE_DATA/Lipid Data.jmp" );

biv = Bivariate(        // biv is the analysis layer.
  Y( :Triglycerides ),
  X( :LDL ),
  Density Ellipse( 0.95, {Line Color( {213, 72, 87} )} ),
  Fit Line( {Line Color( {57, 177, 67} )} ),
);

/* Make sure the second Outline Box (called "Correlation")
in the Bivariate report is open. You can then see which content
is extracted into the Custom report. */
report(biv) [Outline Box( 2 )] << Close( 0 );

reportbiv = biv << Report;  // reportbiv is the report layer.

/* The density ellipse is generated first.
Extract the correlation coefficient. */
corrvalue = reportbiv[Outline Box( 2 )][Number Col Box( 3 )] << Get( 1 );

/* ...followed by Fit Line
Extract the numeric values from the Summary of Fit report
and place them in a matrix. */
sumfit = reportbiv[Outline Box( 4 )][Number Col Box( 1 )] << Get as Matrix;

// Extract the values of RSquare and AdjRSquare as one by one matrices.
rsquare = sumfit[1];
adjrsq = sumfit[2];
avg = sumfit[4];
samplesize = sumfit[5];

/* Extract the first column of the Parameter.
Estimates report as two objects. */
term = reportbiv[Outline Box( 7 )][String Col Box( 1 )] << Get();
```

```
// Clone the report layer as a String Col Box.
cloneterm = reportbiv[Outline Box( 7 )][String Col Box( 1 )] << Clone Box;

// Extract the Parameter Estimates values as a matrix.
est = reportbiv[Outline Box( 7 )][Number Col Box( 1 )] << Get as Matrix;

// Extract the Standard Error values as a matrix.
stde = reportbiv[Outline Box( 7 )][Number Col Box( 2 )] << Get as Matrix;

dvalues = [];
dvalues = samplesize |/ adjrsq |/ rsquare |/ corrvalue;
sfactor = term[2];

dlg = New Window( "Custom Report",
   Outline Box( "Selected Values",
      /* The Lineup box defines a two-column layout, each of which contains
      a Text Box. */
      Lineup Box( N Col( 2 ),
      Text Box( "Factor of Interest: " ),
      Text Box( sfactor ), ),
      tb = Table Box(
         /* Display an empty string in the first column
            and the text in the second column. */
         String Col Box( "  ",
            {"Sample Size: ", "Adjusted RSquare: ", "RSquare: ",
            "Correlation:"}
         ),
         // Insert a 30 pixel x 30 pixel spacer between the columns.
         Spacer Box( Size( 30, 30 ) ),
         ,
         /* Display an empty string in the first column
         and the dvalues in the second column. */
         Number Col Box( "  ", dvalues )
      ),
      // Insert a 1 x 30 spacer.
      Spacer Box( Size( 0, 30 ) ),
      ,
      Table Box(
         /* Display the cloned String Col Box followed by a spacer.
            Then insert the Parameter Estimates and Standard Error values. */
         CloneTerm,
         Spacer Box( Size( 10, 0 ) ),
         ,
         Number Col Box( "Estimate", est ),
         Spacer Box( Size( 10, 0 ) ),
```

```
            ,
            Number Col Box( "Standard Error", stde )
        )
    )
);
tb << Set Shade Headings( 0 ); // Turn off shaded table headings.
tb << Set Heading Column Borders( 0 ); // Turn off table column borders.
Close( sd ); // Close the data table.
```

Figure 17.4 Customized Report from the Bivariate Analysis



Create an Interactive Program

How do you gather numeric input from the users, perform a calculation on that input, and show the results in a new window?

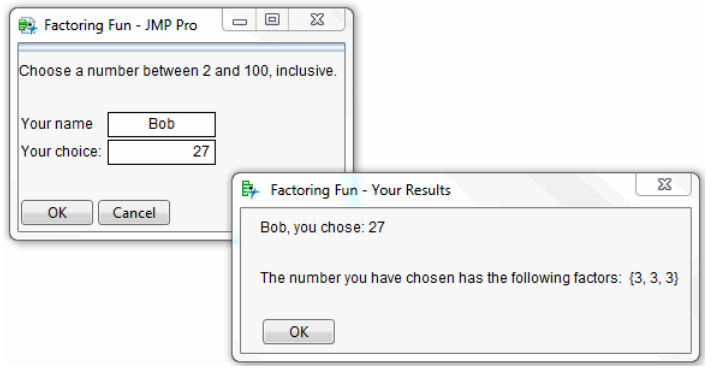
Prime Numbers.jsl asks the user to enter a number and then factors the number or confirms it as a prime number (Figure 17.5). This script is a good example of aligning several types of display boxes, concatenating text, and working with conditional functions.

```
/* How do you gather numeric input from the users, perform a
calculation on that input, and show the results in a new window?

This script demonstrates how to create an interactive program that
asks the user to enter a number and then factors the number or confirms
it as a prime number. */

// Ask the user to enter a name and number.
nw = New Window( "Factoring Fun",

V List Box(
   Text Box( "Choose a number between 2 and 100, inclusive. " ),
   Spacer Box( Size( 25, 25 ) )
    ),
    V List Box(
        Lineup Box(
            2,
            Text Box( "Your name   " ),
            uname = Text Edit Box( "< name > ", << Justify Text( Center ) ),
            Text Box( "Your choice " ),
            uprime = Number Edit Box( 2 )
        ),
        Spacer Box( Size( 25, 25 ) ),
        H List Box(
            Button Box( "OK",
         // Unload responses.
                username = uname << Get Text;
                fromUser0 = uprime << Get;

        // Test input for out of range condition.
                If( fromUser0 <= 1 | fromUser0 > 100,
           // Send message to user that input value is out of range.
                    nw2 = New Window( " Factoring Fun: Message for "
                      || username,
                        <<Modal,
                        Text Box(
                            "The number you chose, " || Char( fromUser0 ) ||
```

```
            " is not between 2 and 100, inclusive.  Please try
            again.  "
        ),
        Button Box( "OK" )

    ),
      /* Else the number is within range.
      Test for a prime number. If not prime, factor it.
      Create a vector which holds the prime numbers
      within specified range. */
    primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37,
    41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97];
    // Count the number of primes in the vector.
    p# = N Row( primes );

    isprime = 0; // Set flag.
    // Make a copy of the value for processing.
    fromuser1 = fromuser0;
    factors = {}; // Initialize list.

     /* Process the value by checking for prime then
     factoring if needed. */
    While( isPrime == 0,

        // Compare value to vector of prime numbers.
        If( Any( fromuser0 == primes ),
      // If found, place value in factor list.
            Insert Into( factors, fromUser0 );
            isPrime = 1 // Set condition to exit While loop.
            ;
        ); // End For loop.

        If( isprime == 0,
            For( q = 1, q <= p#, q++,
                If( Mod( fromuser0, primes[q] ) == 0,
                    fromUser0 = fromUser0 / primes[q];
                    Insert Into( factors, primes[q] );
                    q = p# + 1 // End if-then loop.
                    ;
                ); // End If loop.
            ); // End For loop.
        );  // End If/Then loop.

    ); // End while loop.

    cfUser0 = Char( fromUser1 );
```

```
                        nf = N Items( factors );
                        If( nf >= 2,
                            fmsg = "The number you have chosen has the following
                             factors: ",
                           fmsg = "The number you have chosen is a prime number: "
                        );
                // Show message to user about results.
                        nw3 = New Window( " Factoring Fun - Your Results",
                            <<Modal,
                            Text Box( username || ", you chose: " || cfUser0 ),
                            Spacer Box( Size( 25, 25 ) ),
                            Text Box( fmsg || " " || Char( factors ) ),
                            Spacer Box( Size( 25, 25 ) ),
                            Button Box( "OK" )
                        );
                    );
                ), // End the main OK button script.
                // Close the window and the program.
                Button Box( "Cancel", nw << Close Window )
            )
        )
    );
```

Figure 17.5 Factor Numbers Interactively



Appendix A

Compatibility Notes

Changes in Scripting in JMP 16

This appendix provides information about changes that affect compatibility in JSL scripts. Please update your scripts accordingly so that the scripts run correctly in the latest version of JMP.

Compatibility Issues in JMP 16.1

Legend Position

`Position()` has been phased out so that only `Legend Position()` is output in saved scripts. Previously saved scripts that include both the `Position()` and `Legend Position()` commands generate an error even though the script runs fine.

For example, change the following code:

```
{Legend Position( {3, [-4, -1], 5, [0, 1, 2]} ),
Position( {-4, -1, 0, 1, 2} )}
```

It should read as follows to avoid the error:

```
{Legend Position( {3, [-4, -1], 5, [0, 1, 2]} ),
Legend Position( {-4, -1, 0, 1, 2} )}
```

Compatibility Issues in JMP 16.0

Copy File(), Move File(), Rename File(), and Delete File()

In JMP 15.0, copying a file, moving a file, renaming a file, and deleting a file that didn't exist returned 0. Now, these actions throw an error.

Row Selection() and Case Sensitivity

`Match Case`, `Match Case()`, `Match Case(1)` and a missing `Match Case` do a case sensitive match in `Row Selection()`. `Match Case(0)` does a case insensitive match. Previously, `Row Selection()` matched case insensitively if `Match Case` was not supplied.

Target Level Column Property

You can use the new Target Level column property to specify the level of interest in binary logistic regression models. The default level is High. The main effects will be in the slope of the logistic probability curve, the signs of the linear model estimates, the profiler shape, and the ROC target. Be aware that some scripts that assume the target level in Fit Model or Fit Y by X logistic regression models should be checked.

Opening Multiple Data Tables

`Open()` returns a list of data tables if multiple data tables are created (for example, when importing PDF, HDF5, or Excel files). Previously, the `Open()` command returned a single table, even if multiple tables were created in the process.

Format(), Informat(), and Variations

The results of `Format()`, `Informat()` and their variations are written to the log if the format is invalid. Previously, they returned `m/d/y` when the format was invalid.

Graph Background Color Preference in Contour Plots

The Graph Background Color preference no longer affects the Contour Plot legend background color. Previously, the background color was applied to both the graph background and the legend background.

Set Auto Stretching and Get Auto Stretching for Display Boxes are Deprecated

Use `Set Stretch` and `Get Stretch` instead.

Appendix B

Efficient Scripts

Tips for Writing Scripts That Run Faster

This appendix provides ideas for writing scripts that run faster and more efficiently. For example, it is possible to control how much time is allocated to finding an optimal design. You can collapse outlines in a report that are not necessary. Or you can write static matrices, which is faster than developing them dynamically.

Contents

Using Matrices 943
Design of Experiments 944
Scripting Data Tables 944
Scripting Graphs and Display Boxes 945
Scripting Numbers, Strings, Arrays, and Lists 946
Programming Tips 946
Troubleshooting 946

Using Matrices

- The matrix concatenation operators, || and |/, add columns or rows to a matrix dynamically. However, the script runs faster if you define a matrix in a static fashion instead.

```
// faster example
t1 = Tick Seconds();
A = J( 10, 10000, . );
For( i = 1, i <= 10000, i++,
   a2 = J( 10, 1, Random Gamma( 1, 1 ) );
   // define a2 to be a 10 by 1 vector of random gamma values.
        A[0, i] = a2;
       /* populate the ith column of the matrix A to have the
         values in a2.  The first argument, 0, means "all rows". */
);
t2 = Tick Seconds();
Show( t2 - t1 );
   t2 - t1 = 0.0166666666627862;

// slower example
t3 = Tick Seconds();
B = J( 10, 0, . );
For( i = 1, i <= 10000, i++,
   a2 = J( 10, 1, Random Gamma( 1, 1 ) );
   B = B || a2;
);
t4 = Tick Seconds();
Show( t4 - t3 );
   t4 - t3 = 3.1333333333605;
```

- If you know the size of a matrix that you're going to need, it's typically better to allocate the matrix in its entirety first and then populate it. In the example below, the J() function is doing the allocation of the 100000x1 matrix all at the same time.

```
Delete Symbols( a1, a2 );
// faster example
t1 = Tick Seconds();
a1 = J( 100000, 1, Random Uniform() );
t2 = Tick Seconds();
Show( t2 - t1 );
Delete Symbols( a1 );
   t2 - t1 = 0;

// slower example
t3 = Tick Seconds();
a2 = J( 0, 0 );
```

```
For( i = 1, i <= 100000, i++,
   a2 |/= Random Uniform()
);
t4 = Tick Seconds();
Show( t4 - t3 );
   t4 - t3 = 5.05000000004657;
```

- Use matrices rather than a data table to store numbers.

Design of Experiments

In DOE, there are options such as `Number of Starts` and `Design Search Time` that enable you to control how much time is allocated to finding an optimal design.

```
// Number of Starts example
DOE(
   Custom Design,
   Add Factor( Continuous, -1, 1, "X1", 0 ),
   Add Factor( Continuous, -1, 1, "X2", 0 ),
   Add Factor( Continuous, -1, 1, "X3", 0 ),
   Number of Starts( 1000 ),
   Make Design
);
// Design Search Time example
d = DOE( Screening Design );
dt2 = Open( "$SAMPLE_DATA/Design Experiment/Weld Factors.jmp" );
d << Load Factors;
Close( dt2 );
d << {Set Random Seed( 12345 ), Screening Type( 1 ),
Suppress Cotter Designs, Design Search Time( 8 ),
Number of Column Starts( 50 ), Set Sample Size( 12 ), Make Design};
```

Scripting Data Tables

- Displaying a large data table can cause scripts to run slower. To speed things up, you can make a data table invisible, especially if you need to reference numbers in a report that will be used for future calculations or used in another platform. In this case, there isn't a need to see the current data table.

  ```
  dt = Open( "$SAMPLE_DATA/Big Class.jmp", invisible);
  ```

- Drawing objects is slow. Suppose that you have a graph with a lot of data. The platform does complicated calculations or the data table contains a formula. Adding a row can require a lot of processing. Use `Begin Data Update()` so you only redraw the graph once

instead of for each change. `Begin Data Update()` saves all update messages until `End Data Update()` is reached.

```
dt = Open( "MyData.jmp" );
dt << Distribution(
   Column( :"N=1"n, :"N=5"n), :"N=10"n )
);
Wait(1);
dt << Begin Data Update;
dt << Add Rows( 2000 );
dt << End Data Update;
```

- The use of functions might help speed things up, for example, using the `Summarize()` function rather than creating a Summary table. Maintaining a data table can be too much overhead. See "Store Summary Statistics in Global Variables".
- Data tables are great for keeping tabular data.
  - Keep data table strings under 23 characters. Longer strings have additional overhead.
  - Numeric values use less memory than character values.

Scripting Graphs and Display Boxes

- When displaying large numbers in graphs, you can set a preference to use Fast Markers.

  ```
  Preferences( Fast Marker Threshold( 25000 ) ); // 50000 is the default.
  ```

- Collapse some of the outline nodes in a report to speed up the script.

  ```
  SendToReport(
       Dispatch(
          {"Fit Life by X - Hours BY Arrhenius Celsius (Temp) Regression "},
          "Scatterplot",
          OutlineBox,
          {Close( 1 )}
       ),
  ```

- Construct your display boxes outside `New Window()` and then include them by reference in `New Window()`. This method saves the drawing until the end.

  ```
  tb = Text Box( "Select:" );
  lb = List Box( {"One", "Two", "Three"} );
  New Window( "test", tb, lb );
  ```

- The user might not need to see a report if the numbers will be used in another platform or for a future calculation. You can hide the report.

  ```
  dt= Open( "$SAMPLE_DATA/Big Class.jmp" );
  obj = Bivariate( Y( :Weight ), X( :Height ) );
  obj << Show Window( 0 ); // hide the window
  ```

```
obj << Fit Line( {Confid Curves Fit( 1 )} );
```

Submit the following script to see the report window. Notice that the confidence curves were fit even though the window was not shown.

```
obj << Show Window( 1 );
```

Scripting Numbers, Strings, Arrays, and Lists

- Use a faster algorithm. Replacing an O(n^2) algorithm with an O(n) algorithm is faster than anything else. See https://en.wikipedia.org/wiki/Big_O_notation.
- Avoid concatenating long strings. 100s of characters is OK; 1,000,000s of characters will be slow if you build the string in 1,000,000s of operations.
- Use numeric arrays (matrices, not lists).
  - Use numeric arrays with operators that operate on the entire array.
  - Avoid writing loops to manipulate numeric arrays.
- Use associative arrays for lookups.

Programming Tips

- Avoid `Break()`, `Continue()`, and `Return()` functions for speed. Using them might make it easier for you to read the script.
- Functions versus expressions:
  - User-defined functions have overhead for calling them but might make it easier for you to read the script.
  - Expressions have no overhead for calling but might make it more difficult for you to read the script.

Troubleshooting

Use the JSL Debugger to discover where the script is expending the most time. See “Debug or Profile Scripts”.

Appendix C

References

Morgan, J. (2010). "Expression Handling Functions: Part I." *JMPer Cable* 26:15–19. Accessed January 2, 2018. https://www.jmp.com/content/dam/jmp/documents/en/newsletters/jmper-cable/26_winter_2010.pdf.

Appendix D

Glossary

Terms, Concepts, and Placeholders

| In syntax summaries, | means "or" and separates possible choices. Usually choices separated by | are mutually exclusive. In other words, you have to pick one and cannot list several.

argument An argument is something specified inside the parentheses of a JSL function, message, and so forth. Big Class.jmp is the argument in `Open("Big Class.jmp")`.

You can often infer the meaning by the argument's position. For example, the values *200* and *100* in `size(200, 100)` are implicit arguments. The first value is always interpreted as the width; the second value is always interpreted as the height. See also *named argument*.

Boolean A Boolean value is a yes/no value, something that is on or off, shown or hidden, true or false, 1 or 0, yes or no. A value listed as *being a Boolean value* is one that evaluates to true or false (or missing).

col In syntax summaries, a placeholder for any reference to a data table column. For example, `Column("age")`.

command A generic description for a JSL statement that performs an action. The *Scripting Guide* prefers the more specific terms *function* and *message* when they are applicable.

current data table The current data table is the data table that `Current Data Table()` either returns or is assigned.

current row The current row for scripting is defined to be zero (no row) by default. You can set a current row with `Row()` or `For Each Row`, and so forth.

database Although the term is much more general, for JMP's purposes, the word "database" describes any external data source (such as SQL) accessed through ODBC with JSL's `Open Database` command.

Datafeed A Datafeed is a method to read real-time data continuously, such as from a laboratory measurement device connected to a serial port.

db In syntax summaries, a placeholder for any reference to a display box. For example, `report(Bivariate[1])`.

dt In syntax summaries, a placeholder for any reference to a data table. For example, `Current Data Table()` or `Data Table("Big Class.jmp")`.

eliding operator An eliding operator is one that causes arguments on either side to combine and evaluate differently than if the statement were evaluated strictly left to right. For example, `12<a<13` is a range check to test whether *a* is between 12 and 13: JMP reads the whole expression before evaluating. If < did not elide, the expression would be evaluated left to right as `(12<a)<13`. In other words, it would check whether the result of the comparison (1 or 0, for false or true) is below 13, which of course would always yield 1 for true. The << operator (for `object<<message`, which is equivalent to `Send(object, message)` is another example of an eliding operator.

function A function takes an argument or series of arguments inside parentheses after the function name. For example, the infix operator + has a function equivalent `Add()`. The statements `3 + 4` and `Add(3, 4)` are equivalent. All JSL's operators have function equivalents, but not all functions have operator equivalents. For example, `Sqrt(a)` can be represented only by the function. Also see the `Function` operator for storing a function under a name.

global variable A global variable is a name to hold values that exists for the remainder of a session. Globals can contain many types of values, including numbers, strings, lists, or references to objects. They are called globals because they can be referred to almost anywhere, not just in some specific context.

head The head of a JSL expression is the expression without any of its arguments. For example, the head of the expression `Assign( x, 100.1 )` is `Assign()`.

infix operator An infix operator takes one argument on each side, such as + in arithmetic, `3 + 4` , or the = in an assignment, `a=7`.

L-value Something that can be the destination of an assignment. In this manual, L-value describes an expression that normally returns its current value but that can alternatively receive an assignment to set its value. For example, you would ordinarily use a function such as `Row()` to get the current row number and assign it to something else. For example, `x=Row()`. However, since `Row` is an L-value, you can also place it on the left side of an assignment to set its value. For example, `Row()=10`.

list A list is a multiple-item data type entered in special brace { } notation or with the `List` function. Lists enable scripts to work with many things at once, often in the place of a single thing.

matrix A matrix is a JMP data type for a rectangular array of rows and columns of number. In JSL, matrices are entered in bracket `[ ]` notation or with the `Matrix` function.

message A message is a JSL statement that is directed to an *object*, which knows how to execute the message.

metadata In JMP data tables, metadata are data about the data, such as the source of the data, comments about each variable, scripts for working with the data, and so on.

mousedown An event generated by pressing down the mouse button. See "Handle()" and "Mousetrap()".

mouseup An event generated by releasing the mouse button. See "Handle()" and "Mousetrap()".

name A name is a reference to a JSL object. For example, when you assign the numeric value 3 to a global variable in the statement `a=3`, "a" is a name.

namespace A namespace is a collection of unique names and corresponding values. Namespaces are useful for avoiding name collisions between different scripts.

named argument A named argument is an optional argument that you select from a predetermined set and explicitly define. For example, `HexToNumber(hextext, <Base(number)>)` has a single named argument, `Base`. This is the fixed name that must directly be in the `HexToNumber` argument.

ODBC database The Microsoft standard for Open DataBase Connectivity. JSL supports access to any ODBC-enabled data source through the `Open Database` command.

obj In syntax summaries, a placeholder for any reference to an analysis platform. For example, `Bivariate[1]`.

object An object is a dynamic entity in JMP, such as a data table, a data column, a platform results window, a graph, and so forth. Most objects can receive messages telling them to act on themselves in some way.

operator Usually operator refers to a one- or two-character symbol such as + for addition or <= for less than or equal to.

POSIX POSIX is an acronym for Portable Operating System Interface and is a registered trademark of the IEEE. POSIX pathnames enable you to use one syntax for paths for any operating system, instead of having to use a different syntax for each.

postfix operator A postfix operator takes on argument on its left side (before the operator), such as `a++` for postincrement or `a--` for postdecrement.

pre-evaluated statistics Statistics that are calculated once and used as constants thereafter.

prefix operator A prefix operator takes one argument on its right side (after the operator), such as `!a` for negation.

reference A way to address a scriptable object in order to send it messages. For example,. `column("age")` or `Current Data Table()` or `Bivariate[1]`. Typically a reference is stored in a *global variable* for convenience.

row state A data element type to store any combination of the following attributes for data rows: excluded, hidden, labeled, selected, color, marker, hue, shade.

scalar A simple non-matrix numeric value.

scoping operator A scoping operator forces a name to be interpreted as a particular type of data element, for example the `:` operator in `:name` forces *`name`* to be resolved as a column; the `::` operator in `::name` forces *`name`* to be resolved as a global variable. In the rare case that you do want to share specific variables among all opened projects, use the `:::` root operator.

toggle Omitting the Boolean argument for a row state command toggles the setting. If the option is off, the message turns it on. If the option is on, the message turns it off. Sending such a command repeatedly flips back and forth between on and off. If you include the Boolean argument, the command sets an absolute on or off state, and sending the command repeatedly has no further effect. For all other messages, omitting the Boolean argument enables the option.

vector A matrix with only one column or row.

Appendix E

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.

  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
- You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS”, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Version 16

JSL Syntax Reference

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

Marcel Proust

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

16.1

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020–2021. *JMP® 16 JSL Syntax Reference*. Cary, NC: SAS Institute Inc.

JMP® 16 JSL Syntax Reference

Copyright © 2020–2021, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

March 2021

August 2021

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.

Get the Most from JMP

Whether you are a first-time or a long-time user, there is always something to learn about JMP.

Visit JMP.com to find the following:

- live and recorded webcasts about how to get started with JMP
- video demos and webcasts of new features and advanced techniques
- details on registering for JMP training
- schedules for seminars being held in your area
- success stories showing how others use JMP
- the JMP user community, resources for users including examples of add-ins and scripts, a forum, blogs, conference information, and so on

https://www.jmp.com/getstarted

Contents

JSL Syntax Reference

1 Learn about JMP 9

Documentation and Additional Resources

Formatting Conventions in JMP Documentation 11
JMP Help 12
JMP Documentation Library 12
Additional Resources for Learning JMP 18
JMP Tutorials 18
Sample Data Tables 18
Learn about Statistical and JSL Terms 19
Learn JMP Tips and Tricks 19
JMP Tooltips 19
JMP User Community 20
Free Online Statistical Thinking Course 20
JMP New User Welcome Kit 20
Statistics Knowledge Portal 20
JMP Training 20
JMP Books by Users 21
The JMP Starter Window 21
JMP Technical Support 21

2 JSL Functions 23

Summary of Functions, Operators, and Messages

Assignment Functions 25
Character Functions 29
Character Pattern Functions 44
Comment Functions 56
Comparison Functions 57
Conditional and Logical Functions 62
Constant Functions 75
Date and Time Functions 76
Discrete Probability Functions 85
Display Functions 91
Expression Functions 127

File Functions ... 129
Financial Functions ... 151
Graphics Functions ... 156
HTTP Functions ... 176
List Functions ... 176
MATLAB Integration Functions ... 183
MATLAB JSL Function Interfaces ... 183
Matrix Functions ... 190
Numeric Functions ... 214
Optimization Functions ... 217
Probability Functions ... 221
Programming Functions ... 249
Python Integration Functions ... 268
R Integration Functions ... 274
Random Functions ... 279
Row Functions ... 290
Row State Functions ... 296
SAS Integration Functions ... 299
SQL Functions ... 318
Statistical Functions ... 320
Transcendental Functions ... 335
Trigonometric Functions ... 341
Utility Functions ... 344

3 JSL Messages ... 377
Summary of Messages for Objects and Display Boxes

Alpha Shape ... 380
Associative Arrays ... 380
Classes ... 381
Data Tables ... 383
Columns ... 414
Rows ... 423
Data Filter ... 424
Data Feed (Windows Only) ... 427
Display Boxes ... 430
All Display Boxes ... 430
Axis Boxes ... 440
Border Boxes ... 445
Data Browser Boxes ... 446
Data Filter Source Boxes ... 446

Frame Boxes . . . 446
Display 3D Boxes . . . 448
Excerpt Boxes . . . 449
Filter Col Selector . . . 449
Global Boxes . . . 449
Hier Boxes . . . 449
Matrix Boxes . . . 449
Nom Axis Boxes . . . 450
Number Col Boxes . . . 451
Number Col Edit Boxes . . . 454
Number Edit Box . . . 454
Outline Boxes . . . 455
Panel Boxes . . . 456
Plot Col Boxes . . . 456
Slider Boxes and Range Slider Boxes . . . 457
String Col Boxes . . . 458
Tab Boxes . . . 459
Table Boxes . . . 460
Text Boxes . . . 463
Tree Node and Tree Box . . . 464
Triangulation . . . 466
Windows . . . 468
Dynamic Link Libraries (DLLs) . . . 470
HTML 5 . . . 472
Web Report . . . 472
Images . . . 473
JMP Applications . . . 476
JMP App . . . 477
JMP App Module . . . 478
JMP App Module Instance . . . 479
MATLAB . . . 479
Namespaces . . . 483
Platforms . . . 485
Bubble Plot . . . 490
DOE . . . 491
Partition . . . 491
Response Screening . . . 492
Tabulate . . . 492
Python Integration Messages . . . 493
R Integration Messages . . . 497

SAS Integration Messages 500
Metadata Server Objects 500
SAS Server Objects 501
Stored Processes 510
SAS Results 516
Schedule 519
Segments 520
Sockets 520
SQL 523
Other Objects 527
Zip Archives 527
Journals 528

A SQL Functions Available for JMP Queries 529

Numeric SQL Functions 531
Date-Time SQL Functions 532
String SQL Functions 535
System SQL Functions 536
Aggregate SQL Functions 536

B References 539

C Technology License Notices 541

Chapter 1

Learn about JMP

Documentation and Additional Resources

Learn about JMP documentation, such as book conventions, descriptions of each JMP document, the Help system, and where to find additional support.

Contents

Formatting Conventions in JMP Documentation . . . 11
JMP Help . . . 12
JMP Documentation Library . . . 12
Additional Resources for Learning JMP . . . 18
JMP Tutorials . . . 18
Sample Data Tables . . . 18
Learn about Statistical and JSL Terms . . . 19
Learn JMP Tips and Tricks . . . 19
JMP Tooltips . . . 19
JMP User Community . . . 20
Free Online Statistical Thinking Course . . . 20
JMP New User Welcome Kit . . . 20
Statistics Knowledge Portal . . . 20
JMP Training . . . 20
JMP Books by Users . . . 21
The JMP Starter Window . . . 21
JMP Technical Support . . . 21

Formatting Conventions in JMP Documentation

These conventions help you relate written material to information that you see on your screen:

- Sample data table names, column names, pathnames, filenames, file extensions, and folders appear in Helvetica (or sans-serif online) font.
- Code appears in `Lucida Sans Typewriter` (or monospace online) font.
- Code output appears in *`Lucida Sans Typewriter`* italic (or monospace italic online) font and is indented farther than the preceding code.
- **Helvetica bold** formatting (or bold sans-serif online) indicates items that you select to complete a task:
  - buttons
  - check boxes
  - commands
  - list names that are selectable
  - menus
  - options
  - tab names
  - text boxes
- The following items appear in italics:
  - words or phrases that are important or have definitions specific to JMP
  - book titles
  - variables
- Features that are for JMP Pro only are noted with the JMP Pro icon JMP PRO. For an overview of JMP Pro features, visit https://www.jmp.com/software/pro.

Note: Special information and limitations appear within a Note.

Tip: Helpful information appears within a Tip.

JMP Help

JMP Help in the Help menu enables you to search for information about JMP features, statistical methods, and the JMP Scripting Language (or *JSL*). You can open JMP Help in several ways:

- Search and view JMP Help on Windows by selecting **Help > JMP Help**.
- On Windows, press the F1 key to open the Help system in the default browser.
- Get help on a specific part of a data table or report window. Select the Help tool from the **Tools** menu and then click anywhere in a data table or report window to see the Help for that area.
- Within a JMP window, click the **Help** button.

Note: The JMP Help is available for users with Internet connections. Users without an Internet connection can search all books in a PDF file by selecting **Help > JMP Documentation Library**. See "JMP Documentation Library" for more information.

JMP Documentation Library

The Help system content is also available in one PDF file called *JMP Documentation Library*. Select **Help > JMP Documentation Library** to open the file. You can also download the Documentation PDF Files add-in if you prefer searching individual PDF files of each document in the JMP library. Download the available add-ins from https://community.jmp.com.

The following table describes the purpose and content of each document in the JMP library.

Document Title	Document Purpose	Document Content
Discovering JMP	If you are not familiar with JMP, start here.	Introduces you to JMP and gets you started creating and analyzing data. Also learn how to share your results.
Using JMP	Learn about JMP data tables and how to perform basic operations.	Covers general JMP concepts and features that span across all of JMP, including importing data, modifying columns properties, sorting data, and connecting to SAS.

Document Title	Document Purpose	Document Content
Basic Analysis	Perform basic analysis using this document.	Describes these Analyze menu platforms: • Distribution • Fit Y by X • Tabulate • Text Explorer Covers how to perform bivariate, one-way ANOVA, and contingency analyses through Analyze > Fit Y by X. How to approximate sampling distributions using bootstrapping and how to perform parametric resampling with the Simulate platform are also included.
Essential Graphing	Find the ideal graph for your data.	Describes these Graph menu platforms: • Graph Builder • Scatterplot 3D • Contour Plot • Bubble Plot • Parallel Plot • Cell Plot • Scatterplot Matrix • Ternary Plot • Treemap • Chart • Overlay Plot The book also covers how to create background and custom maps.
Profilers	Learn how to use interactive profiling tools, which enable you to view cross-sections of any response surface.	Covers all profilers listed in the Graph menu. Analyzing noise factors is included along with running simulations using random inputs.

Document Title	Document Purpose	Document Content
Design of Experiments Guide	Learn how to design experiments and determine appropriate sample sizes.	Covers all topics in the DOE menu.
Fitting Linear Models	Learn about Fit Model platform and many of its personalities.	Describes these personalities, all available within the Analyze menu Fit Model platform: • Standard Least Squares • Stepwise • Generalized Regression • Mixed Model • MANOVA • Loglinear Variance • Nominal Logistic • Ordinal Logistic • Generalized Linear Model

Document Title	Document Purpose	Document Content
Predictive and Specialized Modeling	Learn about additional modeling techniques.	Describes these Analyze > Predictive Modeling menu platforms: • Neural • Partition • Bootstrap Forest • Boosted Tree • K Nearest Neighbors • Naive Bayes • Support Vector Machines • Model Comparison • Model Screening • Make Validation Column • Formula Depot Describes these Analyze > Specialized Modeling menu platforms: • Fit Curve • Nonlinear • Functional Data Explorer • Gaussian Process • Time Series • Matched Pairs Describes these Analyze > Screening menu platforms: • Modeling Utilities • Response Screening • Process Screening • Predictor Screening • Association Analysis • Process History Explorer

Document Title	Document Purpose	Document Content
Multivariate Methods	Read about techniques for analyzing several variables simultaneously.	Describes these Analyze > Multivariate Methods menu platforms: • Multivariate • Principal Components • Discriminant • Partial Least Squares • Multiple Correspondence Analysis • Structural Equation Models • Factor Analysis • Multidimensional Scaling • Item Analysis Describes these Analyze > Clustering menu platforms: • Hierarchical Cluster • K Means Cluster • Normal Mixtures • Latent Class Analysis • Cluster Variables
Quality and Process Methods	Read about tools for evaluating and improving processes.	Describes these Analyze > Quality and Process menu platforms: • Control Chart Builder and individual control charts • Measurement Systems Analysis • Variability / Attribute Gauge Charts • Process Capability • Model Driven Multivariate Control Chart • Legacy Control Charts • Pareto Plot • Diagram • Manage Spec Limits • OC Curves

Document Title	Document Purpose	Document Content
Reliability and Survival Methods	Learn to evaluate and improve reliability in a product or system and analyze survival data for people and products.	Describes these Analyze > Reliability and Survival menu platforms: • Life Distribution • Fit Life by X • Cumulative Damage • Recurrence Analysis • Degradation • Destructive Degradation • Reliability Forecast • Reliability Growth • Reliability Block Diagram • Repairable Systems Simulation • Survival • Fit Parametric Survival • Fit Proportional Hazards
Consumer Research	Learn about methods for studying consumer preferences and using that insight to create better products and services.	Describes these Analyze > Consumer Research menu platforms: • Categorical • Choice • MaxDiff • Uplift • Multiple Factor Analysis
Scripting Guide	Learn about taking advantage of the powerful JMP Scripting Language (JSL).	Covers a variety of topics, such as writing and debugging scripts, manipulating data tables, constructing display boxes, and creating JMP applications.
JSL Syntax Reference	Read about many JSL functions on functions and their arguments, and messages that you send to objects and display boxes.	Includes syntax, examples, and notes for JSL commands.

Additional Resources for Learning JMP

In addition to reading JMP help, you can also learn about JMP using the following resources:

- "JMP Tutorials"
- "Sample Data Tables"
- "Learn about Statistical and JSL Terms"
- "Learn JMP Tips and Tricks"
- "JMP Tooltips"
- "JMP User Community"
- "Free Online Statistical Thinking Course"
- "JMP New User Welcome Kit"
- "Statistics Knowledge Portal"
- "JMP Training"
- "JMP Books by Users"
- "The JMP Starter Window"

JMP Tutorials

You can access JMP tutorials by selecting **Help > Tutorials**. The first item on the **Tutorials** menu is **Tutorials Directory**. This opens a new window with all the tutorials grouped by category.

If you are not familiar with JMP, start with the **Beginners Tutorial**. It steps you through the JMP interface and explains the basics of using JMP.

The rest of the tutorials help you with specific aspects of JMP, such as designing an experiment and comparing a sample mean to a constant.

Sample Data Tables

All of the examples in the JMP documentation suite use sample data. Select **Help > Sample Data Library** to open the sample data directory.

To view an alphabetized list of sample data tables or view sample data within categories, select **Help > Sample Data**.

Sample data tables are installed in the following directory:

On Windows: C:\Program Files\SAS\JMP\16\Samples\Data

On macOS: \Library\Application Support\JMP\16\Samples\Data

In JMP Pro, sample data is installed in the JMPPRO (rather than JMP) directory.

To view examples using sample data, select **Help > Sample Data** and navigate to the Teaching Resources section. To learn more about the teaching resources, visit https://jmp.com/tools.

Learn about Statistical and JSL Terms

For help with statistical terms, select Help > Statistics Index. For help with JSL scripting and examples, select **Help > Scripting Index**.

Statistics Index Provides definitions of statistical terms.

Scripting Index Lets you search for information about JSL functions, objects, and display boxes. You can also edit and run sample scripts from the Scripting Index and get help on the commands.

Learn JMP Tips and Tricks

When you first start JMP, you see the Tip of the Day window. This window provides tips for using JMP.

To turn off the Tip of the Day, clear the **Show tips at startup** check box. To view it again, select **Help > Tip of the Day**. Or, you can turn it off using the Preferences window.

JMP Tooltips

JMP provides descriptive tooltips (or *hover labels*) when you hover over items, such as the following:

- Menu or toolbar options
- Labels in graphs
- Text results in the report window (move your cursor in a circle to reveal)
- Files or windows in the Home Window
- Code in the Script Editor

Tip: On Windows, you can hide tooltips in the JMP Preferences. Select **File > Preferences > General** and then deselect **Show menu tips**. This option is not available on macOS.

JMP User Community

The JMP User Community provides a range of options to help you learn more about JMP and connect with other JMP users. The learning library of one-page guides, tutorials, and demos is a good place to start. And you can continue your education by registering for a variety of JMP training courses.

Other resources include a discussion forum, sample data and script file exchange, webcasts, and social networking groups.

To access JMP resources on the website, select **Help > JMP User Community** or visit https://community.jmp.com.

Free Online Statistical Thinking Course

Learn practical statistical skills in this free online course on topics such as exploratory data analysis, quality methods, and correlation and regression. The course consists of short videos, demonstrations, exercises, and more. Visit https://www.jmp.com/statisticalthinking.

JMP New User Welcome Kit

The JMP New User Welcome Kit is designed to help you quickly get comfortable with the basics of JMP. You'll complete its thirty short demo videos and activities, build your confidence in using the software, and connect with the largest online community of JMP users in the world. Visit https://www.jmp.com/welcome.

Statistics Knowledge Portal

The Statistics Knowledge Portal combines concise statistical explanations with illuminating examples and graphics to help visitors establish a firm foundation upon which to build statistical skills. Visit https://www.jmp.com/skp.

JMP Training

SAS offers training on a variety of topics led by a seasoned team of JMP experts. Public courses, live web courses, and on-site courses are available. You might also choose the online e-learning subscription to learn at your convenience. Visit https://www.jmp.com/training.

JMP Books by Users

Additional books about using JMP that are written by JMP users are available on the JMP website. Visit https://www.jmp.com/books.

The JMP Starter Window

The JMP Starter window is a good place to begin if you are not familiar with JMP or data analysis. Options are categorized and described, and you launch them by clicking a button. The JMP Starter window covers many of the options found in the Analyze, Graph, Tables, and File menus. The window also lists JMP Pro features and platforms.

- To open the JMP Starter window, select **View** (**Window** on macOS) **> JMP Starter**.
- To display the JMP Starter automatically when you open JMP on Windows, select **File > Preferences > General**, and then select **JMP Starter** from the Initial JMP Window list. On macOS, select **JMP > Preferences > Initial JMP Starter Window**.

JMP Technical Support

JMP technical support is provided by statisticians and engineers educated in SAS and JMP, many of whom have graduate degrees in statistics or other technical disciplines.

Many technical support options are provided at https://www.jmp.com/support, including the technical support phone number.

Chapter 2

JSL Functions

Summary of Functions, Operators, and Messages

This topic provides abbreviated descriptions for many of JMP's functions, operators, and general object messages. For complete information about functions, see the JMP Scripting Index. In JMP, select **Help > Scripting Index**.

For information about platform messages, see the *Scripting Guide*.

Contents

Assignment Functions . . . 25
Character Functions . . . 28
Character Pattern Functions . . . 43
Comment Functions . . . 53
Comparison Functions . . . 54
Conditional and Logical Functions . . . 59
Constant Functions . . . 71
Date and Time Functions . . . 72
Discrete Probability Functions . . . 80
Display Functions . . . 86
Expression Functions . . . 120
File Functions . . . 122
Financial Functions . . . 144
Graphics Functions . . . 149
HTTP Functions . . . 168
List Functions . . . 169
MATLAB Integration Functions . . . 175
    MATLAB JSL Function Interfaces . . . 175
Matrix Functions . . . 182
Numeric Functions . . . 206
Optimization Functions . . . 210
Probability Functions . . . 213
Programming Functions . . . 243
Python Integration Functions . . . 260
R Integration Functions . . . 266
Random Functions . . . 271
Row Functions . . . 282
Row State Functions . . . 288
SAS Integration Functions . . . 291
SQL Functions . . . 309
Statistical Functions . . . 312
Transcendental Functions . . . 326
Trigonometric Functions . . . 332
Utility Functions . . . 335

Assignment Functions

JSL also provides operators for in-place arithmetic, or *assignment operators*. These operations are all done in place, meaning that the result of the operation is assigned to the first argument. The most basic assignment operator is the = operator (or the equivalent function `Assign`). For example, if *a* is 3 and you do `a+=4`, then *a* becomes 7.

The first argument to an assignment function must be something capable of being assigned (an *L-value*). You cannot do something like `3+=4`, because 3 is just a value and cannot be reassigned. However, you can do something like `a+=4`, because *a* is a variable whose value you can set.

`Add To(`*a*`, `*b*`)`

a+=*b*

Description

Adds *a* and *b* and places the sum into *a*.

Returns

The sum.

Arguments

a Must be a variable.

b Can be a variable, a list, a number, or a matrix.

Notes

The first argument must be a variable, because its value must be able to accept a value change. A number as the first argument produces an error.

For `Add To()`: Only two arguments are permitted. If one or no argument is specified, `Add To()` returns a missing value. Any arguments after the first two are ignored.

For `a+=b`: More than two arguments can be strung together. JMP evaluates pairs from right to left, and each sum is placed in the left-hand variable. All arguments except the last must be a variable.

Example

```
a+=b+=c
```

JMP adds *b* and *c* and places the sum into *b*. Then JMP adds *a* and *b* and places the sum into *a*.

`Assign(a, b)`

a=b

Description

Places the value of *b* into *a*.

Returns

The new value of *a*.

Arguments

a Must be a variable.

b Can be a variable, number, or matrix.

Notes

a must be a variable, because it must be able to accept a value change. A number as the first argument produces an error. If *b* is some sort of expression, it's evaluated first and the result is placed into *a*.

`Divide To(a, b)`

a/=b

Description

Divides *a* by *b* and places the result into *a*.

Returns

The quotient.

Arguments

a Must be a variable.

b Can be a variable, number, or matrix.

`Multiply To(a, b)`

a=b*

Description

Multiplies *a* and *b* and places the product into *a*.

Returns

The product.

Arguments

a Must be a variable.

b Can be a variable, number, or matrix.

Notes

The first argument must be a variable, because its value must be able to accept a value change. A number as the first argument produces an error.

For `Multiply To()`: Only two arguments are permitted. If one or no argument is specified, `Multiply To()` returns a missing value. Any arguments after the first two are ignored.

For *a*`*=`*b*: More than two arguments can be strung together. JMP evaluates pairs from right to left, and each product is placed in the left-hand variable. All arguments except the last must be a variable.

Example

```
a*=b*=c
```

JMP multiplies *b* and *c* and places the product into *b*. Then JMP multiplies *a* and *b* and places the product into *a*.

`PostDecrement(`*a*`)`

a`--`

Description

Post-decrement. Subtracts 1 from *a* and places the difference into *a*.

Returns

a-1

Argument

a Must be a variable.

Notes

If *a*`--` or `Post Decrement(`*a*`)` is inside another expression, the expression is evaluated first, and then the decrement operation is performed. This expression is mostly used for loop control.

`Post Increment(`*a*`)`

a`++`

Description

Post-increment. Adds 1 to *a* and places the sum into *a*.

Returns

a+1

Argument

a Must be a variable.

Notes

If `a++` or `Post Increment(a)` is inside another expression, the expression is evaluated first, and then the increment operation is performed. Mostly used for loop control.

Subtract To(*a*, *b*)

a-=*b*

Description

Subtracts *b* from *a* and places the difference into *a*.

Returns

The difference.

Arguments

a Must be a variable.

b Can be a variable, number, or matrix.

Notes

The first argument must be a variable, because its value must be able to accept a value change. A number as the first argument produces an error.

For SubtractTo(): Only two arguments are permitted. If fewer than two or more than two arguments is specified, SubtractTo() returns a missing value.

For *a*-=*b*: More than two arguments can be strung together. JMP evaluates pairs from right to left, and each difference is placed in the left-hand variable. All arguments except the last must be a variable.

Example

```
a-=b-=c
```

JMP subtracts *c* from *b* and places the difference into *b*. Then JMP subtracts *b* from *a* and places the difference into *a*.

Character Functions

Most character functions take character arguments and quoted return character strings, although some take numeric arguments or return numeric data. Arguments that are literal character strings must be enclosed in quotation marks.

BLOB To Char(*blob*, <Encoding="*enc*">)

Description

Reinterpret binary data as a quoted Unicode string.

Returns

A quoted string.

Required Argument

blob A binary large object.

Optional Argument

Encoding A quoted string that specifies an encoding. The default encoding for the character string is "utf-8". "utf-16le", "utf-16be", "us-ascii", "iso-8859-1", "ascii~hex", "shift-jis", and "euc-jp" are also supported.

Notes

The optional argument "ascii~hex" is intended to make conversions of blobs containing normal ASCII data simpler when the data might contain CR, LF, or TAB characters (for example) and those characters do not need any special attention.

BLOB To Matrix(*blob*, *type*, *bytes*, *endian*, <*nCols*>)

Description

Creates a matrix by converting each byte in the blob to numbers.

Returns

A matrix that represents the blob.

Required Arguments

blob A blob or reference to a blob.

type A quoted string that contains the named type of number. The options are "int", "uint", or "float".

bytes Byte size of the data in the blob. Options are 1, 2, 4, or 8.

endian The quoted endian-ness of your system: "Big" (the first byte is most significant), "Little" (the first byte is the least significant), or "Native" (the machine's native format).

Optional Argument

<*nCols*> The number of columns in the matrix. The default value is 1.

Char(*x*, <*width*>, <*decimal*>, < <<Use Locale(*Boolean*)>)

Description

Converts an expression or numeric value into a quoted character string.

Returns

A quoted string.

Required Argument

x an expression or a numeric value. An expression must be quoted with Expr(). Otherwise, its evaluated value is converted to a quoted string.

Optional Arguments

width A number that sets the maximum number of characters in the quoted string.

decimal A number that sets the maximum number of places after the decimal that is included in the quoted string.

Use Locale(*Boolean*) Preserves locale-specific numeric formatting.

Example

```
Char( Pi(), 10, 4)
   "3.1416"

Char( Pi(), 3, 4)
   "3.1"
```

Notes

The *width* argument overrides the *decimal* argument.

Char To BLOB(*string*, <encoding="*enc*">)

Description

Converts a quoted string of characters into a binary (blob).

Returns

A binary object.

Required Argument

string A quoted string or a reference to a string.

Optional Argument

encoding A quoted string that specifies an encoding. The default encoding for the blob is "utf-8". "utf-16le", "utf-16be", "us-ascii", "iso-8859-1", "ascii~hex", "shift-jis", and "euc-jp" are also supported.

Notes

Converting BLOBs into printable format escapes \ (in addition to ~ " ! and characters outside of the printable ASCII range) into hex notation (~5C for the backslash character).

```
x = Char To BLOB( "abc\def\!n" );
y = BLOB To Char( x, encoding = "ASCII~HEX" );
If(
   y == "abc~5Cdef~0A", "JMP 12.2 and later behavior",
   y == "abc\def~0A", "Pre-JMP 12.2 behavior"
);
   "JMP 12.2 and later behavior" // output
```

Char To Hex(*value*, <*integer*|encoding="*enc*">)

Hex(*value*, <*integer*|encoding="*enc*"|Base(*number*)|Pad To(*number*)>)

Description

Returns the hexadecimal (or other base number system) text corresponding to the given value and encoding, which can be a number a quoted string or a blob. If the value is a number, IEEE 754 64-bit encoding is used unless one of the optional arguments, *integer*, or *Base*, is provided.

Required Argument

value Any number, quoted string, or blob.

Optional Arguments

`integer` A quoted switch that causes the value to be interpreted as an integer.

`encoding` A quoted string that specifies an encoding. The default encoding is `"utf-8"`. `"utf-16le"`, `"utf-16be"`, `"us-ascii"`, `"iso-8859-1"`, `"ascii~hex"`, `"shift-jis"`, and `"euc-jp"` are also supported.

`Base(`*`number`*`)` An integer value between 2 and 36 inclusive. If base is specified, the function returns the text corresponding to the specified number in that base number system instead of hexadecimal.

`Pad To(`*`number`*`)` A value to specify the padded width of the hex output.

`Collapse Whitespace(`*`text`*`)`

Description

Trims leading and trailing whitespace and replaces interior whitespace with a single space. That is, if more than one white space character is present, the `Collapse Whitespace` function replaces the two spaces with one space.

Returns

A quoted string.

Required Argument

`text` A quoted string.

`Concat(`*`a`*`, `*`b`*`)`

`Concat(`*`A`*`, `*`B`*`)`

a||*b*

A||*B*

Description

For quoted strings: Appends the string *b* to the string *a*. Neither argument is changed.

For lists: Appends the list *b* to the list *a*. Neither argument is changed.

For matrices: Horizontal concatenation of two matrices, A and B.

Returns

For quoted strings: A quoted string composed of the string *a* directly followed by the string *b*.

For lists: A list composed of the list *a* directly followed by the list *b*.

For matrices: A matrix.

Arguments

Two or more quoted strings, quoted string variables, lists, or matrices.

Example

```
a = "Hello"; b = " "; c = "World"; a || b || c;
```

```
   "Hello World"
d = {"apples", "bananas"}; e = {"peaches", "pears"}; Concat( d, e );
   {"apples", "bananas", "peaches", "pears"}
A = [1 2 3]; B = [4 5 6]; Concat( A, B );
   [1 2 3 4 5 6]
```

Notes

More than two arguments can be strung together. Each additional quoted string is appended to the end, in left to right order. Each additional matrix is appended in left to right order.

Concat Items

See "Concat Items({string1, string2, ...}, <delimiter>})".

Concat To(*a*, *b*)

Concat To(*a*, *b*)

a||=*b*

A||=*B*

Description

For quoted strings: Appends the string *b* to the string *a* and places the new concatenated string into *a*.

For matrices: Appends the matrix *b* to the matrix *a* and places the new concatenated matrix into *a*.

For lists: Appends the list *b* to the list and places the new concatenated list into *a*.

Returns

For quoted strings: A string composed of the string *a* directly followed by the string *b*.

For matrices: A matrix.

For lists: A list composed of the list *a* directly followed by the list *b*.

Arguments

Two or more quoted strings, quoted string variables, matrices, or lists. The first variable must be a variable whose value can be changed.

Notes

More than two arguments can be strung together. Each additional quoted string, matrix, or list is appended to the end, in left to right order.

Example

```
a = "Hello"; b = " "; c = "World"; Concat To( a, b, c ); Show( a );
   a = "Hello World"
A = [1 2 3]; B = [4 5 6]; Concat To( A, B ); Show( A );
   A = [1 2 3 4 5 6];
```

```
d = {"apples", "bananas"}; e = {"peaches", "pears"}; Concat to(d,e); Show( d
);
d = {"apples", "bananas", "peaches", "pears"};
```

Contains(*whole*, *part*, <*start*>)

Description

Determines whether *part* is contained within *whole*.

Returns

If *part* is found: For lists, quoted strings, and namespaces, the numeric position where the first occurrence of *part* is located. For associative arrays, 1.

If *part* is not found, 0 is returned in all cases.

Required Arguments

whole A quoted string, list, namespace, or associative array.

part For a quoted string or namespace, a string that can be part of the string *whole*. For a list, an item that can be an item in the list *whole*. For an associative array, a key that can be one of the keys in the map *whole*.

Optional Argument

start A numeric argument that specifies a starting point within *whole*. If *start* is negative, contains searches *whole* for *part* backwards, beginning with the position specified by the length of *whole* – *start*. Note that *start* is meaningless for associative arrays and is ignored.

Example

```
nameList={"Katie", "Louise", "Jane", "Jaclyn"};
r = Contains(nameList, "Katie");
```

The example returns a 1 because “Katie” is the first item in the list.

Contains Item(*x*, <*item* | {*list*} | *pattern*>, <*delimiter*>)

Description

Identifies multiple responses by searching for the specified item, list, pattern, or delimiter. The function can be used on columns with the Multiple Response modeling type or column property.

Returns

Returns a Boolean that indicates whether the word (*item*), one of a list of words (*list*), or pattern (*pattern*) matches one of the words in the text represented by *x*. Words are delimited by the characters in the optional quoted delimiter (*delimiter*) string. A comma is the default delimiter. Blanks are trimmed from the ends of each extracted word from the input text string (*x*).

Example

The following example searches for *"pots"* followed by a comma and then outputs the result.

```
x = "Franklin Garden Supply is a leading online store featuring garden decor,
   statues, pots, shovels, benches, and much more.";
b = Contains Item( x, "pots", "," );
If( b,
   Write( "The specified items were found."  ), Write( "No match." )
);
   The specified items were found.
```

Ends With(*string*, *substring*)

Description

Determines whether the quoted `substring` appears at the end of `string`.

Returns

1 if `quoted string` ends with a quoted `substring`, otherwise 0.

Required Arguments

`string` A quoted string or a quoted string variable. Can also be a list.

`substring` A quoted string or a quoted string variable. Can also be a list.

Equivalent Expression

`Right(`*`string`*`, Length(`*`substring`*`)) == substring`

Hex(*value*, <*integer*|encoding="*enc*"|Base(*number*)|Pad To(*number*)>)

See "Char To Hex(value, <integer|encoding="enc">)".

Hex To BLOB(*string*)

Description

Converts the quoted hexadecimal *`string`* (including whitespace characters) to a blob (binary large object).

Example

```
Hex To BLOB( "4A4D50" );
   Char To BLOB("JMP", "ascii~hex")
```

Hex To Char(*string*, <*encoding*>)

Description

Converts the quoted hexadecimal *`string`* to its character equivalent.

Example

Hex To Char("30") results in "0".

Notes

The default quoted encoding for character string is "utf-8". "utf-16le", "utf-16be", "us-ascii", "iso-8859-1", "ascii~hex", "shift-jis", and "euc-jp" are also supported.

`Hex To Number(string, <Base(number)>)`

Description

Returns the number corresponding to the hexadecimal (or other base number system) text.

Required Argument

string A quoted hexadecimal string.

Optional Argument

Base(*number*) An integer between 2 and 36 inclusive. If base is specified, the text is treated as a quoted string representing the number in that base.

Example

```
Hex To Number( "80" );
  128
```

Notes

- 16-digit hexadecimal numbers are converted as IEEE 754 64-bit floating point numbers. Otherwise, the input is treated as a hexadecimal integer.
- Whitespace between bytes (or pairs of digits) and in the middle of bytes is permitted (for example, `FF 1919` and `F F1919`).

`Insert`

See "Insert(source, item, <position>)".

`Insert Into`

See "Insert Into(source, item, <position>)".

`Item(n|[first last], string, <delimiter>, <Unmatched(result string)>, <Include Boundary Delimiters(Boolean)>)`

Description

Returns the *n*th item or the span from the first to last item of the quoted *string* according to the quoted string *delimiters* given. If you include a third argument, any and all characters in that argument are taken to be delimiters.

Required Arguments

n The position of the word being extracted.

[*first last*] A matrix that defines the beginning and end word range to return.

string The quoted string that is evaluated.

Optional Arguments

delimiter The character used as a boundary. If *delimiter* is absent, an ASCII space is used. If *delimiter* is the quoted empty string, each character is treated as a separate word.

Unmatched(*result string*) The quoted string to print if no match is found.

Include Boundary Delimiters(*Boolean*) Determines how delimiters on the front and back of string are treated. The default value is false, which means that the delimiters on these boundaries are ignored. If the value is true, delimiters on these boundaries produce an empty element (similar to consecutive delimiters within a string).

Example

Consecutive delimiters are treated as though they have a word between them. In this example, the delimiters are a comma and a space.

```
Item( 4,"the  quick, brown fox", ", " ); // quick is preceded by two spaces
```

The expression is processed as follows:

the<delim[space]><word2><delim[space]>quick<delim[comma]><word 4><delim[space]>brown<delim[space]>fox

Because word4 is empty, this expression returns a quoted empty string.

Item() is the same as Word() except that Item() treats each delimiter character as a separate delimiter, and Word() treats several adjacent delimiters as a single delimiter.

```
Word( 4,"the  quick, brown fox", ", " ); // quick is preceded by two spaces
```

This expression is processed as follows:

the<delim[2 spaces]>quick<delim[comma + space]>brown<delim[space]>fox

It returns "fox".

Left(*string*, *n*, <*filler*>)

Left({*list*}, *n*, <*filler*>)

Description

Returns a truncated or padded version of the original quoted *string* or *list*. The result contains the left *n* characters or list items, padded with any *filler* on the right if the length of *string* is less than *n*.

Length(*string*)

Description

Returns the length of the given quoted string (in characters), list (in items), associative array (in number of keys), BLOB (in bytes), matrix (in elements), namespace (in number of functions and variables), or class (in number of methods, functions, and variables).

Lowercase(*string*)

Description

Converts any uppercase character found in quoted *string* to the equivalent lowercase character.

Matrix to BLOB(*matrix*, *type*, *bytesEach*, *endian(value)*)

Description

Makes a BLOB from a matrix by converting the matrix elements to 1-byte, 2-byte, or 4-byte signed or unsigned integers or 4-byte or 8-byte floating point numbers.

Required Arguments

matrix The matrix.

type The quoted type of BLOB: `int`, `uint`, or `float`.

bytesEach The number of bytes in each int, uint, or float. Integers can be 1, 2, or 4 bytes each, and floats can be 4 or 8 bytes each.

endian(value) The quoted endian-ness of your system: `"Big"` (the first byte is most significant), `"Little"` (the first byte is the least significant), or `"Native"` (the machine's native format).

Munger(*string*, *start position*, *find|length*, *<replacement string>*)

Description

Computes new quoted character strings from the quoted *string* by inserting or deleting characters. It can also produce substrings, calculate indexes, and perform other tasks depending on how you specify its arguments.

Required Arguments

start position A numeric expression that specifies the starting position to search in the quoted string. If the *start position* is greater than the position of the first instance of the find argument, the first instance is disregarded. If the *start position* is greater than the search string's length, `Munger()` uses the string's length as the *start position*.

find|length Specifies the string or number of characters to find.

Optional Argument

replacement string The quoted replacement string. If *replacement string* is omitted, a substring between start position, and position and length, is returned.

Num(*string*)

Description

Converts a quoted character string into a number.

`Regex(source, pattern, (<replacement string>, <format, "GLOBALREPLACE", "IGNORECASE">>)`

Description

Searches for the quoted *pattern* within the quoted *source* string.

Returns

The matched text as a quoted string or numeric missing if there was no match.

Required Arguments

source A quoted string.

pattern A quoted regular expression.

replacement string The replacement string.

Optional Arguments

format A backreference to the capturing group. The default is \0, which is the entire matched quoted string. \n returns the n^{th} match.

"IGNORECASE" The search is case sensitive, unless you specify "IGNORECASE".

"GLOBALREPLACE" Applies the regular expression to the quoted *source* string repeatedly until all matches are found.

`Remove`

See "Remove(source, position, <n>)".

`Remove From`

See "Remove From(source, position, <n>)".

`Repeat(source, a)`

`Repeat(matrix, a, b)`

Description

Returns a copy of *source* concatenated with itself *a* times. Or returns a matrix composed of *a* row repeats and *b* column repeats. The *source* can be text, a matrix, or a list.

`Reverse`

See "Reverse(source)".

`Reverse Into`

See "Reverse Into(source)".

Right(*string*, *n*, <*filler*>)

Right({*list*}, *n*, <*filler*>)

Description

Returns a truncated or padded version of the original quoted *string* or *list*. The result contains the right *n* characters or list items, padded with any optional *filler* on the left if the length of *string* is less than *n*.

Shift

See "Shift(source, <n>)".

Shift Into

See "Shift Into(source, <n>)".

Starts With(*string*, *substring*)

Description

Determines whether the quoted `substring` appears at the start of the quoted *string*.

Returns

1 if *string* starts with *substring*, otherwise 0.

Arguments

string A quoted string or a reference to one. Can also be a list.

substring A quoted string or a reference to one. Can also be a list.

Equivalent Expression

`Left(string, Length("substring")) = = "substring"`

Substitute

See "Substitute(string, "substring", "replacementString", ...)".

Substitute Into

See "Substitute Into(string, substring, replacementString, ...)".

Substr(*string*, *start*, *length*)

Description

Extracts the characters that are the portion of the first argument beginning at the position given by the second argument and ending based on the number of characters specified in the third argument. The first argument can be a character column or value, or an

expression evaluating to same. The starting argument and the length argument can be numbers or expressions that evaluate to numbers.

Example

This example extracts the first name:

```
Substr( "Katie Layman", 1, 5 );
```

The function starts at position 1, reads through position 5, and ignores the remaining characters, which yields "Katie."

`Text Score(`*`text column`*`, `*`text-to-number`*`, <`*`weighting`*`>, <{`*`support vectors`*`}>, <`*`text explorer setup`*`>)`

Description

Used to create scoring formulas in Text Explorer. Not supported for use with the Stem for Combining option.

Returns

Returns a vector of scores.

Required Arguments

`text column` The data table column.

`text-to-number` An associative array that maps lowercase words to numbers.

Optional Arguments

`weighting` The quoted `"Count"`, `"Binary"`, `"Ternary"`, `"LogCount"`, `"LCA"`, or an array of inverse document frequency weights for TFLogIDF. `"Count"` is the default value. The default value is `"Count"`.

`support vectors` A list of vectors that are used in the text scoring. The number and length of the vectors depends on the *`weighting`* argument.

`text explorer setup` An expression that contains a block of Text Explorer setup information.

`Titlecase(`*`string`*`)`

Description

Converts the quoted *`string`* to title case; each word in the string has an initial uppercase character and subsequent lowercase characters.

Returns

A quoted string.

Argument

`string` A quoted string.

Example

The following function capitalizes the name:

```
Titlecase( "veronica layman ")
```

```
"Veronica Layman"
```

`Trim(string,<"Left"|"Right"|"Both">)`

`Trim Whitespace(string,<"Left"|"Right"|"Both">)`

Description

Removes leading and trailing whitespace from the specified string.

Returns

A quoted string.

Required Argument

string A quoted string.

Optional Argument

`"Left"|"Right"|"Both"` A quoted string that specifies whether whitespace is removed from the left, the right, or both ends of the string. If omitted, whitespace is removed from both ends.

Example

For example, the following command returns `"John"`:

```
Trim( "  John    ", Both )
   "John"
```

`Uppercase(string)`

Description

Converts any lowercase character found in the quoted *string* to the equivalent uppercase character.

`Word(n|[first last], string, <delimiter>, <Unmatched(result string)>)`

Description

Returns the *n*th item of the string, where words are substrings separated by any number of any characters in the *delimiter* argument.

Required Arguments

n The position of the word being extracted.

`[first last]` A matrix that defines the beginning and end word range to return.

string The quoted string that is evaluated.

Optional Arguments

delimiter The character used as a boundary. If *delimiter* is absent, an ASCII space is used. If *delimiter* is the empty quoted string, each character is treated as a separate word.

`Unmatched(result string)` The quoted string to print if no match is found.

Examples

This example returns the last name:

```
Word( 2, "Katie Layman" );
   "Layman"
```

See Also

See "Item(n|[first last], string, <delimiter>, <Unmatched(result string)>, <Include Boundary Delimiters(Boolean)>)" for examples of how `Word()` differs from `Item()`.

Words

See "Words(string, <delimiter>)".

XPath Query(*xml*, *xpath_expression*)

Description

Runs an XPath expression on an XML document.

Returns

A list.

Required Arguments

xml A valid XML document.

xpath_expression A quoted XPath 1.0 expression.

Example

Suppose that you created a report of test results in JMP and exported important details to an XML document. The test results are enclosed in `<result>` tags.

The following example stores the XML document in a variable. The `XPath Query` expression parses the XML to find the text nodes inside the `<result>` tags. The results are returned in a list.

```
rpt =
"\[<?xml version="1.0" encoding="utf-8"?>
<JMP><report><title>Production Report</title>
<result>November 21st: Pass</result>
<result>November 22nd: Fail</result>
<note>Tests ran at 3:00 a.m.</note></report>
</JMP> ]\";
results = XPath Query( rpt, "//result/text()" );
   {"November 21st: Pass", "November 22nd: Fail"}
```

Character Pattern Functions

Pat Abort()

Description

Constructs a quoted pattern that immediately stops the pattern match. The matcher does not back up and retry any alternatives. Conditional assignments are *not* made. Immediate assignments that were already made are kept.

Returns

0 when a match is stopped.

Argument

none

Pat Altern(*pattern1*, <*pattern2*, ...>)

Description

Constructs a quoted pattern that matches any one of the pattern arguments.

Returns

A quoted pattern.

Argument

One or more patterns.

Pat Any(*string*)

Description

Constructs a quoted pattern that matches a single character in the argument.

Returns

A quoted pattern.

Argument

pattern A quoted string.

Pat Arb()

Description

Constructs a quoted pattern that matches an arbitrary quoted string. It initially matches the null quoted string. It then matches one additional character each time the pattern matcher backs into it.

Returns

A quoted pattern.

Argument

none

Example

```
p = "the beginning" + Pat Arb() >? stuffInTheMiddle + "the end";
Pat Match( "in the beginning of the story, and not near the end, there are
   three bears", p );
Show( stuffInTheMiddle );
   stuffInTheMiddle = " of the story, and not near "
```

Pat Arb No(*pattern*)

Description

Constructs a quoted pattern that matches zero or more copies of *pattern*.

Returns

A quoted pattern.

Argument

pattern A quoted pattern to match against.

Example

```
adjectives = "large" | "medium" | "small" | "warm" | "cold" | "hot" | "sweet";
rc = Pat Match( "I would like a medium hot, sweet tea please",
                Pat Arbno( adjectives | Pat Any(", ") ) >> adj +
                ("tea" | "coffee" | "milk") );
Show( rc, adj );
   rc = 1;
   adj = " medium hot, sweet ";
```

Pat At(*varName*)

Description

Constructs a quoted pattern that matches the null quoted string and stores the current position in the source string into the specified JSL variable (*varName*). The assignment is immediate, and the variable can be used with `expr()` to affect the remainder of the match.

Returns

A quoted pattern.

Argument

varName The name of a variable to store the result in.

Example

```
p = ":" + Pat At( listStart ) + Expr(
   If( listStart == 1,
      Pat Immediate( Pat Len( 3 ), early ),
      Pat Immediate( Pat Len( 2 ), late )
   )
);
```

```
early = "";
late = "";
Pat Match( ":123456789", p );
Show( early, late );
early = "";
late = "";
Pat Match( "   :123456789", p );
Show( early, late );
```

First this is produced:

```
early = "123"
late = ""
```

and later this:

```
early = ""
late = "12"
```

Pat Break(*string*)

Description

Constructs a quoted pattern that matches zero or more characters that are not in its argument; it stops or breaks on a character in its argument. It fails if a character in its argument is not found (in particular, it fails to match if it finds the end of the quoted source string without finding a break character).

Returns

A quoted pattern.

Argument

string A quoted string.

Pat Concat(*pattern1*, *pattern2* <*pattern3*, ...>)

Pattern1 + *Pattern2* + ...

Description

Constructs a quoted pattern that matches each pattern argument in turn.

Returns

A quoted pattern.

Argument

Two or more quoted patterns.

Pat Conditional(*pattern*, *varName*)

Description

Saves the result of the quoted pattern match, if it succeeds, to a variable named as the second argument (*varName*) after the match is finished.

Returns

A quoted pattern.

Arguments

pattern A quoted pattern to match against.

varName The name of a variable to store the result in.

Example

```
type = "undefined";
rc = Pat Match(
   "green apples",
   Pat Conditional( "red" | "green", type ) + " apples"
);
Show( rc, type );
   rc = 1;
   type = "green";
```

Pat Fail()

Description

Constructs a quoted pattern that fails whenever the matcher attempts to move forward through it. The matcher backs up and tries different alternatives. If and when there are no alternatives left, the match fails and `Pat Match` returns 0.

Returns

0 when a match fails.

Argument

none

Pat Fence()

Description

Constructs a pattern that succeeds and matches the quoted null string when the matcher moves forward through it, but fails when the matcher tries to back up through it. It is a one-way trap door that can be used to optimize some matches.

Returns

1 when the match succeeds, 0 otherwise.

Argument

none

Pat Immediate(*pattern*, *varName*)

Description

Saves the result of the pattern match to a variable named as the second argument (*varName*) immediately.

Returns

A quoted pattern.

Arguments

pattern A quoted pattern to match against.

varName The name of a variable to store the result in.

Example

```
type = "undefined";
rc = Pat Match(
   "green apples",
   ("red" | "green") >> type + " pears"
);
Show( rc, type );
   rc = 0
   type = "green"
```

Even though the match failed, the immediate assignment was made.

Pat Len(*int*)

Description

Constructs a quoted pattern that matches *n* characters.

Returns

A quoted pattern.

Argument

int An integer that specifies the number of characters.

Pat Look Ahead(*pattern*, *Boolean*)

Description

A zero-width pattern match after the current position.

Arguments

pattern The quoted pattern.

Boolean 0 (the default) indicates a match. 1 indicates a negative match or non-match.

Pat Look Behind(*pattern*, *Boolean*)

Description

A zero-width quoted pattern match before the current position.

Arguments

pattern The quoted pattern.

Boolean 0 (the default) indicates a match. 1 indicates a negative match or non-match.

Pat Match(*source text*, *pattern*, <*replacement string*>, <"NULL">, <"ANCHOR">, <"MATCHCASE">, <"FULLSCAN">)

Description

Pat Match executes the quoted *pattern* against the *source text*. The pattern must be constructed first, either inline or by assigning it to a JSL variable elsewhere.

Returns

1 if the pattern is found, 0 otherwise.

Required Arguments

source text A quoted string or quoted string variable that contains the text to be searched.

pattern A quoted pattern or pattern variable that contains the text to be searched for.

Optional Arguments

replacement string A quoted string that defines text to replace the pattern in the source text.

"NULL" A placeholder for the third argument if ANCHOR, MATCHCASE, or FULLSCAN are necessary *and* there is no replacement text.

"ANCHOR" Starts the pattern match at the beginning of the quoted string. The following match fails because the pattern, "cream", is not found at the beginning of the string:

```
Pat Match( "coffee with cream and sugar", "cream", NULL, ANCHOR );
```

"MATCHCASE" Optional command to consider capitalization in the match. By default, Pat Match() is case insensitive.

"FULLSCAN" Optional command to force Pat Match to try all alternatives, which uses more memory as the match expands. By default, Pat Match() does not use FULLSCAN, and makes some assumptions that allow the recursion to stop and the match to succeed.

Pat Not Any(*string*)

Description

Constructs a pattern that matches a single character that is not in the argument.

Returns

A quoted pattern.

Argument

string A quoted string.

Pat Pos(*int*)

Description

Constructs patterns that match the quoted null string if the current position is *int* from the left end of the string, and fail otherwise.

Returns

A quoted pattern.

Argument

int An integer that specifies the position in a quoted string.

Pat R Pos(*int*)

Description

Constructs patterns that match the quoted null string if the current position is *int* from the right end of the string, and fails otherwise.

Returns

A quoted pattern.

Argument

int An integer that specifies the position in a quoted string.

Pat R Tab(*int*)

Description

Constructs a quoted pattern that matches up to position *n* from the end of the quoted string. It can match 0 or more characters. It fails if it would have to move backwards or beyond the end of the string.

Returns

A quoted pattern.

Argument

int An integer that specifies a position in a quoted string.

Pat Regex(*string*)

Description

Constructs a quoted pattern that matches the regular expression in the quoted *string* argument.

Returns

A quoted pattern.

Argument

string A quoted string.

Pat Rem()

Description

Constructs a quoted pattern that matches the remainder of the quoted string. It is equivalent to `Pat R Tab(0)`.

Returns

A quoted pattern.

Argument

none

Pat Repeat(*pattern*, *minimum*, *maximum*, <"GREEDY"|"RELUCTANT">)

Description

Matches the quoted *pattern* between *minimum* and *maximum* times.

Returns

A quoted pattern.

Required Arguments

pattern A pattern to match against.

minimum An integer that must be smaller than maximum.

maximum An integer that must be greater than minimum.

Optional Argument

"GREEDY"|"RELUCTANT" If GREEDY is specified, it tries the maximum first and works back to the minimum. If RELUCTANT is specified, it tries the minimum first and works up to the maximum.

Notes

- `Pat Arbno(p)` is the same as `Pat Repeat(p, 0, infinity, RELUCTANT)`
- `Pat Repeat(p)` is the same as `Pat Repeat(p, 1, infinity, GREEDY)`
- `Pat Repeat(p, n)` is the same as `Pat Repeat(p, n, infinity, GREEDY)`
- `Pat Repeat(p, n, m)` is the same as `Pat Repeat(p, n, m, GREEDY)`

Pat Span(*string*)

Description

Constructs a pattern that matches one or more (not zero) occurrences of characters in its argument. It is greedy; it always matches the longest possible quoted string. It fails rather than matching zero characters.

Returns

A quoted pattern.

Argument

string A quoted string.

Pat String(*string*)

Description

Constructs a pattern that matches its quoted *string* argument.

Returns

A quoted pattern.

Argument

string A quoted string.

Pat Succeed()

Description

Constructs a pattern that always succeeds, even when the matcher backs into it. It matches the quoted null string.

Returns

1 when the match succeeds.

Argument

none

Pat Tab(*int*)

Description

Constructs a pattern that matches forward to position *int* in the quoted source string. It can match 0 or more characters. It fails if it would have to move backwards or beyond the end of the string.

Returns

A pattern.

Argument

int An integer that specifies a position in a quoted string.

Pat Test(*expr*)

Description

Constructs a pattern that succeeds and matches the quoted null string if *expr* is not zero and fails otherwise.

Returns

A quoted pattern.

Argument

expr An expression.

Notes

Usually the argument is wrapped with `expr()` because the test needs to be made on the current value of variables set by `Pat Immediate`, `Pat Conditional`, and `Pat At`. Without `expr`, the test is based on values that were known when the pattern was constructed, which means the test always succeeds or always fails at pattern execution time, which is probably not what you want.

Example

```
nCats = 0;
whichCat = 3;
string = "catch a catnapping cat in a catsup factory";
rc = Pat Match(
   string,
   "cat" + Pat Test(
      Expr(
         nCats = nCats + 1;
         nCats == whichCat;
      )
   ),
   "dog"
);
Show( rc, string, nCats );
   rc = 1
   string = "catch a catnapping dog in a catsup factory"
   nCats = 3
```

`Regex Match(`*source*`, `*pattern*`, <`*replacement string*`>|<"MATCHCASE">, <"NULL">)`

Description

Executes the pattern match in quoted *pattern* against the quoted *source* string.

Returns

A pattern.

Required Arguments

source A quoted string.

pattern A quoted pattern.

Optional Arguments

replacement string The quoted string that specifies the text to replace the source with.

"MATCHCASE" The search is case insensitive unless you specify MATCHCASE.

"NULL" Indicates that the expression contains MATCHCASE but you don't want to specify a replacement.

Examples

```
Regex Match(
   "person=Fred id=77 friend= favorite=tea", // source
   "(\w+)=(\S*) (\w+)=(\S*) (\w+)=(\S*) (\w+)=(\S*)" // pattern
);
   {"person=Fred id=77 friend= favorite=tea", "person", "Fred", "id", "77",
      "friend", "", "favorite", "tea"}

// case-insensitive, no replacement
Regex Match( "beliEve", "([aeiou])(.*?)(\1)" );
   {"eliE", "e", "li", "E"}
// case-sensitive, no replacement
Regex Match( "beliEve", "([aeiou])(.*?)(\1)", NULL, MATCHCASE );
   {"eliEve", "e", "liEv", "e"}
```

Comment Functions

// comment

Description

Comments to end of line.

Notes

Everything after the // is ignored when running the script.

/* comment */

Description

A comment that can appear in the middle of a line of script.

Notes

Anything between the beginning tag /* and the end tag */ is ignored when running the script. This comment style can be used almost anywhere, even inside lists of arguments. If you place a comment inside a double-quoted string, the comment is treated merely as part of the string and not a comment. You cannot place comments in the middle of operators.

Examples

```
+/*comment*/=
:/*comment*/name
```

are invalid and produce errors. The first comment interrupts += and the second interrupts :name.

```
sums = {(a+b /*comment*/), /*comment*/ (c^2)}
```

is valid JSL; the comments are both ignored.

`//!`

Description

If placed on the first line of a script, this comment line causes the script to be run when opened in JMP without opening into the script editor window.

Notes

You can over-ride this comment when opening the file. Select **File > Open**. Hold the Ctrl key while you select the JSL file and click **Open**. Or right-click the file in the Home Window Recent Files list and select **Edit Script**. The script opens into a script window instead of being executed.

`/*debug step*/`

`/*debug run*/`

Description

If placed on the first line of a script, the script is opened in the debugger when it is run.

Notes

All letters must be lowercase. There must be one space between debug and step or run, and there must be no other spaces present. Only one of these lines can be used, and it must be the first line of the script; a first line that is blank followed by this comment negates the debug command.

Comparison Functions

The comparison operators (<, <=, >, >=) work for numbers, quoted strings, and matrices. For matrices, they produce a matrix of results. If you compare mixed arguments, such as strings with numbers or matrices, the result is a missing value. Comparisons involving lists are not allowed and also return missing values.

The equality operators (== and !=) work for numbers, quoted strings, matrices, and lists. For matrices, they produce a matrix of results; for lists, they produce a single result. If you *test equality* of mixed results (for example. strings with numbers or matrices) the result is 0 or unequal.

Range check operators let you check whether something falls between two specified values:

```
a = 1;
Show( 1 <= a < 3 );
b = 2;
Show( 2 < b <= 3 );
   1 <= a < 3 = 1;
   2 < b <= 3 = 0;
```

Expressions with comparison operators are evaluated all at once, not in sequence

All the comparison operators are *eliding operators*. That means JMP treats arguments joined by comparison operators as one big clause, as opposed to the way most expressions are evaluated one operator at a time. Evaluating as a single clause produces different results than the more usual method of evaluating in pieces. For example, the following two statements are different:

```
12 < a < 13;
(12 < a) < 13;
```

The first statement checks whether *a* is between 12 and 13, because all three arguments and both operators are read and evaluated together. The second statement uses parentheses to regroup the operations explicitly to evaluate from left to right, which would be the normal way to evaluate most expressions. Thus it first checks whether 12 is less than *a*, returning 1 if true or 0 if false. Then it checks whether the result is less than 13, which is always true because 0 and 1 are both less than 13.

All the comparison operators are elided when they are used in matched pairs or in the unmatched pairs <... <= and <=... <. What this means is that if you want a comparison statement to be evaluated one comparison operator at a time, you should use parentheses ( ) to control the order of operations explicitly.

`Equal(a, b, ...)`

`a==b==...`

Description

Compares all the listed values and tests if they are all equal to each other.

Returns

1 (true) if all arguments evaluate to the same value.

0 (false) otherwise.

Arguments

Two or more variables, references, matrices, or numbers.

Notes

If more than two arguments are specified, a 1 is returned only if all arguments are exactly the same. This is typically used in conditional statements and to control loops.

The comparison is case-sensitive for quoted string comparisons.

`Greater(a, b, ...)`

`a>b>...`

Description

Compares all the list values and tests if, in each pair, the left value is greater than the right.

Returns

1 (true) if *a* evaluates strictly greater than *b* (and *b* evaluates strictly greater than *c*, and so on).

0 (false) otherwise.

Arguments

Two or more variables, references, matrices, or numbers.

Notes

If more than two arguments are specified, a 1 is returned only if each argument is greater than the one that follows it. This is typically used in conditional statements and to control loops.

`Greater`, `Less`, `Greater Or Equal`, and `Less Or Equal` can also be strung together (as in `0 < x <= 5`). If you do not group with parentheses, JMP evaluates each pair left to right. You can also use parentheses to explicitly tell JMP how to evaluate the expression.

`Greater or Equal(`*a*`, `*b*`, ...)`

a>=*b*>=...

Description

Compares all the list values and tests if, in each pair, the left value is greater than or equal to the right.

Returns

1 (true) if *a* evaluates strictly greater than or equal to *b* (and *b* evaluates strictly greater than or equal to *c*, and so on).

0 (false) otherwise.

Arguments

Two or more variables, references, matrices, or numbers.

Notes

If more than two arguments are specified, a 1 is returned only if each argument is greater than or equal to the one that follows it. This is typically used in conditional statements and to control loops.

`Greater`, `Less`, `Greater Or Equal`, and `Less Or Equal` can also be strung together (as in `0 < x <= 5`). If you do not group with parentheses, JMP evaluates each pair left to right. You can also use parentheses to explicitly tell JMP how to evaluate the expression.

`Is Missing(`*expr*`)`

Description

Returns 1 if the expression yields a missing value and 0 otherwise.

Less(*a*, *b*, ...)

a<*b*<...

Description

Compares all the list values and tests if, in each pair, the left value is less than the right.

Returns

1 (true) if *a* evaluates strictly less than *b* (and *b* evaluates strictly less than *c*, and so on).

0 (false) otherwise.

Arguments

Two or more variables, references, matrices, or numbers.

Notes

If more than two arguments are specified, a 1 is returned only if each argument is less than the one that follows it. This is typically used in conditional statements and to control loops.

`Greater`, `Less`, `Greater Or Equal`, and `Less Or Equal` can also be strung together (as in `0 < x <= 5`). If you do not group with parentheses, JMP evaluates each pair left to right. You can also use parentheses to explicitly tell JMP how to evaluate the expression.

Less Less Equal(*a*, *b*, *c*, ...)

a<*b*<=*c*<=...

Description

Range check, exclusive below and inclusive above.

Returns

1 (true) if *b* is greater than *a* and less than or equal to *c*.

0 (false) otherwise.

Arguments

a, *b*, *c* Variables, references, matrices, or numbers.

Notes

Returns 1 when two conditions are met: the first argument is less than the second argument, and each remaining argument is less than or equal to its argument on the right. This is typically used in conditional statements and to control loops.

Less or Equal(*a*, *b*, ...)

a<=*b*<=...

Description

Compares all the list values and tests if, in each pair, the left value is less than or equal to the right.

Returns

1 (true) if *a* evaluates strictly less than or equal to *b* (and *b* evaluates strictly less than or equal to *c*, and so on).

0 (false) otherwise.

Arguments

Two or more variables, references, matrices, or numbers.

Notes

If more than two arguments are specified, a 1 is returned only if each argument is less than or equal to the one that follows it. This is typically used in conditional statements and to control loops.

`Greater`, `Less`, `Greater Or Equal`, and `Less Or Equal` can also be strung together (as in `0 < x <= 5`). If you do not group with parentheses, JMP evaluates each pair left to right. You can also use parentheses to explicitly tell JMP how to evaluate the expression.

`Less Equal Less(a, b, c, ...)`

`a<=b<c<...`

Description

A range check, inclusive below and exclusive above.

Returns

1 (true) if *b* is greater than or equal to *a* and less than *c*.

0 (false) otherwise.

Arguments

a, *b*, *c* Variables, references, matrices, or numbers.

Notes

Returns 1 when two conditions are met: the first argument is less than or equal to the second argument, and each remaining argument is less than its argument on the right. This is typically used in conditional statements and to control loops.

`Not Equal(a, b)`

`a!=b`

Description

Compares *a* and *b* and tests if they are equal.

Returns

0 (false) if *a* and *b* evaluate to the same value.

1 (true) otherwise.

Argument

a, b Any variable or number.

Notes

Mostly used for conditional statements and loop control.

Conditional and Logical Functions

And(*a, b*)

a&*b*

Description

The logical And.

Returns

1 (true) if both *a* and *b* are true.

0 (false) if either *a* or *b* is false or if both *a* and *b* are false.

Missing if either *a* or *b* is a missing value or if both *a* and *b* are missing values.

Arguments

Two or more variables or expressions.

Notes

More than two arguments can be strung together. a&b returns 1 (true) only if all arguments evaluate to true.

AndMZ(*a, b*)

a:&*b*

Description

Returns the logical AND of all arguments. Missing values are treated as zeroes.

Returns

1 (true) if both *a* and *b* are true.

0 (false) if either *a* or *b* is false or if both *a* and *b* are false.

0 (false) if either *a* or *b* is a missing value or if both *a* and *b* are missing values.

Arguments

Two or more variables or expressions.

Notes

More than two arguments can be strung together. *a*:&*b* returns 1 (true) only if all arguments evaluate to true.

Break()

Description

Stops execution of a loop completely and continues to the statement following the loop.

Notes

`Break()` works with `For` and `While` loops and also with `For Each Row`.

Choose(*expr*, *r1*, *r2*, *r3*, ..., *resultElse*)

Description

Evaluates *expr*. If the value of *expr* is 1, *r1* is returned; if 2, the value of *r2* is returned, and so on. If no matches are found, the last argument (*resultElse*) is returned.

Returns

The value whose index in the list of arguments matches *expr*, or the value of the last argument.

Arguments

expr An expression or a value.

r1, *r2*, *r3*, ... An expression or a value.

resultElse The argument that is returned when no matches are found.

Continue()

Description

Ends the current iteration of a loop and begins the loop at the next iteration.

Notes

`Continue()` works with `For` and `While` loops, and also with `For Each Row`.

Filter Each(*names*, *container*, *locals*, *body*)

Description

Iterates over a container, which can be a list, an associative array, or a matrix, and returns a subset of the values in the container based on the evaluation of a Boolean expression at each iteration. The value, key, or element, as well as the index number, are all available at each iteration.

- For associative array containers, the key and value can be accessed using a two-item list.
- For matrix containers, a linear index is provided by default, but a two-item list can be used to access the row and column indices.

These symbols are provided within the body of the loop only. A list of local variables can also be provided. If local variables are defined, they are initialized after the first iteration symbols are set.

Returns

A subset from the original container. The return object has the same type as the original container.

Arguments

names Specification of loop control variable names, specified as a list. The form of the list is determined by the type of container. All of the names are optional, so you need to specify them only if you need to refer to them in the *locals* or *body* arguments. If you do not specify any names, the first argument should be an empty list, which can be specified as {} or List().

For a *list* container, the names list contains a name for each value in the list and a name for the index of each value in the list.

For an *associative array* container, the names list contains a two-item list of names and a name for the index of each item in the associative array.

For a *matrix* container, the names list contains a name for each element in the matrix and a second argument for the index of each element in the matrix. The second argument can be specified as a single name or as a two-item list of names that represent symbols for the row and column indices.

When specifying multiple containers using the Across() keyword, the names list contains a list of names that refer to values in each container. The number of names in the first item in the names list must match the number of containers specified in the Across() keyword.

container A list, associative array, or a matrix. The container can be defined in the argument or it can be a reference to a previously defined object.

This argument can also use the Across() keyword, which enables you to use the function across multiple containers. The multiple containers can be specified as separate arguments or as items in a list. The Across() keyword has an optional Count() argument that enables you to specify how containers of different sizes are handled. The available Count() options are: "Shortest", "Longest", "Enforce Equal", and N, where N is a number. If you specify a number, the function iterates through all containers exactly N times; note that the function loops back to the start of containers that have fewer than N items.

locals A list of variables that are local to the function. This is equivalent to other Local variable initializations in JSL. The initialization of the local variables occurs after the first loop control variables have been set, but the local variables do not get initialized again after that.

body Any number of valid JSL expressions, glued together if there are more than one. The result of the JSL expressions should be a Boolean value. If the result of the expression is true, the container value at the current iteration is included in the result; otherwise, the

container value at the current iteration is not included in the result. You can use the `Continue()` function as an equivalent way to return false and skip to the next iteration. You can also use `Break()` function to stop iteration through the loop and proceed to the next expression that follows the loop. See "Break and Continue".

Example

```
values = Filter Each( {x}, {10, 20, 30}, x > 15 );
Show( values );
  values = {20, 30};
```

For(*init*, *while*, *increment*, *body*)

Description

Repeats the statement(s) in the *body* as long as the *while* condition is true. *init* and *increment* control iterations.

Returns

Null.

Arguments

init Initialization of loop control counter.

while Condition for loop to continue or end. As long as the conditional statement while is true, the loop is iterated one more time. As soon as while is false, the loop is exited.

increment Increments (or decrements) the loop counter after while is evaluated every *time* the loop is executed.

body Any number of valid JSL expressions, glued together if there are more than one.

Example

```
mysum = 0; myprod = 1;
For( i = 1, i <= 10, i++, mysum += i; myprod *= i; );
Show( mysum, myprod );
  mysum = 55;
  myprod = 3628800;
```

For Each(*names*, *container*, *locals*, *body*)

Description

Iterates over a container, which can be a list, an associative array, or a matrix, and provides the value, key, or element at each iteration. The index number is also available at each iteration. For associative array containers, the key and value can be accessed using a two-item list. For matrix containers, a linear index is provided by default, but a two-item list can be used to access the row and column indices. These symbols are provided within the body of the loop only. A list of local variables can also be provided. If local variables are defined, they are initialized after the first iteration symbols are set.

Arguments

names Specification of loop control variable names, specified as a list. The form of the list is determined by the type of container. All of the names are optional, so you need to specify them only if you need to refer to them in the *locals* or *body* arguments. If you do not specify any names, the first argument should be an empty list, which can be specified as {} or List().

For a *list* container, the names list contains a name for each value in the list and a name for the index of each value in the list.

For an *associative array* container, the names list contains a two-item list of names and a name for the index of each item in the associative array.

For a *matrix* container, the names list contains a name for each element in the matrix and a second argument for the index of each element in the matrix. The second argument can be specified as a single name or as a two-item list of names that represent symbols for the row and column indices.

When specifying multiple containers using the Across() keyword, the names list contains a list of names that refer to values in each container. The number of names in the first item in the names list must match the number of containers specified in the Across() keyword.

container A list, associative array, or a matrix. The container can be defined in the argument or it can be a reference to a previously defined object.

This argument can also use the Across() keyword, which enables you to use the function across multiple containers. The multiple containers can be specified as separate arguments or as items in a list. The Across() keyword has an optional Count() argument that enables you to specify how containers of different sizes are handled. The available Count() options are: "Shortest", "Longest", "Enforce Equal", and N, where N is a number. If you specify a number, the function iterates through all containers exactly N times; note that the function loops back to the start of containers that have fewer than N items.

locals A list of variables that are local to the function. This is equivalent to other Local variable initializations in JSL. The initialization of the local variables occurs after the first loop control variables have been set, but the local variables do not get initialized again after that.

body Any number of valid JSL expressions, glued together if there are more than one. You can use the Continue() function to skip to the next iteration. You can also use Break() function to stop iteration through the loop and proceed to the next expression that follows the loop. See "Break and Continue".

Example

```
For Each( {value, index}, {10, 20, 30}, Show( value, index ) );
  value = 10;
  index = 1;
  value = 20;
  index = 2;
```

```
value = 30;
index = 3;
```

For Each Row(<*dt*>, *script*)

Description

Repeats the *script* on each row of the data table.

Returns

Null.

Required Argument

script Any valid JSL expressions.

Optional Argument

dt Positional argument that is a reference to a data table. If this argument is not in the form of an assignment, then it is considered a data table expression.

Example

The following example creates data table references and then iterates over each row in Big Class.jmp. If the value of age in a row is greater than 15, the age is printed to the log.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( dt, If( :age > 15, Show( :age ) ) );
```

If(*condition1*, *result1*, <*condition2*, *result2*>, ..., <*elseResult*>)

Description

Evaluates the first of each pair of arguments and returns the evaluation of the *result* expression associated with the first *condition* argument that evaluates to a nonzero result. The *condition* arguments are evaluated in order. If all of the *condition* arguments evaluate to zero, the optional *elseResult* is evaluated and the result is returned. If no *elseResult* is specified, and none of the conditions are true, a missing value is returned. If all of the *condition* arguments evaluate to missing, a missing value is returned.

IfMax(*expr1*, *result1*, *expr2*, *result2*, ... <*all missing results*>)

Description

Evaluates the first of each pair of arguments and returns the evaluation of the result expression (the second of each pair) associated with the maximum of the expressions. If more than one expression is the maximum, the first maximum is returned. If all expressions are missing and a final result is not specified, missing is returned. If all expressions are missing and a final result is specified, that final result is returned. The test expressions must evaluate to numeric values, but the result expressions can be anything.

Returns

The result expression associated with the maximum of the expressions

IfMin(*expr1*, *result1*, *expr2*, *result2*, ... <*all missing results*>)

Description

Evaluates the first of each pair of arguments and returns the evaluation of the result expression (the second of each pair) associated with the minimum of the expressions. If more than one expression is the minimum, the first minimum is returned. If all expressions are missing and a final result is not specified, missing is returned. If all expressions are missing and a final result is specified, that final result is returned. The test expressions must evaluate to numeric values, but the result expressions can be anything.

Returns

The result expression associated with the minimum of the expressions

IfMZ(*condition1*, *result1*, <*condition2*, *result2*>, ..., <*elseResult*>)

Description

Evaluates the first of each pair of arguments and returns the evaluation of the *result* expression associated with the *condition1* argument that evaluates to a nonzero result. The *condition* arguments are evaluated in order. If all of the *condition* arguments evaluate to zero or missing, the optional *elseResult* is evaluated and the result is returned. If no *elseResult* is specified, and none of the conditions are true, a missing value is returned.

Notes

The test arguments are evaluated in order until the first nonzero result. If all test results return zero or missing, the *elseExpr* argument is evaluated.

IfMZ() is equivalent to If() where missing values for evaluated *condition* arguments are treated as zero.

Interpolate(*x*|*xmatrix*|*xlist*, *x1*, *y1*, *x2*, *y2*, ...)

Interpolate(*x*|*xmatrix*|*xlist*, *xmatrix*, *ymatrix*)

Interpolate({*x*, *y*}, *xvector*, *yvector*, *zmatrix*)

Description

Performs linear interpolation for continuous data. There are many ways to specify the function.

In the simplest cases where the first argument is a single numeric value, the *y* value corresponding to a given *x* value between two sets of points or by matrices *xmatrix* and *ymatrix*. The *x* values, (*x1*, *x2*, ...) or *xmatrix*, must be in ascending order.

If the first argument is a matrix or list of numeric values, the resulting matrix or list is a set of interpolated values.

You can perform *bilinear interpolation* using the four-argument case. Here, the first argument is a list of two points, the second and third arguments are vectors that define the grid of x and y values, and the fourth argument is a matrix of data points. The function then finds the interpolated *z* value within the appropriate quadrant of the *zmatrix*.

Returns

The interpolated value or values. In the three-argument cases, the return object type matches the type of the first argument. In the four-argument case, the return object is a number.

`Is Associative Array(name)`

Description

Returns 1 if the evaluated argument is an associative array, or 0 otherwise.

`Is Empty(global)`

`Is Empty(dt)`

`Is Empty(col)`

Description

Returns 1 if the *global* variable, data table, or data column is undefined or holds the `Empty()` value, or 0 otherwise.

`Is Expr(x)`

Description

Returns 1 if the evaluated argument is an expression, or 0 otherwise.

`Is List`

See "Is List(x)".

`Is Name(x)`

Description

Returns 1 if the evaluated argument is a name, or 0 otherwise.

`Is Namespace(namespace)`

Description

Returns 1 if the namespace argument is a namespace; returns 0 otherwise.

`Is Number(x)`

Description

Returns 1 if the evaluated argument is a number or missing numeric value, or 0 otherwise.

`Is Scriptable(x)`

Description

Returns 1 if the evaluated argument is a scriptable object, or 0 otherwise.

`Is String(x)`

Description

Returns 1 if the evaluated argument is a quoted string, or 0 otherwise.

`Match(x, value1, result1, value2, result2, ..., resultElse)`

Description

If *a* is equal to *value1*, then *result1* is returned. If *a* is equal to *value2*, *result2* is returned, and so on.

Notes

The `Match()` function explicitly checks to see if the compare expression *x* is missing and if the value of *value1* is missing, then it returns the value of `result1`; otherwise it continues to compare the expression *x* to each *valueN* value in each *valueN/resultN* pair, ignoring any missing values. If the expression *x* is equal to any of the *valueN* value, then the corresponding *resultN* value is returned. If no matching *valueN* value is found, then the *resultElse* value is returned.

`MatchMZ(x, value1, expr1, value2, expr2, ..., exprElse)`

Description

Evaluates and returns the *exprN* argument that equals *x* or evaluates and returns the *exprElse* argument if no value equals *x*.

Notes

The `MatchMZ()` function works the same as the `Match()` function except that missing values are treated as 0.

`Not(a)`

`!a`

Description

The logical Not.

Returns

0 (false) if *a*>0.

1 (true) if *a*<=0.

Missing value if *a* is missing.

Argument

a Any variable or number. The variable must have a numeric or matrix value.

Notes

Mostly used for conditional statements and loop control.

`Or(a, b)`

`a|b`

Description

The logical Or.

Returns

1 (true) if either of or both *a* and *b* are true.

0 (false) otherwise.

Missing if either are missing.

Arguments

a, *b* Any variable or number.

Notes

Mostly used for conditional statements and loop control.

`OrMZ(a, b)`

`a:|b`

Description

Returns the logical OR of all arguments with missing values treated as zeroes: 1 if any arguments are nonzero and 0 otherwise.

Returns

1 (true) if either of or both *a* and *b* are true.

0 (false) otherwise.

Arguments

a, *b* Any variable or number.

Notes

– Mostly used for conditional statements and loop control.

– `Or()` returns missing if any evaluated argument is missing. `OrMZ()` returns 0 if any evaluated argument is missing.

`Return(<expr1>, <expr2>, ..., <exprN>)`

Description

Returns an expression value from a user-defined function.

Example

This example returns the evaluation of both expressions in the `Return()` function as a list. The `Return()` function can have more than one argument. If only one is present, then the value of the expression is returned. If more than one is present, then the values of all the expressions is returned in a list.

```
f = Function( {a, b},
  Return( a - b, a + b )
);
{lo, hi} = f( 10, 1 );
Show( lo, hi );
Show( f( 7, 15 ) );
   lo = 9;
   hi = 11;
   f(7, 15) = {-8, 22};
```

Notes

`Return()` not enclosed by a function, method, or recursive function call causes an error.

`Step(x0, x1, y1, x2, y2, ...)`

`Step(x0, [x1, x2, ...], [y1, y2, ...])`

Description

Returns the *y* argument corresponding to the largest *x* argument that is less than or equal to *x0*. The *x* points must be specified in ascending order.

`Stop()`

Description

Immediately stops a script that is running.

`Transform Each(names, container, <Output(type)>, <locals>, body)`

Description

Iterates over a container, which can be a list, an associative array, or a matrix, and updates each of the values in the container based on the evaluation of a JSL expression at each iteration. The value, key, or element, as well as the index number, are all available at each iteration. For associative array containers, the key and value can be accessed using a

two-item list. For matrix containers, a linear index is provided by default, but a two-item list can be used to access the row and column indices. These symbols are provided within the body of the loop only. A list of local variables can also be provided. If local variables are defined, they are initialized after the first iteration symbols are set.

Returns

An updated version of the original container. The return object has the same type as the original container, unless the `Output()` keyword is used to designate a different type for the returned container.

Arguments

`names` Specification of loop control variable names, specified as a list. The form of the list is determined by the type of container. All of the names are optional, so you need to specify them only if you need to refer to them in the *`locals`* or *`body`* arguments. If you do not specify any names, the first argument should be an empty list, which can be specified as `{}` or `List()`.

For a *list* container, the names list contains a name for each value in the list and a name for the index of each value in the list.

For an *associative array* container, the names list contains a two-item list of names and a name for the index of each item in the associative array.

For a *matrix* container, the names list contains a name for each element in the matrix and a second argument for the index of each element in the matrix. The second argument can be specified as a single name or as a two-item list of names that represent symbols for the row and column indices.

When specifying multiple containers using the `Across()` keyword, the names list contains a list of names that refer to values in each container. The number of names in the first item in the names list must match the number of containers specified in the `Across()` keyword.

`container` A list, associative array, or a matrix. The container can be defined in the argument or it can be a reference to a previously defined object.

This argument can also use the `Across()` keyword, which enables you to use the function across multiple containers. The multiple containers can be specified as separate arguments or as items in a list. The `Across()` keyword has an optional `Count()` argument that enables you to specify how containers of different sizes are handled. The available `Count()` options are: `"Shortest"`, `"Longest"`, `"Enforce Equal"`, and `N`, where `N` is a number. If you specify a number, the function iterates through all containers exactly N times; note that the function loops back to the start of containers that have fewer than N items.

`Output(type)` Specifies a type for the output. This can be `"List"`, `"Matrix"`, or `"Associative Array"`. By default, the output type matches the type of the input container.

`locals` A list of variables that are local to the function. This is equivalent to other Local variable initializations in JSL. The initialization of the local variables occurs after the

first loop control variables have been set, but the local variables do not get initialized again after that.

body Any number of valid JSL expressions, glued together if there are more than one. The result of the JSL expressions at each iteration is used in the output container. You can use the `Continue()` function to return no value for an iteration and skip to the next iteration. You can also use `Break()` function to stop iteration through the loop and proceed to the next expression that follows the loop. See "Break and Continue".

Example

```
values = Transform Each( {x}, {10, 20}, x + 10 );
Show( values );
   values = {20, 30};
```

While(*expr*, *body*)

Description

Repeatedly tests the *expr* condition and executes the *body* until the *expr* condition is no longer true.

Zero Or Missing(*expr*)

Description

Returns 1 if *expr* yields a missing value or zero, 0 otherwise.

Constant Functions

JMP provides functions for two useful constant functions.

Note: These functions do not take an argument, but the parentheses are required.

e()

Description

Returns the constant *e*, which is 2.7182818284590451...

Pi()

Description

Returns the constant π, which is 3.1415926535897931...

Date and Time Functions

Datetime values are handled internally as numbers of seconds since midnight, January 1, 1904.

The expression `x=01Jan1904` sets *x* to zero, since the indicated date is the base date or "zero date" in JMP. If you examine the values of dates, they should be appropriately large numbers (for example, `5oct1998` is 2990390400).

Abbrev Date(*date*)

Description

Converts the provided *date* to a quoted string.

Returns

A quoted string representation of the date.

Argument

date Can be the number of seconds since the base date (midnight, January 1, 1904), or any date-time operator.

Example

```
Abbrev Date( 29Feb2004 );
```

02/29/2004

As Date(*x*)

Description

Formats the number or expression *x* so that it shows as a date or duration when displayed in a text window. Values that represent one year or more are returned as dates. Values that represent less than a year are returned as durations.

Returns

A date that is calculated from the number or expression provided.

Argument

x Number or expression.

Date Difference(*datetime1*, *datetime2*, *interval name*, <*alignment*>)

Description

Returns the difference in intervals of two date-time values.

Returns

A number.

Required Arguments

datetime1, *datetime2* Date-time values.

interval name A quoted string that contains a date-time interval, such as "month", "day", or "hour".

Optional Arguments

alignment A quoted string:

- "Start" includes full or partial intervals.
- "Actual" counts only whole intervals.
- "Fractional" returns fractional differences using averages for "year", "quarter", and "month" intervals.

Date DMY(*day*, *month*, *year*)

Description

Constructs a date value from the arguments.

Returns

The specified date, expressed as the number of seconds since midnight, 1 January 1904.

Arguments

day The numeric day of month, 1-31. Note that there is no error-checking, so you can enter February 31.

month The numeric month, 1-12.

year The year.

Date Increment(*datetime*, *interval name*, <*increment*>, <*alignment*>)

Description

Adds 1 or more intervals to a starting datetime value.

Returns

Returns the new datetime value.

Required Arguments

datetime The starting datetime value.

interval name A quoted string that contains the name of a datetime interval. "year", "quarter", "month", "week", "day", "hour", "minute", and "second" are supported.

Optional Arguments

increment A number that specifies the number of intervals. The default value is 1.

alignment A quoted string that contains a keyword:

- "Start" truncates the date to the nearest interval prior to adding the increment. For example, it removes the time and outputs the date. "start" is the default value.
- "Actual" retains the full input datetime value.

– "Fractional" allows fractional incremental values using averages for the duration of "Year", "Quarter", and "Month" intervals.

Date MDY(*month*, *day*, *year*)

Description

Constructs a date value from the arguments.

Returns

The specified date, expressed as the number of seconds since midnight, 1 January 1904.

Arguments

month The numeric month, 1-12.

day The numeric day of month, 1-31. Note that there is no error-checking, so you can enter February 31.

year The year.

Day(*datetime*)

Description

Determines the day of the month supplied by the *datetime* argument.

Returns

Returns an integer representation for the day of the month of the date supplied.

Arguments

datetime The number of seconds since midnight, 1 January 1904. This can also be an expression.

Example

```
d1 = Date DMY( 12, 2, 2003 );
   3127852800

Day( 3127852800 );
   12
Day( d1 );
   12
```

Day Of Week(*datetime*)

Description

Determines the day of the week supplied by the *datetime* argument.

Returns

Returns an integer representation for the day of the week of the date supplied. Weeks are Sunday–Saturday.

Arguments

datetime The number of seconds since midnight, 1 January 1904. This can also be an expression.

Day Of Year(*datetime*)

Description

Determines the day of the year supplied by the *datetime* argument.

Returns

Returns an integer representation for the day of the year of the date supplied.

Arguments

datetime The number of seconds since midnight, 1 January 1904. This can also be an expression.

Format(*x*, formatString, *width*|<*width*, *decimal places*>, <"Use Thousands Separator">)

Format(*x*, "Best", <*width*>, <"Use Thousands Separator">)

Format(*x*, ("Fixed Dec"|"Percent"), *width*|<*width*, *decimal places*>, <"Use Thousands Separator">)

Format(*x*, "Pvalue", <*width*>)

Format(*x*, ("Scientific"|"Engineering"|"Engineering SI"), <*width*>|<*width*, *decimal places*>)

Format(*x*,"Precision", *width*|<*width*, *decimal places*>, <"Use Thousands Separator">, <"Keep Trailing Zeroes">, <"Keep All Whole Digits">)

Format(*x*, "Currency", <*currency code*>, <*width*>|<*width*, *decimal places*>, <"Use Thousands Separator">, < <<Use Locale(*Boolean*) >)

Format(*x*, *datetime*, <*width*>)

Format(*x*, ("Latitude DDD"|"Latitude DDM"|"Latitude DMS"|"Longitude DDD"|"Longitude DDM"|"Longitude DDM"), *width*|<*width*, *decimal places*>, ("PUN"|"DIR"|"PUNDIR"))

Format(*x*, "Custom", Formula(), <*width*>)

Description

Converts the value *x* into the quoted *format* that you specify in the subsequent arguments.

Returns

Returns the text that corresponds to the number in the specified format.

Arguments

See *Using JMP* for more information about the arguments. The arguments are also shown in a data table column's Column Info window.

Examples

```
Format( x, "Fixed Dec", 10, 2, "Use Thousands Separator");
Format( x, "Currency", "EUR", 20, <<Use Locale(0)); // ignores computer locale
Format( x, "m/d/y", 10 );
Format( x, "Precision", 10, 2, "Keep trailing zeroes", "Keep All Whole Digits"
  );
Format( x, "Latitude DDD", "PUNDIR"); // "PUN" for punctuation, "DIR" for
  direction, PUNDIR for both
Format( x, "Custom", Formula( Abs( value ) ), 15 );
```

Notes

- You must always precede the number of decimal places with the width.
- If the date format is unknown, an error is written to the log.

Format Date(*x*, *datetime*, <*width*>)

Description

Converts the value of *x* into the quoted *datetime* that you specify in the second argument. Format choices are those shown in the data table Column Info window.

Returns

Returns the number in the specified format.

Arguments

See *Using JMP* for more information about the arguments.

Example

```
Format Date( Today(), "yyyQq" );
   "04/03/2020"
```

Hour(*datetime*, <"12"|"24">)

Description

Determines the hour supplied by the *datetime* argument.

Returns

Returns an integer representation for the hour part of the date-time value supplied.

Arguments

`datetime` The number of seconds since midnight, 1 January 1904. This can also be an expression.

`"12"|"24"` Changes the mode to 12 hours (with am and pm). The default is 24-hour mode.

HP Time()

Description

Returns a high precision time value (in microseconds). This function is only useful relative to another `HP Time()` value. The time value represents the number of microseconds since the start of the JMP session.

Notes

For less precise time values use `Tick Seconds()`.

In Days(*n*)

Description

Returns the number of seconds per *n* days. Divide by this function to express seconds as days.

Informat(*string*, *format*)

Parse Date(*string*, *format*)

Description

Parses a quoted *string* of a given quoted *format* and returns a date/time value. The value is expressed as if surrounded by the `As Date()` function, returning the date in "*ddMonyyyy*" format.

Example

```
Informat( "07152000", "MMDDYYYY" );
```

15Jul2000

Notes

- To see the format options, open the Column Info window on a data table column, select a date/time value for the format, and view the Input Format list.
- If the date format is unknown, an error is written to the log.

See Also

"As Date(x)"

In Hours(*n*)

Description

Returns the number of seconds per *n* hours. Divide by this function to express seconds as hours.

In Minutes(*n*)

Description

Returns the number of seconds per *n* minutes. Divide by this function to express seconds as minutes.

In Weeks(*n*)

Description

Returns the number of seconds per *n* weeks. Divide by this function to express seconds as weeks.

In Years(*n*)

Description

Returns the number of seconds per *n* years. Divide by this function to express seconds as years.

ISO Year(*datetime*)

Description

Returns the ISO year of the *datetime* supplied. ISO years correspond to ISO weeks; they begin on the Monday of the first week that contains at least four days.

Long Date(*date*)

Description

Returns a locale-specific quoted string representation for the *date* supplied, formatted like "Sunday, February 29, 2004" or "Wednesday, November 9, 2011".

MDYHMS(*date*)

Description

Returns a quoted string representation for the *date* supplied, formatted like "2/29/04 00:02:20".

Minute(*datetime*)

Description

Determines the minute supplied by the *datetime* argument, 0-59.

Returns

Returns an integer representation for the minute part of the date-time value supplied.

Month(*date*)

Description

Returns an integer representation for the month of the *date* that is supplied.

Nth Day of Week in the Month(*datetime*)

Description

Determines the day of the week of the *datetime* argument and how many instances of that day of the week have occurred in the month of the *datetime* argument.

Returns

Returns an integer that represents the number of instances of the day of the week of the *datetime* argument that have occurred in the month.

Parse Date()

See "Informat(string, format)".

Quarter(*datetime*)

Description

Returns the annual quarter of a *datetime* value as an integer 1-4.

Second(*datetime*)

Description

Determines the second supplied by the *datetime* argument.

Returns

Returns an integer representation for the second part of the date-time value supplied.

Argument

datetime Number of seconds since midnight, 1 January 1904. This can also be an expression.

Short Date(*date*)

Description

Returns a quoted string representation for the *date* supplied in the format MM/DD/YYYY.

Tick Seconds()

Description

Measures the time taken for a script to run, measured down to the 60th of a second.

Notes

For higher time value resolution (for example, microseconds) use the `HP Time()` function.

`Time Of Day(datetime)`

Description

Returns an integer representation for the time of day of the *datetime* supplied.

`Today()`

Description

Returns the current date and time expressed as the number of seconds since midnight, 1 January 1904. No arguments are available, but the parentheses are still necessary.

`Week Of Year(date, <ruleN>)`

Description

Returns the week of the year that contains a date-time value. Three rules determine when the first week of the year begins.

- With rule 1 (the default), weeks start on Sunday, with the first Sunday of the year being week 2. Week 1 is a partial week or empty.
- With rule 2, the first Sunday begins with week 1, with previous days being week 0.
- With rule 3, the ISO-8601 week number is returned. Weeks start on Monday. Week 1 is the first week of the year with four days in that year. It is possible for the first or last three days of the year to belong to the neighboring year's week number.

`Year(date)`

Description

Returns an integer representation for the year of *date*.

Discrete Probability Functions

`Beta Binomial Distribution(k, p, n, delta)`

Description

Returns the cumulative distribution function (cdf) of the beta binomial distribution. This is the probability that a beta binomially distributed random variable is less than or equal to *k*. The cdf is calculated as the summation of the beta binomial pmf for values of X from 0 to *k*.

Arguments

k The count of interest. *k* must be an integer.

p The probability of success for each trial, which must be between 0 and 1.

n The number of trials, which must be greater than 1.

delta The overdispersion parameter, which must be between Maximum[-*p*/(*n*-*p*-1), -(1-*p*)/(*n*-2+*p*)] and 1. When the overdispersion parameter is zero, the distribution reduces to Binomial(*n*, *p*).

Beta Binomial Probability(*k*, *p*, *n*, *delta*)

Description

Returns the probability mass function (pmf) of the beta binomial distribution. This is the probability that a beta binomially distributed random variable is equal to *k*. The pmf is parameterized as follows:

$$P(X = k; p, n, \delta) = \binom{n}{k} \frac{\Gamma\left(\frac{1}{\delta} - 1\right)\Gamma\left[k + p\left(\frac{1}{\delta} - 1\right)\right]\Gamma\left[n - k + (1-p)\left(\frac{1}{\delta} - 1\right)\right]}{\Gamma\left[p\left(\frac{1}{\delta} - 1\right)\right]\Gamma\left[(1-p)\left(\frac{1}{\delta} - 1\right)\right]\Gamma\left(n + \frac{1}{\delta} - 1\right)}$$

Arguments

k The count of interest. *k* must be an integer.

p The probability of success for each trial, which must be between 0 and 1.

n The number of trials, which must be greater than 1.

delta The overdispersion parameter δ, which must be between Maximum[-*p*/(*n*-*p*-1), -(1-*p*)/(*n*-2+*p*)] and 1. When the overdispersion parameter is zero, the distribution reduces to Binomial(*n*, *p*).

Notes

The beta binomial distribution results from assuming that X|π follows a Binomial(n,π) distribution and π follows a Beta(p(1-δ)/δ,(1-p)(1-δ)/δ) distribution. It is useful when the data are a combination of several Binomial distributions that each have different probabilities of success.

Beta Binomial Quantile(*p*, *n*, *delta*, *cumprob*)

Description

Returns the smallest integer quantile for which the cumulative probability of the Beta Binomial(p, n, delta) distribution is larger than or equal to *cumprob*.

Arguments

p The probability of success for each trial. *p* must be between 0 and 1.

n The number of trials, which must be greater than 1.

delta The overdispersion parameter δ, which must be between Maximum[-*p*/(*n*-*p*-1), -(1-*p*)/(*n*-2+*p*)] and 1. When the overdispersion parameter is zero, the distribution reduces to Binomial(*n*, *p*).

cumprob The cumulative probability of the quantile desired. *cumprob* must be between 0 and 1.

Binomial Distribution(*p*, *n*, *k*)

Description

Returns the cumulative distribution function (cdf) of the binomial distribution. This is the probability that a binomially distributed random variable is less than or equal to *k*. The cdf is calculated as the summation of the binomial pmf for values of X from 0 to *k*.

Arguments

p The probability of success for each trial. *p* must be between 0 and 1.

n The number of trials.

k The number of successes, which must be less than or equal to *n*.

Binomial Probability(*p*, *n*, *k*)

Description

Returns the probability mass function (pmf) of the binomial distribution. This is the probability that a binomially distributed variable is equal to *k*. The pmf is parameterized as follows:

$$P(X = k;p, n) = \binom{n}{k} p^k (1-p)^{n-k}$$

Arguments

p The probability of success for each trial. *p* must be between 0 and 1.

n The number of trials.

k The number of successes, which must be less than or equal to *n*.

Binomial Quantile(*p*, *n*, *cumprob*)

Description

Returns the smallest integer quantile for which the cumulative probability of the Binomial(p, n) distribution is larger than or equal to *cumprob*.

Arguments

p The probability of success for each trial. *p* must be between 0 and 1.

n The number of trials.

cumprob The cumulative probability of the quantile desired. *cumprob* must be between 0 and 1.

Gamma Poisson Distribution(*k*, *lambda*, *sigma*)

Description

Returns the cumulative distribution function (cdf) of the gamma-Poisson distribution. This is the probability that a gamma-Poisson distributed random variable is less than or equal to *k*. The cdf is calculated as the summation of the gamma-Poisson pmf for values of X from 0 to *k*.

Arguments

k The count of interest. *k* must be an integer.

lambda The shape parameter λ, which much be greater than 0. This is the mean of the distribution.

sigma The overdispersion parameter σ, which must be greater than or equal to 1. When the overdispersion parameter is 1, the distribution reduces to a Poisson(λ) distribution.

Gamma Poisson Probability(*k*, *lambda*, *sigma*)

Description

Returns the probability mass function (pmf) of the gamma-Poisson distribution. This is the probability that a gamma-Poisson distributed random variable is equal to *k*. The pmf is parameterized as follows:

$$P(X = k;\lambda, \sigma) = \frac{\Gamma\left(k + \frac{\lambda}{\sigma - 1}\right)}{\Gamma(k+1)\Gamma\left(\frac{\lambda}{\sigma - 1}\right)}\left(\frac{\sigma - 1}{\sigma}\right)^{k}\left(\sigma^{-\left(\frac{\lambda}{\sigma - 1}\right)}\right)$$

where Γ(·) is the Gamma function.

Arguments

k The count of interest. *k* must be an integer.

lambda The shape parameter λ, which much be greater than 0. This is the mean of the distribution.

sigma The overdispersion parameter σ, which must be greater than or equal to 1. When the overdispersion parameter is 1, the distribution reduces to a Poisson(λ) distribution.

Notes

The gamma Poisson distribution results from assuming that X|μ follows a Poisson(μ) distribution and μ follows a Gamma(λ/(σ-1),σ-1) distribution. It is useful when the data are a combination of several Poisson(μ) distributions that each have different values of μ.

Gamma Poisson Quantile(*lambda*, *sigma*, *cumprob*)

Description

Returns the smallest integer quantile for which the cumulative probability of the Gamma Poisson(lambda, sigma) distribution is larger than or equal to *cumprob*.

Arguments

lambda The shape parameter λ, which much be greater than 0. This is the mean of the distribution.

sigma The overdispersion parameter σ, which must be greater than or equal to 1. When the overdispersion parameter is 1, the distribution reduces to a Poisson(λ) distribution.

cumprob The cumulative probability of the quantile desired. *cumprob* must be between 0 and 1.

Hypergeometric Distribution(*N*, *K*, *n*, *x*, <*r*>)

Description

Returns the cumulative distribution function (cdf) of the hypergeometric distribution. This is the probability that a hypergeometrically distributed random variable is less than or equal to *x*. The cdf is calculated as the summation of the hypergeometric pmf for values of X from 0 to *x*.

Required Arguments

N The population size.

k The number of items in the category of interest.

n The sample size.

x The count of interest, which must be less than or equal to *n* and *k*.

Optional Argument

r The odds ratio.

Hypergeometric Probability(*N*, *k*, *n*, *x*, <*r*>)

Description

Returns the probability mass function (pmf) of the hypergeometric distribution. This is the probability that a hypergeometrically distributed random variable is equal to *x*. The pmf is parameterized as follows:

$$P(X = x;N, n, k) = \frac{\binom{k}{x}\binom{N-k}{n-x}}{\binom{N}{n}}, n - x \le N - k$$

Required Arguments

N The population size.

k The number of items in the category of interest.

n The sample size.

x The count of interest, which must be less than or equal to *n* and *k*.

Optional Argument

r The odds ratio.

Neg Binomial Distribution(*p*, *n*, *k*)

Description

Returns the cumulative distribution function (cdf) of the negative binomial distribution. This is the probability that a negative binomially distributed random variable is less than or equal to *k*. The cdf is calculated as the summation of the negative binomial pmf for values of X from 0 to *k*.

Arguments

p The probability of success for each trial. *p* must be between 0 and 1.

n The number of successes.

k The number of failures before the n^{th} success.

Neg Binomial Probability(*p*, *n*, *k*)

Description

Returns the probability mass function (pmf) of the negative binomial distribution. This is the probability that a negative binomially distributed random variable is equal to *k*. The pmf is parameterized as follows:

$$P(X = k; p, n) = \binom{n + k - 1}{k} p^n (1 - p)^k$$

Arguments

p The probability of success for each trial. *p* must be between 0 and 1.

n The number of successes.

k The number of failures before the n^{th} success.

Notes

The return value of the pmf is the probability of observing the n^{th} success after *k* failures have occurred.

Poisson Distribution(*lambda*, *k*)

Description

Returns the cumulative distribution function (cdf) of the Poisson distribution. This is the probability that a Poisson distributed random variable with mean *lambda* is less than or

equal to *k*. The cdf is calculated as the summation of the Poisson pmf for values of X from 0 to *k*.

Arguments

k The number of events in a given time interval. *k* must be an integer.

lambda The shape parameter λ, which much be greater than 0. This is the mean of the distribution.

`Poisson Probability(lambda, k)`

Description

Returns the probability mass function (pmf) of the Poisson distribution. This is the probability that a Poisson distributed random variable with mean *lambda* is equal to *k*. The pmf is parameterized as follows:

$$P(X = k;\lambda) = \frac{e^{-\lambda}\lambda^{k}}{k!}$$

Arguments

k The number of events in a given time interval. *k* must be an integer.

lambda The shape parameter λ, which must be greater than 0. This is the mean of the distribution.

`Poisson Quantile(lambda, cumprob)`

Description

Returns the smallest integer quantile for which the cumulative probability of the Poisson(lambda) distribution is larger than or equal to *cumprob*.

Arguments

lambda The shape parameter λ, which must be greater than 0. This is the mean of the distribution.

cumprob The cumulative probability of the quantile desired. *cumprob* must be between 0 and 1.

Display Functions

`Alpha Shape(Triangulation())`

Description

Returns the alpha shape for the given triangulation.

Border Box(<Left(*pix*)>, <Right(*pix*)>, <Top(*pix*)>, <Bottom(*pix*)>, <Sides(*Boolean*)>, *db*)

Description

Constructs a bordered display box that contains another display box. Optional arguments (Left, Right, Top, Bottom) add space between the border box and what it contains. The other optional argument (Sides) draws borders around the border box on any single side or combination of sides; draws the border in black or the highlight color; makes the background transparent or white or erases the background of a display box that contains it.

Returns

The display box.

Required Argument

db A display box object (for example, a text box or another border box).

Optional Arguments

Left(*pix*) An integer that measures pixels.

Right(*pix*) An integer that measures pixels.

Top(*pix*) An integer that measures pixels.

Bottom(*pix*) An integer that measures pixels.

Sides(*pix*) An integer that maps to settings for the display box.

Notes

The formula for deriving the integer for Sides is: 1*top + 2*left + 4*bottom + 8*right + 16*highlightcolor + 32*whitebackground + 64*erase. Thus, if you want to just draw a black border on the top and bottom, 1+4 = 5. If you want that same box with a white background, 5+32 = 37.

Box Plot Seg(<*data*>, <*frequency*>, <*weight*>, <Vertical(*Boolean*)>)

Description

Returns a display seg that represents a box plot based on the passed x and y values.

Returns

The display box (a box plot).

Optional Arguments

data The data values within the box plot.

frequency The frequency values within the box plot.

weight The weights for observations on continuous Ys.

Vertical(*Boolean*) A vertical (1) or horizontal (0) box plot.

Example

```
win = New Window( "Box Plot Seg Example",
Graph Box(
```

```
      Frame Size( 40, 180 ),
      Y Scale( 0, 100 ),
      Box Plot Seg(
         [20, 30, 40], // data
         [1, 1, 3], // frequencies
         [1, 1, 1], // weights
         1 // vertical
         )
      )
   );
```

Busy Light(< <<Automatic(*Boolean*)>, <Size(*x*, *y*)>, < <<Disable>)

Description

Creates a rotating image to indicate a busy process.

Returns

A rotating image.

Optional Argument and Messages

<<Automatic(*Boolean*) Rotates the image.

Size(*x*, *y*) Specifies the size of the image in pixels.

<<Disable Hides the image.

Example

```
win = New Window( "Example",
   blb = Busy Light( <<Automatic( 1 ), Size( 50, 50 ) ) );
```

Button Box(*title*, *script*, < *<<Set Icon(path)*>, < <<Set Icon Location(left|right)>

Description

Constructs a button with the text *title* that executes *script* when clicked.

Returns

The display box (button box).

Required Arguments

title A quoted string or a string variable.

script A quoted string or a reference to a string that specifies a valid JSL script.

Optional Messages

<<Set Icon("*path*") Displays the image in the quoted pathname on the button. Most common graphic formats are supported, such as GIF, JPG, PNG, BMP, TIF. Since the title argument is optional, you can create a button with only a text title, with only an icon, or with both a text title and an icon. In the last case, the icon is placed next to the text title.

<<Set Icon Location("left"|"right") Allows the position of the icon on a button to be either left or right of the text.

Example

The following example shows a simple button box. When the user clicks the button box, "Pressed" is printed to the log.

```
win = New Window( "Simple Example",
   ex = Button Box( "Press me" )
);
ex << Set Script( Print( "Pressed" ) );
```

Notes

Line-break characters are ignored in button boxes.

Calendar Box(*title*, < <<Date>, < <<Min Date>, < <<Max Date>, < <<Show Time>)

Description

Constructs a pop-up calendar with selectable days and time.

Returns

The display box (calendar box).

Required Argument

title A quoted string or a string variable.

Optional Messages

<<Date The currently selected date.

<<Min Date The earliest date that can be selected.

<<Max Date The latest date that can be selected.

<<Show Time The time that can be specified.

Example

The following example creates a calendar with October 5, 1989 initially selected. The minimum date and maximum date are specified, so the user can select only dates in that range.

```
New Window( "Calendar Box Example", cal = Calendar Box() );
date = Date MDY (10, 5, 1989);
cal << Date( date );
cal << Show Time( 0 ); // omit the time

/* earliest date that can be selected is 60 days before 10/5/1989
"start" truncates the value so the time is not included */
cal << Min Date( Date Increment(date, "Day", -60, "start" ) );

// latest date that can be selected is 60 days after 10/5/1989
cal << Max Date( Date Increment(date, "Day",  60, "start" ) );
```

```
cal << Set Function( Function( {this}, Print( Abbrev Date(this << Get Date())
  ) ) ); // print the abbreviated date to the log
```

Cell Plot(*Y(column(s))*, <*X(column)*>)

Description

Displays each value in a cell graph.

Check Box({*list*}, <*script*>, < <<Get(*n*)>, < <<Set(*n*, *Boolean*)>, < <<Get Selected>, < <<Enable Item(*n*, *Boolean*)>, < <<Item Enabled(*check box item*)>)

Description

Constructs a display box to show one or more check boxes.

Returns

The display box (Check Box).

Required Argument

list A list of quoted strings or a reference to a list of strings.

Optional Argument

script An optional JSL script.

Optional Messages

<<Get(*n*) Returns 1 if the check box item specified by *n* is selected, or 0 otherwise.

<<Set(*n*, *Boolean*) Sets the check box item specified by *n* as either selected (1) or cleared (0).

<<Get Selected Returns a list of quoted strings that contain the names of the check box items that are selected.

<<Enable Item(*n*, *Boolean*) Sets the check box item specified by *n* as either enabled (1) or disabled (0). The state of a disabled check box cannot be changed.

<<Item Enabled(*check box item*) Returns 0 or 1 depending on whether the specific check box item is enabled.

Example

Create three check boxes labeled “one”, “two”, and “three”. The first check box is selected.

```
New Window( "Example", Check Box( {"one", "two", "three"}, <<Set( 1, 1 ) ) );
```

Col Box(*title*, *display boxes*)

Description

Returns a column box made up of the specified display boxes.

Arguments

title The quoted title of the column.

display boxes Display boxes that hold content within the column box.

Example

```
win = New Window( "Example",
   exx = 1;
   exy = 4;
   exz = 8;
   Table Box(
      String Col Box( "strings", {"x", "y", "z"} ),
      Col Box(
         "boxes",
         Slider Box( 0, 10, exx, Show( exx ) ),
         Slider Box( 0, 10, exy, Show( exy ) ),
         Slider Box( 0, 10, exz, Show( exz ) )
      )
   );
);
```

`Col List Box(<Data Table(`*name*`)>, <"All"|"Character"|"Numeric">, <width(`*pixels*`)>, <"Grouped">, <MaxSelected(`*n*`)>, <nLines(`*n*`)>, <MaxItems(`*n*`)>, <MinItems(`*n*`)>, <On Change(`*expr*`)>, < <<Modeling Type("Any"|"Continuous"|"Ordinal"|"Nominal"|Multiple Response"|"Unstructured Text"|"Vector"|"None"|"Row State") >, < <<Set Data Type(Any|Numeric|Character)>, <`*script*`>)`

Description

Constructs a display box to show a list box that allows selection of data table columns.

Returns

The display box (Col List Box).

Optional Arguments

`Data Table(`*name*`)` The quoted name of the data table.

`"All" | "Character" | "Numeric"` Adds all columns of the current data table into the list. Omitting `"All"` results in an empty col list box with the "optional" label. To display "optional character", specify `"Character"`. To display "optional numeric", specify `"Numeric"`.

`width(`*pixels*`)` Sets the width of the list box to `pixels`. `pixels` is a number that measures pixels.

`"Grouped"` Displays grouped columns in the box.

`MaxSelected(`*n*`)` Sets whether only one item can be selected. For *n*>1, *n* is ignored.

`nLines(`*n*`)` Sets the length of the list box to n number of lines. *n* is an integer.

script A script.

`MaxItems(`*n*`)` A number that allows only *n* columns to be added to the list.

`MinItems(`*n*`)` A number that only requires at least *n* columns for the list. If *n*=2, the top two slots in the col list box an initial display of "required numeric" (or whatever you set the data type to be).

On Change(*expr*) Evaluates the expression when the selection in the list changes. Dragging between two column list boxes that have this argument results in both expressions being evaluated. The expression for the target being dragged is evaluated first, then the expression for the source is evaluated.

Optional Messages

<<Set Tips({*Tip text 1*, *Tip text 2*, ...}) Quoted strings that set tool tips for items in the list box. A quoted null string or an empty list results in no tips. A list shorter that the list of items in the list box will use the last tip text for the remaining items in the list and the list box.

<<Set Tip(*Tip text*) A quoted string that overrides any tool tips set using Set Tips() function. If there is a tip set for the box, you cannot set tips for each individual item.

Using Set Tip() with no arguments clears the list box tip and allows the individual item tool tips to be displayed.

Notes

- The MaxSelected(*n*) argument only affects whether one or more than one item can be selected. It does not enforce a limit greater than 1.
- Specialty modeling types can be used only in a role (determined by the platform) that explicitly accepts columns of the same type.

Col Span Box(*title*, *display box args*)

Description

Creates spanned columns headers inside a table box. The top column header spans two child column headers.

Returns

The display box (a Col Span Box).

Arguments

title The title that appears in the box.

display box args Display boxes.

Example

```
win = New Window( "Col Span Box",
   <<Modal,
   Table Box(
      Col Span Box(
         "Confidence Limits",
         neb = Number Col Edit Box( "Upper limits", [0, 0] ),
         Number Col Edit Box( "Lower limits", [0, 0] )
      )
   )
);
```

Combo Box(*{items <(tip string)>, ...}*, *<script>*)

Description

Constructs a display box to show a drop-down list.

Returns

The display box (Combo Box).

Arguments

items The items that the user can select.

tip string A quoted string that specifies tooltip text.

script An optional JSL script.

Context Box(*display box*, ...)

Description

Defines a scoped evaluation context. Each Context Box is executed independently of each other.

Returns

A display box.

Arguments

Any number of display boxes.

Contour Seg(Triangulation, [levels], <zColor([*colors*]|{*colors*}, <"Cycle Colors"|"Interpolate Colors">)>, <"Fill"|"Fill Between"|"Fill Below"|"Fill Above">, <Transparency([]|*t*)>)

Description

Returns a display seg that represents contours of a Triangulation.

Required Arguments

Triangulation The columns to include in the Triangulation.

[levels] A matrix of values that control the contour levels that are drawn.

Optional Arguments

zColor([*colors*]|{colors} Colors for each level, specified as a list or matrix.

"Cycle Colors"|"Interpolate Colors Cycle Colors" alternates the colors (for example, red, green, red, green). With Interpolate Colors, the first contour is red, and the last is green. The contours between smoothly blend the colors.

"Fill Below"|"Fill Between"|"Fill Above"|"Fill" "Fill Below" fills the region below the lines. "Fill Between" fills only the middle region. "Fill Above" fills the region above the lines.. "Fill" works like "Fill Above", but the default is to display lines if no Fill options are specified.

Transparency([]|*t*) The transparency specified as a number or matrix.

Example

```
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
tri = Triangulation( X( :X, :Y ), Y( :POP ) );
{xx, yy} = tri << Get Points();
win = New Window( "Contour Seg Example",
   g = Graph Box(
      X Scale( Min( xx ) - .1, Max( xx ) + .1 ),
      Y Scale( Min( yy ) - .1, Max( yy ) + .1 ),
      Contour Seg(
         tri,
         [0, 400, 1000, 2000, 9000],
         zColor( 5 + [64 32 0 16 48] ),
         Transparency( [1, 1, 1, 1, 1] )
      )
   )
);
```

Notes

The triangulation is computed using the Xs, and the Y is a continuous variable defined at each position. The [levels] in this case defines values of POP that are drawn as lines, one line per level. If any Fill argument is specified, then the filled regions are [level1, level2], [level2, level3], ..., [level-n].

Current Journal()

Description

Gets the display box at the top of the current (topmost) journal.

Returns

Returns a reference to the display box at the top of the current journal.

Current Report()

Description

Gets the display box at the top of the current (topmost) report window.

Returns

Returns a reference to the display box at the top of the current report window.

Current Window()

Description

Returns a reference to the current window.

Data Filter Context Box(*display box*)

Description

Returns a display box that defines the extent of the local data filters that a display tree contains. Data filters and Data Filter Context Boxes can be arranged in a hierarchy and shared among platforms or boxes that the Data Filter Context Boxes contain.

Data Filter Source Box(*display box*)

Description

Defines which graph is the "source" of the selection filter. Selected rows in reports that are within the Data Filter Source box are included for analysis in the other reports that are within a common Data Filter Context Box.

Data Table Box(*data table*)

Description

Returns a table box the represents the specified data table.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Example", Data Table Box( dt ) );
```

Data Table Col Box(*column*)

Description

Returns a column box that corresponds to the specified data table column.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Example",
   Table Box( Data Table Col Box( :name ), Data Table Col Box( :height ) )
);
```

Dialog(*contents*)

Description

`Dialog` is deprecated. Use `New Window()` with the `Modal` argument instead.

Excerpt Box(*report*, *subscripts*)

Description

Returns a display box containing the excerpt designated by the report held at number *report* and the list of display subscripts *subscripts*. The subscripts reflect the current state of the report, after previous excerpts have been removed.

Expr As Picture(*Expr*(...), <Width(*pixels*)>)

Description

Converts expr() to a picture as it would appear in the Formula Editor.

Returns

Reference to the picture.

Required Argument

Expr(...) Place any valid JSL expression that can be displayed as a picture inside expr().

Optional Argument

Width(*pixels*) Sets the width of the box to pix. pix is a number that measures pixels.

Filter Col Selector(<Data Table(*name*)>, <Width(*pixels*)>, <nLines(*n*)>, <*script*>, <OnChange(*expr*)>)

Description

Returns a display box that contains a list of items. Control allows column filtering.

Get Project(*title*|*index*|*display box*|*window*)

Description

Returns a single project.

Examples

The following examples show how to get the window title of various projects.

```
Get Project( 1 ) << Get Window Title;
// first open project
Get Project( "My Project" ) << Get Window Title;
// first project named “My Project”
Get Project( display box ) << Get Window Title;
// parent project of the specified display box
```

Get Project List()

Description

Returns a list of all open projects.

Example

```
Get Project List() << Get Window Title;
// list of the titles of all open projects
```

Get Window(<Project(*title*|*index*|*display box*|*window*)>, <Type(*string*)>, *title*|*index*|*display box*)

Description

Returns a reference to a specific open window by title, index, or display box. When run in a project, Get Window() returns windows from the current project.

Optional Arguments

Project Specifies the title (a quoted string), index, display box, or window from another project.

Type(*string*) To limit the search to particular types of windows, use the Type() argument and one of these quoted strings: "Data Tables", "Journals", "Reports", or "Dialogs".

Examples

The following examples show how to get the window title of various windows.

```
Get Window( 1 ) << Get Window Title;
// first window in the current project

Get Window( "Big Class" ) << Get Window Title;
// Big Class window in the current project

Get Window( ob ) << Get Window Title;
// parent window of specified display box in the current project

Get Window( Project(), 1 ) << Get Window Title;
// first window, no project (global scope)

Get Window( Project( myProject ), "Big Class" << Get Window Title;
// Big Class window in the specified project
```

Get Window List(<Project(*title*|*index*|*display box*|*window*),><Type(*string*)>)

Description

Returns a list of currently open windows. By default, Get Window List() returns references to open windows in the current project. You can return an open window list from something other than the current project by using the Project() argument. To limit the search to particular types of windows, use the Type () argument and one of these quoted strings: "Data Tables", "Journals", "Reports", or "Dialogs".

Optional Arguments

Project Specifies the title, index, display box, or window from another project.

Type To limit the search to particular types of windows, use the Type () argument and one of these quoted strings: "Data Tables", "Journals", "Reports", or "Dialogs".

Examples

```
Get Window List() << Get Window Title;
```

```
// list of open windows in the current project
Get Window List( Type( "Reports" ) ) << Get Window Title;
// list of the titles of open reports in the current project
Get Window List( Project( 0 ), Type( "Reports" ) ); // positional arguments
// list of the titles of open reports outside of a project
Get Window List( 2 );
// second window list
```

Global Box(*global*)

Description

Constructs a box for editing *global* value directly.

Graph()

See "Graph Box(properties, script)".

Graph 3D Box(*properties*)

Description

Constructs a display box with 3-D content.

Returns

The display box.

Arguments

properties Properties can include: Frame Size(x, y), Xname("title"), Yname("title"), Zname("title").

Notes

This display box constructor is experimental.

Graph Box(*properties*, *script*)

Graph(*properties*, *script*)

Description

Constructs a graph with axes.

Returns

The display box (Graph Box).

Arguments

properties Named property arguments: Title("title"), XScale(low, high), YScale(low, high), FrameSize(h, v), XName("x"), YName("y"), SuppressAxes.

script Any script to be run on the graph box.

H Center Box(<*child box*>)

Returns a display box that contains the optional child display box argument. The box is centered in the horizontal space defined by the maximum size of that child display box and all of the other siblings of the center box.

H List Box(<Align("center"|"bottom")>, display box, ...)

Description

Creates a display box that contains other display boxes and displays them horizontally.

Arguments

`Align("center"|"bottom")` Specify center or bottom alignment of the contents in the list box. The contents are bottom aligned by default.

display box Any number of display box arguments can be contained in the list box.

H Scroll Box(<Size(*h*)>, *display box*)

Description

Returns a display box that positions a larger child box using a horizontal scroll bar.

Arguments

`Size(`*h*`)` The horizontal length of the scroll bar.

Notes

The flexible argument is deprecated. Use `Set Stretch` instead. See "V Scroll Box(<Size(v)>, display box)" for an example.

H Sheet Box(<<Hold(*report*), *display boxes*)

Description

Returns a display box that arranges the display boxes provided by the arguments in a horizontal layout. The `<<Hold()` message tells the sheet to own the report(s) that is excerpted.

H Splitter Box(<Size(*h*, *v*)>, *display box*, ...)

Description

Returns a display box that arranges the display boxes provided by the arguments in a horizontal layout (or panel). The splitter enables the user to interactively resize the panel.

Required Argument

display box Any number of display box arguments can be contained in the splitter box.

Optional Argument

`Size(h, v)` Specifies the size of the splitter box in pixels. This size is for the outer splitter box. Inner display boxes are proportionately sized according to the width and height of the outer splitter box.

Optional Messages

`<<Size(n)` Specifies the proportions of the last panel. `<<Size(.25)` resizes the last panel to 25% the splitter box height (or width, for vertical splitter boxes).

`<<Set Sizes({n, n})` Specifies the proportions of each panel. `db<<Set Sizes({.75, .25})` sizes the first panel to 75% and the second panel to 25% of the splitter box height (or width, for vertical splitter boxes).

`<<Close Panel(n, <Boolean>)` Closes the panel that you specify. `<<Close Panel(2)` closes the second panel. With three or more panels, you must include the second Boolean value. That value indicates which panel expands to fill the space left by the closed panel.

- `<<Close Panel(2,0)` closes the second panel; the following sibling takes the extra space.
- `<<Close Panel(2,1)` closes the second panel; the preceding sibling takes the extra space.

`<<Open Panel(n, <Boolean>)` Opens the panel that you specify. With three or more panels, you must include the second Boolean value. Works similar to `<<Close Panel` described above. The panels are initially opened. You use `<<Open Panel` only after using `<<Close Panel`.

`<<Get Sizes()` Returns the proportions of each panel as a list.

Hier Box(text, Hier Box(...), ...)

Description

Constructs a node of a tree (similar to Diagram output) containing text. `Hier Box` can contain additional `Hier Boxes`, allowing you to create a tree. The text can be a quoted string or a `Text Edit Box`.

Hist Seg([*data*], <[*freq column*]>, <[*weight column*]>, <Vertical(*Boolean*)>, <Row States()>

Description

Returns a histogram seg.

Required Argument

data The data in matrix format.

Optional Arguments

freq column The frequency column in matrix format.

weight column The weight column in matrix format.

`Vertical(Boolean)` Displays the histogram vertically by default (or if set to 1). Display the histogram horizontally by setting the value to 0.

`Row States` Specifies a data table reference or row states.

`Icon Box(name)`

Description

Constructs a display box containing an icon, where the argument is a name such as Popup, Locked, Labeled, Sub, Excluded, Hidden, Continuous, Nominal, or Ordinal. The argument can also be a path to an image.

Argument

name A quoted string that is the name of a JMP icon or the path to an icon.

Example

`Icon Box( "Nominal" )` constructs a display box that contains the Nominal icon.

`Icon Box( "$SAMPLE_IMAGES/pi.gif" )` inserts the pi.gif sample image.

Notes

- Some icons are used on both Windows and macOS. Other icons are platform specific.
- Consider installing the Built-In JMP Icons add-in. The add-in lets you view icons interactively and see what they look like in different contexts. Download the add-in from https://community.jmp.com/t5/JMP-Add-Ins/Built-In-JMP-Icons/ta-p/42251.

`If Box(Boolean, display boxes)`

Description

Constructs a display box whose contents are conditionally displayed.

Arguments

Boolean 1 displays the display boxes inside the `If Box`. 0 does not display them.

display boxes Any display box tree.

`If Seg(<State(Boolean)>)`

Description

Returns a display seg that shows or hides display seg children.

Arguments

`State(Boolean)` Determines whether the display seg children are shown (1) or hidden (0).

Example

```
lines = [30 20 80 70, 10 90 90 10, 40 20 60 30];
win = New Window( "Lines Seg Example",
   g = Graph Box( If Seg( true, <<Append( Lines Seg( lines ) ) ) )
```

```
);
```

`Journal Box(string)`

Description

Constructs a display box that displays the quoted string. We recommend that you do not generate the journal text by hand.

`Line Seg(x, y, <Row States(dt|dt, [rows]|dt, {{rows}, ...}|{row states})>, <Sizes(s)>)`

Description

Creates a display seg of connected line segments for the given *x* and *y* values. The optional third argument enables row state assignments from either a data table or independently.

`Lines Seg([x1 y1 x2 y2, ...])`

Description

Returns a display seg with a sequence of line segments for the given *x* and *y* values.

`Lineup Box(<nCol(n)>, <Spacing(pixels, <vspace>), display boxes, ...)`

Description

Constructs a display box to show an alignment of boxes in *n* columns.

`List Box({item, ...}, <Width(pixels)>, <maxSelected(n)>, <nLines(n)>, <script>)`

Description

Creates a display box to show a list box of selection items (quoted strings). The argument can be a list of two-item lists containing the item name and a quoted string that specifies the modeling type or sorting order. Item names are case sensitive by default. The icon appears next to the corresponding item in the list box.

`Marker Seg(x, y, <Row States(dt|dt, [rows]|dt, {{rows}, ...}|{row states})>, <Sizes(s)>)`

Description

Creates a display seg with markers for all of the x and y values. The optional third argument enables row state assignments from either a data table or independently.

Matrix Box(*x*)

Matrix Box(*matrix*, < <<Column Names(*col1*, *col2*, ...)>, < <<Row Names(*row1*, *row2*, ...)>>)

Description

Displays the *matrix* given in the usual array form. Column and row names are quoted strings.

Mouse Box(*display box arguments*)

Description

Returns a box that can make JSL callbacks for dragging and dropping, marking, or clicking and tracking mouse actions.

Arguments

display box arguments Specifies the object that the user interacts with, such as a Text Box or Button Box. See the Scripting Index in the Help menu.

Move to Project(<Source(*project*)|Destination(*project*)>, <Windows({*list of windows to move*})>)

Description

Moves one or more windows into a project or out of a project, or between projects.

Arguments

Source(*project*) The project containing the windows that you want to move.

Destination(*project*) The project to which you want to move the windows.

Windows({*list of windows to move*}) A list of windows to move to the project. If omitted, all windows will be moved. Note that the data table and all of its dependent reports will be moved. However, you need to specify only the data table name or report name in the windows argument to move it.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
report = dt << Run Script( "Bivariate" );
project = New Project();
// move the report and data table to a new project
Move to Project( Destination( project ), Windows( {report} ) );
```

New Image()

New Image(*width*, *height*)

New Image(*filepath*)

New Image(Open(*url*))

New Image(*picture*)

New Image(*matrix*)

New Image(rgb|r|rgba, {*matrix*, ...})

Description

Creates a new image which can then be edited using JSL. Valid formats are png, bmp, jpeg, jpg, tiff, tif, and gif.

Returns

An image.

Arguments

All argument sets are optional, but all arguments within each set are required.

width, height Sets the width and height of the image in pixels.

filepath The quoted filepath to an image.

Open(*url*) Opens the image at the specified URL path.

picture A JSL picture object.

matrix The image as a matrix of JSL color pixels.

rgb|r|rgba, {*matrix*, ...} Specifies the channels ("rgb", "r", or "rgba") and provides a matrix of values (0.0-1.0) for each channel. Examples:

```
New Image( "r", [r matrix] );
New Image( "rgb", {[r matrix], [g matrix], [b matrix]} );
New Image( "rgba", {[r matrix], [g matrix], [b matrix], [a matrix]} );
```

New Project(*arguments*)

Description

Creates a project using the specified script.

Arguments

<<Add Bookmarks({<File(*path*)>, <Folder(*path*, Expanded(*Boolean*)>)>, <Group(*name*, <Expanded(*Boolean*)>, {*contents*}>) Creates bookmarks for frequently used files in the project. The argument is a list of bookmark items, each of which is specified using File(), Folder(), or Group(). Group() accepts File(), Folder(), and Group() as children.

<<Reset Layout Sets the project to use the default layout.

<<Run Script Specifies the data tables and reports that appear in the project.

<<Save(*<path>*) Saves the project. Include a quoted path and file name to save the project to a specific location. Save As is an alias.

<<Set Bookmarks({<File(*path*)>, <Folder(*path*, Expanded(*Boolean*)>)>, <Group(*name*, <Expanded(*Boolean*)>, {*contents*}>) Sets the bookmarks for the project. The argument is a list of bookmark items, each of which is specified using File(), Folder(), or Group(). Group() accepts File(), Folder(), and Group() as children.

<<Set Layout Sets the window layout of the project.

<<Show Bookmarks Shows or hides the bookmarks.

<<Show Log Shows or hides the log.

<<Show Window List Shows or hides the Window List.

Example

The following example creates a project from BigClass.jmp and two reports.

```
project = New Project();
project << Run Script(
   dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
   dt << Run Script( "Bivariate" );
   dt << Run Script( "Distribution" );
);
```

New Window(*title*, *<arguments>*, *display box*)

Description

Makes a new window with the required quoted title and a display box tree.

Optional Arguments

<<Script(*<script>*) Creates a new script window. The optional quoted string *script* is placed inside the script window.

<<Journal Creates an empty journal.

<<Size Window(*x*, *y*) Creates a new window of the specified height and width.

<<Modal Makes the new window a modal window, which prevents any other actions in JMP until the window is closed. If you do not include an **OK** or **Cancel** button, one is added automatically for you. **Note:** If used, this argument must be the second argument, directly after the window title. Available modal window arguments are:

– <<On Open(*expr*) runs expr when the window is created.

Note: In data tables, On Open (or OnOpen) scripts that execute other programs are never run. Set the Evaluate OnOpen Scripts preference to control when the script is run.

– <<On Close(*expr*) runs expr when the window is closed. Returns 0 if the window fails to close.

- – `<<On Validate(expr)` runs `expr` when the **OK** button is pressed. If it returns True, the window is closed otherwise the window remains open.
- – `<<Return Result` changes the window's return value when it closes to match that of the deprecated `Dialog()` function.

`Show Toolbars(Boolean)` Shows or hides the toolbar. The default value is 1. (Windows only.)

`Show Menu(Boolean)` Shows or hides the menu bar. The default value is 1. (Windows only.)

`Suppress AutoHide(Boolean)` Suppresses or uses the auto-hide feature for menus and toolbars. The default value is 1. (Windows only).

Notes

`Dialog()` is deprecated. Use `New Window()` with the `Modal` argument instead.

`Number Col Box(title, numbers)`

Description

Creates a column named *title* (a quoted string) with numeric entries given in list or matrix form.

`Number Col Edit Box(title, {numbers}|[numbers])`

Description

Creates a column named *title* (a quoted string) with numeric entries given in list or matrix form. The numbers can be edited.

`Number Edit Box(initial value, <width>)`

Description

Creates an editable number box that initially contains the *initial value* argument.

Returns

The display box object.

Argument

initial value Any number to use as the initial value. If you use a date or time format, a date and time selector window is created.

<width> Sets the width of the box in characters.

`Outline Box(title, display box, ...)`

Description

Creates a new outline named *title* (a quoted string) containing the listed display boxes.

Page Break Box()

Description

Creates a display box that forces a page break when the window is printed.

Panel Box(*title*, *display box*)

Description

Creates a display box labeled with the quoted string *title* that contains the listed display boxes.

Picture Box(Open(*picture*), *format*)

Description

Creates a display box that contains a graphics picture object.

Returns

A reference to the display box.

Argument

Open(*picture*) Opens the directory that contains the picture.

format Specifies the graphic file format. Specifying the format opens the picture in JMP. If you omit this argument, the picture opens in the default graphics program. Valid formats are the quoted strings "png", "bmp", "jpeg", "jpg", "tiff", "tif", and "gif".

Example

```
New Window( "Example",
   Picture Box( Open( "$SAMPLE_IMAGES/pi.gif", gif ) ) );
```

Platform(*data table*, *script*)

Description

Evaluates the specified script in the context of the specified data table.

Returns

The resulting display box for embedding in a display tree.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
win = New Window( "Platform example",
   H List Box(
      Platform(
         dt,
         Bubble Plot(
            X( :weight ),
            Y( :height ),
```

```
            Sizes( :age ),
            Title Position( 0, 0 )
         )
      ),
      Platform(
         dt,
         Bubble Plot(
            X( :weight ),
            Y( :age ),
            Sizes( :height ),
            Title Position( 0, 0 )
         )
      )
   )
);
```

Plot Col Box(*title*, *numbers*)

Description

Returns a display box labeled with the quoted string *title* to graph the *numbers*. The numbers can be either a list or a matrix.

Poly Seg(*x values*, *y values*)

Description

Returns a display seg that represents a polygon with vertices based on the *x values* and *y values*.

Example

```
x = [10, 50, 90];
y = [10, 90, 10];
win = New Window( "Poly Seg Example",
g = Graph Box( Poly Seg( x, y ) ) );
frame = g[FrameBox( 1 )];
seg = (frame << Find Seg( "Poly Seg" ));
```

Popup Box({*command1*, *script1*, *command2*, *script2*, ...})

Description

Creates a red triangle menu. The single argument is an expression yielding a list of an even number of items alternating between the quoted command string and the expression that you want evaluated when the command is selected. If the command is an empty string, a separator line is inserted.

Notes

Pressing ALT and right-clicking the red triangle menu opens a window with check boxes for the commands.

`Radio Box({item, ...}, <script>)`

Description

Constructs a display box to show a set of radio buttons. The items are quoted strings. The optional script is run every time a radio button is selected.

`Range Slider Box(minValue, maxValue, lowVariable, highVariable, script)`

Description

`Range Slider Box()` returns a display box that shows a range slider control that ranges from *minValue* to *maxValue*. As the two sliders' positions change, their values are placed into *lowVariable* and *highVariable*, and the script is run.

Returns

The display box (Range Slider Box).

Arguments

minValue, *maxValue* Numbers that set the minimum and maximum value the slider represents.

lowVariable The variable whose value is set and changed by the lower slider.

highVariable The variable whose value is set and changed by the upper slider.

script Any valid JSL commands that are run as the slider is moved.

`Report(obj)`

Description

Returns the display tree of a platform *obj*. This can also be sent as a message to a platform: `obj<<Report.`

`Scene Box(x size, y size)`

Description

Creates an *x*-sized by *y*-sized scene box for 3-D graphics.

`Scene Display List`

Description

Returns a display list for 3-D graphics.

Example

```
ex = Scene Display List();
```

```
ex << Color( .9, .9, .9 );
ex << Text( center, middle, .3, "Hello World" );
exScene = Scene Box( 600, 600 );
exScene << Background Color( 0 );
exScene << Show Arcball( always );
New Window( "See HelloWorld.jsl in sample scripts", exScene );
exScene << Perspective( 45, .2, 20 );
exScene << Translate( 0.0, 0.0, -4.5 );
exScene << Arcball( ex, 1.5 );
exScene << Update;
```

Script Box(<*script*>, <*language*>, <*width*>, <*height*>)

Description

Constructs an editable box that contains the quoted string *script*. The editable box is a script window and can both be edited and run as JSL.

Optional Arguments

script A quoted string that appears in the script box.

language A quoted string that provides syntax highlighting for the specified language. Valid languages are "JSL", "Text", "SAS", "SAS Output", "SASLog", "R", "MATLAB", "JavaScript", "C", "SQL", "Python", "JSON", and "XML".

width An integer that sets the width of the script box.

height An integer that sets the height of the script box.

Example

```
// JSON
New Window( "JSON",
   Script Box(
      "{\!"a\!":1,\!"b\!":\!"test\!"}",
      "JSON"
   )
);
```

Scroll Box(<Size(*h*,*v*)>, *display box*, ...)

Description

Creates a display box that positions a larger child box using scroll bars.

Returns

A reference to the scroll box object.

Required Argument

display box Any number of display box arguments can be contained in the scroll box.

Optional Argument

Size(*h*,*v*) The *h* and *v* arguments specify the size of the box in pixels.

Notes

You can send a scroll box object a message to set the background color:

```
<<Set Background Color( {R, G, B} | <color> )
```

The `Flexible` argument is deprecated. Use the `Set Stretch` message instead. See "V Scroll Box(<Size(v)>, `display box`)" for an example.

You can set the Boolean flags for horizontal (h) and vertical (v) scrolling to enable (1) or disable (0) the scroll bars. If scrolling is disabled in a given direction, the Scroll Box will behave as a regular container in that direction.

```
<<Set Scrollers (h, v)
```

To return the flags for scrolling, use the following message:

```
<<Get Scrollers
```

To set the horizontal (h) and vertical (v) positions (in pixels) for the scrollers on the scroll bar:

```
<<Set Scroll Position (h,v)
```

To return the flags for scroll position, use the following message:

```
<<Get Scroll Position
```

To return the maximum positions for horizontal and vertical scrolling, use the following message:

```
<<Get Scroll Extents
```

Example

The following example shows a window containing a scroll box with the specified settings.

```
win = New Window( "Example",
   sb = Scroll Box(
      Size( 150, 75 ),
      List Box(
         {"First Item", "Second Item",
         "Third Item", "Fourth Item",
         "Fifth Item"},
         width( 200 ),
         max selected( 2 ),
         nLines( 6 )
      )
   )
);
win << Set Window Size( 300, 200 );
sb << Set Scrollers( 1, 1 ); // enable both scroll bars
// position the scrollers on the scroll bar
sb << Set Scroll Position( 0, 20 );
```

Shape Seg({Path(*path*), ...}, <Row States(*dt*|*dt*,[*rows*]|dt,{{*rows*}, ...}|{*row states*})>)

Description

Returns a display seg with a collection of shapes.

Required Argument

Path Specifies the path with an Nx3 matrix or with a text representation. A path matrix has three columns for x, y, and flags for each point in the path. The flag values are 0 for control, 1 for move, 2 for a line segment, 3 for a cubic Bézier segment, and are negative if the point also closes the path. Path text supports SVG syntax.

Optional Arguments

Row States Specifies a data table reference and optionally rows or the actual row state.

Example

```
win = New Window( "Shape Seg Example",
   Graph Box(
      Shape Seg(
         {Path( [10 10 1, 10 70 0, 70 70 0, 70 10 -3] ),
         Path( "M20,20 C20,60 60,60 60,20 Z" )},
         Row States( {Selected State( 1 ), Color State( "red" )} )
      )
   )
);
```

Sheet Box(<<Hold(*rpt*), *display box*, ...)

Description

Returns a display box that can organize other display boxes vertically or horizontally.

Sheet Panel Box(*title*, *display box*)

Description

Specifies whether the Sheet Box should be horizontal or vertical.

Slider Box(*minValue*, *maxValue*, *variable*, *script*, <Set Width(*n*)>, <Rescale Slider(*min*, *max*)>)

Description

Creates an interactive slider control.

Returns

The display box (Slider Box).

Required Arguments

minValue, *maxValue* Numbers that set the minimum and maximum value the slider represents.

variable The variable whose value is set and changed by the slider box.

script Valid JSL commands that are run as the slider box is moved.

Optional Arguments

Set Width(*n*) Specifies the width of the slider box in pixels.

Rescale Slider(*minValue*, *maxValue*) Resets the maximum and minimum values for the slider box.

Notes

You can send Set Width and Rescale Slider as messages to a slider object. For example:

```
ex = .6;
New Window( "Example", mybox = Slider Box( 0, 1, ex, Show( ex ) ) );
mybox << Set Width( 200 ) << Rescale Slider( 0, 5 );
```

Spacer Box(<Size(*h*, *v*)>, <Color(*color*)>)

Description

Creates a display box that can be used to maintain space between other display boxes, or to fill a cell in a LineUp Box.

Returns

A reference to the display box.

Optional Arguments

Size(*h*, *v*) The *h* and *v* arguments specify the size of the box in pixels.

Color(*color*) Sets the color of the box to the JSL color argument.

Spin Box(*script*)

Description

Returns a display box that shows a button with up and down controls.

Argument

script Invoked with an argument that indicates the direction of the arrow clicked. Negative is down, and positive is up. A value of 1 indicates a single click, while larger values indicate a repeating action.

Example

```
win = New Window( "Example",
   Lineup Box(
      2,
      nb = Number Edit Box( 3 ),
      sb = Spin Box( Function( {value}, nb << Increment( value ) ) )
   )
);
nb << Set Increment( 1 );
```

Splitter Box(<Size(*x*, *y*)>, *display box*, ...)

Description

Returns a display box that can organize other display boxes horizontally or vertically with interactive control of sizes. Child sizes are specified as a proportion of the width or height of the splitter box. The optional *Size* argument is used only for the top-most splitter box. Lower level display boxes are sized like any other child box.

Use `H Splitter Box()` or `V Splitter Box()`.

String Col Box(*title*, {*string*, ...})

Description

Creates a column in the data table containing the quoted *string* items. The column is named after the quoted *title*.

String Col Edit Box(*title*, {*string*, ...})

Description

Creates a column in the data table containing the quoted *string* items. The string boxes are editable. The column is named after the quoted *title*.

Notes

To retrieve the data, use the following message:

```
data = obj << Get;
```

Tab Box(Tab Page Box(Title(*page title 1*), <*options*>, *contents of page 1*), Tab Page Box(Title(*page title 2*), <*options*>, *contents of page 2*), ...);

Description

(Previously called `Tab List Box`.) Creates a tabbed window pane. The arguments are an even number of items alternating between the name of a tab page and the contents of the tab page. The page titles must be quoted.

Arguments

`Tab Page Box` Returns a display box that can be used in a tab box or as a stand-alone container with a title.

- `Title(page title #)` specifies the title of page 1.
- Options include <<`Closeable`(*Boolean*) (specifies whether the box can be closed), <<`Tip`(*string*) (a quoted string that specifies a tooltip), and <<`Icon`(*string*) (a quoted string that specifies the icon).
- *contents of page #* is a quoted string that specifies the text on the tab.

Example

```
New Window( "Example",
```

```
   Tab Box(
      t1 = Tab Page Box( Title( "alpha" ), Panel Box( "panel", Text Box( "text"
   ) ) ),
      t2 = Tab Page Box( Title( "beta" ), Popup Box( {"x", ex = 1, "y", ex = 2}
   ) ),
   )
);
```

Notes

Certain messages that you can send to `Tab Page Box` have been renamed:

- `Set Title` is called `Title`.
- `Set Tip` is called `Tip`.
- `Set Icon` is called `Icon`.
- `Set Closeable` is called `Closeable`.

Tab List Box(*title*, *tabExpr1*, ...)

See "Tab Box(Tab Page Box(Title(page title 1), <options>, contents of page 1), Tab Page Box(Title(page title 2), <options>, contents of page 2), ...);".

Tab Page Box(Title(*string*), <*options*>, *contents*)

Description

Returns a display box that can be used in a tab box or as a stand-alone container with a title.

Required Argument

`Title` A quoted string that specifies the title on the tab.

Optional Messages

`<<Tip(`*string*`)` A quoted string that specifies a tooltip.

`<<Closeable(`*Boolean*`)` Specifies whether the page can be closed.

`<<Icon(`*string*`)` A quoted string that specifies the icon.

`<<Set Font(font name, <size>, <"bold"|"italic"|"underline"|strikeout">, <angle>` Specifies the font properties.

`<<Set Font Name("`*font name*`")` Specifies the name of the font.

`<<Set Font Scale(f)` Specifies the scale factor for the font. The scale factor is applied to the size that is determined by the base font and point size.

`<<Set Font Size(n)` Specifies the font size in pixels.

`<<Set Font Style("plain"|"bold")` Specifies the font style.

Notes

Certain messages that you can send to `Tab Page Box` have been renamed:

- `Set Title` is called `Title`.

- `Set Tip` is called `Tip`.
- `Set Icon` is called `Icon`.
- `Set Closeable` is called `Closeable`.

`Table Box(`*`display box, ...`*`)`

Description

Creates a report table with the *`display boxes`* listed as columns.

`Text Box(`*`text, <arguments>`*`)`

Description

Constructs a box that contains the quoted string *`text`*.

Arguments

`<<Justify Text(`*`position`*`)` Justifies the text left, center, or right as specified in quotes.

`<<Set Wrap(`*`pixels`*`)` Sets the point at which text wraps.

`Text Edit Box(`*`text, <arguments>`*`)`

Description

Constructs an editable box that contains the quoted string *`text`*.

Arguments

`<<Password Style(`*`Boolean`*`)` Displays asterisks in the box rather than the password.

`<<Set Script` Runs the specified script after the text is edited.

`<<Set Width(`*`pixels`*`)` Sets the point at which text wraps.

`This Project()`

Description

Gets the current project when a JSL script is run from that project.

Example

The following example gets the window title of the current project.

```
project = New Project();
project << Save( "$DOCUMENTS/Test Project.jmpprj" );
project << Run Script(
   New Window( "Project Title",
      Text Box(This Project() << Get Window Title())
   );
);
```

Tree Box(<{*rootnodes*}>, <Size(*width*, *height*)>, <MultiSelect(*Boolean*)>)

Description

Constructs a box to show a hierarchical tree composed of tree nodes.

Arguments

{*rootnodes*} Specifies the names for the root nodes created by `Tree Node()` which the box contains.

`Size(`*width*, *height*`)` Specifies the width and height (in pixels) of the box.

`MultiSelect(`*Boolean*`)` Indicates that more than one item in the tree can be selected.

Tree Node(<*data*>)

Description

Creates a node for display in a Tree Box display. Tree Node is used for both parent and child nodes.

Notes

If you send a root node that contains one or more nodes with the `Set Node Select Script` defining a collapse message, then macOS runs the script twice. Windows doesn't run the script. This behavior on macOS doesn't just affect increments. Any script runs twice. It will print to the log twice, create a column twice, try to delete something twice, and so on.

Triangulation(<*dt*>, X(*col1*, *col1*), <Y(*col*)>)

Description

Returns an object containing the Delaunay triangulation of the given point set. The optional Y will be averaged for duplicate points, and all points in the output will be unique.

Examples

```
tri = Triangulation(
  X( [0 0 1 1], [0 1 0 1] ),
  Y( [0 1 2 3] )
);
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
tri = Triangulation( X( :X, :Y ), Y( :POP ) );
```

V Center Box(*display box...*)

Returns a display box that contains the child display box argument. The box is centered in the vertical space defined by the maximum size of that child display box and all of the other siblings of the center box.

V List Box(<Align("Center"|"Right")>, *display box, ...*)

Description

Creates a display box that contains other display boxes and displays them vertically.

Arguments

`Align` Specify right or center alignment of the contents in the list box. The contents are center aligned by default.

`display box` Any number of display box arguments can be contained in the list box.

V Scroll Box(<Size(*v*)>, *display box*)

Description

Returns a display box that places a scroll bar on the bottom and right if the contents are bigger than the size of the scroll box.

Arguments

`Size(`*`v`*`)` The vertical length of the scroll bar.

`display box` Any number of display box arguments can be contained in the scroll box.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
New Window( "stretchable",
   H Splitter Box(
      Size( 400, 200 ),
      Scroll Box(
         Size( 200, 200 ),
         dt <<Run Script( "Distribution" ),
         <<Set Stretch( "Window", "Window" )
      ),
      Scroll Box(
         Size( 200, 200 ),
         dt <<Run Script( "Bivariate" ),
         <<Set Stretch( "Window", "Window" )
      ),
      <<Set Stretch( "Window", "Window" )
   )
);
```

Notes

The flexible argument is deprecated. Use `Set Stretch` instead.

V Sheet Box(<<Hold(*report*), *display box, ...*)

Description

Returns a display box that arranges the display boxes provided by the arguments in a vertical layout. The <<Hold() message tells the sheet to own the report(s) that is excerpted.

V Splitter Box(<Size(*h*,*v*)>, *display box...*, <*arguments*>)

Description

Returns a display box that arranges the display boxes provided by the arguments in a vertical layout (or panel). The splitter enables the user to interactively resize the panel.

Arguments

Size(*v*) The vertical length of the scroll bar.

display box Any number of display box arguments can be contained in the splitter box.

Optional Arguments

For more information about the optional arguments, see "H Splitter Box(<Size(h, v)>, display box, ...)".

Web Browser Box(*url*)

Description

Creates a display box that contains a web page. Available only on Windows in Internet Explorer.

Returns

A reference to the web browser box object.

Argument

url A quoted string containing the URL to the web page to display.

Example

The following example creates a splitter box with the web browser box on the left and the bubble plot on the right.

```
dt = Open( "$SAMPLE_DATA/PopAgeGroup.jmp" );
New Window( "Example",
  H Splitter Box(
     Size( 800, 300 ),
     wb = Web Browser Box( "http://www.jmp.com" ),
     dt << Run Script( "Bubble Plot Region" )
  )
);
wb << Set Stretch( "Window", "Window" ); // auto stretch horizontally and
  vertically
wb << Set Max Size( 10000, 10000 ); // maximum size in pixels
```

Notes

The <a href> target "_blank" opens the web page in a new Internet Explorer window. The <a href> target "_new" opens the web page in the active Internet Explorer tab.

`Window(<string|int>)`

Returns

Either a list of references to all open windows, or a reference to an explicitly named window.

Arguments

string A quoted string containing the name of a specific open window.

int The number of a specific open window.

Notes

If no argument is provided, a list of all open windows is returned.

If the argument (either a window name or number) does not exist, an empty list is returned.

`Wrap List Box(display box, ...)`

Description

Creates a list box that contains other display boxes and displays them horizontally, but wraps them when printing.

Arguments

display box Any number of display box arguments can be contained in the list box.

Expression Functions

`Arg(expr, i)`

`Arg Expr(expr, i)`

Description

Finds the argument numbered by *i* within the given expression.

Returns

The *i*th argument within the expression *expr*.

`Empty()` if that argument does not exist or is not specified.

Arguments

expr An expression.

i An integer that indicates which argument to return.

Notes

`Arg Expr()` is deprecated. Use `Arg()` instead.

`Eval Expr(`*expr*`)`

Description

Evaluates any expressions within *expr*, but leaves the outer expression unevaluated.

Returns

An expression with all the expressions inside *expr* evaluated.

Argument

expr A valid expression.

`Expr(`*expr, i*`)`

Description

Returns the argument unevaluated (expression-quoting).

Returns

The argument, unevaluated.

`Empty()` if that argument does not exist or is not specified.

Argument

expr An expression.

i An integer that indicates which argument to return.

`Extract Expr(`*expr*`, `*pattern*`)`

Description

Find *expr* matching *pattern*.

Returns

A pattern that matches the specified *pattern*.

Arguments

expr An expression.

pattern Any pattern.

`Head(`*exprArg*`)`

`Head Expr(`*exprArg*`)`

Description

Returns the head of the evaluated expression, without its arguments.

Notes

`Head Expr()` is deprecated. Use `Head()` instead.

`Head Name(expr)`

`Head Name Expr(expr)`

Description

Returns the head of the evaluated expression as a quoted string.

Notes

`Head Name Expr()` is deprecated. Use `Head Name()` instead.

`N Arg(exprArg)`

Description

Returns the number of arguments of the evaluated expression head.

`N Arg Expr(exprArg)`

Description

Returns the number of arguments of the expression head.

Notes

`N Arg Expr()` is deprecated. Use `N Arg()` instead.

`Name Expr(x)`

Description

Returns the unevaluated expression of *x* rather than the evaluation of *x*.

File Functions

`Close(<dt|query>, <noSave|Save(path)>)`

Description

Closes a data table, query, or JSON file. If no arguments are specified, the current file is closed. If the file has been changed, it is automatically saved. All dependent windows are also closed (for example, report windows that are based on the data table).

Returns

Void.

Arguments

dt An optional reference to a data table, query, or JSON file.

`nosave|Save(path)` An optional switch to either save the file to the specified quoted path before closing or to close the file without saving it.

`Close All(<Project(`*`title`*`|`*`index`*`|`*`display box`*`|`*`window`*`)>, `*`type`*`, <"Invisible"|"Private">, <NoSave|Save>)`

Description

Closes all open resources of *type*.

Required Argument

type A named argument that defines the type of resources that you want to close. The allowable types are: `Data Tables`, `Reports`, and `Journals`.

Optional Argument

`Project(`*`title`*`|`*`index`*`|`*`display box`*`|`*`window`*`)` Closes the specified project window.

`"Invisible"` Closes all invisible data tables.

`"Private"` Closes all private data tables.

`NoSave or Save` Saves the specified types of windows before closing or to close without saving.

`Close Database Connection(`*`db connection handle`*`)`

Description

Closes a database connection returned from Create Database Connection.

`Close Log(`*`Boolean`*`)`

Description

Closes the log window.

`Convert File Path(`*`path`*`, <"Absolute"|"Relative">, <"POSIX"|"Windows">, <Base(`*`path`*`)>, <`*`Search`*`>)`

Description

Converts a file path according to the arguments.

Returns

The converted path.

Required Argument

path A quoted pathname that can be either Windows or POSIX.

Optional Arguments

`"Absolute"|"Relative"` Specifies whether the returned pathname is in absolute or relative terms. The default value is `Absolute`.

`"POSIX"|"Windows"` Specifies whether the returned pathname is in Windows or POSIX style. The default is `POSIX`.

`Base(`*`path`*`)` Specifies the quoted base pathname, useful if `Relative` is specified. The default is the default directory. See "Set Default Directory(path)".

`Search` Searches through the specified directories for the specified string. In the example below, JMP searches $SAMPLE_DATA/ and $SAMPLE_DATA/Time Series/ for Air.jmp. If Air.jmp is in both directories, the first instance is returned. Without the search option, `Convert File Path()` uses the default directory instead.

Example

```
Set File Search Path(
   {Convert File Path( "$SAMPLE_DATA/" ),
   Convert File Path( "$SAMPLE_DATA/Time Series/" )}
);
Convert File Path( "Air.jmp", Search );
   Get File Search Path() = {"/C:/Program Files/SAS/JMPPRO/16/Samples/Data/",
      "/C:/Program Files/SAS/JMPPRO/16/Samples/Data/Time Series/"};
   Convert File Path("Air.jmp", search) = "/C:/Program
      Files/SAS/JMPPRO/16/Samples/Data/Time Series/Air.jmp";
```

Copy Directory(*from path*, *to path*, <Recursive(*Boolean*)>)

Description

Copies files from one directory to another, optionally copying subdirectories. The directory name is created in the *to path* and should not be part of the *to path* argument.

Returns

Returns 1 if the directory is copied; otherwise, returns 0.

Required Arguments

from path Specifies the directory containing the files to copy to the new directory. *from path* is a quoted string.

to path Specifies the path where the new directory should be created and to which the files are copied. *to path* is a quoted string.

Optional Argument

`Recursive(`*Boolean*`)>` Specifies whether to copy the *from path* subdirectory structure to the *to path*.

Notes

`Copy Directory(`*from path*, *to path*, *Boolean*`)` is deprecated.

Copy File(*from path*, *to path*)

Description

Copies one file to a new file using the same or a different name.

Returns

Returns 1 if the file is copied; otherwise, returns 0.

Arguments

from path Specifies the path and file name to copy to the new file. *from path* is a quoted string.

to path Specifies the path and file name for the new file. *from path* is a quoted string.

`Create Database Connection(`*dataSourceName*`|"Connect Dialog",<DriverPrompt(`*Boolean*`)>);`

Description

Creates a connection to the specified database or prompts the user to provide database log in information.

Returns

A handle to the database connection.

Required Argument

dataSourceName The quoted server connection string that contains information such as the data source name and user name.

Optional Arguments

`"Connect Dialog"` Opens the Select Data Source window, from which the user selects the database.

`Driver Prompt(`*Boolean*`)` Enables the ODBC driver to prompt for the connection information if necessary.

Examples

Specify the data source name, user name, and password:

```
Create Database Connection( "dsn=Books;UID=johnsmith;password=Christmas" );
```

Request that the ODBC driver prompt the user to enter connection information, because the quoted connection string does not specify the password:

```
Create Database Connection( "dsn=Books;UID=johnsmith", Driver Prompt( 1 ) );
```

Enable the user to select the data source, specify `"Connect Dialog"`:

```
Create Database Connection( "Connect Dialog" );
```

`Create Directory(`*path*`)`

Description

Creates a new directory at the specified path location.

Returns

Returns 1 if the directory is created; otherwise, returns 0.

Arguments

path Specifies the quoted path where the new directory should be located.

Creation Date(*path*)

Description

Returns the creation date for the specified file or directory.

Returns

Creation date.

Arguments

path Specifies the quoted directory or path and file name for the query.

Delete Directory(*path*, <Allow Undo(*Boolean*)>)

Description

Deletes the specified directory and its contents and any subdirectories.

Returns

Returns 1 if the directory is deleted; otherwise, returns 0.

Arguments

path Specifies the path and directory for deletion.

Allow Undo(*Boolean*) Allows undo operations, for example, moving to the Recycle Bin or Trash Can.

Delete File(*path*, <Allow Undo(*Boolean*)>)

Description

Deletes the specified file.

Returns

Returns 1 if the file is deleted; otherwise, returns 0.

Required Argument

path Specifies the quoted path and file name for deletion.

Optional Argument

Allow Undo(*Boolean*) Allows undo operations, for example, moving to the Recycle Bin or Trash Can.

Directory Exists(*path*)

Description

Verifies the specified directory exists.

Returns

Returns 1 if the directory exists; otherwise returns 0.

Arguments

path Specifies the quoted path and directory for verification.

File Exists(*path*)

Description

Verifies the specified file name exists at the specified path.

Returns

Returns 1 if the file exists; otherwise returns 0.

Arguments

path Specifies the quoted path and file name for verification.

File Size(*path*)

Description

Determines the size of the file within the specified path.

Arguments

path Specifies the quoted path and file name.

Example

```
File Size( "$SAMPLE_DATA/Big Class.jmp" );
  13142
```

Files In Directory(*path*, <Recursive(*Boolean*)>)

Description

Returns a list of filenames in the *path* given.

Returns

List of filenames. If Recursive(*Boolean*) is not specified, directory names are included in the list.

Required Argument

path A valid quoted pathname.

Optional Argument

Recursive(*Boolean*) A keyword that causes all folders in the path (and all folders that they contain, and so on) to be searched for files.

Notes

Files In Directory(*path*, "Recursive"|*Boolean*) is deprecated.

Find All(<Project(*title*|*index*|*display box*|*window*)>, *type*, <"Invisible"|"Private">)

Description

Finds all open files of the specified type.

Required Argument

type A named argument that defines the type of resources that you want to close. The allowable types are: `Data Tables`, `Reports`, and `Journals`.

Optional Argument

`Project(`*title*`|index|`*display box*`|`*window*`)` Closes the specified project window.

`"Invisible"` Closes all invisible data tables.

`"Private"` Closes all private data tables.

Example

The following example finds all open data tables:

```
exdt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
exdt2 = Open( "$SAMPLE_DATA/Animals.jmp" );
windows = Find All( Data Tables );
For( i = 1, i <= N Items( windows ), i++,
  Write( Char( windows[i] << Get Window Title ) || "\!N" )
);
  Big Class
  Animals
```

Get Default Directory()

Description

Retrieves the user's default directory. This path is used for subsequent relative paths.

If the default directory was set using `Set Default Directory()`, JMP returns the specified path as long as `Get Default Directory()` and `Set Default Directory()` are in the same script.

See "Set Default Directory(path)".

Returns

The absolute pathname as a quoted string.

Arguments

none

Notes

`Get Default Directory()` also gets the path of an active saved scripting window.

Get Excel Worksheets(*absolute path*)

Description

Retrieves a list of worksheets that are in the specified Microsoft Excel workbook. If no worksheets are found, an empty list is returned. The path is a quoted string.

Notes

The function supports .xlsx and Excel 1997 or later workbooks.

Get File Search Path()

Description

Retrieves the current list of directories to search for opening files.

This list is configured using the `Set File Search Path()` function. See "Set File Search Path(path|{list of paths})".

Returns

A list of pathnames as quoted strings.

Get Path Variable(*name*)

Description

Retrieves the value of name, a path variable.

Returns

The absolute pathname as a quoted string.

Argument

`name` A quoted string that contains a path variable. (Examples: SAMPLE_DATA and SAMPLE_SCRIPTS)

Google Sheet Export(*dt*, *email*, *spreadsheet URL|spreadsheet ID|new spreadsheet name*, *sheet name*)

Description

Exports a data table to a Google sheet.

Returns

"1" if the export is successful.

Arguments

`dt` The data table.

`email` The quoted Google email address. @gmail.com is unnecessary.

`spreadsheet URL|ID` The quoted spreadsheet's URL or ID (the string that precedes "spreadsheets/d/").

`new spreadsheet name` The quoted name of the new spreadsheet that you are creating.

`sheet name` The quoted name of the sheet (or tab) within the spreadsheet.

Notes

- JMP features such as formulas and List Check column properties are not supported in Google Sheets.
- If the spreadsheet is empty, look in the JMP log for error messages. On Windows, select **View > Log**. On macOS, select **Window > Log**.

See Also

See *Using JMP* for details about security, country restrictions, and more.

`Google Sheet Import(`*email*`, `*spreadsheet URL*`|`*spreadsheet ID*`, <`*sheet name*`|{`*sheet name*`, `*sheet name*`}, `*Google Sheet Settings*`>)`

Description

Imports sheets from a Google Sheet.

Returns

A data table (or the first data table imported if several sheets are imported at once).

Required Arguments

email The quoted Google email address. @gmail.com is unnecessary.

spreadsheet URL|*ID* The quoted spreadsheet's URL or ID (which precedes "spreadsheets/d/").

Optional Arguments

sheet name The quoted name of the sheet or sheets that you want to import.

Google Sheet Settings The settings that describe how the data is imported.

See Also

See *Using JMP* for details about security, country restrictions, and more.

`Is Directory(`*path*`)`

Description

Returns 1 if the quoted path argument is a directory and 0 otherwise.

`Is Directory Writable(`*path*`)`

Description

Returns 1 if the directory specified in the quoted path is writable and 0 otherwise.

`Is File(`*path*`)`

Description

Returns 1 if the quoted path is a file and 0 otherwise.

`Is File Writable(`*path*`)`

Description

Returns 1 if the file specified in the quoted path is writable and 0 otherwise.

JSON To Data Table(*JSON string*, (<Private(*Boolean*)>|<Invisible(*Boolean*)>), <Guess(Stack(*Boolean*)|"Tall"|"Wide"|"Huge"|"Pandas")>)

Description

Converts a quoted JSON string to a data table.

Returns

A data table reference. The parsing of an empty value, "" string, missing value, or any other invalid value returns an empty data table.

Required Argument

JSON string The quoted JSON string.

Optional Arguments

Private(*Boolean*) Hides the data table completely. Specify this argument if the user doesn't need to interact with the data table.

Invisible(*Boolean*) Hides the data table from view but shows it in the JMP Home Window.

Guess(Stack(*Boolean*)|"Tall"|"Wide"|"Huge"|"Pandas") Stack applies to nodes that repeat within a parent node that is creating rows. By default, extra values are stored in a single table cell separated by commas. If the value is 1, repeating values are stacked in extra rows. Be careful stacking data. It can cause non-obvious data errors.

"Tall" imports the data in a tall data table. Select this option when the JSON file contains many rows. This option is the default setting.

"Wide" imports the data in a wide data table. Select this option when the JSON file contains many columns.

"Huge" imports the data in a tall and wide data table.

"Pandas" imports data in the pandas format.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt = JSON to Data Table(
   "[ { \!"name\!": \!"KATIE\!", \!"age\!": 12, \!"sex\!": \!"F\!",
   \!"height\!": 59, \!"weight\!": 95 }, { \!"name\!": \!"LOUISE\!",
   \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!": 61, \!"weight\!": 123 }, {
   \!"name\!": \!"JANE\!", \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!":
   55, \!"weight\!": 74 } ]"
);
```

Notes

Stacking the data can cause non-obvious data errors. Values that are supposed to be in the same row might not be. Here's an example of stacking data:

```
d =
"\[
{
       "toys": [{
```

```
                "Wheels": 2,
                "Color": "Red"
        },
        {
                "Wheels": 4,
                "Size": "Large"
        }]
}
]\";

JSON to Data Table( d, Guess( Stack( 1 ) ) );
```

Figure 2.1 Example of Stacking Data that Shows Errors

	Wheels	Color	Size
1	2	Red	Large
2	4		

The first toy should be red and have 2 wheels. The second toy should be large and have 4 wheels.

JSON To List(*JSON string*)

Description

Converts quoted JSON string to a JSL list that represents the structure specified by the JSON data. The parsing of an empty value, "" string, missing value, or any other invalid value returns {}.

Example

```
l = JSON To List(
   "[ { \!"name\!": \!"KATIE\!", \!"age\!": 12, \!"sex\!": \!"F\!",
   \!"height\!": 59, \!"weight\!": 95 }, { \!"name\!": \!"LOUISE\!",
   \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!": 61, \!"weight\!": 123 }, {
   \!"name\!": \!"JANE\!", \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!":
   55, \!"weight\!": 74 } ]"
);
Show( l );
```

JSON Literal(*JSON string*)

Description

Returns a valid JSON Boolean or null constant based on the specification of the parameter. The parsing of an empty value, "" string, missing value, or any other invalid value returns Empty().

Last Modification Date(*path*)

Description

Returns the last modification date of the path.

Returns

Last modification date.

Arguments

path The quoted directory or file name.

Load Text File(*path*, <Charset(character set), Force("Throw"|"Alert"|"Silent")>)>, <Line Separator(character)>, <"XMLParse">|<"SASODSXML">|<"JSON">|<BLOB(<arguments>)>)>)

Description

Reads the text file at *path* into a JSL variable.

Returns

A quoted string.

Required Argument

path A quoted pathname that points to a text file. The path can be a URL.

Optional Arguments

`Charset("charater set")` Specifies the character set. Valid character sets are `"Best Guess"`, `"utf-8"`, `"utf-16"`, `"us-ascii"`,`" windows-1252"`, `"x-max-roman"`, `"x-mac-japanese"`, `"shift-jis"`, `"euc-jp"`, `"utf-16be"`, and `"gb2312"`.

`Force("Throw"|"Alert"|"Silent")` Specifies what happens if the character set cannot be detected.

`Line Separator(`*character*`)` Specifies the quoted end-of-line character. For example, `"\!n"` specifies a line feed character. `"\!t"` specifies a tab.

`"XMLParse"` Converts an XML file into JSL.

`"SASODSXML"` Parses the text file as SAS ODS default XML.

`"JSON"` Converts JSON into an expression tree.

`BLOB(<`*arguments*`>)` Returns data from the file as a blob rather than a quoted string. The following optional arguments are for reading parts of the file:

- `ReadOffsetFromBegin(`*n*`)` specifies the zero-based offset to begin reading from the beginning of the file.
- `ReadOffsetFromEnd(`*n*`)` specifies the zero-based offset to begin reading from the end of the file.
- `ReadLength(`*n*`)` specifies the number of bytes to read from the file, either from the beginning of the file or from one of the offset values.

– `Base64Compressed(Boolean)` specifies how the blob is converted to a printable representation. 0, the default and recommended setting, uses JMP's ASCII~HEX representation. 1 means the blob is compressed and converted to base 64 when printed.

`Move Directory(from path, to path)`

Description

Moves a directory and its contents (including subdirectories) from the quoted path to another quoted path.

Returns

Returns 1 if the directory is moved; otherwise returns 0.

Arguments

`from path` Specifies the path and directory for relocation.

`to path` Specifies the destination path and directory.

`Move File(from path, to path)`

Description

Moves a file from the quoted path to another quoted path with the same or different file name.

Returns

Returns 1 if the file is moved. Otherwise, it returns 0.

Arguments

`from path` Specifies the quoted path and file name for relocation.

`to path` Specifies the quoted destination path and file name.

Notes

On Windows, when you move a file to a folder that does not exist, Windows creates the folder and returns 1. On macOS, the folder is not created, and an error is returned.

`Open(path, <arguments>)`

Description

Opens the data table or other JMP file or object created from a file named by the *`path`*. If no path is specified, the Open window appears. Also opens JSON and HDF5 files. See the examples in the JMP Scripting Index in the Help menu for more information about which arguments apply to specific file types.

Required Argument

`path` The quoted path to the file that you want to open.

Optional Arguments

`Add to Recent Files(Boolean)` Determines whether the file is added to the Recent Files list in the Home Window.

`Charset(character set)` The available character set options for importing text files are `"Best Guess"`, `"utf-8"`, `"utf-16"`, `"us-ascii"`,`" windows-1252"`, `"x-max-roman"`, `"x-mac-japanese"`, `"shift-jis"`, `"euc-jp"`, `"utf-16be"`, and `"gb2312"`.

`Column Names Start(n)|Column Names are on Line(n)` Specifies the line number that column names start in the imported text file. If the text file uses returns between cells, column names could be on multiple lines.

`Columns(colName=colType(colWidth),...)` Specifies the columns by name in the text file to import into a data table where:

– `colName`: Specifies the column name used in the imported text file.

– `colType("Character"|"Numeric")`: Specifies whether the specified column contains character or numeric data.

– `colWidth(n)`: Specifies the integer width of the specified column.

`Columns(<arguments>)` For ESRI shapefiles (.shp), this argument and its settings indicate the following:

– `Shape=numeric(n)`: Specifies the column number in the imported ESRI shapefile that contains the shape number.

– `Part=numeric(n)`: Specifies the column number in the imported ESRI shapefile that contains the part number.

– `X=numeric(n)`: Specifies the column number in the imported ESRI shapefile that contains the decimal degree for the longitude (range of ±180°).

– `Y=numeric(n)`: Specifies the column number in the imported ESRI shapefile that contains the decimal degree for the latitude (range of ±90°).

`"Column Names Only"` Returns a list of column names.

`Compress Allow List Check(Boolean)` Specifies that JMP can compress data table created from the imported text file.

`Compress Character Columns(Boolean)` Specifies that JMP should compress data table columns that contain character data from the imported text file.

`Compress Numeric Columns(Boolean)` Specifies that JMP should compress data table columns that contain numeric data from the imported text file.

`Concatenate Worksheets(Boolean)` Specifies that JMP should combine the imported Excel worksheets into one data table.

`Create Concatenation Column(Boolean)` Specifies that JMP should combine columns from an imported Excel file into one column.

`Data Starts(n)|Data Starts on Line(n)` Specifies the line number where data starts in the imported text file.

`Debug JSL(Boolean)` Opens the specified JSL script in the Debugger instead of opening it.

`End Of Field("Tab"|"Space"|"Comma"|"Semicolon"|"Other"|"None")` Specifies the quoted character used to delimit the end of a field in the imported text file. To specify multiple characters, separate each character designation by a comma. If you use `"Other"`, designate the delimiter with `EOF Other()` argument.

`End Of Line("CRLF"|"CR"|"LF"|"Semicolon"|"Other")` Specifies the quoted character used to delimit the end of a line in the imported text file. To specify multiple characters, separate each character designation by a comma. If you use `"Other"`, designate the delimiter with `EOL Other()` argument.

`EOF Other(char)` If the imported text file uses an end-of-field character other than the one specified by `End of Field()`, this argument specifies the character used.

`EOL Other(char)` If the imported text file uses an end of line character other than the one specified by `End of Line()`, this argument specifies the character used.

`"Excel Wizard"` Opens Microsoft Excel worksheets in the Excel Import Wizard. If you omit this argument, the worksheets open directly as a data table.

`File Type(type)` An optional quoted string that specifies the type of file that you are opening (for example, "text", "journal", "sas", "script", "png", and "jmp"). This can be useful if your file does not have a file extension, the file extension of the file does not match the contents of the file, or you want to import a JSL BLOB. If you do not specify this string, the file opens in the default program for the file extension.

Note: The path argument should be used for a zip archive. The extension (.zip) is not required. See "Zip Archives" for the messages that you can send to a zip archive. The basic functionality is to get a list of files in the zip archive, to read a file in the zip archive into either a quoted string or a blob, and to write files into the zip archive. Note that reading a zip archive temporarily puts the contents into memory. Reading very large zip archives can cause errors.

`"Force Refresh"` Closes the specified JMP (.jrn, .jsl, .jrp, or .jmpappsource) file without saving and tries to reopen the file from disk. This argument deletes any changes made since the last time the file was opened.

`HTML Table(n, <ColumnNames(n)>, <DataStarts(n)>)` To import a table from an HTML web page, use the URL as the filepath. The optional *n* argument specifies which table number, *n*, on the web page to open. If you omit the value, only the first table on the page is imported. The optional `ColumnNames(n)` specifies the row that contains column names. The optional `DataStarts(n)` specifies the row on which the data begins.

Tip: If the table you are importing contains images, they are first imported as text. To load the images in your JMP data table, run the automatically generated table script named Load Pictures. A new expression column containing the images is created. See *Using JMP* for more information about expression columns.

`Ignore Columns(col1, ...)` Specifies the quoted column names in the JMP data table or other JMP file that should not be included in the data table.

`"Invisible"` Opens the file as invisible. This quoted keyword applies to these files: data table, JMP file, external, text, Excel, SAS, ESRI shapefile, and HTML. The data table appears only in the JMP Home Window and the Window menu.

`Labels(Boolean)` Specifies the imported text file contains labels or column headers as the first line in the file. The default value is `True`.

`Lines to Read(n)` Specifies the number of lines in the text file to import. JMP starts counting lines after column names are read.

`Number of Columns(n)` Specifies the number of columns contained in the imported text file.

`Run JSL(Boolean)` Runs the specified JSL file instead of opening it. Include a Boolean argument or an expression that contains a Boolean value. If the script begins with `//!`, which automatically runs the script, include the Boolean value `(0)` to open the script instead.

`Password(password)` Specifies the quoted password for a password-protected SAS file to avoid entering it manually. The password is not encrypted. (Password-protected Microsoft Excel files cannot be imported.)

`"PDF Wizard"` Opens a PDF file in the PDF Import Wizard, where you can control how data is imported.

`"Private"` Opens the table as invisible and without showing it in the JMP Home Window or Window menu. For example, you might create a private data table to hold temporary data that the user does not need to see. This quoted keyword applies to the following files: data table, JMP file, external, text, Excel, SAS, ESRI shapefile, or HTML. Creating a private data table speeds the process of getting to the data; it does not save the computer from allocating the memory necessary to hold the data table data.

`Quarantine Action("Allow Scripts"|"Block Scripts"|"Do Not Open"|"Show Dialog")` Specifies whether scripts run when you open downloaded data tables. Also provides an option to display a window that prompts users to examine or open the data table. If they click Examine, the scripts are disabled.

`Scan Whole File(Boolean)` Specifies how long JMP scans the text file to determine data types for the columns. This is a Boolean value. The default value is true; the entire file is scanned until the data type is determined. To import large files, consider setting the value to false, which scans the file for five seconds.

`Select Columns(col1, ...)` Specifies the quoted column names in the JMP data table or other JMP file that should be included in the data table.

"Strip Quotes"|Strip Enclosing Quotes(*Boolean*) If the fields in the text file are quoted, setting this to True removes the quotes, and setting it to False does not remove the quotes. The default value is True.

Table Contains Column Headers(*Boolean*) Indicates the imported text file contains labels or column headers as the first line in the file. The default value is True.

"Text Wizard" Opens the text file in the text import window, where you can select import options. Otherwise, the Text Data Files options in the JMP preferences apply, and the text file is automatically imported as a data table.

Treat Empty Columns as Numeric(*Boolean*) Specifies that JMP should import text file columns of missing data as numeric rather than character. Possible missing value indicators are a period, a Unicode dot, NaN, or a blank quoted string. The default value is False.

Use Apostrophe as Quotation Mark(*Boolean*) Declares apostrophes as quotation marks in importing text files. This option is not recommended unless your data comes from a nonstandard source that places apostrophes around data fields rather than quotation marks. The default value is False.

Use Labels for Var Names(*Boolean*) For SAS data sets, this option specifies to use SAS labels as JMP columns names. The default value is False.

Use for All Sheets(*Boolean*) Indicates that JMP should use the Worksheets settings for all worksheets in the Excel file to be opened as a data table.

Worksheet Settings(*Boolean*, <*options*>) specifies options for importing an Excel file into a JMP data table. Available options are:

- Has Column Headers(*Boolean*) specifies that the Excel file has column headers in the first row.
- Number of Rows in Headers(*n*) specifies the number of rows in the Excel file used as column headers.
- Headers Start on Row(*n*) specifies the row number in the Excel file where the column headers begin. Default is row 1.
- Data Starts on Row(*n*) specifies the row number in the Excel file where the data begins.
- Data Starts on Column(*n*) specifies the column number in the Excel file where the data begins.
- Data Ends on Row(*n*) specifies the row number in the Excel file where data ends.
- Data Ends on Column(*n*) specifies the column number in the Excel file where the data ends.
- Replicated Spanned Rows(*Boolean*) specifies that the Excel file contains spanned columns should be imported into JMP as spanned columns.
- Suppress Hidden Rows(*Boolean*) indicates that JMP should not import rows hidden in the Excel file.

– `Suppress Hidden Columns(`*`Boolean`*`)` indicates that JMP should not import columns hidden in the Excel file.

– `Treat as Hierarchy(`*`Boolean`*`)` indicates that JMP should treat multiple column headers (merged cells) as hierarchies when importing an Excel file. If `True`, the Excel file opens with the merged columns stacked (Tables > Stacked).

`Worksheets("Sheet Name"|{"Sheet Name", "Sheet Name",...}|`*`n`*`)` Opens the specified Excel file worksheet by name, all worksheets in a list of names, or the indexed number of the worksheet. If the worksheets are not specified, all worksheets in the spreadsheet open as separate data tables.

Note: You can import .xls worksheets from a web site by specifying worksheets arguments. .xlsx worksheets cannot be imported from a web site using `Open()`.

`"XML Wizard"` Opens XML files in the XML Import Wizard. If you omit this argument, the XML file opens as a data table.

```
Year Rule | Two Digit Year Rule
  ("1900-1999"|"1910-2009"|"1920-2019"|"1930-2029"|"1940-2039"|
  "1950-2049"|"1960-2059"|"1970-2069"|"1980-2079"|"1990-2089"|"2000-2099")
```

Indicates the year format used in the imported text file. For example, if the earliest date is 1979, use "1970-2069".

`Open Database(dataSourceName|"Connect Dialog", "SELECT ..."|"SQLFILE..."|tableName, <"Invisible"|"Private">, <outputTableName>)`

Description

Opens the database indicated by *`dataSourceName`* (or opened by the user) with the *`SELECT`* , *`SQLFILE`*, or *`tableName`* arguments.

Returns

A data table named *`outputTableName`*.

Required Argument

`dataSourceName``|"Connect Dialog"` dataSourceName specifies the name of the data source. `"Connect Dialog"` shows a window from which the user selects the data source.

Optional Arguments

`"Invisible"` Creates an invisible data table that hides the table from view but lists it in the JMP Home Window and Window menu.

`"Private"` Hides the data table completely. Creating a private data table speeds the process of getting to the data; it does not save the computer from allocating the memory necessary to hold the data table data.

`outputTableName` The name of the data table in JMP.

Example

```
Open Database(
  "DSN=dBASE Files;DBQ=C:/Program Files/SAS/JMPPRO/16/Samples/Import Data/;",
```

```
    // SQL statement
    "SELECT HEIGHT, WEIGHT FROM Bigclass", // selected columns
    "hw" // name of the output data table
);
```

Parse JSON(JSON string)

Description

Converts JSON string into an associative array or list based on the structure of the JSON data. Convert the result to a list if the parsed JSON object contains more than one member. The parsing of an empty value, "" quoted string, missing value, or any other invalid value returns Empty().

Example

The following example converts JSON into a list:

```
j = Parse JSON(
    "[ { \!"name\!": \!"KATIE\!", \!"age\!": 12, \!"sex\!": \!"F\!",
    \!"height\!": 59, \!"weight\!": 95 }, { \!"name\!": \!"LOUISE\!",
    \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!": 61, \!"weight\!": 123 }, {
    \!"name\!": \!"JANE\!", \!"age\!": 12, \!"sex\!": \!"F\!", \!"height\!":
    55, \!"weight\!": 74 } ]"
);
Show( j );
    j = {["age" => 12, "height" => 59, "name" => "KATIE", "sex" => "F", "weight"
       => 95], ["age" => 12, "height" => 61, "name" => "LOUISE", "sex" => "F",
       "weight" => 123], ["age" => 12, "height" => 55, "name" => "JANE", "sex"
       => "F", "weight" => 74]};
```

Pick Directory(<"Prompt">, <*path*>, <Show Files(*Boolean*)>)

Description

Prompts the user for a directory, returning the directory path as a quoted string.

Returns

The path for the directory that the user selects.

Optional Arguments

"Prompt" The quoted string appears at the top of the Browse window (Windows) or the Finder window (macOS).

path The quoted string specifies the initial directory that appears in the Pick Directory window.

Show Files(*Boolean*) 1 shows files in the Pick Directory window. 0 hide the files. The default is 0.

Pick File(<"Prompt">, <*initial directory*>, <{*filter list*}>, <*first filter*>, <Save Flag(*Boolean*)>, <default file>), <"Multiple">)

Description

Prompts the user to select one or more files in the Open window.

Returns

The path of the file that the user selects.

Optional Arguments

`"Prompt"` A quoted string. If provided, that string appears at the top of the Open window.

`initial directory` A quoted string that is a valid filepath to a folder. If provided, it specifies where the Open window begins. If not provided, or if it's an empty string, the JMP Default Directory is used.

`filter list` A list of quoted strings that define the filetypes to show in the Open window. See the following example for syntax.

`first filter` An integer that specifies which of the filters in the filter list to use initially. If you use an integer that is too large or small for the list (for example, 4 for a list of 3), the first filter in the list is used.

`Save Flag(`*`Boolean`*`)` A Boolean that specifies whether the Open window or Save window is used. 0 lets the user select a file to open in JMP. 1 lets the user save a new, empty file of the selected type in the selected folder. The default value is 0.

`default file` The quoted name of the file that appears in the window by default.

`"Multiple"` The quoted name that lets the user select multiple files if the `Save Flag` value is 0.

Notes

Although all arguments are optional, they are also positional. For example, you cannot specify a filter list without also specifying the caption and the initial directory.

The buffer size in the computer's physical memory affects the number of files the user can open.

Example

The following script assigns *Select JMP File* as the window title; shows the JMP Samples/Data directory; shows *JMP Files* and *All Files* in the File name list and selects *JMP Files*; displays the Open window; and shows the sample data file name Hollywood Movies.jmp.

```
Pick File(
  "Select JMP File",
  "$SAMPLE_DATA",
  {"JMP Files|jmp;jsl;jrn", "All Files|*"},
  1,
  0,
  "Hollywood Movies.jmp"
```

```
);
```

Rename Directory(*old path name*, *new directory name*)

Description

Renames a directory without moving or copying it.

Returns

Returns 1 if the directory is renamed; otherwise, returns 0.

Arguments

old path name A quoted string that specifies the path and old directory name.

new directory name A quoted string that specifies the new directory name.

Notes

When you specify the *new directory Name*, include only the directory name, not the entire path.

Rename File(*old path name*, *new file name*)

Description

Renames a file without moving or copying it.

Returns

Returns 1 if the file is renamed; otherwise, returns 0.

Arguments

old path name Specifies the path and old file name.

new file name Specifies the new file name.

Notes

When you specify the *new name*, include only the file name, not the entire path.

Save Text File(*path*, *text*|BLOB, <Mode("Replace"|"Append")>)

Description

Returns the path name of the created file. *text* is quoted string.

Set Default Directory(*path*)

Description

Sets the default directory, which is used for resolving relative paths.

See Also

"Get Default Directory()"

Set File Search Path(*path*|{*list of paths*})

Description

Sets the current list of directories to search for opening files. Using {"."} as the path configures JMP to use the current directory.

Example

```
Set File Search Path( {"C:/JMP/13/source", "C:/Program
  Files/SAS/JMPPRO/16/Samples"} );
```

See Also

"Get File Search Path()".

Set Path Variable(*name*, <*path*>)

Description

Sets the path stored in the variable.

Required Argument

name The quoted name of the variable.

Optional Argument

path The path.

TripleS Import(*path*, <*arguments*>)

Description

Imports the specified Triple-S Survey (SSS) file. The SSS format consists of a pair of files: .xml or .sss, and a .csv, .dat, or .asc file. Both sets of files must have the same root name and be in the same folder.

Required Argument

path A quoted string that contains the full path to the .xml or .sss file.

Optional Arguments

"Invisible" A quoted keyword that hides the table from view. The data table appears only in the JMP Home Window and the Window menu. Hidden data tables remain in memory until they are explicitly closed, reducing the amount of memory that is available to JMP. To explicitly close the hidden data table, call Close(*dt*), where *dt* is the data table reference returned by TripleS Import.

Use Labels for Imported Column Names(*Boolean*) Converts the label names to column headings. The default value is 1.

Example

```
dt = TripleS Import( "C:/Data/airlines.sss", "Invisible", Use Labels for
  Imported Column Names( 0 ) );
```

Financial Functions

Double Declining Balance(*cost*, *salvage*, *life*, *period*, <*factor*>)

Description

Returns the depreciation of an asset for a specified period of time. The function uses the double-declining balance method or some other depreciation factor.

Required Arguments

cost The initial cost.

salvage The value at the end of the depreciation.

life The number of periods in the depreciation cycle.

period The length of the period, in the same units as *life*.

Optional Argument

factor The rate at which the balance declines. The default value is 2.

Notes

This function is equivalent to the Excel function DDB.

Future Value(*rate*, *nper*, *pmt*, <*pv*>, <*Type(Boolean)*>)

Description

Returns the future value of an investment that is based on periodic, constant payments and a constant interest rate.

Required Arguments

rate The interest rate.

nper The number of periods.

pmt The constant payment.

Optional Arguments

pv The present value. The default value is 0.

Type(*Boolean*) 0 specifies end-of-period payments. 1 specifies beginning-of-period payments. The default value is 0.

Notes

This function is equivalent to the Excel function FV.

Interest Payment(*rate*, *per*, *nper*, *pv*, <*fv*>, <*Type(Boolean)*>)

Description

Returns the interest payment for a given period for an investment that is based on periodic, constant payments and a constant interest rate.

Required Arguments

rate The interest rate.

per The period for which you want the interest.

nper The total number of periods.

pv The present value.

Optional Arguments

fv The future value. The default value is 0.

Type(Boolean) 0 specifies end-of-period payments, and 1 specifies beginning-of-period payments. The default value is 0.

Notes

This function is equivalent to the Excel function IPMT.

Interest Rate(*nper*, *pmt*, *pv*, <*fv*>, <*Type(Boolean)*>, <*guess*>)

Description

Returns the interest rate per period of an annuity.

Required Arguments

nper The total number of periods.

pmt The constant payment.

pv The present value.

Optional Arguments

fv The future value. The default value is 0.

Type(Boolean) 0 specifies end-of-period payments, and 1 specifies beginning-of-period payments. The default value is 0.

guess Your estimate of the rate The default value is 0.1 (10%).

Notes

This function is equivalent to the Excel function RATE.

Internal Rate of Return(*values*, <*guess*>)

Internal Rate of Return(*guess*, *value1*, *value2*, ...)

Description

Returns the internal rate of return for a series of cash flows in the *values* argument.

Arguments

values A one-dimensional matrix of values. If the second form of the function is used, list each value separately.

guess What you think is near the result. The default value is 0.1 (10%). This argument is required if you specify values. It is required if you specify individual values.

Notes

This function is equivalent to the Excel function IRR.

`Modified Internal Rate of Return(`*values*`, `*finance rate*`, `*reinvest rate*`)`

`Modified Internal Rate of Return(`*finance rate*`, `*reinvest rate*`, `*value1*`, `*value2*`, ...)`

Description

Returns the modified internal rate of return for a series of periodic cash flows. The cost of investment and the interest received on reinvested cash is included.

Arguments

values A one-dimensional matrix of values. If the second form of the function is used, list each value separately.

finance rate The interest rate that you pay on the money in the cash flows.

reinvest rate The interest rate that you receive on the cash flows when you reinvest them.

Notes

This function is equivalent to the Excel function MIRR.

`Net Present Value(`*rate*`, `*values*`)`

`Net Present Value(`*rate*`, `*value1*`, `*value2*`, ...)`

Description

Returns the net present value of an investment by using a discount rate and a series of future payments (negative values) and income (positive values).

Arguments

rate The discount rate.

values A one-dimensional matrix of values. If the second form of the function is used, list each value separately.

Notes

This function is equivalent to the Excel function NPV.

`Number of Periods(`*rate*`, `*pmt*`, `*pv*`, <`*fv*`>, <`*type(0|1)*`>)`

Description

Returns the number of periods for an investment that is based on periodic, constant payments and a constant interest rate.

Required Arguments

rate The interest rate.

pmt The constant payment.

pv The present value.

Optional Arguments

fv The future value. The default value is 0.

type(0|1) 0 specifies end-of-period payments, and 1 specifies beginning-of-period payments. The default value is 0.

Notes

This function is equivalent to the Excel function NPER.

`Payment(`*rate*`, `*nper*`, `*pv*`, <`*fv*`>, <`*type(0|1)*`>)`

Description

Returns the payment for a loan that is based on constant payments and a constant interest rate.

Required Arguments

rate The interest rate.

nper The total number of periods.

pv The present value.

Optional Arguments

fv The future value. The default value is 0.

type(0|1) 0 specifies end-of-period payments, and 1 specifies beginning-of-period payments. The default value is 0.

Notes

This function is equivalent to the Excel function PMT.

`Present Value(`*rate*`, `*nper*`, `*pmt*`, <`*fv*`>, <`*type(0|1)*`>)`

Description

Returns the present value of an investment.

Arguments

rate The interest rate per period.

nper The total number of periods.

pmt The constant payment.

fv The future value. The default value is 0.

type(0|1) 0 specifies end-of-period payments, and 1 specifies beginning-of-period payments. The default value is 0.

Notes

This function is equivalent to the Excel function PV.

Principal Payment(*rate*, *per*, *nper*, *pv*, <*fv*>, <*type(0|1)*>)

Description

Returns the payment on the principal for a given period for an investment that is based on periodic, constant payments and a constant interest rate.

Required Arguments

rate The interest rate per period.

per The period for which you want the interest.

nper The total number of periods.

pv The present value.

Optional Arguments

fv The future value. The default value is 0.

type(0|1) An optional switch. 0 specifies end-of-period payments, and 1 specifies beginning-of-period payments. The default value is 0.

Notes

This function is equivalent to the Excel function PPMT.

Straight Line Depreciation(*cost*, *salvage*, *life*)

Description

Returns the straight-line depreciation of an asset for one period.

Arguments

cost The initial cost of the asset.

salvage The value at the end of the depreciation.

life The number of periods in the depreciation cycle.

Notes

This function is equivalent to the Excel function SLN.

Sum Of Years Digits Depreciation(*cost*, *salvage*, *life*, *per*)

Description

Returns the sum-of-years' digits depreciation of an asset for a specified period.

Arguments

cost The initial cost of the asset.

salvage The value at the end of the depreciation.

life The number of periods in the depreciation cycle.

per The length of the period, in the same units as life.

Notes

This function is equivalent to the Excel function SYD.

Graphics Functions

Add Color Theme({*name*, <*flags*>, {*color*}, <{*position*}>)

Description

Creates a custom color theme that you can apply to components such as markers, data table rows, and treemaps. Add the color theme to the JMP Preferences by including `Add Color Theme(...)` inside `Preferences()`.

Returns

Null.

Required Arguments

`name` The quoted name of the color theme.

`color` Lists of RGB values. These values define the blocks in categorical color themes and the gradients in continuous color themes. Each list of RGB values corresponds to a slider in the preferences Color Themes window.

`position` An optional list of numbers between 0 and 1 with one position per color. Each position corresponds to a slider in the preferences Color Themes window. If you omit the position, the sliders are evenly spaced.

Optional Argument

`flags` A flag for the Continuous or Categorical color theme list and category of color.

```
Continuous, <Continuous>, Sequential
Continuous, <Continuous>, Diverging
Continuous, <Continuous>, Chromatic
Categorical, <Continuous>, Sequential
Categorical, <Continuous>, Diverging
Categorical, Qualitative
Categorical, <Continuous>, Chromatic
```

With the default JMP color themes, Sequential colors transition from left to right or right to left. Diverging colors are lighter in the middle. Chromatic colors consist of blocks or gradients of bright color. All categories except for Qualitative can be both continuous and categorical.

If you omit the flag, the color is shown in the Continuous, Sequential and Categorical, Sequential categories.

Examples

The following example creates a continuous color theme named Blue to Purple. The color is in the Diverging category. RGB values are defined in the four lists.

```
Add Color Theme(
  {"Blue to Purple", {"Continuous", "Diverging"}, {{0, 0, 255},
  {57, 108, 244}, "white", {128, 0, 100}}} );
```

Notes

Any style except for Qualitative can be Continuous and Categorical at the same time. For example, the Cool to Warm Diverging theme is in the Continuous and Categorical theme lists. In JMP, select **Preferences > Graphs** to see examples.

To delete a color theme, use `Remove Color Theme()`.

See Also

"Remove Color Theme(name|{name, <flags>, {color, ...}, <{position, ...}>}>)"

`Arc(x1, y1, x2, y2, startangle, endangle)`

Description

Inscribes an arc in the rectangle described by the arguments.

Returns

Null.

Arguments

x1, *y1* The point at the top left of the rectangle

x2, *y2* The point at the bottom right of the rectangle

startangle, *endangle* The starting and ending angle in degrees, where 0 degrees is 12 o'clock and the arc or slice is drawn clockwise from *startangle* to *endangle*.

`Arrow(<pixellength>, {x1, y1}, {x2, y2})`

Description

Draws an arrow from the first point to the second point. The optional first argument specifies the length of the arrow's head lines (in pixels).

Returns

Null.

Required Arguments

{*x1*, *y1*}, {*x2*, *y2*} Two lists of two numbers that each specify a point in the graph.

Optional Argument

pixellength The length of the arrowhead in pixels.

Notes

The two points can also be enclosed in square brackets: `Arrow(<pixellength>, [x1, x2], [y1, y2])`.

`Back Color(name)`

Description

Sets the color used for filling the graph's background.

Returns

Null.

Argument

`name` A quoted color name or a color index (such as `"red"` or `"3"` for the color red).

`Char To Path(path)`

Description

Converts a path specification from a quoted string to a matrix.

Returns

A matrix.

Arguments

`path` A quoted string that contains the path specification.

`Circle({x, y}, radius|PixelRadius(n), <...>, <"Fill">)`

Description

Draws a circle centered at {x, y} with the specified radius.

Returns

Null.

Required Arguments

`{x, y}` A number that describes a point in the graph

`radius` The length of the circle's radius in relation to the vertical axis. If the vertical axis is resized, the circle is also resized.

`PixelRadius(n)` The length of the circle's radius in pixels. If the vertical axis is resized, the circle is *not* resized.

Optional Argument

`"Fill"` A positional argument indicating that all circles defined in the function are filled with the current fill color. If `"Fill"` is omitted, the circle is empty.

Notes

The center point and the radius can be placed in any order. You can also add additional center point and radius arguments and draw more than one circle in one statement. One point and several radii results in a bull's-eye. Adding another point still draws all previous circles, and then adds an additional circle with the last radius specified. This means that this code:

```
Graph Box(circle({20, 30}, 5, {50, 50}, 15))
```

results in three circles, not two. First, a circle with radius 5 is drawn at 20, 30. Second, a circle with radius 5 is drawn at 50, 50. Third, a circle with radius 15 is drawn at 50, 50.

Color To HLS(*color*)

Description

Converts the *color* argument (including any JMP color) to a list of HLS values.

Returns

A list of the hue, lightness, and saturation components of *color*. The values range between 0 and 1.

Argument

color A number from the JMP color index.

Example

The output from ColorToHLS() can either be assigned to a single list variable or to a list of three scalar variables:

```
hls = Color To HLS( 8 );
{h, l, s} = Color To HLS( 8 );
Show( hls, h, l, s );
   hls = {0.778005464480874, 0.509803921568627, 0.976};
   h = 0.778005464480874;
   l = 0.509803921568627;
   s = 0.976;
```

Color To RGB(*color*)

Description

Converts the *color* argument (including any JMP color) to a list of RGB values.

Returns

A list of the red, green, and blue components of *color*. The values range between 0 and 1.

Argument

color A number from the JMP color index.

Example

The output from ColorToRGB() can either be assigned to a single list variable or to a list of three scalar variables:

```
rgb = Color To RGB( 8 );
{r, g, b} = Color To RGB( 8 );
Show( rgb, r, g, b );
   rgb = {0.670588235294118, 0.0313725490196078, 0.988235294117647};
   r = 0.670588235294118;
   g = 0.0313725490196078;
   b = 0.988235294117647;
```

Contour(*xVector*, *yVector*, *zGridMatrix*, *zContour*, <*messages*>)

Description

Draws contours given a grid of values.

Returns

None.

Required Arguments

xVector The *n* values that describe *zGridMatrix*.

yVector The *m* values that describe *zGridMatrix*.

zGridMatrix An *nxm* matrix of values on some surface.

zContour The values for the contour lines.

Optional Messages

<<zColor(color, option) The colors to use for the contour lines.

<<"Fill"|"Fill Between"|"Fill Below"|"Fill Above" Specifies where the contour fill appears.

<<Transparency(vector) Specifies the transparency level of the fill in a vector.

Contour Function(*zExpr*, *xName*, *yName*, (*z*|*zMatrix*), < <<xGrid(*min*, *max*, *incr*)>, < <<yGrid(*min*, *max*, *incr*)>, < <<zColor(*color*, *option*)>, < <<*zLabeled*>, < <<"Filled">, < <<"FillBetween">, < <<"Ternary">, < <<Transparency(*number*|*matrix*))

Description

Draws sets of contour lines of the expression, a function of the two symbols. The *z* argument can be a single value or an index or matrix of values.

Returns

None.

Arguments

zExpr Any expression (for example, Sine(y)+Cosine(x)).

xName, yName The values to use in the expression.

z|zMatrix A z-value or a matrix of z-values.

Optional Messages

<<xGrid, <<yGrid Defines a box, beyond which the contour lines are not drawn.

<<zColor Defines the color in which to draw the contour lines. The argument can be either a scalar or a matrix but must evaluate to numeric.

<<zLabeled Labels the contours.

<<"Filled" Fills the contour levels using the current fill color.

<<"FillBetween" Fills only between adjacent contours using the current fill color. For nz contours specified, this option fills nz-1 regions for the intervals between the nz values. Using this option is recommended over using the <<Filled option.

<<"Ternary" Clips lines to be within the ternary coordinate system inside ternary plots.

<<Transparency(*number*|*matrix*) Sets the transparency level of the fill. A vector of numbers between 0 and 1 are sequenced through and cycled for the z contours. This option should be used only in conjunction with the <<FillBetween option.

Drag Line(*xMatrix*, *yMatrix*, <*dragScript*>, <*mouseUpScript*>)

Description

Draws line segments between draggable vertices at the coordinates given by the matrix arguments.

Returns

None.

Required Arguments

xMatrix A matrix of *x*-coordinates.

yMatrix A matrix of *y*-coordinates.

Optional Arguments

dragScript Any valid JSL script. The script is run at drag.

mouseUpScript Any valid JSL script. The script is run at mouseup.

Drag Marker(*xMatrix*, *yMatrix*, <*dragScript*>, <*mouseUpScript*>)

Description

Draws draggable markers at the coordinates given by the matrix arguments.

Returns

None.

Arguments

xMatrix A matrix of *x*-coordinates.

yMatrix A matrix of *y*-coordinates.

dragScript Any valid JSL script. The script is run at drag.

mouseUpScript Any valid JSL script. The script is run at mouseup.

Drag Polygon(*xMatrix*, *yMatrix*, <*dragScript*>, <*mouseupScript*>)

Description

Draws a filled polygon with draggable vertices at the coordinates given by the matrix arguments.

Returns

None.

Required Arguments

xMatrix A matrix of *x*-coordinates.

yMatrix A matrix of *y*-coordinates.

Optional Arguments

dragScript Any valid JSL script. The script is run at drag.

mouseUpScript Any valid JSL script. The script is run at mouseup.

Drag Rect(*xMatrix*, *yMatrix*, <*dragScript*>, <*mouseupScript*>)

Description

Draws a filled rectangle with draggable vertices at the first two coordinates given by the matrix arguments.

Returns

None.

Required Arguments

xMatrix A matrix of two *x*-coordinates.

yMatrix A matrix of two *y*-coordinates.

Optional Arguments

dragScript Any valid JSL script. The script is run at drag.

mouseUpScript Any valid JSL script. The script is run at mouseup.

Notes

xMatrix and *yMatrix* should each contain exactly two values. The resulting coordinate pairs should follow the rules for drawing a `Rect()`. The first point (given by the first value in *xMatrix* and the first value in *yMatrix*) must describe the top, left point in the rectangle. The second point (given by the second value in *xMatrix* and the second value in *yMatrix*) must describe the bottom, right point in the rectangle.

Drag Text(*xMatrix*, *yMatrix*, *string*, <*dragScript*>, <*mouseUpScript*>)

Description

Draws the *string* (or all the items if a list is specified) at the coordinates given by the matrix arguments.

Returns

None.

Arguments

xMatrix A matrix of *x*-coordinates.

yMatrix A matrix of *y*-coordinates.

string A quoted string to be drawn in the graph.

dragScript Any valid JSL script. The script is run at drag.

mouseUpScript Any valid JSL script. The script is run at mouseup.

Fill Color(*name*|*color*|*rgbList*)

Description

Sets the color used for filling solid areas.

Returns

None.

Arguments

name|color|rgbList The quoted color name, index, or list of RGB values.

`Fill Pattern(`*name|mask|image*`)`

Description

Sets the pattern for filled areas.

Arguments

name|mask|image The quoted color name, matrix of values between 0 and 1, or image path.

`Get Color Theme Detail(`*name*`)`

Description

Returns a script for the specified color theme.

Example

The following example returns the script for the JMP default color theme:

```
Get Color Theme Details( "JMP Default" );
  {"JMP Default", 9221, {{213, 72, 87}, {57, 177, 67}, {64, 111, 223}...}}
```

`Get Color Theme Names(<`*kind*`>)`

Description

Returns a list of all color theme names or color themes of the specified kind. The kinds include `"Continuous"`, `"Categorical"`, `"Sequential"`, `"Diverging"`, `"Qualitative"`, or `"Chromatic"`.

Example

The following example returns all color themes:

```
Get Color Theme Names();
  {"Green to Black to Red", "Green to White to Red", "White to Black"...}
```

The following example returns the diverging color themes:

```
Get Color Theme Names( "diverging" );
  {"Green to Black to Red", "Green to White to Red", "Blue to Gray to Red"...}
```

`Gradient Function(`*zExpr*`, `*xName*`, `*yName*`, [`*zLow*`, `*zHigh*`], zColor([`*colorLow*`, `*colorHigh*`]), < <<xGrid(`*min*`, `*max*`, `*incr*`)>, < <<yGrid(`*min*`, `*max*`, `*incr*`)> < <<Transparency(`*alpha|vector*`))`

Description

Fills a set of rectangles on a grid according to a color determined by the expression value as it crosses a range corresponding to a range of colors.

Required Arguments

zExpr Any expression (for example, Sine(y)+Cosine(x)).

xName, *yName* The values to use in the expression.

zLow, *zHigh* A matrix of low and high values (2 to 10).

zColor([colorLow, colorHigh) The two colors that are blended together (4 is green, 6 is orange).

Optional Messages

<<*xGrid*, <<*yGrid* Specifies a box, beyond which the gradients are not drawn.

<<*zColor* The color in which to draw the contour lines. The argument can be either a scalar or a matrix but must evaluate to numeric.

<<Transparency(*number*|*matrix*) The transparency level of the fill. A vector of numbers between 0 and 1 are sequenced through and cycled for the z contours.

Example

```
Gradient Function(Log(a * a + b * b),
a, b, [2 10],
zColor([4, 6]));
```

H Line(*x1*, *x2*, *y*)

H Line(*y*)

Description

Draws a horizontal line at *y* across the graph. If you specify start and end points on the *x*-axis (*x1* and *x2*), the line is drawn horizontally at *y* from *x1* to *x2*. You can also draw multiple lines by using a matrix of values in the *y* argument.

H Size()

Description

Returns the horizontal size of the graphics frame in pixels.

Handle(*xPos*, *yPos*, *dragScript*, *mouseUpScript*)

Description

Places draggable marker at coordinates given by *xPos*, *yPos*. The first script is executed at drag and the second at mouseup.

Heat Color(*x*, < <<*theme*>)

Heat Color(*x*)

Description

Returns the JMP color that corresponds to *n* in the color "*theme*".

Returns

An integer that is a JMP color.

Required Argument

x A value between 0 and 1.

Optional Message

theme A quoted color theme that is supported by Cell Plot. The default value is "Blue to Gray to Red".

`HLS Color(h, l, s)`

`HLS Color({h, l, s})`

Description

Converts hue, lightness, and saturation values into a JMP color number.

Returns

An integer that is a JMP color number.

Arguments

Hue, lightness, and saturation, or a list containing the three HLS values. All values should be between 0 and 1.

`In Path(x, y, (pathMatrix|pathText)`

Description

Determines it the point described by *x* and *y* falls in *path*.

Returns

True (1) if the point (x, y) is in the given path, False(0) otherwise.

Arguments

x and *y* The coordinates of a point.

pathMatrix|*pathText* Either a matrix or a quoted string describing a path.

`In Polygon(x, y, xMatrix, yMatrix)`

`In Polygon(x, y, xyPolygon)`

Description

Returns 1 or 0, indicating whether the point (*x*, *y*) is inside the polygon that is defined by the *xMatrix* and *yMatrix* vector arguments.

xMatrix, *yMatrix* can also be combined into a two-column matrix (*xyPolygon*), allowing you to use three arguments instead of four. Also, *x* and *y* can be conformable vectors, and then a vector of 0s and 1s are returned based on whether each (*x*, *y*) pair is inside the polygon.

Level Color(*i*)

Level Color(*i*, *n*)

Level Color(*i*, *n*, <*theme*>)

Level Color(*i*, <*theme*>)

Description

Assigns a JMP color to categorical data in a graphic.

Returns

An integer that is a JMP color.

Required Arguments

i An integer that is greater than or equal to 1 and less than or equal to the number of categories specified by n.

n The number of categories.

Optional Argument

theme A quoted color theme from the Value Color list of the Column Properties window. If not specified, the JMP Default color theme is applied.

Notes

When the second argument is a quoted character string and not *n*, then the second argument determines the color theme.

Line({*x1*, *y1*}, {*x2*, *y2*}, ...), <<ValueSpace(*Boolean*)

Line([*x1*, *x2*, ...], [*y1*, *y2*, ...]), <<ValueSpace(*Boolean*)

Description

Draws a line between points.

Arguments

{*x1*, *y1*}, {*x2*, *y2*}|[*x1*, *x2*, ...], [*y1*, *y2*, ...] Can be any number of lists of two points, separated by commas or a matrix of *x* values and a matrix of *y* values.

<<ValueSpace(*Boolean*) Draws lines that follow the projection when the line represents a movement of the underlying data, such as a bubble trail in a bubble plot. The Boolean value can be a constant or an expression.

Line Style(*name*|*number*)

Description

Sets the line style used to draw the graph.

Argument

n Can be either a quoted style name or the style's number:

– 0 or "Solid"

- 1 or "Dotted"
- 2 or "Dashed"
- 3 or "DashDot"
- 4 or "DashDotDot"

Marker(<*rowState*>, {*x1*, *y1*}, {*x2*, *y2*}, ...)

Marker(<*rowState*>, [*x1*, *x2*, ...], [*y1*, *y2*, ...])

Description

Draws one or more markers at the points described either by lists or matrices. The optional *rowState* argument sets the type of marker.

Marker Size(*n*)

Description

Sets the size used for markers.

Mousetrap(*dragScript*, *mouseUpscript*)

Description

Captures click coordinates to update graph properties. The first script is executed at drag and the second script at mouseup.

New Heat Image(*matrix*, <*colorTheme*>)

Description

Creates a heat map image based on a matrix and color theme or gradient.

Arguments

matrix A matrix of values. The function creates a heat map that has the same dimensions as the input matrix.

colorTheme A color theme or set of gradient properties that define the mapping of values to colors. The example below shows the use of a custom gradient.

Example

```
width = 256;
height = 256;
wstride = 16;
hstride = 16;
b = J( hstride, wstride, Random Normal() );
a = Transform Each( {z, {row, col}},
  J( height, width, 0 ),
  b[Floor( (row - 1) / hstride ) + 1,
  Floor( (col - 1) / wstride ) + 1]
```

```
);
E1 = New Heat Image(
   a,
   gradient(
      {Color Theme( "Blue To Gray To Orange" ),
      Scale Type( "Standard Deviation" ),
      Show Missing Color( "On" ),
      Reverse Gradient( 1 )}
   )
);
New Window( "test", Lineup Box( E1 ) );
```

Normal Contour(*prob*, *meanMatrix*, *stdMatrix*, *corrMatrix*, <*colorsMatrix*>, <*fill=x*>)

Description

Draws normal probability contours for *k* populations and two variables.

Required Arguments

prob A scalar or matrix of probabilities.

meanMatrix A matrix of means of size *k* by 2.

stdMatrix A matrix of standard deviations of size *k* by 2.

corrMatrix A matrix of correlations of size *k* by 1.

Optional Arguments

colorsMatrix Specifies the color(s) for the k contour(s). The colors must be specified as JSL colors (either JSL color integer values or return values of JSL Color functions such as RGB Color or HLS Color).

fill=x Specifies the amount of transparency for the contour fill color.

Oval(*x1*, *y1*, *x2*, *y2*, <*Fill(Boolean)*>)

Oval({*x1*, *y1*}, {*x2*, *y2*}, <*Fill(Boolean)*>)

Description

Draws an oval inside the rectangle whose diagonal has the coordinates (*x1, y1*) and (*x2, y2*). *Fill* is Boolean. If *Fill* is 0, the oval is empty. If *Fill* is nonzero, the oval is filled with the current fill color. The default value for *Fill* is 0.

Path((*pathMatrix*|*pathText*), <*Fill(Boolean)*>)

Description

Draws a stroke along the given path. If a fill is specified, the interior of the path is filled with the current fill color.

Required Arguments

pathMatrix An Nx3 matrix. If you don't specify the path matrix, specify the path text.

pathText A quoted string that contains SVG code.

Optional Arguments

Fill(Boolean) Specifies whether a line is drawn (0) or the path is filled (1). The default value is 0.

Notes

A path matrix has three columns, for x and y, and a flag. The flag value for each point can be 0 for control, 1 for move, 2 for line segment, 3 for cubic Bézier segment, and any negative value to close the path.

Path To Char(*path*)

Description

Converts a path specification from a matrix to a quoted string.

Returns

A quoted string.

Argument

path An Nx3 path matrix.

Notes

A path matrix has three columns, for x and y, and a flag. The flag value for each point can be 0 for control, 1 for move, 2 for line segment, 3 for cubic Bézier segment, and any negative value to close the path.

Pen Color(*n*)

Description

Sets the color used for the pen.

Pen Size(*n*)

Description

Sets the thickness of the pen in pixels.

Pick Color(<*window title*>, <*name*|*index*|*RGBlist*>)

Description

Creates a color picker, which enables the user to select a color to apply to graphs. The operating system color picker lets users select a predefined color or create their own color. You can also specify a default color in your script. If you omit the default color, Black is selected.

Returns

The color that the user selected in the operating system's color picker.

Arguments

window title The quoted title of the color picker window.

name The name of the default JMP color.

index The number of the default JMP color.

RGBlist The RGB values of the default color.

Pie(*left*, *top*, *right*, *bottom*, *startAngle*, *endAngle*)

Description

Draws a filled pie slice. The two points describe a rectangle, within which is a virtual oval. Only the slice described by the start and end angles is drawn.

Pixel Line To(*h*, *v*)

Description

Draws a one-pixel-wide line from the current pixel location to the location given in pixel coordinates. Set the current pixel location using the `Pixel Origin()` and `Pixel Move To()` functions.

Pixel Move To(*h*, *v*)

Description

Moves the current pixel location to a new location given in pixel coordinates.

Pixel Origin(x, *y*)

Description

Sets the origin, in graph coordinates, for subsequent `Pixel Line To()` or `Pixel Move To()` functions.

Polygon({*x1*, *y1*}, {*x2*, *y2*}, ...)

Polygon(*xMatrix*, *yMatrix*)

Description

Draws a filled polygon defined by the listed points.

`Polygon Area({x1, y1}, {x2, y2}, ...)`

`Polygon Area(xMatrix, yMatrix)`

Description

Calculates the area of the specified polygon.

Examples

```
area = Polygon Area( {0, 0}, {0, 10}, {10, 10}, {10, 0} );
area = Polygon Area( [10 20 30], [10 30 20] );
```

`Polygon Centroid({x1, y1}, {x2, y2}, ...)`

`Polygon Centroid(xMatrix, yMatrix)`

Description

Calculates the centroid of the specified polygon.

Examples

```
{cx, cy} = Polygon Centroid( {0, 0}, {0, 10}, {10, 10}, {10, 0} );
centroid = Polygon Centroid( [10 20 30], [10 30 20] );
```

`Pixel Path(h, v, (path matrix|path text), <fill=Boolean)>, <scale=n>, <orient={n, n})`

Description

Draws a stroke along the given pixel-based path if the fill is 0, or paints the interior of the path if the fill is not 0.

Required Arguments

h, *v* Specifies the horizontal and vertical position.

path matrix Contains three columns for *h*, *v*, and flags for each point in the path. The flag values are 0 for control, 1 for move, 2 for line segment, 3 for cubic Bézier segment, and are negative if the point also closes the path. Each flag (one for each point) can be 0, 1, 2 or 3, (or negative, if it closes the shape).

If you don't specify the path matrix, specify the path text.

path text Supports SVG syntax. The path is scaled (`scale`) and translated about its origin according to the optional parameters (`fill` and `orient`), with the orientation specified in the axis space.

Optional Arguments

`fill(Boolean)` 0 fills the shape. 1 draws an empty shape.

`orient` Rotates the object. For example, an orientation of `{Sqrt( 2 ), Sqrt( 2 )}` draws the shape on a 45-degree angle.

`scale` Sizes the object. For example, a scale of 1.0 draws the shape as it is defined; a scale of 2.0 draws it twice as big; and a scale of 0.5 draws it half as big.

`Pixel Text(<properties>, {h, v}, text, ...)`

Description

Moves to the {*h, v*} pixel position and draws text that the quoted *text* argument specifies.

Required Arguments

h The horizontal pixel position.

v The vertical pixel position.

`text` The quoted string to display at the start and end of the pixels.

Optional Arguments

`center justified` Center justifies the text.

`right justified` Right justifies the text.

`erased` Omits pixels from the edges of the rectangle of text. The text is drawn over the graph content in the background color, erasing the graph within the rectangle to make the text more readable.

`boxed` Displays a box around the text.

`counterclockwise` Rotates the text counterclockwise.

`clockwise` Rotates the text clockwise.

`Rect(left, top, right, bottom, <Fill(Boolean)>)`

`Rect({left, top}, {right, bottom}, <Fill(Boolean)>)`

Description

Draws a rectangle whose diagonal has the coordinates (*left*, *top*) and (*right*, *bottom*). *Fill* is Boolean. If *Fill* is 0, the rectangle is empty. If *Fill* is nonzero, the rectangle is filled with the current fill color. The default value for *Fill* is 0.

`Remove Color Theme(name|{name, <flags>, {color, ...}, <{position, ...}>}>)`

Description

Removes a custom color theme from the global list, either by name or by the full color theme object.

Required Arguments

name The quoted name of the color theme.

color The RGB values for the color.

Optional Arguments

flags A number that represents metadata such as whether the theme is continuous or categorical. Run `Get Color Theme Details("name")` on the color theme and use the flag that is returned.

position A number between 0 and 1. There is one number per color that indicates where on the gradient that color is, where 0 is the beginning and 1 is the end.

Example

```
Remove Color Theme( {"Yellow Blue", 0, {{255, 255, 0}, {0, 0, 255}}, {0.0,
  1.0}} );
```

`RGB Color(r, g, b)`

`RGB Color({r, g, b})`

Description

Converts red, green, and blue values into a JMP color number.

Returns

An integer that is a JMP color number.

Arguments

Red, green, and blue, or a list containing the three RGB values. All values should be between 0 and 1.

`Text(<properties>, ({x, y}|{left, bottom, right, top}), text)`

Description

Draws the quoted string *text* at the given point, either the x and y axes or the left, bottom, right, and top axes.

Properties can be any of several named arguments: `Center Justified`, `Right Justified`, `Erased`, `Boxed`, `Counterclockwise`, `Position`, and named arguments. The position, named arguments, and quoted strings can be added in any order. The position and named arguments apply to all the strings.

`Text Color(n)`

Description

Sets the color for text strings.

`Text Font(fontName, <size>, <bold italic underline strikeout>, <angle>)`

Description

Sets the font for text strings. Use without arguments to get the current font properties. Angle is in degrees clockwise.

Quote the font properties. To apply multiple properties, specify them together, as in `"bold italic"`.

`Text Size(n)`

Description

Sets the font size in points for text strings.

`Transparency(alpha)`

Description

Sets the transparency of the current drawing, with *alpha* between 0 and 1 where 0 is clear (no drawing) and 1 is completely opaque (the default).

Notes

Not all operating systems support transparency.

`V Line(x, <y1, y2>)`

Description

Draws a vertical line at *x* across the graph. If you specify start and end points on the *y*-axis (*y1* and *y2*), the line is drawn vertically at *x* from *y1* to *y2*. You can also draw multiple lines by using a matrix of values in the *x* argument.

`V Size()`

Description

Returns the vertical size of the graphics frame in pixels

`X Function(zExpr, yName, <Min(min), Max(max), Fill(Boolean), Inc(upper bound))`

Description

Draws a plot of the function as the *yName* is varied over the *y*-axis of the graph.

`X Origin()`

Description

Returns the *x*-value for the left edge of the graphics frame.

`X Range()`

Description

Returns the distance from the left to right edge of the display box. For example, `X Origin() + X Range()` is the right edge.

`X Scale(xMin, xMax)`

Description

Sets the range for the horizontal scale. The default value for *xMin* is 0. The default value for *xMax* is 100.

XY Function(*x(t)*, *y(t)*, *t*, Min(*min*), Max(*max*), (Inc(*bound*)|Steps(*min*))

Description

Combines an expression of *x(t)* and *y(t)* to draw an x-y curve for the specified range of parameter *t*. Each time *t* it is varied between Min and Max, the *x* and *y* expressions are evaluated using the current value of *t*.

Note: Either *Inc()* or *Steps()* is needed if the default granularity misses details.

Y Function(*yExpr*, *xName*, <Min(*min*), Max(*max*), Fill(*Boolean*), Inc(*bound*)>)

Description

Draws a plot of the function as the symbol is varied over the *x*-axis of the graph.

Y Origin()

Description

Returns the *y*-value for the bottom edge of the graphics frame.

Y Range()

Description

Returns the distance from the bottom to top edges of a display box. For example, Y Origin() + Y Range() is the top edge.

Y Scale(*yMin*, *yMax*)

Description

Sets the range for the vertical scale. If you do not specify a scale, it defaults to 0, 100.

HTTP Functions

Decode 64 Blob(*string*)

Description

Decodes the quoted string using Base-64 encoding.

Encode 64 Blob(*string*)

Description

Encodes the quoted string using Base-64 encoding.

List Functions

As List(*matrix*)

Description

Converts a matrix into a list. Multi-column matrices are converted to a list of row lists.

Returns

A list.

Argument

matrix Any matrix.

Concat Items({*string1*, *string2*, ...}, <*delimiter*>})

Description

Converts a list of quoted string expressions into one string, with each item separated by a delimiter. The delimiter is a blank if unspecified.

Returns

The concatenated quoted string.

Arguments

string Any quoted string.

delimiter An optional quoted string that is placed between each item. The delimiter can be more than one character long.

Example

```
str1 = "one";
str2 = "two";
str3 = "three";

comb = Concat Items({str1, str2, str3});
   "one two three"
comb = Concat Items({str1, str2, str3}, " : ");
   "one : two : three"
del = ",";
comb = Concat Items({str1, str2, str3}, del);
   "one,two,three"
```

Eval List({*list*})

Description

Evaluates expressions inside *list*.

Returns

A list of the evaluated expressions.

Arguments

list A list of valid JSL expressions.

`Insert(source, item, <position>)`

`Insert(source, key, value)`

Description

Inserts a new *item* into the *source* at the given *position*. If *position* is not given, *item* is added to the end.

For an associative array: Adds the *key* into the *source* associative array and assigns *value* to it. If the *key* exists in *source* already, its value is overwritten with the new *value*.

Required Arguments

source A quoted string, list, vector, expression, or associative array.

item|key Any value to be placed within *source*. For an associative array, *key* might or might not be present in *source*.

value A value to assign to the *key*.

Optional Arguments

position Optional numeric value that specifies the position in *source* to place the *item* into.

`Insert Into(source, item, <position>)`

`Insert Into(source, key, value)`

Description

Inserts a new item into the *source* at the given position in place. The *source* must be an L-value.

Arguments

source A variable that contains a quoted string, list, vector, display box, expression, or associative array.

item|key Any value to be placed within source. For an associative array, *key* might be present in *source*.

position Optional numeric value that specifies the position in *source* to place the *item* into.

value A value to assign to the *key*.

`Is List(x)`

Description

Returns 1 if the evaluated argument is a list, or 0 otherwise.

Items(*string*, <*delimiter*>, <Include Boundary Delimiters(*Boolean*)>)

Description

Returns a list of (possibly empty) quoted sub-strings separated by exactly one of any of the characters specified in the *delimiter* argument.

Arguments

`string` The quoted string that is evaluated.

`Delimiter` (Optional) The character used as a boundary. If *delimiter* is absent, an ASCII space is used. If *delimiter* is the empty quoted string, each character is treated as a separate word.

`Include Boundary Delimiters(Boolean)` (Optional) Returns the empty strings between the boundary and the delimiter.

Example

```
Items( "http://www.jmp.com", ":/." );
   {"http", "", "", "www", "jmp", "com"}
Items(",toy,", ",");
   {"toy"}
Items(",toy,", ",", Include Boundary Delimiters( 1 ));
   {"", "toy", ""}
   /* There is no text between the boundary (the beginning of the quoted string)
      and the comma delimiter, so you get an empty string. The same priniciple
      applies to the delimiter at the end of the string. */
```

List(a, b, c, ...)

{a, b, c, ...}

Description

Constructs a list from a set of items.

N Items(source)

Description

Determines the number of elements in the *source* specified.

Returns

For a list or display box, the number of items in the list or display box is returned. For an associative array, the number of keys is returned. For a matrix, the number of elements in the matrix is returned. For a namespace, the number of functions and variables in the namespace is returned. For a class object, the number of methods, functions, and variables is returned.

Arguments

`source` A list, associative array, matrix, display box, or namespace.

`Remove(source, position, <n>)`

`Remove(source, {items})`

`Remove(source, key)`

Description

Deletes the *n* item(s), starting from the indicated *position*. If *n* is omitted, the item at *position* is deleted. If *position* and *n* are omitted, the item at the end is removed. For an associative array: Deletes the *key* and its value.

Returns

A copy of the *source* with the items deleted.

Arguments

`source` A quoted string, list, vector, expression, or associative array.

`position or key` An integer (or list of integers) that points to a specific item (or items) in the list or expression.

`n` (Optional) An integer that specifies how many items to remove.

`Remove From(source, position, <n>)`

`Remove From(source, key)`

Description

Deletes the *n* item(s) in place, starting from the indicated *position*. If *n* is omitted, the item at *position* is deleted. If *position* and *n* are omitted, the item at the end is removed. For an associative array: Deletes the *key* and its value. The *source* must be an L-value.

Returns

The original *source* with the items deleted.

Arguments

`source` A quoted string, list, vector, expression, display box, or associative array.

`position or key` An integer (or list of integers) that points to a specific item (or items) in the list or expression.

`n` (Optional) An integer that specifies how many items to remove.

`Reverse(source)`

Description

Reverse order of elements or terms in the *source*.

Argument

`source` A quoted string, list, matrix, or expression.

Reverse Into(source)

Description

Reverses the order of elements or terms in *source* in place.

Argument

source A quoted string, list, matrix, display box, or expression.

Shift(source, <n>)

Description

Shifts an item or *n* items from the front to the back of the *source*.

Arguments

source A quoted string, list, matrix, or expression.

n (Optional) An integer that specifies the number of items to shift. Positive values shift items from the beginning of the *source* to the end. Negative values shift items from the end of the *source* to the beginning. The default value is 1.

Shift Into(source, <n>)

Description

Shifts items in place.

Arguments

source A quoted string, list, matrix, display box, or expression.

n (Optional) An integer that specifies the number of items to shift. Positive values shift items from the beginning of the *source* to the end. Negative values shift items from the end of the *source* to the beginning. The default value is 1.

Sort List({list}|expr)

Description

Sort the elements or terms of *list* or *expr*.

Sort List Into({list}|expr)

Description

Sort the elements or terms of *list* or *expr* in place.

`Substitute(`*`string`*`, "substring", "replacementString", ...)`

`Substitute({list}, listItem, replacementItem, ...)`

`Substitute(Expr(sourceExpr), Expr(findExpr), Expr(replacementExpr), ...)`

Description

This is a search and replace function. It searches for a specific portion (second argument) of the source (first argument), and replaces it (third argument).

If a quoted string, finds all matches to *`substring`* in the source *`string`*, and replaces them with the *`replacementString`*.

If a list, finds all matches to *`listItem`* in the source *`list`*, and replaces them with the *`replacementItem`*.

If an expression, finds all matches to the *`findExpr`* in the *`sourceExpr`*, and replaces them with the *`replacementExpr`*. Note that all expressions must be enclosed within an `Expr()` function.

Arguments

`string, list, sourceExpr` A quoted string, list, matrix, or expression in which to perform the substitution.

`substring, listItem, findExpr` A quoted string, list item, or expression to be found in the source string, list, or expression.

`replacementString, replacementItem, replacementExpr` A quoted string, list item, or expression to replace the found string, list item, or expression.

`Substitute Into(`*`string`*`, substring, replacementString, ...)`

`Substitute Into(list, listItem, replacementItem, ...)`

`Substitute Into(Expr(sourceExpr), Expr(findExpr), Expr(replacementExpr), ...)`

Description

This is a search and replace function, identical to `Substitute()` except in place. It searches for a specific portion (second argument) of the source (first argument), and replaces it (third argument). The first argument must be an L-value.

If a quoted string, finds all matches to *`substring`* in the source *`string`*, and replaces them with the *`replacementString`*.

If a list, finds all matches to *`listItem`* in the source *`list`*, and replaces them with the *`replacementItem`*.

If an expression, finds all matches to the *`findExpr`* in the *`sourceExpr`*, and replaces them with the *`replacementExpr`*. Note that all expressions must be enclosed within an `Expr()` function.

Arguments

`string, list, sourceExpr` A quoted string, list, matrix, or expression in which to perform the substitution.

`substring, listItem, findExpr` A quoted string, list item, or expression to be found in the source string, list, or expression.

`replacementString, replacementItem, replacementExpr` A quoted string, list item, or expression to replace the found string, list item, or expression.

`Words(string, <delimiter>)`

Description

Extracts the words from the quoted *string* according to the delimiters given. The default delimiter is ASCII whitespace. If you include a second argument, any and all characters in that argument are considered delimiters. If *delim* is an empty string, each character is treated as a separate word.

Examples

```
Words( "the quick brown fox" );
   {"the","quick","brown","fox"}
Words( "Doe, Jane P.",", ." );
   {"Doe","Jane","P"}
```

MATLAB Integration Functions

JMP provides the following interfaces to access MATLAB. The basic execution model is to first initialize the MATLAB connection, perform the required MATLAB operations, and then terminate the MATLAB connection. In most cases, these functions return 0 if the MATLAB operation was successful or an error code if it was not. If the MATLAB operation is not successful, a message is written to the Log window. The single exception to this is `MATLAB Get( )`, which returns a value.

MATLAB JSL Function Interfaces

`MATLAB Connect( <named arguments> )`

Description

Initializes the MATLAB integration interfaces and returns an active MATLAB integration interface connection as a scriptable object.

Returns

MATLAB scriptable object.

Named Arguments

`Echo(Boolean)` Sends the MATLAB source lines to the JMP log. The default value is true.

`MATLAB Control( <named arguments> )`

Description

Sends control operations to signal MATLAB with external events such as source line echoing.

Returns

None.

Arguments

None.

Named Arguments

`Echo(Boolean)` Global. Echo MATLAB source lines to the JMP log.

`Visible(Boolean)` Global. Determine whether to show or hide the active MATLAB workspace.

`MATLAB Execute( { list of inputs }, { list of outputs }, mCode, <named arguments> )`

Description

Submits the MATLAB code to the active global MATLAB connection given a list of inputs. Upon completion, retrieves a list of outputs.

Returns

0 if successful, otherwise nonzero.

Arguments

`{ list of inputs }` Positional, name list. List of JMP variable names to send to MATLAB as inputs.

`{ list of outputs }` Positional, name list. List of JMP variable names to retrieve from MATLAB as outputs.

`mCode` Positional, quoted string. The MATLAB code to submit.

Named Arguments

`Expand(Boolean)` Perform an Eval Insert on the MATLAB code prior to submission.

`Echo(Boolean)` Echo MATLAB source lines to the JMP log. Default is true.

Example

The following example sends the JMP variables x and y to MATLAB, executes the MATLAB statement *z = x * y*, and then gets the MATLAB variable z and returns it to JMP.

```
MATLAB Init();
x = [1 2 3];
```

```
y = [4 5 6];
MATLAB Execute( {x, y}, {z}, "z = x * y;" );
Show( z );
```

MATLAB Get(name)

Description

Gets named variable from MATLAB to JMP.

Returns

Value of named variable.

Arguments

`name` Positional. The name of a JMP variable to be sent to MATLAB.

Example

Suppose that a matrix named `qbx` and a structure named `df` are present in your MATLAB connection.

```
// get the MATLAB variable qbx and placed it into a JMP variable qbx
qbx = MATLAB Get( qbx );

/* get the MATLAB variable df and placed it into a JMP data table
referenced by df */
df = MATLAB Get( df );
```

Table 2.1 shows what JMP data types can be exchanged with MATLAB using the `MATLAB Get( )` function. Getting lists from MATLAB recursively examines each element of the list and sends each base MATLAB data type. Nested lists are supported.

Table 2.1 JMP and MATLAB Equivalent Data Types for `MATLAB Get( )`

MATLAB Data Type	JMP Data Type
Double	Numeric
Logical	Numeric (0 | 1)
String	String
Integer	Numeric
Date/Time	Numeric
Structure	Data Table
Matrix	Numeric Matrix
Numeric Vector	Numeric Matrix

Table 2.1 JMP and MATLAB Equivalent Data Types for `MATLAB Get( )` *(Continued)*

MATLAB Data Type	JMP Data Type
String Vector	List of Strings
Graph	Picture object

`MATLAB Get Graphics( format )`

Description

Get the last graphic object written to the MATLAB graph display window in a specific graphic format. The graphic object can be returned in several graphic formats.

Returns

JMP Picture object.

Argument

`format` Positional. The format the MATLAB graph display window contents are to be converted to. Valid formats are `png`, `bmp`, `jpeg`, `jpg`, `tiff`, `tif`, and `gif`.

`MATLAB Get Version`

Description

Returns the version number of MATLAB being used with the JMP MATLAB interfaces.

`MATLAB Init( <named arguments> )`

Description

Initializes the MATLAB integration interfaces.

Returns

Return code.

Named Arguments

`Echo(Boolean)` Sends MATLAB source lines to the JMP log. This option is global. The default value is true.

`MATLAB Is Connected()`

Description

Determines whether a MATLAB connection is active.

Returns

1 if connected, otherwise 0.

MATLAB JMP Name To MATLAB Name(name)

Description

Maps a JMP variable name to its corresponding MATLAB variable name using MATLAB variable name naming rules.

Returns

A quoted string, mapped MATLAB variable name.

Argument

`name` Positional. The name of a JMP variable to be sent to MATLAB.

MATLAB Send(name, <named arguments>)

Description

Sends the named variable from JMP to MATLAB.

Returns

0 if successful, otherwise nonzero.

Arguments

`name` Positional. The name of a JMP variable to be sent to MATLAB.

Named Arguments

The following optional arguments apply to data tables only:

`Selected(Boolean)` Send selected rows from the referenced data table to MATLAB.

`Excluded(Boolean)` Send only excluded rows from the referenced data table to MATLAB.

`Labeled(Boolean)` Send only labeled rows from the referenced data table to MATLAB.

`Hidden(Boolean)` Send only hidden rows from the referenced data table to MATLAB.

`Colored(Boolean)` Send only colored rows from the referenced data table to MATLAB.

`Markered(Boolean)` Send only markered rows from the referenced data table to MATLAB.

`Row States(Boolean, <named arguments>)` Send row states from referenced data table to MATLAB by adding an additional data column named “RowState”. Create multiple selections by adding together individual settings. The row state consists of individual settings with the following values:

- Selected = 1
- Excluded = 2
- Labeled = 4
- Hidden = 8
- Colored = 16
- Markered = 32

The following optional, named Row States arguments are supported:

`Colors(Boolean)` Send row colors. Adds additional data column named "RowStateColor".

`Markers(Boolean)` Send row markers. Adds additional data column named "RowStateMarker".

Example

```
// create a matrix, assign it to X, and send the matrix to MATLAB
X = [1 2 3];
ml = MATLAB Send( X );

/* open a data table, assign a reference to it to dt, and send the
data table along with its current row states to MATLAB */
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
ml = MATLAB Send( dt, Row States( 1 ) );
```

Table 2.2 shows what JMP data types can be exchanged with MATLAB using the `MATLAB Send( )` function. Sending lists to MATLAB recursively examines each element of the list and sends each base JMP data type. Nested lists are supported.

Table 2.2 JMP and MATLAB Equivalent Data Types for `MATLAB Send( )`

MATLAB Data Type	JMP Data Type
Double	Numeric
String	String
Double Matrix	Matrix
Structure	Data Table

Example

```
MATLAB Init( );
X = 1;
MATLAB Send( X );
S = "Report Title";
MATLAB Send( S );
M = [1 2 3, 4 5 6, 7 8 9];
MATLAB Send( M );
MATLAB Submit( "
X
S
M
" );
MATLAB Term( );
```

MATLAB Send File(filename, <MATLAB Name(name)>)

Description

Sends a data file to MATLAB.

Arguments

`filename` Specifies a quoted string that identifies the pathname to the file to be sent to MATLAB.

`MATLAB Name` Enables you to change the name of the file sent to MATLAB.

MATLAB Submit File('pathname', <named arguments>)

Description

Submits statements to MATLAB using a file pointed to by pathname.

Returns

0 if successful, otherwise nonzero.

Arguments

`Pathname` Positional, quoted string. Pathname to file containing MATLAB source lines to be executed.

Named Arguments

`Expand(Boolean)` Perform an Eval Insert on the MATLAB code prior to submission.

`Echo(Boolean)` Echo MATLAB source lines to the JMP log. Default is true.

MATLAB Submit(mCode, <named arguments>)

Description

Submits the MATLAB code to the active global MATLAB connection.

Returns

0 if successful, otherwise nonzero.

Arguments

`mCode` Positional, quoted string. The MATLAB code to submit.

Named Arguments

`Expand(Boolean)` Perform an Eval Insert on the MATLAB code prior to submission.

`Echo(Boolean)` Echo MATLAB source lines to the JMP log. Default is true.

Example

The following example creates two vectors of random points and plots them as x and y variables.

```
MATLAB Init();
mc = MATLAB Submit("/[
   x = rand(5);
   fprintf('%f/n', x);
```

```
    y = rand(5);
    fprintf('%f/n', x);
    z = plot(x, y);
]/" );
```

`MATLAB Term();`

Description

Terminates the currently active MATLAB integration interface.

Returns

1 if an active MATLAB connection exists, otherwise returns 0.

Arguments

None.

Matrix Functions

`All(A, ...)`

Returns

1 if all matrix arguments are nonzero; 0 otherwise.

`Any(A, ...)`

Returns

1 if one or more elements of one or more matrices are nonzero; 0 otherwise.

`B Spline Coef(x, Internal Knot Grid, <degree=3>, <Knot End Points=min(x)||max(x)>)`

Description

Finds the matrix of B-spline coefficients for the data in the *x* argument.

Returns

The matrix of B-spline basis coefficients. This matrix can be used as a design matrix in a linear model. The first column of the matrix contains an intercept term.

Arguments

`x` A row or column vector that contains the data.

`Internal Knot Grid` Either a single number that designates the number of desired knot points based on percentiles of *x* or a vector of values that designate the internal knot points. The number of internal knots must be greater than zero and less than or equal to the number of unique elements in *x* minus two.

`degree` A number that indicates the degree of the B-splines. Defaults to 3.

`Knot End Points` A 2x1 matrix that designates the lower and upper knot points. If this argument is not specified, the default lower and upper knot points are the minimum and maximum values of *x*, respectively.

Notes

This function is used in column formulas created by the Functional Data Explorer platform.

`CDF(Y)`

Description

Returns values of the empirical cumulative probability distribution function for *Y*, which can be a vector or a list. Cumulative probability is the proportion of data values less than or equal to the value of *QuantVec*.

Syntax

```
{QuantVec, CumProbVec} = CDF(YVec)
```

`Chol Update(L, V, C)`

Description

If L is the Cholesky root of an *n*x*n* matrix A, then after calling `cholUpdate` L is replaced with the Cholesky root of A+V*C*V' where C is an *m*x*m* symmetric matrix and V is an n*m matrix.

`Cholesky(A)`

Description

Finds the lower Cholesky root (L) of a positive semi-definitive matrix, L*L' = A.

Returns

L (the Cholesky root).

Arguments

`A` a symmetric matrix.

`Correlation(matrix, < <<"Pairwise">, < <<"Shrink">, < <<Freq(vector)>, < <<Weight(vector)>)`

Description

Calculates the correlation matrix of the data in the *matrix* argument.

Returns

The correlation matrix for the specified matrix.

Argument

`matrix` A matrix that contains the data. If the data has *m* rows and *n* columns, the result is an m-by-m matrix.

`"Pairwise"` Uses the pairwise method for missing values rather than the row-wise method.

`"Shrink"` Performs the Schafer-Strimmer shrinkage estimate.

`<<Freq(vector)` A vector that specifies frequencies for the rows of the matrix argument.

`<<Weight(vector)` A vector that specifies weights for the rows of the matrix argument.

Notes

By default, rows are discarded if they contain missing values. If the "Pairwise" option is specified, all pairs of nonmissing values are used in the correlation matrix calculation.

This function uses multithreading if available, so it is recommended for large problems with many rows.

When a column is constant, the correlations for it are 0, and the diagonal element is also 0.

`Covariance(matrix, < <<"Pairwise">, < <<"Shrink">, < <<Freq(vector)>, < <<Weight(vector)>)`

Description

Calculates the covariance matrix of the data in the *matrix* argument.

Returns

The covariance matrix for the specified matrix.

Argument

`matrix` A matrix that contains the data. If the data has *m* rows and *n* columns, the result is an m-by-n matrix.

`"Pairwise"` Uses the pairwise method for missing values rather than the row-wise method.

`"Shrink"` Performs the Schafer-Strimmer shrinkage estimate.

`<<Freq(vector)` A vector that specifies frequencies for the rows of the matrix argument.

`<<Weight(vector)` A vector that specifies weights for the rows of the matrix argument.

Notes

By default, rows are discarded if they contain missing values. If the "Pairwise" option is specified, all pairs of nonmissing values are used in the covariance matrix calculation.

This function uses multithreading if available, so it is recommended for large problems with many rows.

Design(vector, < levelsList | <<levels, <<ElseMissing >)

Description

Creates a design matrix that contains a column of 1s and 0s for each unique value of a *vector* of values.

Returns

A design matrix with a column of 1s and 0s for each unique value of the argument or a list that contains the design matrix and a list of levels.

Argument

`vector` A vector.

`levelsList` An optional list or matrix argument that specifies the levels in the returned matrix.

`<<levels` An optional argument that changes the return value to a list that contains the design matrix and a list of the levels.

`<<ElseMissing` An optional argument that changes the handling of values in the *vector* argument that do not appear in the *levelsList* argument. If this argument is specified, missing values are placed in the design matrix. Otherwise, 0s are placed in the design matrix.

Notes

Missing values in the *levelsList* argument are not ignored. For example:

```
Show( Design ( ., [. 0 1] ),
Design( 0, [. 0 1] ),
Design( 1, [. 0 1] ),
Design( [0 0 1 . 1], [. 0 1] ),
Design( {0, 0, 1, ., 1}, [. 0 1] ) );
   Design(., [. 0 1]) = [1 0 0];
   Design(0, [. 0 1]) = [0 1 0];
   Design(1, [. 0 1]) = [0 0 1];
   Design([0 0 1 . 1], [. 0 1]) =
   [  0 1 0,
      0 1 0,
      0 0 1,
      1 0 0,
      0 0 1];
   Design({0, 0, 1, ., 1}, [. 0 1]) =
   [  0 1 0,
      0 1 0,
      0 0 1,
      1 0 0,
      0 0 1];
```

`Design Last(vector, < levelsList, <<ElseMissing >)`

Description

Creates a design matrix that contains a column of 1s and 0s for all but the last of the unique values of the argument. The last level is coded as a row of 0s.

Returns

A full-rank design matrix or a list that contains the design matrix and a list of levels.

Arguments

`vector` A vector.

`levelsList` An optional list or matrix argument that specifies the levels in the returned matrix. If this argument is specified, the last level in this list or matrix is treated as the last level in the design matrix. Otherwise, the last level is defined as the largest value in the *`vector`* argument.

`<<ElseMissing` An optional argument that changes the handling of values in the *`vector`* argument that do not appear in the *`levelsList`* argument. If this argument is specified, missing values are placed in the design matrix. Otherwise, 0s are placed in the design matrix.

`Design Nom(vector, < levelsList | <<levels, <<ElseMissing >)`

`DesignF(vector, < levelsList | <<levels, <<ElseMissing >)`

Description

Creates a design matrix that contains a column of 1s and 0s for all but the last of the unique values of the argument. The last level is coded as a row of -1s.

Returns

A full-rank design matrix or a list that contains the design matrix and a list of levels.

Argument

`vector` A vector.

`levelsList` An optional list or matrix argument that specifies the levels in the returned matrix. If this argument is specified, the last level in this list or matrix is treated as the last level in the design matrix. Otherwise, the last level is defined as the largest value in the *`vector`* argument.

`<<levels` An optional argument that changes the return value to a list that contains the design matrix and a list of levels.

`<<ElseMissing` An optional argument that changes the handling of values in the *`vector`* argument that do not appear in the *`levelsList`* argument. If this argument is specified, missing values are placed in the design matrix. Otherwise, 0s are placed in the design matrix.

Notes

Missing values in the *`levelsList`* argument are not ignored. For example:

```
Show( Design Nom( ., [. 0 1] ),
Design Nom( 0, [. 0 1] ),
Design Nom( 1, [. 0 1] ),
Design Nom( [0 0 1 . 1], [. 0 1] ),
Design Nom( {0, 0, 1, ., 1}, [. 0 1] ) );
  Design Nom(., [. 0 1]) = [1 0];
  Design Nom(0, [. 0 1]) = [0 1];
  Design Nom(1, [. 0 1]) = [-1 -1];
  Design Nom([0 0 1 . 1], [. 0 1]) = [0 1, 0 1, -1 -1, 1 0, -1 -1];
  Design Nom({0, 0, 1, ., 1}, [. 0 1]) = [0 1, 0 1, -1 -1, 1 0, -1 -1];
```

Design Ord(vector, < levelsList | <<levels, <<ElseMissing >)

Description

Creates a design matrix that contains a column for all but the last of the unique values of the argument. The first level is coded as a row of 0s. Each subsequent (nth) level in the *levelsList* argument is coded as a row of (n-1) 1s and the rest 0s.

Returns

A full-rank design matrix or a list that contains the design matrix and a list of levels.

Argument

`vector` A vector.

`levelsList` An optional list or matrix argument that specifies the levels in the returned matrix.

`<<levels` An optional argument that changes the return value to a list that contains the design matrix and a list of levels.

`<<ElseMissing` An optional argument that changes the handling of values in the *vector* argument that do not appear in the *levelsList* argument. If this argument is specified, missing values are placed in the design matrix. Otherwise, 0s are placed in the design matrix.

Notes

Missing values in the *levelsList* argument are not ignored. For example:

```
Show( Design Ord( ., [. 0 1] ),
Design Ord( 0, [. 0 1] ),
Design Ord( 1, [. 0 1] ),
Design Ord( [0 0 1 . 1], [. 0 1] ),
Design Ord( {0, 0, 1, ., 1}, [. 0 1] ) );
  Design Ord(., [. 0 1]) = [0 0];
  Design Ord(0, [. 0 1]) = [1 0];
  Design Ord(1, [. 0 1]) = [1 1];
  Design Ord([0 0 1 . 1], [. 0 1]) = [1 0, 1 0, 1 1, 0 0, 1 1];
  Design Ord({0, 0, 1, ., 1}, [. 0 1]) = [1 0, 1 0, 1 1, 0 0, 1 1];
```

Det(A)

Description

Determinant of a square matrix.

Returns

The determinant.

Argument

A A square matrix.

Diag(A, <B>)

Description

Creates a diagonal matrix from a square matrix or a vector. If two matrices are provided, concatenates the matrices diagonally.

Returns

The matrix.

Argument

A a matrix or a vector.

Direct Product(A, B)

Description

Direct (Kronecker) product of square matrices or scalars A[i,j]*B.

Returns

The product.

Arguments

A, B Square matrices or scalars.

Distance(x1, x2, <scales>, <powers>)

Description

Produces a matrix of distances between rows of x1 and rows of x2.

Returns

A matrix.

Arguments

x1, x2 Two matrices.

scales Optional argument to customize the scaling of the matrix.

powers Optional argument to customize the powers of the matrix.

`E Div(A, B)`

`A:/B`

Returns

A matrix that contains the element-by-element division of two matrices.

Arguments

`A, B` Two matrices. Both matrix elements must have the same dimensions.

`E Max(A, B)`

Returns

A matrix that contains the element-by-element maximum values of two or more matrices or scalar arguments.

Arguments

`A, B` Two or more matrices or scalars. All matrix elements must have the same dimensions.

`E Min(A, B)`

Returns

A matrix that contains the element-by-element minimum values of two or more matrices or scalar arguments.

Arguments

`A, B` Two or more matrices or scalars. All matrix elements must have the same dimensions.

`E Mult(A, B)`

`A:*B`

Description

Element-by-element multiplication of two matrices.

Returns

A matrix that contains the element-by-element multiplication of two or more matrices or scalar arguments.

Arguments

`A, B` Two or more matrices or scalars. All matrix elements must have the same dimensions.

Eigen(A)

Description

Eigenvalue decomposition.

Returns

A list {M, E} such that E * Diag(M) * E = A'.

Argument

A A symmetric matrix.

Estimate Bartlett Factor Score(dataRow, ManMeans, LatMeans, S, A)

Description

Estimates factor scores from a structural equation model (SEM) using Bartlett's method.

Returns

A row vector of estimated factor scores based on the structural equation model.

Arguments

dataRow A row vector of data values.

ManMeans A vector of model-implied manifest variable means.

LatMeans A vector of model-implied latent variable means.

S The symmetric S matrix from a structural equation model.

A The rectangular A matrix from a structural equation model.

Estimate Factor Score(dataRow, Covariance, ManMeans, LatMeans)

Description

Estimates factor scores from a structural equation model (SEM).

Returns

A row vector of estimated factor scores based on the structural equation model.

Arguments

dataRow A row vector of data values.

Covariance A model-implied variance-covariance matrix.

ManMeans A vector of model-implied manifest variable means.

LatMeans A vector of model-implied latent variable means.

Note

This function is used in the Save Factor Scores option in an Structural Equation Model report.

Fourier Basis Coef(x, Number Pairs, <Period=max(x)-min(x)+1>)

Description

Finds the matrix of Fourier basis coefficients for the data in the *x* argument.

Returns

The matrix of Fourier basis coefficients. This can be used as a design matrix in a linear model. The first column of the matrix contains an intercept term. The remaining columns contain pairs of basis coefficients, where pair *i* is defined as the `sin()` and `cos()` of *i* * (2 * π / Period) * x.

Arguments

`x` A row or column vector that contains the data.

`Number Pairs` The number of `sin()` and `cos()` pairs for the Fourier basis.

`Period` The period for trigonometric functions that make up the Fourier basis.

Notes

This function is used in column formulas created by the Functional Data Explorer platform.

G Inverse(A)

Description

Generalized (Moore-Penrose) matrix inverse.

H Direct Product(A, B)

Description

Horizontal direct product of two square matrices of the same dimension or scalars.

Hough Line Transform(matrix, <NAngle(number)>, <NRadius(number)>)

Description

Takes a matrix of intensities and transforms it in a way that is useful for finding streaks in the matrix. Produces a matrix containing the Hough Line Transform with angles as columns and radiuses as rows.

Argument

`matrix` A matrix that can be derived from the intensities of an image, but is more likely from a semiconductor wafer that may have defects across in a streak due to planarization machines.

`NAngle(number)` Enter the number of the angle to obtain a different sized transform. The default is 180 degrees.

`NRadius(number)` Enter the number of the radius to obtain a different sized transform. The default is sqrt(NRow*nRow+nCol*nCol).

Identity(n)

Description

Creates an *n*-by-*n* identity matrix with ones on the diagonal and zeros elsewhere.

Returns

The matrix.

Argument

`n` An integer.

Index(i, j, <increment>)

i::j

Description

Creates a column matrix whose values range from *i* to *j*.

Returns

The matrix.

Arguments

`i, j` Integers that define the range: *i* is the beginning of the range, *j* is the end.

`increment` Optional argument to change the default increment, which is +1.

Inv()

See "Inverse(A)".

Inv Update(A, X, 1|-1)

Description

Efficiently update an **X**′**X** matrix.

Arguments

`A` The matrix to be updated.

`X` One or more rows to be added to or deleted from the matrix *A*.

`1|-1` The third argument controls whether the row or rows defined in the second argument, *X*, are added to or deleted from the matrix *A*. `1` means to add the row or rows and `-1` means to delete the row or rows.

Inverse(A)

Inv(A)

Description

Returns the matrix inverse. The matrix must be square non-singular.

Is Matrix(x)

Description

Returns 1 if the evaluated argument is a matrix, or 0 otherwise.

J(nrows, <ncols>, <value>)

Description

Creates a matrix of identical values.

Returns

The matrix.

Arguments

`nrows` Number of rows in matrix. If `ncols` is not specified, `nrows` is also used as `ncols`.

`ncols` Number of columns in matrix.

`value` The value used to populate the matrix. If `value` is not specified, 1 is used.

KDTable(matrix)

Description

Returns a table to efficiently look up near neighbors.

Returns

A KDTable object.

Argument

`matrix` A matrix of k-dimensional points. The number of dimensions or points is not limited. Each column in the matrix represents a dimension to the data, and each row represents a data point.

Messages

`<<Distance between rows(row1, row2)` Returns the distance between two the two specified rows in the KDTable. The distance applies to removed and inserted rows as well.

`<<K nearest rows(stop, <position>)` Returns a matrix. Position is a point that is described as a row vector for the coordinate of a row, or as the number of a row. If *`position`* is not specified, returns the *n* nearest rows and distances to all rows. If *`position`* is specified, returns the *n* nearest rows and distances to either a point or a row. *`Stop`* is either *n* or {*n*, *limit*}. The limit parameter limits the number of rows that will be found. It can be specified one of two ways: a number, like 5, means return the 5 nearest rows. A list, like {5,10}, means return up to 5 nearest rows, stopping when the distance of 10 is exceeded. In the second case, the last row may have a distance greater than 10. Since the command continues until it finds the closest row beyond the stop radius, this point is also returned. This can be especially useful if there are no rows within the radius.

<<Remove rows(number | vector) Remove either the row specified by *number* or the rows specified by *vector*. Returns the number of rows that were removed. Rows that were already removed are ignored.

<<Insert rows(number | vector) Re-insert either the row specified by *number* or the rows specified by *vector*. Returns the number of rows that were inserted. Rows that were already inserted are ignored.

Notes

When rows are removed or inserted, the row indices do not change. You can remove and re-insert only rows that are in the KDTable object. If you need different rows, construct a new KDTable.

`Least Squares Solve(y, X, < <<noIntercept, <<weights(OptionalWeightVector), <<method("Sweep"|"GInv")>)`

Description

Computes least squares regression estimates for the assumed model y = X * beta + error.

Returns

A list that contains the matrix `Beta=Inverse(X'X)X'y` and the estimated variance matrix of Beta.

Optional Named Arguments

<<noIntercept Specifies a no-intercept model.

<<weights(optional weight vector) Specifies a vector of weights to perform weighted least squares.

<<method("Sweep" | "GInv") Specifies the method for solving the normal equations. The default Sweep method is more computationally efficient, but you can also specify the generalized inverse ("GInv") method, which is more numerically stable.

`Linear Regression(y, X, < <<noIntercept, <<printToLog, <<weight(OptionalWeightVector), <<freq(OptionalFreqVector)>)`

Description

Fits a linear regression for the assumed model y = X * beta + error.

Returns

A list that contains a vector of the estimates, a vector of the standard error, and a list of diagnostics. The list of diagnostics contains vectors of the *t* statistics and *p*-values for the estimates, as well as the R-Square and adjusted R-Square values for the regression fit.

Optional Named Arguments

<<noIntercept Excludes the intercept.

<<printToLog Prints a summary of the fit to the log.

<<weight(vector) Specifies a vector of weights to perform weighted least squares.

<<freq(vector) Specifies a vector of frequencies for each row of y and X.

Example

```
n = 10;
x = J( n, 1, Random Normal() );
y = 1 + x * 3 + J( n, 1, Random Normal() );
{Estimates, Std_Error, Diagnostics} = Linear Regression( y, x, <<printToLog );
As Table( y || x );
Bivariate( Y( :Col1 ), X( :Col2 ), Fit Line( 1 ) );
```

`Loc(A)`

`Loc(A, item)`

Description

Returns a matrix of subscript positions where *A* is nonzero and nonmissing. For the two-argument function, Loc returns a matrix of positions where *item* is found within *A*. If the first argument is a list, the second argument is required.

Argument

`A` a matrix or a list

`item` the item to be found within the matrix or list *A*

`Loc Max(A)`

Description

Returns the position of the maximum element in a matrix.

Returns

An integer that is the specified position.

Argument

`A` a matrix

`Loc Min(A)`

Description

Returns the position of the minimum element in a matrix.

Returns

An integer that is the specified position.

Argument

`A` a matrix

`Loc NonMissing(matrix, ..., {list}, ...)`

Description

Returns indices of nonmissing rows in matrices or lists. In lists, the function can also return indices of nonempty characters.

Returns

The new matrix or list.

`Loc Sorted(A, B)`

Description

Returns a column vector of subscript positions where the values of *A* have values less than or equal to the values in *B* based on a binary search. *A* must be a matrix sorted in ascending order without missing values.

Returns

The new matrix, which has the same dimensions as *B*. If a value in *B* is less than the first value in *A*, the returned subscript position for that value is 1.

Argument

`A, B` matrices

`Matrix({{x11, ..., x1m}, {x21, ..., 2m}, {...}, {xn1, ..., xnm}})`

`Matrix({x1, ..., xn})`

`Matrix(n, m)`

Description

Constructs an *n*-by-*m* matrix. The following specification methods are available:

- If you specify a list of *n* lists that each contain *m* row values, the matrix is formed by vertically concatenating the evaluated lists. The list items must evaluate to numeric values or row vectors, and the dimensions of the items must be conformable.
- If you specify a single list of *n* items, the return value is an *n*-by-1 column vector. The items of the evaluated list must evaluate to numeric values.
- If you specify two integer arguments, the return value is a matrix of zeros that contains *n* rows and *m* columns.

Examples

```
Matrix({{1, 2, 3}, {4, [5 6]}, {7, 8, 9}});
  [1 2 3, 4 5 6, 7 8 9]
Matrix({{[1 2 3], 4, 5, 6, 7, 8, 9}});
  [1 2 3 4 5 6 7 8 9]
Matrix({2,3+7});
  [2, 10]
Matrix(2,3);
  [0 0 0, 0 0 0]
```

Matrix Mult(A, B)

C=A*B, ...

Description

Matrix multiplication.

Arguments

A, B, ... Two or more matrices, which must be conformable (all matrices after the first one listed must have the same number of rows as the number of columns in the first matrix).

Notes

Matrix Mult() allows only two arguments, while using the * operator enables you to multiply several matrices.

Matrix Rank(A)

Description

Returns the rank of the matrix A.

Mode({list} or matrix)

Description

Selects the most frequent item from a numeric or character list or a numeric matrix. In the event of a tie, the lower value is selected. If multiple arguments are specified, a combination of numeric values and quoted character strings is acceptable.

Arguments

Specify either a list or a matrix.

Multivariate Normal Impute(yVec, meanYvec, symCovMat, colMin, colMax)

Description

Imputes missing values in yVec based on the mean and covariance.

Arguments

yVec The vector of responses.

meanYvec The vector of response means.

symCovMat A symmetric matrix containing the response covariances. If the covariance matrix is not specified, then JMP imputes with means.

colMin A vector of column minimums. Provides lower bounds for the imputations.

colMax A vector of column maximums. Provides upper bounds for the imputations.

`NChooseK Matrix(n, k)`

Description

Returns a matrix of *n* things taken *k* at a time (*n* select *k*).

`N Col(x)`

`N Cols(x)`

Description

Returns the number of columns in either a data table or a matrix.

Argument

x Can be a data table or a matrix.

`Ortho(A, <Centered(0)>, <Scaled(1)>)`

Description

Orthonormalizes the columns of matrix *A* using the Gram Schmidt method. `Centered(0)` makes the columns to sum to zero. `Scaled(1)` makes them unit length.

`Ortho Poly(vector, order)`

Description

Returns orthogonal polynomials for a *vector* of indices representing spacings up to the *order* given.

`P Spline Coef(x, Internal Knot Grid, <degree=3>)`

Description

Finds the matrix of penalized basis spline (P-spline) coefficients for the data in the *x* argument.

Returns

The matrix of P-spline basis coefficients, which is the truncated power basis of the specified *degree*. The truncated power basis of degree p with knots k_1 through k_K is defined as follows:

$$1, x, x^2, \ldots, x^p, (x-k_1)_+^p, \ldots, (x-k_K)_+^p$$

where $(x - k_1)_+$ is the positive part of $(x - k_1)$ and is set to zero for negative values of $(x - k_1)$.

Arguments

x A row or column vector that contains the data.

`Internal Knot Grid` Either a single number that designates the number of desired knot points based on percentiles of *x* or a vector of values that designate the internal knot points.

`degree` A number that indicates the degree of the P-splines. Defaults to 3.

Notes

This function is used in column formulas created by the Functional Data Explorer platform.

`Parallel Assign({thread_local_var = global_var, ...}, matrix[a, b] = expression using a and b)`

Description

Uses multiple threads to assign values to the matrix. Enables you to take advantage of multiple cores on a computer. The function has two arguments.

- The first argument is a list of assignment statements that copies global variables into each thread's local variable list.
- The second argument is an assignment expression with a left-hand-side that is a matrix with one or two prototype indexes and a right hand side that can be any JSL expression using those indexes and the local variables from the list (and in JMP global: variables).

Example

The following example provides read access to the global namespace.

```
a = 42;
x = [1 2 3, 4 5 6, 7 8 9];
Show( Parallel Assign( {}, x[i, j] = global:a ) );
Show( x );
   Parallel Assign({}, x[i,j] = global:a) = 1;
   x =
   [  42 42 42,
      42 42 42,
      42 42 42];
```

`Print Matrix(M, <named arguments>)`

Description

Returns a quoted string that contains a well-formatted matrix. You can use the function, for example, to print the matrix to the log.

Argument

`M` A matrix.

Optional Named Arguments

`<<ignore locale(Boolean)` Set to false (0) to use the decimal separator for your locale. Set to true (1) to always use a period (`.`) as a separator. The default value is false (0).

`<<decimal digits(n)` An integer that specifies the number of digits after the decimal separator to print.

`<<style("style name")` Use one of three available styles: Parseable is a reformatted JSL matrix expression. Latex is formatted for LaTex. If you specify Other, you must define the following three arguments.

`<<separate("character")` Define the separator for concatenated entries.

`<<line begin("character")` Define the beginning line character.

`<<line end("character")` Define the ending line character.

`QR(A)`

Description

Returns the QR decomposition of *A*. Typical usage is `{Q, R} = QR(A)`.

`Rank Index(vector)`

`Rank(vector)`

Description

Returns a vector of indices that, used as a subscript to the original *vector*, sorts the vector by rank. Excludes missing values. Lists of numbers or quoted strings are supported in addition to matrices.

`Ranking(vector)`

Description

Returns a vector of ranks of the values of *vector*, low to high as 1 to *n*, ties arbitrary. Lists of numbers or quoted strings are supported in addition to matrices.

`Ranking Tie(vector)`

Description

Returns a vector of ranks of the values of *vector*, but ranks for ties are averaged. Lists of numbers or quoted strings are supported in addition to matrices.

`Scoring Impute(rowWithMissing, VMat, colMeanVec, colStdDevVec)`

Description

Provides streaming functionality for the Automated Data Imputation (ADI) algorithm.

Returns

Returns the row vector with the missing values imputed using the standard least squares estimation.

Arguments

`rowwithMissing` A row vector that contains missing values.

`VMat` A loading matrix that is produced by the ADI algorithm.

`colMeanVec` A vector of the column means ignoring missing cells.

`colStdDevVec` a vector of the column standard deviations ignoring missing cells.

`Shape(A, nrow, <ncol>, < <<bycol>)`

Description

Reshapes the matrix *A* across rows to the specified dimensions. Each value from the matrix *A* is placed into the reshaped matrix. By default, the values are placed row-by-row.

Returns

The reshaped matrix.

Arguments

`A` A matrix.

`nrow` The number of rows that the new matrix should have.

`ncol` (Optional) The number of columns the new matrix should have.

`<<bycol` (Optional) Specifies that the values be placed into the reshaped matrix column-by-column, instead of row-by-row.

Notes

If *ncol* is not specified, the number of columns is whatever is necessary to fit all of the original values of the matrix into the reshaped matrix.

If a missing value is specified for *nrow*, the number of rows is whatever is necessary to fit all of the original values of the matrix into the reshaped matrix.

If the new matrix is smaller than the original matrix, the extra values are discarded.

If the new matrix is larger than the original matrix, the values are repeated to fill the new matrix.

Examples

```
a = Matrix({ {1, 2, 3}, {4, 5, 6}, {7, 8, 9} });
   [  1 2 3,
      4 5 6,
      7 8 9]

Shape(a, 2);
   [  1 2 3 4 5,
      6 7 8 9 1]

Shape(a, 2, 2);
   [  1 2,
      3 4]

Shape(a, 4, 4);
```

```
[  1 2 3 4,
   5 6 7 8,
   9 1 2 3,
   4 5 6 7]

Shape(a, 4, 4, <<bycol);
[  1 5 9 4,
   2 6 1 5,
   3 7 2 6,
   4 8 3 7]
```

Solve(A, b)

Description

Solves a linear system. In other words, `x=inverse(A)*b`.

Sort Ascending(source)

Description

Returns a copy of a list or matrix *source* with the items in ascending order.

Sort Descending(source)

Description

Returns a copy of a list or matrix *source* with the items in descending order.

Sparse SVD(X, <nSingularValues=min(nRow, nCol)>,<tolerance=1e-10>)

Description

Computes the singular value decomposition of matrix X using the implicitly restarted, partially reorthogonalized Lanczos method for sparse matrices.

Returns

Returns a list (U, M, V) such that U*diag(M)*V is equal to x.

Spline Coef(x, y, lambda)

Description

Returns a five column matrix of the form `knots||a||b||c||d` where `knots` is the unique values in *x*.

`x` is a vector of regressor variables, `y` is the vector of response variables, and `lambda` is the smoothing argument. Larger values for `lambda` result in smoother splines.

Spline Eval(x, coef)

Description

Evaluates the spline predictions using the `coef` matrix in the same form as returned by `SplineCoef()`, in other words, `knots||a||b||c||d`. The `x` argument can be a scalar or a matrix of values to predict. The number of columns of `coef` can be any number greater than 1 and each is used for the next higher power. The powers of `x` are centered at the knot values. For example, the calculation for `coef` of `knots||a||b||c||d` is `j` is such that `knots[j]` is the largest knot smaller than `x`.

`xx = x-knots[j]` is the centered *x* value:

```
result = a[j] + xx * (b[j] + xx * (c[j] + xx * d[j]))
```

The following line is equivalent:

```
result = a[j] + b[j] * xx + c[j] * xx ^ 2 + d[j] * xx ^ 3
```

Spline Smooth(x, y, lambda)

Description

Returns the smoothed predicted values from a spline fit.

`x` is a vector of regressor variables, `y` is the vector of response variables, and `lambda` is the smoothing argument. Larger values for `lambda` result in smoother splines.

SVD(A)

Description

Singular value decomposition.

Sweep(A, <indices>)

Description

Sweeps, or inverts a matrix a partition at a time.

Trace(A)

Description

The trace, or the sum of the diagonal elements of a square matrix.

Transpose(A)

Description

Transposes the rows and columns of the matrix *A*.

Returns

The transposed matrix.

Arguments

A A matrix.

Equivalent Expression

A‘

`V Concat(A, B, ...)`

Description

Vertical concatenation of two or more matrices.

Returns

The new matrix.

Arguments

Two or more matrices.

`V Concat To(A, B, ...)`

Description

Vertical concatenation in place. This is an assignment operator.

Returns

The new matrix.

Arguments

Two or more matrices.

`V Max(matrix)`

Description

Returns a row vector containing the maximum of each column of *matrix*.

`V Mean(matrix)`

Description

Returns a row vector containing the mean of each column of *matrix*.

`V Median(matrix)`

Description

Returns a row vector containing the median of each column of *matrix*.

`V Min(matrix)`

Description

Returns a row vector containing the minimum of each column of *matrix*.

V Quantile(matrix, p)

Description

Returns a row vector containing the p^{th} quantile of each column of *matrix*.

V Standardize(matrix)

Description

Returns a matrix column-standardized to mean = 0 and standard deviation = 1.

V Std(matrix)

Description

Returns a row vector containing the standard deviations of each column of *matrix*.

V Sum(matrix)

Description

Returns a row vector containing the sum of each column of *matrix*.

Varimax(matrix, <norm=1>)

Description

Performs a varimax rotation.

Returns

A list that contains the rotated matrix and the orthogonal rotation matrix.

Arguments

matrix A matrix to be rotated.

norm Specify 1 to perform a normalized rotation, and specify 0 to perform a non-normalized rotation. The default value is 1.

Vec Diag(A)

Description

Creates a vector from the diagonals of a square matrix *A*.

Returns

The new matrix.

Arguments

A A square matrix.

Notes

Using a matrix that is not square results in an error.

`Vec Quadratic(symmetric matrix, rectangular matrix)`

Description

Constructs an *n*-by-*m* matrix. Used in calculation of hat values.

Returns

The new matrix.

Arguments

Two matrices. The first must be symmetric.

Equivalent Expression

```
Vec Diag(X*Sym*X‘)
```

`VPTree(matrix)`

Description

Returns a table that is used for efficiently looking up nearest neighbors. The vantage-point tree algorithm is particularly useful for wide data problems.

Returns

A VPTree object.

Argument

`matrix` A matrix of *k*-dimensional points. The number of dimensions or points is not limited. Each column in the matrix represents a dimension to the data, and each row represents a data point.

Numeric Functions

`Abs(n)`

Description

Calculates the absolute value of *n*.

Returns

Returns a positive number of the same magnitude as the value of *n*.

Argument

`n` Any number.

`Ceiling(n)`

Description

If *n* is not an integer, rounds *n* to the next highest integer.

Returns

Returns the smallest integer greater than or equal to *n*.

Argument

`n` Any number.

`Derivative(expr, {name, ...}, ...)`

Description

Calculates the derivative of the *expr* expression with respect to *name*.

Returns

Returns the derivative.

Arguments

`expr` Any expression. Indirect arguments (for example, `Name Expr`, `Expr`, `Eval`) are supported.

`name` Can be a single variable or a list of variables.

Notes

Adding an additional variable (`Derivative(expr, name, name2)`) takes the second derivative.

`Floor(n)`

Description

If *n* is not an integer, rounds *n* to the next lowest integer.

Returns

Returns the largest integer less than or equal to *n*.

Argument

`n` Any number.

Examples

```
Floor( 2.7 );
  2
Floor( -.5 );
  -1
```

`Integrate(expr, varname, lowLimit, upLimit, <<Tolerance(1e-10), <<StoreInfo({list}), <<StartingValue(val))`

Description

Integrates an expression with respect to a scalar value, using the adaptive quadrature method from Gander and Gautschi (2000).

Arguments

`expr` an expression that defines the integrand.

`varname` the name of the variable of integration. If this variable contains a value, that value specifies a starting value that is used as a typical value to improve the accuracy of the integral.

`lowLimit` specifies the lower limit of integration. To specify negative infinity as the lower limit of integration, set this to missing.

`upLimit` specifies the upper limit of integration. To specify positive infinity as the upper limit of integration, set this to missing.

`StoreInfo` saves diagnostics of the numerical integration routine to the argument of StoreInfo().

`StartingValue` specifies a starting value that is used as a typical value to improve the accuracy of the integral.

`Invert Expr(expr, name)`

Description

Attempts to unfold *expr* around *name*.

`Mod()`

See "Modulo(number, divisor)"

`Modulo(number, divisor)`

`Mod(number, divisor)`

Description

Returns the remainder when *number* is divided by *divisor*.

Examples

```
Modulo( 6, 5 );
  1
```

`Normal Integrate(muVector, sigmaMatrix, expr, x, nStrata, nSim)`

Description

Returns the result of radial-spherical integration for smooth functions of multivariate, normally distributed variables.

Arguments

`muVector` A vector.

`sigmaMatrix` A matrix.

`expr` An expression in terms of the variable *x*.

`x` The variable used in the expression *expr*.

`nStrata` Number of strata.

`nSim` Number of simulations.

`Num Deriv(f(x,...), <parnum=1>)`

Description

Returns the numerical derivative of the *f(x,...)* function with respect to one of its arguments. You can specify that argument as the second argument in the *Num Deriv* function. If no second argument is specified, the derivative is taken with respect to the function's first argument. The derivative is evaluated using numeric values specified in the *f(x,...)* function expression.

Notes

The `Num Deriv()` function might appear not to produce the correct results as seen here:

```
x = 3;
n = Num Deriv( 3 * x ^ 2 );
    // 9.00000000001455
```

The preceding usage is not correct. The function was designed to be used in the Nonlinear platform to differentiate functions for which it does not know the analytic derivatives. The proper usage takes the following form:

```
x = 3;
f = Function( {x}, 3 * x ^ 2 );
n = Num Deriv( f( x ), 1 );
   // 18.000029999854
```

`Num Deriv2(f(x,...))`

Description

Returns the numerical second derivative of the *f(x,...)* function with respect to *x*. The derivative is evaluated using numeric values specified in the *f(x,...)* function expression.

`Round(n, places)`

Description

Rounds *n* to number of decimal *places* given.

`Simplify Expr(expr(expression))`

`Simplify Expr(nameExpr(global))`

Description

Algebraically simplifies an expression

Optimization Functions

Constrained Maximize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)

Description

Finds the values for the *x* arguments, specified as a list, that maximize the *expr* expression with optional linear constraints. You must either specify lower and upper bounds in parentheses for each argument or with the optional `Set Variable Limit()` message. The *x* arguments can be scalar values or vectors.

In the following messages, `A` is a matrix of coefficients. `x = [x1, x2, ...]` is the vector of arguments. `b` is a vector that forms the right side of the expression.

Messages

`<<Less than EQ({A, b})` Sets the constraint to less than or equal to the specified values `(A*x <= b)`.

`<<Greater Than EQ({A, b})` Sets the constraint to greater than or equal to the specified values `(A*x >= b)`.

`<<Equal To({A, b})` Sets the constraint as equal to the specified values `(A*x = b)`.

`<<Starting Values([x1Start, x2Start, ...])` Specifies a starting point.

`<<Max Iter(int)` An integer that specifies the maximum number of iterations to be performed.

`<<Tolerance(p)` *p* sets the tolerance for the convergence criterion. The default tolerance is 10^{-5}.

`<<Show Details("true")` Returns a list with the final values for (objective value, number of iterations, gradient, and Hessian). Shows the step-by-step results of the optimizer in the log.

`<<SetVariableLimit({low,high})` Specifies vectors for the lower and upper limits for the optimization variables.

Constrained Minimize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)

Description

Finds the values for the *x* arguments, specified as a list, that minimize the *expr* expression with optional linear constraints. You must either specify lower and upper bounds in parentheses for each argument or with the optional `Set Variable Limit()` message. The *x* arguments can be scalar values or vectors.

In the following messages, `A` is a matrix of coefficients. `x = [x1, x2, ...]` is the vector of arguments. `b` is a vector that forms the right side of the expression.

Messages

`<<Less than EQ({A, b})` Sets the constraint to less than or equal to the specified values `(A*x <= b)`.

<<Greater Than EQ({A, b}) Sets the constraint to greater than or equal to the specified values (A*x >= b).

<<Equal To({A, b}) Sets the constraint as equal to the specified values (A*x = b).

<<Starting Values([x1Start, x2Start, ...]) Specifies a starting point.

<<Max Iter(int) An integer that specifies the maximum number of iterations to be performed.

<<Tolerance(p) *p* sets the tolerance for the convergence criterion. The default tolerance is 10^{-5}.

<<Show Details("true") Returns a list with the final values for (objective value, number of iterations, gradient, and Hessian). Shows the step-by-step results of the optimizer in the log.

<<SetVariableLimit({low,high}) Specifies vectors for the lower and upper limits for the optimization variables.

Desirability(yVector, desireVector, y)

Description

Fits a function to go through the three points, suitable for defining the desirability of a set of response variables (*y*s). *yVector* and *desireVector* are matrices with three values, corresponding to the three points defining the desirability function. The actual function depends on whether the desire values are in the shape of a larger-is-better, smaller-is-better, target, or antitarget.

Returns

The desirability function.

Arguments

yVector Three input values.

desireVector the corresponding three desirability values.

y the value of which to calculate the desirability.

LPSolve(A, b, c, L, U, neq, nle, nge, <slackVars(Boolean)>)

Description

Returns a list containing the decision variables (and slack variables if applicable) in the first list item and the optimal objective function value (if one exists) in the second list item.

Arguments

A A matrix of constraint coefficients.

b A matrix that is a column of right hand side values of the constraints.

c A vector of cost coefficients of the objective function.

L, U Matrices of lower and upper bounds for the variables.

neq The number of equality constraints.

`nle` The number of less than or equal inequalities.

`nge` The number of greater than or equal inequalities.

`slackVars(Boolean)` (Optional) Determines whether the slack variables are returned in addition to the decision variables. The default value is `0`.

Notes

The constraints must be listed as equalities first, less than or equal inequalities next, and greater than or equal inequalities last.

Maximize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)

Description

Finds the values for the *x* arguments, specified as a list, that maximize the expression *expr*. You can specify lower and upper bounds in parentheses for each argument. Additional arguments for the function enable you to set the maximum number of iterations, tolerance for convergence, and view more details about the optimization. The Newton-Raphson method is used when an analytical derivative is found for the Hessian. Otherwise, the Symmetric-Rank One method (SR1), a quasi-Newton method, is used.

Messages

`<<Max Iter(int)` An integer that specifies the maximum number of iterations to be performed. The default maximum number of iterations is 250.

`<<Tolerance(p)` *p* sets the tolerance for the convergence criterion. The default tolerance is 10^{-8}.

`<<Details("both" | "displaySteps" | "returnDetails")` Specifies what output is returned. If "displaySteps" is specified, step-by-step results of the optimization appear in the Log window. If "returnDetails" is specified, the function returns a list that contains the final values for the objective value, number of iterations, gradient, and Hessian. Specify "both" to get the return value and the results in the Log.

`<<Gradient(exprList)` Specifies a list of expressions that define the analytical gradient that is used for the optimization. Each expression in the list represents a derivative of the expression *expr*.

`<<Hessian(exprList)` Specifies a list of expressions that define the analytical Hessian that is used for the optimization. Each expression in the list represents the upper triangular portion of the Hessian matrix in row-major order.

`<<Method(NR | SR1)` Specifies either the Newton-Raphson (NR) method or the Symmetric-Rank One (SR1) method for the optimization method.

`<<UseNumericDeriv("true")` Specifies that the optimization use a numeric approximation.

`Minimize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)`

Description

Finds the values for the *x* arguments, specified as a list, that minimize the expression *expr*. You can specify lower and upper bounds in parentheses for each argument. Additional arguments for the function enable you to set the maximum number of iterations, tolerance for convergence, and view more details about the optimization. The Newton-Raphson method is used when an analytical derivative is found for the Hessian. Otherwise, the Symmetric-Rank One method (SR1), a quasi-Newton method, is used.

Messages

`<<Max Iter(int)` An integer that specifies the maximum number of iterations to be performed. The default maximum number of iterations is 250.

`<<Tolerance(p)` *p* sets the tolerance for the convergence criterion. The default tolerance is 10^{-8}.

`<<Details("both" | "displaySteps" | "returnDetails")` Specifies what output is returned. If "displaySteps" is specified, step-by-step results of the optimization appear in the Log window. If "returnDetails" is specified, the function returns a list that contains the final values for the objective value, number of iterations, gradient, and Hessian. Specify "both" to get the return value and the results in the Log.

`<<Gradient(exprList)` Specifies a list of expressions that define the analytical gradient that is used for the optimization. Each expression in the list represents a derivative of the expression *expr*.

`<<Hessian(exprList)` Specifies a list of expressions that define the analytical Hessian that is used for the optimization. Each expression in the list represents the upper triangular portion of the Hessian matrix in row-major order.

`<<Method(NR | SR1)` Specifies either the Newton-Raphson (NR) method or the Symmetric-Rank One (SR1) method for the optimization method.

`<<UseNumericDeriv("true")` Specifies that the optimization use a numeric approximation.

Probability Functions

`Beta Density(x, alpha, beta, <theta=0>, <sigma=1>)`

Description

Returns the probability density function (pdf) evaluated at *x* of the beta distribution. The pdf is parameterized as follows:

$$f(x) = \frac{1}{B(\alpha, \beta)\sigma^{\alpha+\beta-1}}(x-\theta)^{\alpha-1}(\theta+\sigma-x)^{\beta-1}$$

where B(·) is the Beta function.

Arguments

`x` A quantile at which the pdf is evaluated. *x* must be between *theta* and *theta* + *sigma*.

`alpha, beta` Shape parameters α and β, which must both be greater than 0.

`theta` Optional threshold parameter θ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

Notes

The beta distribution is useful for modeling the probabilistic behavior of random variables that are constrained to fall in the interval [0, 1], such as proportions.

`Beta Distribution(x, alpha, beta, <theta=0>, <sigma=1>)`

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the beta distribution. The cdf uses the same parameterization as the `Beta Density()` function.

Arguments

`x` A quantile at which the cdf is evaluated. *x* must be between *theta* and *theta* + *sigma*.

`alpha, beta` Shape parameters α and β, which must both be greater than 0.

`theta` Optional threshold parameter θ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

`Beta Quantile(p, alpha, beta, <theta=0>, <sigma=1>)`

Description

Returns the p^{th} quantile from a beta distribution with shape arguments *alpha* and *beta*. The quantile function does not have a closed form equation.

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`alpha, beta` Shape parameters α and β, which must both be greater than 0.

`theta` Optional threshold parameter θ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

`Cauchy Density(q, <center=0>, <scale=1>)`

Description

Returns the probability density function (pdf) evaluated at *q* of a Cauchy distribution. The pdf is parameterized as follows:

$$f(q) = \frac{1}{\pi\sigma}\frac{1}{1+\left(\frac{q-\mu}{\sigma}\right)^2}$$

Arguments

`q` A quantile at which the pdf is evaluated.

`center` Optional location parameter μ. The default is 0.

`scale` Optional scale parameter, σ, which must be greater than 0. The default is 1.

Cauchy Distribution(q, <center=0>, <scale=1>)

Description

Returns the cumulative distribution function (cdf) probability that a Cauchy distributed random variable is less than *q*. The cdf is parameterized as follows:

$$F(q) = \frac{1}{2} + \frac{1}{\pi}\arctan\left(\frac{x-\mu}{\sigma}\right)$$

Arguments

`q` A quantile at which the cdf is evaluated.

`center` Optional location parameter μ. The default is 0.

`scale` Optional scale parameter, σ, which must be greater than 0. The default is 1.

Cauchy Quantile(p, <center=0>, <scale=1>)

Description

Returns the p^{th} quantile from a Cauchy distribution. The p^{th} quantile is the value for which the probability is *p* that a random value would be less than or equal to *p*. The quantile function is parameterized as follows:

$$F^{-1}(p) = \sigma\tan\left[\pi\left(p+\frac{1}{2}\right)\right] + \mu$$

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`center` Optional location parameter μ. The default is 0.

`scale` Optional scale parameter σ, which must be greater than 0. The default is 1.

ChiSquare Density(q, df, <nc=0>)

Description

Returns the probability density function (pdf) evaluated at *q* of the chi-square distribution. The pdf is parameterized as follows:

$$f(q) = \exp(-\lambda/2) \sum_{r=0}^{\infty} \frac{(\lambda/2)^r}{r!} f_{n+2r}(q)$$

where $f_{n+2r}(q)$ is the density of a central chi-square distribution with $n+2r$ degrees of freedom.

Arguments

q A quantile at which the pdf is evaluated. *q* must be greater than or equal to 0.

df The degrees of freedom *n*, which must be greater than 0.

nc Optional noncentrality parameter λ, which must be nonnegative. The default is 0.

ChiSquare Distribution(q, df, <nc=0>)

Description

Returns cumulative distribution function at quantile *x* for chi-square with *df* degrees of freedom centered at *nc*. The cdf is parameterized as follows:

$$F(q) = \exp(-\lambda/2) \sum_{r=0}^{\infty} \frac{(\lambda/2)^r}{r!} F_{n+2r}(q) \quad ,$$

where $F_{n+2r}(q)$ is the cumulative distribution of a central chi-square distribution with $n+2r$ degrees of freedom.

Arguments

q A quantile at which the cdf is evaluated. *q* must be greater than or equal to 0.

df The degrees of freedom *n*, must be greater than 0.

nc The optional noncentrality parameter λ, must be nonnegative. The default is 0.

ChiSquare Log CDistribution(x, df, <nc=0>)

Description

Returns the log of (1 - value), where value is the cumulative distribution function evaluated at *x* of the chi-square distribution with *df* degrees of freedom and noncentrality parameter *nc*.

ChiSquare Log Density(x, df, <nc=0>)

Description

Returns the log of the value of the probability density function evaluated at *x* of the chi-square distribution with *df* degrees of freedom and noncentrality parameter *nc*.

ChiSquare Log Distribution(x, df, <nc=0>)

Description

Returns the log of the value of the cumulative distribution function evaluated at quantile *x* of the chi-square distribution with *df* degrees of freedom and noncentrality parameter *nc*.

ChiSquare Noncentrality(x, df, prob)

Description

Returns the chi-square distribution noncentrality parameter *nc* that satisfies the following:

prob = ChiSquare Distribution(*x, df, nc*)

Arguments

x A quantile at which the cdf is evaluated.

df The degrees of freedom *n,* which must be greater than 0.

prob The probability of the quantile desired; *prob* must be between 0 and 1.

ChiSquare Quantile(p, df, <nc=0>)

Description

Returns the p^{th} quantile from a chi-square distribution with *df* degrees of freedom, centered at *nc*. The quantile function does not have a closed form equation.

Arguments

p The probability of the quantile desired. *p* must be between 0 and 1.

df The degrees of freedom *n,* which must be greater than 0.

nc Optional noncentrality parameter λ, which must be nonnegative. The default is 0.

Dunnett P Value(q, nTrt, dfe, <lambdaVec=.>)

Description

Returns the *p*-value from Dunnett's multiple comparisons test.

Arguments

q A number that is the test statistic.

nTrt The number of treatments being compared to the control treatment.

dfe The error degrees of freedom.

`lambdaVec` A vector of parameters. If *`lambdaVec`* is missing (`.`), each of the parameters is set to 1/Sqrt(2).

`Dunnett Quantile(1-alpha, nTrt, dfe, <lambdaVec=.>)`

Description

Returns the quantile used in Dunnett's multiple comparisons test.

Arguments

`1-alpha` A number that is the confidence level.

`nTrt` The number of treatments being compared to the control treatment.

`dfe` The error degrees of freedom.

`lambdaVec` A vector of parameters. If *`lambdaVec`* is missing (`.`), each of the parameters is set to 1/Sqrt(2).

`Exp Density(x, <theta=1>)`

Description

Returns the probability density function (pdf) evaluated at *x* of the exponential distribution. The pdf is parameterized as follows:

$$f(x) = \frac{1}{\theta}\exp(-x/\theta)$$

Arguments

`x` A quantile at which the pdf is evaluated. *x* must be greater than or equal to 0.

`theta` Optional scale parameter θ, which must be greater than 0. The default is 1.

`Exp Distribution(x, <theta=1>)`

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the exponential distribution. The cdf is parameterized as follows:

$$F(x) = 1 - \exp(-x/\theta)$$

Arguments

`x` A quantile at which the cdf is evaluated. *x* must be greater than or equal to 0.

`theta` Optional scale parameter θ, which must be greater than 0. The default is 1.

Exp Quantile(p, <theta=1>)

Description

Returns the p^{th} quantile from an exponential distribution with scale parameter *theta*. The quantile function is parameterized as follows:

$$F^{-1}(p) = -\theta\log(1-p)$$

Arguments

p The probability of the quantile desired. *p* must be between 0 and 1.

theta Optional scale parameter θ, which must be greater than 0. The default is 1.

F Density(x, dfnum, dfden, <nc=0>)

Description

Returns the probability density function (pdf) evaluated at *x* for the F distribution with numerator and denominator degrees of freedom *dfnum* and *dfden*, with optional noncentrality parameter *nc*.

$$f(x) = \exp(-\lambda/2)\sum_{r=0}^{\infty}\frac{(\lambda/2)^r}{\mathrm{B}\left(\frac{v_2}{2},\frac{v_1}{2}+r\right)r!}\left(\frac{v_1}{v_2}\right)^{\frac{v_1}{2}+r}\left(1+\frac{v_1}{v_2}x\right)^{-\left(\frac{v_1+v_2}{2}+r\right)}x^{\frac{v_1}{2}-1+r}$$

where B(·) is the Beta function.

Arguments

x A quantile at which the pdf is evaluated. *x* must be greater than 0.

dfnum The degrees of freedom, v_1, of the chi-square distribution in the numerator of the *F*-distribution. *dfnum* must be greater than 0.

dfden The degrees of freedom, v_2, of the chi-square distribution in the denominator of the *F*-distribution. *dfden* must be greater than 0.

nc Optional noncentrality parameter λ, which must be nonnegative. The default is 0.

F Distribution(x, dfnum, dfden, <nc=0>)

Description

Returns the cumulative distribution function (cdf) evaluated at *x* for the F distribution with numerator and denominator degrees of freedom *dfnum* and *dfden* and noncentrality parameter *nc*.

F Log CDistribution(x, dfnum, dfden, <nc=0>)

Description

Returns the log of (1 - value), where value is the cumulative distribution function evaluated at *x* of the F distribution with numerator and denominator degrees of freedom *dfnum* and *dfden*, with optional noncentrality parameter *nc*.

F Log Density(x, dfnum, dfden, <nc=0>)

Description

Returns the log of the value of the probability density function (pdf) evaluated at *x* for the F distribution with numerator and denominator degrees of freedom *dfnum* and *dfden*, with optional noncentrality parameter *nc*.

F Log Distribution(x, dfnum, dfden, <nc=0>)

Description

Returns the log of the value of the cumulative distribution function (cdf) evaluated at *x* for the F distribution with numerator and denominator degrees of freedom *dfnum* and *dfden* and noncentrality parameter *nc*.

F Noncentrality(x, dfnum, dfden, prob)

Description

Returns the F distribution noncentrality parameter *nc* that satisfies the following:

prob = F Distribution(*x, dfnum, dfden, nc*)

See Also

"F Distribution(x, dfnum, dfden, <nc=0>)"

F Power(alpha, dfh, dfm, d, n)

Description

Returns the power from a given situation involving an *F* test or a *t* test.

Arguments

`alpha` The significance level of the test. *alpha* must be between 0 and 1.

`dfh` The hypothesis degrees of freedom. *dfh* must be greater than 0.

`dfm` The degrees of freedom in the whole model. *dfm* must be greater than 0.

`d` The squared effect size, defined as Δ^2/σ^2. In this equation, σ^2 is the error variance and Δ^2 is defined as follows:

$\Delta^2 = (\bar{x} - \mu)^2$ for a one-sample *t* test

$$\Delta^2 = \frac{(\bar{x}_1 - \bar{x}_2)^2}{4} \text{ for a two-sample } t \text{ test}$$

$$\Delta^2 = \sqrt{\sum_{i=1}^{k} \frac{(\bar{x}_i - \bar{x})^2}{k}} \text{ for a } k\text{-sample } F \text{ test}$$

n The total number of observations. *n* must be greater than *dfm*.

F Quantile(x, dfnum, dfden, <nc=0>)

Description

Returns the p^{th} quantile from the F distribution with numerator and denominator degrees of freedom *dfnum* and *dfden* and noncentrality parameter *nc*.

F Sample Size(alpha, dfh, dfm, d, power)

Description

Returns the sample size from a given situation involving an *F* test or a *t* test.

Arguments

alpha The significance level of the test. *alpha* must be between 0 and 1.

dfh The hypothesis degrees of freedom. *dfh* must be greater than 0.

dfm The degrees of freedom in the whole model. *dfm* must be greater than 0.

d The squared effect size, defined as Δ^2/σ^2. In this equation, σ^2 is the error variance and Δ^2 is defined as follows:

$$\Delta^2 = (\bar{x} - \mu)^2 \text{ for a one-sample } t \text{ test}$$

$$\Delta^2 = \frac{(\bar{x}_1 - \bar{x}_2)^2}{4} \text{ for a two-sample } t \text{ test}$$

$$\Delta^2 = \sqrt{\sum_{i=1}^{k} \frac{(\bar{x}_i - \bar{x})^2}{k}} \text{ for a } k\text{-sample } F \text{ test}$$

power The desired power for the test.

Frechet Density(x, mu, sigma)

Description

Returns the probability density function (pdf) evaluated at *x* of the Fréchet distribution. The pdf is parameterized as follows:

$$f(x) = \exp\left[-\exp\left(-\frac{\log(x)-\mu}{\sigma}\right)\right]\exp\left(-\frac{\log(x)-\mu}{\sigma}\right)\frac{1}{x\sigma}$$

Arguments

`x` A quantile at which the pdf is evaluated. *x* must be greater than 0.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`Frechet Distribution(x, mu, sigma)`

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the Fréchet distribution. The cdf is parameterized as follows:

$$F(x) = \exp\left[-\exp\left(-\frac{\log(x)-\mu}{\sigma}\right)\right]$$

Arguments

`x` A quantile at which the cdf is evaluated. *x* must be greater than 0.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`Frechet Quantile(p, mu, sigma)`

Description

Returns the p^{th} quantile from a Fréchet distribution with location *mu* and scale *sigma*. The quantile function is parameterized as follows:

$$F^{-1}(p) = \exp[-\sigma\log\{-\log(p)\} + \mu]$$

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`Gamma Density(x, <alpha=1>, <scale=1>, <threshold=0>)`

Description

Returns the probability density function (pdf) evaluated at *x* of the Gamma distribution. The pdf is parameterized as follows:

$$f(x) = \frac{1}{\Gamma(\alpha)\beta^{\alpha}}(x-\theta)^{\alpha-1}\exp(-(x-\theta)/\beta)$$

Arguments

`x` A quantile at which the pdf is evaluated. *x* must be greater than θ.

`alpha` Optional shape parameter α, which must be greater than 0. The default is 1.

`scale` Optional scale parameter β, which must be greater than 0. The default is 1.

`threshold` Optional threshold parameter θ. The default is 0.

`Gamma Distribution(x, <alpha=1>, <scale=1>, <threshold=0>)`

`IGamma(x, <alpha=1, scale=1, threshold=0>)`

Description

Returns the cumulative distribution function (cdf) evaluated at quantile *x* for the gamma distribution with parameters *alpha*, *scale*, and *threshold*.

`Gamma Log CDistribution(x, <alpha=1>, <scale=1>, <threshold=0>)`

Description

Same as `Log(1 - Gamma Distribution(x, alpha))` except that it has a much greater range.

`Gamma Log Density(x, <alpha=1>, <scale=1>, <threshold=0>)`

Description

Same as `Log(Gamma Density(x, alpha))` except that it has a much greater range.

`Gamma Log Distribution(x, <alpha=1>, <scale=1>, <threshold=0>)`

Description

Same as `Log(Gamma Distribution(x, alpha))` except that it has a much greater range.

`Gamma Quantile(p, <alpha=1>, <scale=1>, threshold>)`

Description

Returns the *p*th quantile from the gamma distribution with the *alpha*, *scale*, and *threshold* parameters given.

GenGamma Density(x, mu, sigma, lambda)

Description

Returns the probability density function (pdf) evaluated at *x* of an extended generalized gamma probability distribution. The pdf is parameterized as follows:

$$f(x) = \begin{cases} \frac{|\lambda|}{x\sigma}\phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda \neq 0 \\ \frac{1}{x\sigma}\phi_{nor}(\omega) & \text{if } \lambda = 0 \end{cases}$$

where $\omega = [\log(x) - \mu]/\sigma$. Note that the following is the pdf for the standardized log-gamma variable with shape parameter $\kappa > 0$:

$$\phi_{lg}(z;\kappa) = \frac{1}{\Gamma(\kappa)}\exp[\kappa z - \exp(z)]$$

Note that $\phi_{nor}(\cdot)$ is the standard normal pdf.

Arguments

`x` A quantile at which the pdf is evaluated. *x* must be greater than 0.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ.

GenGamma Distribution(x, mu, sigma, lambda)

Description

Returns the cumulative distribution function (cdf) of the extended generalized gamma distribution. The cdf is parameterized as follows:

$$F(x) = \begin{cases} \Phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda > 0 \\ \Phi_{nor}(\omega) & \text{if } \lambda = 0 \\ 1 - \Phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda < 0 \end{cases}$$

where $\omega = [\log(x) - \mu]/\sigma$. Note that the following is the cdf for the standardized log-gamma variable with shape parameter $\kappa > 0$:

$$\Phi_{lg}(z;\kappa) = \Gamma_I[\exp(z);\kappa]$$

where $\Gamma_I[\cdot]$ denotes the incomplete gamma function. Note that $\Phi_{nor}(\cdot)$ is the standard normal cdf.

Arguments

`x` A quantile at which the cdf is evaluated. *x* must be greater than 0.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ.

`GenGamma Quantile(p, mu, sigma, lambda)`

Description

Returns the p^{th} quantile from an extended generalized gamma distribution with parameters *mu*, *sigma*, and *lambda*. The quantile function does not have a closed form equation.

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ.

`GLog Density(x, mu, sigma, lambda)`

Description

Returns the probability density function (pdf) evaluated at *x* of a generalized logarithmic distribution. The pdf is parameterized as follows:

$$f(x) = \phi\left\{\frac{1}{\sigma}\left[\log\left(\frac{x+\sqrt{x^2+\lambda^2}}{2}\right)-\mu\right]\right\}\frac{x+\sqrt{x^2+\lambda^2}}{\sigma(x^2+\lambda^2+x\sqrt{x^2+\lambda^2})}$$

where $\phi(\cdot)$ is the standard normal pdf.

Arguments

`x` A quantile at which the pdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ, which must be greater than 0.

Notes

When the shape parameter is equal to zero, the distribution reduces to a Lognormal(μ, σ).

GLog Distribution(x, mu, sigma, lambda)

Description

Returns the probability that a generalized logarithmically distribution random variable is less than *x*. The cdf is parameterized as follows:

$$F(x) = \Phi\left\{\frac{1}{\sigma}\left[\log\left(\frac{x+\sqrt{x^2+\lambda^2}}{2}\right)-\mu\right]\right\}$$

where $\Phi(\cdot)$ is the standard normal cdf.

Arguments

`x` A quantile at which the cdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ, which must be greater than 0.

GLog Quantile(p, mu, sigma, lambda)

Description

Returns the p^{th} quantile from a generalized logarithmic distribution.

IGamma()

See "Gamma Distribution(x, <alpha=1>, <scale=1>, <threshold=0>)".

Johnson Sb Density(q, gamma, delta, theta, sigma)

Description

Returns the probability density function (pdf) evaluated at *q* of a Johnson Sb distribution. The pdf is parameterized as follows:

$$f(q) = \phi\left[\gamma+\delta\ln\left(\frac{q-\theta}{\sigma-(q-\theta)}\right)\right]\left(\frac{\delta\sigma}{(q-\theta)(\sigma-(q-\theta))}\right)$$

where $\phi(\cdot)$ is the standard normal pdf.

Arguments

`q` A quantile at which the pdf is evaluated. *q* must be in the interval *theta* to *theta* + *sigma*.

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Scale parameter σ, which must be greater than 0.

Johnson Sb Distribution(q, gamma, delta, theta, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *q* of a Johnson Sb distribution. The pdf is parameterized as follows:

$$F(q) = \Phi\left[\gamma + \delta \ln\left(\frac{q-\theta}{\sigma-(q-\theta)}\right)\right]$$

where $\Phi(\cdot)$ is the standard normal cdf.

Arguments

q A quantile at which the cdf is evaluated. *q* must be in the interval *theta* to *theta* + *sigma*.

gamma Shape parameter γ.

delta Shape parameter δ, which must be greater than 0.

theta Location parameter θ.

sigma Scale parameter σ, which must be greater than 0.

Johnson Sb Quantile(p, gamma, delta, theta, sigma)

Description

Returns the p^{th} quantile from a Johnson Sb distribution.

Arguments

p The probability of the quantile desired. *p* must be between 0 and 1.

gamma Shape parameter γ.

delta Shape parameter δ, which must be greater than 0.

theta Location parameter θ.

sigma Scale parameter σ, which must be greater than 0.

Johnson Sl Density(x, gamma, delta, theta, sigma)

Description

Returns the probability density function (pdf) evaluated at *x* of a Johnson Sl distribution. The pdf is parameterized as follows:

$$f(x) = \frac{\delta}{|x-\theta|}\phi\left[\gamma + \delta \ln\left(\frac{x-\theta}{\sigma}\right)\right]$$

where $\phi(\cdot)$ is the standard normal pdf.

Arguments

x A quantile at which the pdf is evaluated. *x* must be greater than *theta* if *sigma* is 1 and less than *theta* if *sigma* is -1.

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Parameter σ that indicates if the distribution is skewed positively or negatively. `sigma` must be equal to either +1 (skewed positively) or -1 (skewed negatively).

Johnson Sl Distribution(q, gamma, delta, theta, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *q* of a Johnson Sl distribution.

$$F(x) = \begin{cases} \Phi\left[\gamma + \delta \ln\left(\frac{x-\theta}{\sigma}\right)\right], & \sigma = 1 \\ 1 - \Phi\left[\gamma + \delta \ln\left(\frac{x-\theta}{\sigma}\right)\right], & \sigma = -1 \end{cases}$$

where $\Phi(\cdot)$ is the standard normal cdf.

Arguments

`q` A quantile at which the cdf is evaluated. *q* must be greater than *theta* if *sigma* is 1 and less than *theta* if *sigma* is -1.

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Parameter σ that defines if the distribution is skewed positively or negatively. `Sigma` must be equal to either +1 (skewed positively) or -1 (skewed negatively).

Johnson Sl Quantile(p, gamma, delta, theta, sigma)

Description

Returns the p^{th} quantile from a Johnson Sl distribution.

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Parameter σ that defines if the distribution is skewed positively or negatively. `Sigma` must be equal to either +1 (skewed positively) or -1 (skewed negatively).

Johnson Su Density(x, gamma, delta, theta, sigma)

Description

Returns the probability density function (pdf) evaluated at *x* of a Johnson Su distribution. The pdf is parameterized as follows:

$$f(x) = \frac{\delta}{\sigma}\left[1 + \left(\frac{x-\theta}{\sigma}\right)^2\right]^{-1/2} \phi\left[\gamma + \delta \sinh^{-1}\left(\frac{x-\theta}{\sigma}\right)\right]$$

where $\phi(\cdot)$ is the standard normal pdf.

Arguments

`x` A quantile at which the pdf is evaluated.

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Scale parameter σ, which must be greater than 0.

Johnson Su Distribution(q, gamma, delta, theta, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *q* of a Johnson Su distribution. The cdf is parameterized as follows:

$$F(x) = \Phi\left[\gamma + \delta \sinh^{-1}\left(\frac{x-\theta}{\sigma}\right)\right]$$

where $\Phi(\cdot)$ is the standard normal cdf.

Arguments

`q` A quantile at which the cdf is evaluated.

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Scale parameter σ, which must be greater than 0.

Johnson Su Quantile(p, gamma, delta, theta, sigma)

Description

Returns the p^{th} quantile from a Johnson Su distribution.

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Scale parameter σ, which must be greater than 0.

LEV Density(x, mu, sigma)

Description

Returns the probability density function (pdf) evaluated at *x* of the largest extreme value distribution with location *mu* and scale *sigma*. The pdf is parameterized as follows:

$$f(x) = \frac{1}{\sigma}\exp\left[-\frac{x-\mu}{\sigma} - \exp\left(-\frac{x-\mu}{\sigma}\right)\right]$$

Arguments

`x` A quantile at which the pdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

LEV Distribution(x, mu, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the largest extreme value distribution with location *mu* and scale *sigma*. The cdf is parameterized as follows:

$$F(x) = \exp\left[-\exp\left(-\frac{x-\mu}{\sigma}\right)\right]$$

Arguments

`x` A quantile at which the cdf is evaluated. *x* must be greater than *sigma*.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

LEV Quantile(p, mu, sigma)

Description

Returns the p^{th} quantile from a largest extreme value distribution with location *mu* and scale *sigma*. The quantile function is parameterized as follows:

$$F^{-1}(p) = -\sigma\log(-\log(p)) + \mu$$

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

LogGenGamma Density(x, mu, sigma, lambda)

Description

Returns the probability density function (pdf) evaluated at *x* of a log generalized gamma probability distribution with parameters *mu*, *sigma*, and *lambda*. The pdf is parameterized as follows:

$$f(x) = \begin{cases} \frac{|\lambda|}{\sigma}\phi_{lg}[\lambda\omega + \log(\lambda^{-2});\lambda^{-2}] & \text{if } \lambda \neq 0 \\ \frac{1}{\sigma}\phi_{nor}(\omega) & \text{if } \lambda = 0 \end{cases}$$

where $\omega = [x - \mu]/\sigma$. Note that the following is the pdf for the log-gamma variable with shape parameter $\kappa > 0$:

$$\phi_{lg}(z;\kappa) = \frac{1}{\Gamma(\kappa)}\exp[\kappa z - \exp(z)]$$

Note that $\phi_{nor}(\cdot)$ is the standard normal pdf.

Arguments

`x` A quantile at which the pdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ.

LogGenGamma Distribution(x, mu, sigma, lambda)

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the log generalized gamma distributed random variable (with parameters *mu*, *sigma*, and *lambda*) The cdf is parameterized as follows:

$$F(x) = \begin{cases} \Phi_{lg}[\lambda\omega + \log(\lambda^{-2});\lambda^{-2}] & \text{if } \lambda > 0 \\ \Phi_{nor}(\omega) & \text{if } \lambda = 0 \\ 1 - \Phi_{lg}[\lambda\omega + \log(\lambda^{-2}); \quad \lambda^{-2}] & \text{if } \lambda < 0 \end{cases}$$

where $\omega = [x - \mu]/\sigma$. Note that the following is the cdf for the log-gamma variable with shape parameter $\kappa > 0$:

$$\Phi_{lg}(z;\kappa) = \Gamma_I[\exp(z);\kappa]$$

where $\Gamma_I[\cdot]$ denotes the incomplete gamma function. Note that $\Phi_{nor}(\cdot)$ is the standard normal cdf.

Arguments

`x` A quantile at which the cdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ.

`LogGenGamma Quantile(p, mu, sigma, lambda)`

Description

Returns the p^{th} quantile from a log generalized gamma distribution.

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ.

`Logistic Density(x, mu, sigma)`

Description

Returns the probability density function (pdf) evaluated at *x* of a logistic distribution with location *mu* and scale *sigma*. The pdf is parameterized as follows:

$$f(x) = \frac{1}{\sigma}\frac{\exp\left(-\frac{x-\mu}{\sigma}\right)}{\left[1+\exp\left(-\frac{x-\mu}{\sigma}\right)\right]^2}$$

Arguments

`x` A quantile at which the pdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`Logistic Distribution(x, mu, sigma)`

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the logistic distribution with location *mu* and scale *sigma*. The cdf is parameterized as follows:

$$F(x) = \frac{1}{\left[1 + \exp\left(-\frac{x-\mu}{\sigma}\right)\right]}$$

Arguments

`x` A quantile at which the cdf is evaluated. *x* must be greater than σ.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`Logistic Quantile(p, mu, sigma)`

Description

Returns the p^{th} quantile from a logistic distribution with location *mu* and scale *sigma*. The quantile function is parameterized as follows:

$$F^{-1}(p) = -\sigma\log\left(\frac{1}{p} - 1\right) + \mu$$

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`Loglogistic Density(x, mu, sigma)`

Description

Returns the probability density function (pdf) evaluated at *x* of a loglogistic distribution with location *mu* and scale *sigma*. The pdf is parameterized as follows:

$$f(x) = \frac{1}{x\sigma}\frac{\exp\left(\frac{\log(x)-\mu}{\sigma}\right)}{\left[1 + \exp\left(\frac{\log(x)-\mu}{\sigma}\right)\right]^2}$$

Arguments

`x` A quantile at which the pdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

Loglogistic Distribution(x, mu, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the loglogistic distribution with location *mu* and scale *sigma*. The cdf is parameterized as follows:

$$F(x) = \frac{1}{1 + \exp\left(-\frac{\log(x) - \mu}{\sigma}\right)}$$

Arguments

x A quantile at which the cdf is evaluated.

mu The location parameter μ.

sigma The scale parameter σ, which must be greater than 0.

Loglogistic Quantile(p, mu, sigma)

Description

Returns the p^{th} quantile from a loglogistic distribution with location *mu* and scale *sigma*. The quantile function is parameterized as follows:

$$F^{-1}(p) = \exp\left[-\sigma\log\left(\frac{1}{p} - 1\right) + \mu\right]$$

Arguments

p The probability of the quantile desired. *p* must be between 0 and 1.

mu The location parameter μ.

sigma The scale parameter σ, which must be greater than 0.

Lognormal Density(x, mu, sigma)

Description

Returns the probability density function (pdf) evaluated at *x* of a lognormal distribution with location *mu* and scale *sigma*. The pdf is parameterized as follows:

$$f(x) = \frac{1}{x}\phi\left[\frac{\log(x) - \mu}{\sigma}\right]$$

where $\phi(\cdot)$ is the standard normal pdf.

Arguments

x A quantile at which the pdf is evaluated. *x* must be greater than or equal to 0.

mu The location parameter μ.

sigma The scale parameter σ, which must be greater than 0.

Lognormal Distribution(x, mu, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of a lognormal distribution with location *mu* and scale *sigma*. The cdf is parameterized as follows:

$$F(x) = \Phi\left[\frac{\log(x) - \mu}{\sigma}\right]$$

where $\Phi(\cdot)$ is the standard normal cdf.

Arguments

x A quantile at which the pdf is evaluated. *x* must be greater than or equal to 0.

mu The location parameter μ.

sigma The scale parameter σ, which must be greater than 0.

Lognormal Quantile(x, mu, sigma)

Description

Returns the p^{th} quantile of a lognormal distribution with location *mu* and scale *sigma*.

Normal Biv Distribution(x, y, r, <mu1>, <s1>, <mu2>, <s2>)

Description

Computes the probability that an observation (X, Y) is less than or equal to (*x, y*) with correlation coefficient *r* where X is individually normally distributed with mean *mu1* and standard deviation *s1* and Y is individually normally distributed with mean *mu2* and standard deviation *s2*. If *mu1*, *s1*, *mu2*, and *s2* are not given, the function assumes the standard normal bivariate distribution with *mu1*=0, *s1*=1, *mu2*=0, and *s2*=1.

Normal Density(x, <mean=0>, <stddev=1>)

Description

Returns the probability density function (pdf) evaluated at *x* for the normal distribution with *mean* and *stddev*. The pdf is parameterized as follows:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

Arguments

x A quantile at which the pdf is evaluated.

mu Optional location parameter μ. The default is 0.

sigma Optional scale parameter σ, which must be greater than 0. The default is 1.

Notes

The normal distribution is bell shaped and symmetrical.

`Normal Distribution(x, <mean=0>, <stddev=1>)`

Description

Returns the cumulative distribution function (cdf) evaluated at *x* for the normal distribution with *mean* and *stddev*. The cdf is parameterized as follows:

$$F(x) = \Phi\left(\frac{x-\mu}{\sigma}\right)$$

Note that $\Phi(\cdot)$ is the standard normal cdf, defined as follows:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_0^x \exp\left(-\frac{t^2}{2}\right) dt$$

Arguments

`x` A quantile at which the pdf is evaluated.

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

`Normal Log CDistribution(x, <mean=0>, <std dev=1>)`

Description

Returns 1 - log (value) of the distribution function at quantile *x* for the normal distribution.

`Normal Log Density(x, <mean=0>, <stddev=1>)`

Description

Returns the log of the value of the density function at quantile *x* for the normal distribution with *mean* and *stddev*. The default *mean* is 0. The default *stddev* is 1.

`Normal Log Distribution(x, <mean=0>, <std dev=1>)`

Description

Returns the log of the value of the distribution function at quantile *x* for the normal distribution.

Normal Mixture Density(q, mean, stdev, probability)

Description

Returns the density at *q* of a normal mixture distribution with group means *mean*, group standard deviations *stdev*, and group probabilities *probability*. The *mean*, *stdev*, and *probability* arguments are all vectors of the same size.

Normal Mixture Distribution(q, mean, stdev, probability)

Description

Returns the probability that a normal mixture distributed variable with group means *mean*, group standard deviations *stdev*, and group probabilities *probability* is less than *q*. The *mean*, *stdev*, and *probability* arguments are all vectors of the same size.

Normal Mixture Quantile(p, mean, stdev, probability)

Description

Returns the p^{th} quantile, the values for which the probability is *p* that a random value would be lower. The *mean*, *stdev*, and *probability* arguments are all vectors of the same size.

Normal Quantile(p, <mean=0>, <stddev=1>)

Probit(p, <mean=0>, <stddev=1>)

Description

Returns the p^{th} quantile from the normal distribution with *mean* and *stddev*. The default *mean* is 0. the default *stddev* is 1.

Probit()

See "Normal Quantile(p, <mean=0>, <stddev=1>)".

SEV Density(x, mu, sigma)

Description

Returns the probability density function (pdf) evaluated at *x* of the smallest extreme distribution with location *mu* and scale *sigma*. The pdf is parameterized as follows:

$$f(x) = \frac{1}{\sigma}\exp\left[\frac{x-\mu}{\sigma} - \exp\left(\frac{x-\mu}{\sigma}\right)\right]$$

Arguments

x A quantile at which the pdf is evaluated.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

SEV Distribution(x, mu, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the smallest extreme distribution with location *mu* and scale *sigma*. The cdf is parameterized as follows:

$$F(x) = 1 - \exp\left[-\exp\left(\frac{x-\mu}{\sigma}\right)\right]$$

Arguments

`x` A quantile at which the cdf is evaluated. *x* must be greater than σ.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

SEV Quantile(p, mu, sigma)

Description

Returns the p^{th} quantile of the smallest extreme distribution with location *mu* and scale *sigma*. The quantile function is parameterized as follows:

$$F^{-1}(p) = \sigma\log[-\log(1-p)] + \mu$$

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

SHASH Density(x, gamma, delta, theta, sigma)

Description

Returns the probability density function (pdf) evaluated at *x* of a sinh-arcsinh (SHASH) distribution. The pdf is parameterized as follows:

$$f(x) = \frac{\delta\cosh(w)}{\sqrt{\sigma^2 + (x-\theta)^2}}\phi[\sinh(w)]$$

where

$\phi(\cdot)$ is the standard normal pdf

$$w = \gamma + \delta \sinh^{-1}\left(\frac{x-\theta}{\sigma}\right)$$

Arguments

x A quantile at which the pdf is evaluated.

gamma The shape parameter γ.

delta The shape parameter δ, which must be greater than 0.

theta The location parameter θ.

sigma The scale parameter σ, which must be greater than 0.

SHASH Distribution(x, gamma, delta, theta, sigma)

Description

Returns the cumulative distribution function (cdf) evaluated at *x* of the sinh-arcsinh (SHASH) distribution. The cdf is parameterized as follows:

$$F(x) = \Phi\left[\sinh\left(\gamma + \delta \sinh^{-1}\left(\frac{x-\theta}{\sigma}\right)\right)\right]$$

where $\Phi(\cdot)$ is the standard normal cdf.

Arguments

x A quantile at which the cdf is evaluated.

gamma The shape parameter γ.

delta The shape parameter δ, which must be greater than 0.

theta The location parameter θ.

sigma The scale parameter σ, which must be greater than 0.

SHASH Quantile(p, gamma, delta, theta, sigma)

Description

Returns the p^{th} quantile from a sinh-arcsinh (SHASH) distribution (with parameters *gamma*, *delta*, *theta*, and *sigma*).

Arguments

p The probability of the quantile desired. *p* must be between 0 and 1.

gamma The shape parameter γ.

delta The shape parameter δ, which must be greater than 0.

theta The location parameter θ.

sigma The scale parameter σ, which must be greater than 0.

Students t Density()

See "t Density(x, df, <nc=0>)".

Students t Distribution()

See "t Distribution(q, df, <nc=0>)".

Students t Quantile()

See "t Quantile(p, df, <nc=0>)".

t Density(x, df, <nc=0>)

Students t Density(x, df, <nc=0>)

Description

Returns the probability density function (pdf) evaluated at *x* of the Student's *t* distribution with degrees of freedom *df*. The pdf is parameterized as follows:

$$f(x) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)} \frac{1}{\sqrt{\nu\pi}} \left[1 + \frac{x^2}{\nu}\right]^{-\left(\frac{\nu+1}{2}\right)}$$

Arguments

x A quantile at which the pdf is evaluated.

df The degrees of freedom ν, which must be greater than or equal to 1.

nc The optional noncentrality parameter λ, which must be nonnegative. The default is 0.

t Distribution(q, df, <nc=0>)

Students t Distribution(q, df, <nc=0>)

Description

Returns the probability that a Student's *t* distributed random variable is less than *q*. *nc* defaults to 0.

t Log CDistribution(x, df, <nc=0>)

Description

Returns 1 - log (value) of the normal distribution function at quantile *x* for the *t* distribution.

t Log Density(x, df, <nc=0>)

Description

Returns the log of the value of the density function at quantile *x* for the *t* distribution.

t Log Distribution(x, df, <nc=0>)

Description

Returns the log of the value of the distribution function at quantile *x* for the *t* distribution.

t Noncentrality(x, df, prob)

Description

Returns the *t* distribution noncentrality parameter *nc* that satisfies the following:

prob = T Distribution(*x, df, nc*)

t Quantile(p, df, <nc=0>)

Students t Quantile(p, df, <nc=0>)

Description

Returns the p^{th} quantile from the Student's *t* distribution with degrees of freedom *df*. *nc* defaults to 0.

Tukey HSD P Value(q, n, dfe)

Description

Returns the *p*-value from Tukey's HSD multiple comparisons test.

Arguments

q The test statistic. The test statistic that is specified is Tukey's adjusted critical value, which is the quantile of Tukey's studentized range distribution divided by the square root of 2.

n The number of groups in the study.

dfe The error degrees of freedom, based on the total study sample.

Tukey HSD Quantile(1-alpha, n, dfe)

Description

Returns the quantile used in Tukey's HSD multiple comparisons test. The quantile that is returned is Tukey's adjusted critical value, which is the quantile of Tukey's studentized range distribution divided by the square root of 2.

Arguments

1-alpha The confidence level.

`n` The number of groups in the study.

`dfe` The error degrees of freedom, based on the total study sample.

`Weibull Density(x, shape, <scale=1>, <threshold=0>)`

Description

Returns the probability density function (pdf) evaluated at *x* of the Weibull distribution. The pdf is parameterized as follows:

$$f(x) = \frac{\beta}{\alpha}\left(\frac{x-\theta}{\alpha}\right)^{\beta-1}\exp\left[-\left(\frac{x-\theta}{\alpha}\right)^{\beta}\right]$$

Arguments

`x` A quantile the pdf is evaluated at. *x* must be greater than *threshold*.

`shape` Shape parameter β, which must be greater than 0.

`scale` Optional scale parameter α, which must be greater than 0. The default is 1.

`threshold` Optional threshold parameter θ. The default is 0.

`Weibull Distribution(x, shape, <scale=1>, <threshold=0>)`

Description

Returns the cumulative distribution function (cdf) at *x* of the Weibull distribution. The cdf is parameterized as follows:

$$F(x) = 1 - \exp\left[-\left(\frac{x-\theta}{\alpha}\right)^{\beta}\right]$$

Arguments

`x` A quantile at which the pdf is evaluated. *x* must be greater than *threshold*.

`shape` Shape parameter β, which must be greater than 0.

`scale` Optional scale parameter α, which must be greater than 0. The default is 1.

`threshold` Optional threshold parameter θ. The default is 0.

`Weibull Quantile(p, shape, <scale=1>, <threshold=0>)`

Description

Returns the p^{th} quantile from the Weibull distribution with the parameters given. The quantile function is calculated as follows:

$$F^{-1}(p) = \alpha[\ln(1-p)]^{\frac{1}{\beta}} + \theta$$

Arguments

`p` The probability of the quantile desired. *p* must be between 0 and 1.

`shape` Shape parameter β, which must be greater than 0.

`scale` Optional scale parameter α, which must be greater than 0. The default is 1.

`threshold` Optional threshold parameter θ. The default is 0.

Programming Functions

As Boolean(x)

Description

Evaluates a JSL expression and returns a JSL Boolean value for use with JSON data.

Example

```
x = 45;
b = As Boolean( x > 2 );
Show( b );
   b = true;
```

As Column(name)

As Column(dt, name)

:name

dt:name

Description

This scoping operator forces *name* to be evaluated as a data table column in the current data table (or the table given by the optional data table reference argument, *dt*) rather than as a global variable.

Arguments

`name` Variable name.

`dt` The data table reference

Notes

`:name` refers to a column name in the current data table. You can also specify which data table to refer to by use `dt:name`.

As Constant(expr)

Description

Evaluates an expression once to create a value that does not change after it is computed.

Returns

The result of the evaluation.

Argument

`expr` Any JSL expression.

Notes

A few platforms that can save prediction columns to a data table use `As Constant()`. The function is wrapped around the part of the formula that is constant across all rows. The argument is evaluated for the first row and then the result is used without re-evaluation for subsequent rows.

`As Global(name)`

`::name`

Description

This scoping operator forces *`name`* to be evaluated as a global variable rather than as a data table column.

Arguments

`name` Variable name.

`As List(matrix)`

See "As List(`matrix`)".

`As Name(`*`string`*`)`

Description

Evaluates argument as a quoted string and changes it into a name.

Returns

A name.

`As Namespace(name)`

Description

Accesses the specified namespace. An error is thrown if no such namespace exists.

Returns

The namespace.

Arguments

`name` Unquoted name of a defined namespace.

`As Scoped(namespace, variable)`

`namespace:variable`

Description

Accesses the specified *variable* within the specified *namespace*.

Returns

The value of the variable, or an error the scoped variable is not found.

Arguments

`namespace` The name of a defined namespace.

`variable` A variable defined within *namespace*.

`Associative Array({key, value}, ...)`

`Associative Array(keys, values)`

Description

Creates an associative array (also known as a dictionary or hash map).

Returns

An associative array object.

Arguments

Either list of key-value pairs; or a list, matrix, or data table column that contains keys followed by a list, matrix, or data table column, respectively, that contains the corresponding values.

`Class Exists(class)`

Description

Returns a value indicating whether a class definition represented by the class reference is a defined class.

Returns

0 or 1.

Argument

`class` A quoted string representation of the name of a defined class or reference to an instantiated class object.

`Clear Globals(<name>, <name>, ...)`

Description

Clears the values for all global symbols. Symbols in any scope other than global are not affected. If one or more names are specified, only those global symbols are cleared.

Returns

Null.

Optional Arguments

name Any global variable name(s).

See Also

"Clear Symbols(<name>, <name>, ...)"

`Clear Log()`

Description

Empties the log.

`Clear Symbols(<name>, <name>, ...)`

Description

Clear the values for all symbols in any and all scopes. If one or more names are specified, only those symbols are cleared.

Returns

Null.

Optional Arguments

name Any global variable name(s).

See Also

"Clear Globals(<name>, <name>, ...)"

`Close Log()`

Description

Closes the log.

`Define Class("class name", <Base Class( "base class name", <"base class name", ...> ),> <Show( All(Boolean) ) | Show( <Members(Boolean),> <Methods(Boolean),> <Functions(Boolean)> ),> <Assignment Statements>)`

Description

Defines a new class object.

Example

```
Define Class(
   "aa",
   _init_ = Method( {} ); x = 1; m1 = Method( {a, b}, a * b )
);
```

Delete Classes(<Force(Boolean)>, < <class>, ...>)

Description

Deletes all currently defined classes.

Optional Arguments

`Force(Boolean)` Deletes the class or classes even if they are in use.

`class` Specifies the classes to delete. You can specify more than one class. This argument can be a quoted string representation of the name of a defined class or a reference to an instantiated class object.

Delete Globals(<name>, <name>, ...)

Description

Deletes all global symbols, except global symbols that are locked. Symbols in any scope other than global are not affected. If one or more names are specified, only those global symbols are cleared.

Optional Arguments

`name` Any global variable name(s).

See Also

"Delete Symbols(<name>, <name>, ...)"

Delete Namespaces(<Force(Boolean expression)>, < <namespace reference>, ...>)

Delete(<Force(Boolean expression)>, < <namespace reference>, ...>)

Description

Deletes all currently defined namespaces or one or more specific namespaces.

Optional Arguments

`Force(Boolean expression)` Deletes the namespace even if it's in use.

`namespace reference` Specifies the namespaces to delete. You can specify more than one namespace reference.

Notes

- When you delete a namespace that contains locked namespaces, an error appears in the log. Use the `Force()` argument to delete the locked namespaces.
- With no arguments, `Delete Namespaces()` ignores locked namespaces.

Delete Symbols(<name>, <name>, ...)

Description

Deletes all symbols in any and all scopes. If one or more names are specified, only those symbols are deleted.

Optional Arguments

`name` Any global variable name(s).

See Also

"Delete Globals(<name>, <name>, ...)"

Eval(expr)

Description

Evaluates `expr`, and then evaluates the result of `expr` (unquoting).

Returns

The result of the evaluation.

Argument

`expr` Any JSL expression.

Eval Insert(*string*, <startDel>, <endDel>, < <<Use Locale(1) >)

Description

Allows for multiple substitutions.

Returns

The result.

Arguments

`string` A quoted string with embedded expressions.

`startDel` Optional starting delimiter. The default value is ^.

`endDel` optional ending delimited. The default value is the starting delimiter.

`Use Locale(1)` Optional argument that preserves locale-specific numeric formatting.

Eval Insert Into(*string*, <startDel>, <endDel>)

Description

Allows for multiple substitutions in place. The same operation as in `Eval Insert` is performed, and the result is placed into the quoted *`string`*.

Returns

The result.

Arguments

`string` A quoted string variable that contains a string with embedded expressions.

`startDel` Optional starting delimiter. The default value is ^.

`endDel` optional ending delimited. The default value is the starting delimiter.

Eval List

See "Eval List({list})".

Exit(<NoSave>)

Quit(<NoSave>)

Description

Exits JMP.

Returns

Void.

Arguments

`NoSave` Optional, named command; exits JMP without prompting to save any open files. This command is *not* case-sensitive, and spaces are optional.

First(expr, <expr>, ...)

Description

Evaluates all expressions provided as arguments.

Returns

Only the result of the first evaluated expression.

Arguments

`expr` Any valid JSL expression.

Function({arguments}, <{local variables}>, <Return(<expr>)>, script)

Description

Stores the body *`script`* with *`arguments`* as local variables.

Returns

The function as defined. If the `Return()` argument is specified, the expression is returned.

When called later, it returns the result of the *`script`* given the specified *`arguments`*.

Arguments

`{arguments}` A list of arguments to pass into the function. You can specify some arguments as optional or required.

`{local variables}` A list of variables that are local to the function. You can declare local variables in three ways:

```
{var1, var2}
{var1=0, var1="a string"}
{Default Local}
```

The last option declares that all unscoped variables used in the function are local to the function.

`Return(expr)` This optional argument returns an expression from an user defined function. If a null expression is used, a period, ".", is returned.

`script` Any valid JSL script.

Get Class Names(< <class>, ...>)

Description

Gets a set of names to all classes or the set of specific class references.

Arguments

`class` A quoted string representation of the name of a defined class or a reference to an instantiated class object.

Returns

A list of class names as determined by the arguments to the function.

Get Classes(< <class>, ...>)

Description

Gets a set of references to all classes or the set of specific class references.

Arguments

`class` A quoted string representation of the name of a defined class or a reference to an instantiated class object.

Returns

A list of class references as determined by the arguments to the function.

Get Environment Variable("variable")

Description

Retrieves the value of an operating system environment variable.

Returns

A quoted string that contains the value of the specified environment variable. If the specified variable is not found, an empty string is returned.

Arguments

`"variable"` A quoted string that contains the name of an environment variable.

Notes

On macOS, environment variable names are case-sensitive. On Windows, the names are case-insensitive.

Get Locale Setting()

Gets a local setting such as a decimal setting.

Get Log(<n>)

Description

Returns a list of lines from the log.

Returns

A list of quoted strings. Each string contains one line from the log.

Argument

n Optional, integer. If no argument is specified, all the lines are returned. If a positive number is specified, the first *n* lines are returned. If a negative number is specified, the last *n* lines are returned. If *n*=0, no lines are returned (an empty list). If the log is empty, an empty list is returned.

Get Namespace Names(< <namespace reference>,...>)

Description

Returns a list of the names of all currently defined namespaces.

Example

```
nsaa = New Namespace(
   "aa",
   {
      x = 1
   }
);
nsbb = New Namespace(
   "bb",
   {
      y = 1
   }
);
lns = Get Namespace Names();
Show( lns );
nsaa << Delete;
nsbb << Delete;
```

Get Namespaces(< <namespace reference>,...>)

Description

Returns a list of currently defined namespaces.

Example

```
nsaa = New Namespace(
   "aa",
   {
      x = 1
   }
```

```
);
nsbb = New Namespace(
   "bb",
   {
      y = 1
   }
);
lns = Get Namespaces();
```

Include("pathname", <named arguments>)

Description

Opens the script file identified by the quoted string *pathname*, parses the script in it, and executes it.

Returns

Whatever the included script returns. If you use the <<Parse Only option, Include returns the contents of the script.

Named Arguments

<<Parse Only Parses the script but does not execute the script.

<<New Context Causes the included script to be run its own unique namespace. When the parent and included scripts use the global namespace, include <<Names Default to Here along with <<New Context.

<<Allow Include File Recursion Lets the included script include itself.

Notes

If a trailing space is included in the path name, the space is ignored on Windows. On macOS, the script fails.

Include File List()

Description

Returns a list of files that are included at the point of execution.

Is Class(class)

Description

Returns a value that indicates whether the class reference is a class object.

Argument

A class reference to an instantiated class object.

Returns

Returns a zero or a 1.

Is Log Open()

Description

Returns result if log window is open.

Length

See "Length(string)".

List

See "List(a, b, c, ...)".

Local({name=value, ...}, script)

Description

Resolves *names* to local expressions.

Local Here(expression)

Description

Creates a local Here namespace block. Use this function to prevent name collisions when multiple scripts are executed from the same root namespace (for example, when a script executes two button scripts that have the same variables). The argument can be any valid JSL expression.

Lock Namespaces(<*string*>,|< {*string*}, ...>)

Description

Locks all variables or specified named variables in this namespace and prevents variables from being added, changed, or removed.

Example

```
ns = New Namespace(
   "aaa"
);
ns << Lock Namespaces;
Try( ns << Delete Namespaces, Show( exception_msg ) );
Delete Namespaces();
Try( Delete Namespaces( "aaa" ), Show( exception_msg ) );
```

Lock Globals(name1, name2, ...)

Description

Locks one or more global variables to prevent it or them from being changed.

Lock Symbols(<name>, <name>, ...)

Description

Locks the specified symbols, which prevents them from being modified or cleared. If no symbols are provided, all global symbols are locked. If no symbols are provided and the script has the *Names Default To Here* mode turned on, then all local symbols are locked.

Log Capture(expr)

Description

Evaluates the expr, captures the output that would normally be sent to the log, and instead returns it.

Returns

A quoted string that contains the log output.

Argument

Any valid JSL expression.

Notes

No output appears in the log.

Method({arg1 = val1, ...}, script)

Description

Creates a method within a class. Note that methods use local scoping for all variables that are not explicitly scoped, with the exception of class member variables.

Arguments

`{ arg1 = val1, … }` The set of expected arguments and optional initialization expressions to be passed to the method when called.

`script` Any valid JSL script.

N Items

See "N Items(source)".

Names Default To Here(Boolean)

Description

Determines where unresolved names are stored, either as a global or local (if *Boolean* is 0) or in the `Here` scope (if *Boolean* is 1).

Namespace(name)

Description

Returns a reference to the named namespace (*name*).

Argument

`Name` A namespace name quoted string or a reference to a namespace.

Namespace Exists(name)

Description

Returns 1 if a namespace with the specified *name* exists; otherwise, returns 0.

New Namespace(<"name">, <{expr, ...}>)

Description

Creates a new namespace with the specified name. If a name is not provided, an anonymous name is provided.

Returns

A reference to the namespace.

Arguments

`name` An optional, quoted string that contains the name of the new namespace.

`{list of expressions}` An optional list of expressions within the namespace.

Open Log(<Boolean>)

Description

Opens the log. Include the Boolean argument to make the window active, even if it is already open.

New Object("class name"(constructor arguments))
New Object(class name(constructor arguments))
New Object(class reference(constructor arguments))

Description

Creates an instance object of a class.

Arguments

`"class name"` Name of the class to be instantiated.

`class name` Unquoted name of the class to be instantiated.

`class reference` Reference to an existing class object that will be used to instantiate a new object of the same class.

`constructor arguments` Set of arguments to be passed to the _init_ constructor.

Example

```
Define Class(
   "complex",
   real = 0; imag = 0;
   _init_ = Method( {a, b}, real = a; imag = b; );
   Add = Method( {y}, complex( real + y:real, imag + y:imag ) );
   Sub = Method( {y}, complex( real - y:real, imag - y:imag ) );
   Mul = Method( {y},
      complex( real * y:real - imag * y:imag, imag * y:real + real * y:imag )
   );
   Div = Method( {y},
      t = complex( 0, 0 );
      mag2 = y:Magsq();
      t:real = real * y:real + imag * y:imag;
      t:imag = imag * y:real + real * y:imag;
      t:real = t:real / mag2;
      t:imag = t:imag / mag2;
      t;
   );
   Magsq = Method( {}, real * real + imag * imag );
   Mag = Method( {}, Sqrt( real * real + imag * imag ) );
   To String = Method( {}, Char( real ) || " + " || Char( imag ) || "i" )
);
cl = New Object( complex( 1, 2 ) );
```

Parameter({name=value, ...}, model expression)

Description

Defines formula parameters for models for the Nonlinear platform.

Parse(*string*)

Description

Converts a quoted character string into a JSL expression.

Print(expr, expr, ...)

Description

Prints the values of the specified *expressions* to the log.

Quit()

See “Exit(<NoSave>)”.

Recurse(function)

Description

Makes a recursive call of the defining *function*.

Save Log(pathname)

Description

Writes the contents of the log to the specified file location.

Send(obj, message)

obj << message

Description

Sends a *message* to a platform *object*.

Set Environment Variable("variable", <"value">)

Description

Sets the environment variable to the value specified. If the "value" argument is missing or is an empty quoted string, then the environment variable is deleted from the JMP process environment variable table.

Show(expr, expr, ...)

Description

Prints the name and value of each *expression* to the log.

Show Classes(< <class>,...>)

Description

Shows the contents of user-defined classes in the log. You can specify more than one class. If you do not specify an argument, all user-defined classes are shown in the log.

Example

```
Define Class(
   "aa",
   _init_ = Method( {} ); x = 1; m1 = Method( {a, b}, a * b )
);
Define Class(
   "bb",
   _init_ = Method( {} ); y = 1; m2 = Method( {a, b}, a / b )
);
Show Classes();
   // Class aa
```

```
_init_ = Method( {} );
m1 = Method( {a, b}, a * b );
x = 1;

// Class bb

_init_ = Method( {} );
m2 = Method( {a, b}, a / b );
y = 1;
```

Show Globals()

Description

Shows the values for all global symbols. Symbols in any scope other than global are not shown.

See Also

"Show Symbols()"

Show Namespaces(< <namespace reference>,...>)

Description

Shows the contents of all user-defined namespaces, both named and anonymous. You can specify zero or more namespaces.

Show Symbols()

Description

Shows the values for all symbols in any and all scopes.

See Also

"Show Globals()"

Sort List

See "Sort List({list}|expr)".

Sort List Into

See "Sort List Into({list}|expr)".

Throw("text")

Description

Returns a Throw. If you include *text*, throwing stores *text* in a global *exception_msg*. If *text* begins with "!" and is inside a Try() expression, throwing creates an error message

about where the exception was caught. "!" stops the script even if the `Throw()` is caught by the second argument of `Try()`.

`Try(expr1, expr2)`

Description

Evaluates *expr1*. If the evaluation returns a `Throw`, execution stops, and nothing is returned. *expr2* is evaluated next to return the result.

Examples

```
Try( Sqrt( "s" ), "invalid" );
   "invalid"

Try( Sqrt( "s" ), exception_msg );
   {"Cannot convert argument to a number [or matrix]"(1, 2, "Sqrt",
      Sqrt/*###*/("s"))}
```

Notes

`Expr2` can be a quoted character string or the global exception message (`exception_msg`) that contains more information about the error returned.

`Type(x)`

Description

Returns a quoted string that names the type of object *x* is. The list of possible types is: Unknown, List, DisplayBox, Picture, Column, TableVar, Table, Empty, Pattern, Date, Integer, Number, String, Name, Matrix, RowState, Expression, Associative Array, BLOB.

`Unlock Symbols(name1, name2, ...)`

`Unlock Globals(name1, name2, ...)`

Description

Unlocks the specified symbols that were locked with a `Lock Symbols()` or `Lock Globals()` command.

`Wait(n)`

Description

Pauses *n* seconds before continuing the script. The default setting is 3 seconds. Specifying `Wait(0)` enables one cycle of message processing. For example, you can use this function to allow a button press in the UI. The shortest duration that actually allows JMP to pause is `n = 0.01`. The longest duration you can specify without prompting a JMP dialog is `n = 60*60*4`.

Notes

You can use `Wait(n)` if you want something to stay on the screen long enough to see it, if you need a platform to finish launching before scripting it, or if you need to press buttons in the UI while the script runs.

`Watch(all | name1, ...)`

Description

Shows variables from global, here, and local namespaces and their values in a window. If "all" is provided as the argument, all globals are placed into the window.

Notes

- New globals are not added to the window list.
- Watching associative arrays that have been modified using messages is not supported.

`Wild()`

Description

Only used with `Extract Expr()` for expression matching to denote a wildcard position that matches any expression.

`Wild List()`

Description

Only used with `Extract Expr()` for expression matching to denote a series of wildcard arguments that match any expression.

`Write("text")`

Description

Prints *`text`* to the log without surrounding quotation marks.

Python Integration Functions

`Python Connect(<Echo(Boolean),> <Path(path),> <Use Python Version(`*`string`*`),> <Python System Path(list)>)`

Description

Initializes the Python integration interfaces and returns an active Python integration interface connection as a scriptable object.

Returns

A Python scriptable object.

Optional Named Arguments

`Echo(Boolean)` Global argument. Prints the Python source lines to the JMP log. The default value is true.

`Path` Specifies the path to the Python DLL or shared library.

`Use Python Version(`*`string`*`)` Specifies which version of Python should be used for JMP-to-Python processing.

`Python System Path` Specifies a JSL list of paths that define a Python sys path set on macOS.

`Python Control(<named arguments>)`

Description

Sends control operations to signal Python with external events, such as source line echoing.

Returns

Returns 0 if the call succeeded and 1 if an error occurred.

Optional Named Arguments

`Interactive(Boolean)` Enables interactive mode in the Python matplotlib package. Determines whether the graphics window is released or closed when graphics rendering is complete.

`Echo(Boolean)` Global argument. Prints the Python source lines to the JMP log. The default value is true.

`Python Disconnect`

Description

Terminates the Python interfaces.

`Python Execute({list of inputs}, {list of outputs}, Python_Code, named_arguments)`

Description

Submits Python code to the active global Python integration interface connection given a list of inputs. On completion, returns a list of outputs.

Returns

Returns 0 if successful and 1otherwise.

Positional Arguments

`{list of inputs}` A list of JMP variable names to be sent to Python as inputs.

`{list of outputs}` A list of JMP variable names to be retrieved from Python as outputs.

Python_Code The Python code to submit.

Named Arguments

See "Python Submit(Python_Code, <named_arguments>)".

Example

This example initiates the Python connection, sends a character variable, a numeric variable, and a set of matrices to Python. Python is then instructed to perform a set of matrix operations on the sent matrices. The Python Execute() function then get the set of matrices created by the matrix operations and gets the values of the character and numeric variables that was originally sent. Upon completion of the data retrieval, the Python connection is closed.

```
Python Init();
a = "abcdef";
d = 3.141;
v = [9 8 7, 6 5 4, 3 2 1];
m = [1 2 3, 4 5 6, 7 8 9];
ml = Python Execute(
   {v, m, a, d},
   {x1, x2, y1, y2, z1, z2, a, d},
   "\[
import numpy as np
x1 = np.multiply(v, m) # matrix product
print('x1=', x1)
x2 = np.divide(v, m) # matrix division
print('x2=', x2)
y1 = np.dot(v, m) # dot product of v and m
print('y1=', y1)
y2 = np.dot(m, v) # dot product of m and v
print('y2=', y2)
z1 = np.inner(v, m) # inner product of v and m
print('z1=', z1)
z2 = np.inner(m, v) # inner product of m and v
print('z2=', z2)
]\"
);
Show( v, m, ml, x1, x2, y1, y2, z1, z2, a, d );
Python Term();
   x1= [[  9.  16.  21.]
    [ 24.  25.  24.]
    [ 21.  16.   9.]]
   x2= [[ 9.          4.          2.33333333]
    [ 1.5         1.          0.66666667]
    [ 0.42857143  0.25        0.11111111]]
   ...
```

Python Get(name)

Description

Gets a named variable from Python to JMP.

Returns

Returns the value of the named variable.

Argument

`name` The name of the Python variable to be sent to JMP. The argument can represent any of the following Python data types: numeric, quoted string, matrix, list, or data frame.

Example

```
Python Init(); // initiate the Python connection

qbx = "The right stuff";

// send the qbx variable and sample data table "Animals.jmp" to Python
Python Send( qbx );

dt = Open( "$SAMPLE_DATA/Animals.jmp" );
Python Send( dt );
Close( dt, nosave );

// get the Python variable qbx and place it into a JMP variable qbx
qbx = Python Get( qbx );

/* get the Python variable dt and place it into a JMP data table
referenced by df */
df = Python Get( dt );

Python Term();

Show( qbx );
df << New Data View;
Wait( 10 );
Close( df, nosave );
Python Term();
   qbx = "The right stuff";
   0
```

Python Get Graphics(format)

Description

Gets the last graphics object written to the Python graph display window in the specified graphics format. The graphics object can be returned in several different graphic formats.

Returns

Returns a JMP picture object.

Argument

`format` The format that the Python graph display window contents are to be converted to. Valid formats are `png`, `bmp`, `jpeg`, `jpg`, `tiff`, `tif`, and `gif`.

`Python Get Version`

Description

Returns the version number of Python being used with the JMP Python interfaces.

`Python Init(<Echo(Boolean),> <Path(path),> <Use Python Version(`*`string`*`),> <Python System Path({list})>`

Description

Initializes the Python integration interfaces.

Returns

Returns 0 if operation is successful and 1 if not successful.

Optional Named Arguments

`Echo(Boolean)` Global argument. Prints the Python source lines to the JMP log. The default value is true.

`Path` Specifies the path to the Python DLL or shared library.

`Use Python Version(`*`string`*`)` Specifies which version of Python should be used for JMP-to-Python processing.

`Python System Path` Specifies a JSL list of paths that define a Python sys path set on macOS.

`Python Is Connected`

Description

Determines whether a Python integration interface connection is currently connected to Python.

Returns

Returns 1 if connected and 0 otherwise.

`Python JMP Name to Python Name(name)`

Description

Maps a JMP variable name to its corresponding Python variable name using Python variable name naming rules.

Returns

A quoted string, the mapped Python name.

Argument

`name` The name of the JMP variable to be sent to Python.

`Python Send(name)`

Description

Sends a named variable from JMP to Python.

Returns

Returns 0 if successful.

Argument

`name` The name of the JMP variable to be sent to Python.

`Python Send File(filename, <, Python Name(name)>)`

Description

Sends a data file to Python. The filename argument is a quoted string that specifies a pathname to the file to be sent to Python.

`Python Submit(Python_Code, <named_arguments>)`

Description

Submits Python code to the active global Python integration interface connection.

Returns

Returns 0 if successful and non-zero otherwise.

Named Arguments

`Python_Code` The Python code to submit. Statements can be a quoted string value or a list of string values.

`Expand(Boolean)` (Optional) Performs an `Eval Insert()` on the Python code before submission.

`Echo(Boolean)` (Optional) Prints the Python source lines to the JMP log.

Example

```
Python Init(); // initiate the Python connection
commands =
"
friends = ['john', 'pat', 'gary', 'michael']
print(friends)
for i, name in enumerate(friends):
    print( \!"iteration {iteration} is {name}\!".format(iteration=i,
  name=name))
";
```

```
Python Submit( commands );
Python Term();
   ['john', 'pat', 'gary', 'michael']
   iteration 0 is john
   iteration 1 is pat
   iteration 2 is gary
   iteration 3 is michael

   0
```

Python Submit File(path)

Description

Submits statements to Python using the file specified in the path name.

Argument

`path` The path to the file that contains the Python source lines to be executed.

Python Term

Description

Terminates the currently active Python integration interface.

Returns

Returns 0 if successful and 1 otherwise.

R Integration Functions

R Connect(<named_arguments>)

Description

Returns the current R connection object. If there is no connection to R, it intializes the R integration interfaces and returns an active R integration interface connection as a scriptable object.

Returns

R scriptable object.

Arguments

`Echo(Boolean)` (Optional) Sends all source lines to the JMP log. This option is global. The default value is `true`.

R Control(Interrupt|Async(Boolean)|Echo(Boolean))

Description

Changes the control options for R.

R Execute({ list of inputs }, { list of outputs }, "rCode", <named_arguments>)

Description

Submit the specified R code to the active global R connection given a list of inputs. On completion, the outputs are returned into the specified list.

Returns

0 if successful; nonzero otherwise.

Arguments

`{ list of inputs }` A list of JMP variable names to be sent to R as inputs.

`{ list of outputs }` A list of JMP variable names to contain the outputs returned from R.

`rCode` A quoted string that contains the R code to submit.

`Expand(Boolean)` An optional, Boolean, named argument. Performs an `Eval Insert()` on the R code before submitting to R.

`Echo(Boolean)` An optional, Boolean, named argument. Sends all source lines to the JMP log. This option is global. The default value is `true`.

Example

Send the JMP variables x and y to R, execute the R statement z <- x * y, and then get the R variable z and return it to JMP.

```
x = [1 2 3];
y = [4 5 6];
rc = R Execute( {x, y}, {z}, "z <- x * y" );
```

R Get(variable_name)

Description

Gets the named variable from R to JMP.

Returns

The value of the named variable.

Argument

`name` Required. The name of an R variable whose value to return to JMP.

Example

Assume that a matrix named qbx and a data frame named df are present in your R connection.

```
// get the R variable qbx and placed it into a JMP variable qbx
qbx = R Get( qbx );

// get the R variable df and placed it into a JMP data table referenced by df
df = R Get( df );
```

R Get Graphics("format")

Description

Gets the last graphics object written to the R graph display window in the specified format.

Returns

A JMP picture object.

Argument

`format` Required. Specifies the graphics format to be used. Valid formats are `png`, `bmp`, `jpeg`, `jpg`, `tiff`, `tif`, and `gif`.

R Get Version

Description

Returns the version number of R being used with JMP R interfaces.

R Init(named_arguments)

Description

Initializes the R session.

Returns

0 if the initialization is successful; any nonzero value otherwise.

Argument

`Echo(Boolean)` (Optional) Sends all source lines to the JMP log. This option is global. The default value is `true`.

R Is Connected()

Description

Determines whether a connection to R exists.

Returns

1 if connected; 0 otherwise.

Arguments

None.

R JMP Name to R Name(name)

Description

Maps the specified JMP variable name to the corresponding R variable name using R naming rules. Requires an active connection to R.

Argument

`name` The name of a JMP variable to be sent to R.

Returns

A quoted string that contains the R name.

`R Send( name, <R Name( name )>)`

Description

Sends named variables from JMP to R.

Returns

0 if the send is successful; any nonzero value otherwise.

Arguments

`name` required. The name of a JMP variable to be sent to R.

`R Name(name)` (Optional) You can give the variable that you send to R a different name. For example

```
R Send(Here:x, R Name("localx"))
```

For data tables only:

`Selected(Boolean)` optional, named, Boolean. Send only selected rows from the referenced data table to R.

`Excluded(Boolean)` optional, named, Boolean. Send only excluded rows from the referenced data table to R.

`Labeled(Boolean)` optional, named, Boolean. Send only labeled rows from the referenced data table to R.

`Hidden(Boolean)` optional, named, Boolean. Send only hidden rows from the referenced data table to R.

`Colored(Boolean)` optional, named, Boolean. Send only colored rows from the referenced data table to R.

`Markered(Boolean)` optional, named, Boolean. Send only markered rows from the referenced data table to R.

`Row States(Boolean, <named arguments>)` optional, named. Includes a Boolean argument and optional named arguments. Send row state information from the referenced data table to R by adding an additional data column named “RowState”. Multiple row states are created by adding together individual settings. Here are the individual values:

- Selected = 1
- Excluded = 2
- Hidden = 4
- Labeled = 8
- Colored = 16

– Markered = 32

Here are the named arguments for the `Row States()` argument:

`Colors(Boolean)` optional, named, Boolean. Sends row colors. Adds additional data column named "RowStateColor".

`Markers(Boolean)` optional, named, Boolean. Sends row markers. Adds additional data column named "RowStateMarker".

Examples

Create a matrix, assign it to X, and send the matrix to R:

```
X = [1 2 3];
rc = R Send( X );
```

Open a data table, assign a reference to it (*dt*), and send the data table, along with its current row states, to R:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
rc = R Send( dt, Row States(1) );
```

R Send File("pathname", <R Name("name")>)

Description

Sends the specified data file from JMP to R.

Returns

0 if the send is successful; any nonzero value otherwise.

Arguments

`pathname` required. A quoted string that contains a pathname for a file.

`R Name(name)` (Optional) You can give the data file that you send to R a different name.

R Submit("rCode", <named_arguments>)

Description

Submits the specified R code to the active global R connection.

Returns

0 if successful; nonzero otherwise.

Arguments

`rCode` A required, quoted string that contains the R code to submit.

`Expand(Boolean)` An optional, Boolean, named argument. Performs an `Eval Insert()` on the R code before submitting to R.

`Echo(Boolean)` An optional, Boolean, named argument. Sends all source lines to the JMP log. This option is global. The default value is `true`.

`Async(Boolean)` An optional, Boolean, named argument. If set to true (1), the submit can be canceled either by pressing the ESCAPE key, or by using this message to an R connection: `rconn<<Control( Interrupt( 1 ) )`. False (0) is the default value.

Example

```
rc = R Submit("\[
  x <- rnorm(5)
  print(x)
  y <- rnorm(5)
  print(y)
  z = plot(x, y)
]\" );
```

`R Submit File( "pathname" )`

Description

Submits statements to R using a file pointed in the specified *pathname*.

Returns

0 if successful; nonzero otherwise.

Argument

`Pathname` A quoted string that contains the pathname to the file that contains the R code to be executed.

`R Term()`

Description

Terminates the currently active R integration interface.

Returns

Returns 0 if the termination is successful and -1 otherwise.

Arguments

None

Random Functions

`Col Shuffle(<By var,...>)`

Description

Creates a random ordering of the row numbers of the current data table when used in a column formula.

Note: This function is generally used in a column formula.

Returns

A random integer between 1 and the number of rows in the current data table.

Argument

`By var` (Optional) A By variable enables you to randomly order the rows within the groups of the By variable values.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "Shuffle", Numeric, Continuous, Set Formula( Col Shuffle() )
  );
```

This example creates a column formula that shuffles the order of the row numbers (1 to 40) each time the formula is evaluated. Each number appears only once.

Make Validation Formula(rates, <<Stratification Columns(cols), <<Grouped Columns(cols), <<Cutpoint Column(col))

Description

Generates a validation column. This function is primarily used by the Make Validation Column utility.

Arguments

`rates` Vector of three rates that specify the training, validation, and test rates, respectively.

`<<Stratification Columns` Assigns one or more stratification columns.

`<<Grouped Columns` Assigns one or more grouping columns.

`<<Cutpoint Column` Assigns a numeric cutpoint column.

Predictive and Specialized Modeling

Random Beta(alpha, beta, <theta=0>, <sigma=1>)

Description

Returns a random number from a beta distribution with two shape parameters, *alpha* and *beta*, and optional parameters *theta* and *sigma*.

Arguments

`alpha, beta` Shape parameters α and β, which must both be greater than 0.

`theta` Optional threshold parameter θ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

Random Beta Binomial(n, p, <delta=0>)

Description

Returns a random number from a beta binomial distribution for *n* trials with probability *p* and overdispersion parameter *delta*.

Arguments

n The number of trials, which must be greater than or equal to 2. If the specified *n* is not an integer, the non-integer part is truncated.

p The probability of success for each trial, which must be between 0 and 1.

delta The overdispersion parameter δ, which must be between Maximum[-*p*/(*n*-*p*-1), -(1-*p*)/(*n*-2+*p*)] and 1. The default is 0.

Random Binomial(n, p)

Description

Returns a random number from a binomial distribution with *n* trials and probability *p* of the event of interest occurring.

Arguments

p The probability of success for each trial, which must be between 0 and 1.

n The number of trials.

Random Category(probA, resultA, probB, resultB, <..., ...,> resultElse)

Description

Returns one of the specified result expressions at random, chosen from pairs of probability and result expressions. A random uniform number is generated and compared to the *prob* arguments to determine which *result* argument is returned.

Arguments

probA Numeric value between 0 and 1 that represents the probability of the corresponding result expression being returned.

resultA Expression that corresponds to *probA*.

resultElse Expression that is returned if no previous result expression has been returned.

Random Cauchy()

Description

Returns a random number from a Cauchy distribution with a median of zero.

Random ChiSquare(df, <nc=0>)

Description

Returns a random number from a chi-square distribution with given *df* (degrees of freedom) and optional noncentrality parameter.

Arguments

df The degrees of freedom *n*, which must be greater than 0.

nc Optional noncentrality parameter λ, which must be nonnegative. The default is 0.

Random Exp()

Description

Returns a random number from an exponential distribution with scale parameter equal to 1. Equivalent to the negative log of `Random Uniform`.

Random F(dfnum, dfden, <noncentral=0>)

Description

Returns a random number from an F distribution with a given *dfnum*, *dfden*, and optional noncentrality parameter.

Arguments

`dfnum` The degrees of freedom, v_1, of the chi-square distribution in the numerator of the *F*-distribution. *dfnum* must be greater than 0.

`dfden` The degrees of freedom, v_2, of the chi-square distribution in the denominator of the *F*-distribution. *dfden* must be greater than 0.

`noncentral` Optional noncentrality parameter λ, which must be nonnegative. The default is 0.

Random Frechet(<mu=0>, <sigma=1>)

Description

Returns a random number from a Fréchet distribution with the location *mu* and scale *sigma*.

Arguments

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

Random Gamma(alpha, <scale=1>)

Description

Returns a random numbers from a gamma distribution for given *alpha* and optional *scale*.

Arguments

`alpha` The shape parameter α, which must be greater than 0.

`scale` Optional scale parameter β, which must be greater than 0. The default is 1.

Random Gamma Poisson(lambda, <sigma=1>)

Description

Returns a random number from a gamma Poisson distribution with parameters *lambda* and *sigma*.

Arguments

`lambda` The shape parameter λ, which much be greater than 0.

`sigma` Optional overdispersion parameter σ, which must be greater than or equal to 1. The default is 1. When the overdispersion parameter is 1, the distribution reduces to a Poisson(λ) distribution.

`Random GenGamma(<mu=0>, <sigma=1>, <lambda=0>)`

Description

Returns a random number from an extended generalized gamma distribution with parameters *mu*, *sigma*, and *lambda*.

Arguments

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

`lambda` Optional shape parameter λ. The default is 0.

`Random Geometric(p)`

Description

Returns a random number from the geometric distribution with probability *p* that a specific event occurs at any one trial.

`Random GLog(mu, sigma, lambda)`

Description

Returns a random number from a generalized logarithmic distribution with parameters *mu*, *sigma*, and *lambda*.

Arguments

`mu` The location parameter μ.

`sigma` The scale parameter σ, which must be greater than 0.

`lambda` A shape parameter λ, which must be greater than 0.

`Random Index(n, k)`

Description

Returns a *k* by 1 matrix of random integers between 1 and *n* with no duplicates.

`Random Integer(n)`

`Random Integer(k, n)`

Description

Returns a random integer from 1 to *n* or from *k* to *n*.

Random Johnson Sb(gamma, delta, theta, sigma)

Description

Returns a random number from a Johnson Sb distribution with parameters *gamma*, *delta*, *theta*, and *sigma*.

Arguments

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Scale parameter σ, which must be greater than 0.

Random Johnson Sl(gamma, delta, theta, <sigma=1>)

Description

Returns a random number from a Johnson Sl distribution with parameters *gamma*, *delta*, *theta*, and optional *sigma*.

Arguments

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Optional parameter σ that indicates if the distribution is skewed positively or negatively. `sigma` must be equal to either +1 (skewed positively) or -1 (skewed negatively). The default is +1.

Random Johnson Su(gamma, delta, theta, sigma)

Description

Returns a random number from a Johnson Su distribution with parameters *gamma*, *delta*, *theta*, and *sigma*.

Arguments

`gamma` Shape parameter γ.

`delta` Shape parameter δ, which must be greater than 0.

`theta` Location parameter θ.

`sigma` Scale parameter σ, which must be greater than 0.

Random LEV(<mu=0>, <sigma=1>)

Description

Returns a random number from an LEV distribution with the location *mu* and scale *sigma*.

Arguments

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

Random LogGenGamma(<mu=0>, <sigma=1>, <lambda=0>)

Description

Returns a random number from a log generalized gamma distribution with parameters *mu*, *sigma*, and *lambda*.

Arguments

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

`lambda` Optional shape parameter λ. The default is 0.

Random Logistic(<mu=0>, <sigma=1>)

Description

Returns a random number from a logistic distribution with location *mu* and scale *sigma*.

Arguments

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

Random Loglogistic(<mu=0>, <sigma=1>)

Description

Returns a random number from a loglogistic distribution with location *mu* and scale *sigma*.

Arguments

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

Random Lognormal(<mu=0>, <sigma=1>)

Description

Returns a random number from a lognormal distribution with location *mu* and scale *sigma*.

Arguments

`mu` Optional location parameter μ. The default is 0.

`sigma` Optional scale parameter σ, which must be greater than 0. The default is 1.

Random Multivariate Normal(mean, covar, <nrows=1>)

Description

Returns a random vector from a multivariate normal distribution with mean vector *mean* and covariance matrix *covar*. To generate multiple vectors, specify an integer greater than 1 for the *nrows* argument. When *nrows* is greater than 1, the return value is a matrix. The number of columns in the random vector or matrix is equal to the number of rows in the *covar* argument.

Arguments

mean Mean vector for the multivariate normal distribution.

covar Covariance matrix for the multivariate normal distribution. This matrix must be a symmetric square matrix that contains the same number of columns as the mean vector.

nrows Optional argument that specifies the number of random vectors returned. The default number of rows is 1.

Random Negative Binomial(n, p)

Description

Returns a random number from a negative binomial distribution for *n* successes with probability of success *p*.

Random Normal(<mu=0>, <sigma=1>)

Description

Returns a random number from a normal distribution with mean *mu* and standard deviation *sigma*.

Arguments

mu Optional location parameter μ. The default is 0.

sigma Optional scale parameter σ, which must be greater than 0. The default is 1.

Random Normal Mixture(meanvec, sdvec, probabvec)

Description

Returns a random number from a normal mixture distribution with the specified arguments.

Arguments

meanvec A vector that contains group means.

sdvec A vector that contains the group standard deviations.

probvec A vector that contains the group probabilities.

Random Poisson(lambda)

Description

Returns a random number from a Poisson distribution with shape parameter *lambda*.

Arguments

lambda The shape parameter λ, which much be greater than 0.

Random Reset(seed)

Description

Restarts the random number sequences with *seed*.

Notes

You can use the **Random Reset** function to set a random seed that enables you to reproduce a set of random numbers. The random number algorithms in JMP are sometimes refined, and therefore the results might not be consistent across different versions of JMP.

Random Seed State(<seed state>)

Description

Retrieves or restores the random seed state to or from a BLOB object.

Random SEV(<mu=0>, <sigma=1>)

Description

Returns a random number from an SEV distribution with the specified location *mu* and scale *sigma*.

Arguments

mu Optional location parameter μ. The default is 0.

sigma Optional scale parameter σ, which must be greater than 0. The default is 1.

Random SHASH(gamma, delta, theta, sigma)

Description

Returns a random number from a sinh-arcsinh (SHASH) distribution with parameters *gamma*, *delta*, *theta*, and *sigma*.

Arguments

gamma The shape parameter γ.

delta The shape parameter δ, which must be greater than 0.

theta The location parameter θ.

sigma The scale parameter σ, which must be greater than 0.

`Random Shuffle(matrix)`

Description

Returns the matrix with the elements shuffled into a random order.

`Random t(df, <noncentral=0>)`

Description

Returns a random number from a t distribution with the specified *df* (degrees of freedom). The noncentrality argument may be negative or positive. The default value of *noncentral* is 0.

`Random Triangular(min, mode, max)`

`Random Triangular(mode, max)`

`Random Triangular(mode)`

Description

Generates a random number from a triangular distribution between 0 and 1 with the *mode* that you specify. The triangular distribution is typically used for populations that have a small number of data.

Arguments

`min` Specifies the lower limit of the triangular distribution. The default value is 0.

`mode` Specifies the mode of the triangular distribution.

`max` Species the upper limit of the triangular distribution. The default value is 1.

Notes

If you specify only the mode, the minimum value is 0, and the maximum value is 1. If you specify the mode and maximum value, the minimum value is 0 by default.

`Random Uniform()`

`Random Uniform(x)`

`Random Uniform(min, max)`

Description

Generates a random number from a uniform distribution between 0 and 1. `Random Uniform(x)` generates a number between 0 and x. `Random Uniform (min, max)` generates a number between *min* and *max*. The result is an approximately even distribution.

Random Weibull(shape, <scale=1>)

Description

Returns a random number from a Weibull distribution with parameters *shape* and optional *scale*.

Arguments

`shape` Shape parameter β, which must be greater than 0.

`scale` Optional scale parameter α, which must be greater than 0. The default is 1.

Resample Freq(<rate=1, <column>>)

Description

Generates a frequency count for sampling with replacement. If no arguments are specified, the function generates a 100% resample.

Note: This function is generally used in a column formula.

Arguments

`rate` (Optional) Specifies the rate of resampling. The default value is 1. A negative value specifies that fractional frequencies are allowed.

`column` (Optional) If you specify *column*, you must also specify *rate*. The sample size is calculated by the rate multiplied by the sum of the specified column. If rate is negative, then the sample size is the negative of the rate multiplied by the sum of the specified column. If you do not specify a column, the generated frequencies sum to the number of rows.

Example

To ensure that the numbers in the frequency column match each time you run the script, use `As Constant()`. `As Constant()` evaluates an expression to create a constant value that does not change after it has been computed.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dc = dt << New Column( "column",
   Formula(
      As Constant(
         Random Reset( 123 );
         0;
      ) + Resample Freq()
   )
);
dc << Eval Formula;
```

Notes

– A typical use of this function generates a column with many 1s, some 0s, some 2s, and so forth, corresponding to which rows were randomly assigned any of *n* randomly selected rows.

– A typical use of this with an *existing* frequency column produces a new frequency column whose values are similar to the old frequency column (have the same expected value); however, the values vary somewhat due to random selection at the rates corresponding to the old frequency column.

Row Functions

```
As Table(matrix, <matrix 2, ...>, < <<invisible >, < <<private >, <
<<Column Names({list}) >)
```

Description

Creates a new data tables from the *matrix*.

Returns

The new data table.

Argument

`matrix` Any matrix.

`<<invisible` Creates an invisible data table that hides the table from view but lists it in the JMP Home Window and Window menu.

`<<private` Hides the table completely. Creating a private data table speeds the process of getting to the data; it does not save the computer from allocating the memory necessary to hold the data table data.

`<<Column Names(list)` The list specified column names for the data. The argument is a list of quoted column names.

```
Col Stored Value(<dt>, col, <row>)
```

Description

Returns the data values stored in the column and disregards values assigned through column properties (such as Missing Value Codes).

Arguments

`dt` Optional reference to a data table. If this value is not supplied, the current data table is used.

`col` Name of the column.

`row` (Optional) Row name or number. If this value is not specified, the current row is used.

Example

Suppose that the Missing Value Codes column property is assigned to the `x1` column to treat "999" as a missing value. Another column includes a formula that calculates the

mean. To use the value "999" instead of a missing value to calculate the mean, use `Col Stored Value()` in the formula:

```
Mean( Col Stored Value( :x1 ), :x2, :x3 )
```

`Column(<dt>, "name", "formatted")`

`Column(<dt>, n)`

Description

Gets a reference to the data table column.

Arguments

`dt` Optional reference to a data table. If this is not supplied, the current data table is used.

`name` A quoted string that is the name of the column.

`formatted` A quoted string that returns the formatted string of the cell value.

`n` The column number.

`Column Name(n)`

Description

Determines the name of the column specified by number.

Returns

The name of the n^{th} column as an expression (not a quoted string).

Argument

`n` The number of a column.

`Count(from, to, step, times)`

Description

Used for column formulas. Creates row by row the values beginning with the *from* value and ending with the *to* value. The number of *steps* specifies the number of values in the list between and including the *from* and *to* values. Each value determined by the first three arguments of the count function occurs consecutively the number of *times* that you specify. When the *to* value is reached, `count` starts over at the *from* value. If the *from* and *to* arguments are data table column names, `count` takes the values from the first row only. Values in subsequent rows are ignored.

Returns

The last value.

Arguments

`from` Number, column reference, or expression. `Count` starts counting with this value.

`to` Number, column reference, or expression. `Count` stops counting with this value.

`step` Number or expression. Specifies the number of steps to use to count between *from* and *to*, inclusive.

`times` Number or expression. Specifies the number of times each value is repeated before the next step.

Examples

/* the rows in the column named *colname* are filled with the series 0, 3, 6, 0,

```
... until all rows are filled */
For Each Row(:colname[row()]=count(0, 6, 3, 1))

/* the rows in the column named colname are filled with the series 0, 0, 3,
   3, 6, 6, 0, ... until all rows are filled */
For Each Row(:colname[row()]=count(0, 6, 3, 2))
```

Notes

`Count()` is dependent on `Row()`, and is therefore mainly useful in column formulas.

Current Data Table(<dt>)

Description

Without an argument, gets the current (topmost) data table. With an argument, sets the current data table.

Returns

Reference to the current data table.

Argument

`dt` Optional name of or reference to a data table.

Notes

Private tables cannot be made current with `Current Data Table()`.

Data Table(n)

Data Table("name")

Get Data Table(<project(title|index|box|window),> name|index)

Description

Gets reference to the *n*th open data table or the table with the given *name* in a global variable.

Returns

Reference to the specified data table.

Argument

`n` Number of a data table.

`name` Quoted string, name of a data table.

`Dif(col, n)`

Description

Calculates the difference of the value of the column *col* in the current row and the value *n* rows previous to the current row.

Returns

The difference.

Arguments

`col` A column name (for example, `:age`).

`n` A number.

`Dim(<dt|matrix>)`

Description

Returns a row vector with the dimensions of the current data table, a specified data table, or a matrix. The dimensions are the number of rows and the number of columns and are listed in that order.

Arguments

`dt` A data table.

`matrix` A matrix.

Notes

If no argument is specified, the dimensions of the current data table are returned.

`Get Data Table List(<Project(title|index|box|window>)`

Description

Returns a list of all open data tables.

Notes

Use `Project(0)` to specify no project when running the expression in a project.

`Lag(col, n)`

Description

Returns for each row the value of the column *n* rows previous.

`N Row(dt); NRow(matrix)`

`N Rows(dt); NRows(matrix)`

Description

Returns the number of rows in the data table given by *dt* or in the *matrix.*

N Table()

Description

Returns the number of open data tables. Private tables are not included.

New Column("name", <"data type">, <"modeling type">, <Width(n)>, Format("format", width, precision), <Formula()>, <Set Values>, <Like(column reference)>, <actions>)

Description

Adds a new column named *"name"* after the last column in *dt*. Unless otherwise specified, columns are numeric, continuous, and 12 characters wide.

Returns

A column reference.

Notes

Can also be used as a message: `dt<<New Column`.

The `Like()` argument copies the data type, modeling type, format, formula, and other properties from the reference column into the new column.

See Also

"dt<<New Column(name, <data type>, <modeling type>, <Format(format, width)>, <Formula()>, <Set Values({..., ..., })>, <Set Property(properties)>)"

New Table("name", <visibility("invisible" | "private" | "visible")>, <actions>)

Description

Creates a new data table with the specified *name*.

Arguments

`name` A quoted string that contains the name of the new table.

`visibility` Optional quoted keyword. `invisible` hides the data table from view but lists it in the JMP Home Window and Window menu. `private` hides the table completely. `visible` shows the data table. "`visible`" is the default value.

Note: Creating a private data table speeds the process of getting to the data; it does not save the computer from allocating the memory necessary to hold the data table data.

`actions` Optional argument that can define the new table.

```
Row()
```

```
Row() = y
```

Description

Returns or sets the current row number. No argument is expected.

```
Sequence(from, to, <step size>, <repeat times>)
```

Description

Produces an arithmetic sequence of numbers across the rows in a data table. The *step size* and *repeat times* arguments are optional, and the default value for both is 1.

```
Subscribe to Data Table List(<subscriber name|"">,
<OnOpen(function)|OnClose(function)|OnRename(function)>)
```

Description

Subscribes to the data table list. You will be notified when a new data table has been added, closed, or renamed.

```
Subscript(a, b, c)
```

```
list[i]
```

```
matrix[b, c]
```

Description

Subscripts for lists extract the *i*th item from the *list*, or the *b*th row and the *c*th column from a *matrix*.

```
Suppress Formula Eval(Boolean)
```

Description

Turns off automatic calculation of formulas for all data tables.

```
Unsubscribe to Data Table List(<subscriber name>,
<"OnOpen"|"OnClose"|"All">)
```

Description

Removes a subscription to the data table list that has been added through `Subscribe to Data Table List()`.

Row State Functions

As Row State(i)

Description

Converts *i* into a row state value.

Returns

A row state from the *i* given.

Argument

`i` an integer

Color Of(rowstate)

Description

Returns or sets the color index.

Returns

The color index of *rowstate*.

Argument

`rowstate` a row state argument

Example

Set the color of the fifth row to red.

```
Color Of( Rowstate( 5) ) = 3
```

Color State(i)

Description

Returns a row state with the color index of *i*.

Returns

A row state.

Argument

`i` index for a JMP color

Combine States(rowstate, rowstate, ...)

Description

Generates a row state combination from two or more row state arguments.

Returns

A single numeric representation of the combined row states.

Arguments

`rowstate` Two or more row states.

`Excluded(rowstate)`

Description

Returns or sets an excluded index.

Returns

The excluded attribute, 0 or 1.

Argument

`rowstate` One or more row states.

`Excluded State(num)`

Description

Returns a row state for exclusion from the *num* given.

`Hidden(rowstate)`

Description

Returns or sets the hidden index.

`Hidden State(num)`

Description

Returns a row state for hiding from the *num* given.

`Hue State(num)`

Description

Returns a hue state from the *num* given.

`Labeled(rowstate)`

Description

Returns or sets the labeled index.

`Labeled State(num)`

Description

Returns a labeled state from the *num* given.

Marker Of(rowstate)

Description

Returns or sets the marker index of a row state.

Marker State(num)

Description

Returns a marker state from the *num* given.

Row State(<dt,> <n>)

Description

Returns the row state changed from the initial condition of the active row or the *n*th row.

Arguments

dt Optional positional argument: a reference to a data table. If this argument is not in the form of an assignment, then it is considered a data table expression.

n The row number.

Example

The following example creates the data table references and then returns the row state of row 1 in Big Class.jmp:

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt2 = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
Row State( dt1, 1 );
```

Selected(rowstate)

Description

Returns or sets the selected index.

Selected State(num)

Description

Returns a selected state from the *num* given.

Shade State(num)

Description

The Shade State function assigns 5 shade levels to a color or hue.

SAS Integration Functions

As C Expr(x)

Description

Returns a C programming language representation of the expression.

Returns

A quoted string.

As JavaScript Expr(x)

Description

Returns a JavaScript representation of the expression.

Returns

A quoted string.

As JSON Expr(x)

Description

Returns a JSON (JavaScript Object Notation) representation of the expression.

Returns

A quoted string.

As Python Expr(x)

Description

Returns a Python representation of the expression.

Returns

A quoted string.

As SAS Expr(x)

Description

Converts an expression to a version that is more suitable for SAS DATA step. The code must be wrapped in a PROC DS2 call. Use `Expr(...)` for literal expressions. Use `NameExpr(name)` for expressions stored in a variable. Otherwise, the expression returns the expression to convert.

Returns

A quoted string.

Current Metadata Connection()

Description

Returns the active SAS metadata server connection, if any, as a scriptable object.

Current SAS Connection()

Description

Gets the active global SAS server connection, if any, as a scriptable object.

Get SAS Version Preference()

Description

Returns the SAS version selected in the SAS Integration page of the Preferences as a quoted string.

JMP6 SAS Compatibility Mode(Boolean)

Description

Setting this to 1 (true) causes SAS operators to operate in a mode compatible with JMP 6 capabilities.

Meta Connect(<"machine", port>, <"authDomain">, <"username">, <"password">, <named arguments>)

Meta Connect(<Profile("profile name")>, <Password("password")>, <named arguments>)

Meta Connect(<Environment("environment name")>, <named arguments>)

Description

Connects to a SAS Metadata Server. If no arguments are specified, an empty connection window appears. If some arguments are specified, a window partially filled in with the argument values appears. If all arguments are specified, the connection is made and no window appears.

Returns

1 if connection is successful, 0 if not.

Arguments

machine (Optional) A quoted string that contains the DNS name of the machine.

port Required if machine is specified. A quoted string or integer that contains the port on which the metadata server listens.

authDomain (Optional) A quoted string that contains the authentication domain for the credentials supplied. Not necessary unless *username* and *password* are included.

username (Optional) A quoted string that contains the user name for the connection.

`password` (Optional) A quoted string that contains the password for the connection.

Optional Named Arguments

`Profile("profile_name")` A quoted string that contains the name of the metadata server connection profile from which connection information should be retrieved.

`Environment("environment_name")` A quoted string that contains the name of the WIP environment from which connection information should be retrieved.

`Password("password")` A quoted string that contains the password for the specified profile name.

`CheckPreferenceOnly(0|1)` If specified, `Meta Connect` returns the status of the **I want to connect to a SAS Metadata Server** option in the SAS Integration page of JMP Preferences. If that box is checked, **Meta Connect** returns 1; if not, 0.

`Repository(`*`string`*`)` Takes a quoted string that contains the name of the repository to which to connect.

`ProfileLookup(0|1)` If *`machine`* and *`port`* are specified rather than a profile name, and `ProfileLookup` is specified, an attempt is made to find a metadata server connection profile with a machine name and port matching those provided. If one is found, other connection information (such as authentication domain, user name, and password) is obtained from that profile.

`Prompt(Always|Never|IfNeeded)` Takes one of the keywords `Always` (always prompt before attempting to connect), `Never` (never prompt, just fail), or `IfNeeded` (the default; prompt if connection with the given arguments fails).

`SASVersion("<version number>" <,Strict>)` Attempts to change the SAS version preference to the specified value before making the metadata server connection. If the SAS version is already locked to a different version than the one specified, the SASVersion argument will fail. By default, if the SAS version cannot be set, JMP will try the metadata server connection. However, if you include `Strict` as the second argument, the inability to change the SAS version will be treated as an error, and JSL processing will stop. If you do not include `Strict`, the SAS version argument is treated as a hint and will set the version preference if it can. JMP will still try to connect if the version cannot be set. The order you put these arguments in can make a difference. The attempt to change the SAS Version is made immediately when that argument is encountered. That can affect the validity of other arguments, particularly for `MetaConnect`. Valid values for `SASVersion` are “9.3” and “9.4”. Note: Using the `SASVersion` argument has the same effect as changing the SAS Server Version on the SAS Integration Preferences page.

Notes

– If no arguments are included and if no profile is saved, the Connect to SAS Metadata Server window appears.

– If you connect to a physical workspace server, there is no metadata server involved, so metadata security is never applied. You must connect to a SAS Metadata Server and

then connect to a logical workspace server. Then metadata security is enforced on the metadata-defined libraries you access.

`Meta Create Profile("profile", <named arguments>)`

Description

Creates a metadata server connection profile and adds it to the current user's set of saved metadata server connection profiles.

Returns

1 if *`profile`* was successfully created, otherwise 0.

Arguments

`profile` A quoted string that contains the name of the created profile. If a profile by the given name already exists, `MetaCreateProfile` fails unless *`Replace`* is specified.

Optional Named Arguments

`HostName("name")` A quoted string that contains the name of the host computer running the SAS Metadata Server that this profile will connect to.

`Port(n)` The port number (*n*) that the SAS Metadata Server is listening for connections on.

`AuthenticationDomain("domain") | AuthDomain("domain")` A quoted string that sets the authentication domain to use for the connection.

`Description("desc") | Desc("desc")` A quoted string that sets a description for this profile.

`Password("password")` A quoted string that contains the password to store in this profile.

`Replace(0|1)` If *`name`* matches a profile that already exists, `Replace` must be specified for the existing profile to be replaced by the one provided. The default value is False (0).

`UserName("username")` A quoted string that contains the user name that this profile uses to connect to the SAS Metadata Server.

`UseSingleSignOn(0|1)` If specified, this profile attempts to use Single Sign-On (currently also known as Integrated Windows Authentication) to connect to the SAS Metadata Server. This option is valid only for connecting to SAS 9.3 or higher Metadata Servers. If `UseSingleSignOn` is True(1), *`UserName`* and *`Password`* cannot be specified. The default value is False (0).

`Meta Delete Profile("name")`

Description

Deletes the named metadata server connection profile from the current user's set of saved metadata server connection profiles

Returns

1 if profile was successfully deleted, otherwise 0.

Argument

`name` A quoted string that contains the name of the profile to delete.

`Meta Disconnect()`

Description

Disconnect the current SAS Metadata Server connection, if any.

Returns

Void.

`Meta Get Environments()`

Description

Returns a list of the SAS Environments that are defined in the SAS Environments definition file, which is configured in the SAS Preferences.

`Meta Get Repositories()`

Description

Gets a list of the repositories available on the current SAS Metadata Server connection.

Returns

A list of repository names as quoted strings.

`Meta Get Servers()`

Description

Get a list of the SAS Servers that are registered in the SAS Metadata Repository to which the session is currently connected.

Returns

A list of server names as quoted strings.

`Meta Get Stored Process("path")`

Description

Get a stored process object from the currently connected SAS Metadata Repository.

Returns

Stored Process scriptable object.

Arguments

`path` Quoted string that is the path to the stored process in metadata, starting at the BIP Tree.

Meta Is Connected()

Description

Determines whether a current connection to a SAS Metadata Server exists.

Returns

1 if a connection exists; 0 otherwise.

Arguments

None.

Meta Set Repository("repositoryName")

Description

Set the SAS Metadata Repository to use for metadata searches.

Returns

1 if setting the repository was successful, 0 otherwise.

Arguments

`repositoryName` A quoted string that contains the name of the repository to make current.

SAS Assign Lib Refs("libref", "path", <"engine">, <"engine options">)

Description

Assign a SAS libref on the active global SAS server connection.

Returns

1 if successful, 0 otherwise.

Arguments

`libref` A quoted string that contains a library reference (8-character maximum) to assign.

`path` A quoted string that contains the full path on the SAS server to the library being assigned.

`engine` Optional, quoted string that contains the engine for the SAS server to use when accessing members of this library.

`engine options` Optional, quoted string that contains the options needed for the engine being used.

SAS Connect(<"machine_name">, <"port">, <named_arguments>)

Description

Connect to a local, remote, or logical SAS server.

Returns

SAS Server scriptable object.

Arguments

`machine_name` (Optional) A quoted string that can contain a physical machine name or the name of a metadata-defined (logical) server. In the first case, the port must be provided. In the second case, a port must *not* be provided. If neither *`name`* nor *`port`* are included, and JMP is running on Windows, a connection to SAS on the local machine (via COM) is attempted, and all named arguments are ignored.

`port` (Optional) A quoted string or integer. If *`name`* is a physical machine name, this is the port on that machine to connect to. If *`name`* is a metadata-defined (logical) server, *`port`* must *not* be included.

Optional Named Arguments

`UserName("name")` A quoted string that contains the user name for the connection.

`Password("password")` A quoted string that contains the password for the connection.

`ReplaceGlobalConnection(0|1)` A Boolean. The default value is `True`. If `True`, and a successful SAS server connection is made, this connection replaces the active SAS connection that becomes the target of other global SAS JSL function calls. If `False`, the global SAS connection is not changed, and the returned SASServer scriptable object should be used to send messages to this server connection.

`ShowDialog(0|1)` A Boolean. The default value is `False`. If `True`, other arguments (except `ReplaceGlobalConnection`) are ignored and the SAS Server Connection window appears. This provides the JSL programmer a way to open the SAS Connect window.

`Prompt(Always|Never|IfNeeded)` A keyword. `Always` means always prompt before attempting to connect. `Never` means never prompt even if the connection attempt fails (just fail and send an error message to the log), and `IfNeeded` (the default value) means prompt if the attempt to connect with the given arguments fails (or is not possible with the information given).

`ConnectLibraries(0|1)` A Boolean. Defaults to the SAS Integration Preference setting governing whether to automatically connect metadata-defined libraries when connecting to a SAS server. If true, all metadata-defined libraries are connected at SAS server connection time, which can be time-consuming. If false, metadata-defined libraries are not connected. To connect specific libraries later, use the `SAS Connect Libref` global function or `Connect Libref` message to a SAS server object.

`SASVersion("<version number>" <,Strict>)` Attempts to change the SAS version preference to the specified value before making the metadata server connection. If the SAS version is already locked to a different version than the one specified, the SASVersion argument will fail. By default, if the SAS version cannot be set, the metadata server connection will still be tried. However, if you include `Strict` as the second argument, the inability to change the SAS version will be treated as an error and JSL processing will stop. If you do not include `Strict`, the SAS version argument is treated as a hint and will set the version preference if it can, but if it cannot it will still try to connect. The order you put these arguments in can make a difference. The attempt to

change the SAS Version is made immediately when that argument is encountered. That can affect the validity of other arguments, particularly for MetaConnect. Valid values for `SASVersion` are "9.3" and "9.4". Note: Using the `SASVersion` argument has the same effect as changing the SAS Server Version on the SAS Integration Preferences page.

Example

```
// prompt for login credentials
Meta Connect( "dev.company.com", 28561 );

sas = SAS Connect( "SASApp" );

// dump some libraries and data sets to the JMP log
Show( sas << Get Librefs() );
Show( sas << Get Data Sets( "Chocolate Enterprises 2017" ));

sas << Import Data( "Chocolate Enterprises 2017", "Products" );

/* The preceding lines produce the following text in the JMP log and import
   the data set: */
sas << Get Librefs:{"BOOKS", "CHOCOENT", "EGSAMP", "TEST", "TST92", "MAPS",
   "SASDATA", "SASHELP", "SASUSER", "SGFDATA", "TEMPDATA", "V6LIB", "WORK"}
sas << Get Data Sets("Chocolate Enterprises 2008"):{"CHOC_DATA",
   "CHOC_SURVEY", "CUSTOMERS", "ORDER_DETAIL", "PRODUCTS", "SALES_ANALYSIS",
   "SALES_SUMMARY"}
```

`Get Librefs` returns the short library names, not the longer logical names. However, you can use either one, and metadata security will still be applied for metadata-defined libraries.

SAS Connect Lib Refs(libref)

Description

Connects a SAS libref on the active SAS server connection.

Returns

1 if successful and 0 otherwise.

SAS Deassign Lib Refs("libref")

Description

De-assign a SAS libref on the active global SAS server connection.

Returns

1 if successful; 0 otherwise.

Arguments

`libref` A quoted string that contains the library reference to de-assign.

SAS Disconnect()

Description

Disconnect the active global SAS connection, if any.

Returns

1 if a SAS connection exists and was successfully disconnected, 0 otherwise.

Arguments

None.

SAS Export Data(dt, "library", "dataset", <named_arguments>)

Description

Exports a JMP data table to a SAS data set in a library on the active global SAS server connection.

Returns

1 if the data table was exported successfully; 0 otherwise.

Arguments

`dt` data table or a reference to a data table.

`"library"` the library to which to export the data table.

`"dataset"` the name of the new SAS data set.

Optional Named Arguments

`Columns(list)|Columns(col1, col2, ...)` A list of columns or a comma-separated list of columns.

`Password("password")` A quoted string that contains the password to serve as the READ, WRITE, and ALTER password for the exported SAS data set. If the exported data set is replacing an existing data set with an ALTER password, this password is used as the ALTER password for overwriting the data set. If *`Password`* is specified, values for *`ReadPassword`*, *`WritePassword`*, and *`AlterPassword`* are ignored.

`ReadPassword("password")` A quoted string that contains the password to serve as the READ password for the exported SAS data set.

`WritePassword("password")` A quoted string that contains the password to serve as the WRITE password for the exported SAS data set.

`AlterPassword("password")` A quoted string that contains the password to serve as the ALTER password for the exported SAS data set. If the exported data set is replacing an existing data set with an ALTER password, this password is used as the ALTER password for overwriting the data set.

`PreserveSASColumnNames(0|1)` A Boolean. If true *and* the JMP data table originally came from SAS, the original SAS column names are used in the exported SAS data set. The default value is `False`.

`PreserveSASFormats(0|1)` A Boolean. If true *and* the JMP data table originally came from SAS, the original SAS formats and informats are applied to the columns in the exported SAS data set. The default value is `True`.

`ReplaceExisting(0|1)` A Boolean. If true, an existing SAS data set with the specified name in the specified library is replaced by the exported SAS data set. If false, a data set with the specified name already exists in the specified library; the export is stopped. The default value is false.

`SaveJMPMetadata(0|1)` Includes SAS 9.4 Extended Attributed to store JMP metadata (such as table script and column properties). Default is 0 (disabled).

`HonorExcludedRows(0|1)` A Boolean. If true, any rows in the JMP data table that are marked as excluded are not exported. The default value is false.

Notes

Information about the export is sent to the log.

SAS Get Data Sets("libref")

Description

Returns a list of the data sets defined in a SAS library.

Returns

A list of quoted strings.

Arguments

`libref` A quoted string that contains the SAS libref or friendly library name associated with the library for which the list of defined SAS data sets is returned.

SAS Get File("source", "dest", "encoding")

Description

Get a file from the active global SAS server connection. JMP creates a FILENAME statement (with an encoding, if specified) and uses it to read the file on the SAS server.

Returns

1 if successful, 0 otherwise.

Arguments

`source` A quoted string that contains the full path of file on the server to be downloaded to the client machine.

`dest` A quoted string that contains the full path on the client machine for where to put the copy of the file downloaded from the server.

`encoding` A quoted string that contains the encoding used in the file (for example, `"utf-8"`). The server must support the specified encoding.

SAS Get File Names("fileref")

Description

Get a list of filenames found in the given fileref on the active global SAS server connection.

Returns

A list of quoted strings.

Arguments

`fileref` A quoted string that contains the name of the fileref from which to retrieve filenames.

SAS Get File Names In Path("path")

Description

Get a list of filenames found in the given path on the active global SAS server connection.

Returns

A list of quoted strings.

Arguments

`path` A quoted string that contains the directory path on the server from which to retrieve filenames.

SAS Get File Refs()

Description

Get a list of the currently defined SAS filerefs on the active global SAS server connection.

Returns

List of two lists. The first list is a list of quoted strings of fileref names. The second is a corresponding list of strings of physical names.

SAS Get Lib Refs(<named arguments>)

Description

Get a list of the currently defined SAS librefs on the current global SAS server connection.

Returns

A list of quoted strings.

Named Arguments

`Friendly Names(0|1)` Optional, Boolean. If `True`, then for any libraries that have friendly names (metadata-defined libraries), the friendly name is returned rather than the 8-character libref.

SAS Get Log()

Description

Retrieve the SAS Log from the active global SAS server connection.

Returns

A quoted string.

SAS Get Output()

Description

Retrieve the listing output from the last submission of SAS code to the current global SAS server connection.

Returns

A quoted string.

SAS Get Results()

Description

Retrieve the results of the previous SAS Submit as a scriptable object, which allows significant flexibility in what to do with the results.

Returns

A SAS Results Scriptable object.

SAS Get Var Names(*string*, <"dataset">, <password("password")>)

Description

Retrieves the variable names contained in the specified data set on the current global SAS server connection.

Returns

A list of quoted strings.

Arguments

`string` A quoted string that contains one of the following:

- The name of the SAS Library containing the SAS data set to be imported. In that case, the *`dataset`* name argument is required.
- The full member name of the SAS data set to be imported, in the form "libname.membername".
- The SAS Folders tree path to a logical SAS data table to be imported. This option requires a connection to a SAS 9.3 or higher Metadata Server.

`dataset` (Optional) A quoted string that contains the name of the data set from which to retrieve variable names.

`password("password")` (Optional) A quoted string that contains the read password for the data set. If this is not provided and the data set has a read password, the user is prompted to enter it.

`SAS Import Data(`*`string`*`, <"dataset">, <named arguments>)`

Description

Import a SAS data set from the active global SAS server connection into a JMP table.

Returns

JMP Data Table object.

Arguments

`string` A quoted string that contains one of the following:

- The name of the SAS Library containing the SAS data set to be imported. In that case, the *"dataset"* name argument is required. The name can be a friendly metadata library name or a SAS 8-character library name.
- The full member name of the SAS data set to be imported, in the form "libname.membername".
- The SAS Folders tree path to a logical SAS data table to be imported. This option requires a connection to a SAS 9.3 or higher Metadata Server.

`dataset` (Optional) A quoted string that contains the name of the data set.

Optional Named Arguments

`Columns("list")|Columns(col1, col2, ...)` A quoted string list or multiple strings that contain the names of columns to include in the import.

`ConvertCustomFormats(0|1)` The default value is True (1). If True and custom formats are found in the SAS data set being imported, an attempt is made to convert the SAS custom formats to JMP value labels for those columns.

`Invisible(0|1)` The default value is False (0). If true, the JMP data table is hidden from view. The data table appears only in the JMP Home Window and the Window menu. Hidden data tables remain in memory until they are explicitly closed, reducing the amount of memory that is available to JMP. To explicitly close the hidden data table, call `Close(dt)`, where *dt* is the data table reference returned by SASImportData.

`Where("filter")` A quoted string that contains the filter to use when importing data, as in `Where("salary<50000")`.

`Password("password")` A quoted string that contains the read password for the data set. If this is not provided and the data set has a read password, the user is prompted to enter it.

`UseLabelsForVarNames(0|1)` If `True`, the labels from the SAS data set become the column names in the resulting JMP table. If `False`, the variable names from the SAS data set become the column names in the JMP table. The default value is `False`.

`RestoreJMPMetadata(0|1)` Includes SAS 9.4 Extended Attributed to store JMP metadata. Default is 0 (disabled).

`Sample(named arguments)` optional, named. Allows a random sample of the SAS data set to be imported into JMP. If both Where and Sample are specified, the WHERE clause is used to filter the SAS data set first, and then a random sample of the resulting rows is taken based on the arguments supplied to Sample. Note that Sample uses PROC SURVEYSELECT on the SAS server, which is available only if the SAS/STAT package is licensed and installed on that server. The documentation for PROC SURVEYSELECT might be helpful in understanding how sampling is performed. By default (if no arguments are supplied), a 5% simple random sample is taken. Here are the available arguments (all optional):

- `Simple | Unrestricted`: If `Simple` is specified, sampling is performed without replacement. If `Unrestricted` is specified, sampling is performed with replacement. These two options are mutually exclusive and only one can be specified.
- `SampleSize(int) | N(int)`: Total number of rows for the sample, or number of rows per strata level for stratified sampling
- `SampleRate(number) | Rate(number) | Percent(number)`: Specifies the sampling rate. For stratified sampling, the rate is applied to each strata level. Note that the supplied value is assumed to be a percentage, so `SampleRate(3.5)` means a 3.5% sampling rate.
- `Strata({col1, col2, ...}) | Strata(col1, col2, ...)`: Perform stratified random sampling using the column names supplied as Strata variables.
- `NMin(int)`: Minimum number of rows (either overall or per strata level for stratified sampling) to return. Only applies to rate-based sampling.
- `NMax(int)`: Maximum number of rows (either overall or per strata level for stratified sampling) to return. Only applies to rate-based sampling.
- `Seed(int)`: Number to use as the seed for sampling. Useful for replicating the same sample. By default, the seed is a random number based on time of day. See PROC SURVEYSELECT documentation.
- `OutputHits(0|1)`: Boolean; the default value is false. When doing Unrestricted sampling, if the same row of the input data set is selected more than once, by default that row still appears only once in the resulting JMP data table, with the NumberHits column indicating the number of times that the row was selected. Setting OutputHits to true causes an input row that is selected multiple times to appear multiple times in the resulting JMP data table.
- `SelectAll(0|1)`: Boolean, the default value is true. If `SelectAll` is true, PROC SURVEYSELECT selects all stratum rows whenever the stratum sample size exceeds the total number of rows in the stratum. If `SelectAll` is false and PROC SURVEYSELECT finds a case where the stratum sample size exceeds the total number

of rows in a given stratum, an error results and sampling fails. `SelectAll` only applies to Simple random sampling.

`SQLTableVariable(0|1)` If `True`, an SQL table variable is created in the resulting JMP table that shows the SQL that was submitted to SAS to obtain the data. If `False`, the SQL table variable is not created. The default value is `True`. If an SQL table variable is created and the data set required a read password, the password is masked.

SAS Import Query("sqlquery", <named arguments>)

Description

Execute the requested SQL query on the current global SAS server connection, importing the results into a JMP data table.

Returns

JMP Data Table object.

Arguments

`sqlquery` A quoted string that contains the SQL query to perform and from which to import the result.

Optional Named Arguments

`ConvertCustomFormats(0|1)` The default value is true. If true and custom formats are found in the SAS data set being imported, an attempt is made to convert the SAS custom formats to JMP value labels for those columns.

`Invisible(0|1)` The default value is false. If true, the JMP data table is hidden from view. The data table appears only in the JMP Home Window and the Window menu. Hidden data tables remain in memory until they are explicitly closed, reducing the amount of memory that is available to JMP. To explicitly close the hidden data table, call `Close(dt)`, where *`dt`* is the data table reference returned by `SAS Import Query`.

`UseLabelsForVarNames(0|1)` The default value is true. If `True`, the labels from the SAS data set become the column names in the resulting JMP table. If `False`, the variable names from the SAS data set become the column names in the JMP table.

`RestoreJMPMetadata(0|1)` Includes SAS 9.4 Extended Attributes to store JMP metadata. Default is 0 (disabled).

`SQLTableVariable(0|1)` The default value is true. If `True`, an SQL table variable is created in the resulting JMP table that shows the SQL that was submitted to SAS to obtain the data. If `False`, the SQL table variable is not created. If an SQL table variable is created and the data set required a read password, the password is masked.

SAS Is Connected()

Description

Discovers whether there is an active global SAS server connection.

Returns

1 if an active global SAS connection exists, 0 otherwise.

SAS Is Local Server Available()

Description

Returns True if a local SAS Server is available; otherwise, returns False.

SAS Load Text File("path")

Description

Download the file specified in path from the active global SAS server connection and retrieve its contents as a quoted string.

Returns

A quoted string.

Arguments

"path" A quoted string that contains the full path on the server of the file to download and retrieve the contents as a string.

SAS Name("name")

SAS Name({list of names})

Description

Converts JMP variable names to SAS variable names by changing special characters and blanks to underscores and various other transformations to produce a valid SAS name.

Returns

A quoted string that contains one or more valid SAS names, separated by spaces.

Argument

"name" A quoted string that represents a JMP variable name; or a list of quoted JMP variable names.

SAS Open For Var Names("path")

Description

Opens a SAS data set only to obtain the names of its variables, returning those names as a list of quoted strings.

Returns

A list of variable names in the file.

Argument

path A quoted string that is a pathname of a SAS data set.

SAS Send File("source", "dest", "encoding")

Description

Send a file from the client machine to the active global SAS server connection. JMP creates a FILENAME statement (with an encoding, if specified) and uses it to save the file on the SAS server.

Returns

1 if successful, 0 otherwise.

Arguments

`source` A quoted string that contains the full path of the file on the client machine to be uploaded to the server.

`dest` A quoted string that contains the full path on the server that receives the file uploaded from the client machine.

`encoding` A quoted string that contains the encoding used in the file (for example, "utf-8"). The server must support the specified encoding.

SAS Submit("sasCode", <named arguments>)

Description

Submit some SAS code to the active global SAS server connection.

Returns

1 if successful, 0 otherwise.

Arguments

`sasCode` A quoted string that contains the SAS code to submit.

Optional Named Arguments

`Async(0|1)` A Boolean. If True (1), the submit occurs asynchronously (in the background). Use the `Get Submit Status()` message on the SAS Server Scriptable Object to determine the status of the submit. The default value is False (0).

`ConvertCustomFormats(0|1)` A Boolean. When SAS data sets generated by submitted SAS code are imported into JMP after the submit completes (see `Open Output Datasets`), the value of `ConvertCustomFormats` determines whether an attempt is made to convert any custom formats found on columns in the SAS data to JMP value labels. The default value is True (1).

`DeclareMacros(var1, var2, ...)` JSL variable names. Provides a simple way to pass the values of JSL variables to SAS as macro variables. Each JSL variable specified should evaluate to a quoted string or numeric value. Fully qualified JSL variables names, only the variable name is sent to SAS. For example, `namespace:variable_name` becomes `variable_name` in SAS.

`GetSASLog(<Boolean|OnError>, <JMPLog|Window>)` A Boolean. If no arguments are supplied, the SAS Log is retrieved and displayed in the location indicated in SAS Integration Preferences. The first argument to `GetSASLog` can be either a Boolean value

or the keyword OnError. If a Boolean value is supplied, true means display the SAS Log, and false means not to display it. OnError instructs JMP to only show the SAS Log if an error occurred in the submit. The second argument to GetSASLog tells JMP where to display the SAS Log. If JMPLog is specified, the SAS Log is appended to the JMP Log. If Window is specified, the SAS Log is opened in a separate window.

`GraphicsDevice(`*`string`*`) or GDevice(`*`string`*`)` A quoted string that specifies a value for the GDEVICE SAS option to be used for graphics generated by the submitted SAS code. The value must be a valid SAS graphics device. The default value is determined in Preferences.

`Interactive(0|1)` JMP includes the `QUIT` statement in the generated wrapper code. Interactive PROCs work even if JMP is generating the ODS wrapper. On every SUBMIT, specify the argument that is part of an interactive sequence. Otherwise, QUIT will be generated in both the prologue-generated and epilogue-generated code.

`NoOutputWindow` A Boolean. If `True`, the SAS Output window containing the listing output from the submission does not appear. The default value is `False`.

`ODS(0|1)` If true, additional SAS code is submitted causing ODS results to be generated for the submitted SAS code. The default value is determined in Preferences.

`ODSFormat(`*`string`*`)` A quoted string that determines the format of generated ODS results. Valid values are “HTML”, “RTF”, and “PDF”. The default value is determined in Preferences.

`ODSGraphics(0|1)` If true, ODS statistical graphics are generated for the submitted SAS code. Setting `ODSGraphics` to true causes ODS to also be set to true. The default value is determined in Preferences.

`ODSStyle(`*`string`*`)` A quoted string that specifies the ODS Style to use when generating ODS results. *`String`* must be a valid SAS Style. The default value is determined in Preferences.

`ODSStyleSheet(path)` A quoted string that specifies a local CSS style sheet to use when formatting generated ODS results. *`Path`* must be a path to a CSS file valid for the client machine (the machine running JMP). The default value is determined in Preferences.

`OnSubmitComplete(script)` A quoted string that specifies a JSL script that should be run when the submit completes. This is especially useful for asynchronous submits. If script is the name of a defined JSL function, that function is executed, with the SAS Server scriptable object passed as the first argument.

`OpenODSResults(0|1)` If true, ODS results that are generated by the submitted SAS code (due to ODS being true) are automatically opened after the submit completes. The default value is True (1).

`OpenOutputDatasets(<All|None|dataset1, dataset2, ...>)` JMP detects when submitted SAS code creates new SAS data sets. `OpenOutputDatasets` (which can be abbreviated `OutData`) determines what, if anything, is done with those data sets with the SAS Submit completes. If `All` is specified, all data sets generated by the SAS code are imported into JMP when the SAS Submit completes. If `None` is specified, none of the

generated data sets are imported. If there are specific data sets known to be generated by the submitted SAS code that you want to be imported into JMP when the SAS submit completes, you can alternative provide their names, and only the requested data sets are imported. The default value is determined in Preferences.

`Title(`*`string`*`)` A quoted string that specifies the window title to use for the window that displays ODS output from the submit.

`SAS Submit File("filename", <named arguments>)`

Description

Submit a SAS code file to the active global SAS server connection.

Returns

1 if successful; 0 otherwise.

Arguments

`filename` A quoted string that contains the name of file containing SAS code to submit.

Named Arguments

Same as for SAS Submit.

SQL Functions

Note: Database table names that contain the characters `$# -+/%()&|;?` are not supported.

`As SQL Expr(x, <style>)`

Description

Converts an expression to code that you can use in an SQL Select statement. Use `Expr(...)` for literal expressions. Use `NameExpr(name)` for expressions stored in a variable. Otherwise, the expression returns the expression to convert.

Returns

A quoted string that contains the expression converted to valid SQL syntax for use in an SQL Select statement.

```
New SQL Query(Connection
("ODBC:connection_string")|("SAS:connection_string"),
Select(Column("column", "t1")), From(Table("table", <Schema("schema")>,
```

```
<Alias("t1")>)), <Options(JMP 12 Compatible(1)|JMP 13 Compatible(1)|Run on
Open(1)))>
```

```
New SQL
Query(Connection("ODBC:connection_string;")|("SAS:connection_string;"),
Custom("SELECT col1, col2, col3 FROM table;")), <Options(JMP 12
Compatible(1)|JMP 13 Compatible(1)|Run on Open(1))>
```

Creates an SQL Query object for the specified connection, columns, data table, or for the custom SQL query.

Returns

A data table that contains the queried data. The data table includes the quoted SQL query string and table scripts for modifying and updating the query.

Arguments

`Connection` The quoted string for an ODBC or SAS connection.

`Select` The column that you want to select and its alias.

`From` The table that is queried and the optional schema and column alias.

`Custom` An SQL statement that selects columns from the specified table.

`Version` The minimum JMP version required to open the query. If this condition is not met, a message regarding compatibility is written to the log, and the query does not open.

`Options` Boolean. `JMP 12 Compatible` is included in generated scripts when you select the Query Builder preference to create a JMP 12 compatible option or select the corresponding Query Builder red triangle menu option. The option enables JMP 12 users to run a JMP 13 query that might contain compatibility issues. Include `Run on Open(1)` to run the query when opened rather than opening the query in edit mode.

Example

```
New SQL Query(
   Connection(
      "ODBC Connection String..."
   ),
   QueryName( "g6_Movies" ),
   Select( Column( "ItemNo", "t1" ), Column( "LengthMins", "t1" ), Column(
   "Genre", "t1" ) ),
   From( Table( "g6_Movies", Schema( "SQBTest" ), Alias( "t1" ) ) )
) << Run Background( On Run Complete( dt = queryResult ) );

Show( dt );
```

Notes

- Query Builder creates a symbol called `queryResult` in the context of an `On Run Complete()` script. This is a reference to the data table imported by the query. `queryResult` enables you to assign a global variable to the table for later use.
- `New SQL Query()` always closes the ODBC connection after performing the query.

- After you run `New SQL Query()`, the database connection that you make appears in the Database > Open Table window as an available connection. But the connection you see in the Database > Open Table window is due to JMP keeping track of how to connect to that database; no ODBC connection is left open.
- There is not a command to close an ODBC connection established with `New SQL Query()` because `New SQL Query()` always closes the connection after performing the query.
- After you run `New SQL Query()`, the database connection appears in the Database > Open Table window as an available connection. JMP is keeping track of how to connect to that database; the ODBC connection was not left open.

```
Query(<<dt1|Table(dt1, alias1)>, ..., <dtN, aliasN)>>, <private |
invisible>, <scalar>, sqlStatement )
```

Description

Performs a SQL query on selected data tables.

Returns

The result of the query, either a data table or a single value.

Arguments

`dt1, dtN` (Optional) A variable that has been assigned to the data table.

`Table` (Optional) Passes a reference to the data table.

`alias1, aliasN` Specifies the alias of the database table.

`private` (Optional) Avoids showing the resulting data table. Using a private data table speeds the process of getting to the data; it does not save the computer from allocating the memory necessary to hold the data table data.

`invisible` (Optional) Hides the resulting data table from view. The data table appears only in the JMP Home Window and the Window menu. Hidden data tables remain in memory until they are explicitly closed, reducing the amount of memory that is available to JMP. To explicitly close the hidden data table, call `Close(dt)`, where *`dt`* is the data table reference.

`scalar` (Optional) Indicates that the query returns a single value.

`sqlStatement` Required. The SQL statement, most likely a SELECT statement. The statement must be the last argument.

Example

The following example selects all data for students who are older than 14 years of age.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
result = Query( Table( dt, "t1" ), "SELECT * FROM t1 WHERE age > 14;" );
```

See Also

Appendix A, “SQL Functions Available for JMP Queries”

Statistical Functions

Arc Finder(X(col), Y(col), Group(lot, wafer))

Description

Finds arcs in the point data and creates a new column that identifies the arcs.

Example

```
dt = Open( "$SAMPLE_DATA/Wafer Stacked.jmp" );
Arc Finder(
   Group( :Lot, :Wafer ),
   X( :X_Die ),
   Y( :Y_Die ),
   Min Distance( 12 ), // minimum distance among 3 points to seed an arc
   Min Radius( 15 ), // minimum radius of the acceptable arc
   Max Radius( 2000 ), // maximum radius of acceptable arc
   Max Radius Error( 2 ), // how close a point needs to be added
   Min Arc Points( 5 ), // how many points to define an arc
   Number of Searches( 500 ), // how many random probes of data
   Max Number Arcs( 3 ) // number of arcs searched for
);
dt << Color or Mark by Column( :Arc Number );
dt << Graph Builder(
   Size( 1539, 921 ),
   Variables( X( :X_Die ), Y( :Y_Die ), Wrap( :Lot_Wafer Label ), Color( :Arc
   Number ) ),
   Elements( Points( X, Y, Legend( 6 ) ) )
);
```

Notes

- The function is scaled for data that have a range of 30 to 50 units.
- The function is suitable only for data that are subset to the interesting defect points.
- It is not suitable when the density of points is high.

ARIMA Forecast(column, length, model, estimates, from, to)

Description

Determines the forecasted values for the specified rows of the specified column using the specified model and estimates.

Returns

A vector of forecasted values for *column* within the range defined by *from* and *to*.

Arguments

`column` A data table column.

`length` Number of rows within the column to use.

`model` Messages for Time Series model options.

`estimates` A list of named values that matches the messages sent to `ARIMA Forecast()`. If you perform an ARIMA Forecast and save the script, the estimates are part of the script.

`from, to` Define the range of values. Typically, *`from`* is between 1 and *`to`*, inclusive. If *`from`* is less than or equal to 0, and if *`from`* is less than or equal to *`to`*, the results include filtered predictions.

`Best Partition(xindices, yindices, <<Ordered, <<Continuous Y, <<Continuous X)`

Description

Experimental function to determine the optimal grouping.

Returns

A list.

Arguments

`xindices, yindices` Same-dimension matrices.

`Col Cumulative Sum(name, <By var, ...>)`

`Cumulative Sum(name)`

Description

Returns the cumulative sum for the current row. `Col Cumulative Sum` supports By columns, which do not need to be sorted.

Arguments

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

`Col Maximum(name, <By var, ...>)`

`Col Max(name)`

Description

Calculates the maximum value across all rows of the specified column. The result is internally cached to speed up multiple evaluations.

Returns

The maximum value that appears in the column.

Arguments

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Col Mean(name, <By var, ...>)`

Description

Calculates the mean across all rows of the specified column. The result is internally cached to speed up multiple evaluations.

Returns

The mean of the column.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Col Median(name, <By var, ...>)`

Description

Calculates the median across all rows of the specified column. The ordering is cached internally to speed up multiple evaluations.

Returns

The median of the column.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Col Minimum(name, <By var, ...>)`

`Col Min(name)`

Description

Calculates the minimum value across all rows of the specified column. The result is internally cached to speed up multiple evaluations.

Returns

The minimum value that appears in the column.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Col Moving Average(name, options, <By var, ...>)`

`Moving Average(name, options)`

Description

Returns the moving average over a given interval based at the current row. `Col Moving Average` supports By columns.

Arguments

`name` A column name.

`Weighting(1|0|n)` Required positional argument. Determines how the values are weighted. 1 indicates uniform weighting. 0 indicates incremental weighting (a ramp or triangle). Any other number is the parameter for an exponential moving average (EWMA or EMA).

`Before(1|0|n)` Positional argument. Controls the size of the range (or window) by including the specified number of items before the current item in the average (in addition to the current item). The default value, -1, means all of the preceding items.

`After(1|0|n)` Positional argument. Controls the size of the range (or window) by including the specified number of items after the current item in the average (in addition to the current item). The default value, 0, means no following items.

`Partial Window is Missing` Boolean positional argument. Controls how missing values are treated. By default, missing values are ignored. 0 computes the average of partial windows.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Examples

```
// equal weighting of a five-item lagging range
Col Moving Average( x, 1, 4 );

// ramp weighting of all preceding items
Col Moving Average( x, 0 );

// triangle weighting of a five-item centered range
Col Moving Average( x, 0, 2, 2 );

// exponential weighting of all preceding items
Col Moving Average( x, 0.25 );
```

Col N Missing(name, <By var, ...>)

Description

Calculates the number of missing values across all rows of the specified column. The result is internally cached to speed up multiple evaluations.

Returns

The number of missing values in the column.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

Col Number(name, <By var, ...>)

Description

Calculates the number of nonmissing values across all rows of the specified column. The result is internally cached to speed up multiple evaluations.

Returns

The number of nonmissing values in the column.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Col Quantile(name, p, <ByVar>)`

Description

Calculates the specified quantile *p* across all rows of the specified column. The result is internally cached to speed up multiple evaluations.

Returns

The value of the quantile.

Argument

`name` A column name.

`p` A specified quantile *p* between 0 and 1.

`ByVar` (Optional) A By group.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Col Quantile( :height, .5 );
   63
```

63 is the 50th percentile, or the median, of all rows in the height column.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Col Rank(column, <ByVar, ...>, <<tie("average"|"arbitrary"|"row"|"minimum"))`

Description

Ranks each row's value, from 1 for the lowest value to the number of columns for the highest value. Ties are broken arbitrarily.

Arguments

`column` The column to be ranked.

`ByVar` (Optional) A By variable to compute statistics across groups of rows.

`<<tie` Determines how the tie is broken. A tie occurs when the values being ranked are the same. For the data `[33 55 77 55]`, `33` has rank 1 and 77 has rank 4, and the question is how to assign ranking for the `55`s. `average` reports the average of the possible rankings, 2.5, for both 55s. `arbitrary` matches JMP 12 behavior by assigning the possible rankings in an unspecified order, which could be 2 and 3 or 3 and 2. `row` assigns the ranks in the order that they originally appear. (The first 55 would be 2 and the second 55 would be 3.)

`minimum` gives both values the lowest possible rank, 2.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

Col Simple Exponential Smoothing(column, alpha, <ByVar>)

Description

Returns the simple exponential smoothing prediction for the current row using smoothing weight *alpha*.

Arguments

`column` The column of time series observations.

`alpha` The smoothing weight.

`ByVar` (Optional) A By variable to compute predictions across groups of rows. By variables do not need to be presorted.

Notes

The predicted value for row *t* is given by the following:

```
Predicted[t] = alpha * Observed[t-1] + (1-alpha) * Predicted[t-1]
```

By definition, `Predicted[1] = Observed[1]`.

Col Standardize(name,<By var, ...>)

Description

Calculates the column mean divided by the standard deviation across all rows of the specified column.

Returns

The standardized mean.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. If a By variable is specified, the values are standardized against the mean and standard deviation of their corresponding By variable group.

Notes

Standardizing centers the variable by its sample standard deviation. Thus, the following commands are equivalent:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "stdht", Formula( Col Standardize( height ) ) );
dt << New Column( "stdht2",
  Formula( (height - Col Mean( height )) / Col Std Dev( height ) )
);
```

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

Col Std Dev(name,<By var, ...>)

Description

Calculates the standard deviation across rows in a column. The result is internally cached to speed up multiple evaluations.

Returns

The standard deviation.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Col Sum(name,<By var, ...>)`

Description

Calculates the sum across rows in a column. Calculating all missing values (`Col Sum(.,.)`) returns missing. The result is internally cached to speed up multiple evaluations.

Returns

The sum.

Argument

`name` A column name.

`By var` (Optional) A By variable to compute statistics across groups of rows. Use the By variable in a column formula or in a `For Each Row()` function.

Notes

If a data value is assigned by a column property (such as Missing Value Codes), use `Col Stored Value()` to base the calculation on the value stored in the column instead.

See Also

"Col Stored Value(<dt>, col, <row>)"

`Fit Censored(Distribution("name"), YLow(vector) | Y(Vector), <YHigh(vector)>, <Weight(vector)>, <X(matrix)>, <Z(matrix)>, <HoldParm(vector)>, <Use random sample to compute initial values(percent)>, <Use first N observations to compute initial values(nobs)>)`

Description

Fits a distribution using censored data.

Returns

A list that contains parameter estimates, the covariance matrix, the log-likelihood, the AICc, the BIC, and a convergence message. See *Fitting Linear Models*.

Arguments

`Distribution("name")` The quoted name of the distribution to fit.

`YLow(vector) | Y(Vector)` If you do not have censoring, then use `Y` and an array of your data, and do not specify `YHigh`. If you do have censoring, then specify `YLow` and `YHigh` as the lower and upper censoring values, respectively.

Optional Arguments

`YHigh(vector)` A vector that contains the upper censoring values. Specify this only if you have censoring and also specify YLow.

`Weight(vector)` A vector that contains the weight values.

`X(matrix)` The regression design matrix for location.

`Z(matrix)` The regression design matrix for scale.

`HoldParm(vector)` An array of specified parameters. The parameters should be nonmissing where they are to be held fixed, and missing where the are to be estimated. This is primarily used to test hypotheses that certain parameters are zero or some other specific value.

`Use random sample to compute initial values(percent)` A percent of the observations to be used in the computation of the initial values. Specify this if the data vector is large.

`Use first N observations to compute initial values(nobs)` A number of observations at the start of the data vector to be used in the computation of the initial values. Specify this if the data vector is large.

Fit Circle(Xvec, Yvec)

Description

Fits a circle that best goes through three or more points using a least squares approach. If only three points are specified, a direct solution can be found, and the sum of squared errors is zero.

Returns

A list that contains the X and Y coordinates of the center point of the circle, the length of the radius, and the sum of squared errors.

Arguments

`Xvec` Vector of X coordinates of three or more points.

`Yvec` Vector of Y coordinates of three or more points.

Syntax

```
{Xcenter, yCenter, radius, SSE} = Fit Circle(Xvec, Yvec)
```

Hier Clust(x)

Description

Returns the clustering history for a hierarchical clustering using Ward's method (without standardizing data).

Argument

`x` A data matrix.

IRT Ability(Q1, <Q2, Q3, ... Qn,> parmMatrix)

Description

Returns scores for the latent variable in an item response theory model with n binary items and a matrix of known parameters. The parameter matrix should contain as many rows as there are parameters in the model and as many columns as there are items in the analysis.

Arguments

`Q1, Q2, ..., Qn` A set of n binary items.

`parmMatrix` A matrix of parameters from an item response theory model.

`KDE(vector, <named arguments>)`

Description

Returns a kernel density estimator with automatic bandwidth selection.

Argument

`vector` A vector.

Optional Named Arguments

`<<weights` Must be a vector of the same length as vector, and can contain any nonnegative real numbers. `Weights` represents frequencies, counts, or similar concepts.

`<<bandwidth(n)` A nonnegative real number. Enter a value of 0 to use the bandwidth selection argument.

`<<bandwidth scale(n)` A positive real number.

`<<bandwidth selection(n)` *n* must be 0, 1, 2, or 3, corresponding to Sheather and Jones, Normal Reference, Silverman rule of thumb, or Oversmoother, respectively.

`<<kernel(n)` *n* must be 0, 1, 2, 3, or 4, corresponding to Gaussian, Epanechnikov, Biweight, Triangular, or Rectangular, respectively.

`LenthPSE(x)`

Description

Returns Lenth's pseudo-standard error of the values within a vector.

Argument

`x` A vector.

`Max()`

See "Maximum(var1, var2, ...)".

`Maximum(var1, var2, ...)`

`Max(var1, var2, ...)`

Description

Returns the maximum value of the arguments or of the values within a single matrix or list argument. If multiple arguments are specified, they must be all numeric values or all quoted strings.

`Mean(var1, var2, ...)`

Description

Returns the arithmetic mean of the arguments or of the values within a single matrix or list argument.

`Median(var1, var2, ...)`

Description

Returns the median of the arguments or of the values within a single matrix or list argument.

`Min()`

See "Minimum(var1, var2, ...)".

`Minimum(var1, var2, ...)`

`Min(var1, var2, ...)`

Description

Returns the minimum value of the arguments or of the values within a single matrix argument. If multiple arguments are specified, they must be either all numeric values or all quoted strings.

`N Missing(expression)`

Description

Rowwise number of missing values in variables specified.

`Number(var1, var2, ...)`

Description

Rowwise number of nonmissing values in variables specified.

`Product(i=initialValue, limitValue, bodyExpr)`

Description

Multiplies the results of *bodyExpr* over all *i* until the *limitValue* and returns a single product.

`Quantile(p, arguments)`

Description

Returns the p^{th} quantile of the arguments. The first argument can be a scalar or a matrix of values between 0 and 1. The remaining arguments can also be specified as values within a single matrix or list argument.

`Range(var1, var2, ...)`

Description

Returns the minimum and maximum values of the arguments. The result is returned as a two-element row vector that contains the minimum and the maximum.

`Robust PCA(X, <Lambda(2/sqrt(max(nrow, ncol)))>, <tolerance=1e-10>, <maxit(75)>, <Center(1)>, <Scale(1)>)`

Description

Performs a sequence of singular value decompositions and thresholding steps to decompose the data matrix into a low-rank matrix and a sparse matrix of residuals.

Returns

`A` The low-rank matrix estimation.

`E` The sparse matrix of residuals.

`S` A vector of singular values.

Arguments

`X` A data matrix.

`Lambda` Specifies a value greater than 0 that determines the sparsity of the matrix of residuals. For larger values of Lambda, the matrix of residuals is more sparse.

`tolerance` The convergence criterion.

`maxit` The maximum number of SVD iterations.

`Center` Centers the data prior to performing the SVD iterations.

`Scale` Scales the data prior to performing the SVD iterations

`Std Dev(var1, var2, ...)`

Description

Rowwise standard deviation of the variables specified.

`Sum(var1, var2, ...)`

Description

Rowwise sum of the variables specified. Calculating all missing values (`Sum(.,.)`returns missing.

`SSQ(x1, ...)`

Description

Returns the sum of squares of all elements. Takes numbers, matrices, or lists as arguments and returns a scalar number. Skips missing values.

`Summarize(<dt>, <by>, <count>, <sum>, <mean>, <min>, <max>, <stddev>, <corr>, <quantile>, <first>)`

Description

Gathers summary statistics for a data table and stores them in global variables.

Returns

None.

Arguments

`dt` Optional positional argument: a reference to a data table. If this argument is not in the form of an assignment, then it is considered a data table expression.

All other arguments are optional and can be included in any order. Typically, each argument is assigned to a variable so you can display or manipulate the values further.

`name=By(col | list | Eval)` Using a BY variable changes the output from single values for each statistic to a list of values for each group in the BY variable.

`Summarize YByX(X(<x columns>, Y (<y columns>), Group(<grouping columns>), Freq(<freq column>), Weight(<weight column>))`

Description

Calculates all Fit Y by X combinations on large-scale data sets.

Returns

A data table of *p*-values and LogWorth values for each Y and X combination.

Arguments

`X(col)` The factor columns used in the fit model.

`Y(col)` The response columns used in the fit model.

`Group(gcol)` The group of columns used in the fit model.

`Freq(col)` The frequency (for each row) column used in the fit model.

`Weight(col)` The importance (or influence) column used in the fit model.

Notes

Performs the same function as the Response Screening platform.

`Summation(init, limitvalue, body)`

Description

Summation sums the results of the *body* statement(s) over all *i* to return a single value.

`Tolerance Limit(1-alpha, p, n)`

Description

Constructs a *1-alpha* confidence interval to contain proportion *p* of the means with sample size *n*.

Transcendental Functions

`Arrhenius(n)`

Description

Converts the temperature *n* to the value of explanatory variable in Arrhenius model.

Returns

11604.5181215503/(n+273.15)

Argument

n Temperature in Celsius.

Notes

This is frequently used as a transformation.

`Arrhenius Inv(n)`

Description

The inverse of the Arrhenius function. Converts the value *n* to the temperature in Celsius.

Returns

(11604.5181215503/n)-273.15

Argument

n The value of the converted explanatory variable in Arrhenius model.

Notes

This is frequently used as a transformation.

`Beta(a, b)`

Description

$$\frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

Returns

Returns the beta function.

Arguments

`a, b` numbers

`Cytometry Logicle(x, T, W, M, A)`

Description

Computes a cytometry logicle transformation. For more information about the logicle transformation, see Moore and Parks (2012).

`Cytometry Logicle Inverse(y, T, W, M, A)`

Description

Computes the inverse cytometry logicle transformation. For more information about the logicle transformation, see Moore and Parks (2012).

`Digamma(n)`

Description

The derivative of the log of the gamma function (`LGamma`).

Returns

The digamma function evaluated at *n*.

Argument

`n` A number

`Exp(a)`

Description

Raises e to the power *a*.

Returns

e^a.

Argument

`a` A number

Equivalent Expression

`e()^a`

`ExpM1(x)`

Description

Returns a more accurate calculation of *Exp(x)-1* when *x* is very small.

`Factorial(n)`

Description

Multiplies all numbers 1 through *n*, inclusive

Returns

The product.

Arguments

`n` Any integer

Notes

One and only one argument must be specified.

`FFT({list}, <named arguments>)`

Description

Conducts a Fast Fourier Transformation (FFT) on a list of matrices.

Returns

The function takes one matrix, or a list of matrices for complex numbers. The returned value is a list of two matrices with the same dimensions as the first argument.

Argument

`List` A list of one or two matrices. If one is provided, it is considered to be the real part. If two are provided, the first is the real part and the second is the imaginary part. Both matrices must have the same dimensions, and both must have more than one row.

Optional Named Arguments

`<<inverse(Boolean)` If true (1), an inverse FFT is conducted.

`<<multivariate(Boolean)` If true (1), a multivariate FFT is conducted. If false(0), a spatial FFT is conducted.

`<<scale(number)` Multiplies the return values by the specified *number*.

`Fit Transform To Normal(Distribution("name"), Y(vector), <Freq(vector))`

Description

Fits a transformation to normality for a vector of data. This includes the Johnson Sl, Johnson Sb, Johnson Su, and GLog distributions.

Returns

A list that contains parameter estimates, the covariance matrix, the log-likelihood, AICc, a convergence message, and the transformed values. See *Fitting Linear Models*.

`Gamma(t, <limit>)`

Description

The gamma function of *x*, or for each row if *x* is a column:

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx$$

Returns

The gamma.

Arguments

`t` a number or a column

`limit` optional limit. The default is infinity.

Notes

`Gamma(t, limit)` is the same integral as `Gamma(t)` but with the limit of integration that is defined instead of infinity.

`LGamma(t)`

Description

Returns the log gamma function for *t*, which is the natural log of gamma.

`Ln(n)`

Description

Returns the natural logarithm (base *e* logarithm) of *n*.

`Log(n, <base>)`

Description

Returns the natural logarithm (base *e* logarithm) of *n*. An optional second argument lets you specify a different base. For example, `Log(n,3)` for the base 3 logarithm of *n*. The Log argument can be any numeric expression. The expression `Log(e())` evaluates as 1, and `Log(32,2)` is 5.

`Log10(n)`

Description

Returns the common (base 10) logarithm of *n*.

`Log1P(n)`

Description

Same as Log(1 + *x*), except that it is more accurate when *x* is very small.

`Logist(x)`

Description

Returns 1/(1+Exp(-x)), which converts a number in the domain -∞...+∞ into range 0...1. The function is useful in logistic regression.

`Logist Percent(p)`

Description

Similar to the `Logist()` function but with the result scaled from 0 to 100.

`Logit(p)`

Description

Returns `log(p/(1-p))`.

`Logit Percent(p)`

Description

Similar to the `Logit()` function with the argument 0 to 100 rather than 0 to 1.

`N Choose K(n, k)`

Description

This function returns the number of n things taken k at a time ("n choose k") and is computed in the standard way using factorials, as $n!/(k!(n-k)!)$. For example, `NChooseK(5,2)` evaluates as 10.

Notes

This is implemented internally in JMP using lGamma functions. The result is not always an integer.

`Power(a, <b>)`

`a^b`

Description

Raises *a* to the power of *b*.

Returns

The product of *a* multiplied by itself *b* times.

Arguments

a Can be a variable, number, or matrix.

b (Optional) Can be a variable or a number.

Notes

For `Power()`, the second argument (*b*) is optional, and the default value is 2. `Power(a)` returns a^2.

`Root(n, <r>)`

Description

Returns the *r*th root of *n*, where *r* defaults to 2 for square root.

`SbInv(z, gamma, delta, theta, sigma)`

Description

Returns a transformation of a standard normal variable to a double bounded Johnson variable.

`SbTrans(x, gamma, delta, theta, sigma)`

Description

Returns a transformation of a double bounded Johnson variable to a standard normal variable.

`Scheffe Cubic(x1, x2)`

Description

Returns `x1*x2*(x1-x2)`. This function supports notation for cubic mixture models.

`SHASHInv(z, gamma, delta, theta, sigma)`

Description

Returns a transformation of a standard normal variable to a sinh-arcsinh (SHASH) variable. The transformation is calculated as σ*`sinh((arcsinh(z)-`γ`)/`δ`)+`θ.

`SHASHTrans(x, gamma, delta, theta, sigma)`

Description

Returns a transformation of a sinh-arcsinh (SHASH) variable to a standard normal variable. The transformation is calculated as `sinh(`γ+δ*`arcsinh((x-`θ`)/`σ`))`.

`SlInv(z, gamma, delta, theta, sigma)`

Description

Returns a transformation of a standard normal variable to a Johnson Sl variable.

`SlTrans(x, gamma, delta, theta, sigma)`

Description

Returns a transformation of a Johnson Sl variable to a standard normal variable.

`Sqrt(n)`

Description

Returns the square root of *n*.

`Squash(expr)`

Description

An efficient computation of the function 1/ [1 + exp(*expr*)].

`Squish(expr)`

Description

Equivalent to `Squash(-expr)`, or $1/(1+e^{-\text{expr}})$.

`SuInv(z, gamma, delta, theta, sigma)`

Description

Returns a transformation of a standard normal variable to an unbounded Johnson variable.

`SuTrans(x, gamma, delta, theta, sigma)`

Description

Returns a transformation of an unbounded Johnson variable to a standard normal variable.

`Trigamma()`

Description

Returns the trigamma function evaluated at *n*. The trigamma function is the derivative of the digamma function.

Trigonometric Functions

JMP's trigonometric functions expect all angle arguments in radians.

`ArcCosH(x)`

Description

Inverse hyperbolic cosine.

Returns

The inverse hyperbolic cosine of *x*.

Argument

x Any number, numeric variable, or numeric expression.

`ArcCosine(x)`

`ArCos(x)`

Description

Inverse cosine.

Returns

The inverse cosine of *x*, an angle in radians.

Argument

x Any number, numeric variable, or numeric expression.

`ArcSine(x)`

`ArSin(x)`

Description

Inverse sine.

Returns

The inverse sine of *x*, an angle in radians.

Argument

x Any number, numeric variable, or numeric expression.

`ArcSinH(x)`

Description

Inverse hyperbolic sine.

Returns

The inverse hyperbolic sine of *x*.

Argument

x Any number, numeric variable, or numeric expression.

`ArcTangent(x1, <x2=1>)`

`ArcTan(x1 <x2=1>)`

`ATan(x1 <x2=1>)`

Description

Inverse tangent.

Returns

The inverse trigonometric tangent of *x1/x2*, where the result is in the range -Pi()/2, Pi()/2.

Argument

`x1` Any number, numeric variable, or numeric expression.

`x2=1` Specifies *atan2*.

`ArcTanH(x)`

Description

Inverse hyperbolic tangent.

Returns

The inverse hyperbolic tangent of *x*.

Argument

`x` Any number, numeric variable, or numeric expression.

`CosH(x)`

Description

Hyperbolic cosine.

Returns

The hyperbolic cosine of *x*.

Argument

`x` Any number, numeric variable, or numeric expression.

`Cosine(x)`

`Cos(x)`

Description

Cosine.

Returns

The cosine of *x*.

Argument

`x` Any number, numeric variable, or numeric expression. The angle in radians.

`Sine(expr)`

`Sin(expr)`

Description

Returns the sine.

`SinH(expr)`

Description

Returns the hyperbolic sine.

`Tangent(expr)`

`Tan(expr)`

Description

Returns the tangent of an argument given in radians.

`TanH(expr)`

Description

Returns the hyperbolic tangent of its argument.

Utility Functions

`Add(a, b, ...)`

`a+b+...`

Description

Adds the values of the listed arguments. No arguments are changed.

Returns

The sum.

Arguments

For `Add()`, a comma-separated list of variables, numbers, or matrices.

For `a+b`, any number of variables, numbers, or matrices.

Notes

- Any number of arguments is permitted. If no argument is specified, `Add()` returns 0.
- `Add()` returns missing if any arguments are missing. To ignore missing values, use `Sum()`.

See Also

the *Scripting Guide*

See "Sum(var1, var2, ...)"

Beep()

Description

Produces an alert sound.

Returns

Null.

BLOB MD5(blob)

Description

Converts the *blob* argument into a 16-byte blob.

Notes

The 16-byte blob is the MD5 checksum, or the hash, of the source blob.

BLOB Peek(blob, offset, length)

Description

Creates a new blob from a subrange of bytes of the *blob* argument.

Returns

A blob object.

Arguments

`blob` a binary large object.

`offset` An integer that specifies how many bytes into the blob to begin construction. The first byte is at offset 0, the second byte at offset 1.

`length` An integer that specifies how many bytes to copy into the new blob, starting at the offset.

Build Information()

Description

Returns the build date and time, whether it's a release or debug build, and the product name in a quoted comma-delimited string.

Caption({*h*, *v*}, "text", <Delayed(seconds)>, <Font(font)>, <Font Size(size)>, <Text Color("color")>, <Back Color("color")>, <Spoken(Boolean)

Description

Displays a caption window at the location described by *{h, v}* that displays *text*. The caption can be delayed before being displayed by *seconds*, or can be spoken. You can also specify the font type, size, and color and background color.

Returns

Null.

Arguments

`{h, v}` a list with two values. *h* is the horizontal displacement from the top left corner of the monitor in pixels. *v* is the vertical displacement from the top left corner in pixels.

`text` A quoted string or a reference to a string that is to be displayed in the caption.

`Delayed(seconds)` *seconds* is optional delay before displaying the caption. Setting this option causes this caption and all subsequent captions to be delayed by the specified number of seconds.

`Font(font)` Specify the font type.

`Font Size(size)` Specify the font size.

`Text Color("color")` Specify the color of text.

`Back Color("color")` Specify the background color.

`Spoken(Boolean)` Causes *text* to be spoken as well as displayed. The current setting (on or off) remains in effect until switched by another `Caption` statement that includes a `Spoken` setting.

Datafeed()

See "Open Datafeed()".

Debug Break()

When the JSL Debugger is open, this function stops a JSL script from executing at that point in the script. This function is useful for tracking in the debugger under user-specified conditions. If the JSL Debugger is not running, this function does not execute.

Decode64 BLOB(*string*)

Description

Decodes a printable quoted string of base 64 text into a blob.

Returns

A blob.

Arguments

`string` A quoted base-64 encoded string.

Example

```
Decode64 BLOB( "dGhlIHF1aWNrIGJyb3duIGZveA==" );
   Char To BLOB( "the quick brown fox", "ascii~hex" )
```

Decode64 Double(*string*)

Description

Creates a floating point number from a quoted base-64 encoded string.

Returns

A floating point number.

Arguments

`string` A quoted base-64 encoded string.

Disable JMP Live URL(url)

Description

Specifies the URLs that users cannot publish to. This preference must be added to jmpStartAdmin.jsl.

See the Common Tasks chapter in the *Scripting Guide* for details about the location of jmpStartAdmin.jsl.

If a URL appears in both the `Disable JMP Server()` and `Enable JMP Server()` lists, publishing to the URL is disabled.

Example

```
Disable JMP Live URL( "https://public.jmp.com" )
```

Divide(a, b)

Divide(x)

a/b

Description

Divides *a* by *b*. If only one argument is given (`divide(x)`), divides 1 by *x*.

Returns

The quotient of *a*/*b*; or the reciprocal of *x* (1/*x*) if only one argument is provided.

Arguments

`a, b, x` Can be a variable, number, or matrix.

Notes

If both arguments are matrices, it does matrix division.

Empty()

Description

Used in the formula editor for making empty boxes.

Returns

Missing.

Arguments

None.

Enable JMP Live URL(url)

Description

Specifies the URLs that users can publish to. This preference must be added to jmpStartAdmin.jsl.

See the Common Tasks chapter in the *Scripting Guide* for details about the location of jmpStartAdmin.jsl.

If a URL appears in both the `Enable JMP Server()` and `Disable JMP Server()` lists, publishing to the URL is disabled.

Example

```
Enable JMP Live URL( "https://public.jmp.com" )
```

Encode64 BLOB(x)

Description

Encodes a blob into a printable quoted string of base 64 text.

Returns

A quoted base-64 encoded string.

Example

```
Encode64 BLOB( Char To BLOB( "the quick brown fox" ) );
   "dGhlIHF1aWNrIGJyb3duIGZveA=="
```

Encode64 Double(n)

Description

Creates a quoted base-64 encoded string from a floating point number.

Returns

A quoted base-64 encoded string.

Arguments

`n` A floating point number.

Faure Quasi Random Sequence(nDim, nRow)

Description

Generates a sequence of spacefilling quasi random numbers using the Faure sequence.

Get Addin("id")

Description

Retrieves a registered add-in by *id*.

Returns

A scriptable object for the add-in. Returns empty if no add-in with the specified ID was found.

Argument

"id" The ID of an installed add-in.

Get Addins()

Returns

A list of all registered add-ins.

Get Addr Info("address", <port>)

Description

Converts a name to its numeric address.

Returns

A list of quoted strings. The first element is the command (Get Addr Info). The second is the results (for example, "ok" if the command was successful). The third is a list of strings of information. Included in that information is the address that corresponds to the name that was supplied.

Arguments

address A quoted string that specifies the name (for example, "www.sas.com").

port The port of the address.

Get Clipboard()

Description

Returns text from the computer's clipboard. If the content is not text, the result is null.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Copy Table Script( "Distribution" );
s = Get Clipboard();
nw = New Window( "Script", Script Box( s ) );
```

`Get Name Info("address", <port>)`

Description

Converts a numeric address to its name.

Returns

A list of quoted strings. The first element is the command (`GetNameInfo`). The second is the results (for example, "ok" if the command was successful). The third is a list of strings of information. Included in that information is the port name that corresponds to the address that was supplied.

Arguments

`address` A quoted string that specifies the numeric address (for example, `"149.173.5.120"`).

`port` The port of the address.

`Get Platform Preferences(<platform <(option, ...)> ... >)`

`Get Platform Preference(<platform <(option, ...)> ... >)`

Description

Returns the preferences for the specified platforms.

Returns

A list of platform preferences.

Argument

`platform` (Optional) Specifies the platform name. If not specified, all platform preferences are returned. You can specify one or more preferences for a platform.

`option` (Optional) Specifies the preference value. If not specified, all platform preference values are returned.

Notes

Table 2.3 describes the syntax for getting platform preferences.

Table 2.3 Get Platform Preferences() Syntax

Syntax	Description
`Get Platform Preferences( )`	Returns the current option values for all platform preferences.
`Get Platform Preferences( Platform )`	Returns the current option values for the specified platform preferences.

Table 2.3 Get Platform Preferences() Syntax *(Continued)*

Syntax	Description
`Get Platform Preferences( Platform( Option ) )`	Returns the current option values for the specified platform.
`Get Platform Preferences( <<Changed )`	Returns the current option values that have changed for all platforms.
`Get Platform Preferences( Platform( <<Changed ) )`	Returns the current option values that have changed for all platform preferences.
`Get Platform Preferences( Platform( Option ( <<Changed ) ) )`	Returns the current option values that have changed for the specified platform.

Examples

Suppose that the user modified several platform preferences through the JMP Platforms window or a script.

```
Platform Preferences(
  Distribution( Set Bin Width( 2 ), Horizontal Layout( 1 ) ),
  Model Dialog( Keep Dialog Open( 1 ) ),
  Graph Builder( Legend Position( "Bottom" ) )
);
```

To return all of the modified platform preferences, use `Get Platform Preferences( <<Changed )`:

```
Get Platform Preferences( <<Changed );
  Platform Preferences(
    Distribution( Horizontal Layout( 1 ), Set Bin Width( 2, <<On ) ),
    Graph Builder( Legend Position( "Bottom", <<On ) ),
    Model Dialog( Keep dialog open( 1 ) )
  );
```

Get Preferences(<preference_name>)

Get Preference(<preference_name>)

Description

Returns the settings for the specified preferences.

Returns

A list of preference settings.

Argument

`preference_name` (Optional) If no preference is specified, all preferences are returned. Otherwise, the settings for the specified preference are returned.

Notes

The preferences for the following areas are not accessible in JSL: Text Data Files, Windows Specific, Mac OS Settings, Fonts, Communications, Script Editor, and JMP Updates. For more information about getting platform preferences, see "Get Platform Preferences(<platform <(option, ...)> ... >)".

`Glue(expr1, expr2, ...)`

`expr1; expr2`

Description

Evaluates each argument in turn.

Returns

The result of the last argument evaluated.

Arguments

One or more valid JSL expressions.

Notes

A semicolon placed between expressions is the more commonly used form for `Glue()`.

`Gzip Compress(blob)`

Description

Compresses a blog of data into a gzip blob.

Example

```
Gzip Compress(
   Char To BLOB(
      "random data does not usually compress well and may get larger"
   )
);
   Char To BLOB(

      "~1F~8B~08~00~00~00~00~00~00~0A~0D~CA~C1~0D~00~21~08~04~C0V~B6~B5~CDA~FC
      ~80~5C~00c~EC^~E7=~C9)~E1~106~21~A1~85~19~8DU~8Bf~07_~F8~9FZ~85~ADfx~13~
      CE~83~A1~0Dc~0E~CD~0B~94*~16~1E=~00~00~00",
      "ascii~hex"
```

`Gzip Uncompress(blob)`

Description

Uncompresses a blob of gzip data into a blob.

Example

```
Gzip Uncompress( /*typically this data might come from Gzip Compress() but
  might also come from a .gz file using Load Text File() with the blob
  option*/
  Char To BLOB(

  "~1F~8B~08~00~00~00~00~00~00~0A~0D~CA~C1~0D~00~21~08~04~C0V~B6~B5~CDA~FC~8
  0~5C~00c~EC^~E7=~C9)~E1~106~21~A1~85~19~8DU~8Bf~07_~F8~9FZ~85~ADfx~13~CE~8
  3~A1~0Dc~0E~CD~0B~94*~16~1E=~00~00~00",
     "ascii~hex"
  )
);
  Char To BLOB(
     "random data does not usually compress well and may get larger",
     "ascii~hex"
  )
```

Host Is("argument")

Description

Determines whether the host environment is the specified OS.

Returns

True (1) if the current host environment matches the argument, false (0) otherwise.

Argument

`Argument Windows"` or "`Mac`" tests for the specified operating system.

Is Alt Key()

Description

Returns 1 if the Alt key is being pressed, or 0 otherwise.

Notes

On a macOS, `Is Alt Key()` tests for the Option key.

Is Command Key()

Description

Returns 1 if the Command key is being pressed, or 0 otherwise.

Is Context Key()

Description

Returns 1 if the Context key is being pressed, or 0 otherwise.

Is Control Key()

Description

Returns 1 if the Control key is being pressed, or 0 otherwise.

Notes

On a macOS, `Is Control Key()` tests for the Command key.

Is Option Key()

Description

Returns 1 if the Option key is being pressed, or 0 otherwise.

Is Shift Key()

Description

Returns 1 if the Shift key is being pressed, or 0 otherwise.

JMP Product Name()

Description

Returns either `"Standard"`, `"Pro"`, or `"Student"`, depending on which version of JMP is licensed.

JMP Version()

Description

Returns the version number of JMP that you are running.

Returns

release.revision<.fix>

Arguments

none

Load DLL("path" <,AutoDeclare(Boolean | Quiet | Verbose)>)

Load DLL("path" <, Quiet | Verbose>)

Description

Loads the DLL in the specified *path*.

Arguments

`path` A pathname that specifies where to load the DLL.

`AutoDeclare(Boolean | Quiet | Verbose)` Optional argument. `AutoDeclare(1)` and `AutoDeclare(Verbose)` write verbose messages to the log. `AutoDeclare(Quiet)`

turns off log window messages. If you omit this option, verbose messages are written to the log.

`Quiet | Verbose` Optional argument. When you use `Declare Function()`, this option turns off log window messaging (`Quiet`) or turns on log window messaging (`Verbose`).

See Also

Once a DLL is loaded, you send the DLL object messages to interact with it. See "Dynamic Link Libraries (DLLs)" for more information about these messages.

`Mail("address"|"addresses", "subject", "message", <"attachment filepath" | {"attachment 1 filepath", "attachment 2 filepath", ...}>)`

Description

(Windows) Sends e-mail (using MAPI) to the *`address`* with the specified *`subject`* and *`message`* texts. Sends one or more attachments specified by the optional *`attachment`* argument. The attachment argument can evaluate to a quoted string or list of strings.

(macOS) Creates an e-mail in the user's Mail application. The user must click **Send** in the e-mail. In Microsoft Outlook, you must manually add attachments to the e-mail.

Examples

To send an email with multiple attachments on Windows:

```
Mail(
   "yourname@company.com",
   "New data and script",
   "Today's updated data table and script are attached.",
   {"$DOCUMENTS/wd.jsl", "$DOCUMENTS/survey.jmp"}
);
```

or:

```
list = {"$DOCUMENTS/wd.jsl", "$DOCUMENTS/survey.jmp"};
Mail(
   "yourname@company.com",
   "New data and script",
   "Today's updated data table and script are attached.",
   list
);
```

To send an email to multiple recipients:

```
Mail(
   {"hername@company.com", "hisname@company.com"},
   "Database updates",
   "Today's sales database contains the numbers from last month."
);
```

Notes

On macOS, `Mail()` works on Yosemite and later operating systems.

Main Menu(*string*, <*string*>)

Description

Execute the command found on JMP's menu named by the quoted *string*.

Arguments

`string` The internal path name as shown in the menu editor for items. For example, "NEW" is the internal name for the New subcommand in the File menu.

`string` (Optional) The name of the window to send the command to.

Examples

`Main Menu()` accepts either a full path or a partial path. If a partial name is used, and there are other menu items with the same name, the first menu item found is executed. JMP searches the top-level menu (File, Tables, DOE, and so on) first for the partial name and then searches inside each of those menus in order.

```
Main Menu( "File:New:Data Table" ); // full path
Main Menu( "Data Table" ); // partial path
```

Minus(a)

-a

Description

Reverses the sign of *a*.

Returns

-*a* if *a* is positive (`a=3; -a=-3; Minus(a)=-3`).

a if *a* is negative (`a=-3; -a=3; Minus(a)=3`).

0 if *a* is 0 (`a=0; -a=0; Minus(a)=0`).

Missing if *a* is missing.

Argument

`a` Can be variable or a number. A variable must contain a number or a matrix.

Multiple File Import(arguments)

Description

Imports one or more files into a data table. You can create this JSL by selecting Save Script to Script Window from the Multiple File Import window.

Returns

Creates a Multiple File Import Object. The object accepts messages to set a folder, filter files, and specify import options.

Arguments

`<<Set Folder` Specifies the folder that contains the files you want to import.

<<Set Name Filter (Optional) Specifies the file name or extension of the files. The name filter uses * to represent zero-or-more characters and ? to represent exactly one character. * and ? also match a period. The default setting is *.*, or all files. Because you can specify files using a semicolon-delimited list of filters, file names that include semicolons or | must be specified using ? or *.

<<Set Name Enable(Boolean) Enables the name filter. The setting is off by default.

<<Set Size Filter (Optional) Filters the file list by file size. Specify the sizes by kB (kilobytes, or 1000 bytes) in a list. The default values are based on the size range of the files in the file list.

<<Set Size Enable(Boolean) (Optional) Enables the size filter. The default setting is off.

<<Set Date Filter (Optional) Filters the file list by date and time. Specify the date and time in a list in seconds. The default values are based on the date and time range of the files in the file list.

<<Set Date Enable(Boolean) (Optional) Enables the date filter. The default setting is off.

<<Set Add File Name Column(Boolean) (Optional) Includes a column that contains the imported file name. The default setting is off.

<<Set Add File Size Column(Boolean) (Optional) Includes a column that contains the size of the imported file. The default setting is off.

<<Set Add File Date(Boolean) (Optional) Includes a column that contains the time and date stamp of the imported file. The default setting is off.

<<Set Import Mode(Row Per File|Row Per Line|CSV Data) (Optional) Specifies the format of the file that is imported: whole file on one row, one line on one row, and CSV. CSV Data is the default setting.

<<Set Charset(Best Guess|utf-8|utf-16|us~ascii|windows-1252|x-mac-roman|x-mac-japanese|shift-jis|euc-jp|utf-16be|gb2312) (Optional) The character set in the imported file. The character set specified in the General preferences (Open Text File Charset) is set by default.

<<Set Stack Mode(Stack Similar|TablePerFile) (Optional) Specifies how the files are combined. Stack Similar is the default setting. (When JMP detects that the files have the same columns, the files are concatenated into a single data table.)

<<Set CSV Has Headers(Boolean) (Optional) Specifies whether the CSV file contains a header row. The setting is on by default.

<<Set CSV Allow Numeric(Boolean) (Optional) Sets the data type to numeric. The setting is on by default.

<<Set CSV First Header Line(*n*) (Optional) Specifies the header row number. 1 is the default setting.

<<Set CSV Number of Header Lines(*n*) (Optional) Specifies the number of header rows. 1 is the default setting.

<<Set CSV First Data Line(n) (Optional) Specifies the first line that contains data. 2 is the default setting.

<<Set CSV EOF Comma(Boolean) (Optional) Specifies a comma delimiter. The setting is on by default.

<<Set CSV EOF Tab(Boolean) (Optional) Specifies a tab delimiter.

<<Set CSV EOF Space(Boolean) (Optional) Specifies a space delimiter.

<<Set CSV EOF Spaces(Boolean) (Optional) Specifies spaces as the delimiter.

<<Set CSV EOF Other("") (Optional) Specifies a custom delimiter.

<<Set CSV EOF CRLF(Boolean) (Optional) Specifies carriage return and line feed end-of-line characters. The setting is on by default.

<<Set CSV EOF CR(Boolean) (Optional) Specifies a carriage return end-of-line character. The setting is on by default.

<<Set CSV EOF LF(Boolean) (Optional) Specifies a line feed end-of-line character.

<<Set CSV Semicolon(Boolean) (Optional) Specifies a semicolon end-of-line character. The setting is off by default.

<<Set CSV EOL Other("") (Optional) Specifies a custom end-of-line character.

<<Set CSV Quote("") (Optional) Specifies the character used as a quote. The default setting is \!", a double quotation mark.

<<Set CSV Escape("") (Optional) Specifies the escape sequence such as a backlash instead of doubling the quotation mark.

<<Import Data Imports the data.

Example

```
mfi = Multiple File Import(
  <<Set Folder( "$SAMPLE_IMPORT_DATA" ),
  <<Set Name Filter( "UN*.csv" ), // import files with this name
  <<Set Name Enable( 1 ) // display the file name in a column
)
<<Import Data();
```

Multiply(a, b, ...)

a*b*...

Description

Multiplies all values. No arguments are changed.

Returns

The product.

Arguments

Any number of variables, numbers, or matrices.

Notes

Any number of arguments is permitted. If no argument is specified, Multiply() returns 1.

Name(*string*)

Description

A name is something to call an item.

- If the name begins with an alphabetic character or underscore, and continues with alphanumeric characters, whitespace, Unicode mathematical symbols and certain punctuation (apostrophes, percentage signs, periods, backslashes, and underscores, then you can use the name directly in scripts.
- You can use names that do not follow these rules with the Name() keyword.

Examples

("name"n) is preferred. Name(*string*) is deprecated.

```
"my-var-name"n = "hello";
Length( "my-var-name"n )
   5
```

New OAuth2 Token(user(yourgoogleaccount@gmail.com), client ID(*string*), client secret(*string*), refresh token(*string*), token URL(*string*))

Description

Creates an OAuth2 token for securely accessing data across different web APIs.

Arguments (Required)

`user` The user name, email, or personal identifier for the account being accessed.

`client ID` The public identifier that acts as an API key.

`client secret` The private identifier that corresponds to Client ID.

`refresh token` A token used to get access tokens.

`token URL` The URL that access tokens are received from. Unique to every service and accessible through their API or OAuth page.

Arguments (Authorization Code Grant)

`redirect URL( "https://app.getpostman.com/oauth2/callback" )` The URL that an access code is sent back to. Unless your company or the service provides one, we recommended that you create a free Postman account and use this redirect.

`client secret( "1aB893cdDeFf2D" )` The private identifier that corresponds to Client ID.

`request auth( ... )` Extra parameters indicating that you'll use an Authorization Code flow. Requires `Auth URL()`. Some services require scope. You can add custom fields with Fields.

`scope( "spreadsheets email docs" )` A space-separated list of scopes. Unique to every service, and accessible through their API or OAuth page. Only usable in `Request Auth()`.

`auth URL( "https://www.example.com/oauth2/v1/authorize" )` The URL for requesting authorization. Unique to every service, and accessible through their API or OAuth page. Only usable in `Request Auth()`.

Arguments (Implicit Grant)

`redirect URL( "https://app.getpostman.com/oauth2/callback" )` The URL that an access code is sent back to. Unless your company or the service provides one, it's recommended to create a free Postman account and use this redirect.

Arguments (Resource Owner)

`password( "wordpass123" )` The password that corresponds to the username.

`client secret( "1aB893cdDeFf2D" )` The private identifier that corresponds to Client ID.

Arguments (Custom Data)

`fields( fields )` Custom fields that are equivalent to HTTP Request's Form(fields(fields)). Can be specified both in `New OAuth2 Token()` and in `Request Auth()`. Only necessary if the service requires information that is not defined in the OAuth 2.0 standard.

`headers( headers )` Custom headers that are equivalent to `HTTP Request's Headers( headers )`. Can only be specified in `New OAuth2 Token()`. Only necessary if the service requires information not defined in the OAuth 2.0 standard.

Example

```
token = New OAuth 2 Token (
  User( "yourgoogleaccount@gmail.com" ),
  Refresh Token( "1a2b3c4e5F" ),
  Token URL( "https://www.example.com/oauth2/token" ),
  Client ID( "12ab" ),
  Client Secret( "3456dEfG" )
);
```

See Also

See your API documentation for more information about how to get values such as the client secret and token URL.

Open Datafeed()

Datafeed()

Description

Creates a Datafeed object and window.

Returns

A reference to the Datafeed object.

Arguments

No arguments are required. You usually set up the basic operation of the data feed within the `Open Datafeed()` command, however.

`Open Help("Help"|"Statistics Index"|"Scripting Index", ...)`

Description

Opens the specified help window.

`Parse XML(`*`string`*`, On Element("tagname", Start Tag(expr), End Tag(expr), Text))`

Description

Parses an XML expression using the `On Element` expressions for the specified XML tags.

Example

```
XMLData = "
<Book name='Foods'>All you want to know
       <Chapter num='1'>Fruit
             <kind>Apple</kind>
             <kind>Cherry</kind>
             <ps>I love dessert!</ps>
       </Chapter>
       <Chapter num='2'>Bread
             <kind>Wheat</kind>
             <kind>Corn</kind>
             <ps>I love sandwiches!</ps>
       </Chapter>
       <Chapter num='3'>Veggy
             <kind>Squash</kind>
             <kind>Cabbage</kind>
             <ps>I love anything else!</ps>
       </Chapter>
        and more.
</Book> ";

// variables are initialized so text can be concatenated
title = "";
subtitle = "";
chap = "";
chapnum = "";
ps = "";

Parse XML( XMLData,
      On Element( "Book",
            // capture the name attribute during the start of the Book
            Start Tag( title = XML Attr( "name" ) ),
           /* this book has split the subtitle and needs to join the text;
              the joined text will be used by endTag.
              Text(...) supplies the JSL.*/
```

```
                Text( subtitle = subtitle || " -- " || Trim( XML Text() ) ),
                  /* after endTag processes the variables, set them back
                  to their initial state, just in case there is a second book
                  to process in the same XML. */
                endTag( Write( "\!n", title, " ", subtitle ); title = ""; subtitle
        = ""; )
            ),
            On Element( "Chapter",
                  // capture the chapter number during the start of the Chapter
                  Start Tag( chapnum = XML Attr( "num" ) ),
                /* the chapter text is joined together, newlines
                  and extra space is trimmed, and a single space is used to
                  separate the separated texts. The <kind> tag is ignored by
                  this ParseXML specification. The <kind> text is processed
                  by this Text(...) because it wasn't consumed by any other
                  On Element. */
                  Text( chap = chap || Trim( XML Text() ) || " " ),
                /* after endTag processes the variables, set them back to
                  their initial state, because there is another chapter
                  that needs to start with a clean slate.*/
                endTag( Write( "\!n", chapnum, " ", chap, " ps: ", ps ); chapnum =
        ""; chap = ""; ps = ""; )
            ),
            On Element( "ps", End Tag( ps = XML Text() ) )
    );
      1 Fruit Apple Cherry  ps: I love dessert!
      2 Bread Wheat Corn  ps: I love sandwiches!
      3 Veggy Squash Cabbage  ps: I love anything else!
      Foods  -- All you want to know -- and more.
```

`Platform Preferences(platform(option(value)), ...)`

`Platform Preference(platform(option(value)), ...)`

`Set Platform Preferences(platform(option(value)), ...)`

`Set Platform Preference(platform(option(value)), ...)`

Description

Sets and resets values for platform options and turns the options on and off.

Arguments

`platform` Specifies the platform of the preference.

`option` Specifies the preference name.

`value` Specifies the preference value.

Notes

Table 2.4 describes the syntax for setting platform preferences.

Table 2.4 Platform Preferences() Syntax

Syntax	Description

`Platform Preferences( <<Default )`
`Platform Preferences( <<Factory Default )`
`Platform Preferences( Default )`
	Resets all platform preferences to their default values.

`Platform Preferences( Platform( <<Default ) )`
`Platform Preferences( Platform( <<Factory Default ) )`
`Platform Preferences( Platform( Default ) )`
	Resets the specified platform preferences to their default values.

`Platform Preferences( Platform( option ( <<Default ) ) )` `Platform Preferences( Platform( option ( <<Factory Default ) ) )`	Resets the specified platform option to its default value.
`Platform Preferences( Platform( option( value, <<On ) ) )`	Sets the value of the specified platform option and turns it on.
`Platform Preferences( Platform( option( value, <<Off ) ) )`	Sets the value of the specified platform option and turns it off.

Example

The following expression selects (or turns on) Set Bin Width in the Distribution platform preferences and sets the value to 2:

```
Platform Preferences( Distribution( Set Bin Width( 2 ) ) );
```

The following expression changes the Set Bin Width value and turns the option off:

```
Platform Preferences( Distribution( Set Bin Width( 2, <<Off ) ) );
```

The following expression resets the default Set Bin Width value and deselects the preference:

```
Platform Preferences( Distribution( Set Bin Width( <<Default ) ) );
```

Polytope Uniform Random(samples, A, b, L, U, neq, nle, nge, <nwarm=200>, <nstride=25>)

Description

Generates random uniform points over a convex polytope.

Arguments

`Samples` The number of random points to be generated.

`A` The constraint coefficient matrix.

`B` The right hand side values of constraints.

`L, U` The lower and upper bounds for the variables.

`neq` The number of equality constraints.

`nle` The number of less than or equal inequalities.

`nge` The number of greater than or equal inequalities.

`nwarm` (Optional) The number of warm-up repetitions before points are written to the output matrix.

`nstride` (Optional) The number of repetitions between each point that is written to the output matrix.

Notes

The constraints must be listed as equalities first, less than or equal inequalities next, and greater than or equal inequalities last.

```
Preferences(pref1(value1), ...)
Preference(pref1(value1), ...)
Pref(pref1(value1), ...)
Prefs(pref1(value1), ...)
Set Preferences(pref1(value1), ...)
Set Preference(pref1(value1), ...)
```

Description

Sets preferences for JMP.

Arguments

`Add Files Opened by Scripts to the Recent Files List(Boolean)` Determines whether a file that is opened by a script is added to the Home Window's Recent Files list.

`Analysis Destination(window)` Specifies where to route new analyses.

`Annotation Font("font", size, "style")` Font choice for annotations in reports.

`Axis Font("font", size, "style")` Font choice for axis labels.

`Axis Title Font("font", size, "style")` Font choice for axis titles.

`Background Color( {R, G, B} | <color> )` Sets the background color for windows.

`Calculator Boxing(Boolean)` Turns on boxing to show hierarchy of expressions.

`Conditional Formatting Rules` Creates rules for conditionally formatting text in reports. See "Examples".

`Data Table Font("font", size, "style")` Font choice for data tables.

`Data Table Title on Output(Boolean)` Titles reports with name of data table.

`Date Title on Output(Boolean)` Titles reports with current date.

`Evaluate OnOpen Scripts("always"|"never"|"prompt")` Determines whether an `On Open` table script is run after the user opens the data table. By default, the user is prompted. Their choice is remembered each time they open the data table in the current JMP session. Scripts that execute other programs are never run.

`Excel Has Labels(Boolean)` When on, forces JMP to interpret the first row of data as column headings.

`Excel Selection(Boolean)` When on, the user is prompted for which non-blank Excel worksheets should be imported from an Excel workbook.

`File Location Settings(<Directory Type>("<path>"<,"initial directory">))` Valid directory types are:

`Data Files Directory` Sets the default location for data files.

`Help Files Directory` Sets the default location for help files.

`Installation Directory` By default, this location is set to the JMP installation folder on Windows:

"C:/Program Files/SAS/JMP/16" or "C:/Program Files/SAS/JMPPro/16"

`License File Path` Sets the default location for JMP license file.

`Preferences File Directory` Sets the default location for the preferences settings file.

`Save As Directory` Sets the default location for Save As file operations.

`Foreground Color(color)` Sets the foreground color for windows.

`Formula Font("font", size, "style")` Font choice for the formula editor.

`Graph Background Color(color)` Sets the color for the background area inside the graph frame.

`Graph Marker Size(size)` Default size for drawing markers.

`Heading Font("font", size, "style")` Font choice for table column headings in reports.

`Initial JMP Starter Window(Boolean)` Specifies whether the JMP Starter window is shown at launch.

`Initial Splash Window(Boolean)` Enables you to show or suppress the initial splash screen.

`Maximum JMP Call Depth(size)` Sets the default for the maximum call depth (or stack size) for JMP in which JSL built-in functions, user-defined functions, or `Recurse()` function calls can be made. By default, the maximum call depth is set to 256KB.

Each thread that JMP creates has a 2MB stack by default. Increasing the maximum call depth can cause a physical runtime stack overflow, so incrementally increase this preference in small amounts until you find the best value that works for your JSL script.

`Marker Font("font", size, "style")` Font choice for markers used in plots.

`Monospaced Font("font", size, "style")` Font choice for monospaced text.

`ODBC Suppress Internal Quoting(Boolean)` Prevents internal quoting in SQL statements that contain table and variable names with mixed case and spaces.

`Outline Connecting Lines(Boolean)` Draws lines between titles for same-level outline nodes.

`Print Settings(option(value), ... )` Changes print options on the Page Setup window:

`Margins( <n>, <n>, <n>, <n>)` sets the left, top, right, and bottom margins. Margins are in inches.

`Margins(<n>)` sets all margins to the same value in inches.

`Orientation("portrait" | "landscape")` changes the page's print orientation.

`Headers(<"char">, <"char">, <"char">)` specifies text that appears in the left, middle, and right header.

`Headers(<"char">)` specifies the only text in the header.

`Footers(<"char">, <"char">, <"char">)` specifies text that appears in the left, middle, and right footer.

`Footers(<"char">)` specifies the only text in the footer.

`Scale(<n>)` decreases or increases the percentage at which the content prints.

`Show Explanations(Boolean)` Some analyses have optional text that explains the output.

`Show Menu Tips(Boolean)` Turns menu tips on or off.

`Show Status Bar(Boolean)` Turns display of the status bar on or off.

`Small Font("font", size, "style")` Font choice for small text.

`Text Font("font", size, "style")` Font choice for general text in reports.

`Thin Postscript Lines(Boolean)` macOS only. Specifies that line widths drawn to a Postscript printer be narrower than otherwise.

`Title Font("font", size, "style")` Font choice for titles. Arguments are name of font (for example, `"Times"`), size in points, and style (`"bold"`, `"plain"`, `"underline"`, `"italic"`).

`Use Triple-S Labels as Headings(Boolean)` When on, this argument forces JMP to interpret label names as column headings. Example: `Pref(Name("Use Triple-S Labels as Headings")(0) )`; turns off the preference.

Examples

The following expressions reset all preferences to their default values.

```
Preferences( "Default" );
Preferences( "Factory Default" );
```

The following script creates conditions for formatting text in reports.

```
Preferences(
    Conditional Formatting Rules(
```

```
            RuleSet(
               RuleName( "Warning" ),

               // if the value is not equal to 0, format the text as 80% gray
               NotEqualTo( Value( 0 ), Format( TextAlpha( 0.8 ) ) )
             )
        )
    );
```

Notes

The preferences for the following areas are not accessible in JSL: Text Data Files, Windows Specific, Mac OS Settings, Fonts, Communications, Script Editor, and JMP Updates.

See Also

"Platform Preferences(platform(option(value)), ...)"`Register Addin("unique_id", "home_folder", <named_arguments>)`

Description

Register a JMP Add-In and load the add-in if it registers successfully.

Returns

If successful, returns a scriptable object representing the registered add-in. If unsuccessful, returns Empty.

Arguments

`unique_id` A quoted string that contains the unique identifier for the add-in. The string can contain up to 64 characters. The string must begin with a letter and contain only letters, numbers, periods, and underscores. Reverse-DNS names are recommended to increase the likelihood of uniqueness.

`home_folder` A quoted string that contains the filepath for the folder containing the add-in files. The filepath must conform to the valid pathname requirement for the host operating system.

`DisplayName( "name" )` An optional, quoted string that contains a name that can be displayed in the JMP user interface wherever add-in names are displayed, instead of the unique ID.

`JMPVersion("version")` An optional string that contains a specific version of JMP. The default value is "All", which enables the add-in to be loaded and run in any version of JMP that supports add-ins. "Current" restricts the use of the add-in to only the current version. Any quoted version number (for example, "7" or "9") restricts the add-in to a single specific version of JMP.

`LoadsAtStartup(Boolean)` An optional Boolean. The default value is True (1), which causes the add-in to be loaded when JMP is started. If the value is False (0), the add-in is not loaded automatically.

`LoadNow(Boolean)` Loads the add-in immediately.

Notes

If a file named addin.def is found in the specified home folder, values from that file are used for any optional arguments that are not included in the `Register Addin()` function.

Example

In the following example, the first argument is the unique identifier. The second argument identifies where the add-in is installed. The third argument is the name that appears where add-in names are displayed (for example, the **View > Add-Ins** menu on Windows).

```
Register Addin("com.company.lee.dan.MyAddIn","$DOCUMENTS/myaddin",
  displayname( "Calculator Addin" ));
```

The second argument becomes the `$ADDIN_HOME` path variable definition. When you refer to the add-in scripts, be sure to include a trailing slash after the path variable.

```
Include("$ADDIN_HOME(com.jmp.jperk.texttocols)/texttocols.jsl");
```

Revert Menu()

Description

Resets your JMP menus to factory defaults.

Run Program(Executable("path/filename.exe"), Options({"/a", "/b", "..."}), Read Function(expression), Write Function(expression), Parameter(expression))

Description

Runs the external program specified by the `Executable` argument, with the command line arguments specified by the `Options` argument.

Results

Returns either a quoted string, a blob, or a `Run Program` object as controlled by the `Read Function` argument.

Arguments

`Executable` The path to the executable. On macOS, type the full path to the executable.

`Options` Command line arguments for the executable.

`Read Function` If `Read Function( "text" )` is specified, a quoted string is returned. If `Read Function( "blob" )` is specified, a blob is returned. The script waits until the external program closes its stdout. `Run Program` then returns all data that the external program has written to its stdout as a quoted string or a blob.

If `Read Function` is not specified, a `Run Program` object is returned.

`Write Function` Optional argument that accepts a function as its value; it does not accept `"text"` or `"blob"`.

`Parameter` Optional argument to read and write the expression in `Read Function`.

Notes:

- Use global variables when `Run Program()` is inside a function.
- The `Run Program` object, which is returned if `Read Function` is not specified, accepts the following messages to read data from the external program's stdout:
  - `<<Read`: reads any available data as a quoted string. If no data is available, an empty string is returned.
  - `<<Can Read`: returns `true` if there is data available to be read.
  - `<<Is ReadEOF`: returns `true` when the external program has completed and all its data has been read.

  You can use these messages to poll for data and process the data as it is produced by the external program.
- A `Run Program` object accepts the following messages to write data to the external program's `stdin`:
  - `<<Write( "text" )`: sends data to the external program's stdin.
  - `<<Can Write`: returns `true` if the external program will accept data immediately; otherwise, calling `<<Write` causes your script to block.
  - `<<WriteEOF`: signals to the external program that you are done sending data to it.
- Instead of sending messages to the returned `Run Program` object, you can specify the `Read Function` argument as an inline function. `RP` is the `Run Program` object.

```
RP = Run Program(
   Executable( ... ),
   Read Function(
      Function( {RP},
          <your code here>
         RP << Read
      )
   )
);
```

The `Parameter(optParm)` argument is optional in `Read Function`. If specified, the functions defined for `Read Function` and `Write Function` can receive a second argument, which is the value of `optParm`.

Examples

The following script is an example of the `Write Function` argument. `RP` is the `Run Program` object. In this context, it accepts the `<<Write` and `<<WriteEOF` messages.

```
RP = Run Program(
   Executable( ... ),
   Write Function(
      Function( {RP},
         <your code here>
```

```
            RP << Write( "Program finished." )
        )
    )
);
```

The following script shows an example of `Parameter(optParm)` argument:

```
RP = Run Program(
   Executable( ... ),
   Parameter( x ),
   Read Function( Function( {RP, optParm},... ) )
);
```

Within the `Read Function`, `optParm` contains the value of *x*. Do not attempt to access the `optParm` argument in your function if you have not specified a `Parameter` argument.

Schedule(n, script)

Description

Queues an event to run the *script* after *n* seconds.

Set Clipboard(*string*)

Description

Evaluates the quoted *string* argument looking for a character result, and then places the string on the clipboard.

Example

```
Set Clipboard( "copy me" );
```

SetJVMOption(Version("<version number>"))

Description

Sets the Java Runtime Environment (JRE) version that you want JMP to use (rather than the version installed with JMP). This script must be run before JMP connects to the JRE.

Argument

`version` (Windows only) In the Windows registry, there are two requirements for the JavaSoft/Java Runtime Environment key: the key must include a quoted string called "RuntimeLib" that points to a valid jvm.dll. And the Java Runtime Environment key must include a key named after the quoted JVM version number.

Set Platform Preference()

Set Platform Preferences()

See "Platform Preferences(platform(option(value)), ...)".

`Set Preference()`

`Set Preferences()`

See "Preferences(pref1(value1), ...)".

`Set Toolbar Visibility( "toolbar name" | default | all, window type | all, "true" | "false" )`

Description

On Windows, shows or hides a toolbar based on the window type or for all windows.

Arguments

`toolbar name | default | all` The internal name of the toolbar (see the **View > Toolbars** list in JMP), the default toolbar for the specified window type, or all toolbars. Include quotes around `"toolbar name"`.

`window type | all` Data table, script, report, journal, or all windows.

`true | false` A quoted string that shows or hides the toolbar.

`Shortest Edit Script( A, B )`

`Shortest Edit Script( strings( A, B, <matrix( 0|1 )>, <limit( number )> ) )`

`Shortest Edit Script( lines( A, B, <matrix( 0|1 )>, <limit( number )>, <separators( characters )>, <ignore( characters )|ignore white space( )> ) )`

`Shortest Edit Script( sequences( nA, nB, Function( {iA, iB}, adata[iA] == bdata[ib] ) ) )`

Description

Compares two quoted strings, lines, or sequences.

Returns

Returns a list or a matrix of edit commands. The simplest form returns a list. quoted `strings` and `lines` return a matrix (if set to 1) or a list. `sequences` returns a matrix.

There are three possible commands: common data in both quoted strings, delete data from the first string, and keep data from the second string.

Optional Arguments

`matrix` Indicates whether the returned value is a matrix.

`limit` Stops the evaluation when the edit list exceeds the specified number of inserted or deleted items. Two quoted random strings have a lot of common characters in a lot of distinct sections. The function runs for a long time trying to find a best match. `limit` stops the function sooner.

Optional Lines Arguments

`matrix` Indicates whether the returned value is a matrix.

`limit` Stops the evaluation when the edit list exceeds the specified number of inserted or deleted items. Two quoted random strings have a lot of common characters in a lot of distinct sections. The function runs for a long time trying to find a best match. `limit` stops the function sooner.

`separators` A character that separates words.

`ignore` Ignores the specified spaces or characters in a line.

`ignore white space` Ignores white space in a line.

Optional Sequences Argument

`Function` A user-defined function.

Examples

The following example compares two quoted strings with three common sequences of characters between them.

```
Shortest Edit Script( "abcdef", "abdezgh" );
```
```
{{"Common", "ab"}, {"Remove", "c"}, {"Common", "de"}, {"Insert", "zgh"},
   {"Remove", "f"}}
```

The following example examines each line in quoted string aa and bb:

```
aa = "this is
a test of
shortest
edit script
lines with several words";

bb = "this is
a test 2 of
shortest
edit ?., script
lineswithseveral words";

Shortest Edit Script( lines( aa, bb, separators( "\!n" ),

  // quote and newline separators
  ignore( "?., " ) ) ); // ignore these characters and spaces
{{"Common", "this is // lines in aa and bb contain "this is"
"}, {"Remove", "a test of // only on line 2 of aa
"}, {"Insert", "a test 2 of // only on line 2 of bb
"},
{"Common", "shortest
edit script
lines with several words"}}
// lines in aa and bb contain "shortest", "edit script", and "lines with
   several words"
```

Show Addin Builder Dialog()

Description

Opens a window in which you can make custom add-ins.

Show Addins Dialog()

Description

Opens the Add-In Status window (**View > Add-Ins**).

Arguments

None.

Show Commands()

Description

Lists scriptable objects and operators. Arguments are All, DisplayBoxes, Scriptables, Scriptable Objects, StatTerms, Translations.

Show Preferences(<"all">)

Description

Shows current preferences. If no argument is specified, preferences that have been changed are shown. If "all" is given as the argument, all preferences are shown.

Show Properties(object)

Description

Shows the messages that the given *object* can interpret, along with some basic syntax information.

Sobol Quasi Random Sequence(nDim, nRow)

Description

Generates a sequence of space-filling quasi random numbers using the Sobol sequence in up to 4000 dimensions.

Socket(<STREAM | DGRAM>)

Description

Creates a socket.

Returns

The socket that was created.

Arguments

STREAM | DGRAM Optional argument to specify whether the socket is a stream or datagram socket. If no argument is supplied, a stream socket is created.

`Speak(text, <wait(Boolean)>)`

Description

Calls system's speech facilities to read aloud the text. If Wait is turned on, script execution pauses until speaking is done.

`Status Msg("message")`

Description

Writes the quoted message string to the status bar.

`Subtract(a, b)`

`a-b-...`

Description

Subtracts the values of the listed arguments, left to right. No arguments are changed.

Returns

The difference.

Arguments

Two or more variables, numbers, or matrices.

Notes

Two or more arguments are permitted.

`Unregister Addin("unique_id")`

Description

Unregisters (removes) a previously registered add-in.

Argument

`unique_id` A quoted string that contains the unique identifier for the add-in to unregister.

`Web(`*`string`*`, <JMP Window>)`

Description

Opens the URL stored in the quoted *`string`* in the default web browser.

The `http://` prefix in the URL is optional.

Examples

```
url = "www.jmp.com"; // open the URL in the default web browser
Web( url );

Web( "www.jmp.com" ); // open the URL in the default web browser

Web( "www.jmp.com", JMP Window ); // open the URL in the JMP browser window
```

XML Attr("attr name")

Description

Extracts the quoted string value of an XML argument in the context of evaluating a `Parse XML` command.

XML Decode("xml")

Description

Decodes symbols in XML to ordinary text. For example, `&` becomes &, and `<` becomes <.

Argument

`xml` A quoted string containing XML.

XML Encode("text")

Description

Prepares text for embedding in XML. For example, & becomes `&`, and < becomes `<`.

Argument

`xml` A quoted string containing plain text.

XML Text()

Description

Extracts the quoted string text of the body of an XML tag in the context of evaluating a `Parse XML` command.

Chapter 3

JSL Messages

Summary of Messages for Objects and Display Boxes

This topic provides abbreviated descriptions for many of JMP's general object messages. For complete information about object messages, see the JMP Scripting Index. In JMP, select **Help > Scripting Index**.

For information about platform messages, see *Scripting Guide*.

Contents

Alpha Shape . . . 380
Associative Arrays . . . 380
Classes . . . 381
Data Tables . . . 383
Columns . . . 412
Rows . . . 420
Data Filter . . . 421
Data Feed (Windows Only) . . . 424
Display Boxes . . . 427
All Display Boxes . . . 427
Axis Boxes . . . 437
Border Boxes . . . 442
Data Browser Boxes . . . 443
Data Filter Source Boxes . . . 443
Frame Boxes . . . 443
Display 3D Boxes . . . 445
Excerpt Boxes . . . 445
Filter Col Selector . . . 446
Global Boxes . . . 446
Hier Boxes . . . 446
Matrix Boxes . . . 446
Nom Axis Boxes . . . 447
Number Col Boxes . . . 447
Number Col Edit Boxes . . . 450
Number Edit Box . . . 451
Outline Boxes . . . 451
Panel Boxes . . . 452
Plot Col Boxes . . . 452
Slider Boxes and Range Slider Boxes . . . 453
String Col Boxes . . . 454
Tab Boxes . . . 455
Table Boxes . . . 456
Text Boxes . . . 459
Tree Node and Tree Box . . . 461
Triangulation . . . 463
Windows . . . 464
Dynamic Link Libraries (DLLs) . . . 467
HTML 5 . . . 469
Web Report . . . 469

Images 470
JMP Applications 473
JMP App 473
JMP App Module 475
JMP App Module Instance 475
MATLAB 476
Namespaces 479
Platforms 481
Bubble Plot 487
DOE 487
Partition 488
Response Screening 488
Tabulate 489
Python Integration Messages 490
R Integration Messages 493
SAS Integration Messages 496
Metadata Server Objects 496
SAS Server Objects 497
Stored Processes 506
SAS Results 512
Schedule 514
Segments 515
Sockets 515
SQL 519
Other Objects 523
Zip Archives 523
Journals 523

Alpha Shape

For the following messages, ashape stands for an alpha shape or a reference to one.

---

ashape <<Get Alpha

Returns the current alpha value.

---

ashape <<Set Alpha(*alpha*)

Sets the current *alpha* value and recomputes the triangulation.

---

ashape <<Get Tri Alpha

Returns the alpha values for each triangle.

Associative Arrays

For the following messages, map stands for an associative array or a reference to one.

---

map<<First

Returns the first key within *map*, or Empty() if *map* has no keys. Note that keys are returned in lexicographical order.

---

map<<Get Contents

Returns a list of all key-value pairs within *map*.

---

map<<Get Keys

Returns a list of all the keys within *map*.

---

map<<Get Default Value()

Returns the implicit value of all absent keys, or Empty() if none has been set.

---

map<<Get Value(*key*)

Returns the value for the *key* within *map*.

`map<<Get Values(<{keyList}>)`

If no argument is provided, a list of all values within *map* is returned.

If a list of keys is provided, a list of the values corresponding to only those keys is returned.

`map<<Insert(key, value)`

Inserts the *key* into *map* and assigns *value* to it. If *key* already exists in *map*, its value is replaced by the new *value* given. This message is equivalent to the function `Insert Into`.

`map<<Next(key)`

Returns the key following the given *key* within the *map*, or `Empty()` if *map* has no keys. Note that keys are returned in lexicographical order.

`map<<Remove(key)`

Removes the *key* and *value* from *map*. This message is equivalent to the function `Remove From`.

`map<<Set Default Value(v)`

Sets the implicit value of all absent keys. Any key added without a value is assigned this value by default.

Classes

`obj<<Clone`

Returns a reference to a new class object that is a copy of the *obj* class object.

`obj<<Contains(quoted string)`

Returns 1 if the *obj* class object contains the specified quoted string expression, and 0 otherwise.

`obj<<Delete Class`

Deletes the *obj* class object.

obj<<Equal(*classref*)

Returns 1 if the *classref* class object is equal to the *obj* class object, and 0 otherwise.

obj<<First

Returns the quoted string representation of the name of the first member (item) in the *obj* class object. The members (items) in the class object are sorted in alphabetical order.

obj<<Get Contents

Returns a list of members (items) in the *obj* class object. Each element in the list is a two-item list that contains a key and an associated value.

obj<<Get Keys

Returns a list of keys within the *obj* class. Each key is a quoted string representation of the name of a member (item) in the *obj* class object.

obj<<Get Name

Returns a quoted string representation of the name of the *obj* class object.

obj<<Get Value(*key quoted string*)

Returns the value of the specified member (item) within the *obj* class object. The quoted *key quoted string* argument specifies the key to the member (item).

obj<<Get Values

Returns a list of values of the members (items) in the *obj* class object. Each element in the list is the expression that represents the value of each member (item) in the class.

obj<<Insert(*quoted string*, *value*)

Inserts a member (item) into the *obj* class object. The quoted *string* argument is the name of the member (item), and the *value* argument is the expression value of the member (item).

obj<<Lock Class(<*quoted string*|*{quoted stringList}*>)

Locks the *obj* class object, or locks specific members (items) within the *obj* class object. When a class object is locked, members (items) cannot be added, changed, or removed. The quoted *string* or quoted *stringlist* arguments specify a member (item) to lock. You can also specify a list of quoted strings to lock multiple members (items).

obj<<N Items

Returns the number of members (items) in the *obj* class object.

obj<<Next(*quoted string*)

Returns the quoted *string* representation of the name of the member (item) in the *obj* class object that follows the member (item) specified by the quoted *string*. The members (items) in the class object are sorted in alphabetical order.

obj<<Remove(<*string*|*{stringList}*>)

Removes the member (item) specified by the quoted *string* or quoted *stringlist* from the *obj* class. You can remove multiple members (items) using a list of quoted strings.

obj<<Show Contents

Shows the contents of the *obj* class object in the log window.

obj<<Unlock Class(<*quoted string*|*{stringList}*>)

Unlocks the *obj* class object, or unlocks specific members (items) within the *obj* class object. When a class object is unlocked, members (items) can be added, changed, or removed. The quoted *string* or *quoted stringlist* specify a member (item) to unlock. You can also specify a list of quoted strings to unlock multiple members (items).

Data Tables

dt<<Add Column Properties(*property argument*, ...)

Adds the specified properties (such as Value Order and Missing Value Codes) to the selected column.

dt<<Add Multiple Columns(*column prefix*, *n*, <"Before First"|"After Last"|After(*column*)>, "Character"|"Numeric"|"Row State", <Field Width(*n*)>)

Description

Adds *n* columns to *dt* at the position indicated.

Required Arguments

column prefix The prefix to add to the new columns names.

n The number of columns to add.

Character A new character column.

`Numeric` A new numeric column.

`Row State` A new row state column.

Optional Arguments

`Before First` Adds the columns before the first column.

`After Last` Adds the columns after the last column.

`After(`*column*`)` Adds the columns after the specified column.

`Field Width(`*n*`)` Specifies the width of the columns.

Notes

If you omit arguments, or the arguments are incorrectly specified, the Add Multiple Columns window appears.

`dt<<Add Rows(<`*n*`>, <"At Start"|"At End"|After(`*row number*`)>|{`*column name=value pairs*`})`

Description

Adds rows at the start, at the end, or after a specified row in the data table. The message can also add rows based on the specified column name and value pairs. Those rows are added to the end of the data table.

Notes

If you omit arguments, or the arguments are incorrectly specified, the Add Rows window appears.

`dt<<Add Scripts to Table(`*script*`, ...)`

`dt<<Add Properties to Table(`*script*`, ...)`

Adds the specified scripts to the data table.

`dt<<Anonymize(<Columns(`*column list(s)*`)>, <Output Table Name(`*name*`)>);`

Removes unique identifiers from data, some column properties, and table scripts. Applies to a data table or the specified list of columns. The new data table has the name that is specified by the quoted *name* argument.

`dt<<Begin Data Update`

Holds off display updating to allow for quick updating of data table cells. Use `End Data Update` in conjunction with this command to turn display updating back on.

Notes

`Begin Data Update` does not affect the data refresh due to some other table manipulations. For example, when you delete or add columns, the data table is updated and then the data update begins.

dt<<Clear Column Selection

Deselects all selected columns.

dt<<Clear Edit Lock(<"Modify Cells">, <"Add Rows">, <"Add Columns">, <"Delete Rows">, <"Delete Columns">)

Description

Allows the specified data table operation again.

Notes

If no arguments are specified, all locks are cleared.

dt<<Clear Row States

Cancels any row states in effect.

dt<<Clear Select

Turns off the current selection.

dt<<Clone Formula Column(*column*, *n*, Substitute Column Reference(*column1*, *{list}*))

Creates *n* new formula columns, substituting references to *column1* with columns from the *list* into the formula from the original *column*.

dt<<Close Data Grid(*Boolean*)

If true, closes the data table grid.

dt<<Close Side Panels(*Boolean*)

If true, closes the side panel in a data table.

dt<<Color or Mark by Column(*column*, *<named arguments>*)

dt<<Color by Column(*column*, *<named arguments>*)

dt<<Marker By Column(*column*, *<named arguments>*);

Description

Assigns colors or markers according to the values of a data table column. If no optional arguments are provided, colors are assigned according to the default color theme.

Required Argument

`column` The column to color or mark.

Optional Named Arguments

`Color(n)` Uses the specified JMP color.

`Add Marker(Boolean)` Shows or hides the marker in the data table.

`Color Theme(color theme)` Uses the specified quoted *color theme*.

`Marker Theme(marker theme)` Uses the specified quoted *marker theme:* `"Standard"`, `"Hollow"`, `"Paired"`, `"Classic"`, or `"Alphanumeric"`.

`Continuous Scale|Continuous Scale(Boolean)` Assigns colors in a chromatic sequential fashion based on the values in the highlighted column.

`Reverse Scale|Reverse` Reverses the color scheme in use.

`Excluded Rows(Boolean)` If true, applies the row states to excluded columns.

`"Make Window with Legend"` Creates a separate window with a legend.

`dt<<Color Rows by Row State`

Colors the rows in the data table grid using the color assignments by row states. Send the message again to turn off the row colors.

`dt<<Combine Columns(Delimiter("delim"), Columns(column1, column2, etc.), Column Name(quoted string))`

Combines several columns into a single column. Each source columns' values are separated by the delimiter specified as the quoted *delim* argument.

Examples

```
dt = Open( "$SAMPLE_DATA/Consumer Preferences.jmp" );
dt << Combine Columns(
  Delimiter( "," ),
  Columns(
     :Brush After Waking Up,
     :Brush After Meal,
     :Brush Before Sleep,
     :Brush Another Time
  ),
  Column Name( "When to Brush" )
);
```

`dt<<Compress File When Saved(Boolean)`

Compresses the file when the data table is saved.

`dt<<Compress Selected Columns({column1, ...})`

Compresses the listed columns into the most compact form that is possible. Columns with character data are compressed to 1 byte if there are fewer than 255 levels. Columns with numeric data are compressed to 1 byte if the integers are between -127 and 127.

`dt<<Concatenate(dt2|Data Table(name)|Multiple Data Table(name) arguments, (<"Private"|"Invisible">), <Output Table Name(name)>|"Append to First Table">, <"Keep Formulas">, <"Create Source Column">)`

Description

Creates a new table (*name*) from the rows of *dt* and *dt2*. By default, Concatenate creates a new data table and appends the rows of each data table that is specified.

Returns

A reference to the concatenated data table.

Required Arguments

`dt2|Data Table(name)|Multiple Data Table(name)` A data table reference or the names of the data table or data tables that you would like to combine.

Optional Arguments

`"Private"` A quoted keyword that opens the data table without displaying it in a data table window.

`"Invisible"` A quoted keyword that hides the data table. Use this argument to keep the data table hidden but use it in a subsequent expression. The data table is displayed in the Home Window's Window List and the Window > Unhide list.

`Output Table name(name)` The name of the final data table. If you do not enter a name, JMP names the data table Untitled # (for example, Untitled 1).

`"Append to First Table"` Appends rows to the first data table reference or data table name in the first argument. This option is an alternative to creating a new data table.

`"Keep Formulas"` Includes formulas in the final data table.

`"Create Source Column"` Adds a column called Source Table to the new data table.

Notes

`"Private"` and `"Invisible"` only apply if *not* using `"Append to First Table"`.

`dt<<Copy Column Properties`

Copies all of the column properties for the selected columns into a list of separate lists of properties. Optionally, you can specify a list of source columns instead of preselecting them in the data table.

Example

```
dt = Open( "$SAMPLE_DATA/Tiretread.jmp" );
dt << Select Columns( :MODULUS, :ELONG );
dt << Copy Column Properties;
New Window( "Script", Script Box( "//Try paste here" ) );
```

or

```
dt = Open( "$SAMPLE_DATA/Tiretread.jmp" );
dt << Copy Column Properties( {:MODULUS, :ELONG} );
```

```
New Window( "Script", Script Box( "//Try paste here" ) );
```

dt<<Copy Selected Properties

Description

Copies the selected table properties to the clipboard.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Properties( {"Distribution", "Oneway"} );
proplist = dt << Copy Selected Properties();
New Window( "Script", Script Box( "//Try pasting here" ) );
```

dt<<Copy Table Script("No Data")

Copies the script to recreate the data table onto the clipboard so that it can be pasted somewhere else. Add the "No Data" argument to omit the data.

dt<<Copy Table Scripts

dt<<Copy Selected Properties

Description

Copies the selected scripts to the clipboard.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Properties( {"Distribution", "Oneway"} );
proplist = dt << Copy Table Scripts();
New Window( "Script", Script Box( "//Try pasting here" ) );
```

dt<<Data Filter(<Location(*x*, *y*), <"Close Outline">, <"Local">, <Inverse(*Boolean*)>, <Show Columns Selector(*Boolean*)>, <Title(*quoted string*)>, <Save and Restore Current Row States(*Boolean*)>, <Conditional(*Boolean*)>, <Auto Clear(*Boolean*)>, <Group By AND(*Boolean*)>, <Show Histograms and Bars(*Boolean*)>, <Count Excluded Rows(*Boolean*)>, <Mode(...)>, <Add Filter((*cols*(...), <Where(...)>, <Display(...)>, <Select Missing(*cols*)>, <Order By Count(*cols*)>)>, <Favorites(...)>, <Animation(...)>)

Constructs a data filter. If no arguments are specified, the Add Filter Columns window appears.

Optional Arguments

Location(*x*, *y*) Moves the data filter window to the specified location. *x* and *y* are measured in pixels. 0,0 is the top left of the monitor.

"Close Outline" Closes the data filter outline.

"Local" Enables the filter to be embedded in reports to filter one or more platforms without affecting other reports.

Inverse(*Boolean*) Selects all but the specified rows for all filters.

Show Columns Selector(*Boolean*) If true, a column list is shown that adds a new column to the filter.

Title(*quoted string*) The title that is displayed on the outline.

Save and Restore Current Row States(*Boolean*) Restores your current row states when the Data Filter window is closed.

Conditional(*Boolean*) Limits the categories displayed for the selected filter column.

Auto Clear(*Boolean*) If you have more than one nominal or ordinal column selected in the Data Filter, this option clears any other selections before making a new selection.

Group By AND(*Boolean*) Enables you to create a filter group, specify OR, and add one or ore filters to create second filter group. If you specify Grouped By And, the behavior is reversed and grouped by AND instead.

Show Histograms and Bars(*Boolean*) Shows or hides the histogram and bars in the data filter.

Count Excluded Rows(*Boolean*) Shows or hides the number of excluded rows.

Mode The three modes of filtering: Select(*Boolean*) shows or hides the selected rows in the data table in a highlighted state; Show(*Boolean*) shows or includes the unselected rows and shows the Hide icon; Include(*Boolean*) shows or includes the unselected rows and shows the Exclude icon.

The global data filter default is Select(), Show(0), and Include(0), The local data filter default is Show(1), Include(1), Select() is not a valid option.

Add Filter Creates the data filter. Arguments include Columns(), Where(), Display(), Select Missing(*cols*), and Order By Count(*cols*). Columns() takes one or more column names separated by commas. You can add one or more Where clauses to define the filter.

Where Defines a Where clause by which the data is filtered.

Display(*column*, *size*, *display type*) Sets how the specified categorical column levels are displayed in the filter. The arguments are Blocks Display, List Display, Single Category Display, Check Box Display, Radio Box Display. In categorical columns, you can include the Find(Set Text(*quoted string*)) argument to include and initialize the search field. Display can also be included for a continuous column and can contain a size argument.

Select Missing Cols(*cols*) Selects missing values in continuous columns.

Order by Count(*cols*) For a categorical column, this option sorts the values in decreasing order by count.

Favorites Saves the current data filter criteria as a favorite.

Animation Cycles through the sorted values of the specified column, selecting and deselecting rows. Optional arguments include Animate Column(*col*), Animate

Rate(*number*), and "Forward"|"Backward"|"Bounce"highlights values from first to last. "Backward" highlights values from last to first. "Bounce" highlights forward and then backward repeatedly.

---

dt<<Data View(*<named arguments>*)

Description

Duplicates the data table in a new window. If you specify one of the following quoted arguments, the new data table includes only the corresponding rows.

Returns

A reference to the data view.

Optional Named Arguments

Excluded The new data table includes only the rows that are marked as excluded in the original data table.

Labeled|Labelled The new data table includes only the rows that are marked as labeled in the original data table.

Hidden The new data table includes only the rows that are marked as hidden in the original data table.

Selected The new data table includes only the rows that are selected in the original data table.

---

dt<<Delete Columns(*column1*, *column2*, ...)

dt<<Delete Column

Description

Deletes one or more columns from the data table *dt*. Specify which column or columns to delete. Without an argument, deletes the selected columns, if any.

---

dt<<Delete Rows(*<n>*)

dt<<Delete Rows(*{n, o, p, ...}*)

dt<<Delete Rows(*{n::q}*)

dt<<Delete Rows([n, o, p])

dt<<Delete Row(*preceding arguments*)

Description

Deletes the currently selected rows or rows specified. Returns the number of rows that were deleted.

dt<<Delete Scripts(*table script name*|*{table script1, table script2, ...}*)

Description

Deletes the specified data table script or scripts with the quoted name or names, or deletes a list of data table scripts.

Notes

In JMP versions prior to 14, use `Delete Table Property` to delete a table script.

dt<<Delete Table Property(*name*|*{property1, property2, ...}*)

Deletes a table property (for example, a script or variable) with the quoted *name*.

dt<<Delete Table Variable(*name*)

Deletes a table variable with the quoted *name*.

dt<<Disable Undo(*Boolean*)

If true, disables undo operations in the data table.

dt<<End Data Update

Resumes display updating after a `Begin Data Update` message. These commands are used for quick updates of the data table when many changes have to be made. Speed is gained by turning off display updating.

dt<<Exclude

dt<<Unexclude

Toggles selected rows in *dt* from excluded to unexcluded or *vice versa*.

dt<<Get All Columns As Matrix

Returns the values from all columns of *dt* in a matrix. Character columns are numbered according to the levels, starting at 1.

dt<<Get As Matrix(*<{list of columns by name}>*|*<{list of columns by number}>*, *<column range>*)

Returns values from the numeric columns of *dt* in a matrix. The default output is all numeric columns.

Examples

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
```

```
cols = dt1 << Get As Matrix(); // returns all numeric columns
Show( cols );
   cols =
   [ 12 59 95,
      12 61 123,
      12 55 74,...]
colnums = dt1 << Get As Matrix( {4, 5} ); // returns columns four and five
Show( colnums );
   colnums = [ 59 95, 61 123, 55 74, 66 145, 52 64, 60 84, 61 128, ...]

dt2 = Open( "$SAMPLE_DATA/Probe.jmp" );
colrange = dt2 << Get As Matrix( 10::22); // returns columns 10 through 22
Show( colrange );
   colrange =
   [ -0.08818069845438 0.711340010166168 1.85904002189636 0.396923005580902
      4.50656986236572 7.86504983901978 1.53891003131866 -2.76178002357483
      0.0711032971739769 5.75577020645142 -3.62023997306824 -0.971698999404907
      -0.0525696985423565, ...]
```

dt<<Get As Report

Returns the data table as a report. If rows and columns are selected in the data table, only those rows and columns are in the report.

Example

The following script returns `Big Class.jmp` as a report and displays it and a distribution in one window.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dtRpt = dt << Get As Report;
distRpt = V List Box(
   dt << Distribution(
      Continuous Distribution( Column( :weight ) ),
      Nominal Distribution( Column( :age ) )
   )
);
New Window( "Example", H List Box( dtRpt, distRpt ) );
```

dt<<Get Cell Height

Returns the data table cell height in pixels.

dt<<Get Column Names(*quoted string, <modeling type>, <data type>*)

Description

Returns a list of column names in a data table. The quoted string returns a list of quoted strings rather than a list of column references.

Required Argument

quoted string Returns a list of quoted strings rather than a list of column references.

Optional Arguments

modeling type The quoted modeling type. The options are "Continuous", "Ordinal", "Nominal", "Multiple Response", "Unstructured Text", "None", and "Vector".

data type The quoted data type. The options are "Numeric", "Character", "Row State", and "Expression".

Notes

The data types and the modeling types get only the specified types of columns. More than one of each type can be specified.

---

dt<<Get Column Reference(*{list of column names}|[matrix of column numbers]*)

dt<<Get Column References(*{list of column names}|[matrix of column numbers]*)

Returns the column references of the quoted strings in the list or matrix. If no list or matrix is used, JMP returns a list of references to all columns.

---

dt<<Get Display Width

Returns the column display width in pixels.

---

dt<<Get Edit Lock

Returns a list of the disallowed operations on the data table (if cells cannot be edited; rows cannot be added or deleted; and columns cannot be added or deleted).

---

dt<<Get Excluded Columns

Returns the currently excluded columns in the data table.

---

dt<<Get Excluded Rows

Returns the rows that are excluded in the data table.

---

dt<<Get Header Height

Returns the column header's display height (in pixels).

---

dt<<Get Hidden Columns

Returns the columns that are hidden in the data table.

dt<<Get Hidden Rows

Returns the currently hidden rows in the data table.

dt<<Get Journal

Returns a quoted string that contains journal source for the display box.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( Y( :weight ), X( :height ) );
rbiv = biv << Report;
Print( rbiv << Get Journal );
```

dt<<Get Label Columns

dt<<Get Labeled Columns

dt<<Get Labelled Columns

Returns the currently labeled columns in the data table.

Example

In PopAgeGroup.jmp, the Country and Year columns are labeled. The following script returns a list of the labeled column names.

```
dt = Open( "$SAMPLE_DATA/PopAgeGroup.jmp" );
dt << Get Labeled Columns;
   {:Country, :Year}
```

dt<<Get Labeled Rows

dt<<Get Labelled Rows

Returns the currently labeled rows in the data table.

dt<<Get Name

Returns the name of the data table.

dt<<Get Path

Returns the absolute path for the JMP data table. Note that this function is not for imported data that is not saved yet.

dt<<Get Property(*name*)

Returns the script from the quoted property *name*.

dt<<Get Row Change Function

Returns the expression that is evaluated when a row is selected.

dt<<Get Row ID Width

Returns the row ID display width in pixels.

dt<<Get Row States

Returns a vector containing the row state for every row in the data table or data filter.

dt<<Get Rows Where(*where clause*)

Returns the rows in the data table that match the specified Where criteria. Here are some examples:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Get Rows Where( :sex == "M" );
dt << Get Rows Where( :sex == "M" & :age < 15 );
```

dt<<Get Script(<*script name*>)

Returns the script specified by the quoted *script name*. If the *script name* is omitted, `Get Script` returns a text representation of the data table and all scripts in the table.

dt<<Get Script Group(<*group name*>)

Description

Returns the list of table scripts in the quoted *group name*. If no group name is specified, a list of all table scripts in all groups is returned.

dt<<Get Script Group Names

Description

Returns the list of names of table script groups.

dt<<Get Scroll Locked Columns

Returns a list of columns that are locked from scrolling.

dt<<Get Selected Columns(<*quoted string*>)

Description

Returns a list of selected columns as column references. Include the quoted *string* argument to return the selected column names as a list of strings.

dt<<Get Selected Properties(*<{list of properties}>*)

Description

Returns the selected table properties in a list.

Optional Argument

list of properties Specifies the properties to get.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Select Properties( {2, 4} );
proplist = dt << Get Selected Properties();
// returns the second and fourth table scripts and highlights them
// in the data table
```

dt<<Get Selected Rows()

Returns the selected rows.

dt<<Get Table Script Names()

Returns a list of the names of all the scripts and properties in the data table.

dt<<Get Table Variable(*name*)

Returns the value of the quoted *name* variable.

dt<<Get Table Variable Names

Returns a list of the names of all the variables in the data table.

dt<<Go To Row(*n*)

Locates and selects row number *n* in *dt*.

dt<<Group Columns(*{column1, column2, ...}*)

dt<<Group Columns(*group name*, *column*, *n*)

dt<<Group Columns(*first column*, *n*)

Description

Groups the columns under the specified quoted *group name*. You can provide either a list of columns to group, or a column name and the number of columns to group. In the latter case, the number *n* specifies to group the column given with the *n*-1 columns that follow.

dt<<Group Scripts({*script1, script2, ...*})

Description

Groups a list of table scripts in the data table.

dt<<Hide

dt<<Unhide

Toggles selected rows in *dt* from hidden to unhidden or *vice versa*.

dt<<Hide and Exclude

Hides the selected rows from graphs and excludes them from contributing to calculations.

dt<<Invert Column Selection(<{*list of columns*}>)

Description

Selects any column that is currently deselected and deselects any column that is currently selected. If the *list of columns* is specified, the columns that are not in the list are selected.

dt<<Invert Row Selection

Selects any row that is currently deselected and deselects any row currently selected.

dt<<Is Dirty

Returns 1 if the table has been modified from its saved state. Otherwise, returns 0.

dt<<Join(With(*Data Table(name)*), (<"Private">|<"Invisible">), Select(*columns*), Select With(*columns*), (By Matching Columns(*column1=column2, ...*)|"Cartesian"|"By Row Number"), <"Merge Same Name Columns">, <"Match Flag">, <Copy Formula(*Boolean*)>, <Suppress Formula Evaluation(*Boolean*)>, <"Update">, <Drop Multiples(*Boolean*, *Boolean*)>, <Include Non Matches(*Boolean*, *Boolean*)>, <"Preserve Main Table Order">, <Output Table Name(*name*))>

Description

Combines data tables *dt* and *Data Table* side to side.

Returns

A data table.

Required Arguments

With(*Data Table(name)*) Specifies the data table to join with the active table.

"Private" Specifies a quoted keyword that opens the data table without displaying it in a data table window.

"Invisible" Specifies a quoted keyword that hides the data table. Use this argument to keep the data table hidden but use it in a subsequent expression. The data table is displayed in the Home Window's Window List and the Window > Unhide list.

Select(*columns*) Selects the data table to join with the active table.

Select With(*columns*)

By Matching Columns(*column1=column2*) Selects columns in both tables whose values and data types match.

"Cartesian" Joins two tables using a Cartesian fashion, where it forms a new data table consisting of all possible combinations of the rows from two original data tables. JMP crosses the data in the first table with the data in the second to display all combinations of the values in each set.

"By Row Number" Joins the two tables side by side.

"Merge Same Name Columns" Data from the second table replaces the data of the same name columns in the original table. Note that missing values in the first table are replaced by nonmissing values in the second.

"Match Flag" Determines whether the Match Flag column is created when you are matching by column.

Copy Formula(*Boolean*) Includes formulas from the main table and/or the second table in the output columns.

Suppress Formula Evaluation(*Boolean*) Prevents JMP from evaluating columns' formulas during the creation of the new table.

"Update" Column data from the second table change the data of the same name columns in the original table. The results are displayed in a new data table. Note the following: JMP does not replace data with missing values; the output table uses the same columns as the original table. Thus, when you use "Update", Select Columns, the "Update" option is available only when joining by row number or by matching columns.

Drop Multiples(*Boolean*, *Boolean*) Specifies that you want the new table to contain only one row for each name. Applies only when matching by columns.

Include Non Matches(*Boolean*, *Boolean*) Includes non-matching columns in the main table and new data table. Applies only when matching by columns.

"Preserve Main Table Order" Maintains the order of the original data table in the joined table, instead of sorting by the matching columns.

Output Table Name(*name*) Specifies the name of the joined table. If you do not specify a name, JMP names the data table Untitled # (for example, Untitled 1).

---

dt<<Journal

Makes a journal from the data table. Only the data grid is included, not notes, variables, or scripts.

Note

Journals that are created in JMP 14 or later might contain compressed matrix data for large matrices. If you have JSL scripts that open journals and extract data from them, you might need to use the `Get Journal` message (which does not compress the matrices) rather than saving the journal to disk with the `Journal` message.

See Also

See "dt<<Get Journal"

`dt<<Journal Link(<Save(<path>)|Embed()>, <Button Name(<name>)>))`

Adds a link to the data table in the current journal. If a journal does not exist, a new one is created.

Optional Arguments

`path` Specifies the quoted *path* where the table is saved. If omitted, the data table should already have a disk location (previously saved or loaded), otherwise the journal link is incomplete and will not reload the table.

`Embed` Embeds a JSL script to recreate the data table.

`Button Name(name)` Specifies the name that is displayed on the button. The *name* argument is quoted. If the button name is not specified, the button is named after the data table.

`dt<<Label`

`dt<<Unlabel`

Toggles selected rows in *dt* from labeled to unlabeled or *vice versa*.

`dt<<Last Modified`

Returns the date on which the data table was last saved.

`dt<<Layout`

`Layout` is deprecated and will be removed in a future release. Use `Journal` instead.

`dt<<Lock Data Table`

Description

Locks the data table so that data and column properties cannot be added or changed.

See Also

"dt <<Set Edit Lock(<"Modify Cells">, <"Add Rows">, <"Add Columns">, <"Delete Rows">, <"Delete Columns">)"

dt<<Make Indicator Columns(<Append Column Name(*Boolean*)>, <Include Missing(*Boolean*)>)

Creates indicator columns of 0 and 1 values for the specified categorical columns.

Example

The following example creates indicator columns for the sex column. Append Column Name creates columns named sex_F and sex_M. Otherwise, the columns are named after each level (F and M). Include Missing includes missing values.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Make Indicator Columns(
   Columns( :sex ),
   Append Column Name( 1 ),
   Include Missing( 1 )
);
```

dt<<Make RowState Handler

Creates a row state handler function. The argument of the function holds the rows whose row states get changed.

dt<<Make SAS Data Step

Returns the data table as a SAS Data Step.

dt<<Make SAS Data Step Window

Returns the data table as a SAS Data Step and places it in a SAS script window.

dt<<Make Validation Column

Creates a column that is used to divide the data into training and validation sets.

dt<<Marker by Column(*column*)

Description

Assigns markers according to the values of the specified data table *column*.

See Also

"dt<<Marker By Column(column, <named arguments>);"

dt<<Markers(*n*)

Assigns marker *n* to the selected rows.

dt<<Maximize Display

Deprecated. Use Optimize Display instead.

Forces the data table to remeasure all of its columns and zoom to the best-sized window.

dt<<Move Script Group(*group name*, "To First"|"To Last"|After(table *script name*)|After(*group*))

Description

Rearranges the table script groups that are specified by the quoted *group name*.

dt<<Move Selected Column(*name(s)*, "To First"|"To Last"|After(*name*))

dt<<Move Selected Columns(*name(s)*, "To First"|"To Last"|After(*name*))

Description

Moves the selected column or columns in the data table to the specified position. The *name* argument is quoted.

Example

The following example moves the age column to the last column in Big Class.jmp:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Go To( :age );
dt << Move Selected Columns( "To Last" );
```

You may also use a list to specify the column names.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
list = {"name", "sex"};
dt << Move Selected Columns( list, "To Last" );
```

dt<<Move Rows("At Start"|"At End"|After(*n*))

Moves the selected rows in the data table to the specified position. *n* represents a row number.

dt<<New Column(*name*, *<data type>*, *<modeling type>*, <Format(*format*, *width*)>, <Formula()>, <Set Values(*{..., ..., }*)>, <Set Property*(properties)*>)

Description

Adds a new column titled with the quoted name after the last column in *dt*. Unless otherwise specified, columns are numeric, continuous, and 12 characters wide.

Returns

A column reference.

Required Argument

`name` The name of the new column.

Optional Arguments

`data type` A quoted string that specifies the data type. The options are `"Numeric"`, `"Character"`, `"Row State"`, or `"Expression"`.

`modeling type` A quoted string that describes the modeling type (`"Continuous"`, `"Nominal"`, `"Ordinal"`, `"Multiple Response"`, `"Unstructured Text"`, `"None"`, or `"Vector"`).

`Format(`*`format`*`, `*`width`*`)>` Sets the format type and column width.

`Set Values({})` Specifies the data in the column.

`Formula` Specifies the column formula.

`Set Property(`*`properties`*`)` Specifies any messages that data table columns support. Action arguments are found in the Column Properties menu in the New Column window. `Axis` and `Link Reference` are action argument.

See Also

For examples of setting other numeric format properties, see "col<<Format(<width>, <decimal places>, <"Use Thousands Separator">)".

dt<<New Data Box()

Makes a data table view in a display box tree. Useful for displaying the data table and report in one window. A data browser box is created when you send the `New Data Box` message to the data table object.

Example

The following script creates a data table view and report in one window. The data table is placed in a data browser box. The width of that box is set to 800 pixels. Because auto stretch is turned off, the data table view remains 800 pixels wide even if you stretch the right border of the window.

```
dtA = Open( "$SAMPLE_DATA/Semiconductor Capability.jmp", invisible );
nw = New Window( "Example",
  H List Box(
     V List Box( dtbox = dtA << New Data Box() ),
     dtA << Distribution(
        Continuous Distribution( Column( :NPN1 ) ),
        Continuous Distribution( Column( :PNP1 ) )
     )
  )
);
dtbox << Set Stretch( "Off", "Off" ) << Set Width( 800 );
```

dt<<New Data View

Opens a duplicate of the data table. The second data table is identical to and linked to the original data table, so that any changes made in one are reflected in the other. Closing either data table also closes the other and all references to the data tables are deleted.

This can be useful to show an invisible data table.

dt<<New Script(*name*, *script*)

dt<<Set Property(*name*, *script*)

Creates a new table property (also called a *table script*) using the quoted *name* that stores the specified *script*.

Use New Script() or Set Property() rather than the deprecated New Property() and New Table Property().

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Set Property( "Bivariate Example", Bivariate( Y( :weight ), X( :height
  ), Fit Line ) );
```

dt<<New Table Variable(*name*, *number*)

dt<<Set Table Variable(*name*, *number*)

Creates a new table variable with the quoted *name* and the *number*.

dt<<Next Selected

Scrolls data table down to show the next selected row that is not already in view.

dt<<Optimize Display

Forces the data table to remeasure all of its columns and zoom to the best-sized window.

dt<<Original Order

Restores saved order of columns in *dt*.

dt<<Paste Column Properties

Pastes multiple lists of column properties to multiple columns. Optionally, you can specify a list of target columns instead of selecting them in the data table.

Example

```
dt = Open( "$SAMPLE_DATA/Tiretread.jmp" );
dt << Copy Column Properties( {:MODULUS, :ELONG} );
dt2 = New Table( "test it",
```

```
    New Column( "T1", numeric, continuous ),
    New Column( "T2", numeric, continuous ),
    New Column( "T3", numeric, continuous ),
    Add Rows( 10 )
);
dt2 << Paste Column Properties( {:T1, :T3} );
// pastes the column properties from MODULUS and ELONG to T1 and T3
```

`dt<<Predictor Screening`

`dt<<Screen Predictors`

Description

Used to identify strong predictors.

Example

```
dt = Open( "$SAMPLE_DATA/Boston Housing.jmp" );
obj = dt << Predictor Screening(
  Y( :mvalue ),
  X( :crim, :indus, :chas, :nox, :rooms, :age, :distance, :radial )
);
```

`dt<<Previous Selected`

Scrolls data table up to show the previous selected row that is not already in view.

`dt<<Print Window(<"Show Dialog">)`

Prints the window. If the optional named argument `"Show Dialog"` is specified, the print window is displayed. Otherwise, the window is printed to the default printer using the current settings, and no print window is displayed.

`dt<<Rename Script Group(`*`old name`*`, `*`new name`*`)`

Description

Renames the table script group.

Example

```
dt << Rename Script Group( "Maps", "Street Maps" );
```

`dt<<Reorder By Data Type`

Reorders columns in *dt*, row state first, then character, then numeric.

`dt<<Reorder By Modeling Type`

Reorders columns in *dt* to continuous, then ordinal, then nominal.

`dt<<Reorder By Name`

Reorders columns in *dt* to alphanumeric order by name.

`dt<<Rerun Formulas`

Recalculates all formula-based data table variables. Recalculations are performed in the proper dependency order.

`dt<<Reverse Order`

Reverses columns in *dt* from current order.

`dt<<Revert`

Reverts to the most recently saved version of *dt*.

`dt<<Row Selection(Select Where(condition), <current selection("Extend"|"Restrict"|"Clear")> <Dialog("Keep Dialog Open")>)`

Description

Selects all rows that meet the specified condition.

Required Argument

`Select Where(condition)` Specifies the condition by which the rows are selected.

Optional Arguments

`current selection("Extend"|"Restrict"|"Clear")` Extends, restricts, or clears the existing selections. `Clear` is the default value.

`Dialog("Keep Dialog Open")` Shows the dialog so that the user can edit the options.

`dt<<Run Formulas`

Performs all pending formula evaluations, including evaluations that are pending as a result of evaluating other formulas.

`dt<<Run Script(`*name*`)`

Finds the table property with the quoted name and runs it as a JSL script.

`dt<<Save(`*path*`)`

`dt<<Save As(`*path*`)`

Description

Saves the table in the specified quoted *path*.

`dt<<Save Database(connection information, table name, <"Replace">)`

Saves the data table to the database named using the quoted connection information and quoted table name. The `"Replace"` option replaces the existing database with the current database.

`dt<<Save Script to Script Window`

Saves a script to reproduce the data table in a script editor window. Appends the script to any script that currently appears in the script editor.

`dt<<Select All Rows`

Selects all rows in the data table.

`dt<<Select Columns(<column1>, <column2>,...|"All")`

Selects the specified columns (or all columns) in the data table.

`dt<<Select Duplicate Rows`

Description

Selects the second and subsequent duplicate rows. If columns are selected, duplicate values are found in the rows of those columns. The duplicate values are case sensitive.

`dt<<Select Excluded`

Selects only those rows in the data table that are currently excluded.

`dt<<Select Hidden`

Selects only those rows in the data table that are currently hidden.

`dt<<Select Labeled`

Selects only those rows in the data table that are currently labeled.

`dt<<Select Randomly(n|p|Sample Size(n)|Sampling Rate(p))`

Randomly selects the given percentage *p* of the rows in the data table, or the number of rows *n*. If you specify an argument with a keyword, numbers between 0 and 1 represent a percentage, and numbers larger than 1 represent a number of rows.

`dt<<Select Rows([row1, row2, ...])`

Selects the rows given in the list of row numbers.

dt<<Select Script Group(*<group name|{group1, group2, ...}>*)

Selects the table script group specified as a quoted *group name* or a list of quoted strings. If no argument is provided, all groups are selected.

dt<<Select Where(*condition*, <Current Selection("Extend"|"Restrict"|"Clear")>)

Description

Selects the rows in *dt* where the condition evaluates as true.

dt<<Set Dirty(*Boolean*)

Marks the data table as changed, even if no changes have been made.

dt <<Set Edit Lock(<"Modify Cells">, <"Add Rows">, <"Add Columns">, <"Delete Rows">, <"Delete Columns">)

Description

Prevents cells from being modified; rows from being added or deleted; and columns from being added or deleted.

dt <<Set Cell Height(*n*)

Sets the cell height to the specified number of pixels.

dt<<Set Header Height(*n*)

Sets the column header's height to the specified number of pixels.

dt<<Set Label Columns(*column1, columns2, ...*)

Assigns the specified columns as label columns.

dt<<Set Matrix(*[matrix]*)

Inserts the specified matrix into a data table, adding new columns and rows as necessary.

dt<<Set Name(*name*)

Description

Specifies a name for the table. The *name* argument is quoted.

Returns

The data table name as a quoted string.

Notes

A change was made to the `Set Name` message so that now the new table name is returned as a quoted string. In previous releases, `Set Name` returned a scriptable data table object. As a result of this change, JMP scripts might need to be updated for the desired result to be returned. For example, rewrite the following script:

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" ) << Set Name( "Test" );
```

Separate the messages so that *dt* represents the data table instead of "Test":

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Set Name( "Test" );
```

The result is the same as in previous releases but will run successfully in both earlier and newer versions of JMP.

`dt<<Set Property(`*name*`, `*script*`)`

See "dt<<New Script(name, script)".

`dt<<Set Label Columns(`*column(s)*`, ...)`

`dt<<Set Label Columns`

Turns on the Label attribute for the specified columns. If no columns are listed, it turns the Label attribute off.

`dt<<Set Row ID Width(`*n*`)`

Sets the row ID display width to the specified number of pixels. If *n* is set to zero, the row ID display width automatically resizes.

`dt<<Set Row States(`*[matrix]*`)`

Sets the row states for all rows in the data table.

`dt<<Set Scroll Lock Columns(`*column name, ...*`)`

Locks scrolling for the columns specified as quotes quoted strings. If no columns are listed, unlocks scrolling.

`dt<<Set Table Variable(`*name*`, `*value*`)`

See "dt<<New Table Variable(name, number)".

```
dt<<Sort(<"Private">|<"Invisible">, <"Replace Table">, By(columns),
Order("Descending" | "Ascending"), <Output Table Name(name))>
```

Description

Creates a new table (named after the quoted name) by rearranging the rows of *dt* according to the values of one or more *columns*.

Returns

A reference to the sorted table.

```
dt<<Split(Split(columns), Split By(column), <Group(column)>,
<"Private">|<"Invisible">, <Remaining Columns("Keep All"|"Drop All"|
Keep(columns)|Drop(columns))>, <Copy Formula(Boolean)>, <Suppress
Formula Evaluation(Boolean)>, <Sort by Column Property>, <Output
Table (name)>)
```

Description

Unstacks multiple rows for each `Split` column into multiple columns as identified by the `Split by` column. The `Split` and `Split by` arguments are required.

Returns

A reference to the split data table.

Required Arguments

`Split(columns)` The column to split.

`Split By(column)` The column to split by.

Optional Arguments

`Group` Splits data within the specified groups.

`Remaining Columns("Keep All"|"Drop All"| Keep(columns)|Drop(columns))` Specifies what to do with the remaining columns in the resulting table. `Keep All` is the default setting.

Note: `Keep All` includes all columns in the output data table. However, the *values* of every column are not included. Because multiple rows are collapsed to a single row in the output data table, some values of the kept columns are dropped.

`Copy Formula(Boolean)` Includes column formulas from the source table in the resulting table.

`Suppress Formula Evaluation(Boolean)` Stops any copied formulas from being evaluated. True is the default setting.

`Sort by Column Property` Sorts the order of the output columns by the sort column property that is defined for the Split by column.

`Output Table(name)` Generates the output to the specified table name.

---

dt<<Stack(<"Private">|<"Invisible">, Columns(*columns*), <Source Label Column(*quoted string*)>, <Stacked Data Column(*quoted string*)>, <Copy Formula(*Boolean*)>, <Suppress Formula Evaluation(*Boolean*)>, <Drop All Other Columns(*Boolean*)|Name(*non-stacked columns*)(Keep(*column1, column2, ...*))|Name(*non-stacked columns*) (Drop(*column1, column2, ...*))>, <Output Table(*name*)>), <Number of Series(*n*)>, <"Contiguous">

Description

Creates a new table by combining the values from several columns in *dt* into one column.

Returns

A reference to the stacked data table.

---

dt<<Subscribe("keyname"(<"*client*">), On Delete Columns(<*function*>|<*script*>)|On Add Columns(<*function*>|<*script*>)|On Add Rows(<*function*>|<*script*>)|On Delete Rows(<*function*>|<*script*>)|On Rename Column(<*function*>|<*script*>)|On Close(<*function*>|<*script*>)|On Save(<*function*>|<*script*>)|On Rename(<*function*>|<*script*>))

Description

Subscribes to a data table to get messages regarding changes in the data table.

Returns

The keyname.

Arguments

"keyname"(<*client*>) Specifies the subscription name so that it can be referenced. The quoted client triggers a close confirmation when a close is attempted on the data table, warning that other open windows depend on the data table.

On Delete Columns(<*function*>|<*script*>) Returns the keyname when columns are deleted.

On Add Columns(<*function*>|<*script*>) Returns the keyname when columns are added.

On Add Rows(<*function*>|<*script*>) Returns the keyname when rows are added.

On Delete Rows(<*function*>|<*script*>) Returns the keyname when rows are deleted.

On Rename Column(<*function*>|<*script*>) Returns the keyname when columns are renamed.

On Close(<*function*>|<*script*>) Returns the keyname when the data table is closed. Takes one argument, a function. The function requires only one argument, the data table name.

On Save(<*function*>|<*script*>) Returns the keyname when the data table is saved.

`On Rename(<function>|<script>)` Returns the keyname when a rename is attempted on the data table. The function can be either the name of a previously defined function or the function itself.

Notes

Each subscription option remains in effect until you unsubscribe.

```
dt<<Subset(<"Private">|<"Invisible">, <"Selected Columns">,
<Columns(column list)>, <"Selected Rows">, <Rows([number, number,
...])>, <By(column list)>, <Sampling Rate(fraction)>, <Sample
Size(integer)>, <Stratify(column list)>, <Link to Original Data
Table(Boolean)>, <Copy Formula(Boolean)>, <Suppress Formula
Evaluation(Boolean)>, <"Keep by Columns">)
```

Description

Creates a new table from the rows and columns that you specify in *dt*.

Returns

A reference to the subset data table.

```
dt<<Summary(<"Private">|<"Invisible">, <Group(column)>,
<Subgroup(column)>, <N(column)>, <Mean(column)>, <Std Dev(column)>,
<Min(column)>, <Max(column)>, <Range(column)>, <Sum(column)>,
<CV(column)>, <Freq(column)>, <Weight(column)>, "Include Marginal
Statistics", <Link to Original Data Table(Boolean)>, <Statistics
Column Name Format(Stat(column)|Column|Stat of Column|Column Stat)>)
```

Description

Creates a new table of summary statistics for the column that you specify, according to groups and subgroups. `Statistics Column Name Format` values are quoted.

Returns

A reference to the summary data table.

```
dt<<Suppress Formula Eval(Boolean)
```

Turns off automatic calculation of formulas for data table *dt*.

```
dt<<Text to Columns(Delimiters(<separator>, <"Tab">, <"Newline">),
Columns(column1, column2...))
```

Makes a set of text columns or indicator columns from a delimited text column. `"newline"` includes the three forms: \r, \n, and \r\n. The separator is quoted.

Example

```
dt = Open( "$SAMPLE_DATA/Consumer Preferences.jmp" );
dt << Text To Columns(
```

```
    delimiter( "," ),
    columns( :Brush Delimited )
);
```

`dt<<Transpose(Columns(`*`columns`*`), Rows([`*`matrix`*`]), Output Table Name(`*`name`*`))`

Description

Creates a new table (named after the quoted *name*) from the rows and columns that you specify.

Returns

A reference to the transposed data table.

`dt<<Ungroup Columns({`*`column1, column2, ...`*`})`

Ungroups the columns defined in the list argument.

`dt<<Ungroup Scripts(`*`Name of Script Group`*`|{`*`script1, script2,...`*`})`

Description

Removes the specified table scripts or group from the group. The *Name of Script Group* argument is quoted.

`dt<<Unsubscribe(`*`keyname`*`, "On Delete Columns"|"On Add Columns"|"On Add Rows"|"On Delete Rows"|"On Close"|"On Col Rename"|"All")`

Releases any previous subscriptions to the data table *dt*. The *keyname* argument is quoted.

`dt<<Update from Database`

Updates the data in the table *dt* with data reimported from the database.

Columns

`col<<Add Column Properties(`*`name`*`, `*`expression`*`)`

Adds the quoted column property *name* with the *expression* given. You can add any standard column property by name or a user-specified property.

`col<<Add From Row States`

Updates a row state column with any currently used row state changes that are not the default state.

`col<<Add To Row States`

Copies all row state values in a column that are not the default state to the currently used row state in the data table.

`col<<Color Cells(color)`

Description

Colors the cells of the column within the data table grid. Use any quoted named color or 0 to clear the color.

`col<<Color Cell by Value(Boolean)`

Description

Colors the cells of the column in the data table grid using the value color property.

`col<<Copy Column Properties`

Copies the column properties into the buffer.

`col<<Copy From Row States`

Copies all row state values currently used in the data table to a column.

`col<<Copy to Row States`

Copies all row state values in the column to the currently used row state in the data table.

`col<<Data Type(type, <Format(format quoted string)>, <Input Format(format quoted string)>, <width>)`

`col<<Set Data Type(type, <Format(format quoted string)>, <Input Format(format quoted string)>, <width>)`

Description

Sets the *data type* to *col*.

Required Argument

`type` Specifies the `"Numeric"`, `"Character"`, `"Row State"`, or `"Expression"` data type.

Optional Arguments

`Format(format quoted string)` Specifies the way the data are displayed, such as `h:m` for hours and minutes. The *format quoted string* argument is quoted.

`Input Format(format quoted string)` Specifies the way the data are input. The *format quoted string* argument is quoted.

`width` (Optional for numeric data) Specifies 1, 2, or 4 (the number of bytes in the column).

`col<<Delete Formula`

Deletes the formula from a column.

`col<<Delete Property(`*`name`*`)`

`col<<Delete Column Property(`*`name`*`)`

Deletes the quoted property name from a column.

`col<<Eval Formula`

Forces the formula to evaluate (perhaps again). If formula suppression is enabled, the evaluation is not performed.

`col<<Exclude(`*`Boolean`*`)`

Turns the excluded or unexcluded state on, depending on the Boolean argument.

`col<<Format(<`*`width`*`>, <`*`decimal places`*`>, <"Use Thousands Separator">)`

`col<<Format("Best", <`*`width`*`>, <"Use Thousands Separator">)`

`col<<Format(("Fixed Dec"|"Percent"), <`*`width`*`>, <`*`decimal places`*`>, <"Use Thousands Separator">)`

`col<<Format("Pvalue", <width>)`

`col<<Format(("Scientific"|"Engineering"|"Engineering SI"), <`*`width`*`>, <`*`decimal places`*`>)`

`col<<Format("Precision", <`*`width`*`>, <`*`decimal places`*`>, <"Use Thousands Separator">, <"Keep Trailing Zeroes">, <"Keep All Whole Digits">)`

`col<<Format("Currency", <"Currency Code">, <`*`width`*`>, <`*`decimal places`*`>, <"Use Thousands Separator">)`

`col<<Format("Datetime", <`*`width`*`>, <`*`input format`*`>)`

`col<<Format(("Latitude DDD"|"Latitude DDM"|"Latitude DMS"|"Longitude DDD"|"Longitude DDM"|"Longitude DDM"), <`*`width`*`>, <`*`decimal places`*`>, ("PUN"|"DIR"|"PUNDIR"))`

`col<<Format("Custom", Formula(...), <`*`width`*`>, <`*`input format`*`>)`

Description

Sets the numeric display specified format.

Arguments

See *Using JMP* for more information about the arguments.

Examples

```
col<<Format( 10, 2, "Use thousands separator");
col<<Format( "Currency", "EUR", 20 );
col<<Format( "m/d/y", 10 );
col<<Format( "Precision", 10, 2, "Keep trailing zeroes", "Keep all whole
  digits" );
col<<Format( "Latitude DDD", "PUNDIR"); // "PUN" for punctuation, "DIR" for
  direction, PUNDIR for both
col<<Format( "Custom", Formula( Abs( value ) ), 15 );
```

`col<<Formula(`*expression*`)`

`col<<Set Formula(`*expression*`)`

Sets the formula for the variable and evaluates it.

`col<<Get Column Field Width`

Returns the field width used for displaying data in the column.

`col<<Get Data Type`

Returns the data type of *col*.

`col<<Get Data Type Length`

Returns the data type and length of the data column. Only the data type is returned if the data length is not fixed, as with character columns.

`col<<Get Format`

Returns the format of the column.

`col<<Get Formula`

Returns the formula.

`col<<Get Hidden`

Returns 1 if the column is hidden.

`col<<Get Input Format`

Returns the format used for input and storing of data for the column.

`col<<Get Labeled`

Returns 1 if the column is labeled.

`col<<Get List Check`

Returns the list check definition. If list check is not defined for the column, a message is sent to the log stating so.

`col<<Get Lock`

Returns the current Lock setting.

`col<<Get Modeling Type`

Returns the modeling type of the column.

`col<<Get Name`

Returns the name of the column.

`col<<Get Property("property name")`

Returns the specified property definition. If the specified property is not defined for the column, a message is sent to the log stating so.

`col<<Get Range Check`

Returns the range check definition. If range check is not defined for the column, a message is sent to the log stating so.

`col<<Get Role`

Returns the preselected role of *col*.

`col<<Get Script`

Returns the script to reproduce the column.

`col<<Get Scroll Locked`

Returns 1 if the column is scroll locked.

`col<<Get Selected`

Returns 1 if the column is selected, or 0 otherwise.

`col<<Get Stored Values`

Returns the values in the columns without considering the Missing Value Codes column property.

`col<<Get Value Labels`

Returns the value labels definition. If value labels is not defined for the column, a message is sent to the log stating so.

`col<<Get Use Value Labels`

Returns 1 if the value labels are set to be used for the column, or 0 otherwise.

`col<<Get Values`

Returns the values in the column.

`col<<Hide(Boolean)`

Turns the Hide attribute on or off according to the Boolean argument given.

`col<<Ignore Errors`

Ignores formula evaluation errors in a column, and sets the cell value to missing when a formula error occurs.

`col<<Input Format(`*`format`*`)`

Sets the quoted *`format`* used for input and storage for the column. The argument is the name of any JMP format (for example, `"ddmmyyyy"` for a date column).

`date_col<<Is Transformed On SAS Export`

Returns true if the data in the resulting SAS data set for the date column will be changed when it is exported to SAS.

`col<<Label(`*`Boolean`*`)`

Turns the Label attribute on or off according to the Boolean argument given.

`col<<Lock(`*`Boolean`*`)`

`col<<Set Lock(`*`Boolean`*`)`

Turns the Lock attribute on or off according to the Boolean argument given.

`col<<Preselect Role(role)`

Preselects the specified *role* for the column. Choices are `"Y"`, `"X"`, `"Weight"`, `"Freq"`, and `"None"`, or `"No Role"`.

`col<<Reset Transform`

Removes the cached data for the transform column. Accessing column data rebuilds the caches. Use this option to reduce memory or to allow recalculation if the formula depends on external information.

`col<<Set Display Width(n)`

Sets the column display width to the *n* in pixels. If *n* is set to zero, the column display width automatically resizes.

`col<<Set Each Value(n)`

Sets all the values in the column to *n*.

`col<<Set Excluded`

Excludes the column.

`col<<Set Field Width(n)`

Sets the field width for the column to *n*.

`col<<Set Hidden`

Hides the column.

`col<<Set Labeled`

`col<<Set Labelled`

Uses the column's data values for labels.

`col<<Set Modeling Type(type)`

Sets the modeling type for the variable. Choices are `"Continuous"`, `"Ordinal"`, `"Nominal"`, `"None"`, `"Row State"`, `"Unstructured"`, `"Multiple Response"`, or `"Vector"`.

`col<<Set Name(name)`

Sets the name for the column. The *name* argument is quoted.

`col<<Set Property(name, expression)`

Sets the quoted property name to the *expression* given. You can set any standard column property by name or a user-specified property.

Examples

The following example adds the Value Colors column property to the sex column, with pink for females and blue for males.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column( "sex" ) << Set Property( "Value Colors", {"F" = 78, "M" = 69} );
```

The following example adds a custom column property named *Date recorded* to the height column.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Column( "height" ) << Set Property( "Date recorded", 05Jan1990 );
```

See Also

Scripting Guide

Using JMP

`col<<Set Scroll Locked(Boolean)`

Turns the Scroll Lock attribute on or off according to the Boolean argument given.

`col<<Set Selected(Boolean)`

Sets the column to be selected or not selected.

`col<<Set Use for Marker`

`col<<Use for Marker`

Uses the values in the column as markers in graphs. Designed to use with expression columns and character columns that have IDs. In the Big Class Families.jmp sample data table, the picture column is specified to use as markers in graphs. Not supported in Bubble Plot.

`col<<Set Values([matrix] or {list})`

`col<<Values([matrix] or {list})`

Sets values for the matrix (for numeric variables) or list (for character variables).

`col<<Suppress Eval(Boolean)`

Turns off automatic calculation of formulas for the column.

col<<Use For Marker(*Boolean*)

Uses the values in the column as markers in graphs or turns off the option. Designed to use with expression columns and character columns that have IDs. In the Big Class Families.jmp sample data table, the picture column is specified to use as markers in graphs.

Rows

row<<Colors(*n*)

Assigns the color *n* to the selected rows.

row<<Exclude(*Boolean*)

row<<Unexclude(*Boolean*)

Turns the excluded or unexcluded state on for the selected rows according to the Boolean argument given. Omit the argument to toggle the row state.

row<<Hide(*Boolean*)

row<<Unhide(*Boolean*)

Turns the Hide attribute on or off according to the Boolean argument given. Omit the argument to toggle the row state.

row<<Hide and Exclude

Shows or hides the selected rows from appearing on graphs, and excludes or unexcludes them from contributing to calculations.

row<<Label(*Boolean*)

row<<Unlabel(*Boolean*)

Turns the Label attribute on or off according to the Boolean argument given. Omit the argument to toggle the row state.

row<<Markers(*marker*)

Assigns the quoted "*marker*" to the selected rows.

row<<Next Selected

Causes the next selected row in the data table to blink.

`row<<Previous Selected`

Causes the previous selected row in the data table to blink.

`row<<Row Editor`

Opens the Row Editor window for the selected rows.

Data Filter

`dtf<<Add Favorites(`*`name`*`)`

Description

Associates the current filter selection with the quoted *"name"* and saves it in the Favorites list.

Returns

The favorite as a quoted string.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
df = dt << Data Filter(
   Add Filter(
      Columns( :age, :sex, :height, :weight ),
      Where( :sex == "F" ),
      Where( :height >= 55 & :height <= 65 )
   ),
   Mode( Select (1) )
);
Wait( 1 ); // for demonstration purposes
fav1 = df << Add Favorites( "Female Average Ht" );
```

`dtf<<Add Filter(Columns(`*`column1`*`, <`*`column2`*`>), <Where(`*`clause`*`)>)`

Add one or more filter columns in a new OR group.

`dtf<<Auto Clear(`*`Boolean`*`)`

Clears all currently selected rows before setting a new selection.

`dtf<<Clear`

Clears the currently selected rows.

`dtf<<Close`

Closes the data filter window.

`dtf<<Columns(column1, column2, ...)`

Sets the columns to use in the data filter.

`dtf<<Data Table Window`

Shows the data table that the data filter window is using.

`dtf<<Delete All`

Removes all filters that are set.

`dtf<<Delete(column1, column2, ...)`

Removes the specified columns from the data filter.

`dtf<<Display(column, <Size(x, y)>, "Blocks Display"|"List Display"|"Single Category Display"|"Checkbox Display")`

Sets how the specified categorical column levels are displayed in the filter.

`dtf<<Get Script`

Returns the data filter script as text in the log.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
df = dt << Data Filter(
     Add Filter( Columns( :age, :sex ), Where( :age == 12 ) )
);
txt = df << Get Script;
Show( txt );
```

`dtf<<Local Data Filter`

Embeds the data filter in the specified window. See *Scripting Guide* for more information about local data filters.

`dtf<<Location(x, y)`

Moves the data filter window to the specified location. *x* and *y* are measured in pixels. 0,0 is the top left of the monitor.

dtf<<Make Filter Change Handler(*function*)

Creates a data filter handler to handle notification that the filter has changed. The number of rows filtered is returned in the argument to the function.

Example

```
dt = Open( "$SAMPLE_DATA/PopAgeGroupSubset.jmp" );
dist = Distribution( Automatic Recalc( 1 ), Continuous Distribution( Column(
   :POP ) ) );
filter = dist << Local Data Filter( Add Filter( Columns( :Region ) ) );
f = Function( {a}, Print( a ) );
rs = filter << Make Filter Change Handler( f );
```

dtf<<Make Subset

Creates a new subset data table that contains the rows that are selected in the data filter.

dtf<<Match(Filter Columns(*column1*, column2, ...), Where(*clause*)

Sets the filter conditions for each column. The *Where clause* is used for all the columns listed. To use different Where clauses for different columns, send the Match message separately for each column.

dtf<<Mode(Select(*Boolean*)|Show(*Boolean*)|Include(*Boolean*))

Sets the action, or mode, that is used when rows are selected using the data filter.

dtf<<Save and Restore Current Row States

Saves the current row states for the data table, and then restores those states when the data filter is closed.

dtf<<Show Columns Selector(*Boolean*)

Displays or hides the column selector after completing a filter.

dtf<<To Clipboard

Creates a Where clause from the current state of the data filter and places it on the clipboard, where it can be pasted elsewhere.

dtf<<To Data Table

Creates a Where clause from the current state of the data filter and saves it as a property to the data table.

`dtf<<To Journal`

Creates a Where clause from the current state of the data filter and appends it to the current journal. If there is no current journal, a new journal is opened and the Where clause is added to it.

`dtf<<To Row State Column`

Creates a row state column whose formula is the Where clause.

`dtf<<To Script Window`

Creates a Where clause from the current state of the data filter and appends it to the current script window. If there is no current script window, a new script window is opened and the Where clause is added to it.

`dtf<<Use Floating Window(`*`Boolean`*`)`

Sets whether the data filter window floats on top of its associated data table or behaves as a normal window.

`dtf<<Where(`*`clause`*`)`

Sets a condition for selecting rows.

Data Feed (Windows Only)

`feed<<Close`

Closes the data feed object and its window.

`feed<<Connect(`*`port settings`*`)`

Sets up port settings for the connection to the device.

`feed<<Disconnect`

Disconnects the device from the data feed queue but leaves the data feed object active.

`feed<<EOL("CR", "LF", "CRLF")`

Sets the line ending value used as a separator when parsing incoming lines of data. The value is also used as the terminator in outgoing lines of data.

– `"CR"`: ASCII character 13 (carriage return)

- "LF": ASCII character 10 (line feed)
- "CRLF": Uses both CR and LF in sequence.

`feed<<Get Line`

Returns and removes one line from the data feed queue.

`feed<<Get Lines`

Returns as a list and removes all lines from the data feed queue.

`feed<<Print Queue`

Prints the internal queue of messages to the log window.

`feed<<Queue Line(`*`quoted string`*`)`

Sends one quoted *`string`* (or *line*) to the end of the data feed queue. `Queue Line` is primarily useful for testing your script without requiring it to be attached to a device. You can essentially simulate the data coming from the device to make sure the rest of your code handles the values properly when it's really attached to a working device.

`feed<<Restart`

Restarts processing queued lines.

`feed<<Set Script(`*`script`*`)`

Assigns the *`script`* that is run each time a line of data is received.

`feed<<Stop`

Stops processing queued lines.

`feed<<Write(`*`quoted string`*`)`

Description

Sends a quoted *`string`* to the data feed device.

Example

```
exfeed = Open Datafeed(
   Connect( Port( "com1" ), Baud rate( 4800 ), Parity( "even" ), DataBits( 8 )
   ),
   Set Script(
      ex = exfeed << Get Line;
      Show( ex );
```

```
    )
);
exfeed << Write( "Ready" );
/* Example - send a message to external device over the serial port to trigger
   data messages. This can be used to send control messages to a sensor or
   other attached device.*/
```

feed<<Write Line(*quoted string*)

Description

Sends a quoted *string* to the data feed device. If EOL has been set for the data feed, the quoted strings are terminated by the specified EOL value. If EOL has not been set, the line is terminated with CRLF.

Example

```
exfeed = Open Datafeed(
   Connect( Port( "com1" ), Baud rate( 4800 ), Parity( "even" ), DataBits( 8 )
   ),
   Set Script(
      ex = exfeed << Get Line;
      Show( ex );
   )
);
exfeed << Write Line( "Ready" );
/* Example - send a message to external device over the serial port to trigger
   data messages. This can be used to send control messages to a sensor or
   other attached device.*/
```

feed<<Write Lines({*quoted string1, quoted string2, quoted string3*})

Description

Sends a list of quoted *strings* to the data feed device. If EOL has been set for the data feed, the quoted strings are terminated by the specified EOL value. If EOL has not been set, the line is terminated with CRLF.

Example

```
exfeed = Open Datafeed(
   Connect( Port( "com1" ), Baud rate( 4800 ), Parity( "even" ), DataBits( 8 )
   ),
   Set Script(
      ex = exfeed << Get Line;
      Show( ex );
   )
);
exfeed << Write Lines( {"Ready", "Set", "Go"} );
```

```
/* Example - send a message to external device over the serial port to trigger
   data messages. This can be used to send control messages to a sensor or
   other attached device.*/
```

Display Boxes

For additional examples, see the JMP Scripting Index.

All Display Boxes

---

db<<Add Text Annotation(Text(*quoted string*), Text Box(<*x1, y1, x2, y2*>))

Draws a text annotation box at the specified pixel location that contains the quoted *string*. The Text Box argument controls where the text annotation box is drawn in the window, from the upper left corner to the lower right corner.

Note that *x1*, *y1*, *x2*, and *y2* are not graph axis values but the specific pixel locations in the window. Exactly where the text box appears depends on the user's window size, display resolution, and so on.

---

db<<Append(*db2*)

Adds *db2* as the last child of the *db*.

---

db<<Child

Returns the child of the box.

---

db<<Class Name

Returns the name of the display class for the box.

---

db<<Clone Box

Makes a new copy of the display box.

---

db<<Close Window

Closes the containing window.

---

db<<Copy Picture

Puts a picture of the box on the clipboard.

db<<Delete

Deletes the display box.

db<<Enable(*Boolean*)

Controls the ability to interact with the display box. 0 disables the display box. 1 enables the display box.

db<<Get HTML

Returns a quoted string containing HTML source for the box.

db<<Get Journal

Returns a quoted string containing journal source for the box.

db<<Get Menu Item State(*index*)

Returns the popup menu item state of the index menu item. The state can be normal (0), checked (1), or disabled (-1).

db<<Get Menu Items

Description

Returns the menu items used for popup menu when the button is clicked. Menu items are returned in a list.

See Also

For submenus see "db<<Get Submenu(index)".

db<<Get Menu Script

Returns the menu script attached to the calling object.

db<<Get Page Setup()

Returns the page setup settings.

Example

The example below creates a new window and returns the page setup configuration.

```
w = New Window( "Window",
   Text Box( "Page Setup Test" )
);
w << Get Page Setup();
```

The results of the message:

```
{Margins( {0.75, 0.75, 0.75, 0.75} ), Scale( 1 ), Portrait( 1 ),
Paper Size( "Letter" )}
```

`db<<Get Picture( <Scale(n)> )`

Captures *db* as a picture object. The `Scale(n)` argument is a factor of the original picture size. For example, `Scale(2)` makes the picture object twice as large.

`db<<Get RTF`

Returns a quoted string containing RTF source for the box.

`db<<Get Script`

Returns the script for recreating the display box.

`db<<Get Size`

Returns either { *x, y* } or { *h, v* } in pixels:

```
xy = DisplayBox << Get Size;
```

Returns x and y in pixels:

```
{ x, y } = DisplayBox << Get Size;
```

`db<<Get Submenu(index)`

Returns the number of submenu items under the given menu item.

Example

The example below creates a menu containing *"A"*, *"B"*, and *"C"* with *"A"* having a submenu *"A1"* and *"A2"* and *"B"* having a submenu *"B1"*, *"B2"*, and *"B3"*. `<<Get Submenu(inc)` returns the number of submenu items under each indexed menu item.

```
New Window( "Title",
obj = Outline Box( "title" ) );
submenus = { };
obj << Set Menu Script(
   {"A", "", "A1", Print( "A1" ), "A2", Print( "A2" ),
   "B", "", "B1", Print( "B1" ), "B2", Print( "B2" ), "B3", Print( "B3" ),
   "C", Print( "C" )}
);
obj << Set Submenu( 1, 2 ); // menu A with 2 items in submenu A1 and A2
obj << Set Submenu( 4, 3 ); // menu B with 3 items in submenu B1, B2, and B3
For( inc = 1, inc <= N Items( Words( obj << Get Menu Script, "," ) )/2, inc++,
   Insert Into( submenus, obj << Get Submenu( inc ) );
);
submenus;
   {2, 0, 0, 3, 0, 0, 0, 0}
```

The log output indicates that index(1) contains two submenu items and index(3) contains three submenu items.

`db<<Get Text`

Returns a quoted string containing the text of the box.

`db<<Horizontal Alignment(`*`position`*`)`

Aligns a child display box inside the display parent box according to the specified *`position`*. The default value is `"Left"`, or you can specify `"Center"`, or `"Right"`.

Example

```
New Window( "Example",
   Outline Box( "Parent display box",
      Button Box( "OK", <<Horizontal Alignment( "Center" ) )
   )
);
```

`db<<Inval`

Invalidates the display box area in the window. The window is updated the next time the operating system has an opportunity to update windows (for example, when the user resizes the display box).

Notes

Consider including the message `<<Update Window` rather than including `Wait(0)`. The problem with using `Wait(`*`n`*`)` is knowing how large *`n`* should be.

Many display box messages, such as `<<Set Text`, automatically mark the box as invalid, so the `<<Inval` message is usually unnecessary. Some interactive scripts that use sliders with JSL callbacks might need `<<Update Window` to keep various parts of the display synchronized with the slider.

`db<<Is Enabled`

Returns the enabled state of the control. The message is supported in `Busy Light Box()`, `Button Box()`, `Calendar Box()`, `Check Box()`, `Col List Box()`, `Combo Box()`, `Completion Box()`, `Filter Col Selector()`, `gtext()`, `List Box()`, `Number Edit Box()`, `Popup Box()`, `Radio Box()`, `Range Slider Box()`, `Slider Box()`, `Spin Box()`, `Text Edit Box()`, `Tree Box()`, `Tree Map Box()`, and `Tree Map Seg()`.

`db<<Journal`

Appends the box to the journal.

db<<Journal Window

Appends the containing window of the display box to the journal; compare with Journal.

db<<Move Window(*x*, *y*)

Moves the window to the (*x*, *y*) location on your screen.

db<<Page Break

Inserts a page break before the box.

db<<Parent

Returns the parent of this display box.

db<<Prepend(*db2*)

Add *db2* to the display tree before *db*.

db<<Prev Sib

Returns the previous sibling of the display box.

db<<Reshow

Invalidates the display box's area in the window and immediate removes invalid areas from the window.

db<<Save Capture(<*path*>, <*format*>, <Add Sibling(*n*)>)

Saves the display box as a graphic to the specified quoted *path* in the specified quoted *format*. The optional Add Sibling argument adds the number of sibling display boxes to include in the capture. The default value is 1, which captures only the specified display box. Note that the specified portion of the report is not guaranteed to be scrolled into view or unobstructed by other windows. If the display box is not visible, the saved graphic will not contain the contents that you expect.

If you omit the path, you are prompted to name and save the file when running the path.

db<<Save HTML(<*path*>, <*format*>)

Saves the HTML source and folder of graphics to the quoted *path* and in the quoted *format*. If you omit the *path* argument, you are prompted to name and save the file when running the script.

db<<Save Interactive HTML(*<path>*, "Is Static")

Saves the display box as a web page (that includes interactive HTML features) in the quoted *path*. Non-JMP users can then explore the data. Note that the data is embedded in the web page.

Arguments

path A optional quoted *path* that specifies the location where the web page will be saved.

"Is Static" Omits the data from the web page and saves a static version of the web page.

Examples

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( y( weight ), x( height ) );
rbiv = (biv << Report);
rbiv << Save Interactive HTML( "$DOCUMENTS/MyInteractiveHTML.htm" );
```

db<<Save Journal(*<path>*)

Saves the journal source for the box in the quoted *path*. If you omit the argument, you are prompted to name and specify the graphic type.

db<<Save MSWord(*<path>*)

(Windows Only) Saves the display box as a Microsoft Word document in the quoted *path*. If you omit the *path* argument, you are prompted to name and save the file when running the script.

db<<Save PDF(*<path>*, <Show Page Setup(*Boolean*)>, <Portrait(*Boolean*)>)

Description

Saves a PDF of the display box in the quoted *path*.

Optional Arguments

path Saves the file in the quoted *path*. If you omit the argument, you are prompted to name and save the file when running the script.

Show Page Setup(*Boolean*) (Windows only) Displays the Page Setup window, where you can specify page orientation, headers and footers, margins, page scale, and paper size.

Portrait(*Boolean*) Displays the content in portrait or landscape orientation.

Notes

The PDF file contains headers and footers. Use Save Picture to omit these components.

`db<<Save Picture(<path>, <format>)`

Description

Saves a picture of the display box in the quoted *path* and with the specified quoted format.

Notes

- If you omit the quoted *path* argument, you are prompted to name and save the file when running the script.
- Valid file formats include "PDF", "PNG", "GIF", "JPG" or "JPEG", "EPS", "SVG", and "EMF".
- On Windows, the Windows Specific preferences determine the resolution (or DPI), or you can run the following script:

  `Pref( Save Image DPI( number ) );`
- On macOS, the operating system determines the DPI.
- Use `Save Picture` to export a report as a PDF file with no headers or footers. Use `Save PDF` to include these components.

`db<<Save Presentation(<path>, <Template(path)>, <Insert("Begin"|"End"|n)|Replace("Begin"|"End"|n)|Append>, <Outline Titles(title location)>, <format>)`

Saves display boxes in a Microsoft PowerPoint presentation. You can open the file in any presentation software program.

Optional Arguments

`path` Saves the file in the quoted *path*. You must include the .pptx extension in the filename. If you omit the *path* argument, you are prompted to name and save the file when running the script.

`Template(path)` Specifies the quoted *path* of a custom PowerPoint template. Without this argument, JMP uses the default template located in the `pptx` folder of the installation directory.

Include a simple table in your template, or a default table format is applied to report tables. For an example on Windows, see `/pptx/JMPExportTemplate.pptx` in the JMP installation folder.

`Insert` Determines where the slides are inserted in an existing presentation.

– *n* inserts the slides as the *n*th slide number.

– "Begin" inserts the slides at the beginning of the presentation.

– "End" inserts the slides at the end of the presentation.

`Replace` Determines which slides are replaced in an existing presentation. The arguments are *n*, "Begin", and "End" as described for `Insert`.

`Append` The slides are inserted at the end of an existing presentation.

`Outline Titles` The location of the outline title and any parent outline titles on the slide. By default, the immediate parent outline title appears as a slide title above the slide content, with any parent outline titles positioned in the bottom left corner of the slide.

- `"None"` omits the slide title above the graphic and the outline titles.
- `"Hide"` omits the outline titles.
- `"TopLeft"`, `"TopRight"`, `"BottomLeft"`, `"BottomRight"` determine the position of any of the parent outline titles on the slide.

`format` The format of the embedded graphics. Options are `"Native"`, `"EMF"`, `"PNG"`, `"JPG"`, `"BMP"`, `"GIF"`, `"TIF"`. On Windows, the native format is EMF. On macOS, the native format is PDF. See "Notes" for compatibility issues. Without this argument, JMP applies the "Image Format for PowerPoint" General preference.

Notes

Windows does not support the native PDF graphics produced on macOS. macOS does not support the native EMF graphics produced on Windows. For cross-platform compatibility, specify `"PNG"`, `"JPG"`, `"GIF"`, or `"TIF"`.

If no arguments are provided, the user is prompted to name and save the file.

`db<<Save RTF(<path>, format)`

Saves the file in the specified `quoted path` and with the quoted `format`. If you omit the `path` argument, you are prompted to name and save the file when running the script.

`db<<Save Text(<path>, format)`

Saves a file containing the text of the box in the quoted `path` and with the specified quoted `format`. If you omit the `path` argument, you are prompted to name and save the file when running the script.

`db<<Scroll Window(Display Box|relative-vertical-pixels|relative-horizontal-pixels, relative-vertical-pixels|{absolute-vertical-pixels, absolute-horizontal-pixels})`

Scrolls the containing window.

`db<<Select`

`db<<Deselect`

Selects (highlights) or deselects the box.

db<<Set Menu Item State(*index*, 0|1|-1)

Sets the popup menu item at *index* to be normal (0), selected (1), or disabled (-1).

db<<Set Page Setup<Margins(*left*, *right*, *top*, *bottom*)>, <Scale(*s*)>,<Portrait(*Boolean*)>, <Paper Size(*paper size*)>)

db<<Set Page Setup<Margins({*left*, *right*, *top*, *bottom*})>, <Scale(*s*)>,<Portrait(*Boolean*)>, <Paper Size(*paper size*)>)

Sets the page settings. Margins are set in inches. Scale variable *s* is a number in the range of 10 (for 1000%) to 0.2 (for 20%) with the default as 1 (for 100%). If Portrait is True the page is oriented for portrait, otherwise the page is landscape. Paper Size is a quoted string specifying the paper size, for example, "Letter" or "Legal".

Example

The example below creates a new window and configures the page setup.

```
w = New Window( "Window",
   Text Box( "Page Setup Test" )
);
w << Set page setup(
   margins( 1, 1, 1, 1 ),
   scale( 1 ),
   portrait( 1 ),
   paper size( "Letter" )
);
```

db<<Set Print Headers(*left header*, *center header*, *right header*)

Description

Sets the left, center, and right header for print output.

Example

```
w = New Window( "Window", Text Box( "Header Example" ) );
w << Set Print Headers(
   "Today is: &d;", // left
   "&wt;", // center
   "Page &pn; of &pc;" // right
);
w << Print Window;
```

db<<Set Print Footers(*left footer*, *center footer*, *right footer*)

Description

Sets the left, center, and right footer for print output.

Example

```
w = New Window( "Window", Text Box( "Footer Example" ) );
```

```
w << Set Print Footers(
   "Today is: &d;", // left
   "&wt;", // center
   "Page &pn; of &pc;" // right
);
w << Print Window;
```

db<<Set Stretch(x,y)

Description

Sets the horizontal and vertical stretching of display boxes.

Arguments

"Window" The box stretches to fill the space provided by the window, The sizing depends on the minimum and maximum properties for the box.

"Fill" The box stretches to fill the available space in its container.

"Off" The box does not stretch.

"Neutral" Many boxes start in the neutral state. A leaf box (a box that has no children) that is neutral resolves to "Off". Most container boxes aggregate the stretching behavior (and minimum and maximum) of their child boxes.

db<<Set Submenu (*index*, *submenu count*)

Description

Sets the submenu items for the item (specified by index number) by specifying the number of items in the submenu.

Example

The example below creates a menu containing “A”, “B”, and “C” with “A” having a submenu “A1” and “A2” and “B” having a submenu “B1”, “B2”, and “B3”.

```
New Window( "title", ob = Outline Box( "title" ) );
ob << Set Menu Script(
   {"A", "", "A1", Print( "A1" ), "A2", Print( "A2" ),
   "B", "", "B1", Print( "B1" ), "B2", Print( "B2" ), "B3", Print( "B3" ),
   "C", Print( "C" )}
);
ob << Set Submenu(1, 2); // menu A with 2 items in submenu A1 and A2
ob << Set Submenu(4, 3); // menu B with 3 items in submenu B1, B2, and B3
```

db<<Set Report Title(*title*)

Sets a new *title*. The *title* is quoted.

Show Properties(*db*)

Shows the messages a given display box can interpret.

db<<Sib

Returns the sibling of the display box.

db<<Sib Append(*db2*)

Appends a display as a sibling to this one. The argument must evaluate to a display box owner or reference.

db<<Size Window(*x*, *y*)

Resizes the containing window.

db<<Update Window

Updates the window that holds the display box (and possibly other windows as well, depending on the operating system) if there are invalidated regions. Previously invalidated box areas are redrawn with their new content.

Notes

In some interactive JSL scripts that combine sliders with JSL callbacks, you might need to use <<Update Window to keep parts of the display synchronized with the slider.

db<<Zoom Window

Resizes the window to be large enough to show all of its contents.

Axis Boxes

Axis Box<<Axis Settings(*<named arguments>*)

Opens the Axis Specification window or specifies axis settings, such as tick marks and axis labels.

If no arguments are included, the axis specification window appears.

Otherwise, specify named arguments for each axis.

- Specify the Y axis as Axis Box(1).
- Specify the X axis as Axis Box(2).

Optional Named Arguments

All Axes

```
Scale("Linear"|"Log"|"Power"|"Geodesic"|"Geodesic US"|"Custom
  Scale"|"Normal Probability|Weibull Probability|Frechet
  Probability|Logistic Probability|Exponential Probability|Gamma
```

Probability|Beta Probability|Mixture of 2 Normals Probabilities|Mixture of 3 Normals Probabilities) Specifies the scale of the axis. If the type is Custom Scale, this message expects two additional named arguments: Scale to Internal(*expr*) and Scale to External(*expr*).

Min(*n*) Changes the minimum value on the axis.

Max(*n*) Changes the maximum value on the axis.

Reverse Order(*Boolean*) Reverses the axes by reversing the minimum and maximum values.

Inc(*n*) Shows the numbers at the specified increments.

Set Font(*font*) Specifies the quoted *font* that is applied to the numbers. The JMP Font preferences determine the default font.

Set Font Size(*points*) Specifies the size of the font that is applied to the numbers. The JMP Font preferences determine the default font.

Set Font Style("Strikeout"|"Underline") Specifies the quoted style that is applied to the numbers.

Automatic Font Size(*Boolean*) JMP attempts to decrease the font size (down to a certain minimum) if all of the labels cannot fit at the default size. If 0, the font size is not decreased.

Automatic Tick Marks(*Boolean*) Turns on tick marks only if one or more labels are hidden (due to insufficient space).

Label Orientation("Automatic"|"Horizontal"|"Vertical"|"Perpendicular"|"Parallel"|"Angled") Rotates the axis label. The default value is "Automatic", which is based on the width of the labels.

Lower Frame(*Boolean*) Shows a frame below the labels. The default value is off.

Value Labels Displays the label that you specify instead of the data value.

Inside Ticks(*Boolean*) Shows tick marks inside or outside of the axis.

Add Ref Line({Label Row Nesting(*n*), *begin range*, <*end range*>, <"Solid"|"Dotted"|"Dashed"|"DashDot"|"DashDotDot">, <*color*>, <*label*>, <width*(n)*>, <*opacity*(%)>}) Defines the reference line range, line pattern, color, label, width, and opacity. A solid, black, 1-pixel line is the default setting. Label Row Nesting(*n*) specifies the number of nested rows on the axis. The *color* and *label* arguments are quoted.

Categorical Axes

Wrap Lines(*n*) Wraps long labels across multiple lines (*n*).

Numeric Axes

Format(*arguments*) Specifies the format of the numeric axis data. See the Format list in a numeric column's column properties for arguments. If you specify the a datetime

format, also include the `Interval` argument: `"Numeric"`, `"Year"`, `"Quarter"`, `"Month"`, `"Week"`, `"Day"`, `"Hour"`, `"Minute"`, or `"Second"`.

`Minor Ticks(`*`number`*`)` Specifies the *number* of minor tick marks between major tick marks.

`Tick Offset(`*`number`*`)` Specifies the starting point of the tick marks.

`Major Ticks(`*`Boolean`*`)` Shows or hides a major tick mark between each number.

`Minor Ticks(`*`Boolean`*`)` Shows or hides a minor tick mark between each number.

`Show Major Grid(`*`Boolean`*`)` Shows or hides a grid line at each major tick mark.

`Show Minor Grid(`*`Boolean`*`)` Shows or hides a grid line at each minor tick mark.

`Major Grid Line Color(`*`color`*`)` Sets the color for the major grid (if enabled) using the quoted *color*.

`Minor Grid Line Color(`*`color`*`)` Specifies the color of the grid line at each minor tick mark.

Example

The following example creates a bivariate plot and defines basic settings for the X and Y axes.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( X( :height ), Y( :weight ), FitLine );
rbiv = biv << Report;
xaxis = rbiv[Axis Box( 2 )];
yaxis = rbiv[Axis Box( 1 )];
xaxis << Axis Settings( Show Major Grid( 1 ) );
yaxis << Axis Settings( Decimal( 10, 3 ) );
```

`Axis Box<<Add Axis Label(`*`quoted string`*`)`

Adds an axis label with the specified quoted string.

`Axis Box<<Add Ref Line(`*`number`*`, `*`linestyle`*`, <`*`color`*`>, <`*`label`*`>, <`*`width`*`>)`

Adds a reference line at *number* in the specified *linestyle* (`"Solid"|"Dashed"|"Double"`), *color* (using the quoted *color*), quoted *label*, and *width* (in pixels).

`Axis Box<<Decimal(`*`width`*`, `*`decimal places`*`)`

Changes the numeric format for axis values.

`Axis Box<<Format(`*`name`*`)`

Changes to the numeric format given by the quoted *name*.

`Axis Box<<Get Inc(`*n*`)`

Gets the increment value of the axis.

`Axis Box<<Inc(`*n*`)`

Sets the increment between ticks.

`Axis Box<<Interval(`*format*`)`

Specifies the units used for `Inc()` with date/time formats: `"Numeric"`, `"Year"`, `"Quarter"`, `"Month"`, `"Week"`, `"Day"`, `"Hour"`, `"Minute"`, or `"Second"`.

`Axis Box<<Label Orientation(`*format*`)`

Rotates the axis label to one of the following formats: `"Automatic"` (the default setting based on the width of the label), `"Horizonal"`, `"Vertical"`, `"Perpendicular"`, `"Parallel"`, and `"Angled"`.

`Axis Box<<Major Grid Line Color(`*color*`)`

Sets the color for the major grid (if enabled) using the quoted *color*.

`Axis Box<<Max(`*maximum*`)`

Changes the maximum value on the axis.

`Axis Box<<Minor Grid Line Color(`*color*`)`

Sets the color for the minor grid (if enabled) using the quoted *color*.

`Axis Box<<Min(`*minimum*`)`

Changes the minimum value on the axis.

`Axis Box<<Minor Ticks(`*number*`)`

Specifies the *number* of minor tick marks between major tick marks.

`Axis Box<<Remove Axis Label`

Removes any label added with `Add Axis Label`.

Axis Box<<Reverse Scale(*Boolean*)

Reverses the normal scale direction so that the highest value is on the left or bottom (that is, closest to the origin).

Axis Box<<Revert Axis

Restores the axis' original settings (from time of creation).

Axis Box<<Scale(*type*)

Changes the scale of the axis to *type* ("Linear"|"Log"|"Exp Prob"|"Weibull Prob"|"Logistic Prob"|"Frechet Prob"|"Normal"|"Cube Root"|"Johnson Su Scale"|"Geodesic"|"Geodesic US"|"Custom Scale"|"Power"|"Gamma Prob"|"Beta Prob"|"Mixture of 2 Normals Prob"|"Mixture of 3 Normals Prob").

If the type is Custom Scale, this message expects two additional named arguments: Scale to Internal(expr) and Scale to External(expr).

Axis Box<<Tick Font(*name*, *<size>*, *<style|style style...>*, *<angle>*)

Sets the font name (quoted), size, and quoted properties for tick marks. To specify more than one style, include a space between each style and place them in quotes.

Axis Box<<Show Labels(*Boolean*)

Shows or hides labels for the axis values.

Axis Box<<Show Major Grid(*Boolean*)

Adds or removes grid lines at the major tick values.

Axis Box<<Show Major Ticks(*Boolean*)

Shows or hides major tick marks.

Axis Box<<Show Minor Grid(*Boolean*)

Adds or removes grid lines at the minor tick values.

Axis Box<<Show Minor Ticks(*Boolean*)

Shows or hides minor tick marks.

Axis Box<<Tick Label List(*<i>*, *{text1,text2, ...}*,*<{n1, n2, ...}>*)

Sets the values and positions of the axis tick labels.

Note: Major tick increments are automatically set to 1.0 if the tick labels are not specified.

Required Arguments

`{text1,text2, ...}` Specifies the quoted string titles for your labels.

Optional Arguments

i Specifies the label row index. Leaving it out clears any existing label rows and creates one new one as specified. Including it allows you to override any particular label row; using an index higher than the current number of label rows adds a new label row on to the end.

`{n1, n2, ...}` Specifies the values corresponding to each label. If the value list is omitted, the labels will be on integer increments starting with 1.

Border Boxes

Note: Border boxes support only one display box argument.

`Border Box<<Set Background Color({r, g, b}|<color>)`

Sets the background color for a border box. Specify an optional quoted list of RGB values or *color*. For example:

```
border box<<Set Background Color("red");
```

or

```
border box<<Set Background Color( {255, 192, 3} );
```

`Border Box<<Set Color({r, g, b}|<color>)`

Sets the border color for a border box. Specify a list of RGB values or a quoted `color`. For example:

```
border box<<Set Color("red");
```

`Border Box<<Get Color`

Gets the border color for a border box.

`Border Box<<Set Style(style)`

Sets the border style for a border box. Specify the *style* as one of the following numbers or keywords: `0` (`"Solid"`), `1` (`"Dotted"`), `2` (`"Dashed"`), `3` (`"DashDot"`), or `4` (`"DashDotDot"`). For example:

```
border box<<Set Style("Dotted");
```

Border Box<<Get Style

Gets the border style for a border box.

Data Browser Boxes

dbb<<Set Data Table(*<data table>*)

Sets the data table for the data browser box.

Data Filter Source Boxes

dfsb<<Set Row States(*dt*, *rs*)

Sets the row states for the given data table within the filter. Selections made in this row state will not be linked with the data table, but will be included in the reports linked to the selection filter.

Frame Boxes

Frame Box<<Add Graphics Script(*<order>*,*<description>*, *<script>*)

Description

Adds a script to draw graphics in the frame box.

Optional Arguments

order Specifies the order in which the graphics elements are drawn. The value can be the keyword "Back" or "Forward" or an integer that specifies the drawing order for a number of graphics element. 1 means the object is drawn first.

description A quoted string that appears in the Customize Graph window next to the graphics script. The *description* argument is quoted.

script A JSL script.

Example

In the following example, the graphics script draws the line first and then draws the other graphics elements: the grid lines, references lines, and markers that create the bivariate plot. Without the 1 order argument, the line is drawn last and covers up the markers.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
obj = dt << Bivariate( Y( :weight ), X( :height ) );
Report( obj )[FrameBox( 1 )] <<
Add Graphics Script(
  1, // draws the line first
  Description( "Pen Script" ),
```

```
    Pen Color( "red" );
    Pen Size( 5 );
    Y Function( 60 + 120 / 2 * (1 + Sine( (2 * Pi() * (x - 50)) / 22.5 )), x );
);
```

`Frame Box<<Append Seg`

Adds a display seg to the specified Frame Box.

`Frame Box<<Background Color({`*`RGB values`*`}|<`*`color`*`>)`

Changes the background color. Specify a list of RGB values or a quoted *color*.

`Frame Box<<Child Seg`

Returns the display seg child of the Frame Box.

`Frame Box<<Edit Graphics Script`

Brings up a dialog box to view, edit, or delete the current graphics scripts.

`Frame Box<<Find Seg`

Returns a display seg with the specified argument (for example, the name of a seg).

`Frame Box<<Frame Size(`*`x, y`*`)`

Resets the size of the frame, in pixel units.

`Frame Box<<Make Table of Graphs Like This`

Creates a data table of graphs.

`Frame Box<<Marker Size(`*`size`*`)`

Changes the marker size. The values are 0 (dot), 1 (small), 2 (medium), and so on.

`Frame Box<<Row Colors(`*`color`*`)`

`Frame Box<<Row Markers(`*`marker`*`)`

`Frame Box<<Row Exclude(`*`Boolean`*`)`

`Frame Box<<Row Hide(`*`Boolean`*`)`

`Frame Box<<Row Label(`*`Boolean`*`)`

Forwards commands to the data table associated with the report, so that the row states of selected rows can be manipulated. For `Row Exclude`, `Row Hide`, and `Row Label`, omitting

the argument toggles the option. If the option is off, the message turns it on. If the option is on, the message turns it off.

`frame box<<Set Background Fill(`*`Boolean`*`)`

Enables or disables filling the background with the background color. Use this option when you want to paste a graph and make the background transparent.

Example

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = Bivariate( y( weight ), x( height ) );
rbiv = biv << Report;
framebox = rbiv[Frame Box( 1 )];
// set background color
framebox << Background Color( "red" );
// for demonstration purposes: wait to see the color change
Wait( 1 );
// turn off background fill color
framebox << Set Background Fill( 0 );
```

`frame box<<X Axis(<Min(`*`minimum`*`)>, <Max(`*`maximum`*`)>, <Inc(`*`n`*`)>, <`*`named arguments`*`>)`

Scales the X coordinate system.

`frame box<<Y Axis(<`*`Min(min)`*`>, <`*`Max(max)`*`>, <Inc(`*`n`*`)>, <`*`named arguments`*`>)`

Scales the Y coordinate system.

Display 3D Boxes

`Graph 3D Box()`

Sends display commands to the 3D plot.

Excerpt Boxes

`Excerpt Box(`*`rptnum`*`, `*`lstSubscripts`*`)`

Returns a display box containing the excerpt designated by the report held at number `rptnum` and the list of display subscripts `lstSubscripts`. The subscripts reflect the current state of the report after previous excerpts have been removed.

Filter Col Selector

`Filter Col Selector(<Data Table(name)>, <width(pixels)>, <nLines(n)>, <script>, <onChange(expr)>)`

Returns a display box that contains a list of items. The control supports column filtering.

Global Boxes

`Global Box(value)`

Creates a display box that shows the value of a global variable.

Hier Boxes

`Hier Box(title, Hier Box(...), Hier Box(...), ...)`

Returns a display box with the *title* (quoted) that contains a hierarchy of quoted strings.

Matrix Boxes

`Matrix Box<<Get`

Returns the matrix contents.

`Matrix Box<<Make Into Data Table(<Invisible(Boolean)|Private(Boolean)>)`

Description

Turns the matrix into a new data table. `Invisible(1)` hides the data table from view. An invisible data table can be open from the JMP Home Window or the Window menu. `Private(1)` opens the data table without displaying it in a data table window. A private data table is generally for scripts that want better control of the data table by not exposing it to general use.

Returns

A reference to the new data table.

Matrix Box<<Set Format(*<width>*, *<decimal places>*, *<*"Use Thousands Separator">)

Description

Sets the numeric format for matrix elements.

Arguments

A number of other formats can be set on matrix boxes. See "Number Col Box<<Set Format(<width>|<width, decimal places>, <"Use Thousands Separator">)" for more information about the syntax.

Matrix Box<<Sort(*column number*, *ascending*)

Sorts the rows of the matrix based on the column number specified by column_num. The default sort order is ascending.

If column number is 0, the sort is removed.

ascending is a Boolean value. If ascending is "True", the sort is performed in ascending order. If ascending is "False", the sort is in descending order.

Nom Axis Boxes

Nom Axis Box<<Divider Lines(*Boolean*)

Adds or removes divider lines between labels in the axis box.

Nom Axis Box<<Lower Frame(*Boolean*)

Adds or removes a lower frame around the axis.

Nom Axis Box<<Rotated Tick Labels(*Boolean*)

Rotates or unrotates the labels at each tick value.

Number Col Boxes

Number Col Box<<Add Element(*item*)

Adds the *item* to the Number Col Box. *item* can be a single number, a list of numbers, or a matrix.

`Number Col Box<<Bootstrap(`*`nsample`*`, Random Seed(`*`number`*`), Fractional Weights(`*`Boolean`*`), Split Selected Column(`*`Boolean`*`), Discard Stacked Table if Split Works(`*`Boolean`*`)`

Description

Bootstraps the analysis, repeating it many times with different resampling weights and collecting tables as selected.

Arguments

`nsample` Sets the number of times that you want to resample the data and compute the statistics. A larger number results in more precise estimates of the statistics' properties. By default, the number of bootstrap samples is set to 2,500.

`Random Seed(`*`number`*`)` Sets a random seed that you can re-enter in subsequent runs of the bootstrap analysis to duplicate your current results. By default, no seed is set.

`Fractional Weights(`*`Boolean`*`)` Performs a Bayesian bootstrap analysis. In each bootstrap iteration, each observation is assigned a weight that is calculated as described in *Basic Analysis*. The weighted observations are used in computing the statistics of interest. By default, the fractional weights option is not selected and a simple bootstrap analysis is conducted.

`Split Selected Column(`*`Boolean`*`)` Places bootstrap results for each statistic in the column that you selected for bootstrapping into a separate column in the Bootstrap Results table. Each row of the Bootstrap Results table (other than the first) corresponds to a single bootstrap sample.

If you exclude this option, a Stacked Bootstrap Results table appears. For each bootstrap iteration, this table contains results for the entire report table that contains the column that you selected for bootstrapping. Results for each row of the report table appear as rows in the Stacked Bootstrap Results table. Each column in the report table defines a column in the Stacked Bootstrap Results table.

`Discard Stacked Table if Split Works(`*`Boolean`*`)` (Applicable only if the Split Selected Column option is included.) Determines the number of results tables produced by Bootstrap. If the `Discard Stacked Table if Split Works` option is not selected, then two Bootstrap tables are shown. The Stacked Bootstrap Results table, which contains bootstrap results for each row of the table containing the column that you selected for bootstrapping, gives bootstrap results for every statistic in the report, where each column is defined by a statistic. The unstacked Bootstrap Results table, which is obtained by splitting the stacked table, provides results only for the column that is selected in the original report.

`Number Col Box<<Get`

`Number Col Box<<Get(`*`i`*`)`

Gets the values in a list, or the *i*th value.

Number Col Box<<Get As Matrix

Gets the values in a matrix, specifically a column vector.

Number Col Box<<Get Format

Returns the current format.

Number Col Box<<Get Heading

Returns the column heading text.

Number Col Box<<Remove Element(*row number*)

Removes an element from the column at the specified position.

Number Col Box<<Set Format(*<width>*|*<width, decimal places>*, *<*"Use Thousands Separator">)

Number Col Box<<Set Format("Best", *<width>*, <"Use Thousands Separator">)

Number Col Box<<Set Format(("Fixed Dec"|"Percent"), *<width>*|*<width, decimal places>*, <"Use Thousands Separator">)

Number Col Box<<Set Format("Pvalue", *<width>*)

Number Col Box<<Set Format(("Scientific"|"Engineering"|"Engineering SI"), *<width>*|*<width, decimal places>*)

Number Col Box<<Set Format("Precision", *<width>*|*<width, decimal places>*, <"Use Thousands Separator">, <"Keep Trailing Zeroes">, <"Keep All Whole Digits">)

Number Col Box<<Set Format("Currency", *<currency code>*, *<width>*|*<width, decimal places>*, <"Use Thousands Separator">)

Number Col Box<<Set Format(*datetime*, *<width>*, *<input format>*)

Number Col Box<<Set Format(("Latitude DDD"|"Latitude DDM"|"Latitude DMS"|"Longitude DDD"|"Longitude DDM"|"Longitude DDM"), *<width>*|*<width, decimal places>*, ("PUN"|"DIR"|"PUNDIR"))

Number Col Box<<Set Format("Custom", Formula(...), *<width>*, *<input format>*)

Description

Sets the column format.

Arguments

Using JMP describes the arguments. Note that `Matrix Box()`, `Number Col Box()`, `Number Col Edit Box()`, `Number Edit Box()` have the same `Set Format` syntax.

Examples

```
<<Set Format( 10, 2, "Use thousands separator");
<<Set Format( "Currency", "EUR", 20, );
<<Set Format( "m/d/y", 10 );
<<Set Format( "Precision", 10, 2, "Keep trailing zeroes", "Keep all whole
  digits" );
<<Set Format( "Latitude DDD", "PUNDIR"); // "PUN" for punctuation, "DIR" for
  direction, PUNDIR for both
<<Set Format( "Custom", Formula( Abs( value ) ), 15 );
```

Notes

- For a list of currency codes, see *Scripting Guide*. The currency code is based on the locale if the code is omitted.
- If you don't specify the format, set the decimal places to greater than 100 for datetime values and to 97 for *p*-values.
- You must always precede the number of decimal places with the width.
- Options can be defined in a list or a variable, or they can be in a `Function()` that is evaluated.

```
ncbFunc = Function({}, {"Fixed", 12, 5});
```

`number col box<<Set Heading(`*quoted string*`)`

Changes the column heading text.

Number Col Edit Boxes

`Number Col Edit Box<<Set Format(`*<width>*`, `*<decimal places>*`, <"Use Thousands Separator">|`*<other options>*`)`

Description

Sets the column format.

Arguments

A number of other formats can be set on number col edit boxes.

See Also

"Number Col Box<<Set Format(<width>|<width, decimal places>, <"Use Thousands Separator">)"

Number Col Edit Box<<Remove Element(*x position*, *y position*, *i*)

Removes an element from the column at the specified position.

Number Edit Box

Number Edit Box<<Set Format(*<width>*, *<decimal places>*, <"Use Thousands Separator">|*<other options>*)

Description

Sets the column format.

Arguments

A number of other formats can be set on number edit boxes.

See Also

"Number Col Box<<Set Format(<width>|<width, decimal places>, <"Use Thousands Separator">)"

Outline Boxes

Outline Box<<Close(*Boolean*)

Closes the outline box.

Outline Box<<Close All Below

Closes all the node's child nodes.

Outline Box<<Close All Like This

Closes all nodes similar to this outline box.

Outline Box<<Close Where No Outlines

Closes all nodes that do not have children.

Outline Box<<Get Title

Gets the title of the outline box.

Outline Box<<Horizontal(*Boolean*)

Horizontally aligns the node's children.

`Outline Box<<Open All Below`

Opens all the node's child nodes.

`Outline Box<<Open All Like This`

Opens all nodes similar to this outline box.

`Outline Box<<Set Menu Script(`*{quoted string1, script1, quoted string2, script2, ...}*`)`

Adds an entry to the menu when the red triangle on an outline box is selected.

`Outline Box<<Set Title(`*title*`)`

Specifies the quoted *title* of the outline box.

Panel Boxes

`Panel Box<<Get Title`

Gets the title of the panel box.

`Panel Box<<Set Title(`*title*`)`

Specifies the *title* (quoted) of the panel box.

Plot Col Boxes

`Plot Col Box<<Get As Matrix`

Gets the values in a matrix, specifically a column vector.

`Plot Col Box<<Get Labels`

Gets the labels for each row.

`Plot Col Box<<Remove Element(`*row number*`)`

Removes an element from the column at the specified position.

`Plot Col Box<<Set Labels(`*{list}*`)`

Sets the labels for each row.

`Plot Col Box<<Set Scale(`*minimum*`, `*maximum*`, <"format">, <`*width*`>, <`*decimal places*`>, <"Use Thousands Separator">)`

Description

Specifies the minimum and maximum values of the horizontal axis.

Arguments

The format arguments specify a format that is used when the first two arguments are 0 and 1. A number of formats can be set on plot col boxes. See "Number Col Box<<Set Format(<width>|<width, decimal places>, <"Use Thousands Separator">)" for more information about the syntax.

`Plot Col Box<<Set Values(`*[matrix]* or *{list}*`)`

Sets values for the matrix (for numeric variables) or list (for character variables).

Slider Boxes and Range Slider Boxes

`Slider Box<<Get(`*<index>*`)`

`Range Slider Box<<Get Lower(`*<index>*`)`

`Range Slider Box<<Get Upper(`*<index>*`)`

Returns the current value of the slider.

`Slider Box<<Get Max()`

Returns the maximum value possible for the range slider and slider.

`Slider Box<<Get Min()`

Returns the minimum value possible for the range slider and slider.

`Slider Box<<Get Var`

`Range Slider Box<<Get Lower Var`

`Range Slider Box<<Get Upper Var`

Returns the variable name associated with the slider.

Slider Box<<Set(*float*, <*index*>, <Run Script(*Boolean)*>)

Range Slider Box<<Set Lower(*float*, <*index*>, <Run Script(*Boolean*)>)

Range Slider Box<<Set Upper(*float*, <*index*>, <Run Script(*Boolean*)>)

Sets the value of the slider. Run Script(*Boolean*) controls whether an on-change script runs after the Set, Set Lower, or Set Upper message.

Slider Box<<Set Max(*float*, <*index*>)

Sets the maximum value possible for the range slider and slider.

Slider Box<<Set Min(*float*, <*index*>)

Sets the minimum value possible for the range slider and slider.

Slider Box<<Set Script(<*script*>)

Sets a script to be run when the range sliders and slider is updated.

Slider Box<<Set Var(*slider variable*)

Range Slider Box<<Set Lower Var(*slider variable*)

Range Slider Box<<Set Upper Var(*slider variable*)

Sets the variable name associated with the slider.

String Col Boxes

String Col Box<<Add Element(*item*)

Adds the item to the String Col Box. Item can be a single quoted string or a list of quoted strings.

String Col Box<<Get

String Col Box<<Get(*i*)

Gets the values in a list or the *i*th value.

String Col Box<<Get Heading

Returns the column heading text.

String Col Box<<Remove Element(*row number*)

Removes an element from the column at the specified position.

String Col Box<<Set Allow Text Search(*Boolean*)

Description

In table boxes with selectable rows, allows a quoted string column that has focus to respond to keyboard input to change the selected row.

Example

```
// Run the example.
// Select K2.
// Type the letter g. Notice the last row is selected.
// Type the letters ki. Notice the third row is selected.
New Window( "Mountains",
   tb = Table Box(
      sb =
      String Col Box( "Mountain",
         {"K2", "Delphi", "Kilimanjaro",
         "Grand Teton"}
      ),
      Number Col Box( "Elevation (meters)",
         {8611, 681, 5895, 4199}
      ),
      Plot Col Box( "", {8611, 681, 5895, 4199} )
   )
);
tb << Set Selectable Rows( 1 );
sb << Set Allow Text Search( 1 );
```

String Col Box<<Set Heading(*title*)

Changes the column heading specified in the quoted *title*.

String Col Box<<Set Justify(*justification*)

Specifies the alignment of the contents in the String Col Box to "Right", "Left", or "Center".

Tab Boxes

Tab Box<<Get Tab Margin()

Returns a list of the current margins in pixels for the tab box in this order: Left, Top, Right, and Bottom.

`Tab Box<<Set Style("Tab"|"Combo" |"Outline"|"Vertical Spread"|"Horizontal Spread"|"Minimize Size")`

Changes the appearance of the tab box from a tab to a combo box or outline node.

`"Vertical Spread"` and `"Horizontal Spread"` change the orientation of the tab title.

`"Minimize Size"` bases the tab style on the width of the tab title.

`Tab Box<<Set Tab Margin(`*n*`|{...})`

Sets the tab margin for the tab box. If a single number is specified, all four margins are set to that number of pixels. If a list of two numbers is specified, the left and right margins are set to the first number, and the top and bottom margins are set to the second number. If a list of four numbers is specified, the margins are set in this order: `{left, top, right, bottom}`.

`Tab Box<<Show Tabs(`*Boolean*`)`

Shows or hides the tabs for tab boxes. If you hide the tabs, you need to provide another way to select and show tabs. For example, a list box that contains a list of references to the tabs. The default value is 1.

Table Boxes

`Table Box<<Bootstrap(`*nsample*`, Random Seed(`*number*`), Fractional Weights(`*Boolean*`), Split Selected Column(`*Boolean*`), Discard Stacked Table if Split Works(`*Boolean*`)`

Description

Bootstraps the analysis, repeating it many times with different resampling weights and collecting tables as selected.

See Also

"Number Col Boxes"

`Table Box<<Get`

Gets the entries of the table in list form.

`Table Box<<Get As Matrix(<"Visible">)`

Gets the numeric entries of the table in matrix form. `"Visible"` means that only visible columns will be included.

Table Box<<Get Click Sort

Returns 1 if the table can be sorted by clicking a column header and 0 otherwise.

Table Box<<Get Locked Columns

Returns the number of columns that cannot be dragged with the cursor or have any columns dropped before them.

Table Box<<Get Row Change Function

Returns the expression that is evaluated when a row is selected.

Table Box<<Get Selectable Rows

Returns True if the table box currently allows row selection.

Table Box<<Get Selected Row Color

Returns the index number of the background color of the selected rows in the table box.

Table Box<<Make Combined Data Table

Returns a reference to the data table. Same as `Make Data Table`, but also searches the report for report tables with the same columns and combines all of these into the new data table.

Table Box<<Make Data Table(*name*)

Returns a reference to the data table. Turns the table entries into a new data table with the quoted *name* argument.

Table Box<<Reorder Columns(*from column index*, *to column index*)

Puts the column specified with *from column index* in the place of the column specified with *to column index*. The indexes are 0-based. For example, indicate the first column with "0", and indicate the second column with "1".

Table Box<<Set Cell Changed Function(Function({*this*, *col box*, *row*},<*script*>))

Description

Sets a function that is called whenever the user edits a cell in a column in a table.

Example

This example prints the new values for the changed cell to the log.

```
New Window( "Mountains",
   tb = Table Box(
      quoted string Col Edit Box(
         "Mountain",
         {"K2", "Delphi", "Kilimanjaro",
         "Grand Teton"}
      ),
      Number Col Edit Box(
         "Elevation (meters)",
         {8611, 681, 5895, 4199}
      ),
      Plot Col Box( "", {8611, 681, 5895, 4199} )
   )
);
tb <<
Set Cell Changed Function(
   Function( {this, col, row},
      Print(
         (col << Get Heading) || ": row:" ||
         Char( 3 ) || " is now " ||
         Char( col << Get( row ) )
      )
   )
);
```

Table Box<<Set Click Sort(*Boolean*)

Specifies if the table can be sorted by clicking a column header.

Table Box<<Set Column Borders(*Boolean*)

Draws a line on each side of the column.

Table Box<<Set Heading Column Borders(*Boolean*)

Draws a line on each side of the column headings.

Table Box<<Set Locked Columns(*n*)

Locks the first *n* columns. You cannot drag the locked columns or drag columns before them.

Table Box<<Set Row Borders(*Boolean*)

Draws a line above and below each row.

Table Box<<Set Row Change Function(*function*)

Sets the expression that is evaluated when a row is selected.

Table Box<<Set Selectable Rows(*Boolean*)

Makes the rows of the table box selectable or not.

Table Box<<Set Selected Row Color(*color*)

If the rows of the table box are selectable (`Set Selectable Rows(True)`), sets the background color (specified in the quoted *color* argument) for the selected rows.

Table Box<<Set Shade Cells(*Boolean*)

Shades the background of every cell in the table.

Table Box<<Set Shade Alternate Rows(*Boolean*)

Shades the background of every other row in the table.

Table Box<<Set Shade Heading(*Boolean*)

Shades the background in column headings.

Table Box<<Set Underline Headings(*Boolean*)

Draws a line underneath the column headings.

Table Box<<Sort By Column(<*column number*|*column title*>, <Ascending(*Boolean*)>

Sorts all rows based on the values in the specific column number or quoted column title. The default order sorting is descending.

Text Boxes

Text Box<<Font Color(*n*)

Sets the color for `Text` quoted strings.

Text Box<<Get Hidden State

Returns the current state of a text box.

Text Box<<Get Text

Returns the quoted string content of the box.

Text Box<<Get Tip

Returns the tooltip for the text box (or a text edit box).

Text Box<<Markup

Returns text formatted with the specified HTML tags. The HTML must be well-formed; make sure you close nested tags correctly.

The following example returns text formatted in bold, italic, and underlined.

```
w = New Window( "Formatted Text",
   Text Box( "This is <b>bold</b> text. This is <b><i>bold italic</i></b>
   text. This is <u>underlined</u> text.",
   <<Markup) );
```

Text Box<<Rotate Text(*direction*)

Rotates the text 90 degrees "Left" or "Right", or returns it to horizontal.

Text Box<<Set Font(*name*, *<size>*, *<style|style style...>*, *<angle>*)

Sets the font specified in the quoted *name* argument and the properties for text quoted strings. To specify more than one style, include a space between each *style* and place them in quotes.

Text Box<<Set Font Size(*n*)

Sets the font size in points for text quoted strings.

Text Box<<Set Script(*script*)

Associate a script with a text box. The script executes when the user presses Enter (or the text edit box otherwise loses focus).

Text Box<<Set Text(*quoted string*)

Sets the text in the box as specified in the quoted *string* argument.

Text Box<<Set Tip(*quoted string*)

Sets the tooltip for the text box (or a text edit box) as specified in the quoted *string* argument.

`Text Box<<Set Wrap(n)`

Set the wrap point, in pixels, in pixels (*n*).

Tree Node and Tree Box

For the following messages, `node` stands for a tree node or a reference to one and `root` stands for a tree box or a reference to one.

Caution: If you send a root node that contains one or more nodes with the `Set Node Select Script` defining a collapse message, then macOS runs the script twice. Windows doesn't run the script. This behavior on macOS doesn't just affect increments. Any script runs twice. It will print to the log twice, create a column twice, try to delete something twice, and so on.

`node<<Append(<node>)`

Inserts a referenced tree node after this node's children.

`node<<Collapse`

Closes the node. The behavior is not guaranteed if the node has a collapsed parent.

`node<<Expand`

Opens the node. The behavior is not guaranteed if the node has a collapsed parent.

`node<<Get Dimmed(<node>)`

Gets the option to dim text (decrease the opacity) for the node.

`node<<Get Font Style(<node>)`

Gets the font style for the node.

`node<<Get Tip`

Returns the tooltip for the node.

`node<<Prepend(<node>)`

Inserts a tree node before this node's children.

`node<<Remove`

Removes the given tree node and all its children from the tree display box.

`node<<Set Dimmed(`*`Boolean`*`)`

Sets the option to dim text (decrease the opacity)) for the node.

`node<<Set Font Style("Plain"|"Bold")`

Specifies the font style for the node.

`node<<Set Selected(`*`<node>`*`)`

Selects the node. The behavior is not guaranteed if the node has a collapsed parent.

`node<<Set Tip(`*`tooltip`*`)`

Sets a tooltip for the node. The *`tooltip`* argument is quoted.

`root<<Collapse(`*`<node>`*`)`

Collapses the given tree node.

`root<<Expand(`*`<node>`*`)`

Expands the given tree node.

`root<<Get Selected(`*`<node>`*`)`

Gets the currently selected tree node.

- In a single-item tree, the currently selected tree node or `Empty` is returned.
- Table 3.1 shows the results for a `Tree Box()` that contains the `MultiSelect` argument.

Table 3.1 Multi-Select Tree Results

Items Selected in Tree	Result
no items selected	empty
single item selected	list of one tree node
multiple items selected	list of selected tree nodes

`root<<Is Multiselect`

Returns 1 for a `MultiSelect` tree and 0 for a single-select tree.

`root<<Set Selected(node|{nodes}, <Boolean>)`

Selects the given tree node in the tree display box. In list of tree nodes, all nodes in the list are selected for `MultiSelect` trees. Otherwise, the first node in the list is selected. Specify the Boolean argument to indicate whether the node or nodes should be selected or unselected. The default value is 1, which selects the nodes.

Notes:

- On Windows, the `Set Selected` message expands all nodes between the selected node or nodes and the root of the tree; items that are selected deep within the tree are shown. The expansion state does not change for nodes that were previously selected.
- On macOS, the `Set Selected` message does not change the tree expansion state.

Triangulation

For the following messages, `tri` stands for a triangulation or a reference to one.

`tri<<Get N Points`

Returns the number of unique points in the triangulation.

`tri<<Get Points`

Returns the coordinates of the unique points in the triangulation.

`tri<<Get Y`

Returns the Y values of the unique points in the triangulation.

`tri<<Get N Hull Points`

Returns the number of points on the boundary of the triangulation.

`tri<<Get Hull Points`

Returns the indices of the points on the boundary of the triangulation.

`tri<<Get N Hull Edges`

Returns the number of edges on the boundary of the triangulation.

`tri<<Get Hull Edges`

Returns the indices of the edges on the boundary of the triangulation.

`tri<<Get Hull Path`

Returns the boundary of the triangulation as a path.

`tri<<Get N Triangles`

Returns the number of triangles.

`tri<<Get Triangles`

Returns the indices of the triangles in the form of an Nx3 matrix.

`tri<<Get N Edges`

Returns the number of edges in the triangulation.

`tri<<Get Edges`

Returns the indices of the edges in the form of an *N*x2 matrix.

`tri<<Subset(`*`{indices}`*`)`

Returns a triangulation resulting from the given subset of points.

`tri<<Peel`

Peel the boundary layer of a triangulation, returning a new triangulation.

Windows

`window<<Bring Window to Front`

Brings the window to the front.

`window<<Close Window(`*`<nosave>`*`)`

Closes the window. If the optional argument nosave is specified, the window (journal, report, and so forth) is closed without saving or prompting.

`window<<Get Content Size`

Returns the size of the window's contents.

`window<<Get Window Icon`

Returns the name of the window's icon.

`window<<Get Window Position`

Returns the position of the window.

`window<<Get Window Size`

Returns the size of the window.

`window<<Get Window Title`

Returns the title of the window.

`window<<Inval`

Invalidate the display box. The window updates either when the `<<Update Window` message is sent or when the operating system has time for the update. See "window<<Reshow" for another method.

`window<<Maximize Display`

Maximizes the window. Deprecated.

`window<<Maximize Window(`*`Boolean`*`)`

Maximizes the window. Deprecated.

`window<<Minimize Window(`*`Boolean`*`)`

Minimizes the window.

`window<<Move Window(`*`x`*`, `*`y`*`)`

Moves the window to the specified position.

`window<<On Close(`*`script`*`)`

Runs the script when the window is closed.

`window<<Pad Window(`*`Boolean`*`)`

Turns padding around a window's contents on (`1`) or off (`0`). The default value is off.

`window<<Print Window`

Prints the window to the default printer. Note that the Print window is not opened and user input is not required.

`window<<Reshow`

Invalidates the display box and updates the window with the new content. See `<<Inval` and `<<Update Window` messages if more control over timing of the update is required.

`window<<Save Window to Report(`*`pathname`*`, <Embed Data(`*`Boolean`*`)>)`

Saves the current report window to a JMP report file (.jrp).

`window<<Set Main Window`

Sets the specified window as the default window that appears when JMP is run.

`window<<Set Window Icon(`*`icon name`*`)`

Sets the window's icon as specified in the quoted *`icon name`* argument.

`window<<Set Window Size(`*`x`*`, `*`y`*`)`

Resizes the window.

`window<<Show Window(`*`Boolean`*`)`

1 shows the window (only if the window is not currently open). 0 hides the window. If the window is also minimized (on Windows) or docked (on macOS), showing the window restores it to the normal state and brings it to the front.

`window<<Size Window(`*`x`*`, `*`y`*`)`

Resizes the window.

`window<<Update Window`

Updates or refreshes the window holding the display box if there are invalidated regions. See "window<<Inval" and "window<<Reshow" for additional methods.

`window<<Window Class Name`

Returns the name of the window class for the display box. Valid responses include: DataTable, FormulaEditor, Starter, Journal, Launcher, Report, Dialog, DialogWithMenu, ModalDialog, FindReplace, User, Generic, ToolWindow, FindReplace, AppBuilder, and Debugger.

`window<<Zoom Window`

Resizes the window to be large enough to show all of its contents.

Dynamic Link Libraries (DLLs)

dll object<<Call DLL(*function name*, *signature*, *arguments*)

Calls the specified function in the DLL with the specified signature and arguments.

dll object<<Declare Function(*name*, *<named arguments>*)

Description

Declares the return type and argument types for the specified function so that it can be successfully invoked. You can use one of the named arguments for Convention: "STDCALL" or "PASCAL", or "CDECL". The *type* argument for Returns takes the same named arguments as Arg. The *name* argument is quoted.

Optional Named Arguments

Alias(*name*) Specifies a quoted *name* that you can include if you don't like the name encoded in the DLL.

Arg(*type*, *<description>*, *<access mode>*, *<array>*) Arg can appear multiple times, once for each argument to be sent to the function.

type is one of these keywords that specifies the argument type: "Int8", "UInt8", "Int16", "UInt16", "Int32", "UInt32", "Int64", "UInt64", "Float", "Double", "Ansiquoted string", "Unicodequoted string", "Struct", "IntPtr", "UIntPtr", or "ObjPtr".

description is a quoted string that describes the argument for reference.

access mode is an optional keyword that specifies how the argument is passed. "input" specifies that the argument is passed by value. "output" specifies that the argument is passed by address with the initial value undefined. "update" specifies that the argument is passed by reference and the value of the JSL variable is set as the initial value. The default value is "input".

array is an optional keyword. It is valid only if the *type* is specified as "Double" and the access mode is specified as either "input" or "update". Specifies that the exported function expects an array of doubles.

Convention(*calling convention*) Specifies the calling convention: "STDCALL" or "PASCAL", or "CDECL". The default value is "STDCALL". STDCALL and PASCAL are equivalent.

MaxArgs(*n*) Specifies the maximum number of arguments that can be supplied.

MinArgs(*n*) Specifies the minimum number of arguments that can be supplied.

Returns(*type*) Specifies the data type that the function returns: "Int8", "UInt8", "Int16", "UInt16", "Int32", "UInt32", "Int64", "UInt64", "Float", "Double", "Ansiquoted string", "Unicodequoted string", "Struct", "IntPtr", "UIntPtr", or "ObjPtr".

`StackOrder(`*`order`*`)` Specifies the order in which arguments are placed on the stack when calling the function. Valid values are `"L2R"` (left-to-right) and `"R2L"` (right-to-left). The default value is `"R2L"`.

`StackPop(`*`pop`*`)` Specifies how the exported function expects the stack to be cleared after the function returns. Valid values are `"CALLER"` and `"CALLEE"`. The default value is `"CALLEE"`.

`StructArg(`*`Arg(...)`*`, `*`<Arg(...)>`*`, ..., `*`<access mode>`*`, `*`<pack mode>`*`, `*`<description>`*`)` Can appear multiple times. If an exported DLL function requires that a structure argument be passed in as an argument, use `StructArg` to declare the structure members. The `Arg` arguments use the same syntax as for `Arg` arguments to `Declare Function` (one for each structure member), an `access mode` indicator and a `pack mode` indicator.

`access mode` is an optional keyword that indicates whether the struct argument should be passed by value (`input`) or by reference (`update`).

`pack mode` is an optional integer that determines how the structure is packed. Valid values are 1, 2, 4, 8, and 16. The default value is 8.

`description` is an optional, quoted string that contains a description of the structure for reference.

`dll object<<Get Declaration JSL`

Sends the declaration JSL from the DLL object to log.

`dll object<<Load DLL(`*`path`*`, <AutoDeclare(`*`Boolean`*`|"Quiet"|"Verbose")>)`

`dll object<<Load DLL(`*`path`*`, <"Quiet"|"Verbose">)`

Description

Loads the DLL from the specified path.

Required Argument

`path` A quoted path that specifies where to load the DLL.

Optional Named Arguments

`AutoDeclare(`*`Boolean`*`|"Quiet"|"Verbose")` `AutoDeclare(1)` and `AutoDeclare("Verbose")` write verbose messages to the log. `AutoDeclare("Quiet")` turns off log window messages. If you omit this option, verbose messages are written to the log.

`Quiet|Verbose` When you use `Declare Function`, this option turns off log window messaging (`"Quiet"`) or turns on log window messaging (`"Verbose"`).

`dll object<<Show Functions`

Sends the declared functions for the DLL object to the log.

`dll object<<Unload DLL`

Unloads the DLL.

HTML 5

Web Report

`webreport<<Publish(<Add Image(...)>, <Add Report(...)>, <Add Reports(...)>, <Public(`*Boolean*`)>, <Index(...), <User Name(...)>, <Password(...)|`*password function*`)>, <Prompt("IfNeeded"|"Always"|"Never")>, <URL(...)>, <Publish Data(`*Boolean*`)>, Replace(<`*id*`>, Prompt("IfNeeded"|"Always"|"Never"))>)`

Description

Publishes the web report to the JMP server.

Returns

On success, the URL of the published report is returned.

Optional Arguments

`Add Image` Inserts an image at the top of the index page. Valid formats are `"png"`, `"bmp"`, `"jpeg"`, `"jpg"`, `"tiff"`, and `"tif"`. `Title` and `Description` are optional. `Title` appears above the image. `Description` appears below the image. Use `File(`*filepath*`)` or just a quoted string. Here is an example:

```
webrpt << Add Image( File( "C:\Users\Public\JMP\Projects\WebJMP\atlas.jpg" ), Title( "Atlas" ), Description( "Holding up the world as always." ) );
```

`Add Report` Adds a report to publish within the web report.

`Public(`*Boolean*`)` Specifies whether the public has access to the report. By default, the report is private.

`Index` The name of the index page for multiple reports. You can also specify the description.

`User Name` Specifies the user name registered on the JMP server.

`Password` Specifies the user's password. You can also define a password function.

`Prompt` Displays a window in which the user types the server URL, user name, and password.

`URL` The location that you are publishing to.

`Publish Data(`*Boolean*`)` Includes the data in the HTML. Reports contains static rather than interactive images. In a public report, you might not want to share the data.

`Replace` Replaces the report. Get the URL from the address field in the browser where the page is displayed.

Images

The Scripting Index provides examples for processing images. In JMP, select **Help > Scripting Index** to view this interactive resource.

Additional resources are available from the JMP File Exchange at https://community.jmp.com/community/file-exchange.

`img<<Crop(Left(`*pix*`), Right(`*pix*`), Top(`*pix*`), Bottom(`*pix*`))`

Creates a new image from an existing image to the specified dimensions (in pixels).

`img<<Filter(`*name*`, <`*n*`>)`

Filters the image based on the specified algorithm. Filtering is useful for cleaning up noise in the image.

Note: All of the JMP image filters are supported at the operating system level. Images that are processed on Windows might differ from images processed on macOS.

Argument

`name` Specifies the quoted name of a JMP image filter. The following filters are available:

- `"Despeckle"` removes defects (that is, speckles) from a scanned or captured image (for example, scratches, dust, etc.).
- `"Edge"` identifies pixels in an image where the brightness changes sharply and darkens pixels with no sharp change. Edge detection is used to detect changes in surface, depth, material, and lighting.
- `"Enhance"` reduces the contrast between pixels in a noisy image.
- `"Median"` reduces noise (that is, the random variation) and smooths an image by comparing each pixel's brightness with its neighbors' and, if the value is very different, replaces it with the average of the neighbors' values.
- `"Negate"` creates the negative of the color or gray-scale image by changing each pixel color to its complementary color.
- `"Normalize"` changes a color image's pixels to use the full range of the file format's number system. Normalization will make the image's colors more intense.
- `"Sharpen"` reduces blur by sharpening edges of an image.
- `"Contrast"`, *n* brightens or darkens an image. A higher number (>0.0) brightens an image; a lower number (<0.0) darkens an image.

- "Gamma", *n* corrects the image visual display (brightness and intensity) to account for differences in monitor hardware. A higher number (> 1.0) lightens the image; a lower number (< 1.0) darkens the image.
- "Reduce Noise", *n* reduces the random variation (or noise) that occurs with higher ISO sensitivity or longer exposure times.
- "Gaussian Blur", radius, sigma reduces image noise and detail creating a smoother image. Radius is equal to the blur radius around each pixel and sigma is the standard deviation of the Gaussian distribution. Gaussian blur is commonly used when resizing or performing edge detection.

`img<<Flip Both`

Flips the image from left to right and top to bottom.

`img<<Flip Horizontal`

Flips the image from left to right.

`img<<Flip Vertical`

Flips the image from top to bottom.

`img<<Get EXIF`

Returns EXIF data from the image (such as the shutter speed and aperture value) in an associative array.

`img<<Get N Frames`

Description

Returns the number of frames in a multi-frame TIF or animated GIF file, where the number of frames begins with frame 0.

Example

The following example places a four-frame TIF file in a new window and shows the image that is in the first frame.

```
img = New Image( "$DOWNLOADS/Multiframe.tif" );
nframes = img << Get N Frames(); // return 4
img << Set Current Frame( 1 ); // show image 1
win = New Window( "Multi-Frame TIFF", img );
```

`img<<Get Size`

`img<<Size`

Returns a list containing the width and height (in pixels) of the image.

`img<<Rotate(`*degrees*`)`

Rotates the image by the specified number of degrees.

`img<<Save Image(`*path*`)`

Saves the image to the quoted *path*.

`img<<Scale(`*scale*`|`*xscale*`, `*yscale*`)`

Resizes the image by the specified dimensions. Provide one argument to resize both the width and height. Provide two arguments to resize the width and height separately.

Examples

```
img = New Image( "$SAMPLE_IMAGES/tile.jpg" );
xs = 2;
img << Scale( xs );
New Window( "Tilex 2", img );

img = New Image( "$SAMPLE_IMAGES/tile.jpg" );
img << Scale( 2, 0.5 ); // scale image width by 2 and height by 1/2
New Window( "Tile squished", img );
```

Notes

Using `Scale` is an alternative to getting the size of the image, multiplying by the scale factor, and then setting the size.

`img<<Set Current Frame`

Description

Sets the frame that shows in a multi-frame TIFF or animated GIF file. Specify 0 through the number of frames minus 1. For example, with four frames, you can specify frame 0 through frame 3.

See Also

"img<<Get N Frames"

`img<<Set Size(`*width*`, `*height*`)`

Resizes the image to the specified dimensions (in pixels). To scale the image proportionally, specify a width and height that correspond to the aspect ratio in the original image.

`img<<Transparency(`*fraction*`)`

Sets the transparency for the image where the fraction is between 0.0 (full transparency) to 1.0 (no transparency).

JMP Applications

The JMP Application Builder and JMP Dashboard Builder use the same infrastructure to design and execute applications and dashboards. Because a dashboard is a special form of an application, this section uses the term *application* to describe how both dashboard and application objects use scripting.

See the Scripting Index in the JMP Help menu for examples.

JMP App

The `JMP App` object is the main controller for JMP applications built by Application Builder or Dashboard Builder. Scripts both inside and outside of a JMP application can use a `JMP App` object.

A JMP application can have one of three states: initial (with no editor, and the application is not running), running, or editing. A `JMP App` object only exists in one state at a time; if you are editing a JMP Application and choose to run it, a copy of the JMP application is created before it is run.

`app<<Combine Windows({`*reports or data tables*`})`

Description

Combines the given list of platform reports or data tables into a new module. The application should be in the initial state when this message is sent.

`app<<Debug`

Invokes the JSL Debugger on a JMP application. The application script will run first. The Debugger then breaks as each module is created, invoking the module scripts. In the Debugger, set breakpoints to debug the scripts that are associated with the application or modules.

`app<<Edit`

Starts Application Builder on a JMP application that is in the initial state. There is no editor, and the application is not running.

`app<<Get Modules`

Gets the list of modules associated with an application. In Application Builder, each module corresponds to a tab in the workspace, which describes the layout and behavior for one type of window in the application.

`app<<Get Namespace`

The `JMP App()` object automatically creates an anonymous namespace for the variables created within the application script. Use this message to get a handle to this namespace to inspect or modify variables. There is a default symbol in this namespace called `thisApplication`, which holds a reference to the application itself.

`app<<Get Windows`

Gets a list of all windows created as instances of JMP app modules. Some modules might create more than one instance. All windows might not exist at the same time, so the number of windows might vary and might differ from the number of modules.

`app<<Open File(`*`path`*`)`

Resets the state of an existing application from a the .jmpapp or .jmappsource file. *`path`* is quoted.

`app<<Relaunch Analysis`

Creates a new copy of a running application and runs the new instance.

`app<<Run`

Runs the application. The application script runs first, and depending on settings, one or more JMP app module instance objects might be created automatically.

`app<<Save Script to Add-In`

`app<<Save Script to Data Table`

`app<<Save Script to Journal`

`app<<Save Script to Script Window`

Saves the script for the application to the given destination. An application script consists of a `JMP App()` object that contains the definition for the application. Scripts saved to an add-in, data table, or journal include a `Run` message to run the application. A script saved to the script window includes an `Edit` message to open Application Builder.

JMP App Module

A JMP application module is a definition of the display box layout and behavior for a single component in a JMP application or dashboard. Depending on the module type, the component might represent a window in the application or just part of a window.

module<<Create Instance

Use `Create Instance` within a `JMP App()` or `JMP App Module()` script to create an instance of a JMP app module. By default, one instance of each JMP application module is created when an application is run. For more complex applications with multiple windows, such as a launcher and report combination, it might be necessary to change the default settings and control how the module instance is created.

module<<Get Instance

Returns a handle to the application that owns a module.

JMP App Module Instance

The JMP application module instance is a running realization of a JMP app module, a window on the screen, or a collection of display box elements that can be inserted into another window.

inst<<Create Objects

Appears in the default template for a JMP app module script. This message controls the point at which the display and window for a module instance are created. The message appears in the script so that the script writer can choose to do certain setup before the objects are created. One example of this setup is for a parameterized application.

inst<<Get Box

Returns a handle to the top-most display box associated with a module instance. This might be useful to issue display or window commands, such as the `Save to PDF` or `Close Window` messages.

inst<<Get Namespace

Like `JMP App()`, each JMP application module instance also creates an anonymous namespace for all variables created in the module script. The namespace also includes all the variables that represent the display boxes in the module. This namespace contains a default symbol named `thisModuleInstance` that refers to itself.

`inst<<Get User Data`

`inst<<Set User Data`

Stores and retrieves a JSL value in the JMP application module instance. The value could be a number, quoted string, list, associative array, or other JSL type that is returned with the `Type()` function.

MATLAB

The MATLAB interfaces are scriptable using a MATLAB connection object. Use the MATLAB Connect() JSL function to obtain a scriptable MATLAB connection object.

`mlconn<<Control(<Echo(`*`Boolean`*`)>, <Visible(`*`Boolean`*`)>)`

Controls the execution of MATLAB.

Returns

None.

Optional Global Arguments

`Echo(`*`Boolean`*`)` Echo MATLAB source lines to the JMP Log window.

`Visible(`*`Boolean`*`)` Determine whether to show or hide the active MATLAB workspace.

`mlconn<<Disconnect()`

Description

Disconnects this MATLAB integration interface connection.

`mlconn<<Execute({`*`list of inputs`*`}, {`*`list of outputs`*`}, `*`mCode`*`, <Expand(`*`Boolean`*`)>, <Echo(`*`Boolean`*`)>)`

Submits MATLAB code to the active global MATLAB integration interface connection given a list of inputs and upon completion a list of outputs are retrieved.

Returns

0 if successful, otherwise nonzero.

Required Arguments

`{`*`list of inputs`*`}` Positional, name list. List of JMP variable names to be sent to MATLAB as inputs.

`{`*`list of outputs`*`}` Positional, name list. List of JMP variable names to be retrieved from MATLAB as outputs.

`mCode` Positional, quoted string. The MATLAB code to submit.

Optional Named Arguments

Expand(*Boolean*) Performs an Eval Insert on the MATLAB code prior to submission.

Echo(*Boolean*) Echos MATLAB source lines to the JMP Log window. Default is true.

---

mlconn<<Get Graphics(*format*)

Gets the last graphic object written to the MATLAB graph display window. The graphic object can be returned in several graphic formats.

Returns

JMP Picture object.

Optional Argument

format Positional. The quoted format the MATLAB graph display window contents are to be converted to. Valid formats are "png", "bmp", "jpeg", "jpg", "tiff", and "tif".

---

mlconn<<Get Version()

Gets the current version of the installed MATLAB.

Returns

Matrix, returns a vector of length 3 containing the MATLAB version number.

---

mlconn<<Get(*name*)

Description

Gets a named variable from MATLAB to JMP.

Returns

Value of named variable.

Required Argument

name The name of a JMP variable to be retrieved from MATLAB.

---

mlconn<<Is Connected()

Description

Determines whether connection is active.

Returns

1 if connected, otherwise 0.

---

mlconn<<JMP Name To MATLAB Name(*jmp name*)

Description

Maps a JMP variable name to its corresponding MATLAB variable name using MATLAB variable name naming rules.

Returns

A quoted string, a mapped MATLAB name.

Required Argument

`jmp name` Positional. The name of a JMP variable to be sent to MATLAB.

`mlconn<<Send(name, <named arguments>)`

Description

Sends the named variable from JMP to MATLAB.

Returns

0 if successful, otherwise nonzero.

Required Argument

`name` Positional. The name of a JMP variable to be sent to MATLAB.

Named Arguments

The following arguments are for data tables only:

`Selected(Boolean)` Send selected rows from the referenced data table to MATLAB.

`Excluded(Boolean)` Send only excluded rows from the referenced data table to MATLAB.

`Labeled(Boolean)` Send only labeled rows from the referenced data table to MATLAB.

`Hidden(Boolean)` Send only hidden rows from the referenced data table to MATLAB.

`Colored(Boolean)` Send only colored rows from the referenced data table to MATLAB.

`Markered(Boolean)` Send only markered rows from the referenced data table to MATLAB.

`Row States (Boolean, <named arguments>)` Send row states from referenced data table to MATLAB by adding an additional data column named "RowState". Create multiple selections by adding together individual settings. The row state consists of individual settings with the following values:

- Selected = 1
- Excluded = 2
- Labeled = 4
- Hidden = 8
- Colored = 16
- Markered = 32

Row State Optional Named Arguments

The following optional, named Row States arguments are supported:

`Colors(Boolean)` Send row colors. Adds additional data column named "RowStateColor".

Markers(*Boolean*) Send row markers. Adds additional data column named "RowStateMarker".

---

mlconn<<Submit(*mCode*, *<named arguments>*)

Description

Submits MATLAB code to the active global MATLAB integration interface connection.

Returns

0 if successful, otherwise nonzero.

Required Argument

mCode Positional quoted string. The MATLAB code to submit.

Named Arguments

Expand(*Boolean*) Perform an Eval Insert on the MATLAB code prior to submission.

Echo(*Boolean*) Echo MATLAB source lines to the JMP log. The default is true.

---

mlconn<<Submit File(*path*)

Description

Submits statements to MATLAB using a quoted *path*.

Returns

0 if successful, otherwise nonzero.

Arguments

path Positional quoted string. The path to a file containing the MATLAB source lines to be executed.

Namespaces

---

ns<<Contains(*string*)

Returns 1 or 0, depending on whether the specified quoted string exists within the namespace.

---

ns<<Delete Namespace

Removes this namespace from the internal global list.

To delete variables in the namespace, use the Remove(*variable name*) message.

---

ns<<First

Returns a quoted string that contains the first variable name used within the namespace.

`ns<<Get Contents`

Returns a list of key-value pairs, which are each enclosed in a list. Each key is a quoted string that contains a variable name, and each value is the unevaluated expression that the variable contains.

`ns<<Get Keys`

Returns a list of variable names.

`ns<<Get Name`

Returns the name of this namespace.

`ns<<Get Value(`*`variable name`*`);`

Returns the unevaluated expression that the quoted *`variable name`* contains in this namespace.

`ns<<Get Values`

Returns a list of unevaluated expressions that each variable in the namespace contains.

`ns<<Get Values({`*`variable name1, variable name2, ...`*` });`

Returns a list of unevaluated expressions that each quoted variable in the namespace specified in the list argument contains. If a requested variable name is not found, an error is returned.

`ns<<Insert(`*`variable name`*`, `*`expr`*`);`

Inserts into this namespace a quoted variable named *`variable name`* that holds the expression *`expr`*.

`ns<<Lock Namespace(<`*`variable name, ...`*`>)`

Locks all specified variables in the namespace and prevents quotes variables from being added or removed. If no variables are specified, all variables in the namespace are locked.

`ns<<N Items`

Returns the number of variables contained in the namespace.

`ns<<New Namespace(name, <{list of expressions}>)`

Creates a namespace where all functions and variables created are defined only within the optional quoted *name* argument.

`ns<<Next(variable name)l`

Returns the name of the variable that follows the specified quoted variable.

`ns<<Remove(variable name, ...)`

Removes the specified quoted variable or list of variables.

`ns<<Show Contents`

Shows the contents of a namespace in the log.

`ns<<Unlock Namespace(variable name, ...);`

Unlocks the specified quoted variables in the namespace. If no variables are specified, all variables are unlocked.

Platforms

`obj<<Action`

Evaluates expressions. Useful for quoted stringing together multiple platforms interrupted by user input.

`obj<<Automatic Recalc`

Redoes the analysis automatically for exclude and data changes. If automatic recalc is on, you should use `Wait(0)` commands to let the triggers take effect and do the recalculation.

Note: Not supported on all platforms.

`obj<<Bring Window To Front`

Brings the current window to the front.

`obj<<Close Window`

Closes window identified by *obj*, typically a platform surface.

`obj<<Column Switcher(`*`default column, {column1, column2, ...}`*`)`

Adds a control panel to a platform for switching variables.

`obj<<Copy ByGroup Script`

Creates a script to produce this analysis containing By variables and place it on the clipboard.

`obj<<Copy Script`

Creates a script to produce this analysis and place it on the clipboard.

`obj<<Data Table Window`

Makes the associated data table window active (front-most).

`obj<<Get Data Table`

Returns a reference to the data table.

`obj<<Get Script`

Returns script to reproduce the analysis as an expression in the log.

`obj<<Get Script With Data Table`

Creates a script to reproduce the analysis, specifically referencing the source data table, and returns it as an expression in the log.

`obj<<Get Timing`

Times the launch of the platform and returns it in the log.

`obj<<Get Web Support`

Returns the score for the display tree that is about to be saved as interactive HTML. Possible values are -1 (unsupported), 0 (supported), and 1 (supported). If the score does not equal -1, interactive HTML is supported and the `Save Interactive HTML` message can be used.

`obj<<Get Window Position`

Gets the position of the window. Returns an ordered pair.

obj<<Get Window Size

Gets the window size, in pixels. Returns an ordered pair.

obj<<Ignore Platform Preferences(*Boolean*)

Ignores the current settings of the platform's preferences. The message is ignored when sent to the platform after creation.

obj<<Journal Window

Appends the contents of the window to the journal.

obj<<Local Data Filter

Filters data to specific groups or ranges, but stays local to the platform.

obj<<Maximize Window

Maximizes the window. Equivalent to pushing the maximize button in the corner of the window. This message takes an optional Boolean argument:

```
// maximize the window:
obj<<Maximize Window(1)
// restore the window:
obj<<Maximize Window(0)
```

obj<<Minimize Window

Minimizes the window. Equivalent to pushing the minimize button in the corner of the window. This message takes an optional Boolean argument:

```
// minimize the window
obj<<Minimize Window( 1 )
// restore the window
obj<<Minimize Window( 0 )
```

obj<<Move Window(*x*, *y*)

Moves the window to the (*x*, *y*) location on your screen.

obj<<Print Window

Sends the selected window to the printer.

obj<<Redo Analysis

Reruns the analysis with the same options.

`obj<<Redo ByGroup Analysis`

Reruns the same analysis involving By groups.

`obj<<Relaunch Analysis`

Returns to the launch window for this analysis.

`obj<<Relaunch ByGroup`

Returns to the launch window for this analysis involving By groups.

`obj<<Remove Column Switcher`

Removes all Column Switchers that were added to the platform.

`obj<<Remove Local Data Filter`

Removes all Local Data Filters that were added to the platform.

`obj<<Report`

`Report(`*`obj`*`)`

Returns a display box reference for the report in the platform window.

`obj<<Report View`

Determines the level of detail visible in a platform report. Full shows all detail and Summary shows only select content, dependent upon the specific platform. For customized behavior, use the `Set Summary Behavior` message with display boxes.

`obj<<Save ByGroup Script to Data Table(<`*`name`*`>, <Append Suffix(`*`Boolean`*`)>, <Prompt(`*`Boolean`*`)>, <Replace(`*`Boolean`*`)>)`

Description

Creates a table script to produce the analysis involving By variables and saves it as a table script in the data table.

Optional Arguments

`name` The name of the script. *`name`* is quoted. If omitted, the platform names the script. For example, in Tabulate, the script is named "Tabulate". In Bivariate, the script might be named "Bivariate of height by weight" to reflect the platform and column names.

`Append Suffix(`*`Boolean`*`)` If true, appends a numerical suffix to the script name. This suffix differentiates the script from an existing script with the same name.

`Prompt(`*`Boolean`*`)` If true, prompts the user to specify a script name.

Replace(*Boolean*) If true, replaces an existing script with the same name.

`obj<<Save ByGroup Script to Journal`

Creates a table script to produce the analysis involving By variables and adds a button to the journal containing this script.

`obj<<Save ByGroup Script to Script Window`

Creates a script to produce the analysis involving By variables and appends it to the current Script window.

`obj<<Save Script for All Objects`

Saves script to reproduce all analyses found within the object's window in the Script Journal window.

`obj<<Save Script for All Objects to Data Table`

Saves a script for all report objects to the current data table. The script is named after the platform unless you specify the script name in quotes.

```
obj << Save Script for All Objects To Data Table("My Script")
```

`obj<<Save Script to Data Table`

Saves script to reproduce analysis as a property in the associated data table.

`obj<<Save Script to Journal`

Creates a script to produce the analysis and adds a button to the journal containing this script.

`obj<<Save Script to Report`

Saves script to reproduce analysis as a text box at the top of the report.

`obj<<Save Script to Script Window`

Saves a script to reproduce analysis in the Script Journal.

`obj<<Scroll Window(`*x*`, `*y*`)`

`obj<<Scroll Window({`*x*`, `*y*`})`

Scrolls the window *x* pixels to the right and *y* pixels down from the current position. Negative coordinates go left and up. If the coordinates are a list in braces { }, they are

absolute coordinates. The window scrolls to the point *x* pixels from the left and *y* pixels from the top.

`obj<<SendToReport`

Used with the Dispatch function to customize the appearance of a report.

`obj<<SendToByGroup`

Sends messages to open platforms or turn on platform features to each level of a by-group.

`obj<<Show Window(`*Boolean*`)`

1 shows the window (brings it to the front). 0 hides the window. If the window is also minimized (on Windows) or docked (on macOS), showing the window restores it to the normal state and brings it to the front.

`obj<<Size Window(`*x*`, `*y*`)`

Resizes the window to *x* pixels wide by *y* pixels high.

`obj<<Title(`*new title*`)`

Sets the quoted title of the platform.

`obj<<Top Report`

Returns a reference to the top display box in the report. Useful for By groups or other cases when several platform reports are in one window.

`obj<<View Web XML`

Returns the XML used to create the interactive HTML report. The XML code appears in the log.

`obj<<Zoom Window`

Resizes the window to be large enough to show all of its contents.

Bubble Plot

---

bp<<Set
Shape("Circle"|"Triangle"|"Square"|"Diamond"|"Arrow"|"Custom")

Description

Sets the shape for the bubble. When specifying a custom shape, use the `Set Custom Path` message to specify the path for the bubbles.

See Also

"bp<<Set Custom Path(path matrix|path text)".

---

bp<<Set Custom Path(*path matrix*|*path text*)

Sets a path for custom bubbles.

Arguments

path matrix An *N*x3 matrix.

path text A quoted string that contains SVG code.

See Also

"bp<<Set Shape("Circle"|"Triangle"|"Square"|"Diamond"|"Arrow"|"Custom")".

DOE

---

obj<<Get Prediction Variances

Description

Returns a vector of the MC variances used for the FDS plot.

Example

```
dt = Open( "$SAMPLE_DATA/Design Experiment/Bounce Data.jmp" );
d = DOE( Evaluate Design, X( :Silica, :Sulfur, :Silane ), Y( :Stretch ) );
d << Get Prediction Variances;
```

---

obj<<Set Number of FDS Points()

Description

Enables you to specify the number of runs used to generate an FDS plot.

Example

```
dt = Open( "$SAMPLE_DATA/Design Experiment/Bounce Data.jmp" );
d = DOE( Evaluate Design, X( :Silica, :Sulfur, :Silane ), Y( :Stretch ) );
d << Set Number of FDS Points( 20000 );
```

Partition

`obj<<Initial Splits(`*`condition`*`, `*`{left}`*`, `*`{right}`*`)`

Description

Describes the splits that are performed.

Example

```
dt = Open( "$SAMPLE_DATA/Car Poll.jmp" );
obj = Partition(
  Y( :country ),
  X( :sex, :marital status, :age, :type, :size ),
  Method( "Decision Tree" ),
  Initial Splits( :size == {"Large"}, {}, {:size == {"Medium"}} )
);
```

Notes

The condition is for the left side and is either `[name compareoperator value]` or `[name == list of values]`. The left is an empty list if the right has splits. Omit the right side if there are no splits. The left and right continue recursively in this format.

Response Screening

`obj<<Get PValues`

Returns a reference to a PValues table.

`obj<<Save PValues`

Stores the *p*-values in an output data table.

`obj<<Save Compare Means`

Stores the means comparisons in an output data table.

`obj<<Save Mean`

Stores the means in an output data table.

`obj<<Save Outlier Indicator`

Saves Outlier Indicator for each fit.

`obj<<Save Std Residuals`

Saves the residual formula for each fit.

obj<<Select Columns

Select columns in the original table corresponding to selected rows in this table.

Tabulate

obj<<Display Column Width(*<Data Column(Column Table(n),<column name path>), Row Label(Row Table(n), <column name path>)>, <width>*)

Returns or sets the display pixel width of a column in a Tabulate table.

Required Argument

Row Label Use Row Table and heading for columns in the row labels area.

Optional Arguments

Data Column Use Column Table and column references to define columns in the main body of the table.

column name path Specifies the Column Table or Row Table (both quoted), and the series of column headings that traces the path of the column. **Note:** Column Table or Row Table can be omitted if the table referenced is the first table.

width Specifies the pixel width of the column.

Examples

```
dt = Open( "$SAMPLE_DATA/Car Poll.jmp" );
obj = dt << Tabulate(
  Add Table(
     Column Table(
        Grouping Columns( :sex, :marital status ),
        Analysis Columns( :age ),
        Statistics( Sum, "% of Total" )
     ),
     Row Table( Grouping Columns( :type ) ),
     Row Table( Grouping Columns( :country, :size ) )
  )
);
Wait( 3 ); // for demonstration purposes
obj << Display Column Width( Row Label( Row Table( 2 ), "country" ), 150 );
Wait( 3 ); // for demonstration purposes
obj << Display Column Width(
  Data Column(
     Column Table( 1 ),
     "sex",
     "Female",
     "marital status",
     "Married",
     "age",
```

```
        "Sum"
    ),
    150
);
```

Python Integration Messages

The Python interfaces are also scriptable using a Python connection object. A scriptable Python connection object can be obtained using the `Python Connect()` function. See "Python Connect(<Echo(Boolean),> <Path(path),> <Use Python Version(string),> <Python System Path(list)>)".

`pythconn<<Control(<Interactive(`*`Boolean`*`)>|<Echo(`*`Boolean`*`)>)`

Description

Controls the execution of Python.

Optional Named Arguments

`Interactive(`*`Boolean`*`)` Enables interactive mode in the Python matplotlib package. Determines whether the graphics window is released or closed when graphics rendering is complete.

`Echo(`*`Boolean`*`)` Global argument. Prints the Python source lines to the JMP log. The default value is true.

`pythconn<<Disconnect`

Disconnects the Python integration interface connection.

`pythconn<<Execute(`*`{list of inputs}, {list of outputs}, Python code, <named arguments>`*`)`

Description

Submits Python code to the active global Python integration interface connection given a list of inputs. On completion, returns a list of outputs.

Returns

Returns 0 if successful and 1 otherwise. The results are returned using the list of outputs. Given each element of the JMP output list, the corresponding Python variable value is returned.

Positional Arguments

`{list of inputs}` A list of JMP variable names to be sent to Python as inputs.

`{list of outputs}` A list of JMP variable names to be retrieved from Python as outputs.

Python code The quoted Python code to submit.

Optional Named Arguments

See "pythconn<<Submit(Python code, <Expand(Boolean)>, <Echo(Boolean)>)".

---

pythconn<<Get(*name*)

Description

Gets a named variable from Python.

Returns

Returns the value of the named variable.

Argument

name The name of the JMP variable to be received from Python. The argument can represent any of the following Python data types: numeric, quoted string, matrix, list, or data frame.

---

pythconn<<Get Graphics(*format*)

Description

Gets the last graphics object written to the Python graph display window in the specified graphics format. The graphics object can be returned in several different graphic formats.

Returns

Returns a JMP picture object.

Argument

format The format that the Python graph display window contents are to be converted to. Valid formats are PNG, BMP, JPEG, JPG, TIFF, and TIF.

---

pythconn<<Get Version

Description

Gets the current version of the Python installation.

Returns

Returns a list of length 5 that contains the five components of the version number: major, minor, micro, releaselevel, and serial. The releaselevel value is a quoted string.

---

pythconn<<Is Connected

Description

Determines if the connection is active.

Returns

Returns 1 if connected and 0 otherwise.

pythconn<<JMP Name to Python Name(*name*)

Description

Maps a JMP variable name to its corresponding Python variable name using Python variable name naming rules.

Argument

name The name of the JMP variable to be sent to Python. Some variable names allowed by JMP are not allowed by Python. When you send using these variables from JMP to Python (the Send message), their names get changed. Use JMP Name to Python Name to determine what the variable name was changed to.

pythconn<<Send(*name*)

Description

Sends a named variable from JMP to Python.

Returns

Returns 0 if successful and non-zero otherwise.

Argument

name The name of the JMP variable to be sent to Python.

pythconn<<Submit(*Python code*, <Expand*(Boolean)*>, <Echo*(Boolean)*>)

Description

Submits Python code to the active global Python integration interface connection.

Returns

Returns 0 if successful and 1 otherwise.

Required Arguments

Python code The quoted Python code to submit.

Optional Arguments

Expand(*Boolean*) Performs an Eval Insert() on the Python code before submission.

Echo(*Boolean*) Prints the Python source lines to the JMP log. The default value is true.

pythconn<<Submit File(*path*)

Description

Submits statements to Python using the quoted path name.

Argument

path The quoted path to the file that contains the Python source lines to be executed.

R Integration Messages

The R interfaces are also scriptable using an R connection object. A scriptable R connection object can be obtained using the `R Connect()` function.

`rconn<<Control(Interrupt|Async(`*`Boolean`*`)|Echo(`*`Boolean`*`))`

Changes the control options for R. If `Async` is set to true (`1`) for `R Submit()`, this message immediately stops the execution of the R code that was submitted.

`rconn<<Disconnect()`

Disconnects this R connection.

`rconn<<Is Connected()`

Returns 1if the R connection is active, 0 otherwise.

`rconn<<Send File(`*`name`*`, <named arguments>)`

Send the specified JMP variable to R.

Returns

0 if successful, nonzero otherwise.

Required Argument

`name` A quoted string contains the name of a JMP variable to send to R.

Optional Named Arguments for Data Tables

`Selected(`*`Boolean`*`)` If true, sends only the selected rows from the referenced data table to R.

`Excluded(`*`Boolean`*`)` If true, sends only the excluded rows from the referenced data table to R.

`Labeled(`*`Boolean`*`)` If true, sends only labeled rows from the referenced data table to R.

`Hidden(`*`Boolean`*`)` If true, sends only hidden rows from the referenced data table to R.

`Colored(`*`Boolean`*`)` If true, sends only colored rows from the referenced data table to R.

`Markered(`*`Boolean`*`)` If true, sends only markered rows from the referenced data table to R.

`Row States(`*`Boolean`*`, <named arguments>)` Includes a Boolean argument and optional named arguments. Sends row state information from the referenced data table to R by adding an additional data column named “RowState”. The row state value consists of individual settings with the values shown in Table 3.2.

Table 3.2 Row States

Multiple row states are created by adding together individual settings.	Selected = 1 Excluded = 2 Labeled = 4 Hidden = 8 Colored = 16 Markered = 32
Arguments	`Colors(`*`Boolean`*`)` (Optional) If true, sends row colors and adds an additional data column named "RowStateColor". `Markers(`*`Boolean`*`)` (Optional) If true, sends row markers and adds an additional data column named "RowStateMarker".

`rconn<<Send(`*`name, <R Name(name)>`*`)`

Sends the quoted JMP data file to R. The *`name`* argument can represent any of the following data types: numeric, quoted string, matrix, list, or data table.

`rconn<<Get(`*`name`*`)`

Returns data from R. The *`name`* argument can represent any of the following data types: numeric, quoted string, matrix, list, or data table.

Returns

The value of the specified variable.

Arguments

`name` Specifies the quoted name of a JMP variable to be retrieved from R.

`rconn<<Get Graphics(`*`type`*`)`

Gets the last graphics object written to the R graph display window. The graphics object can be returned in different graphic formats.

Returns

A JMP picture object.

Required Argument

`type` The format the R graph display window contents are converted to. Valid formats are `"png"`, `"bmp"`, `"jpeg"`, `"jpg"`, `"tiff"`, and `"tif"`.

`rconn<<Submit(R code, Expand(Boolean), Echo(Boolean))`

Submits the quoted R code.

Returns

0 if successful, nonzero otherwise.

Required Argument

`code` Specifies the quoted R code to submit.

Optional Named Arguments

`Expand(Boolean)` Performs an `Eval Insert()` on the R code before submitting the code.

`Echo(Boolean)` Echoes the R source lines to the JMP log. The default value is true.

`Rconn<<Submit File(path)`

Submits statements to R using the file in the quoted *path*.

Arguments

`path` Specifies the quoted path to the file that contains R code to be executed.

`rconn<<Execute({list of inputs}, {list of outputs}, R code, <named arguments>)`

Submits the quoted R code to the R connection using the list of inputs. Upon completion, a list of outputs is returned.

Returns

0 if successful, nonzero otherwise.

Required Arguments

R code Specifies the quoted R code to submit.

`{list of inputs}` List of JMP variable names to be sent to R as inputs.

`{list of outputs}` List of JMP variable names to be retrieved from R as outputs.

Optional Named Arguments

See "rconn<<Submit(R code, Expand(Boolean), Echo(Boolean))".

`rconn<<Control(<Echo(Boolean)>)`

Controls the execution of R.

Returns

Void.

Optional Named Argument

`Echo(Boolean)` Echoes the R source lines to the JMP log.

`rconn<<Get Version()`

Gets the current version of R that is installed.

Returns

A vector of length 3 containing the R version number.

`rconn<<JMP Name To R Name(`*name*`)`

Maps a quoted JMP Name to its corresponding R Name using R variable name naming rules.

Returns

A quoted string that contains the quoted R name.

Arguments

name A quoted string that specifies the name of a JMP variable to be sent to R.

SAS Integration Messages

Metadata Server Objects

`metaserver<<Disconnect()`

Description

Disconnects the metadata server.

Returns

Void.

`metaserver<<Get Display Name()`

Description

Gets the display name of the metadata server.

Returns

A quoted string.

`metaserver<<Get Host Name()`

Description

Gets the host (machine) name of the metadata server.

Returns

A quoted string.

`metaserver<<Get Port()`

Description

Gets the port used for the metadata server connection.

Returns

An integer.

`metaserver<<Get User Identity()`

Description

Gets the identify of the connected user as defined in metadata.

Returns

A quoted string.

`metaserver<<Get User Name()`

Description

Gets the user name (login ID) that was used for the metadata server connection.

Returns

A quoted string.

SAS Server Objects

`sasconn<<Assign Libref(`*`libref`*`, `*`path`*`, `*`engine`*`, `*`engine options`*`)`

Description

Assign a SAS libref on this SAS server connection.

Returns

Void.

Arguments

See “SAS Assign Lib Refs("libref", "path", <"engine">, <"engine options">)”.

`sasconn<<Cancel Submit()`

Description

Cancels the currently running SAS Submit for this server that is presumably running asynchronously.

Returns

1 if a running submit was found and canceled; 0 otherwise.

`sasconn<<Clear Log History()`

Description

Clears the SAS Log history for this server.

Returns

Void.

`sasconn<<Clear Output History()`

Clears the SAS Output history for this server.

`sasconn<<Connect(<User Name(`*name*`)>, <Password(`*password*`)>, <Prompt("Always"|"Never"|"IfNeeded")>)`

Description

Attempt to reconnect a SAS server connection object that has become disconnected.

Returns

1 if the connection was successful, 0 otherwise.

Optional Named Arguments

`User Name(`*name*`)` Specifies the quoted user name for the connection.

`Password(`*password*`)` Specifies the quoted password for the connection.

`Prompt` A quoted keyword. `"Always"` means always prompt before attempt to connect. `"Never"` means never prompt even if the connection attempt fails (just fail with an error message going to the log), and `"IfNeeded"` (the default) means prompt if the attempt to connect with the given arguments fails (or is not possible with the information given).

`sasconn<<Deassign Libref(`*libref*`)`

Description

De-assign the quoted SAS libref on this SAS server connection.

Returns

Void.

Arguments

`libref` Specifies the quoted library reference.

`sasconn<<Disconnect()`

Description

Disconnect this SAS server connection.

Returns

Void.

sasconn<Does Module Exist(*module name*)

Description

Determines whether the specified SAS module exists in the SAS installation represented by the SAS connection. This can be helpful in determining whether certain SAS products are installed. The SAS DATA Step function MODEXIST is used to determine module existence.

Returns

1 if the specified module is found to exist, 0 if it does not exist.

Argument

`module name` Specifies the quoted SAS module, the existence of which should be checked. Do not include any extension.

sasconn<<Export Data(*dt*, *library*, *dataset*, *<named arguments>*)

Description

Exports a JMP data table to the specified SAS data set in the specified library on the active SAS server connection.

Returns

1 if the data table was exported successfully; 0 otherwise.

Optional Named Arguments

See "SAS Export Data(dt, "library", "dataset", <named_arguments>)".

sasconn<<Get Data Sets(*libref*)

Description

Returns a list of the data sets defined in a SAS library on this SAS server connection.

Returns

A list of quoted strings.

Arguments

libref Specifies the quoted SAS libref or friendly library name associated with the library for which the list of defined SAS data sets will be returned.

sasconn<<Get Error Count()

Description

Gets the count of the number of errors encountered in the previous SAS Submit.

Returns

An integer.

`sasconn<<Get File(source, dest)`

Description

Downloads a file from this SAS server connection.

Returns

Void.

Arguments

See "SAS Get File("source", "dest", "encoding")".

`sasconn<<Get File Names(fileref)`

Description

Gets a list of filenames found in the quoted *fileref* on this SAS server connection.

Returns

A list of quoted strings.

Arguments

fileref A quoted string that contains the name of *fileref* from which to retrieve filenames.

`sasconn<<Get File Names In Path(path)`

Description

Gets a list of filenames found in the quoted *path* on the current SAS server connection.

Returns

A list of quoted strings.

Arguments

path The quoted directory path on the server from which to retrieve filenames.

`sasconn<<Get File Refs()`

Description

Gets a list of the currently defined SAS filerefs on this SAS server connection.

Returns

A list of quoted strings.

`sasconn<<Get Librefs(<named arguments>)`

Description

Gets a list of the currently defined SAS librefs on this SAS server connection.

Returns

A list of quoted strings.

Optional Named Arguments

See "SAS Get Lib Refs(<named arguments>)".

`sasconn<<Get Log()`

Description

Retrieves the SAS Log from the last SAS Submit from this SAS server connection.

Returns

A quoted string.

`sasconn<<Get Option Name()`

Description

Queries SAS for the value of a SAS option variable.

Returns

A quoted string.

Example

The following script iterates through the define variables and prints out the values:

```
option_names = sasconn << Get Option Names();
For(i=1, i <= N Items(option_names), i++,
   option_value = sasconn << Get Option Value (option_names[i]);
   output = option_names[i] || "=" || char(option_value) || "/!n";
   Write(output);
);
```

`sasconn<<Get Output()`

Description

Retrieves the listing output from the last submission of SAS code to this SASServer object.

Returns

A quoted string.

`sasconn<<Get Results()`

Description

Retrieves the results of the previous SAS Submit as a scriptable object, which allows significant flexibility in what to do with the results.

Returns

A SAS Results Scriptable Object.

`sasconn<<Get Submit Status()`

Description

Gets the current status of a SAS Submit for this server that is presumably running asynchronously.

Returns

1 if the submit has not started; 2 if the submit is running; 3 if the submit has been canceled; 10 if the submit has completed successfully; 11 if the submit has completed with errors.

`sasconn<<Get Var Info(`*`libref`*`, `*`dataset`*`, <Password(`*`password`*`)>)`

Description

Returns information about the variables the specified SAS data set.

Required Arguments

`libref` The library name.

`dataset` The quoted name of the data set from which to retrieve variable names.

Optional Argument

`Password(`*`password`*`)` The quoted *`password`* for the connection.

`sasconn<<Get Var Names(`*`libref`*`, `*`dataset`*`, <`*`named arguments`*`>)`

Description

Retrieves the variable names contained in the specified data set on this SAS server connection.

Returns

A list of quoted strings.

Arguments

See "SAS Get Var Names(`string`, <"dataset">, <password("password")>)".

`sasconn<<Get Version(<"Long">)`

Description

Returns the SAS version as a quoted string such as "9.3" or "9.4".

Returns

A quoted string that contains the SAS version.

Optional Argument

`Long` A quoted keyword that specifies to return the long SAS version, which corresponds to the SYSVLONG SAS macro (for example, "9.02.02M0P01152009").

sasconn<<Get Work Folder()

Description

Returns the full path of the folder corresponding to the WORK library for this server.

Returns

A quoted string that specifies the work folder path.

sasconn<<Import Data(*library*, *dataset*, *<named arguments>*)

Description

Imports a SAS data set from this SAS server connection into a JMP table.

Returns

A JMP Data Table object.

Arguments

See "SAS Import Data(string, <"dataset">, <named arguments>)".

sasconn<<Import Query(*sqlquery*, *<named arguments>*)

Description

Executes the requested SQL query on this SAS server connection, importing the results into a JMP data table.

Returns

A JMP data table object.

Arguments

See "SAS Import Query("sqlquery", <named arguments>)".

sasconn<<Is Connected()

Description

Determines whether this SAS Server object is currently connected to SAS.

Returns

1 if *sasconn* is connect, 0 otherwise.

sasconn<<Is Product Available(*product name*)

Description

Determines whether the quoted SAS product is both licensed and installed in the session represented by the SAS connection. The SAS DATA Step functions SYSPROD and MODEXIST are used to determine the licensed and installed status of the product.

Returns

1 if the specified product is licensed, 0 if the product is not licensed, or -1 if the specified product is not recognized by SAS. This function throws an exception if the requested product is not one for which JMP knows how to check the installed status.

Required Argument

`product name` The quoted SAS product for which licensing should be checked. The product name can be specified with or without the "SAS/" prefix.

`sasconn<<Is Product Licensed(`*`product name`*`)`

Description

Determines whether the quoted SAS product is licensed in the session represented by the SAS connection. The SAS DATA Step function SYSPROD is used to determine the licensing status of the product.

Returns

1 if the specified product is licensed, 0 if the product is not licensed, or -1 if the specified product is not recognized by SAS.

Required Argument

`product name` The quoted SAS product for which licensing should be checked. The product name can be specified with or without the "SAS/" prefix.

`sasconn<<Kill Session(<`*`n`*`>)`

Description

If no argument is provided, the SAS connection is immediately terminated.

Returns

Void.

Arguments

`n` An optional number. The system waits n seconds for a normal shut-down before immediately terminating the SAS connection.

`sasconn<<Load Text File(`*`path`*`, <named arguments>)`

Description

Downloads the file specified in the quoted *`path`* from the active SAS server connection and retrieve its contents as a quoted string.

Returns

A quoted string.

Arguments

See "SAS Load Text File("path")".

sasconn<<Open Log Window()

Description

Opens (or brings to the front) the SAS Log window for this server.

Returns

Void.

sasconn<<Open Output Window()

Description

Opens (or brings to the front) the SAS Output window for this server.

Returns

Void.

sasconn<<Open SAS Results()

Description

Opens the results from the previous SAS Submit. Intended to be used with asynchronous SAS submits or the use of the OnSubmitComplete option to SAS Submit to give the JSL author a way to conditionally open the results of a submit.

Returns

Void.

sasconn<<Open Submit Results()

Description

Opens all the results from the last SAS Submit command.

Returns

Void.

sasconn<<Send File(*source*, *dest*)

Description

Uploads a file to this SAS server connection.

Returns

Void.

Arguments

See “SAS Send File("source", "dest", "encoding")”.

`sasconn<<Submit(sas code, <named arguments>)`

Description

Submits quoted SAS code to this SAS server connection.

Returns

Void.

Arguments

See "SAS Submit("sasCode", <named arguments>)".

`sasconn<<Submit File(filename, <named arguments>)`

Description

Submits a SAS code file to this SAS server connection.

Returns

Void.

Arguments

See "SAS Submit File("filename", <named arguments>)".

Stored Processes

`stp<<Begin Run(<named arguments>)`

Description

Starts this stored process executing in the background. This message is paired with `End Run`, which should also be called at some point after `Begin Run` to wait for the stored process to complete.

Returns

`-1` = execution failed.

`1` = not started.

`2` = running.

`3` = canceled.

`10` = completed successfully.

`11` = completed with errors.

Optional Named Arguments

Same as `Run`, except `AutoOpenResults` and `NoAlerts` are not supported. They are available on `EndRun`.

`AutoResume(<filename>)` If specified with no argument, it specifies that the stored process results should be auto-opened when the stored process completes. If a quoted

filename is specified, filename is opened rather than all results of the stored process being auto-opened.

`AutoResumeScript(`*`script`*`)` Specifies that after stored process execution completes, the quoted script should be evaluated. If the script is a function taking at least one argument, the function is evaluated with the scriptable stored process object passed as the first (and only) argument. `AutoResume` and `AutoResumeScript` are mutually exclusive.

`stp<<Delete Results(`*`<named arguments>`*`)`

Description

Deletes all results from the execution of this stored process.

Returns

`1` if deletion is successful, `0` otherwise (error message to JMP log).

Optional Named Arguments

`NoAlerts(`*`Boolean`*`)` If `True`, the user is not prompted for confirmation before the attempt is made to delete results.

`DeleteDirectory(`*`Boolean`*`)` If true, deletes the directory containing the stored process results along with the result files themselves. The default value is true.

`stp<<Edit Param Values()`

Description

Opens the stored process window for interactively setting parameter values.

Returns

1 if the user clicks OK to dismiss the window, 0 if the user clicks Cancel.

`stp<<End Run(`*`<named arguments>`*`)`

Description

Waits a specified amount of time (or forever) for a stored process started with `Begin Run` to complete. If the stored process is complete, retrieves the results, and opens them.

Returns

`-1` = execution failed.

`1` = not started.

`2` = running.

`3` = canceled.

`10` = completed successfully.

`11` = completed with errors.

Optional Named Arguments

AutoOpenResults(*Boolean*) Optional, Boolean. If True, results are automatically opened if the stored process completes in the time specified by *MaxWait*. If False, results are not automatically opened, and can be manually opened via the object returned by the Get Results message. Default is True.

MaxWait(*milliseconds*) An integer that specifies the maximum amount of time in milliseconds to wait for the stored process to complete. If MaxWait is not specified, End Run waits forever for the stored process to complete.

NoAlerts(*Boolean*) If True, error messages are sent to the JMP log rather than message boxes. The default value is False.

---

stp<<Get Metadata Id()

Description

Returns the metadata ID of the stored process.

Returns

A quoted string.

---

stp<<Get Metadata Path()

Description

Returns the full metadata path of the stored process.

Returns

A quoted string.

Stp<<Get Name()

Description

Returns the name of the stored process.

Returns

A quoted string.

---

stp<<Get Param Enum Labels(*name*)

Description

Gets the enumeration labels specified by the quoted *name* for a parameter.

Returns

A list of quoted strings.

Arguments

name Specifies the quoted name of the parameter whose enumeration labels to retrieve.

`stp<<Get Param Enum Values(`*name*`)`

Description

Gets the possible enumerated values for a parameter.

Returns

A list of quoted strings.

Arguments

`name` Specifies the quoted name of the parameter whose possible enumerated values to retrieve.

`stp<<Get Param Names(<`*named arguments*`>)`

Description

Gets a list of parameter names for this stored process of specific types.

Returns

A list of quoted strings.

Optional Named Arguments

`Visible(`*Boolean*`)` If true, gets only visible parameters. If `False`, gets only non-visible parameters. If not specified, gets both visible and non-visible parameters.

`Modifiable(`*Boolean*`)` If true, gets only modifiable parameters. If `False`, gets only non-modifiable parameters. If not specified, gets both modifiable and non-modifiable parameters.

`Required(`*Boolean*`)` If true, gets only required parameters. If `False`, gets only non-required parameters. If not specified, gets both required and non-required parameters.

`Expert(`*Boolean*`)` If true, gets only expert parameters. If `False`, gets only non-expert parameters. If not specified, gets both expert and non-expert parameters.

`stp<<Get Param Value(`*name*`)`

Description

Gets the current value of the specified parameter.

Returns

A quoted string.

Required Argument

name Specifies the name of the parameter whose value to retrieve.

`stp<<Get Results()`

Description

Gets the results generated by the execution of this stored process as a scriptable object.

Returns

A SAS Results scriptable object.

`stp<<Get Status()`

Description

Gets the execution status of the stored process.

Returns

-1 = execution failed.

1 = not started.

2 = running.

3 = canceled.

10 = completed successfully.

11 = completed with errors.

`stp<<Get Status Message()`

Description

Gets the message associated with the failure of the stored process, if any.

Returns

A quoted string.

`stp<<Reset Param Values()`

Description

Resets all parameter values to their metadata-defined default values.

Returns

Void.

`stp<<Run(<named arguments>)`

Description

Executes this stored process object in the foreground.

Returns

-1 = execution failed.

1 = not started.

2 = running.

3 = canceled.

10 = completed successfully.

11 = completed with errors.

Optional Named Arguments

AutoOpenResults(*Boolean*) If True, results are automatically opened when the stored process completes. If False, results are not auto-opened, and can be manually opened via the object returned by the GetResults message. The default value is True.

UserName(*username*) Specifies the quoted *user name* under which to run the stored process.

Password(*password*) Specifies the quoted *password* for *UserName*.

AuthDomain(*authDomain*) Specifies the quoted authentication domain of the credentials (*username*, *password*) given.

ODSDest(*dest*) Specifies the quoted ODS destination ("HTML", "PDF", "tagsets.SASReport12") for any ODS-generated results from the stored process. This requires the stored process SAS code to call %STPBEGIN. The default value is "HTML".

GraphicsDevice(*device*) Specifies the quoted SAS graphics device to use when generating graphics in ODS results. This requires the stored process SAS code to call %STPBEGIN. The default value is "GIF".

ODSStyle(*style name*) Specifies the quoted ODS style to apply to the results. This requires the stored process SAS code to call %STPBEGIN. There is no default value.

ODSStyleSheet(*path*) Specifies the quoted *path* to a CSS file on the client machine that is to be applied to generated ODS results. This requires the stored process SAS code to call %STPBEGIN. There is no default value.

NoAlerts(*Boolean*) If True, error messages are sent to the JMP log rather than message boxes. The default value is False.

---

stp<<Set Param Value(*name*, *value*)

Description

Sets the value of the specified stored process parameter to the specified value.

Returns

1 if successful, 0 otherwise (value can violate the parameter's constraints).

Arguments

name Specifies the quoted name of the parameter whose value to set.

value Specifies the quoted string that you want to set the parameter to.

---

stp<<Set Results Directory(*path*)

Description

Sets the quoted *path* on the client machine where stored process results are placed.

Returns

A quoted string.

Arguments

`path` Specifies the full quoted path of the directory where results of the stored process execution should be placed. The directory must exist or be creatable. If the results directory is not set, a temporary location appropriate for the operating system will be used, and that directory can be retrieved from the stored process Results scriptable object after the stored process executes.

SAS Results

`results<<Delete All Result Files()`

Description

Deletes all files created by the SAS Submit or Stored Process execution. Note that any result files that are still in use are not deleted.

Returns

1 if the deletion was successful; 0 if some of the files could not be deleted.

`results<<Get Directory()`

Description

Gets the directory where the results generated by the stored process or SAS submit are located.

Returns

A quoted string.

`results<<Get Log()`

Description

Get the SAS Log from the execution of the stored process or SAS submit.

Returns

A quoted string.

`results<<Get Main Result File Name(<Fullpath(Boolean)>)`

Description

Gets the full path of the main result file generated by the stored process or SAS submit.

Returns

A quoted string.

Optional Named Argument

`Fullpath(Boolean)` If true, the main result filename is returned as a full path. The default value is false.

`results<<Get Output()`

Description

Gets the SAS Listing output from the execution of the stored process or SAS submit.

Returns

A quoted string.

`results<<Get Output Datasets()`

Description

Get a list of output data set generated by the SAS Submit that created this SAS Results object.

Returns

A list of data set names in the form "libname.membername".

`results<<Get Result File Info(<Mimetype(`*`mime-type`*`)>, <Fullpath(`*`Boolean`*`)>)`

Description

Get information about result files that were generated by the execution of the stored process or SAS submit.

Returns

List of two lists of quoted strings. The first list is filenames, and the second list is the MIME-type of the corresponding file from the first list.

Optional Arguments

`Mimetype(`*`mime_type`*`)` Restricts the set of files for which information is returned to only those files with the specified quoted MIME-type. If not specified, information about all generated files is returned.

`Fullpath(Boolean)` If true, the filename returned for each result file is returned as a full path; if false, only the name of the file is returned. The default value is false.

`results<<Make JMP Report()`

Description

Parses the ODS XML results and creates a JMP report.

Returns

The display box for the report.

results<<Open All Results()

Description

Opens all results generated by the execution of the stored process or SAS submit.

Returns

Void.

results<<Open Result File(*filename*, <Run Script(*Boolean*)>)

Description

Attempts to open the result file with the given name.

Returns

JMP Data Table if one was opened.

Required Argument

filename Specifies the quoted name of the file from the generated results to open. filename should just be the name of the file, not the full path. If *filename* is a filename with no extension, both JMP data tables and JSL scripts in the results are searched for a match, and if both exist, both are opened.

Optional Argument

Run Script(*Boolean*) If true, and if *filename* is a JSL script, the script is executed. If false, *filename* is just opened, even if it is a JSL script.

results<<Run Script(*filename*)

Description

Looks for the JSL file in the results with the given *filename* and runs it if it finds it.

Returns

Void.

Argument

"filename" Specifies the quoted name of the JSL file from the generated results to open. The *filename* argument should just be the name of the file, not the full path, and it does not need to include the .jsl extension.

Schedule

See also "Schedule(n, script)".

sch<<Clear Schedule()

Cancels all scheduled events.

sch<<Close()

Closes the scheduler.

sch<<Restart()

Restarts the scheduler after it was stopped from running all scheduled events.

sch<<Show Schedule()

Shows a list of all scheduled events.

sch<<Stop()

Stops the scheduler from running all scheduled events.

Segments

Pie Seg(<*style*>, {*x*, *y*}, *radius*, [*values*])

Description

Creates a pie seg at the specified origin, with the specified radius, based on given values.

Required Arguments

{*x*, *y*} Specifies the x and y coordinates at which the pie seg is displayed.

radius Specifies the radius.

values Specifies the values specified in matrix format.

Optional Argument

style A quoted string that specifies the style: "Pie" (traditional pie chart with each slice sized by the Summary Statistic), "Ring" (each variable or level of a stratifying variable is represented by a concentric ring), or "Coxcomb" (the central angles for all slices are equal).

Sockets

skt<<Accept(<*callback*, *timeout*>)

Description

Tells the server socket to accept a connection and return a new connected socket.

Returns

A list of up to four items. The first is a quoted string that echoes the command (`"accept"`). The second is a quoted string, either `"ok"` or an error. The third is a quoted string that specifies the name of the machine that just connected. The fourth is a reference to the socket that you can send more messages.

Optional Arguments

`callback` Specifies the name of a function to receive the data.

`timeout` If you use a *`callback`*, *`timeout`* specifies how long the function should wait for an answer. For a server socket, 0 is an acceptable value because a server should not shut down because no one has connected to it recently.

`skt<<bind(`*`localhost`*`, `*`port`*`)`

Description

Associates a port on the local machine with the socket.

Returns

A list of two quoted strings. The first string is the command name (“bind”) and the second is “ok” if successful or an error.

Required Arguments

`localhost` Specifies the quoted local machine. You cannot bind to another machine.

`port` Specifies the port that should be used.

`skt<<Close()`

Description

Closes a socket.

Returns

A list of two quoted strings. The first string is the command name (“close”) and the second is “ok” if successful.

`skt<<Connect(`*`socketname`*`, `*`port`*`)`

Description

Connects to a listening socket.

Returns

A list of two quoted strings. The first string is the command name (“connect”) and the second is “ok” for a successful connection or an error sent back by the other socket.

Arguments

`socketname` Specifies the name of the other socket. If you are connecting to a web server, this is the web address (the name is preferred to the IP address).

port Specifies the port of the other socket to connect through.

`skt<<GetPeerName()`

Description

Retrieves the address and port of the socket at the other end of the connection.

Returns

A list of four quoted strings. The first echoes the command (`"getpeername"`). The second is either "ok" or an error. The third and fourth are the address and the port.

`skt<<Get Sock Name()`

Description

Retrieves the address and port of the socket at this end of the connection.

Returns

A list of four quoted strings. The first echoes the command (`"getsockname"`). The second is either "ok" or an error. The third and fourth are the address and the port.

`skt<<ioctl(FIONBIO, Boolean)`

Description

Controls the socket's blocking behavior.

Returns

A list of two quoted strings. The first string is the command name ("ioctl") and the second is "ok" if successful or an error.

Arguments

`FIONBIO, 1 FIONBIO` means Non-Blocking I/O. If true, turns on the behavior and the argument.

`skt<<Listen()`

Description

Tells the server socket to listen for connections.

Returns

A list of two quoted strings. The first echoes the command (`"listen"`) and the second is `"ok"` or an error message.

`skt<<recv(n, <callback, timeout>)`

`skt<<recvfrom(n, <callback, timeout>)`

Description

Receives either a stream message (`recv`) or a datagram message (`recvfrom`) from the other socket. If the two optional arguments are used, the data is not received immediately. Instead, the data is received when the function *callback* is called.

Returns

A list of three quoted strings. The first string is the command name ("recv" or "recvto"). The second is "ok" if successful or an error message if not. The third string is the data that was received. If a callback function is used, a fourth element is the socket that was used in the original `recv` or `recvfrom` message.

Required Argument

n Specifies the number of bytes to receive from the other socket.

Optional Arguments

callback Specifies the name of a function to receive the data.

timeout If you use a *callback*, *timeout* specifies how long the function should wait for an answer.

`skt<<Send(stream)`

`skt<<SendTo(dgram)`

Description

Sends the data in the argument to the other socket. `Send` sends a stream and `sendto` sends a datagram.

Returns

A list of three quoted strings. The first string is the command name ("send" or "sendto"). The second is "ok" if successful or an error message if not. The third string is any portion of the stream that could not be sent, or empty if all the data was sent correctly.

Arguments

stream Specifies the command to send to the other socket.

dgram Specifies the command to send to the other socket.

Notes

Either argument might need to contain binary data. JMP represents non-printable ASCII characters with a tilde (~) followed by the hexadecimal number. For example,

```
skt<<send(("GET / HTTP/1.0~0d~0a~0d~0a");
```

sends a "get request" to an HTTP server.

SQL

obj<<Custom SQL(*sql*)

Description

Changes the query to a custom SQL query and sets the SQL.

Required Argument

`sql` The quoted SQL query.

obj<<Generate SQL

Returns the SQL that the query generates when you run it.

obj<<Modify

Opens the query in Query Builder.

obj<<PostQueryScript(*script as text*)

Sets a JSL script that runs after the query finishes executing. *script as text* is quoted JSL code.

obj<<Query Name(<*new name*>)

Gets (without the *new name* argument) or sets (with the *new name* argument) the name of the query. The name of the query is used as the name of the data table that results from running the query.

obj<<Run(<"Private"|"Invisible">, <Update Table(*table*)>, <OnRunComplete(*script*)>, <OnRunCanceled(*script*)>, <OnError(*script*)>)

Description

Runs the SQL query in the background or foreground depending on the Query Builder preference "Run queries in the background when possible".

Returns

Null (if the query runs in the background) or a data table (if the query runs in the foreground).

Optional Named Arguments

`"Private"` A quoted keyword that opens the data table that the query produces without displaying it in a data table window. `"Private"` is available only if `OnRunComplete` is included in the script.

"Invisible" A quoted keyword that hides the data table that the query produces. Use this argument to keep the query result hidden but use it in a subsequent query. The data table is displayed in the Home Window's Window List and the Window > Unhide list.

Update Table Updates the specified data table. Runs the query in the foreground.

OnRunComplete Specifies a script to run after the query is complete. To get the resulting data table, include OnRunComplete. The OnRunComplete script needs to be defined in the global namespace, as indicated by the double colons in this example:

```
Names Default To Here( 1 );
::onComplete = Function( {dt},
  {default local},
  Write(
     "\!NQuery is complete!  Result name: \!"",
     dt << Get Name,
     "\!", Number of rows: ",
     N Rows( dt )
  )
);

query = Include( "rentals_fam_romcom.jmpquery" );
query << Run Background( On Run Complete( ::onComplete ) );
```

OnRunCanceled Specifies a script to run after the user cancels the query.

OnError Specifies a script to run if an error occurs.

Notes

If you want the data table that results from the background query, use the OnRunComplete optional argument. You can include a script that runs when the query completes and then assigns a data table reference to the resulting data table. Or you might pass the name of a function that accepts a data table as its first argument. That function is called when the query completes.

Examples

The following example opens a query that you previously saved from Query Builder. The query opens privately, that is, without opening Query Builder. The query runs, and the resulting data table opens.

```
query = Open( "c:/My Data/Movies.jmpquery", "Private");
dt = query << Run();
```

You can include a .jmpquery file in a script and run the query in the background using the <<Run Background message.

```
query = Include( "C:/Queries/movies.jmpquery");
query <<Run Background();
```

The following example queries the database, opens the resulting data table, and prints the number of data table rows to the log.

```
confirmation = Function( {dtResult},
  Write( "\!NNumber of rows in query result: ", N Rows( dtResult ) )
);
query = New SQL Query(
  Connection(
     "ODBC:DSN=SQL
  Databases;APP=MYAPP;TrustedConnection=yes;WSID=D79255;DATABASE=SQB;"
  ),
  QueryName( "movies_to_update" ),
  Select( Column( "YearMade", "t1" ), Column( "Rating", "t1" ) ),
  From( Table( "g6_Movies", Schema( "SQB" ), Alias( "t1" ) ) ),

);
query << Run( OnRunComplete( confirmation ) );
```

`Run Background(<OnRunComplete(`*script*`), <"Private"|"Invisible">>, <OnRunCanceled(`*script*`)>, <OnError(`*script*`)>)`

Description

Runs the SQL query in the background. The running query is not displayed.

Returns

Null (or the data table object, if `OnRunComplete` is included).

Optional Named Arguments

`OnRunComplete(`*script*`)` Specifies a script to run after the query is complete. To get the resulting data table, include `OnRunComplete`. The `OnRunComplete` script needs to be defined in the global namespace, as indicated by the double colons in this example:

```
Names Default To Here( 1 );
::onComplete = Function( {dt},
  {default local},
  Write(
     "\!NQuery is complete!  Result name: \!"",
     dt << Get Name,
     "\!", Number of rows: ",
     N Rows( dt )
  )
);

query = Include( "rentals_fam_romcom.jmpquery" );
query << Run Background( On Run Complete( ::onComplete ) );
```

`"Private"` Does not open the resulting data table. Specify only with `OnRunComplete`. If you include `private` in a background query, JMP opens the data table as invisible instead.

`"Invisible"` Hides the data table. Use this argument to keep the query result hidden but use it in a subsequent query. The data table is displayed in the Home Window's Window List and the Window > Unhide list.

`OnRunCanceled` Specifies a script to run after the user cancels the query.

`OnError` Specifies a script to run if an error occurs.

Notes

All queries except for SAS queries run in the background based on the Query Builder preference "Run the queries in the background when possible", which is selected by default. For SAS queries, `Run Background()` is ignored.

You can include a .jmpquery file in a script and run the query in the background using the `Run Background` message.

```
query = Include( "C:/Queries/movies.jmpquery");
query <<Run Background();
```

`Run Foreground(<OnRunComplete(`*`script`*`), <"Private"|"Invisible">>, <OnRunCanceled(`*`script`*`)>, <OnError(`*`script`*`)>)`

Description

Runs the SQL query in the foreground.

Returns

A data table that opens when the query is finished.

See Also

"Run Background(<OnRunComplete(`script`), <"Private"|"Invisible">>, <OnRunCanceled(`script`)>, <OnError(`script`)>)"

`obj<<Save`

Saves the query to its associated file. The save fails if the query does not yet have an associated file.

`obj<<Save As(`*`path`*`, <Replace Existing(`*`Boolean`*`))`

Saves the query to the specified file. If the file already exists, the save fails unless `Replace Existing` is true.

Other Objects

Zip Archives

`list = za<<Dir`

Returns a list of member names

`data = za<<Read(`*`member name`*`, <Format("blob")>)`

Returns a quoted string that contains the entire quoted *`member name`*. A zip file consists of filenames, also called "member names".

Notes

For remote files, JMP copies the URL data to the local disk. When the zip archive is no longer accessible, the local data file is deleted.

`actualname = za<<Write(`*`member name`*`, `*`member data`*`, <"replace">)`

Writes a quoted string or quoted blob to a zip archive member file. If the quoted *`member name`* isn't in the current zip file, the returned `actualname` is the same as *`member name`*. This member name will be changed to prevent overwriting an existing member; the name actually used is returned. The quoted *`member data`* argument is the data to write into the zip file's member of that name. *`replace`* creates the file with a temporary name, deletes the old file, and renames the temporary file to the existing name.

Journals

`jnl<<Save HTML(<`*`path`*`>, <`*`format`*`>)`

Saves the journal as HTML.

Optional Arguments

`"path"` Specifies the quoted path for the saved HTML file (for example, `"c:/myFile.html"`).

`format` Specifies the quoted graphic file format. JPG, PNG, and TIFF formats are supported. The graphics are saved in a subdirectory named gfx.

`jnl<<Save RTF(<`*`path`*`>, <format>)`

Saves the journal as an RTF file.

Optional Arguments

`"path"` Specifies the quoted path for the saved RTF file (for example, `"c:/myFile.rtf"`).

"format" Specifies the quoted file format for the embedded graphics. "JPG", "PNG", and "EMF" formats are supported on Windows. All journals are saved as PDF files by default on macOS.

Notes

If no *path* or *format* are provided, you are prompted to name the file and specify the format on Windows. On macOS, you are prompted to name the file. The file is saved as a PDF file by default.

---

jnl<<Save PDF(<*path*>, <Show Page Setup(*Boolean*)>, <Portrait(*Boolean*)>)

Saves the journal as a PDF file.

Optional Arguments

"path" The quoted *path* for the saved PDF file (for example, "c:/myFile.pdf").

Show Page Setup If set to true, opens the Page Setup window to let the user change the margin, magnification level, and other page layout options.

Portrait Determines whether the page orientation is portrait or landscape. Overrides the user's selection in the Show Page Setup window.

Appendix A

SQL Functions Available for JMP Queries

The `Query()` JSL function performs a SQL query on selected tables and exports the data to a data table. The following example first assigns the `t1` alias to Big Class.jmp. `name`, `age` greater than 13, and `height` are then selected from the `t1` table.

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Query( Table( dt, "t1" ),
       "SELECT t1.name, t1.age, t1.height FROM t1
       WHERE t1.age > 13" );
```

You can use SQL functions in a query. For example, `SELECT CURRENT_TIMESTAMP` returns the current UTC/GMT time stamp as a SQLite time string:

```
Query( Scalar, "SELECT CURRENT_TIMESTAMP;" );
```

This appendix lists the numeric, date-time, string, system SQL, and aggregate functions that you can use in SQL queries. “Yes” in the SQLite column indicates native SQLite functions. See the Online SQLite documentation at https://www.sqlite.org/lang.html.

Contents

Numeric SQL Functions 531
Date-Time SQL Functions 532
String SQL Functions 535
System SQL Functions 536
Aggregate SQL Functions 536

Numeric SQL Functions

The numeric SQL functions are described here.

Numeric Function	Native SQLite	Description
ABS(*number*)		Returns the absolute value of the specified number.
ACOS(*cosine*)		Returns the angle in radians for the specified cosine.
ASIN(*sin*)		Returns the angle in radians for the specified sine.
ATAN(*tangent*)		Returns the angle in radians for the specified tangent.
ATAN2(*x*, *y*)		Two-argument arctangent function.
CEILING(*number*) CEIL(*number*)		Returns the smallest integer larger than the specified number.
COS(*radians*)		Returns the cosine of the specified angle in radians.
COT(*radians*)		Returns the cotangent of the specified angle in radians.
DEGREES(*radians*)		Converts an angle in radians to an angle in degrees.
EXP(*number*)		Returns the constant *e* raised to the specified power.
FLOOR(*number*)		Returns the largest integer smaller than the specified number.
LN(*number*) LOG(*number*)		Returns the natural logarithm of the specified number.
LOG10(*number*)		Returns the common logarithm of the specified number.
MAX(*n1*, *n2*, ...)	Yes	Returns the largest of the specified numbers. A minimum of two numbers must be specified.
MIN(*n1*, *n2*, ...)	Yes	Returns the smallest of the specified numbers. A minimum of two numbers must be specified.
MOD(*dividend*, *divisor*)		Returns the remainder when *dividend* is divided by *divisor*. Floating-point values are truncated to integers before the modulus operation is performed.
PI()		Returns the value of the constant pi (π).
POWER(*number*, *power*) POW(*number*, *power*)		Raises *number* to the specified *power*.

Numeric Function	Native SQLite	Description
RADIANS(*degrees*)		Converts an angle in degrees to an angle in radians.
RANDOM() RANDOM(*max*) RANDOM(*min*, *max*)		Returns a random number. RANDOM() returns a number between 0 and 1. RANDOM(max) returns a number between 0 and *max*. RANDOM(min, max) returns a number between *min* and *max*. This function is equivalent to the Random Uniform() JSL function, and its seed can be controlled using the Random Reset() JSL function. RANDOM can be shortened to RAND.
RANDOMBLOB(*length*)	Yes	Returns an N-byte blob that contains pseudo-random bytes. See the SQLite Online documentation at https://www.sqlite.org/lang.html.
ROUND(*number*, *<precision>*)		Rounds *number* to the number of decimal places given by *precision*. The default value of *precision* is 0, and *precision* can be negative.
SIGN(*number*)		Returns 1 if *number* is positive, -1 if *number* is negative, or 0 if *number* is zero.
SIN(*radians*)		Returns the sin of the specified angle in radians.
SQRT(*number*)		Returns the square root of *number*.
TAN(*radians*)		Returns the tangent of the specified angle in radians.
TRUNCATE(*number*, *<precision>*)		Truncates *number* at the number of decimal places given by *precision*. The default value of *precision* is 0, and *precision* can be negative. TRUNCATE() can be shortened to TRUNC().

Date-Time SQL Functions

Using date-time functions in JMP queries is complicated by the fact that the SQL engine that handles JMP queries (SQLite) uses different formats for storing dates than JMP does. SQLite stores date-times as strings. However, JMP stores date-times as the number of seconds since January 1, 1904. When you have columns in your table that contain date-times, the conversions are handled automatically. However, when you use functions that return date-times, you might need to let JMP know when a conversion is required.

Consider the CURRENT_TIMESTAMP function. CURRENT_TIMESTAMP is a built-in SQLite function that returns the current UTC/GMT time stamp as a SQLite time string:

```
Query( Scalar, "SELECT CURRENT_TIMESTAMP;" );
```

returns:

```
"2016-02-16 15:44:42"
```

The string could perhaps be parsed as a date to return it as a JMP date. To prevent the need to do so, wrap the CURRENT_TIMESTAMP function in the JMPDATE() function:

```
Query( Scalar, "SELECT JMPDATE( CURRENT_TIMESTAMP );" );
```

returns:

```
3538482531
```

The string is an unformatted JMP date. However, if you pass a SQLite time string to another SQL date-time function, you do not need to use JMPDate(); the value will be converted to a JMP date automatically. Here is an example:

```
Query( Scalar, "SELECT EXTRACT('YEAR', CURRENT_TIMESTAMP);" );
```

Using native SQLite date-time functions (date(), time(), datetime(), julianday(), strftime()) in JMP queries is not recommended because JMP date-time values are not compatible with those functions.

Date-Time Function	Naive SQLite	Description
CURRENT_DATE	Yes	Returns the current date (UTC/GMT) as a SQLite time string.
CURRENT_TIME	Yes	Returns the current time (UTC/GMT) as a SQLite time string.
CURRENT_TIMESTAMP	Yes	Returns the current date and time (UTC/GMT) as a SQLite time string.
DATEDIFF(*date1*, *date2*, *interval*, <*alignment = "Start"*>)		Computes the difference between two dates in units specified by *interval*, based on *alignment*. This function works the same as the Date Difference() JSL function. Valid values for *interval* are: "Year", "Quarter", "Month", "Week", "Day", "Hour", "Minute" and "Second". Valid values for *alignment* are "Start", "Actual" and "Fractional". If *alignment* is not specified, "Start" is used.

Date-Time Function	Naive SQLite	Description	
EXTRACT(*datepart*, *datetime*, *<use_locale = 1>*)		Extracts a specific part of a date or date-time value. *Datetime* is a JMP date-time value or a SQLite time string. *Use_locale* is optional and applies only to date name parts such as "MonthName" and "DayName" and determines whether values from the current language or English are returned. The following values of *datepart* are supported:	
		"Year"	Returns the year as a number.
		"Month"	Returns the numeric month (1-12).
		"MonthName"	Returns the full name of the month in the current language (*use_locale = 1*) or English (*use_locale = 0*).
		"Mon", "MMM"	Returns the abbreviated name of the month.
		"Day"	Returns the day of the month (1-31).
		"DayName"	Returns the name of the day of the week.
		"DayOfWeek"	Returns the numeric day of the week (1-7).
		"DayOfYear"	Returns the numeric day of the year (1-366).
		"Quarter"	Returns the numeric quarter (1-4).
		"Hour"	Returns the hour (0-23).
		"Minute"	Returns the minute (0-59).
		"Second"	Returns the seconds, including any fractional part.
		"Date"	Returns just the date portion of a date-time value as a JMP date-time value.
		"Time"	Returns just the time portion of a date-time value as a JMP date-time value.
JMPDATE(*SQLite time string*)		Converts a SQLite time string to the equivalent JMP date-time value.	
NOW()		A synonym for TODAY().	
TODAY()		Returns the JMP date-time value of the current moment in *local* time, which matches the JMP Today() function.	

String SQL Functions

The string SQL functions are described here
.

Function	Native SQLite	Description
HEX(*binary*)	Yes	SQLite built-in function that converts a BLOB to a string of hexadecimal characters. Useful when paired with the RANDOMBLOB() function.
JLEFT(*string*, *len*, *<pad>*)		Like the JSL Left() function. Returns *len* characters from the beginning of *string*. If *pad* is specified and fewer than *len* characters are present in *string*, the result is padded with *pad* out to length *len*.
JRIGHT(*string*, *len*, *<pad>*)		Like the JSL Right() function. Returns *len* characters from the end of *string*. If *pad* is specified and fewer than *len* characters are present in *string*, the result is padded with *pad* at the front out to length *len*.
LENGTH(*string*)	Yes	SQLite equivalent of the ANSI standard CHAR_LENGTH() function. Returns the length of its string argument in characters.
LOCATE(*string1*, *string2*) POSITION(*string1*, *string2*)		Returns the (1-based) starting position of *string1* within *string2*, returning 0 if *string1* is not found within *string2*.
LOWER(*string*)		Returns a copy of *string* with all uppercase characters converted to lowercase.
LTRIM(*string*, *<trimchars>*)	Yes	Trims any characters contained in *trimchars* from the beginning of *string* and returns the result. If *trimchars* is omitted, spaces are trimmed.
PRINTF(*format*, *<arg1, ..., argN>*)	Yes	Allows constructing strings using placeholders and arguments. See the SQLite Online documentation at https://www.sqlite.org/lang.html.
REPLACE(*string*, *find*, *replace*)	Yes	Replaces all instances of *find* in *string* with *replace* and returns the result. If *replace* is numeric, it is converted to a string.
REVERSE(*string*)		Returns a copy of *string* with the order of the characters reversed.

Function	Native SQLite	Description
RTRIM(*string*, *<trimchars>*)	Yes	Trims any characters contained in *trimchars* from the end of *string* and returns the result. If *trimchars* is omitted, spaces are trimmed.
SPACE(*length*)		Returns a string consisting of *length* space characters.
SUBSTR(*string*, *start*, *<length>*)	Yes	Returns the substring of *string* starting at *start* (1-based) that is *length* characters long. If *length* is omitted, the substring starting at *start* and continuing to the end of *string* is returned.
TRIM(*string*, *<trimchars>*)		Trims any characters contained in *trimchars* from the end of *string* and returns the result. If *trimchars* is omitted, spaces are trimmed.
UPPER(*string*)		Returns a copy of *string* with all lowercase characters converted to uppercase.

System SQL Functions

The system SQL functions are described here.

Function	SQLite	Description
COALESCE(*arg1*,..., *argN*)	Yes	Returns the first argument passed to it that is non-NULL. Returns NULL if all arguments are NULL. Requires at least two arguments.
IFNULL(*arg1*, *arg2*)	Yes	Returns *arg1* if not NULL, otherwise *arg2*. Basically, IFNULL is a two-argument version of COALESCE().
NULLIF(*arg1*, *arg2*)	Yes	Returns *arg1* if *arg1* and *arg2* are different and returns NULL if the arguments are equal. Used when you have non-NULL values in your database that you want to treat as NULL.

Aggregate SQL Functions

When passing a single argument to an aggregate function, that argument can be preceded by the keyword DISTINCT, which filters out duplicate values.

For all aggregations other than COUNT(*), NULL and missing values are ignored.

Function	SQLite	Description
AVG(*num_expr*)		Computes the average of *num_expr* for the rows in the group. *Num_expr* must be numeric.
COUNT(*expr*) COUNT(*)		Counts the number of times *expr* is not NULL in the group. COUNT(*) returns the total number of rows in the group.
GROUP_CONCAT(*expr*, *<separator = ','>*)	Yes	Concatenates all non-NULL values of *expr* and returns them as a string. Numeric values of *expr* are converted to character. If *separator* is present, it is placed between the values. The default separator is a comma. DISTINCT can be used only with GROUP_CONCAT() if *separator* is not specified.
MAX(*expr*)		Returns the maximum value of *expr* in the group. *Expr* can be character or numeric.
MIN(*expr*)		Returns the minimum value of *expr* in the group. *Expr* can be character or numeric.
STDDEV_POP(*num_expr*)		Computes the population standard deviation of *num_expr* for the group.
STDDEV_SAMP(*num_expr*)		Computes the sample standard deviation of all *num_expr* for the group.
SUM(*num_expr*)		Returns the sum of *num_expr* for the group. If no non-NULL values are found, SUM() returns NULL.
TOTAL(*num_expr*)	Yes	Same as SUM(*num_expr*), except TOTAL() returns 0.0 if no non-NULL values are found.
VAR_POP(*num_expr*)		Computes the population variance of *num_expr* for the group.
VAR_SAMP(*num_expr*)		Computes the sample variance of *num_expr* for the group.

Appendix B

References

Gander, W., and Gautschi, W. (2000). "Adaptive Quadrature–Revisited." *BIT Numerical Mathematics* 40:84–101.

Moore, W. A., and Parks, D. R. (2012). "Update for the logicle data scale including operational code implementations." *Cytometry Part A* 81A:273–277. Accessed January 26, 2021. https://onlinelibrary.wiley.com/doi/full/10.1002/cyto.a.22030.

Appendix C

Technology License Notices

- Scintilla is Copyright © 1998–2017 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

  NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
- Progressë® Telerik® UI for WPF: Copyright © 2008-2019 Progress Software Corporation. All rights reserved. Usage of the included Progress® Telerik® UI for WPF outside of JMP is not permitted.
- ZLIB Compression Library is Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Made with Natural Earth. Free vector and raster map data @ naturalearthdata.com.
- Packages is Copyright © 2009–2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  Neither the name of the WhiteBox nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- iODBC software ia Copyright © 1995–2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  – Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- This program, "bzip2", the associated library "libbzip2", and all documentation, are Copyright © 1996–2019 Julian R Seward. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

  3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@acm.org

bzip2/libbzip2 version 1.0.8 of 13 July 2019

- R software is Copyright © 1999–2012, R Foundation for Statistical Computing.
- MATLAB software is Copyright © 1984-2012, The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.
- libopc is Copyright © 2011, Florian Reuter. All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and / or other materials provided with the distribution.
  – Neither the name of Florian Reuter nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- libxml2 - Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

  Copyright © 1998–2003 Daniel Veillard. All Rights Reserved.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

- Regarding the decompression algorithm used for UNIX files:

  Copyright © 1985, 1986, 1992, 1993

  The Regents of the University of California. All rights reserved.

  THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- Snowball is Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Pako is Copyright © 2014–2017 by Vitaly Puzrin and Andrei Tuputcyn.

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- HDF5 (Hierarchical Data Format 5) Software Library and Utilities are Copyright 2006–2015 by The HDF Group. NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998-2006 by the Board of Trustees of the University of Illinois. All rights reserved. DISCLAIMER: THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.
- agl-aglfn technology is Copyright © 2002, 2010, 2015 by Adobe Systems Incorporated. All Rights Reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  – Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  – Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  – Neither the name of Adobe Systems Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- dmlc/xgboost is Copyright © 2019 SAS Institute.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libzip is Copyright © 1999–2019 Dieter Baron and Thomas Klausner.

  This file is part of libzip, a library to manipulate ZIP archives. The authors can be contacted at <libzip@nih.at>.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- OpenNLP 1.5.3, the pre-trained model (version 1.5 of en-parser-chunking.bin), and dmlc/xgboost Version .90 are licensed under the Apache License 2.0 are Copyright © January 2004 by Apache.org.

  You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
- You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- LLVM is Copyright © 2003–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- clang is Copyright © 2007–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS", WITHOUT WARRANTIES OR CONDITIONS OF

ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- lld is Copyright © 2011–2019 by the University of Illinois at Urbana-Champaign.

  Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS”, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

- libcurl is Copyright © 1996–2021, Daniel Stenberg, daniel@haxx.se, and many contributors, see the THANKS file. All rights reserved.

  Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

  THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

  Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.